Oracle® Communications Billing and Revenue Management

Pipeline Manager Reference

Release 15.1
F93239-01
April 2025

ORACLE

Oracle Communications Billing and Revenue Management Pipeline Manager Reference, Release 15.1

F93239-01

Contents

4 Pipeline Manager Data Modules

5 Pipeline Manager iRules

6 Pipeline Manager iScripts

7 Pipeline Manager iScript Functions

8 Pipeline Manager Input and Output Modules

9 Pipeline Manager Framework Modules

10 Pipeline Manager Utilities

11 Pipeline Manager Opcode Reference

12 Revenue Assurance Manager Reports

Preface

This book provides reference information for the modules and utilities in Oracle Communications Billing and Revenue Management Pipeline Manager.

Audience

This book is intended for pricing administrators and charging experts.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customer access to and use of Oracle support services will be pursuant to the terms and conditions specified in their Oracle order for the applicable services.

Diversity and Inclusion

Oracle is fully committed to diversity and inclusion. Oracle respects and values having a diverse workforce that increases thought leadership and innovation. As part of our initiative to build a more inclusive culture that positively impacts our employees, customers, and partners, we are working to remove insensitive terms from our products and documentation. We are also mindful of the necessity to maintain compatibility with our customers' existing technologies and the need to ensure continuity of service as Oracle's offerings and industry standards evolve. Because of these technical constraints, our effort to remove insensitive terms is ongoing and will take time and external cooperation.

1

BRM Rating EDR Container Description

Learn about the rating EDR container fields that are used by Oracle Communications Billing and Revenue Management (BRM) Pipeline Manager.

Topics in this document:

- Discount Sub-Balance Packet
- Associated SMS Extension Record
- Associated MMS Extension Record
- SGSN Information
- Profile Event Ordering
- Associated Roaming Extension Record
- Associated RAP Extension
- Total Advised Charge Value List
- Field Usage

Naming Conventions

The definitions listed in Table 1-1 are used in this document to describe the record format.

Table 1-1 Record Format Definitions

Symbol	Meaning
X	Alphanumeric, left-justified, filled with trailing spaces to the right.
Z	Numeric, left-justified, filled with trailing spaces to the right.
H	Hexadecimal value (0-9, A-F), right-justified, filled with leading zeros to the left.
9	Numeric, right-justified, filled with leading zeros to the left.
(m)	Specifies the length in characters: mandatory.
[n]	Specifies the decimal precision. Optional.

Oracle CDR Format

The Oracle CDR format is the standard file structure used by Pipeline Manager to process CDRs during the input and output processes.

The Oracle CDR format has the following characteristics:

- Each record is separated by a newline character (**\n**).
- Each record contains data for one service only.
- Each record contains a fixed number of fields.
- Each field is tab delimited (**\t**).
- Each field is in a specified position within a record. For example, the first field in the Header Record is the Record Type, the second is the Sender, and so forth.

> **Note:**
>
> If a field does not have a value, the field is left blank. The result is a tab followed by another tab.

- Each field has a specified data type and format. For example, the **A number** must be a string that is 10 characters long**.**

To process CDRs, Pipeline Manager converts the CDRs to the internal EDR format by using the stream format, grammar, and mapping description files.

The stream format file describes the structure of the Oracle CDR format.

The following example shows the section of the stream format description file that describes the format of the Header Record. The Header Record is identified by the record type **010**. The fields are separated by a tab (**\t**), and the record is terminated by a newline character (**\n**). It specifies the list the fields in the record. The first field is the record type (RECORD_TYPE), the second is the record number (RECORD_NUMBER), and so forth. RECORD_TYPE uses the AscString() data type, RECORD_NUMBER uses the AscInteger( ) data type.

```
HEADER(SEPARATED)
{
Info
{
Pattern = "010.*\n";
FieldSeparator = '\t';
RecordSeparator = '\n';
}
RECORD_TYPE                     AscString();
RECORD_NUMBER                   AscInteger();
SENDER                          AscString();
RECIPIENT                       AscString();
SEQUENCE_NUMBER                 AscInteger();
ORIGIN_SEQUENCE_NUMBER          AscInteger();
CREATION_TIMESTAMP              AscDate();
TRANSMISSION_DATE               AscDate("%Y%m%d");
TRANSFER_CUTOFF_TIMESTAMP       AscDate();
UTC_TIME_OFFSET                 AscString();
SPECIFICATION_VERSION_NUMBER    AscInteger();
RELEASE_VERSION                  AscInteger();
ORIGIN_COUNTRY_CODE              AscString();
SENDER_COUNTRY_CODE              AscString();
DATA_TYPE_INDICATOR              AscString();
IAC_LIST                         AscString();
CC_LIST                          AscString();
UTC_END_TIME_OFFSET              AscString();
}
```

The grammar files are used to verify the data formats and to normalize the data. For example, if a field is supposed to be 10 characters, Pipeline Manager uses the grammar file to perform this check. If the data is of an incorrect format, the CDR is rejected.

The mapping files are used to map CDR fields to the EDR container fields.

The following example shows a section of the InMap description file, which is used during the input process. This example shows how the fields in the Header Record of a CDR are mapped to the EDR container fields.

```
HEADER
{
STD_MAPPING
{
RECORD_TYPE                 -> HEADER.RECORD_TYPE;
RECORD_NUMBER               -> HEADER.RECORD_NUMBER;
SENDER                      -> HEADER.SENDER;
RECIPIENT                   -> HEADER.RECIPIENT;
SEQUENCE_NUMBER             -> HEADER.SEQUENCE_NUMBER;
```

```
ORIGIN_SEQUENCE_NUMBER       -> HEADER.ORIGIN_SEQUENCE_NUMBER;
CREATION_TIMESTAMP           -> HEADER.CREATION_TIMESTAMP;
TRANSMISSION_DATE            -> HEADER.TRANSMISSION_DATE;
TRANSFER_CUTOFF_TIMESTAMP    -> HEADER.TRANSFER_CUTOFF_TIMESTAMP;
UTC_TIME_OFFSET              -> HEADER.UTC_TIME_OFFSET;
SPECIFICATION_VERSION_NUMBER -> HEADER.SPECIFICATION_VERSION_NUMBER;
RELEASE_VERSION              -> HEADER.RELEASE_VERSION;
ORIGIN_COUNTRY_CODE          -> HEADER.ORIGIN_COUNTRY_CODE;
SENDER_COUNTRY_CODE          -> HEADER.SENDER_COUNTRY_CODE;
DATA_TYPE_INDICATOR          -> HEADER.DATA_TYPE_INDICATOR;
IAC_LIST                     -> HEADER.IAC_LIST;
CC_LIST                      -> HEADER.CC_LIST;
UTC_END_TIME_OFFSET          -> HEADER.UTC_END_TIME_OFFSET;
}
}
```

EDR Format Structure

The BRM EDR format consists of the following components:

1. Exactly one Header Record. Record type 010.
2. Zero or more Basic Records in no specific order:
   a. Basic Detail Record; for example, record type 020.
   b. More basic records might be defined in the future.
3. Zero or more Associated Records, related to one Basic Record, in the following order:
   a. Associated Service Extension Records; for example, record type 520.
   b. Associated CAMEL Extension Records. Record type 700.
   c. Associated BRM Balance Record. Record type 900.
   d. Associated Zone Breakdown Record; for example, record type 960.
   e. Associated Charge Breakdown Record; for example, record type 981.
   f. Associated Message Description Record. Record type 999.
4. Exactly one Trailer Record. Record type 090.

Example Structure

Table 1-2 contains an example BRM EDR structure.

Table 1-2 BRM EDR Example Structure

Record	Description
Header Record: 010	Once. Mandatory.
Basic Detail Record: 040	Once. Optional.
Associated GPRS Extension Record: 540	Once. Optional.
Associated CAMEL Extension Record: 700	Once. Optional.
Associated BRM Balance Record: 900	Once. Optional.
Supplementary Balance Impact Packet Record: 600	*n* times. Mandatory 1-*n*.
Supplementary Sub-Balance Impact Packet Record: 605	*n* times. Mandatory 1-*n*.

Table 1-2 (Cont.) BRM EDR Example Structure

Record	Description
Supplementary Sub-Balance Info Packet Record: 607	*n* times. Mandatory 1-*n*.
Associated Zone Breakdown Record: 961	*n* times. Optional.
Supplementary Zone Packet Record: 660	*n* times. Mandatory 1-*n*.
Associated Charge Breakdown Record: 981	*n* times. Optional.
Supplementary Charge Packet Record: 660	*n* times. Mandatory 1-*n*.
Associated Message Description Record: 999	*n* times. Optional.
Basic Detail Record: 070	Once. Optional.
Associated WAP Extension Record: 550	Once. Optional.
Associated BRM Balance Record: 900	Once. Optional.
Supplementary Balance Impact Packet Record: 600	*n* times. Mandatory 1-*n*.
Supplementary Sub-Balance Impact Packet Record: 605	*n* times. Mandatory 1-*n*.
Supplementary Sub-Balance Info Packet Record: 607	*n* times. Mandatory 1-*n*.
Associated Charge Breakdown Record: 981	*n* times. Optional.
Supplementary Charge Packet Record: 660	*n* times. Mandatory 1-*n*.
Basic Detail Record: 021	Once. Optional.
Associated GSM Extension Record: 520	Once. Optional.
Supplementary Service Event Record: 520	*n* times. Optional.
Basic Service Event Record: 520	*n* times. Optional.
Associated CAMEL Extension Record: 700	Once. Optional.
Associated Charge Breakdown Record: 981	*n* times. Optional.
Supplementary Charge Packet Record: 660	*n* times. Mandatory 1-*n*.
Basic Detail Record: 127	Once. Optional.
Associated Charge Breakdown Record: 981	*n* times. Optional.
Basic ...	None
Associated …	None
Supplementary …	None
Trailer Record: 090	Once. Mandatory.

Expected File Name

The BRM EDR file name uses a format of SOL42_*SenderRecipientSequence_number*.DAT. Table 1-3 describes the attributes used in the file name.

Table 1-3 EDR File Name Attributes

Item	Format	Description
Sender	X(5)	code for the sender of the file (for example, D00D1)
Recipient	X(5)	code for the recipient of the file (for example, SOL42)
Sequence_number	9(6)	sequence number of the file (000000 to 999999)

Example: "SOL42_D00D1SOL42004711.DAT"

Record Type Ranges

The Record Type Ranges listed in Table 1-4 are defined.

Table 1-4 Record Type Ranges

Range	Record type
000 - 009	Reserved for internal usage
010	Header Record
011 - 019	Reserved for Basic Address Records
020 - 089	Basic Detail Records
090	Trailer Record
091 - 099	Reserved for internal usage
100 - 299	Basic Detail Records
300 - 319	Basic Recharge Records
320 - 399	Free
400 - 499	Reserved for further Basic Record Types
500 - 599	Associated Service Extension Records
600 - 699	Supplementary Records (for former Sub-Blocks of Associated Records)
700 - 749	Associated CAMEL / IN Records
750 - 799	Reserved for further Associated Record Types
800 - 899	Free
900 - 949	Associated Balance Records
950 - 959	Reserved for further Record Types
960 - 969	Associated Zone Breakdown Records
970 - 998	Associated Charge Breakdown Records
999	Associated Message Description Record

> **Note:**
>
> Not all of the given Record Types have been defined. Undefined values are reserved for future use.

Header Record (RECType 010)

This record is always the first record within a file. Table 1-5 describes the fields in the Header Record.

Table 1-5 Header Record Fields

Name	Format	Description
RECORD_ LENGTH	Integer	Optional for backward compatibility.
RECORD_TYPE	String	Extended to be 3 bytes long, first byte denotes the market; for example, GSM, ISDN.010. Derivation: Mandatory. Set by the first processor and left unchanged.
RECORD_NUMBER	9(9)	Sequence number of the record in the file. Ensures a linear sequence order for all records; for example, as a sorting criteria. Derivation: Mandatory. Set by the first processor. Always **000000001**.
SENDER	X(10)	Unique identifier of the PLMN or physical (network) operator, which is sending the file; used to determine the network, which is the sender of the data. The full list of mobile codes in use is given in MoU TADIG PRD TD. 13: PLMN Naming Conventions. Specifies a unique NOSP_ID together with the RECIPIENT. Can also be used to determine the network operator responsible for the CDRs. Derivation: Optional, but should be defaulted if not present on the input side, for example, by own NO-Id, for example, 'DTAG'. Set by the first processor and left unchanged.
RECIPIENT	X(10)	Unique identifier of the PLMN or physical (network) operator to whom the file is being sent. See the MoU TADIG PRD TD. 13: PLMN Naming Conventions for a list of mobile codes. Specifies a unique NOSP_ID together with the SENDER. Can also be used to determine the reseller or service provider who is responsible for billing these events. Derivation: Optional, but should be defaulted; for example, by your own NO-Id, such as 'DTAG'. Set by the first processor and left unchanged.
SEQUENCE_NUMBER	9(6)	Unique reference that identifies each file sent by the VPLMN or logical sender to a particular HPLMN or logical recipient. It indicates the file number of the specific file type, starting at 1 and increments by one for each new file of that type sent. Separate sequence numbering must be used for test and chargeable data. Having reached the maximum value (999999), the number restarts at 1. Validates duplicate sequence numbers and sequence number gaps. **Note:** In the case of retransmission, this number does not increment. Range: 000001 - 999999 for test data and chargeable data. Derivation: Optional, if no sequence check is performed. Mandatory, if a sequence check is performed. Should be set by the first processor and can be changed by any following processor; for example, in case of recycling to assure a unique and linear sequence order to all following processors.

Table 1-5 (Cont.) Header Record Fields

Name	Format	Description
ORIGIN_SEQUENCE_NUMBER	9(6)	Original file sequence number as generated the first time. Identical content as SEQUENCE_NUMBER, but will never be changed. Used as a reference to the original file, if any processor has changed the file sequence number. Derivation: Mandatory, defaulted by SEQUENCE_NUMBER. Set by the first processor and left unchanged.
SEQ_CHECK_KEY	String	Derivation: Optional if no sequence check is performed. Mandatory if a sequence check is performed.
SEQ_GEN_KEY	String	Derivation: Optional if no sequence check is performed. Mandatory if a sequence check is performed.
CREATION_TIMESTAMP	YYYYMMDDHHMISS	Date and time on which the file was created. Not required by GSM MoU BA. 12, but might be useful for operational purposes. Can be used to validate that at least one file/stream has been generated every day. Optional Field Usage: It is possible to read/write dates in number of seconds since 01.01.1970 00:00:00; for example, 12345. The internal representation is the format YYYYMMDDHHMISS anyway. This is just an optional input/output format conversion. Derivation: Mandatory, defaulted with the FILESYSTEM-SYSDATE or a Transaction-Start-Timestamp. Set by the first processor and left unchanged.
TRANSMISSION_DATE	YYYYMMDD	Date on which the file was sent from the sender network to the recipient network or data clearing house. Can be used to calculate the run time of a file/stream between creation and transmission. Also used as a default TRANSFER_CUTOFF_TIMESTAMP. Derivation: Mandatory, defaulted with SYSDATE. Set by the first processor and left unchanged.
TRANSFER_CUTOFF_TIMESTAMP	YYYYMMDDHHMISS	Date and time used to select calls for transfer. All records available prior to the timestamp are transferred. This gives an indication to the recipient as to how current the information is. Can be used to validate that all CDRs are prior to this date and time. Their CHARGING_START_TIMESTAMP must be equal or less. Optional Field Usage: It is possible to read/write dates in number of seconds since 01.01.1970 00:00:00; for example, 12345. The internal representation is the format YYYYMMDDHHMISS anyway. This is just an optional input/output format conversion. Derivation: Mandatory, defaulted with TRANSMISSION_DATE. Set by the first processor and left unchanged.

Table 1-5 (Cont.) Header Record Fields

Name	Format	Description
UTC_TIME_OFFSET	X(5)+/-HHMI	All timestamps are sender (VPLMN) local time. So that the time can be equated to time in the recipient (HPLMN) local time, the sender gives the difference between local time and UTC time. UTC Time Offset = Local Time minus UTC Time. Can be used to translate the TRANSFER_CUTOFF_TIMESTAMP into a unified UTC time. This might be useful if a centralized rating and billing will take place. Example: Washington DC, USA 1000hrs10/10/97 UTC Time1500hrs10/10/97 UTC Time Offset= 10 - 15 = -0500 Madrid, Spain1600hrs10/10/97 UTC Time1500hrs10/10/97 UTC Time Offset= 16 - 15 = +0100 **Note:** Where dates are different, 24 is added to the time of the greater date. Derivation: Mandatory. Set by the first processor and left unchanged.
SPECIFICATION_VERSION_NUMBER	9(2)	Uniquely identifies the format. Different specification versions indicate that the record structure has changed; for example, field length, new fields, and new record types. Used for encoding different formats. Range: 01 Derivation: Mandatory. Set by the first processor and left unchanged.
RELEASE_VERSION	9(2)	Indicates the release version within the Specification Version Number. Different Release Versions indicates that only the content of fields has changed. Used for encoding different formats. Derivation: Mandatory. Set by the first processor and left unchanged.
ORIGIN_COUNTRY_CODE	X(8)	International access and country code, which applies within the country of the network where the CDR originated. Might be useful for an international billing center to distinguish between national and international calls; for example, within the basic detail record. Range: 0049; for example, for Germany. Derivation: Mandatory. Set by the first processor and left unchanged.

Table 1-5 (Cont.) Header Record Fields

Name	Format	Description
SENDER_COUNTRY_CODE	X(8)	International access and country code that applies within the country of the sender (VPLMN). This might be different from the originating code if the sender is a clearing house or third-party operator. Might be useful for an international billing center. Range: 0049; for example, for Germany Derivation: Mandatory. Set by the first processor and left unchanged.
DATA_TYPE_INDICATOR	X(1)	The type of data contained within the file; for example, test or chargeable data. Any customer billing processor should ignore test data or at least separate these streams. Values: T: Test Data Space: Chargeable Data Derivation: Mandatory. Set by the first processor and left unchanged.
IAC_LIST	X(30)	Comma-separated list of all international access codes used within this file. Used during number normalization to detect numbers already starting with these IACs. Those numbers will not be normalized anymore. Example: "001,002" for two IACs Derivation: Optional. Set by the first processor and left unchanged.
CC_LIST	X(30)	Comma-separated list of all country codes used within this file. Used during number normalization to detect all numbers already starting with these CCs. Those numbers are normalized by adding a default IAC. Example: "49,33,1" for two CCs Derivation: Optional. Set by the first processor and left unchanged.
TAP_DECIMAL_PLACES	Integer	Derivation: Optional, but mandatory for RAP output. Set by the input grammar.
OPERATOR_SPECIFIC INFO	String	Derivation: Optional, default = '' Stores a key that identifies the CDR used to generate a specific EDR. Useful for RAP or CIBER return. Must be set by an iScript.
CIBER_FILLER	String	Optional.
CIBER_RECORD_TYPE	String	Optional. See CIBER specs for usage.

Table 1-5 (Cont.) Header Record Fields

Name	Format	Description
RETURN_INDICATOR	String	Optional. See CIBER specs for usage.
CURRENCY	String	Optional. See CIBER specs for usage.
SETTLEMENT_PERIOD	String	Optional. See CIBER specs for usage.
CLEARINGHOUSE_ID	String	Optional. See CIBER specs for usage.
BATCH_REJECT_REASON	String	Optional. See CIBER specs for usage.
BATCH_CONTENTS	String	Optional. See CIBER specs for usage.
SENDING_CLEARINGHOUSE_BID	String	Optional. See CIBER specs for usage.
CREATION_PROCESS	String	Process that created output stream.
SCHEMA_VERSION	String	Version number for schema.
EVENT_TYPE	String	BRM event type.
RAP_FILE_SEQ_NO	String	Optional. Indicates the returned account procedure (RAP) file in which the recipient public data network (PMN) returned the TAP file batch to the sender PMN. This field is a unique reference. Used in TAP files.
QUERYABLE_FIELDS_MAPPING	String	Optional Calculated for suspense handling. Contains the database column names and data types that map to queryable fields. Use this format: *column_name*:*data_type*[;*column_name*:*data_type*[;...]]
BATCH_ID	String	Optional. Set to the actual file batch ID.
UTC_END_TIME_OFFSET	X(5)	Timezone where the call terminated. Derivation: Optional.
BATCH_CTRL_INFO_START_INDEX	Integer	BatchControlInfo block start index.
BATCH_CTRL_INFO_END_INDEX	Integer	BatchControlInfo block end index.
ACCOUNTING_INFO_START_INDEX	Integer	AccountingInfo block start index.
ACCOUNTING_INFO_END_INDEX	Integer	AccountingInfo block end index.
NETWORK_INFO_START_INDEX	Integer	NetworkInfo block start index.
NETWORK_INFO_END_INDEX	Integer	NetworkInfo block end index.
MESSAGE_DESCRIPTION_START_INDEX	Integer	MessageDescriptionInfoList block start index.

Table 1-5 (Cont.) Header Record Fields

Name	Format	Description
MESSAGE_DESCRIPTION_END_INDEX	Integer	MessageDescriptionInfoList block end index.
NOTIFICATION_START_INDEX	Integer	Notification block start index.
DELAYED_ERROR_BLOCK	String	Stores the block name that has the fatal error.
OBJECT_CACHE_TYPE	Integer	Cache residency type: • **0**: Convergent • **1**:- Prepaid • **2**: Postpaid
TAP_FILE_TYPE	String	Type of TAP file, TAP3 or TAP311.

Basic Detail Record (RECType 020-089, 100-299)

This record references a billable event. This basic record is the primary record within the BRM format structure. Table 1-6 lists the fields in the Basic Detail Record.

Table 1-6 Basic Detail Record Fields

Name	Format	Description
RECORD_TYPE	String	Extended to be 3 bytes long. First byte denotes the market. 020 MOC Switch Mobile Originating Call 021 TA_MOC TAP Mobile Originating Call (Roaming**) 022 CFW Mobile Switch Call Forwarding 023 RCF/RFD Mobile Roaming Call Forwarding 024 SMO Mobile Short Message Originating 025 SMT Mobile Short Message Terminating 026 VMO Mobile Voice Mail Originating 027 OAB Mobile Operator Assisted Call (Basic) 028 OAS Mobile Operator Service (Call Completion) 029 MSS Mobile Supplementary Service Event 030 MTC Switch Mobile Terminating Call 031 TA_MTC TAP Mobile Termination Call (Roaming**)

Table 1-6 (Cont.) Basic Detail Record Fields

Name	Format	Description
RECORD_TYPE (cont.)	String	040 SGSN_MOC Serving GPRS Support Node Originating 041 SGSN_MOT Serving GPRS Support Node Terminating 042 GGSN_MOC Gateway GPRS Support Node Originating 043 GGSN_MOT Gateway GPRS Support Node Terminating 044 GPRS_SMO GPRS - Short Message Originating 045 GPRS_SMT GPRS - Short Message Terminating 046 HSCSD_MOC Mobile HSCSD Originating Call 047 HSCSD_MOT Mobile HSCSD Terminating Call 048 TA_GPRSOC TAP GPRS Originating (Roaming**) 049 TA_GPRSTC TAP GPRS Termination (Roaming**) 050 SCU Basic Service Center Usage Record 060 VAS Basic Value Added/Event Record 070 WAP Basic WAP Record 120 POC ISDN/Public Switch Originating 121 DX_POC ISDN/Public Switch Orig.(data exchange) 122 PCF ISDN/Public Switch Call Forwarding 126 PVM ISDN/Public Switch Voice Mail Originating 127 POB ISDN/Public Operator Assisted Call (Basic) 128 POS ISDN/Public Operator Service (Call Com-pl) 130 PTC ISDN/Public Switch Termination Call 131 DX_PTC ISDN/Public Switch Term. (data exchange) 220 IOCBasic Internet Record Other record types might be defined when necessary. Derivation: Mandatory. Set by the first processor and left unchanged. **Note:** Only record types 021 and 031 are treated as roaming calls.
RECORD_NUMBER	9(9)	Sequence number of record in file. Ensures a linear sequence order for all records; for example, as a sorting criteria. Values: Minimum: 000000002 Maximum: 999999998 Derivation: Mandatory. Set by the first processor. Important: Following modules might change this record number; for example, if new record types are inserted.
DISCARDING	9(1)	Indicates if an EDR should be discarded or rejected. Values: 0: Proceed (default) 1: Reject 2-9: Discard The values from 2 to 9 represent different discarding reasons. Derivation: Mandatory. Might be set by any processor.

Table 1-6 (Cont.) Basic Detail Record Fields

Name	Format	Description
CHAIN_REFERENCE	X(10)	Identifies an EDR as part of a long event that has been split into multiple EDRs. Condition: Only present if more than one record is raised for a call (default=Spaces). Value: Any six-digit number. Derivation: Optional. Might only be set by the first processor.

Table 1-6 (Cont.) Basic Detail Record Fields

Name	Format	Description
SOURCE_NETWORK_TYPE	X(1)	The source network type; for example, GSM 900. This is needed for specific implementation models such as some satellite operators where the network originating the chargeable record might be lost. **Note:** This is a temporary solution pending further developments. Values: Mobile-Networks: A: S-41 AMPS A B: S-41 AMPS B C: S-41 Satellite D: S-95 CDMA E: S-136 TDMA F: PDC G: GSM 900 H: GSM 1800 I: GSM 1900 J: GSM 90011800 K: GSM 90011900 L: GSM Satellite M: UMTS N: Telematic O: GPRS - GGSN P: GPRS - SGSN Intercarrier-Networks: W: Inroute X: Outroute T: Transit Z: undefined Fixed-Networks: 0: General Fixed Network 1: Analog 2: ISDN 3: ADSL 4: Multiplex Other-Networks: 9: Internet Other values might apply according to the related original input format. Derivation: Optional. Might be set by the first processor and might be changed by an interconnect rating processor.

Table 1-6 (Cont.) Basic Detail Record Fields

Name	Format	Description
SOURCE_NETWORK	X(14)	Network code from which the call or message was routed. This could either be PLMN_ID or any logical operator code. Used for interconnect rating. Condition: In case of interconnect rating it is overwritten by the network operator code related to the inroute. See TRUNK_INPUT. Derivation: Optional (only mandatory for the interconnect processor).
DESTINATION_NETWORK_TYPE	X(1)	Indicates the destination network type; for example, GSM 900. This is needed for specific implementation models such as some satellite operators where the network terminating the chargeable record might be lost. **Note:** This is a temporary solution pending further developments.Values: See SOURCE_NETWORK_TYPE. Derivation: Optional. Might be set by the first processor and might be changed by an interconnect rating processor.
DESTINATION_NETWORK	X(14)	Network towards which the call or message is routed. Condition: Where a short message has not been delivered or where optimal routing is not used, the field is set to spaces. In case of interconnect rating, it is overwritten by the network operator code related to the outroute. See TRUNK_OUTPUT. Derivation: Optional (only mandatory for interconnect rating).
TYPE_OF_A_IDENTIFICATION	X(1)	Specifies if the number used to identify the subscriber within the network is an IMSI or an MSISDN. This type does not relate to the A Number representation. Values: A: Internet or Account Number (default for internet) C: Calling Card Number I: IMSI M: MSISDN (default for fixed networks) P: IP Number S: SIM-ICC (default for GSM) X: undefined Derivation: Optional. Set by the first processor and left unchanged.

Table 1-6 (Cont.) Basic Detail Record Fields

Name	Format	Description
A_MODIFICATION_INDICATOR	H(2)	Specifies if the called or calling number has been modified by the VPLMN; for example, for privacy reasons. Can be used to evaluate how to handle the number internally; for example, print the last three digits in clear text (anonymize) or suppress the complete CDR during printing a detailed invoice. Condition: PLMNs are not forced to implement this parameter. If not implemented, the number must not be modified. Values: 00: Default setting (undefined) and normal 01: Social number 02: Anonymized number 04: Special number (for example, premium rate) 08: Modified number (for example, vanity routing or short number translation) Derivation: Optional. Set by the first processor and left unchanged.
A_TYPE_OF_NUMBER	Z(1)	Type of address associated with a particular destination or calling number. Condition: Not all networks support this parameter. Values: 0: Nature of address unknown (default) 1: International number 2: National significant number 3: Network-specific number 4: Subscriber number 5: Abbreviated number Derivation: Optional, default=0. From bits 7 - 5 of octet 1 of the GSM Address String type as defined in TS GSM 09.02. Set by the first processor and left unchanged.

Table 1-6 (Cont.) Basic Detail Record Fields

Name	Format	Description
A_NUMBERING_PLAN	X(1)	The numbering plan associated with a particular destination or calling number. Might be useful when analyzing the A Number for normalization reasons; for example, to interpret the structure of the number (for example, distinguish IP numbers. Condition: Not all networks support this parameter. Values: 0: Type of plan unknown (default) 1: ISDN telephony (CCITT E.164) 3: Data numbering plan (CCITT X.121) 4: Telex numbering plan (CCITT F. 69) 5: Reserved for national use 6: Land mobile numbering plan (CCITT E212) 8: National numbering plan 9: Private numbering plan A: Internet, IP-number v4 B: Internet, IP-number v6 Derivation: Optional. From bits 4 - 1 of octet 1 of the GSM Address String type as defined in TS GSM 09.02. The list of values is a comprehensive list of known values and some might not occur in practice. Set by the first processor and left unchanged.

Table 1-6 (Cont.) Basic Detail Record Fields

Name	Format	Description
A_NUMBER	X(40)	Identifies the billable party; for example, the ISDN number, call-line-identity, or source IP address. For MTC and MOC calls, this number contains the subscriber number to be billed, which has not automatically to be the originating number. Condition: Can be used as an alternative to the IMSI, but could also be an Internet account number. Values: Defined in TS GSM 03.03 or in international notation. Should always be: *International_access_codeCountry_codeNational_destination_codeSubscriber_number* Examples: Fixed: +49410676810 Mobile: 01729183333 (Roaming-MOC: 0000*MNC/MCC*, 000026202) IPv4: 192.168.10.2 (always 4 token, each 3 decimals) IPv6: ABCD:10:2:1AF:0:1F0A:F:1F0 (always 8 token, each 4 hex) Derivation: Mandatory. From the GSM item MSISDN as defined in TS GSM 12.05. Set by the first processor and left unchanged, but is normalized: Fixed: 0049410676810 Mobile: 00491729183333 (Roaming-MOC: 0000*MNC/MCC*, 000026202) IPv4: 192168010002 (dots removed, filled with leading zeros) IPv6: ABCD0010000201AF00001F0A000F01F0 (colons removed, filled with leading zeros)
B_MODIFICATION_INDICATOR	H(2)	See A_MODIFICATION_INDICATOR. Optional.
B_TYPE_OF_NUMBER	Z(1)	See A_TYPE_OF_NUMBER. Mandatory.
B_NUMBERING_PLAN	X(1)	See A_NUMBERING_PLAN. Optional, only if available.

Table 1-6 (Cont.) Basic Detail Record Fields

Name	Format	Description
B_NUMBER	X(40)	Identifies the called number. Condition: If there is no called number available (for example, Internet or Telematic), a dummy number has to be inserted instead; for example, 0049. If the rating does not involve zoning, profile-related features, or special numbers, the BRM rating works with even an incomplete called number. Values: Defined in TS GSM 03.03 or in international notation. Should always be: *International_access_codeCountry_codeNational_destination_codeSubscriber_number* Examples: Fixed: 0049410676810 (normal) Fixed: 0049112 (emergency) Fixed: 004970012345678 (vanity) Fixed: 004911833 (special) Mobile: 00491729183333 (normal) Mobile: 0049172559183333 (mailbox) Mobile: 00490172112 (emergency) Mobile: 0049017222255 (special mobile-number) Mobile: 004911833 (special fixed-number) Mobile: 000026202 (Roaming-MTC: 0000*MNC/MCC*) IPv4: 192.168.10.2 (always 4 token, each 3 decimals) IPv6: ABCD:10:2:1AF:0:1F0A:F:1F0 (always 8 token, each 4 hex) Derivation: Mandatory. From the GSM item Called Number as defined in TS GSM 12.05. This item is of type Address String and is further expanded into the items type of number, numbering plan, and the number sent across the air-interface as defined in TS GSM 04.08 and 09.02. Set by the first processor and left unchanged, but is normalized. Fixed: 0049410676810 Mobile: 00491729183333 (Roaming-MOC: 0000*MNC/MCC*, 000026202) IPv4: 192168010002 (dots removed, filled with leading zeros) IPv6: BCD0010000201AF00001F0A000F01F0 (colons removed, filled with leading zeros)
DESCRIPTION	X(50)	Description text for this usage scenario; for example, call destination description for the B Number or service description used. Values: For example, "HAMBURG" or "004940" For example, "Travel Info" or "Wakeup Call" Value is related to the original input format. Derivation: Optional. Calculated by a rating processor or directly taken out of the original CDR stream. Might be changed by any processor.
C_MODIFICATION_INDICATOR	H(2)	See A_MODIFICATION_INDICATOR. Optional. Only mandatory if C Number is present.

Table 1-6 (Cont.) Basic Detail Record Fields

Name	Format	Description
C_TYPE_OF_NUMBER	Z(1)	See A_TYPE_OF_NUMBER. Optional. Only mandatory, if C Number is present.
C_NUMBERING_PLAN	X(1)	See A_NUMBERING_PLAN. Optional. Only mandatory if C Number is present.
C_NUMBER	X(40)	Third-party number; for example, where the call was first terminated in the case of terminated transit or routed, forwarded calls. This field contains the number initiating the call forwarding. **Note:** To avoid any doubt, the 'A to C' and 'C to B' legs of a call-forwarding scenario must be treated as separate calls and the originating and terminating records should never be chained together. Condition: Only present where it is available. Values: See B_NUMBER. Derivation: Optional. From the GSM item Called Number as defined in TS GSM 12.05. This item is of type Address String and is further expanded into the items type of number, numbering plan, and the number sent across the air-interface as defined in TS GSM 04.08 and 09.02. Set by the first processor and left unchanged.
USAGE_DIRECTION	X(1)	Describes the direction of the connection that was established. Can be used by any rating and post-processor to identify the direction of the event; for example, to determine a specific call scenario. Values: 0: Originated usage 1: Terminated usage 2: Roaming originated usage 3: Roaming terminated usage Examples: 0: MOC, mobile originated WAP, … 1: MTC, mobile terminated WAP, … 2: roaming MOC 3: roaming MOC Derivation: Mandatory. Set by the first processor and left unchanged.

Table 1-6 (Cont.) Basic Detail Record Fields

Name	Format	Description
CONNECT_TYPE	X(2)	Type of connection. Can be used to identify the type of the call; for example, to determine a specific call scenario. Values: 01: Mobile to Mobile 02: Mobile to Land 03: Land to Mobile 04: Land to Land 10: Effective POTS Call 11: Effective ISDN Call 12: Effective Call Diversion 13: Subscriber Procedure 14: Subscriber Service Command 15: Effective ISDN-E Call 16: Ineffective POTS Call 17: Ineffective ISDN Call 18: Ineffective Call Diversion 19: Ineffective ISDN-E Call 20: Non Call Related Output 30: Anonymous login Other values might apply according to the related original input format. Derivation: Mandatory. Set by the first processor and left unchanged.
CONNECT_SUB_TYPE	X(2)	Detailed description of the connection or call type. Can be used to identify the type of the call; for example, to determine a specific call scenario. Values: 01: Mobile-Call 02: Local-Call (for example, BestCity, BestFriend, etc.) 03: National-Call 04: International-Call Derivation: Mandatory. Set by the first processor and left unchanged.
CALLED_COUNTRY_CODE	String	Derivation: Optional, but should be set when the CONNECT_TYPE indicates an international call
BASIC_SERVICE	X(3)	Uniquely identifies the basic usage-related service. A service is defined by: • Service type • Service code

Table 1-6 (Cont.) Basic Detail Record Fields

Name	Format	Description
BASIC_SERVICE	X(1)	Specifies the type of basic service. Values: 0: Tele Service (for example, GSM Tele, ISDN, analog, standard, etc.) 1: Bearer Service 2: Supplementary Service 3: Telematic Service (only if present) 4: Internet Service 5: ISDN mobile (only if present) 6: Mailbox (only if present) 7: VPN mobile (only if present) 8: GPRS 9: Switch related (for example, for direct fixed network support) E: Event/VAS Services W: WAP Other service types might be defined when necessary. Derivation: Mandatory. Set by the first processor and left unchanged.
SERVICE_CODE	X(2)	The Service Code is either a Teleservice Code or Bearer Service Code as determined by Service Type. Values: **Note:** Other values might apply according to the related original input format.All Tele Services: 10: All Voice 11: Telephony (default) 12: Emergency calls 13: GPRS (default) 14: HSCSD (default) 15: WAP (default) 20: All Short Message Service 21: Short Message MT/PP 22: Short Message MO/PP 30: MHS 31: Advanced MHS access 40: All Videotext 41: Videotext access profile 42: Videotext access profile 2 43: Videotext access profile 3 50: All Teletext Transmission Services 51: Teletext (Circuit Switch) 52: Teletext (Packet Switch) 61: Facsimile Group 3 & alternative voice 62: Automatic Facsimile Group 3 63: Automatic Facsimile Group 4

Table 1-6 (Cont.) Basic Detail Record Fields

Name	Format	Description
SERVICE_CODE (continued)	X(2)	All Bearer Services: 10: 3.1 KHz Bearer Group 20: Circuit data asynchronous 21: Duplex Asynchronous 300bps data circuit 22: Duplex Asynchronous 1200bps data circuit 23: Duplex Asynchronous 1200/75bps data circuit 24: Duplex Asynchronous 2400bps data circuit 25: Duplex Asynchronous 4800bps data circuit 26: Duplex Asynchronous 9600bps data circuit 30: Circuit data synchronous 32: Duplex Synchronous 1200bps data circuit 34: Duplex Synchronous 2400bps data circuit 35: Duplex Synchronous 4800bps data circuit 36: Duplex Synchronous 9600bps data circuit 40: PAD access asynchronous 41: Duplex Asynchronous 300bps PAD access 42: Duplex Asynchronous 1200bps PAD access 43: Duplex Asynchronous 1200/75bps PAD access 44: Duplex Asynchronous 2400bps PAD access 45: Duplex Asynchronous 4800bps PAD access 46: Duplex Asynchronous 9600bps PAD access 50: PAD access synchronous 54: Duplex Synch. 2400bps PAD access 55: Duplex Synch. 4800bps PAD access 56: Duplex Synch. 9600bps PAD access 60: All alternate voice/data c.d.a 61: Alt. Voice/Asynch. 300bps unrest'd digital 62: Alt. Voice/Asynch. 1200bps unrest'd digital 63: Alt. Voice/Asynch. 1200/75bps unrest'd digital 64: Alt. Voice/Asynch. 2400bps unrest'd digital 65: Alt. Voice/Asynch. 4800bps unrest'd digital 66: Alt. Voice/Asynch. 9600bps unrest'd digital 70: Alternate voice data c.d.s 72: Alt. Voice/Synch. 1200bps unrest'd digital 74: Alt. Voice/Synch. 2400bps unrest'd digital 75: Alt. Voice/Synch. 4800bps unrest'd digital 76: Alt. Voice/Synch. 9600bps unrest'd digital 80: Voice followed by data c.d.a

Table 1-6 (Cont.) Basic Detail Record Fields

Name	Format	Description
SERVICE_CODE (continued)	X(2)	All Bearer Services: (continued) 81: Voice then Asynch. 300bps unrest'd digital 82: Voice then Asynch. 1200bps unrest'd digital 83: Voice then Asynch. 1200/75bps unrest'd digital 84: Voice then Asynch. 2400bps unrest'd digital 85: Voice then Asynch. 4800bps unrest'd digital 86: Voice then Asynch. 9600bps unrest'd digital 90: All Voice followed by data c.d.s 92: Voice then Synch. 1200bps unrest'd digital 94: Voice then Synch. 2400bps unrest'd digital 95: Voice then Synch. 4800bps unrest'd digital 96: Voice then Synch. 9600bps unrest'd digital
SERVICE_CODE (continued)	X(2)	All Telematic Service: 01: SMS 02: E-Mail 03: Pull-Service 04: Short-Fax 05: Push-Service All Internet Service: 10: all Internet 11: direct Access 12: WebMaster Emergency 13: Voice over IP 14: Fax over IP 20: E-Mail 30: Active Channel 40: Video on demand 41: Music on demand 50: Newsgroup access 62: Internet Fax other values Can be used according to the network. All Switch-related Services or WAP/GPRS Services: 01...99: see related switch documentation (values used 1:1) AA...ZZ: see related switch documentation (values used 1:1) All Event/VAS Services: 00: all default Event/VAS usage other Service Codes might be defined when necessary. Derivation: Mandatory. Set by the first processor and left unchanged.

Table 1-6 (Cont.) Basic Detail Record Fields

Name	Format	Description
QOS_REQUESTED	X(5)	The type of QoS requested by the Terminal Equipment (TE) at usage setup or the QoS requested to the Network Equipment (NE). The QoS Requested is typically a normalized billable QoS value. For detailed network-related QoS attributes, see the related Associated Service Extension Record. Condition: The use of a QoS might not be appropriate; for example, call forwarding cases, short message services. Applies only where appropriate information is available. Values for Radio Channel: H(2) 00: Half Rate Channel 01: Full Rate Channel 02: Dual Rate Channel, Half Rate Preferred 03: Dual Rate Channel, Full Rate Preferred Values for Quality of Service: X(5) xxxxx: Any alphanumeric representation of the NE Value is according to the related original input format. Derivation: Optional. From the GSM item RadioChanRequested, a component of RadioChanInfo as defined in TS GSM 12.05. or directly out of the NE-interface. Encoded as per TS GSM 04.08. Set by the first processor and left unchanged.
QOS_USED	X(5)	The type of QoS negotiated by the network. The QoS used is typically a normalized billable QoS value. For detailed network-related QoS attributes, see the related Associated Service Extension Record. Condition: The use of a QoS might not be appropriate; for example, call forwarding cases, short message services. Applies only where appropriate information is available. Values for Radio Channel: H(2) 00: Half Rate Channel 01: Full Rate Channel Values for Quality of Service: X(5) xxxxx: any alphanumeric representation of the NE Values is according to the related original input format. Derivation: Optional. From the GSM item channel type, a component of RadioChannel-Used as defined in TS GSM 12.05. Encoded as per TS GSM 04.08 or directly out of the NE-interface. Set by the first processor and left unchanged.

Table 1-6 (Cont.) Basic Detail Record Fields

Name	Format	Description
CALL_COMPLETION_INDICATOR	X(3)	Indicates whether a call was correctly completed or not. Optionally defines a close cause reason code. Single-Byte Values X(1): if no reason code is available: C: Call completed, charged as usual D: Call dropped, treatment open, but will first be charged as usual T: Call completed, test call, will not be charged Optional Values after rating processor: 0: After rating processor: not rated yet: should be rated later by the billing post-processor: call completed 1: After rating processor: rated: should not be rated again by a billing post-processor - call completed 2: After rating processor: not rated yet: should be rated later by the billing post-processor: call dropped 3: After rating processor: rated: should not be rated again by a billing post-processor: call dropped Double-Byte Values X(2) - if a reason code is available: 00: normal 01: partial record 02: partial call re-establishment 03: unsuccessful call attempt 04: abnormal release 05: camel init call release 16: volume limit 17: time limit 18: network element switch 19: max. change condition 20: management intervention other values can be used according to the network. Derivation: Mandatory, default C. Set by the first processor and left unchanged.
LONG_DURATION_INDICATOR	X(1)	Specifies which part of the call, in the case of split calls. Values: S: Single (only one record present) F: First (the first record in the row of split records) I: Intermediate (one of the middle records in the row of records) L: Last (the last record in the row of split records) Derivation: Mandatory, default S. Set by the first processor and left unchanged.

Table 1-6 (Cont.) Basic Detail Record Fields

Name	Format	Description
CHARGING_START_TIMESTAMP	YYYYMMDDHHMISS	Timestamp used for start of charging. In the mobile originated case, this is as determined by the VPLMN's charging rules. In the mobile terminated case, it is also at the discretion of the VPLMN, even though the information is for use in charging by the VPLMN. Format: YYYYMMDDHHMISS; for example, 19990518190357 Optional Field Usage: It is possible to read/write dates in number of seconds since 01.01.1970 00:00:00; for example, 12345. The internal representation is the format YYYYMMDDHHMISS anyway. This is just an optional input/output format conversion. Derivation: Mandatory. From the GSM item answer-time or seizure-time as defined in TS GSM 12.05. Set by the first processor and left unchanged.
CHARGING_END_TIMESTAMP	YYYYMMDDHHMISS	Timestamp used for end of charging. In the mobile originated case, this is as determined by the VPLMN's charging rules. In the mobile terminated case, it is also at the discretion of the VPLMN, even though the information is for use in charging by the VPLMN. Format: YYYYMMDDHHMISS; for example, 19990518190357 Optional Field Usage: It is possible to read/write dates in number of seconds since 01.01.1970 00:00:00; for example, 12345. The internal representation is the format YYYYMMDDHHMISS anyway. This is just an optional input/output format conversion. Derivation: Optional. Might be set by the first processor and left unchanged. **Note:** If not present, this value can be calculated by using the start timestamp and the duration.
CREATED_TIMESTAMP	Date	Optional. The time that the event was created in BRM.

Table 1-6 (Cont.) Basic Detail Record Fields

Name	Format	Description
UTC_TIME_OFFSET	X(5)+/-HHMI	All timestamps are VPLMN or originating network local time. So that the time can be equated to time in the HPLMN or recipient network, the sender gives the difference between local time and UTC time. Can be used to translate the CHARGING_START/END_TIMESTAMP into a unified UTC time. This might be useful if a centralized rating and billing will take place. Values: UTC Time Offset = Local Time minus UTC Time Example: Washington DC, USA 1000hrs 10/10/97 UTC Time 1500hrs 10/10/97 UTC Time Offset = 10 - 15 = -0500 Madrid, Spain 1600hrs 10/10/97 UTC Time 1500hrs 10/10/97 UTC Time Offset = 16 - 15 = +0100 Sydney, Australia 0100hrs 11/10/97 UTC Time 1500hrs 10/10/97 UTC Time Offset = (01 + 24) - 15 = +1000 (Note that where dates are different, 24 is added to the time of the greater date.) Derivation: Mandatory. Set by the first processor and left unchanged.
DURATION	9(15)	Duration-based charge indicates that the field represents a Chargeable Duration. Can be used to evaluate all duration-based functions; for example, determination of the pricing rating steps. Condition: For event-based charges or an inter-network account charge, the field is not relevant. URC.01 implementation of the TD.17 item Chargeable Units. Maximum Value: 999999999999999 Derivation: Mandatory, default 0. Set by the first processor and left unchanged but might be patched by some kind of rating processors.
TOTAL_CALL_EVENT_DURATION	Integer	The total duration of the event. This should be set for all time-based services; for example, telephony. Mandatory. The default value is **0**.

Table 1-6 (Cont.) Basic Detail Record Fields

Name	Format	Description
DURATION_UoM	X(3)	Unit of Measurement associated with Chargeable Quantity Value. Can be used to interpret the quantity value, but usually not needed, because the quantity itself is sufficient for all rating steps. Values: SEC: Seconds (default) MIN: Minutes HRS: Hours or any other applicable value describing a timed quantity unit of measurement. Derivation: Mandatory, default 'SEC'. Set by the first processor and left unchanged.
VOLUME_SENT_	9(15)	In addition to the basic duration quantity value, a special volume might be defined to keep an additional rating relevant measurement. This is typically BYTES sent by the initiator (A number). Maximum Value: 999999999999999 Can be used to evaluate additional volume-based functions; for example, determination of the pricing rating steps. Derivation: Mandatory, default 0. Set by the first processor and left unchanged but might be patched by some kind of rating processors.
VOLUME_SENT_UoM	X(3)	The Unit of Measurement associated with VOLUME_SENT. Can be used to interpret the quantity value, but usually not needed because the quantity itself is sufficient for all rating steps. Values: BYT: Bytes/Characters (default) KBY: Kilobytes MBY: Megabytes GBY: Gigabyte or any other applicable value describing a metered quantity unit of measurement. Derivation: Mandatory, default 'BYT'. Set by the first processor and left unchanged.
VOLUME_RECEIVED	9(15)^	In addition to the basic duration value, a special volume might be defined to keep an additional rating relevant measurement. This is typically BYTES received by the initiator (A Number). Maximum Value: 999999999999999 Can be used to evaluate an additional volume-based functions; for example, determination of the pricing rating steps. Derivation: Mandatory, default 0. Set by the first processor and left unchanged but might be patched by some kind of rating processors.

Table 1-6 (Cont.) Basic Detail Record Fields

Name	Format	Description
VOLUME_RECEIVED_UoM	X(3)	The Unit of Measurement associated with VOLUME_RECEIVED value. Can be used to interpret the quantity value, but usually not needed because the quantity itself is sufficient for all rating steps. Values: • BYT: Bytes/Characters (default) • KBY: Kilobytes • MBY: Megabytes • GBY: Gigabyte • Any other applicable value describing a metered quantity unit of measurement Derivation: Mandatory, default 'BYT'. Set by the first processor and left unchanged.
NUMBER_OF_UNITS	9(15)	One of the following: • Original charged units (for example, beats, clicks) as applied by the sender • Rounded total volume charged by the sender • Number of events associated with this record (for example, number of SMS messages or number of internet hits/clicks) Might be useful for analyzing how many units the event was originally treated by or for storing a fourth quantity. Condition: Applies only if available. Alternative URC.01 implementation of the TD.17 item Chargeable Units. Maximum Value: 4294967296 Derivation: Mandatory, default 0. Set by the first processor and left unchanged, but might be changed by some type of rating processors.
NUMBER_OF_UNITS_UoM	X(3)	The Unit of Measurement associated with NUMBER_OF_UNITS value. Can be used to interpret the quantity value, but usually not needed because the quantity itself is sufficient for all rating steps. Values: CLK: Clicks (anonymous quantity) (default) MSG: Messages PAG: Pages PAC: Packets PIC: Pieces RTS: Points MTR: Meters KMR: Kilometer SPD: Speed TRN: Transactions or any other applicable value describing a metered quantity unit of measurement. Derivation: Mandatory, default CLK. Set by the first processor and left unchanged.

Table 1-6 (Cont.) Basic Detail Record Fields

Name	Format	Description
RETAIL_IMPACT_CATEGORY	X(10)	Impact category defining the usage scenario specific rate; for example, the zone value used for customer rating. Values: 00000: undefined impact category (default) 00001 - 99999: user defined Alternatively a user-defined string can be used as a value. Derivation: Optional, but mandatory after any rating processor, default 00000. Might be changed by any processor.
RETAIL_CHARGED_AMOUNT_VALUE	9(11)	The charge for the event (for example, the retail price). This includes any toll charge but does not include any CAMEL invocation fee. Values: Space: No price given, like NULL in a database Variable floating point format: Given value, might be 0.000. The floating decimal point must be set. Minimum: -9999999999 Maximum: 99999999999 Examples: '00000000125' for 125,00 '00000012.50' for 12,50 '-0012.56780' for -12,5678 Derivation: Optional, but mandatory after any customer rating processor.
RETAIL_CHARGED_AMOUNT_CURRENCY	X(3)	Currency code as defined within the associated charge; for example, DEM or EUR. Derivation: Optional, but mandatory whenever the RETAIL_CHARGED_AMOUNT_VALUE is set. Use the three-digit ISO currency code.
RETAIL_CHARGED_TAX_TREATMENT	X(1)	Charges might be inclusive or exclusive of tax. Can be used to interpret the amount value and to distinguish between net and gross charges. Values: Y: Tax included in the charge N: Tax not included in the charge (default) Derivation: Optional, but mandatory whenever the RETAIL_CHARGED_AMOUNT_VALUE is set.

Table 1-6 (Cont.) Basic Detail Record Fields

Name	Format	Description
RETAIL_CHARGED_TAX_RATE	9(4)	Defines the tax rate applicable to the charge. Because some national legal definitions dictate that the tax rate applicable is determined by the invoice date, there is a possibility that the rate on the invoice might differ from the rate on the transfer. However, the likelihood of this happening is extremely low. Can be used to interpret the amount value and to convert between net and gross charges. Values: 0000 through 9999 (2 fixed decimals) Example: 16.00% 1600 Derivation: Optional, but mandatory whenever the RETAIL_CHARGED_AMOUNT_VALUE is set.
RETAIL_CHARGED_TAX_VALUE	Decimal	Derivation: Calculated, default = 0.0
WHOLESALE_IMPACT_CATEGORY	X(10)	Wholesale/Advice of Charge - Impact category used for purchase rating. Values: 00000: undefined impact category (default) 00001 - 99999: user-defined Derivation: Optional. Might be changed by any processor.
WHOLESALE_CHARGED_AMOUNT_VALUE	9(11)	Wholesale/Advice of Charge: charge for the event (for example, the wholesale price). This includes any toll charges. Can be used to keep the original purchase charge, to evaluate a record-based margin together with the charged amount value. Values: Space: No price given, like NULL in a database Floating point format: Given value, might be 0.000. If no floating point exists, the last three digits are always taken as decimals) Minimum: -9999999999 Maximum: 99999999999 Examples: '00000012500' for 12,50 '-0001200100' for -1.200,10' '00000012.50' for 12,50 '00012.50000' for 12,50 Derivation: Optional. Usually transmitted by the sender (origin network operator) of the file, but might also be recalculated by any processor to represent the purchase charge.
WHOLESALE_CHARGED_AMOUNT_CURRENCY	X(3)	See RETAIL_CHARGED_AMOUNT_CURRENCY.
ZONE_DESCRIPTION	String	Calculated. Used by zoning and rating modules.
IC_DESCRIPTION	String	Used by the zoning and rating modules.

Table 1-6 (Cont.) Basic Detail Record Fields

Name	Format	Description
ZONE_ENTRY_NAME	String	Calculated. Used by zoning and rating modules.
WHOLESALE_CHARGED_TAX_TREATMENT	X(1)	See RETAIL_CHARGED_TAX_TREATMENT.
WHOLESALE_CHARGED_TAX_RATE	9(4)	See RETAIL_CHARGED_TAX_RATE.
WHOLESALE_CHARGED_TAX_VALUE	Decimal	Derivation: Calculated, default = 0.0
TARIFF_CLASS	X(10)	Tariff Class contains tariff information as represented within the original CDR format; for example, wholesale tariff model identification. Can be used to evaluate the original charge configuration in conjunction with the BASIC_AoC_AMOUNT_VALUE. Condition: Only present if original purchase charge information is available. Values: Dependent on the original format. No format conversion will take place. See the appropriate documentation of the original format. Derivation: Optional (but should by mandatory for all mobile (GSM) related records). Might be set by any processor.
TARIFF_SUB_CLASS	X(10)	Contains detailed tariff information as represented within the original CDR format; for example, wholesale zone identification. Can be used to evaluate the origin charge configuration in conjunction with the WHOLESALE_CHARGED_AMOUNT_VALUE. Condition: Only present if origin purchase charge information is available. Values: Dependent on the original format. No format conversion will take place. See the appropriate documentation of the original format. Derivation: Optional (but should by mandatory for all mobile (GSM) related records). Might be set by any processor.
USAGE_CLASS	X(5)	Specifies a format-related usage scenario; for example, call forwarding, roaming, mailbox request, or local calls. Can be used to evaluate the origin call scenario. The call class can be used to convert a scenario into a combined zone value or to identify specific rating specialties. Therefore the call class consists of original record fields. Values: Dependent on the original format. No format conversion or normalization will take place. The content is derived from any rule-based translations of any available raw event attributes, to represent all possible usage scenarios of the origin format. 00000: undefined usage class Derivation: Mandatory. Should be set by the first processor and left unchanged.

Table 1-6 (Cont.) Basic Detail Record Fields

Name	Format	Description
USAGE_TYPE	X(5)	Specifies a customer-related usage scenario; for example, customer-to-customer call, birthday call, or closed-user-group calls. Can be used to evaluate an A Number-customer and B Number-customer related scenario (using direct access to specific customer-info-fields). The call type can be used to convert a scenario into a combined zone value or to calculate a record-based discount when estimating the charging amounts. Values: User definable values might be used. The content of the field depends on the rule-based configuration. 00000: undefined usage type Derivation: Optional. Might be changed by any rating or billing processor.
EVENT_TYPE	String	BRM event type.
SERVICE_TYPE	String	BRM service type. When the service has a subscription service, the string is separated by semicolons; for example, /service/gsm/data;/service/gsm)
BILLCYCLE_PERIOD	YYYYMMBC	Defines the next open billcycle period this event belongs to. Can be used to group or split the EDR stream into billcycle-related smaller portions. Condition: Only present if a rating or pre-billing processor evaluates the next billcycle period for the A number customer. Values: YYYY: The actual year of the next open billcycle period. MM: The actual month of the next open billcycle period. BC: The billcycle identifier. Derivation: Optional, but should be mandatory for any pre-billing processor.
PREPAID_INDICATOR	9(2)	Specifies if the event is a prepaid event. Can be used to identify prepaid scenarios within a mixed post-/prepaid environment. Values: Default: no prepaid scenario prepaid scenario Derivation: Mandatory. Set by the first processor and left unchanged.
NUMBER_ASSOCIATED_RECORDS	9(2)	Number of associated records attached to this basic detail record. Can be used to evaluate how many associated records have to be read ahead. Values: 00: No associated records attached, next record is a basic one. 01-99: Number of associated records followed by this record. Derivation: Mandatory. Must be changed by any processor if new associated records are added.

Table 1-6 (Cont.) Basic Detail Record Fields

Name	Format	Description
NUMBER_OF_CDRS	Integer	Number of CDRs that were compiled into this EDR during call assembly. Derivation: Optional. Calculated.
ERROR_REJECT_TYPE	String	Used by the FCT_Reject to reject the DETAIL to another stream than the standard reject stream. Derivation: Optional, default = ' '
OPERATOR_SPECIFIC_INFO	String	Stores a key to identify the CDR used to generate a specific EDR. Useful for RAP or CIBER return. Derivation: Optional, default = ' ' Must be set by an iScript.
DISCOUNT_KEY	String	N/A
GEOGRAPHICAL_LOCATION	String	Conditional in TAP 3.10. Indicates the geographical location of the terminal equipment. Format: This field contains comma-separated tag-value pairs that indicate the geographical location of the serving network, serving BID, serving location description, longitude, and latitude. The tag values of the corresponding fields are as follows: • ServingNetwork: 1 • ServingBID: 2 • ServingLocationDescription: 3 • Longitude: 4 • Latitude: 5 Example 1: If the TAP field values are as follows: • ServingNetwork: **AIRTEL** • ServingBID: **AIRBID** • ServingLocationDescription: **Bangalore** • Longitude: **111** • Latitude: **103** The value of DETAIL.GEOGRAPHICAL_LOCATION would be: **1,AIRTEL, 2,AIRBID, 3,Bangalore, 4,111,5,103** Example 2: If the TAP field values are as follows: • ServingNetwork: **AIRTEL** • ServingBID: **AIRBID** • Latitude: **103** The value of DETAIL.GEOGRAPHICAL_LOCATION would be: **1,AIRTEL, 2,AIRBID, 5,103**
FRAUD_MONITOR_INDICATOR	String	Conditional in TAP 3.10. Indicates that the chargeable subscriber is flagged for fraud information collection purposes. Possible values: **1 -** Fraud Monitored Subscriber If the field is present, it should have a value of **1**.
ORIGINAL_BATCH_ID	String	Optional, but might be set equal to the original file batch ID.

Table 1-6 (Cont.) Basic Detail Record Fields

Name	Format	Description
BATCH_ID	String	Optional, but might be set equal to the recycle or rerate file batch ID.
NE_CHARGING_START_TIMESTAMP	Date	Network Element date/time stamp. Time at which the call started. Derivation: Optional.
NE_CHARGING_END_TIMESTAMP	Date	Network Element date/time stamp. Time at which the call ended. Derivation: Optional.
UTC_NE_START_TIME_OFFSET	X(5)	Optional.
UTC_NE_END_TIME_OFFSET	X(5)	Optional.
UTC_END_TIME_OFFSET	X(5)	Timezone where the call terminated. Derivation: Optional.
INCOMING_ROUTE	X(30)	Incoming route name. Optional.
ROUTING_CATEGORY	X(20)	Category denoting the routing of the call to the destination party. Optional.
DISCARD_REASON	String	The reason for discarding the EDR. This field is set by FCT_Discard.
CREDIT_LIMIT_CHECK	Integer	Specifies whether to perform credit limit checking on the EDR: • 1 = Perform a credit limit check • 0 = Skip the credit limit check Mandatory.
CREDIT_LIMIT_CHECK_RESULT	Integer	Specifies whether the EDR passed or failed the credit limit check: • 1 = The EDR passed the simple credit limit check • 0 = The EDR failed the simple credit limit check Mandatory.
UNRATED_QUANTITY	Decimal	Unrated quantity filled in after credit limit check.
REFRESH_BALANCE	Integer	Specifies whether the latest balance information should be retrieved from the database. When this field is set, the discounting module calls the balance module to get the latest balance information from the database, whether or not a balance packet is present in the EDR.
OBJECT_CACHE_TYPE	Integer	Cache residency type. • **0**: Convergent • **1**: Prepaid • **2**: Postpaid
DELAYED_ERROR_BLOCK	String	Stores the block name that has the fatal error.
EVENT_ID	String	Used by Revenue Assurance.
ITEM_TAG	String	Used by FCT_ItemAssign. Calculated.
RERATE_TAG	Integer	Used for re-rating

Table 1-6 (Cont.) Basic Detail Record Fields

Name	Format	Description
DROPPED_CALL_QUANTITY	Decimal	Duration of a dropped call.
DROPPED_CALL_STATUS	Integer	Status of a dropped call. • 0 = No dropped call service-level ERA associated with the service. • 1 = The call is a dropped call. • 2 = Continuation call. • 3 = Both a dropped call and a continuation call. • 4 = Does not meet the criteria for either a dropped call or a continuation call.
NET_QUANTITY	Decimal	Contains the summation of the BALANCE_PACKET.PIN_QUANTITY for the associated RUM.
INTERN_attributes (shown below)	As shown below	The following INTERN_ attributes are used by specific modules to temporarily store calculated values. They are all mandatory, but only from a definition point of view. The content value of these fields will be filled automatically by the appropriate modules. All other modules should use these values as read only.
INTERN_ZONE_MODEL	Integer	The internal zone model used by zoning, rating, and discounting Mandatory. Calculated.
INTERN_NETWORK_MODEL	String	The internal network model code. Mandatory. Calculated.
INTERN_NETWORK_OPERATOR	String	The internal network operator code. Used by Interconnect aggregation. Mandatory. Calculated.
INTERN_APN_GROUP	String	The internal APN group code used by zoning. Mandatory. Calculated.
INTERN_TERMINATING_SWITCH_IDENTIFICATION	String	The internal terminating switch ID. Used by output mapping Mandatory. Calculated.
INTERN_BILLING_CURRENCY	String	The internal billing currency used by exchange rate conversion. Mandatory. Calculated.
INTERN_HOME_CURRENCY	String	The internal home currency used by exchange rate conversion. Mandatory. Calculated.
INTERN_SLA_USC_GROUP	String	The internal customer-related SLA-based usage scenario map group code. Mandatory. Calculated.
INTERN_SLA_RSC_GROUP	String	The internal customer-related SLA-based rate service class map group code Mandatory. Calculated.
INTERN_SLA_IRULE_SET	String	The internal customer-related SLA-based irule_set-code. Mandatory. Calculated.
INTERN_PROCESS_STATUS	Integer	Possible values are • 0 = normal (default) • 1 = recycling • 2 = recycling-test Mandatory. Calculated.

Table 1-6 (Cont.) Basic Detail Record Fields

Name	Format	Description
INTERN_BALANCE_GROUP_ID	String	The balance group of the service to which the event belongs. Optional.
INTERN_SERVICE_BILL_INFO_ID	String	The billinfo of the service's balance group. Optional.
INTERN_DISCOUNT_OWNER_ACCT_ID	String	Optional.
INTERN_SERVICE_BILL_INFO_ID	String	The billinfo of the service's balance group. Optional.
ACCOUNT_ID	String	The POID of the Customer A account. Optional.
TB_RECORD_NUMBER	Integer	The Trigger Billing Record number used by the Trigger Bill Output Mapping to assign the array index set by the grammar. Optional.
RAP_FILE_SEQ_NO	String	Indicates the Returned Account Procedure (RAP) file in which the Recipient PMN returned the TAP file record to the Sender PMN. This field is a unique reference. Used in TAP files. Optional.
PROFILE_LABEL_LIST	String	A list of unique labels of all shared profiles having attributes matching a specific EDR field or event attribute. Optional. Calculated.
DROPPED_CALL_QUANTITY	Decimal	When the EDR is flagged as a continuation call, this field stores the duration of the associated dropped call.
DROPPED_CALL_STATUS	Integer	Specifies whether the EDR is for a normal call (**0**), a dropped call (**1**), a continuation call (**2**), both a dropped call and a continuation call (**3**), or an already processed EDR (**4**).

Associated Revenue Assurance Extension Record

Table 1-7 lists the fields in the Associated Revenue Assurance Extension Record. This record is optional with an occurrence of 0 or 1 time only.

Table 1-7 Associated Revenue Assurance Extension Record Fields

Name	Format
BATCH_ID	String
CDR_FILE_NAME	String
START_TIME	String
EDR_STATUS	String
REVENUE_STREAM	String
OUTPUT_STREAM	String
DISCOUNT_AMOUNT	Decimal
OLD_DISCOUNT_AMOUNT	Decimal

Table 1-7 (Cont.) Associated Revenue Assurance Extension Record Fields

Name	Format
CHARGED_AMOUNT	Decimal
OLD_CHARGED_AMOUNT	Decimal

Associated GSM/Wireline Extension Record (RECType 520)

This record is optional and will be generated only if the related Basic Detail Record indicates a GSM or Wireline service. Table 1-8 describes the fields in the Associated GSM/Wireline Extension Record.

Table 1-8 Associated GSM/Wireline Extension Record Fields

Name	Format	Description
RECORD_TYPE	String	Extended to be 3 bytes long. Value: 520 - GSM/Wireline Extension Record Derivation: Mandatory. Set by the first processor and left unchanged.
RECORD_NUMBER	9(9)	Sequence number of record in file. Used to ensure a linear sequence order for all records; for example, as a sorting criteria. Values: Minimum: 000000002 Maximum: 999999998 Derivation: Mandatory. Set by the first processor. **Important:** Following modules might change this record number; for example, if new record types are inserted.
PORT_NUMBER	X(24)	Identifies the customer to charge; for example, the IMSI or SIM number. Condition: For Value Added Services and APLMN Service Center Usage, either the IMSI or MSISDN might be supplied, although one of them must be supplied and, where available, the IMSI is preferred. For normal mobile calls, the SIM number is preferred. Derivation: Optional. From the GSM item served IMSI as defined in TS GSM 12.05. Defined in TS GSM 03.03.

Table 1-8 (Cont.) Associated GSM/Wireline Extension Record Fields

Name	Format	Description
DEVICE_NUMBER	X(24)	Identifies the equipment used by the subscriber during the call; for example, the International Mobile Equipment Identity number (IMEI). Condition: Not present where the terminal equipment is not involved in the call; for example, in forwarded call cases. It is not mandatory for the VPLMN to transfer this information. Derivation: Optional. From the GSM item IMEI as defined in TS GSM 12.05. Defined in TS GSM 03.03. **Note:** Even though the IMEI is 16 digits in length, the check digit is not transmitted.
A_NUMBER_USER	X(40)	The customer who owns the number from which the call was originated, for terminated calls. Not used for rating, but could be used on invoices. Condition: There is no calling number present where it is unavailable. Could be different from the A Number; for example, in case of VPN calls. For VPN calls, the A Number contains the party to be billed, and this field contains the user initiating the call. Values: See A_NUMBER. Derivation: Optional. From the GSM item Calling Number as defined in TS GSM 12.05. This item is of type Address String and is further expanded into the items type of number, numbering plan, and the number sent across the air-interface as defined in TS GSM 04.08 and 09.02 or in international notation. Set by the first processor and left unchanged.
DIALED_DIGITS	X(40)	The number dialed by the customer when establishing a call, or the number to which the call is forwarded or transferred. Can be used for managing disputes. Condition: There might be no called number for the basic service emergency call but operators might optionally insert the digits 112 or their national emergency number into this field. The notation should always be local; for example, 04106768124. Values: See B_NUMBER. Derivation: Optional. From the GSM item Calling Number as defined in TS GSM 12.05. This item is of type Address String and is further expanded into the items type of number, numbering plan, and the number sent across the air-interface as defined in TS GSM 04.08 and 09.02 or in international notation. Set by the first processor and left unchanged.
BASIC_DUAL_SERVICE	X(3)	A dual service can be used in context with twin or duo cards.

Table 1-8 (Cont.) Associated GSM/Wireline Extension Record Fields

Name	Format	Description
VAS/PRODUCT_CODE	X(10)	A classification of Value Added Services as generated by the sender. Can be used to map to a specific internal service code to implement specific usage scenarios for any rating purposes. Values: VMAIL: Voice Mail Services SEC: Secretarial Services OPER: Telephonic Operator Services FI: Financial Information TRAVEL: Travel Information This is not a definitive list and might be added to through MoU-TADIG from time to time or might be user defined. Derivation: Optional. From the GSM item vasCode as defined in GSM TD17. Set by the first processor and left unchanged.
ORIGINATING_SWITCH_IDENTIFICATION	X(15)	Identifies the MSC or SwitchID handling the origin of the call. In case of mobile roaming calls (GSM), this field contains the MOC-related MCC/MNC. In case of wireline networks, this field contains the primary switch that generated this CDR. Can be used by any interconnect rating processor to uniquely identify the trunk names but will only be used if the trunk names are only unique within the related switch. See TRUNK_INPUT and TRUNK_OUTPUT. Can also be used to normalize the A Number for MOC roaming. In case of roaming, this field contains the MCC/MNC. Encoding: Encoded as one of the following according to the requirements of the sender: • The MSISDN of the MSC as per GSM 03.03; for example, 44836100456. • The signaling point code as per GSM 03.03; for example, 253464. • The MCC/MNC (TADIG, PLMN) for mobile roaming calls; for example, 26201. • MCC = Mobile Country Code, MNC = Mobile Network Code • A name; for example, "HELSINKI": Must be uppercase. • A switchID as set up within a local fixed network structure. Derivation: Optional, only mandatory in case of interconnect records (for intercarrier rating/billing reasons,) or for mobile (roaming) records. **Note:** The switch might not be needed if the trunk names are unique within the total network). Set by the first processor and left unchanged.

Table 1-8 (Cont.) Associated GSM/Wireline Extension Record Fields

Name	Format	Description
TERMINATING_SWITCH_IDENTIFICATION	X(15)	Identifies the MSC or Switch ID handling the termination of the call. In case of mobile roaming calls (GSM), this field contains the MTC-related MCC/MNC. In case of wireline networks, this field contains the secondary switch or is empty. Can be used to normalize the B Number in case of MTC-roaming cause. In case of roaming, this field contains the MCC/MNC. Encoding: Encoded as one of the following according to the requirements of the sender: • The MSISDN of the MSC as per GSM 03.03; for example, 44836100456. • The signaling point code as per GSM 03.03; for example, 253464. • The MCC/MNC (TADIG, PLMN) for mobile roaming calls; for example, 26201 • MCC = Mobile Country Code; MNC = Mobile Network Code • A name; for example, "HELSINKI": Must be uppercase. • A switch ID as set up within a local fixed network structure. Derivation: Optional, only mandatory in case mobile (roaming) records. Set by the first processor and left unchanged.
TRUNK_INPUT	X(15)	Trunk identification, inroute address in network switches. Used for interconnect rating to identify the inroute leg of a call. The inroute leg references a related network operator from which the call was received and how to treat this inroute leg in case of intercarrier rating. Encoding: Must uniquely identify a bundled line trunk: • Within the given ORIGINATING_SWITCH_IDENTIFICATION. • With the global network structure. Derivation: Optional, only mandatory in case of interconnect records (for intercarrier rating/ billing reasons). Set by the first processor and left unchanged.
TRUNK_OUTPUT	X(15)	Trunk identification, outroute address in network switches. Can be used by any interconnect rating processor to identify the outroute leg of a call. The outroute leg references a related network operator to which the call was routed or terminated and how to treat this outroute leg in case of intercarrier rating. Encoding: Must uniquely identify a bundled line trunk: • Within the given TERMINATING_SWITCH_IDENTIFICATION. • With the global network structure. Derivation: Optional, only mandatory in case of interconnect records. Set by the first processor and left unchanged.

Table 1-8 (Cont.) Associated GSM/Wireline Extension Record Fields

Name	Format	Description
LOCATION_AREA_INDICATOR	X(10)	Identifies the MSC responsible for handling the call and the location of the equipment making or receiving the call. The definition of these items can be found in the Data Dictionary under MSC Identification, Location Area, and Cell Id. Can be used to map to a specific internal service code to implement a event-dependent rating. Condition: Is not available if not supported by the network or the call does not terminate at the equipment; for example, in call forwarding cases. Values: The Location Area Code is a two-octet string as defined in TS GSM 04.08. For the TAP, the octets are converted to a decimal number in the range 0x00000000 to 0xFFFFFFFF. Derivation: Optional. From the GSM item locationAreaCode as defined in TS GSM 12.05 or directly taken from the sender (VAS). Set by the first processor and left unchanged.
CELL_ID	X(10)	The cell from which the call originated. Can be used to identify the location of the caller. Condition: Operators might not transfer the cell identity. Only available if the call originates or terminates from a mobile phone; for example, not available in call divert cases. Values: The cell identity is a two-octet string as defined in TS GSM 04.08. However, an original hex value is copied. Derivation: Optional. From the GSM item Cell Id as defined in TS GSM 12.05. Set by the first processor and left unchanged.
MS_CLASS_MARK	9(1)	The power capability of the equipment making or receiving the call. Mobiles and transmobiles usually have class 2 capability, handhelds class 4, and PCN applications class 5. Some transmobiles have reduced capability and are classified as class 3. Usually not used. Condition: Only available if supported by the network and the call originates or terminates from the equipment. Is not available in call forwarding cases. Values: 1. Class Mark 2 2. Class Mark 3 3. Class Mark 4 4. Class Mark 5 Other values might apply according to the related original input format. Derivation: Optional. From the GSM item msclassmark as defined in TS GSM 12.05. Set by the first processor and left unchanged.

Table 1-8 (Cont.) Associated GSM/Wireline Extension Record Fields

Name	Format	Description
TIME_BEFORE_ANSWER	9(5)	The number of seconds until a call was successfully established, defined by the time between the call setup attempt and call answer. Can be used as a QoS parameter. Values: Minimum: 00000 Maximum: 99999 Derivation: Optional. Set by the first processor and left unchanged.
BASIC_AoC_AMOUNT_VALUE	9(11)	A monetary amount assigned to the event by any rating processor and charged to the recipient of the file. This does not include any surcharges. Used for roaming or interconnect rating. Can be used to keep the original purchase charge, to evaluate a record based margin together with the charged amount value. Values: Space: No price given, like NULL in a database Floating point format: Given value, might be 0.000. If no floating point exists, the last three digits are always taken as decimals) Minimum: -9999999999 Maximum: 99999999999 Example: '00000012500' for 12,50 '-0001200100' for -1.200,10 '00000012.50' for 12,50 '00012.50000' for 12,50 Derivation: Optional. Usually handed over by the sender of the file, but might also be recalculated by any processor to represent the purchase charge.
BASIC_AoC_AMOUNT_CURRENCY	X(3)	See RETAIL_CHARGED_AMOUNT_CURRENCY.

Table 1-8 (Cont.) Associated GSM/Wireline Extension Record Fields

Name	Format	Description
ROAMER_AoC_AMOUNT_VALUE	9(11)	A monetary amount assigned to the event by any rating processor and charged to the recipient of the file. This is typically a special add-on or surcharge. **Note:** The total wholesale charge of a roaming event should be calculated as: Basic AoC Amount + Roamer AoC Amount Used for roaming and interconnect rating. Can be used to keep the original purchase charge, to evaluate a record based margin together with the charged amount value. Values: Space: No price given, like NULL in a database Floating point format: Given value, might be 0.000. If no floating point exists the last 3 digits are always taken as decimals) Minimum: -999999999 Maximum: 99999999999 Example: '00000012500' for 12,50 '-0001200100' for -1.200,10 '00000012.50' for 12,50 '00012.50000' for 12,50 Derivation: Optional. Usually handed over by the sender (origin network operator) of the file, but might also be recalculated by any processor to represent the purchase charge.
ROAMER_AoC_AMOUNT_CURRENCY	X(3)	See RETAIL_CHARGED_AMOUNT_CURRENCY.
NUMBER_OF_SUPPLEMENTARY_SERVICE_PACKETS	9(2)	Defines the number of Supplementary Service Records following these base fields. For example, 05 means that 5 records are following. Can be used to evaluate how the record structure continues. Values: 00 - 99: Either zero or N records are following Derivation: Mandatory. Dependent on the input how many supplementary service records are present.
NUMBER_OF_BS_PACKETS	Integer	Defines the number of Basic Service Records following the Supplementary Service Record. Values: Default = 0 Derivation: Mandatory. Dependent on the number of basic service records present.
SERVING_NETWORK	String	Conditional in TAP 3.10 files. Indicates the network in which the call event was originally created. This field is a unique identifier.
B_CELL_ID	X(10)	Cell ID of the B party receiving the call. Derivation: Optional.

Table 1-8 (Cont.) Associated GSM/Wireline Extension Record Fields

Name	Format	Description
A_TERM_CELL_ID	X(10)	Cell ID of the A party when the call terminated. Derivation: Optional.
CALL_REFERENCE	String	CallReference item. Optional

Supplementary Service Event Record (RECType 520)

This optional record is used for all non-call related supplementary service actions. The information attributable to a supplementary service event includes basic event information, location information, equipment information, and details of the supplementary service used.

The record applies only to mobile calls (GSM). Derived from the GSM item parameters as defined in TS GSM 12.05.

Table 1-9 describes the fields in the Supplementary Service Event Record.

Table 1-9 Supplementary Service Event Record Fields

Name	Format	Description
RECORD_TYPE	String	Extended to be 3 bytes long. Value: 620 GSM/Wireline Extension Record Derivation: Mandatory. Set by the first processor and left unchanged.
RECORD_NUMBER	9(9)	Sequence number of record in file. Can be used to ensure a linear sequence order for all records; for example, as a sorting criteria. Values: Minimum: 000000002 Maximum: 999999998 Derivation: Mandatory. Set by the first processor. **Important:** Following modules might change this record number; for example, if new record types are inserted.

Table 1-9 (Cont.) Supplementary Service Event Record Fields

Name	Format	Description
ACTION_CODE	H(1)	Qualifies the way in which the supplementary service is used. Values: 0: Registration 1: Erasure 2: Activation 3: Deactivation 4: Interrogation 5: Invocation 6: Registration of Password 9: Switch related Other values might apply according to the related original input format.

Table 1-9 (Cont.) Supplementary Service Event Record Fields

Name	Format	Description
SS_EVENT	H(2)	Uniquely defines the supplementary service or a group of supplementary services. Values: 00: All supplementary services 10: All line identification services 11: Calling number identification presentation 12: Calling number identification restriction 13: Connected number identification presentation 14: Connected number identification restriction 15: Malicious Call Identification 20: all call forwarding 21: Call forwarding unconditional 28: All conditional Call Forwarding 29: Call forwarding on mobile subscriber busy 2A: Call forwarding on no reply 2B: Call forwarding on subscriber not reachable 30: All call offering services 31: Call transfer 32: Mobile Access Hunting 40: all call completion services 41: Call waiting 42: Call hold 43: Completion of calls to busy subscribers 50: All multiparty services 51: multiparty service 60: All community of interest services 61: closed user groups 70: all charging supplement services 71: Advice of charge (charging) 72: Advice of charge (information) 80: All additional info transfer services 81: User to user signaling 90: All call barring 91: All Barring of outgoing Call Services 92: Barring of all outgoing calls 93: Barring of all outgoing international calls 94: Barring of all OG international except HPLMN 99: All Barring of incoming Call Services

Table 1-9 (Cont.) Supplementary Service Event Record Fields

Name	Format	Description
SS_EVENT (contd)	N/A	Uniquely defines the supplementary service or a group of supplementary services (contd). 9A: Barring of all incoming calls 9B: Barring of all IC calls when outside HPLMN all Switch related Services: 01...59: see related switch documentation (values used 1:1) Other values might apply according to the related original input format.
SS_PARAMETERS	String	Optional.
THIRD_PARTY_NUMBER	String	Optional.
CLIR_INDICATOR	Integer	Optional.
CHARGING_START_TIMESTAMP	Date	Optional.
CHARGING_END_TIMESTAMP	Date	Optional.
UTC_END_TIME_OFFSET	X(5)	Timezone where the call terminated. Derivation: Optional.
BASIC_SERVICE_CODE_LIST	String	Optional.

Associated Roaming Extension Record

Table 1-10 lists the fields in the Associated Roaming Extension Record.

Table 1-10 Associated Roaming Extension Record Fields

Name	Format	Description
RECORD_TYPE	String	None
RECORD_NUMBER	Integer	None
TAP_FILE_SEQ_NO	Integer	None
RAP_FILE_SEQ_NO	Integer	None
RAP_RECORD_TYPE	String	None
SENDER	String	None
RECIPIENT	String	None
TAP_FILE_PATH	String	None
START_MISSING_SEQ_NUM	Integer	None
END_MISSING_SEQ_NUM	Integer	None
SUSPENSION_TIME	Date	None
PORT_NUMBER	String	None
TOTAL_TAX_REFUND	Decimal	None

Table 1-10 (Cont.) Associated Roaming Extension Record Fields

Name	Format	Description
TOTAL_DISCOUNT_REFUND	Decimal	None
GUARANTEED_BIT_RATE	String	None
MAXIMUM_BIT_RATE	String	None
HSCSD_INDICATO	String	None
SMS_ORIGINATOR	String	None
SMS_DESTINATION_NUMBER	String	None
DISCOUNTABLE_AMOUNT	Decimal	None
DISCOUNT_CODE	Integer	None
NETWORKACCESS_IDENTIFIE	String	None
ISM_SIGNALLING_CONTEXT	Integer	None
IMSI	String	None
HOME_BID	String	None
HOMELOCATION_DESCRIPTION	String	None
MOBILE_ID_NUMBER	String	None
MOBILE_DIR_NUMBER	String	None
TOTAL_ADVISEDCHARGE	Decimal	None
TOTAL_ADVISEDCHARGE_REFUND	Decimal	None
TOTAL_COMMISSION	Decimal	None
TOTAL_COMMISSION_REFUND	Decimal	None
ITEM_OFFSET	Integer	None
ERROR_CODE	Integer	None
TOTAL_SEVERE_RETURN_VALUE	Decimal	None
RETURN_DETAILS_COUNT	Integer	None
CLIR_INDICATOR	String	None

Associated RAP Extension Record

Table 1-11 lists the fields in the Associated RAP Extension Record.

Table 1-11 Associated RAP Extension Record Fields

Name	Format	Description
PATH_ITEMID	Integer	None
ITEM_OCCURRENCE	Integer	None
ITEM_LEVEL	Integer	None

Basic Service Event Record (RECType 520)

This optional record is used to store related TAP data.

The record applies only to mobile calls (GSM). Derived from the GSM item parameters as defined in TS GSM 12.05.

Table 1-12 lists the fields in the Basic Service Event Record.

Table 1-12 Basic Service Event Record Fields

Name	Format	Description
RECORD_TYPE	String	Extended to be 3 bytes long. Value: 520 GSM/Wireline Extension Record Derivation: Mandatory. Set by the first processor and left unchanged.
RECORD_NUMBER	9(9)	Sequence number of record in file. Can be used to ensure a linear sequence order for all records; for example, as a sorting criteria. Values: Minimum: 000000002 Maximum: 999999998 Derivation: Mandatory. Set by the first processor. **Important:** Following modules might change this record number; for example, if new record types are inserted.
CHAIN_REFERENCE	String	Mandatory.
LONG_DURATION_INDICATOR	String	Mandatory.
BASIC_SERVICE	String	None
QOS_REQUESTED	String	None
QOS_USED	String	None
CHARGING_START_TIMESTAMP	Date	None
CHARGING_END_TIMESTAMP	Date	None
UTC_TIME_OFFSET	String	None
NUMBER_OF_UNITS	Decimal	None
WHOLESALE_CHARGED_AMOUNT_VALUE	Decimal	None
WHOLESALE_CHARGED_TAX_RATE	Integer	None
WHOLESALE_CHARGED_TAX_VALUE	Decimal	None
SPEECH_VERSION_REQUESTED	String	None
SPEECH_VERSION_USED	String	None
TRANSPARENCY_INDICATOR	String	None
FNUR	String	None
AIUR_REQUESTED	String	None
USER_PROTOCOL_INDICATOR	Integer	None
DATA_VOLUME_REFERENCE	String	None

Most-Called Information

This block contains the aggregated amount, duration, and occurrences of the most-called numbers. The number will be listed in the LIST attribute. The values listed in Table 1-13 can be used in EVAL expressions to give discounts based on most-called numbers.

Table 1-13 Most-Called Information Fields

Name	Format	Description
AMOUNT	Decimal	Aggregated amount.
COUNT	Decimal	Aggregated occurrences.
LIST	Integer	Number.
QUANTITY	String	Aggregated duration.

HSCSD Information Packet Record

This optional record is used to store related TAP data.

The record applies only to mobile calls (GSM). Derived from the GSM item parameters as defined in TS GSM 12.05.

High Speed Circuit Switched Data allows users subscribing to the General Bearer Service to use higher transmission rates by using multiple traffic channels simultaneously. This group element must contain Basic HSCSD parameters as at call setup and may also contain details of changes to those parameters.

Table 1-14 lists the fields in the HSCSD Information Packet Record.

Table 1-14 HSCSD Information Packet Record

Name	Format	Description
NUMBER_OF_CHANNELS	String	NumberOfChannels item. Mandatory.
CHANNEL_CODING_OK_LIST	Integer	ChannelCodingAcceptable list (comma-separated integers). Mandatory.
CHANNEL_CODING_USED	Integer	ChannelCoding item. Mandatory.
NUMBER_OF_CHANNELS_USED	Integer	NumberOfChannelsUsed item. Mandatory.
PM_LIST	Block	Optional. HSCSDParameterModification list.
AIUR	Integer	AiurRequested item.
MAX_NUMBER_OF_CHANNELS	Integer	NumberOfChannels item. Optional.
CHANNEL_CODING_USED	Integer	ChannelCoding item. Mandatory.

Table 1-14 (Cont.) HSCSD Information Packet Record

Name	Format	Description
NUMBER_OF_CHANNELS_USED	Integer	NumberOfChannelsUsed item. Mandatory.
INITIATING_PARTY	Integer	InitiatingParty item. Mandatory.
MODIFICATION_TIMESTAMP_	Date	ModificationTimestamp item. Mandatory.
UTC_TIME_OFFSET	String	NumberOfChannels item.

Associated GPRS Extension Record (RECType 540)

This record stores GPRS service information. This record is optional and will be generated only if the related Basic Detail Record indicates a GPRS service.

Table 1-15 describes the fields in the Associated GPRS Extension Record.

Table 1-15 Associated GPRS Extension Record Fields

Name	Format	Description
RECORD_TYPE	String	Extended to be 3 bytes long. Value: 540 Associated GPRS Extension Record
RECORD_NUMBER	9(9)	Sequence number of record in file. Can be used to ensure a linear sequence order for all records; for example, as a sorting criteria. Derivation: Mandatory. Set by the first processor. **Important:** Following modules might change this record number; for example, if new record types are inserted.
PORT_NUMBER	X(24)	Identifies the customer IMSI or SIM number. Option: For Value Added Services and APLMN Service Center Usage, either the IMSI or MSISDN might be supplied, although one of them must be supplied and, where available, the IMSI is preferred. For normal mobile calls, the SIM number is preferred. Derivation: Optional. From the GSM item served IMSI as defined in TS GSM 12.05. Defined in TS GSM 03.03.

Table 1-15 (Cont.) Associated GPRS Extension Record Fields

Name	Format	Description
DEVICE_NUMBER	X(24)	Identifies the equipment used by the subscriber during the call; for example, the International Mobile Equipment Identity number (IMEI). Condition: Not present where the terminal equipment is not involved in the call; for example, in forwarded call cases. It is not mandatory for the VPLMN to transfer this information. Derivation: Optional. From the GSM item IMEI as defined in TS GSM 12.05. Defined in TS GSM 03.03. **Note:** Even though the IMEI is 16 digits in length, the check digit is not transmitted.
A_NUMBER_USER	X(40)	The customer who owns the number from which the call was originated, for terminated calls. Not used for rating, but could be used on invoices. Condition: There is no calling number present where it is unavailable. Could be different from the A Number; for example, in case of VPN calls. For VPN calls, the A Number contains the party to be billed, and this field contains the user initiating the call. Values: See A_NUMBER. Derivation: Optional. From the GSM item Calling Number as defined in TS GSM 12.05. This item is of type Address String and is further expanded into the items type of number, numbering plan and the number sent across the air-interface as defined in TS GSM 04.08 and 09.02 or in international notation. Set by the first processor and left unchanged.
DIALED_DIGITS	X(40)	The number dialed by the customer when establishing a call or the number to which the call is forwarded or transferred. Can be used for managing disputes. Condition: There might be no called number for the basic service emergency call but operators might optionally insert the digits 112 or their national emergency number into this field. The notation should always be local; for example, 04106768124. Values: See B_NUMBER. Derivation: Optional. From the GSM item Calling Number as defined in TS GSM 12.05. This item is of type Address String and is further expanded into the items type of number, numbering plan and the number sent across the air-interface as defined in TS GSM 04.08 and 09.02 or in international notation. Set by the first processor and left unchanged.

Table 1-15 (Cont.) Associated GPRS Extension Record Fields

Name	Format	Description
VAS/PRODUCT_CODE	X(10)	A classification of Value Added Services as generated by the sender. Can be used to map to a specific internal service code to implement specific usage scenarios for any rating purposes. Values: VMAIL: Voice Mail Services SEC: Secretarial Services OPER: Telephonic Operator Services FI: Financial Information TRAVEL: Travel Information This is not a definitive list and might be added to through MoU-TADIG from time to time or might be user defined. Derivation: Optional. From the GSM item vasCode as defined in GSM TD17. Set by the first processor and left unchanged.
ORIGINATING_SWITCH_IDENTIFICATION	X(15)	Identifies the MSC or SwitchID handling the origin of the call. In case of mobile roaming calls (GSM), this field contains the MOC-related MCC/MNC. In case of wireline networks, this field contains the primary switch that generated this CDR. Can be used by any interconnect rating processor to uniquely identify the trunk names but will only be used if the trunk names are only unique within the related switch. See TRUNK_INPUT and TRUNK_OUTPUT. Can also be used to normalize the A Number for MOC roaming. In case of roaming, this field contains the MCC/MNC. Encoding: Encoded as one of the following according to the requirements of the sender: • The MSISDN of the MSC as per GSM 03.03; for example, 44836100456. • The signaling point code as per GSM 03.03; for example, 253464. • The MCC/MNC (TADIG, PLMN) for mobile roaming calls; for example, 26201. • MCC = Mobile Country Code; MNC = Mobile Network Code • A name; for example, "HELSINKI": Must be uppercase. • A switchID as set up within a local fixed network structure. Derivation: Optional, only mandatory in case of interconnect records (for intercarrier rating/billing reasons,) or for mobile (roaming) records. Set by the first processor and left unchanged. The switch might not be needed if the trunk names are unique within the total network).

Table 1-15 (Cont.) Associated GPRS Extension Record Fields

Name	Format	Description
TERMINATING_SWITCH_IDENTIFICATION	X(15)	Identifies the MSC or Switch ID handling the termination of the call. In case of mobile roaming calls (GSM), this field contains the MTC-related MCC/MNC. In case of wireline networks, this field contains the secondary switch or is empty. Can be used to normalize the B Number in case of MTC-roaming cause. In case of roaming, this field contains the MCC/MNC. Encoding: Encoded as one of the following according to the requirements of the sender: • The MSISDN of the MSC as per GSM 03.03; for example, 44836100456. • The signaling point code as per GSM 03.03; for example, 253464. • The MCC/MNC (TADIG, PLMN) for mobile roaming calls; for example, 26201. • MCC = Mobile Country Code; MNC = Mobile Network Code • A name; for example, "HELSINKI": Must be uppercase. • A switch ID as set up within a local fixed network structure. Derivation: Optional, only mandatory in case of mobile (roaming) records. Set by the first processor and left unchanged.
MS_CLASS_MARK	9(1)	The power capability of the equipment making or receiving the call. Mobiles and transmobiles usually have class 2 capability, handhelds class 4, and PCN applications class 5. Some transmobiles have reduced capability and are classified as class 3. Usually not used. Condition: Only available if supported by the network and the call originates or terminates from the equipment. Is not available in call forwarding cases. Values: 1. Class Mark 2 2. Class Mark 3 3. Class Mark 4 4. Class Mark 5 Other values might apply according to the related original input format. Derivation: Optional. From the GSM item msclassmark as defined in TS GSM 12.05. Set by the first processor and left unchanged.
ROUTING_AREA	X(10)	Routing Area at the time of record creation (S-CDR only).

Table 1-15 (Cont.) Associated GPRS Extension Record Fields

Name	Format	Description
LOCATION_AREA_INDICATOR	X(10)	Identifies the MSC responsible for handling the call and the location of the equipment making or receiving the call. The definition of these items can be found in the Data Dictionary under MSC Identification, Location Area, and Cell Id. Can be used to map to a specific internal service code to implement a event-dependent rating. Condition: Is not available if not supported by the network or the call does not terminate at the equipment; for example, in call forwarding cases. Values: The Location Area Code is a two-octet string as defined in TS GSM 04.08. For the TAP, the octets are converted to a decimal number in the range 0x00000000 to 0xFFFFFFFF. Derivation: Optional. From the GSM item locationAreaCode as defined in TS GSM 12.05 or directly taken from the sender (VAS). Set by the first processor and left unchanged.
CHARGING_ID	Decimal	PDP context identifier used to identify this PDP context in different records created by GSNs. This field is a charging identifier which can be used together with GGSN address to identify all records produced in SGSN(s) and GGSN involved in a single PDP context. Charging ID is generated by GGSN at PDP context activation and transferred to context requesting SGSN. At inter-SGSN routing area update, charging ID is transferred to the new SGSN as part of each active PDP context. Different GGSNs allocate the charging ID independently of each other and might allocate the same numbers. The CGF and/or BS might check the uniqueness of each charging ID together with the GGSN address and optionally (if still unambiguous) with the record opening timestamp.
SGSN_ADDRESS	X(64)	Current SGSN Address used. Optional.
GGSN_ADDRESS	X(64)	IP Address of the GGSN currently used. Optional.
WLAN_ADDRESS	String	Optional.
APN_ADDRESS	X(64)	The logical name of the connected access point to the external packet data network. APN comprises network identifier and operator identifier. This field contains the logical Access Point Name used to determine the actual connected access point. APN comprises network identifier and operator identifier. APN can also be a wildcard, in which case SGSN selects the access point address. See GSM 03.03 [4] and GSM 03.60 [8] for more information about APN format and access point decision rules.
NODE_ID	X(64)	Name of the recording entity; for example, could be the charging gateway name.
TRANS_ID	9(10)	Sequence number which the recording entity generates (NODE_ID). The number is allocated sequentially including all CDR types. It links together the CDR of a same recording entity.
SUB_TRANS_ID	9(10)	Partial record sequence number. This field contains a running sequence number which links the partial records generated for a PDP context/GPRS session. It can be used in post-processing to detect missing CDRs for a GPRS session. It links together the CDRs/events of a same session.

Table 1-15 (Cont.) Associated GPRS Extension Record Fields

Name	Format	Description
NETWORK_INITIATED_PDP	9(1)	Network Initiated PDP context. The network initiates a context when it calls an ME. Values: 0: False 1: True
PDP_TYPE	X(4)	Defines the PDP type; for example, X.25, IP, PPP, or IHOSS:OSP (see GSM 09.60 for exact format).
PDP_ADDRESS	X(64)	PDP address of the served IMSI (Ipv4, IPv6, X.121).
PDP_REMOTE_ADDRESS	X(255)	List of PDP address of remote host (comma-separated value, G-CDR only, X25 only).
PDP_DYNAMIC_ADDRESS	9(1)	Indicates that the PDP address has been dynamically allocated for that particular PDP context. This field is missing if address is static; for example, part of PDP context subscription. Dynamic address allocation might be relevant for charging; for example, the duration of PDP context as one balance element offered and possibly owned by network operator. Values: 0: False 1: True
DIAGNOSTICS	X(255)	Includes a more detailed technical reason for the release of the connection and might contain one of the following: • A MAP error from GSM 09.02 [17] • A Cause from GSM 04.08 [16] The diagnostics might also be extended to include manufacturer and network-specific information.
CELL_ID	X(10)	The cell from which the call originated. Can be used to identify the location of the caller. Condition: Operators might not transfer the cell identity. Only available if the call originates or terminates from a mobile phone; for example, not available in call divert cases. Values: The cell identity is a two-octet string as defined in TS GSM 04.08. However, an original hex value is copied. Derivation: Optional. From the GSM item Cell Id as defined in TS GSM 12.05. Set by the first processor and left unchanged.
CHANGE_CONDITION	9(1)	The condition that triggers the creation of this volume container as defined by ETSI. Values: 0: Quality of Service Change 1: Tariff Change 2: Record Closed

Table 1-15 (Cont.) Associated GPRS Extension Record Fields

Name	Format	Description
QoS_REQUESTED_PRECEDENCE	X(1)	The priority applicable to a GPRS connection. Condition: Mandatory within groups GSM Quality Of Service Requested. Values: 0: Unspecified 1: High Priority 2: Normal Priority 3: Low Priority Derivation: GSM item QoS Precedence (GSM 12.15).
QoS_REQUESTED_DELAY	X(1)	The transfer delay applicable to a GPRS connection. Condition: Mandatory within groups GSM Quality Of Service Requested. Values: 0: Delay class 1 1: Delay class 2 2: Delay class 3 3: Delay class 4 Derivation: GSM item QoSDelay (GSM 12.15).
QoS_REQUESTED_RELIABILTY	X(1)	The reliability applicable to a GPRS connection. Condition: Mandatory within groups GSM Quality Of Service Requested. Values: 0: Unspecified Reliability 1: Acknowledged GTP 2: Unacknowledged GTP/acknowledged LLC 3: Unacknowledged GTP/ acknowledged RLC 4: Unacknowledged GTP/LLC/RLC 5: Unacknowledged unprotected data Derivation: GSM item QoS Reliability (GSM 12.15).

Table 1-15 (Cont.) Associated GPRS Extension Record Fields

Name	Format	Description
QoS_REQUESTED_PEAK_THROUGHPUT	X(2)	The peak throughput applicable to a GPRS connection. Condition: Mandatory within groups GSM Quality Of Service Requested. Values: 0: Unspecified 1: Up to 100 octets per seconds 2: Up to 200 octets per seconds 3: Up to 400 octets per seconds 4: Up to 800 octets per seconds 5: Up to 1600 octets per seconds 6: Up to 3200 octets per seconds 7: Up to 6400 octets per seconds 8: Up to 12800 octets per seconds 9: Up to 25600 octets per seconds Derivation: GSM item QoS Peak Throughput (GSM 12.15).
QoS_REQUESTED_MEAN_THROUGHPUT	X(2)	The mean throughput applicable to a GPRS connection. Condition: Mandatory within groups GSM Quality Of Service Requested. Values: 0: Best Effort 1: Mean 100 octets per hour 2: Mean 200 octets per hour 3: Mean 500 octets per hour 4: Mean 1000 octets per hour 5: Mean 2000 octets per hour 6: Mean 5000 octets per hour 7: Mean 10000 octets per hour 8: Mean 20000 octets per hour 9: Mean 50000 octets per hour 10: Mean 100000 octets per hour 11: Mean 200000 octets per hour 12: Mean 500000 octets per hour 13: Mean 1000000 octets per hour 14: Mean 2000000 octets per hour 15: Mean 5000000 octets per hour 16: Mean 10000000 octets per hour 17: Mean 20000000 octets per hour 18: Mean 50000000 octets per hour Derivation: GSM item QoS Mean Throughput (GSM 12.15).
QoS_USED_PRECEDENCE	X(1)	Quality of Service Precedence class. See QoS_REQUESTED_PRECEDENCE.

Table 1-15 (Cont.) Associated GPRS Extension Record Fields

Name	Format	Description
QoS_USED_DELAY	X(1)	QOS delay class, defined by ETSI. See QoS_REQUESTED_DELAY.
QoS_USED_RELIABILTY	X(1)	QOS reliability class, defined by ETSI. See QoS_REQUESTED_RELIABILTY.
QoS_USED_PEAK_THROUGHPUT	X(2)	QOS peak throughput class, defined by ETSI. See QoS_REQUESTED_PEAK_THROUGHPUT.
QoS_USED_MEAN_THROUGHPUT	X(2)	QOS mean throughput class, defined by ETSI. See QoS_REQUESTED_MEAN_THROUGHPUT.
NETWORK_CAPABILITY	X(10)	MS network capability information element of the served MS on PDP context activation or on GPRS attachment as defined in GSM 04.08 [16]. Condition: Optional. Derivation: GSM item network capability (GSM 04.08 [16]).
SGSN_CHANGE	9(1)	Indicates that this is the first record after an inter-SGSN routing area update. Condition: Mandatory. Values: 0: default, if this is not the 1st record 1: indicates the first record after an inter SGSN-change
START_SEQUENCE_NO	String	Optional.
END_SEQUENCE_NO	String	Optional.
B_CELL_ID	X(10)	Cell ID of the B party receiving the call. Derivation: Optional.
A_TERM_CELL_ID	X(10)	Cell ID of the A party when the call terminated. Derivation: Optional.
PDP_CONTEXT_START_TIMESTAMP	Date	Conditional in TAP 3.10. Indicates the start time of the PDP context when the Call Event Details (GPRS Call) represents an intermediate or last partial of a PDP context. Used in TAP files. Format: *CCYYMMDDHHMMSS*
PDP_UTC_TIME_OFFSET	String	Conditional in TAP 3.10. Indicates the UTC time offset for PDP_CONTEXT_START_TIMESTAMP.

Table 1-15 (Cont.) Associated GPRS Extension Record Fields

Name	Format	Description
SERVICE_USED_CHARGING_START_TIMESTAMP	Date	Conditional in TAP 3.10. Indicates the start time for charging GPRS calls. This field is present when the value is not the same as the associated Call Event Start Timestamp field (DETAIL.ASS_GPRS_EXT.GS_PACKET.CHARGING_START_TIMESTAMP). Used in TAP files. Format: *CCYYMMDDHHMMSS*
SERVICE_USED_UTC_TIME_OFFSET	String	Conditional in TAP 3.10. Indicates the UTC time offset for SERVICE_USED_CHARGING_START_TIMESTAMP.
TYPE_OF_CONTROLLING_NODE	Integer	Conditional in TAP 3.10.
GPRS_SERVICE_USAGE_PACKET	Block	*n* times. Optional. Mandatory.
CHARGING_START_TIMESTAMP	Date	Optional.
CHARGING_END_TIMESTAMP	Date	Optional.
UTC_TIME_OFFSET	String	Optional.
QOS_REQUESTED_PRECEDENCE	String	Optional.
QOS_REQUESTED_DELAY	String	Optional.
QOS_REQUESTED_RELIABILITY	String	Optional.
QOS_REQUESTED_PEAK_THROUGHPUT	String	Optional.
QOS_REQUESTED_MEAN_TRHOUGHPUT	String	Optional.
QOS_USED_PRECEDENCE	String	Optional.
QOS_USED_DELAY	String	Optional.
QOS_USED_RELIABILITY	String	Optional.
QOS_USED_PEAK_TRHOUGHPUT	String	Optional.
QOS_USED_MEAN_TRHOUGHPUT	String	Optional.
VOLUME_RECEIVED	Decimal	Mandatory.
VOLUME_SENT	Decimal	Mandatory.

Table 1-15 (Cont.) Associated GPRS Extension Record Fields

Name	Format	Description
UMTS_QOS_REQUESTED	String	Optional. Identifies the UMTS Quality of Service requested for GPRS calls. Used in TAP files. Format: This field contains comma-separated tag-value pairs of the following TAP fields with their respective tags as shown below: • QoS Traffic Class: 1 • QoS Max Bit Rate Uplink: 2 • Qos Max Bit Rate Downlink: 3 • Qos Guaranteed Bit Rate Downlink: 4 • Qos Guaranteed Bit Rate Uplink: 5 • Qos Allocation Retention Priority: 6 The fields QoS Traffic Class, QoS Max Bit Rate Uplink, Qos Max Bit Rate Downlink are Mandatory. The others are optional. Example 1: If the TAP field values are as follows: • QoS Traffic Class: **3** • QoS Max Bit Rate Uplink: **63** • Qos Max Bit Rate Downlink: **128** • Qos Guaranteed Bit Rate Downlink: **61** • Qos Guaranteed Bit Rate Uplink: **250** • Qos Allocation Retention Priority: **3** The value of the EDR field is **1,3,2,63,3,128,4,61,5,250,6,3** Example 2: If the TAP field values are as follows: QoS Traffic Class: **2** QoS Max Bit Rate Uplink: **56** Qos Max Bit Rate Downlink: **128** Qos Guaranteed Bit Rate Uplink: **250** The value of the EDR field is: **1,2,2,56,3,128,5,250**
UMTS_QOS_USED	String	Optional. Identifies the UMTS Quality of Service used for GPRS calls. Used in TAP files. The description for this field is identical to the description for UMTS_QOS_REQUESTED.

Associated WAP Extension Record (RECType 570)

Stores information for WAP events. This record is optional and will only be generated if the related Basic Detail Record indicates a WAP service.

Table 1-16 describes the fields in the Associated WAP Extension Record.

Table 1-16 Associated WAP Extension Record Fields

Name	Format	Description
RECORD_TYPE	String	Extended to be 3 bytes long. Value: 570 Associated WAP Extension Record
RECORD_NUMBER	9(9)	Sequence number of record in file. Can be used to ensure a linear sequence order for all records; for example, as a sorting criteria. Derivation: Mandatory. Set by the first processor. **Important:** Following modules might change this record number; for example, if new record types are inserted.
PORT_NUMBER	X(24)	Identifies the customer IMSI or SIM number. Option: For Value Added Services and APLMN Service Center Usage either the IMSI or MSISDN might be supplied, although one of them must be supplied and, where available, the IMSI is preferred. For normal mobile calls, the SIM number is preferred. Derivation: Optional. From the GSM item served IMSI as defined in TS GSM 12.05. Defined in TS GSM 03.03.
DEVICE_NUMBER	X(24)	Identifies the equipment used by the subscriber during the call; for example, the International Mobile Equipment Identity number (IMEI). Condition: Not present where the terminal equipment is not involved in the call; for example, in forwarded call cases. It is not mandatory for the VPLMN to transfer this information. Derivation: Optional. From the GSM item IMEI as defined in TS GSM 12.05. Defined in TS GSM 03.03. **Note:** Even though the IMEI is 16 digits in length, the check digit is not transmitted.
SESSION_ID	X(64)	Session ID as provided by the WAP gateway.
RECORDING_ENTITY	X(64)	Name of the recording Entity; for example, the WAP gateway or mediation device.
TERMINAL_CLIENT_ID	X(64)	The served WAP terminal client ID (WAP gateway user identity).
TERMINAL_IP_ADDRESS	X(64)	IP address of the WAP terminal.
DOMAIN_URL	X(255)	URL implementing the service.
BEARER_SERVICE	X(3)	See BASIC_SERVICE.
BEARER_SERVICE_CODE	X(2)	See SERVICE_CODE.

Table 1-16 (Cont.) Associated WAP Extension Record Fields

Name	Format	Description
HTTP_STATUS	9(3)	HTTP status code from the origin server or servlet. Values: 100: CONTINUE 101: SWITCHING_PROTOCOLS 200: SUCCESS 201: CREATE 202: ACCEPTED 203: NON-AUTHORITATIVE_INFORMATION 204: NO_CONTENT 205: RESET_CONTENT 206: PARTIAL_CONTENT 300: MULTIPLE_CHOICE 301: MOVED_PERMANENTLY 302: FOUND 303: SEE_OTHER 304: NOT_MODIFIED 305: USE_PROXY 307: TEMPORARY_REDIRECT 400: BAD_REQUEST 401: UNAUTHORIZED 402: PAYMENT_REQUIRED 403: FORBIDDEN 404: NOT_FOUND 405: METHOD_NOT_ALLOWED 406: NOT_ACCEPTABLE 407: PROXY_AUTHENTICATION_REQUIRED 408: REQUEST_TIMEOUT 409: CONFLICT 410: GONE 411: LENGTH_REQUIRED 412: PRECONDITION_FAILED 413: REQUEST_ENTITY_TOO_LARGE 414: REQUEST-URI_TOO_LONG 415: UNSUPPORTED_MEDIA_TYPE 416: REQUESTED_RANGE_NOT_SATISFIABLE 417: EXPECTATION_FAILED 500: INTERNAL_SERVER_ERROR 501: NOT_IMPLEMENTED 502: BAD_GATEWAY 503: SERVICE_UNAVAILABLE 504: GATEWAY_TIMEOUT 505: HTTP_VERSION_NOT_SUPPORTED

Table 1-16 (Cont.) Associated WAP Extension Record Fields

Name	Format	Description
WAP_STATUS	9(3)	The WSP/WAP status code. Values: 16: CONTINUE 17: SWITCHING_PROTOCOLS 20: OK, SUCCESS 33: CREATED 34: ACCEPTED 35: NON-AUTHORITATIVE_INFORMATION 36: NO_CONTENT 37: RESET_CONTENT 38: PARTIAL_CONTENT 48: MULTIPLE_CHOICE 49: MOVED_PERMANENTLY 50: MOVED_TEMPORARILY 51: SEE_OTHER 52: NOT_MODIFIED 53: USE_PROXY 55: TEMPORARY_REDIRECT 64: BAD_REQUEST 65: UNAUTHORIZED 66: PAYMENT_REQUIRED 67: FORBIDDEN 68: NOT_FOUND 69: METHOD_NOT_ALLOWED 70: NOT_ACCEPTABLE 71: PROXY_AUTHENTICATION_REQUIRED 72: REQUEST_TIMEOUT 73: CONFLICT 74: GONE 75: LENGTH_REQUIRED 76: PRECONDITION_FAILED 77: REQUEST_ENTITY_TOO_LARGE 78: REQUEST-URI_TOO_LONG 79: UNSUPPORTED_MEDIA_TYPE 80: REQUESTTED_RANGE_NOT_SATISFIABLE 81: EXPECTATION_FAILED 96: INTERNAL_SERVER_ERROR 97: NOT_IMPLEMENTED 98: BAD_GATEWAY 99: SERVICE_UNAVAILABLE 100: GATEWAY_TIMEOUT 101: HTTP_VERSION_NOT_SUPPORTED

Table 1-16 (Cont.) Associated WAP Extension Record Fields

Name	Format	Description
ACKNOWLEDGE_STATUS	9(1)	Acknowledge status of the response. Values: 1: OK acknowledgment has been received. 2: Response terminated by the server. 3: Response terminated by the terminal. 4: Acknowledgment has not been received. 5: Acknowledgment is not used with this connection type.
ACKNOWLEDGE_TIME	YYYYMMDD HHMISS	Time of the acknowledgment.
EVENT_NUMBER	X(60)	Assigned user event number as generated by the WAP gateway.
GGSN_ADDRESS	X(64)	IP Address of the GGSN currently used.
SERVER_TYPE	X(64)	A description of the type of server providing the service.
CHARGING_ID	Decimal	PDP context identifier used to identify this PDP context in different records created by GSNs. This field is a charging identifier which can be used together with GGSN address to identify all records produced in SGSN(s) and GGSN involved in a single PDP context. Charging ID is generated by GGSN at PDP context activation and transferred to context requesting SGSN. At inter-SGSN routing area update, charging ID is transferred to the new SGSN as part of each active PDP context. Different GGSNs allocate the charging ID independently of each other and might allocate the same numbers. The CGF and/or BS might check the uniqueness of each charging ID together with the GGSN address and optionally (if still unambiguous) with the record opening timestamp.
WAP_LOGIN	X(24)	Login used during the WAP session. This might occur in addition to the MSISDN; for example, this field might contain a user name of a session which has been opened within a WAP session. Condition: Optional. Might be mandatory for specific WAP scenarios. Derivation: Set by the first processor and left unchanged.
IDENTIFIER	String	Mandatory.
TYPE	Integer	Mandatory.

Associated CAMEL Extension Record (RECType 700)

In the following associated record of the sol42 format extended CAMEL service information could be stored. This record is optional and is attached to any other associated service extension record.

Table 1-17 lists the Associated CAMEL Extension Record fields.

Table 1-17 Associated CAMEL Extension Record Fields

Name	Format	Description
RECORD_TYPE	String	Extended to be 3 bytes long Value: 700: Associated CAMEL Extension Record
RECORD_NUMBER	9(9)	Sequence number of record in file. Can be used to ensure a linear sequence order for all records; for example, as a sorting criteria. Derivation: Mandatory. Set by the first processor. **Important:** Following modules might change this record number; for example, if new record types are inserted.
SERVER_TYPE_OF_NUMBER	Z(1)	Optional, but should be defaulted to '0'. Could be used by some modules to perform number-normalization.
SERVER_NUMBERING_PLAN	X(1)	Optional.
SERVER_ADDRESS	X(40)	Identifies the server interrogated. Mandatory.
SERVICE_LEVEL	Z(1)	Identifies the level of CAMEL service provided [0-3]. Mandatory.
SERVICE_KEY	Z(10)	Identifies the CAMEL service Logic to be applied. Mandatory.
DEFAULT_CALL_HANDLING_INDICATOR	Z(1)	Indicates whether or not a CAMEL call encountered default handling. Values: 0: Continue the call 1: Release the call
MSC_TYPE_OF_NUMBER	Z(1)	Optional, but should be defaulted to '0'. Could be used by some modules to perform number-normalization.
MSC_NUMBERING_PLAN	X(1)	Optional.
MSC_ADDRESS	X(40)	Identifies the MSC that generated the CAMEL reference number (might be different from SERVER_ADDRESS). Mandatory.
CAMEL_REFERENCE_NUMBER	X(20)	In association with the MSC_ADDRESS, provides a unique identifier for each CAMEL invocation. Mandatory.
CAMEL_INITIATED_CF_INDICATOR	Z(1)	Optional, but should be defaulted to 0 (1=CAMEL call forwarding).
CAMEL_MODIFICATION_LIST	X(20)	Optional, comma-separated string of integers.
DEST_GSMW_TYPE_OF_NUMBER	Z(1)	Optional, but should be defaulted to '0'. Could be used by some modules to perform number-normalization.
DEST_GSMW_NUMBERING_PLAN	X(1)	Optional.

Table 1-17 (Cont.) Associated CAMEL Extension Record Fields

Name	Format	Description
DEST_GSMW_NUMBER	X(40)	Optional, used to identify CAMEL redirection destination (could contain an MSISDN, IP, LOGIN, etc.) when the primary extension is of type GSM (for example, ASS_GSMW_EXT).
DEST_GSMW_NUMBER_ORIGINAL	X(40)	Optional, DEST_GSM_NUMBER as received (before normalization).
DEST_GPRS_APN_ADDRESS	X(64)	Optional (but might be mandatory for specific zoning scenarios; for example, if an apn_group is used) when the primary extension is of type GPRS (ie., ASS_GPRS_EXT).
DEST_GPRS_PDP_REMOTE_ADDRESS	X(255)	Optional.
CSE_INFORMATION	X(40)	Optional, the information downloaded by the CAMEL server.
GSM_CALL_REFERENCE_NUMBER	X(20)	Optional.
EXCHANGE_RATE	Decimal	Contains the exchange rate which has been used to convert the Incoming currency to the internal currency as indicated in the field CHARGED_CURRENCY_TYPE. Can be used to convert the virtual currency SDR (which is used in conjunction of TAP) to internal currencies and convert the Charge back to SDR after Rating. This would be a typical usage for Interconnection Rating. Values: Variable floating point format: Given value, might be 0.000. The floating decimal point must be set. Minimum: -9999999999 Maximum: 99999999999 Examples: '00000000125' for 125,00 '00000012.50' for 12,50 '-0012.56780' for -12,5678 Derivation: Optional, defaulted 00000000001 (=1,00).

Associated Suspense Extension Record (RECType 720)

Table 1-18 describes the fields in the Associated Suspense Extension Record. This record is optional and can appear once.

Table 1-18 Associated Suspense Extension Record Fields

Name	Format	Description
RECORD_TYPE	String	Mandatory. Default = **720**
RECORD_NUMBER	Integer	Mandatory. Auto-generated.

Table 1-18 (Cont.) Associated Suspense Extension Record Fields

Name	Format	Description
SUSPENSE_STATUS	Integer	Mandatory. Calculated.
SUSPENSE_REASON	Integer	The suspense reason. Mapped from the error code. Mandatory. Calculated.
SUSPENSE_SUBREASON	Integer	Mandatory. Calculated.
RECYCLE_KEY	String	Search key for choosing EDRs to recycle. Optional.
ERROR_CODE	Integer	Mandatory. Calculated.
SUSPENSE_ID	Integer	Original suspense POID ID. Mandatory when recycling.
PIPELINE_NAME	String	The name of the pipeline, derived from the pipeline registry. Mandatory when recycling. Calculated.
SOURCE_FILENAME	String	The source file name. The same as INTERNAL.STREAM_NAME. Mandatory. Calculated.
SERVICE_CODE	String	Equal to DETAIL.INTERN_SERVICE_CODE. Mandatory. Calculated.
EDR_RECORD_TYPE	String	Equal to DETAIL.RECORD_TYPE. Mandatory. Calculated.
EDR_BUF	String	A stored representation of the EDR container including fields overwritten and enriched by the pipeline. Mandatory when recycling. Calculated.
UTC_OFFSET_SECONDS	Integer	This value enables Suspense Management Center to represent call record times using the same time zone used in the records themselves. The value represents the offset between the time zone of the EDR and UTC in seconds. Mandatory. Calculated.
EDR_SIZE	Integer	The size of DETAIL.ASS_SUSPENSE_EXT.EDR_BUF. Mandatory. Calculated.
QUERYABLE_FIELDS	String	The queryable field values defined in the registry. Separated by tab characters. Mandatory. Calculated.
OVERRIDE_REASONS	String	Optional. May be set equal to the override reason code during recycling.
ACCOUNT_POID	String	Optional. Calculated.
SUSPENDED_FROM_BATCH_ID	String	Optional. Calculated.
PIPELINE_CATEGORY	String	Mandatory. Calculated.
RECYCLING_MODE	Integer	Mandatory. Calculated. Equal to DETAIL.INTERN_PROCESS_STATUS

Associated Content Extension Record (RECType 550)

This optional record is used to store related TAP data. Table 1-19 describes the fields in the Associated Content Extension Record.

Table 1-19 Associated Content Extension Record Fields

Name	Format	Description
RECORD_TYPE	String	Mandatory. Default = 550.
RECORD_NUMBER	Integer	Sequence number of record in file. Can be used to ensure a linear sequence order for all records; for example, as a sorting criteria. Derivation: Mandatory. Set by the first processor. **Important:** Following modules might change this record number; for example, if new record types are inserted.
FRAUD_MONITOR_INDICATOR	String	Optional, but should be defaulted to '0'. Could be used by some modules to perform number-normalization.
RAP_FILE_SEQ_NO	String	Optional.
ORDER_PLACED_TIMESTAMP	Date	Optional.
ORDER_PLACED_UTC_TIME_OFFSET	String	Mandatory if ORDER_PLACED_TIMESTAMP is present.
REQUESTED_DELIVERY_TIMESTAMP	Date	Optional.
REQ_DELIVERY_UTC_TIME_OFFSET	String	Mandatory if REQUESTED_DELIVERY_TIMESTAMP is present.
ACTUAL_DELIVERY_TIMESTAMP	Date	Optional.
ACT_DELIVERY_UTC_TIME_OFFSET	String	Mandatory if ACTUAL_DELIVERY_TIMESTAMP is present.
TOTAL_TRANSACTION_DURATION	Integer	Optional.
TRANSACTION_STATUS	Integer	Optional.
CHARGED_PARTY_INFO	Block	Charged party information block. Mandatory.
ID_LIST	N.A.	ChargedPartyId list. Mandatory.
HOMEID_LIST	N.A.	ChargedPartyHomeId list. Optional.
LOCATION_LIST	N.A.	ChargedPartyLocation list. Optional.
EQUIPMENT	N.A.	ChargedPartyEquipment block. Optional.
SERVING_PARTIES_INFO	Block	ServingPartiesInformation block. Mandatory.
PROVIDER_LIST	N.A.	ContentProviderId list. Optional.

Table 1-19 (Cont.) Associated Content Extension Record Fields

Name	Format	Description
ISP_LIST	N.A.	InternetServiceProviderId list. Optional.
NETWORK_LIST	N.A.	Network list. Optional.
SERVICE_USED_LIST	N.A.	ContentServiceUsed list. Mandatory.
CONTENT_TRANSACTION_CODE	Integer	ContentTransactionCode item. Mandatory.
OBJECT_TYPE	Integer	ObjectType item. Optional.
TRANSACTION_DESCRIPTION_SUPP	Integer	TransactionDescriptionSupp item. Optional.
TRANSACTION_DETAIL_DESCRIPTION	String	TransactionDetailDescription item. Optional.
TRANSACTION_IDENTIFIER	String	TransactionIdentifier item. Mandatory.
TRANSACTION_AUTH_CODE	String	TransactionAuthCode item. Optional.
DATA_VOLUME_INCOMING	Integer	DataVolumeIncoming item. Optional.
DATA_VOLUME_OUTGOING	Integer	DataVolumeOutgoing item. Optional.
TOTAL_DATA_VOLUME	Integer	TotalDataVolume item. Optional.
CHARGE_REFUND_INDICATOR	Integer	ChargeRefundIndicator item. Optional.
CONTENT_CHARGING_POINT	Integer	ContentChargingPoint item. Optional.
PAID_INDICATOR	Integer	PaidIndicator item. Optional.
PAYMENT_METHOD	Integer	PaymentMethod item. Optional.
ADVISED_CHARGE_CURRENCY	String	AdvisedChargeCurrency item. Optional.
ADVISED_CHARGE	Decimal	AdvisedCharge item. Optional. Mandatory in the AdvisedChargeInformation block.
COMMISSION	Decimal	Commission item. Optional.

Associated Location Extension Record

The OutGrammar stores information of Content and Location from the EDR container into the output TAP blocks. This is performed using ASN calls of iScript in TAP version 3.10 OutGrammar.

Table 1-20 describes the fields in the Associated Location Extension Record.

Table 1-20 Associated Location Extension Record Fields

Name	Format	Description
RECORD_TYPE	String	Mandatory. Must be set to 560.
RECORD_NUMBER	Integer	Mandatory. Auto-generated.
FRAUD_MONITOR_INDICATOR	String	FraudMonitorIndicator item. Optional.
RAP_FILE_SEQ_NO	String	RapFileSequenceNumber item. Optional.
REC_ENTITY_CODE	Integer	RecEntityCode item. Mandatory.
CALL_REFERENCE	String	CallReference item. Optional.
GMLC_ADDRESS	String	N/A
TRACKING_CUSTOMER_INFORMATION	Block	TrackingCustomerInformation block. Optional.
ID_LIST	N.A.	TrackingCustomerId list. Mandatory.
HOME_ID_LIST	N.A.	TrackingCustomerHomeId list. Mandatory.
LOCATION_LIST	N.A.	TrackingCustomerLocation list. Mandatory.
EQUIPMENT	N.A.	TrackingCustomerEquipment block. Optional.
LCS_SP_INFORMATION LCSSP_INFO	Block	LCSSPInformation block. Optional.
ID_LIST	N.A.	LCSSPId list. Mandatory.
ISP_LIST	N.A.	InternetServiceProviderId list. Optional.
NETWORK_LIST	N.A.	Network list. Optional.
TRACKED_CUSTOMER_INFORMATION	Block	TrackedCustomerInformation block. Optional.
ID_LIST	N.A.	TrackedCustomerId list. Mandatory.

Table 1-20 (Cont.) Associated Location Extension Record Fields

Name	Format	Description
HOME_ID_LIST	N.A.	TrackedCustomerHomeId list. Mandatory.
LOCATION_LIST	N.A.	TrackedCustomerLocation list Mandatory.
EQUIPMENT	N.A.	TrackedCustomerEquipment block. Optional.
LOCATION_SERVICE_USAGE	Block	LocationServiceUsage block. Mandatory.
LCSQosRequested	Block	Mandatory.
LCS_REQUEST_TIMESTAMP	Date	Mandatory.
LCS_REQ_UTC_OFFSET	String	LCSRequestTimestamp item. Mandatory.
H_ACCURACY_REQUESTED	Integer	HorizontalAccuracyRequested item. Optional.
V_ACCURACY_REQUESTED	Integer	VerticalAccuracyRequested item. Optional.
RESPONSE_TIME_CATEGORY	Integer	ResponseTimeCategory item. Optional.
TRACKING_PERIOD	Integer	TrackingPeriod item. Optional.
REQ_TRACKING_FREQUENCY	Integer	TrackingFrequency (requested) item. Optional.
LCSQosDelivered	Block	Optional.
LCS_TRANS_STATUS	Integer	LCSTransactionStatus item. Optional.
H_ACCURACY_DELIVERED	Integer	HorizontalAccuracyDelivered item. Optional.
V_ACCURACY_DELIVERED	Integer	VerticalAccuracyDelivered item. Optional.
RESPONSE_TIME	Integer	ResponseTime item. Optional.
POSITIONING_METHOD	Integer	PositioningMethod item. Optional.
DEL_TRACKING_PERIOD	Integer	TrackingPeriod item. Optional.
DEL_TRACKING_FREQUENCY	Integer	TrackingFrequency (delivered) item. Optional.
AGE_OF_LOCATION	Integer	AgeOfLocation item. Optional.

Table 1-20 (Cont.) Associated Location Extension Record Fields

Name	Format	Description
CHARGING_TIMESTAMP	Date	ChargingTimeStamp item. Optional.
CHARGING_UTC_OFFSET	String	ChargeInformationList data is stored in DETAIL.ASSOCIATED_CHARGE_BREAKDOWN.CHARGE_PACKET. Mandatory if LCSRequestTimestamp is given.

Associated Value Added Service (VAS) Extension Record (RECType 710)

A Value Added Service (VAS) item represents usage of value added services outside of a standard call; i.e., unrelated to either a Mobile Originated Call or a Mobile Terminated Call. VAS consists of Chargeable Subscriber and Value Added Service Used, which are mandatory; conditionally, Network Type and RAP File Sequence Number; optionally, Operator Specific Information.

Table 1-21 describes the fields in the Associated Value Added Service Extension Record.

Table 1-21 Associated Value Added Service (VAS) Extension Record Fields

Name	Format	Description
RECORD_TYPE	String	Mandatory. Default = 710.
RECORD_NUMBER	Integer	Mandatory. Auto-generated.
VAS_CODE	Integer	Mandatory.
VAS_SHORT_DESC	String	Optional.
VAS_DESC	String	Optional.
CHARGING_START_TIMESTAMP	Date	Optional.
CHARGING_END_TIMESTAMP	Date	Optional.
UTC_TIME_OFFSET	String	Optional.

Associated BRM Balance Record (RECType 900)

Stores data to be loaded into the BRM database.

Associated BRM Billing Records might occur more than once for each Basic Detail Record. This is the case if more than one balance is affected by one event.

Table 1-22 describes the fields in the Associated BRM Balance Record.

Table 1-22 Associated BRM Balance Record Fields

Name	Format	Description
RECORD_TYPE	String	Extended to be 3 bytes long. Values: 900: Associated BRM Balance Record Usage: Determination of the different record types. Derivation: Mandatory. Set by the first processor and left unchanged.
RECORD_NUMBER	9(9)	Sequence number of record in file. Can be used to ensure a linear sequence order for all records; for example, as a sorting criteria. Derivation: Mandatory. Set by the first processor. **Important:** Following modules might change this record number; for example, if new record types are inserted.
ACCOUNT_POID	X(255)	POID of the account. Example: 1 /account 123456789 0 Derivation: Mandatory.
SERVICE_POID	X(255)	POID for the service. Example: 1 /service/ip/gprs 123456789 0 Derivation: Mandatory.
ITEM_POID	X(255)	POID of the item object affected due to this event. Applies only to the balance array element that impacts currency balance elements. This might be different from the PIN_FLD_ITEM_OBJ field in the base **/event** class. Example: 1 /item/misc 123456789 0 Derivation: Mandatory.
ORIGINAL_EVENT_POID	X(255)	POID of the original recorded event. Set only if the event has been extracted for pipeline rerating. Example: "1 /event/delayed 123456 0" Derivation: Optional.

Table 1-22 (Cont.) Associated BRM Balance Record Fields

Name	Format	Description
PIN_TAX_LOCALES	X(255)	Used for tax calculation. Values: "*order_origin*|*order_accept*|*ship_from*|*ship_to*" (Note that these fields are separated by pipes (|).) Each of these values (*order_origin*, *order_accept*, *ship_from*, *ship_to*) is an address in the following format: *city*;*zipcode*;*state*;*country*; [*geocode*,*location_mode*,*international_indicator*] **Note:** Be aware of the semicolon separators and enclosing brackets. For example, "cupertino;95014;CA;US; [5723121,2,0]"Derivation: Optional. *order_origin*, *order_accept*, and *ship_from* addresses are all the same and are derived from account profile object tax supplier information. *ship_to* is the address in the first element of the account's NAMEINFO array. *geocode* is either a geocode or NPA-NXX (the first 6 digits of the phone number). *location_mode* is 1 if it is a geocode and 2 if its NPA-NXX. *international_indicator* is 0 (US) or 1 (International). **Important:** This field might not be implemented in this release.
PIN_TAX_SUPPLIER_ID	X(255)	POID of the **/profile/tax_supplier** object used to tax this event. NULL if there is no tax supplier specified. Derivation: Optional. **Important:** This field might not be implemented in this release.
PIN_PROVIDER_ID	X(255)	POID of the remittance service provider account. Example: 1 /account 123456789 0 Derivation: Optional. **Important:** This field might not be implemented in this release.
PIN_INVOICE_DATA	X(255)	Stores the data in the event that is include in the invoice. T The data is mapped to BRM fields and is stored as a string in the format: "@INTEGRATE#PIN_FLD_CALLING_NUMBER#PIN_FLD_CALLED_NUMBER#PIN_FLD_SVC_TYPE# PIN_FLD_SVC_CODE#PIN_FLD_NUMBER_OF_UNITS#PIN_FLD_USAGE_CLASS#PIN_FLD_DNIS#PIN_FLD_BAL_IMPACTS" Arrays follow this format: PIN_FLD_BAL_IMPACTS = *ID,PIN_FLD_RATE_TAG,PIN_FLD_QUANTITY>* For example: @INTEGRATE#004917165210#0049171235292#T1#11#0#USAGE#GSMThing#<1 /item 3456 1#1234,10.0#rateTag#1.000000#|1 /item 5678 1#6789#20.0#rateTag#1.000000>

Table 1-22 (Cont.) Associated BRM Balance Record Fields

Name	Format	Description
NUMBER_OF_BALANCE_IMPACT_PACKETS	9(2)	Specifies the number of packets following these base fields (dynamic structure); for example, 03 means that 3 packets are following. Must be used to evaluate how the record structure continues. Values: 00 - 99: optionally, 0..n packets might follow Derivation: Mandatory.

Supplementary Balance Impact Packet Record (RECType 600)

The packets can optionally be used to store an event-related balance impact array. Within this structure, N-times PIN_BALANCE_IMPACTs are created, each containing one balance impact per RESOURCE_ID and optionally per GL_ID.

This record is used for evaluation event-related balance impacts together with the REL to map rating-internal Charge-Packets to BRM related balance impacts.

- **Condition**: Only relevant if present. If present, a mapping to all PIN-related values has to take place.
- **Derivation**: Optional. From the BRM object **/event/PIN_FLD_BAL_IMPACTS**. Will be optionally generated by a post-processor. If not present, the mapping will take place within the Rated Event (RE) Loader.

Table 1-23 describes the Supplementary Balance Impact Packet Record fields.

Table 1-23 Supplementary Balance Impact Packet Record Fields

Name	Format	Description
RECORD_TYPE	String	Extended to be 3 bytes long. Value: 600
RECORD_NUMBER	9(9)	Sequence number of record in file. Can be used to ensure a linear sequence order for all records; for example, as a sorting criteria. Derivation: Mandatory. Set by the first processor. **Important:** Following modules might change this record number; for example, if new record types are inserted.
ACCOUNT_POID	String	POID of the account that the balance impact applies to. Derivation: Mandatory. Calculated.
BAL_GRP_POID	String	Balance group that the balance impact applies to. Derivation: Mandatory. Calculated.
OBJECT_CACHE_TYPE	Integer	Mandatory. Calculated.

Table 1-23 (Cont.) Supplementary Balance Impact Packet Record Fields

Name	Format	Description
ITEM_POID	String	POID of the item that the balance impact applies to. Derivation: Mandatory. Calculated.
PIN_RESOURCE_ID	9(9)	Numeric value of the balance element that is impacted; for example, 840 for US dollars. Values: Any configured BRM balance element ID. Derivation: Mandatory.
PIN_RESOURCE_ID_ORIG	Integer	Optional.
PIN_IMPACT_CATEGORY	X(255)	Name of the BRM impact category that was used to generate this balance impact for the rated event. Values: Any configured BRM impact category. Derivation: Mandatory.
PIN_IMPACT_TYPE	Integer	Mandatory. Calculated.
PIN_GL_ID	9(9)	GLID associated with this balance impact. Values: Any configured BRM general ledger ID. Derivation: Optional, default 0. Derived from IFW_RATEPLAN_CNF.GLACCOUNT. Might be mapped from the BRM object **/event/PIN_FLD_BAL_IMPACTS.PIN_FLD_GL_ID**.
RUM_ID	Integer	Mandatory. Calculated. Default = 0.
PIN_OFFERING_POID	String	Optional. Calculated.
PIN_TAX_CODE	X(255)	Tax code for the rate that was used. When taxes do not apply, this field is set to 0. Derivation: Optional. From IFW_RATEPLAN_CNF.GLACCOUNT->TAXCODE. Might be mapped from the BRM object **/event/PIN_FLD_BAL_IMPACTS.PIN_FLD_GL_ID**.

Table 1-23 (Cont.) Supplementary Balance Impact Packet Record Fields

Name	Format	Description
PIN_RATE_TAG	X(255)	Description of the rate used. Same as the PIN_FLD_DESCR in **/rate**. Can be used to more precisely describe the balance impact detail; for example, the following concatenated, comma-separated rating-related Charge-Packet values used: TIMEZONE, DAY_CODE, TIME_INTERVAL. Values: Free defined text value. Derivation: Optional, default empty. From the BRM object **/event/PIN_FLD_BAL_IMPACTS.PIN_FLD_RATE_TAG**. The post-mapping processor might decide what mapping-rule applies to this attribute.
PIN_LINEAGE	X(255)	Lineages of event fields if zone map is used in charge selection. Can be used to more precisely describe the balance impact detail; for example, the following concatenated, comma-separated rating-related Charge-Packet values used: ZONEMODEL, SERVICE_CODE, SERVICE_CLASS, IMPACT_CATEGORY,.RESOURCE, RUMGROUP, PRICEMODEL. Values: Free defined text value. Derivation: Optional, default empty. From the BRM object **/event/PIN_FLD_BAL_IMPACTS.PIN_FLD_LINEAGE**. The post-mapping processor might decide what mapping-rule applies to this attribute.
PIN_NODE_LOCATION	X(255)	Lineage information for the charge offer. See description in charge offers array of **/account**. Can be used to more precisely describe the balance impact detail; for example, the following concatenated, comma-separated rating-related Charge-Packet values used: REVENUEGROUP, DISCOUNTMODEL. Values: Free defined text value. Derivation: Optional, default empty. From the BRM object **/event/PIN_FLD_BAL_IMPACTS.PIN_FLD_NODE_LOCATION**. The post-mapping processor might decide what mapping-rule applies to this attribute.

Table 1-23 (Cont.) Supplementary Balance Impact Packet Record Fields

Name	Format	Description
PIN_QUANTITY	9(15)	Charged quantity value (beats, duration, volume), as calculated via the related RATEPLAN. Contains the rounded quantity value as it has been calculated during rating. Values: Maximum: 999999999999999 **Note:** In case of Multiple-RUM rating, this value might not be totalizable because different UoMs can logically not be aggregated. In this case, the value is set to 0.Derivation: Optional, default 0. From the BRM object **/event/ PIN_FLD_BAL_IMPACTS.PIN_FLD_QUANTITY**. The post-mapping processor might decide what mapping-rule applies to this attribute; for example, multiple charge packet values might be totalized.
PIN_AMOUNT	9(11)	Amount of impact for one balance element to the account balance. The value might be either positive or negative. The value is added to the PIN_FLD_CURRENT_BAL field of the PIN_FLD_BALANCES array in the account object specified by PIN_FLD_ACCOUNT_OBJ. **Note:** In case of Multiple-RUM rating, this value might be a totalized value. Values: Space: No price given, like NULL in a database Variable floating point format: Given value, might be 0.000. The floating decimal point must be set. Minimum: -9999999999 Maximum: 99999999999 Examples: '00000000125' for 125,00 '00000012.50' for 12,50 '-0012.56780' for -12,5678 Derivation: Mandatory. From the BRM object **/event/ PIN_FLD_BAL_IMPACTS.PIN_FLD_AMOUNT**. The post-mapping processor might decide what mapping-rule applies to this attribute; for example, multiple charge packet values might be totalized. **Note:** This value does not include any granted discounts.
PIN_AMOUNT_ORIG	Decimal	Optional.
PIN_PERCENT	Decimal	Optional
PIN_AMOUNT_DEFERRED	Decimal	Optional. Calculated.

Table 1-23 (Cont.) Supplementary Balance Impact Packet Record Fields

Name	Format	Description
PIN_DISCOUNT	9(11)	The discount applied to this balance impact. Can be used to determine the total charge amount value. Note: The AMOUNT value never contains this DISCOUNT value. Values: Space: No price given, like NULL in a database Variable floating point format: Given value, might be 0.000. The floating decimal point must be set. Minimum: -9999999999 Maximum: 99999999999 Examples: '00000000125' for 125,00 '00000012.50' for 12,50 '-0012.56780' for -12,5678 Derivation: Mandatory, default 0.
PIN_INFO_STRING	X(2000)	Stores the pricing type.

Supplementary Sub-Balance Impact Packet Record (RECType 605)

Stores balance impacts for sub-balances.

Table 1-24 describes the fields in the Supplementary Sub-Balance Impact Packet Record.

Table 1-24 Supplementary Sub-Balance Impact Packet Record Fields

Name	Format	Description
RECORD_TYPE	String	Extended to be 3 bytes long. Value: 605 Mandatory.
RECORD_NUMBER	9(9)	Sequence number of record in file. Can be used to ensure a linear sequence order for all records; for example, as a sorting criteria. Derivation: Mandatory. Set by the first processor. **Important:** Following modules might change this record number; for example, if new record types are inserted.
BAL_GRP_POID	String	Balance group that the balance impact applies to. Derivation: Mandatory. Calculated.

Table 1-24 (Cont.) Supplementary Sub-Balance Impact Packet Record Fields

Name	Format	Description
PIN_RESOURCE_ID	9(9)	Numeric value of the balance element that is impacted; for example, 840 for US dollars. Values: Any configured BRM balance element ID. Derivation: Mandatory.
NEXT_BAL	Decimal	None.
DELAYED_BAL	Decimal	None.
GRANTOR	String	The charge offer or discount offer that granted this balance element.
VALID_FROM_DETAILS	Integer	Sub-balance start time mode (such as first-usage or relative) and relative offset and unit.
VALID_TO_DETAILS	Integer	Sub-balance end time mode (such as relative) and relative offset and unit.

Supplementary Sub-Balance Info Packet Record (RECType 607)

Stores validity dates for sub-balances.

Table 1-25 lists the fields in the Supplementary Sub-Balance Info Packet Record.

Table 1-25 Supplementary Sub-Balance Info Packet Record Fields

Name	Format	Description
RECORD_TYPE	String	Extended to be 3 bytes long. Value: 607 Mandatory.
RECORD_NUMBER	9(9)	Sequence number of record in file. Can be used to ensure a linear sequence order for all records; for example, as a sorting criteria. Derivation: Mandatory. Set by the first processor. **Important:** Following modules might change this record number; for example, if new record types are inserted.
PIN_AMOUNT	Decimal	Sub-balance amount.
VALID_FROM	Date	Valid from date for this sub-balance.
VALID_TO	Date	Valid to date for this sub-balance.

Tax Jurisdiction Packet

Table 1-26 lists the fields in the Tax Jurisdiction Packet.

Table 1-26 Tax Jurisdiction Packet Fields

Name	Format
RECORD_TYPE	String
RECORD_NUMBER	Integer
PIN_TAX_TYPE	String
PIN_TAX_VALUE	Decimal
PIN_AMOUNT	Decimal
PIN_TAX_RATE	String
PIN_AMOUNT_GROSS	Decimal

EDR Container Fields for Balance Monitoring

The following fields are used for handling balance monitor information.

MONITOR_LIST (DETAIL.CUST_A.ML)

The MONITOR_LIST packet contains information about the balance monitor.

Table 1-27 lists the fields in the MONITOR_LIST packet.

Table 1-27 MONITOR_LIST Packet Fields

Name	Format	Description
BALANCE_GROUP_ID	String	Balance monitor group ID. Mandatory.
MONITOR_OWNER_ACCT_ID	String	Monitor owner's account ID. Mandatory.
MONITOR_OWNER_ID	String	Monitor owner ID. Mandatory.
MONITOR_OWNER_TYPE	String	Monitor owner type. Mandatory.

MONITOR_PACKET (DETAIL.ASS_PIN.MP)

The MONITOR_PACKET packet stores information about the balance monitor impacts. This information is added to the Associated Billing Record to be loaded into the database.

Table 1-28 lists the fields in the MONITOR_PACKET packet.

Table 1-28 MONITOR_PACKET (DETAIL.ASS_PIN.MP) Fields

Name	Format	Description
RECORD_TYPE	String	Type of record. Extended to be 3 bytes long. Possible value: 800

Table 1-28 (Cont.) MONITOR_PACKET (DETAIL.ASS_PIN.MP) Fields

Name	Format	Description
RECORD_NUMBER	9(9)	Sequence number of record in file. Can be used to ensure a linear sequence order for all records; for example, as a sorting criteria. Derivation: Mandatory. Set by the first processor. Important: This record number can change if the sequence of records changes; for example, if new record types are inserted.
ACCOUNT_POID	String	POID of the account that the monitor balance impact applies to. Derivation: Mandatory. Calculated.
BAL_GRP_POID	String	Balance monitor group that the monitor balance impact applies to. Derivation: Mandatory. Calculated.
PIN_RESOURCE_ID	9(9)	Numeric value of the balance element that is impacted; for example, 840 for US dollars. Possible value: Any configured balance element ID. Derivation: Mandatory.
PIN_AMOUNT	9(11)	Amount of impact for one balance element to the monitor balance. The value might be either positive or negative. The value is added to the PIN_FLD_CURRENT_BAL field of the PIN_FLD_BALANCES array in the account's monitor object specified by PIN_FLD_ACCOUNT_OBJ field. **Note:** In case of Multiple-RUM rating, this value might be a total value. Possible values: Price (see below for maximum and minimum). If no price given, space; for example, NULL in a database. The format is variable floating point. The floating decimal point must be set if the given value is not in the required format. Example: '00000000125' for 125,00 '00000012.50' for 12,50 '-0012.56780' for -12,5678 Derivation: Mandatory. Derived from the object **/event/PIN_FLD_BAL_IMPACTS.PIN_FLD_AMOUNT**. The post-mapping processor decides what mapping rule applies to this attribute; for example, add multiple charge packet values. **Note:** This value does not include any granted discounts.

MONITOR_SUB_BAL_IMPACT (DETAIL.ASS_PIN.MSBI)

Table 1-29 lists the fields in the MONITOR_SUB_BAL_IMPACT packet.

Table 1-29 MONITOR_SUB_BAL_IMPACT Fields

Name	Format	Description
RECORD_TYPE	String	Type of record. Extended to be 3 bytes long. Possible value: 805
RECORD_NUMBER	9(9)	Sequence number of the record in file. Can be used to ensure a linear sequence order for all records; for example, as a sorting criteria. Derivation: Mandatory. Set by the first processor. **Important:** This record number can change if the sequence of records changes; for example, if new record types are inserted.
BAL_GRP_POID	String	Balance monitor group that the monitor balance impact applies to. Derivation: Mandatory. Calculated.
PIN_RESOURCE_ID	9(9)	Numeric value of the balance element that is impacted; for example, 840 for US dollars. Possible values: Any configured balance element ID. Derivation: Mandatory.
MONITOR_SUB_BAL	SB	Sub-balance monitor.

MONITOR_SUB_BAL (DETAIL.ASS_PIN.MSB)

Table 1-30 lists the MONITOR_SUB_BAL Packet fields.

Table 1-30 MONITOR_SUB_BAL Fields

Name	Format	Description
RECORD_TYPE	String	Type of record. Extended to be 3 bytes long. Possible value: 807 Derivation: Mandatory.
RECORD_NUMBER	Integer	Contains the UTC time offset that normalizes the VALID_FROM timestamp to the UTC time zone.
PIN_AMOUNT	Decimal	Sub-balance amount.
VALID_FROM	Date	Contains a timestamp of the event end time, rounded to midnight.
VALID_TO	Date	Contains a timestamp of the VALID_FROM time plus 1 day.
CONTRIBUTOR	String	None
NEXT_BAL	Decimal	None
DELAYED_BAL	Decimal	None
ACCOUNT_POID_STR	String	None
SERVICE_POID_STR	String	None

Table 1-30 (Cont.) MONITOR_SUB_BAL Fields

Name	Format	Description
OFFERING_POID_STR	String	None
START_T	Date	None
UTC_TIME_OFFSET	String	None
DESCRIPTION	String	Optional.
FLAGS	Integer	Optional.

Associated Invoice Data Record (RECType @INTEGRATE)

The Associated Invoice Data Record stores data for displaying on invoices.

Table 1-31 lists the fields in the Associated Invoice Data Record.

Table 1-31 Associated Invoice Data Record Fields

Name	Format	Description
RECORD_TYPE	String	The name of the invoice data template, preceded by the @ symbol. Values: @INTEGRATE
A_NUMBER	String	See A_NUMBER.
B_NUMBER	String	See B_NUMBER.
BASIC_SERVICE	String	See BASIC_SERVICE.
NUMBER_OF_UNITS	Decimal	See NUMBER_OF_UNITS.
USAGE_CLASS	String	See USAGE_CLASS.
TERMINATING_SWITCH_IDENTIFICATION	String	See TERMINATING_SWITCH_IDENTIFICATION.
BALANCE_IMPACT	N/A	Balance impact data.
INVOICE_DATA_TERMINATOR	String	N/A

Associated Zone Breakdown Record (RECType 960-969)

Stores zoning information. For each evaluated zone type, a single Zone Breakdown Record is generated, following the Basic Detail Record (020, 021, 030, 031, etc.). A new Basic Record or the Trailer Record end the sequence of Zone Breakdown Records. Also for zone values already contained within the Basic Detail Record, these sub-details could be generated.

Table 1-32 lists the fields in the Associated Zone Breakdown Record.

Table 1-32 Associated Zone Breakdown Record Fields

Name	Format	Description
RECORD_LENGTH	Integer	Optional for backward compatibility.

Table 1-32 (Cont.) Associated Zone Breakdown Record Fields

Name	Format	Description
RECORD_TYPE	String	Extended to be 3 bytes long. Values: 960: Standard Zoning (multiple global Zoning per logical EDR Format) 961: Segmentation Zoning (multiple Zoning per customer segment)
RECORD_NUMBER	9(9)	Sequence number of record in file. Can be used to ensure a linear sequence order for all records; for example, as a sorting criteria. Derivation: Mandatory. Set by the first processor. **Important:** Following modules might change this record number; for example, if new record types are inserted.
CONTRACT_CODE	X(20)	External unique contract code as defined within the associated billing system. Uniquely identifies a charge offer-related contract. Could by used by any post-processors to look up and reference contract, subscriber, and customer data (if needed later on within this post processor). Derivation: Optional. As assigned to an ACCOUNT Object and referenced by a primary CLI. Alternatively the A Number could be used as reference.
SEGMENT_CODE	X(5)	External Segmentation ID as defined within the associated billing system or as defined within the rating process. Segments could vertically group multiple subscriber (for example, for quality reasons) or network operator. Could be used by any post-processor to identify the related customer/network segment that was used during the rating processor for this A Number. Derivation: Only mandatory for RECORD_TYPE 981, 984. As assigned to a SUBSCRIBER Object related to the A Number or as assigned to a network operator related to the file stream.
CUSTOMER_CODE	X(20)	External Customer Code as defined within the associated billing system. Could group multiple subscribers. Could be used by any post-processors as an alternative identifier to look up and reference subscriber or customer data (if needed later on within this post-processor). Derivation: Optional. As assigned to a CUSTOMER Object and referenced by a primary CLI.

Table 1-32 (Cont.) Associated Zone Breakdown Record Fields

Name	Format	Description
ACCOUNT_CODE	X(20)	External Customer-Account Code as defined within the associated billing system. Could group multiple charge offers assigned to a customer. A customer might have multiple accounts. Could be used by any post-processors as an alternative identifier to look up and reference subscriber or customer data (if needed later on within this post-processor). Derivation: Mandatory. As assigned to a CUSTOMER Object and referenced by a primary CLI.
SYSTEM_BRAND_CODE	X(5)	External system or brand, specialist system Code as defined within the associated rating or billing. Could be used for vendor-specific reasons (for example, reseller code or target system identification for post processing, NOSP identification, etc.). Derivation: Mandatory, default 0. As defined within the SYSTEM_BRAND Object and assigned to a PRODUCT Object referenced by a primary CLI.
SERVICE_CODE	X(5)	Internal (mapped, normalized) Service Code used for the zone determination within the associated rating or billing processor, out of the object IFW_SERVICE (.CODE). Could be used by any post-processor to evaluate the service that was really used during the related rating process. Derivation: Mandatory. The external service code is mapped to a unique representation, either: • Out of the service code included in the origin record (might be mapped). • Out of the service code associated to the SUBSCRIBER's A Number.
CUSTOMER_RATEPLAN_CODE	X(10)	The Original charge offer related and Customer/Subscriber specific charge as defined within the associated billing system. If no customer data is present, the actual, internally used charge could be used instead. Could be used by any post-processor to evaluate the charge that was really used during the related rating process. Derivation: Only mandatory for RECORD_TYPE 981, 984. As assigned to an ACCOUNT Object and referenced by a primary CLI or as assigned to the associated charge.

Table 1-32 (Cont.) Associated Zone Breakdown Record Fields

Name	Format	Description
SLA_CODE	X(5)	The Original charge offer-related and customer-specific Service Level Agreement as defined within the associated billing system. If no customer data is present, the actual, internally default value could be used instead. Could be used by any post-processor to evaluate the charge that was really used during the related rating process. Derivation: Only mandatory for RECORD_TYPE 981, 984. As assigned to an ACCOUNT Object and referenced by a primary CLI or as assigned to the associated charge.
CUSTOMER_BILLCYCLE	X(2)	The Customers associated Billcycle Code as defined within the associated billing system. Could be used by any post-processor to evaluate the billcycle period that applies to this call. Derivation: Only mandatory for RECORD_TYPE 981, 984. As assigned to a CUSTOMER Object and referenced by a primary CLI.
CUSTOMER_CURRENCY	X(3)	The Customers associated Currency as defined within the associated billing system. Could be used by any post-processor to evaluate the currency and to apply exchange rates that apply to this call. Derivation: Only mandatory for RECORD_TYPE 981, 984. As assigned to a CUSTOMER Object and referenced by a primary CLI.
CUSTOMER_TAX_GROUP	X(5)	The Customers associated Tax Group Code as defined within the associated billing system. Could be used by any post-processor to evaluate the tax rate (together with the charge configuration related G/L account's tax code) that applies to this call. Derivation: Only mandatory for RECORD_TYPE 981, 984. As assigned to a CUSTOMER Object and referenced by a primary CLI.
NUMBER_OF_ZONE_PACKET	9(2)	Defines the number of supplementary zone records following these base fields (dynamic structure); for example, a number of '05' means that 5 records are following. Must be used to evaluate how the record structure continues. Values: 01 - 99: A minimum of at least 1 record is required Derivation: Mandatory.

Supplementary Zone Packet Record (RECType 660)

For each zone model (evaluated by any rating processor), one packet is added to this structure.

This applies only to standard and segmentation zoning (where multiple zoning is possible). All other zone values are listed in the charge breakdown records.

Table 1-33 lists the fields in the Supplementary Zone Packet Record.

Table 1-33 Supplementary Zone Packet Record Fields

Name	Format	Description
RECORD_TYPE	String	Extended to be 3 bytes long. Value: 660: Zone Packet Derivation: Mandatory. Set by the first processor and left unchanged.
RECORD_NUMBER	9(9)	Sequence number of record in file. Can be used to ensure a linear sequence order for all records; for example, as a sorting criteria. Derivation: Mandatory. Set by the first processor. **Important:** Following modules might change this record number; for example, if new record types are inserted.
ZONEMODEL_CODE	X(10)	External Zone Model Code as defined in the related ZONEMODEL Object (.CODE) of the rating processor. Could be used by any post-processor to evaluate the zone model that was used during the related rating process. Derivation: Mandatory.
ZONE_RESULT_VALUE_WS	X(5)	Wholesale zone result value as defined for the zone model references by the field ZONEMODEL_CODE. Could be used by any post-processor to evaluate the wholesale zone value that was estimated during the related rating process. Derivation: Optional.
ZONE_RESULT_VALUE_RT	X(5)	Retail zone result value as defined for the zone model references by the field ZONEMODEL_CODE. Could be used by any post-processor to evaluate the retail zone value that was estimated during the related rating process. Derivation: Mandatory.
ZONE_ENTRY_NAME	String	Calculated, will be used by zoning and rating modules.
ZONE_DESCRIPTION	String	Calculated, will be used by zoning and rating modules.

Table 1-33 (Cont.) Supplementary Zone Packet Record Fields

Name	Format	Description
DISTANCE	9(5)	Distance value as calculated by any geographical zone model. Could be used by any post-processor to evaluate the distance value that was estimated during the related rating process. Condition: This applies only if the associated zone model references to a geographical one. Values: The value is given in full and rounded kilometers; for example, 00150 for 150 km. Derivation: Optional. Dependent on the setup of the zone models within the related rating processor. This value represents the internal calculated distance.

Associated Charge Breakdown Record (RECType 970-998)

Stores charge data. For each evaluated charge or partial charge, a single Charge Breakdown Record might be generated, following the Basic Detail Record. A new Detail Record or the Trailer Record end the sequence of Charge Breakdown Records. For charge values already contained within the Basic Detail Record, these sub-details could be generated.

Table 1-34 lists the fields in the Associated Charge Breakdown Record.

Table 1-34 Associated Charge Breakdown Record Fields

Name	Format	Description
RECORD_LENGTH	Integer	Optional for backward compatibility.
RECORD_TYPE	String	Extended to be 3 bytes long. Values: 980: Global Charge (multiple EDR-format-related charge) 981: Customer Charge (subscriber-related charge) 982: Reseller/SP Charge (specialist-system-related charge) 983: Content Provider Charge (content-related charge) 990: Carrier Interconnection Charge (trunk-related charge) 991: Reseller Interconnection Charge (EDR-format-related charge) Derivation: Mandatory. Set by the first processor and left unchanged.
RECORD_NUMBER	9(9)	Sequence number of record in file. Can be used to ensure a linear sequence order for all records; for example, as a sorting criteria. Derivation: Mandatory. Set by the first processor. **Important:** Following modules might change this record number; for example, if new record types are inserted.

Table 1-34 (Cont.) Associated Charge Breakdown Record Fields

Name	Format	Description
CONTRACT_CODE	X(20)	External unique contract code as defined within the associated billing system. Uniquely identifies a charge offer-related contract. Could by used by any post-processors to look up and reference contract, subscriber, and customer data (if needed later on within this post-processor). Derivation: Optional. As assigned to an ACCOUNT Object and referenced by a primary CLI. Alternatively the A Number could be used as reference.
SEGMENT_CODE	X(5)	External Segmentation ID as defined within the associated billing system or as defined within the rating process. Segments could vertically group multiple subscriber (for example, for quality reasons) or network operator. Could be used by any post-processor to identify the related customer/network segment that was used during the rating processor for this A Number. Derivation: Only mandatory for RECORD_TYPE 981, 984. As assigned to a SUBSCRIBER Object related to the A Number or as assigned to a network operator related to the file stream.
CUSTOMER_CODE	X(20)	External Customer Code as defined within the associated billing system. Could group multiple subscribers. Could be used by any post-processors as an alternative identifier to look up and reference subscriber or customer data (if needed later on within this post-processor). Derivation: Optional. As assigned to a CUSTOMER Object and referenced by a primary CLI.
ACCOUNT_CODE	X(20)	External Customer-Account Code as defined within the associated billing system. Could group multiple charge offers assigned to a customer. A customer might have multiple accounts. Could be used by any post-processors as an alternative identifier to look up and reference subscriber or customer data (if needed later on within this post-processor). Derivation: Mandatory. As assigned to a CUSTOMER Object and referenced by a primary CLI.
SYSTEM_BRAND_CODE	X(5)	External system or brand, specialist system Code as defined within the associated rating or billing. Could be used for vendor-specific reasons (for example, reseller code or target system identification for post-processing, NOSP identification, etc.) Derivation: Mandatory, default 0. As defined within the SYSTEM_BRAND Object and assigned to a PRODUCT Object referenced by a primary CLI.

Table 1-34 (Cont.) Associated Charge Breakdown Record Fields

Name	Format	Description
SERVICE_CODE	X(5)	Internal (mapped, normalized) Service Code used for the zone determination within the associated rating or billing processor, out of the object IFW_SERVICE (.CODE). Could be used by any post-processor to evaluate the service that was really used during the related rating process. Derivation: Mandatory. The external service code is mapped to a unique representation, either: • Out of the service code included in the origin record (might be mapped). • Out of the service code associated to the SUBSCRIBERs A Number.
CUSTOMER_RATEPLAN_CODE	X(10)	The Original charge offer related and Customer/Subscriber specific charge as defined within the associated billing system. If no customer data is present, the actual, internally used charge could be used instead. Could be used by any post-processor to evaluate the charge that was really used during the related rating process. Derivation: Only mandatory for RECORD_TYPE 981, 984. As assigned to an ACCOUNT Object and referenced by a primary CLI or as assigned to the associated charge.
SLA_CODE	X(5)	The Original charge offer-related and customer-specific Service Level Agreement as defined within the associated billing system. If no customer data is present, the actual, internally default value could be used instead. Could be used by any post-processor to evaluate the charge that was really used during the related rating process. Derivation: Only mandatory for RECORD_TYPE 981, 984. As assigned to an ACCOUNT Object and referenced by a primary CLI or as assigned to the associated charge
CUSTOMER_BILLCYCLE	X(2)	The Customer's associated Billcycle Code as defined within the associated billing system. Could be used by any post-processor to evaluate the billcycle period that applies to this call. Derivation: Only mandatory for RECORD_TYPE 981, 984. As assigned to a CUSTOMER Object and referenced by a primary CLI

Table 1-34 (Cont.) Associated Charge Breakdown Record Fields

Name	Format	Description
CUSTOMER_CURRENCY	X(3)	The Customer's associated Currency as defined within the associated billing system. Could be used by any post-processor to evaluate the currency and to apply exchange rates that apply to this call. Derivation: Only mandatory for RECORD_TYPE 981, 984. As assigned to a CUSTOMER Object and referenced by a primary CLI.
CUSTOMER_TAXGROUP	X(5)	The Customers associated Tax Group Code as defined within the associated billing system. Could be used by any post-processor to evaluate the tax rate (together with the charge configuration-related G/L account's tax code) that applies to this call. Derivation: Only mandatory for RECORD_TYPE 981, 984. As assigned to a CUSTOMER Object and referenced by a primary CLI.
NUMBER_OF_CHARGE_PACKETS	9(2)	Defines the number of CBRs (charge breakdown records); does not reflect the actual number of charge packets per CBR.
NUMBER_OF_TAX_PACKETS	Integer	Mandatory. Calculated. Default = 1.
CUSTOMER_OPENING_BALANCE	Decimal	If prepaid rated call, the opening balance for the subscriber. Optional.
CUSTOMER_CLOSING_BALANCE	Decimal	If prepaid rated call, the closing balance for the subscriber. Optional.
RUM_NAME	String	Optional. Calculated.
FU_DISCOUNT_OBJECTS	String	The account's discounts that have first-usage start times which were used to discount the event. Mandatory. Calculated.

Update Balance Packet

This block contains initialized sub-balances of related balance elements based on a bundle.

Table 1-35 lists the fields in the Update Balance Packet.

Table 1-35 Update Balance Packet Fields

Name	Format	Description
RECORD_TYPE	String	The type of call record. Mandatory.

Table 1-35 (Cont.) Update Balance Packet Fields

Name	Format	Description
BALANCE_GROUP_ID	Integer	POID of the account's balance group for which a balance element balance starts on first usage. Mandatory.
RESOURCE_ID	Integer	ID of the associated balance element. Mandatory.
RECORD_NUMBER	Integer	Sequence number of the record in the file. Mandatory.
VALID_FROM	Date	The balance element balance start time. Mandatory.
VALID_TO	Date	The balance element balance end time. Mandatory.
VALID_FROM_DETAIL	Integer	The start time mode (such as first-usage or relative), relative offset unit (such as minutes, months, or cycles), and number of offset units. Mandatory.
VALID_TO_DETAIL	Integer	The end time mode (such as relative), relative offset unit (such as minutes, months, or cycles), and number of offset units. Mandatory.
CONTRIBUTOR	String	Balance group contributor.
GRANTOR	String	Balance group grantor.
GRANT_VALID_FROM	Date	Grant validity start time.
GRANT_VALID_TO	Date	Grant validity end time.

RUM Map Block

RUM_MAP block contains all the RUMs that are used in the ACB block.

Table 1-36 lists the fields in the RUM Map Block.

Table 1-36 RUM Map Block Fields

Name	Format	Description
RECORD_TYPE	String	None
RECORD_NUMBER	Integer	None
RUM_NAME	String	None
NET_QUANTITY	Decimal	Contains the summation of BALANCE_PACKET.PIN_QUANTITY for RUM_NAME.
UNRATED_QUANTITY	Decimal	None

Supplementary Minimum Charge Information

Minimum charge information (the MINIMUM_CHARGE block) prevents charging a customer less than the minimum charge for a call. The values are taken from the pricing configuration.

Table 1-37 lists the Supplementary Minimum Charge Information fields.

Table 1-37 Supplementary Minimum Charge Information Fields

Name	Format	Description
RESOURCE	String	Balance Element used when rating the call.
TOTAL_CHARGE_VALUE	Decimal	Total charge of the call.
MINIMUM_CHARGE_VALUE	Decimal	Minimum charge of the call.

Split Charge Packet

FCT_Discount splits charge packets if necessary during prepaid authorization. Each split charge packet represents a segment with a single net rate, including discounts.

Table 1-38 lists the fields in the Split Charge Packet.

Table 1-38 Split Charge Packet Fields

Name	Format	Description
RESOURCE_ID	Integer	Numeric ID of the balance element. Used for filtering in the discount detail. Copied from the original charge packet.
RUM	String	RUM name. Copied from the original charge packet.
QUANTITY_FROM	Decimal	Split charge packet start quantity. Calculated by the module.
QUANTITY_TO	Decimal	Split charge packet end quantity. Calculated by the module.
CHARGED_AMOUNT_VALUE	Decimal	Amount of the charge for this split charge packet. Calculated by the module.
INTERN_PACKET_INDEX	Integer	The index of the split charge packet.
INTERN_SRC_PACKET_INDEX	Integer	The packet index of the charge packet from which this split charge packet was generated.

Supplementary Last Beat Information

Information about the last beat (the LAST_BEAT_INFO block) is mainly used for abnormal call terminations.

Table 1-39 lists the fields in the Supplementary Last Beat Information block fields.

Table 1-39 Supplementary Last Beat Information Fields

Name	Format	Description
LAST_BEAT_QUANTITY	Decimal	The length of a beat. The value of a beat (for example, clicks or bytes) is defined in the pricing and determined by *FCT_MainRating*.

Table 1-39 (Cont.) Supplementary Last Beat Information Fields

Name	Format	Description
LAST_BEAT_CHARGE	Decimal	The charge for a beat.

Charge Breakdown Record Tax Packet (RECType 660)

Add code to the OutGrammar to store tax information from the EDR container into the output TAP blocks.

Block. *n* times. Optional.

Table 1-40 lists the fields in the Charge Breakdown Record Tax Packet.

Table 1-40 Charge Breakdown Record Tax Packet Fields

Name	Format	Description
RECORD_TYPE	String	Mandatory. Must be set to 660.
RECORD_NUMBER	Integer	Mandatory. Auto-generated.
TAX_CODE	Integer	None
TAX_RATE	String	None
TAX_VALUE	Decimal	None
TAX_PERCENT	Decimal	None
TAX_VALUE_ORIG	Decimal	Optional. Used when exchange rate is configured.
TAX_TYPE	String	None
CHARGE_TYPE	String	None
TAXABLE_AMOUNT	Decimal	None
TAX_QUANTITY	Decimal	None
RELATED_RECORD_NUMBER	Integer	None
RELATED_CHARGE_INFO_ID	Integer	None
CHARGE_INFORMATION_COUNTER	Integer	None
CHARGE_INFORMATION_COUNTER	Integer	None

Associated Message Description Record (RECType 999)

Stores errors that occur in preprocessing. For each error a single Message Description Record is generated, following the Basic Record. A new Basic Record or the Trailer Record end the sequence of Message Description Records.

Table 1-41 lists the fields in the Associated Message Description Record.

Table 1-41 Associated Message Description Record Fields

Name	Format	Description
RECORD_TYPE	String	Value: 999: Message Description Record Derivation: Mandatory. Set by the first processor and left unchanged. Usage: Determination of the different record types.
RECORD_NUMBER	9(9)	Sequence number of record in file. Can be used to ensure a linear sequence order for all records; for example, as a sorting criteria. Derivation: Mandatory. Set by the first processor. Important: Following modules might change this record number; for example, if new record types are inserted.
SYSTEM	X(8)	Description or Code of the system which produced this record; for example, Host name or process name.
MESSAGE_SEVERITY	X(1)	Severity code for this message. Values: N: Normal (Hint) W: Warning E: Error (could be either a minor, major, or critical error) Derivation: Mandatory. Set by the first processor and left unchanged.
MESSAGE_ID	N(7)	An error code used to cross-reference the call to the relevant description. Values: 0000000 through 9999999 Derivation: Mandatory. Set by the first processor and left unchanged.
MESSAGE_DESCRIPTION	X(50)	Description of the error. It is mandatory but the content is entirely at the discretion of the pre-processor. Derivation: Mandatory. Set by the first processor and left unchanged.

Associated TAP Error Record

Table 1-42 lists the Associated TAP Error Record fields.

Table 1-42 Associated TAP Error Record Fields

Name	Format
RECORD_TYPE	String
RECORD_NUMBER	Integer
ERROR_NAME	String
ERROR_SEVERITY	Integer

Table 1-42 (Cont.) Associated TAP Error Record Fields

Name	Format
TAP3_ERROR_CODE	String
TAP3_ERROR_APPLICATION_TAG	String
TAP3_ERROR_DEPTH	String

Associated SMS Record (RECType 580)

This optional record is used to store SMS call data.

Table 1-43 lists the fields in the Associated SMS Record.

Table 1-43 Associated SMS Record Fields

Name	Format	Description
RECORD_TYPE	String	Record type for Associated SMS Record. Value: 580 Derivation: Mandatory. Set by the first processor.
RECORD_NUMBER	9(9)	Sequence number of record in file. Used to ensure a linear sequence order for all records; for example, as a sorting criteria. Values: Minimum: 000000002 Maximum: 999999998 Derivation: Mandatory. Auto-generated. Set by the first processor. Important: Record number may change; for example, if new record types are inserted.
CONTENT_INDICATOR	X(1)	Indicator as to the contents of the message sent or received. Derivation: Optional.
ORIGINATING SWITCH IDENTIFICATION	X(10)	SMS-C from which the SMS was issued by the A party. Derivation: Optional.
DESTINATION SWITCH IDENTIFICATION	X(10)	SMS-C from which the SMS was issued to the B party. Derivation: Optional.
PROVIDER ID	X(2)	Unique Service Provider Identifier. Derivation: Optional.
SERVICE ID	X(2)	Unique Service ID. Derivation: Optional.

Table 1-43 (Cont.) Associated SMS Record Fields

Name	Format	Description
DEVICE NUMBER	X(24)	Identifies the equipment used by the subscriber during the call; for example, the International Mobile Equipment Identity number (IMEI). Derivation: Optional.
PORT NUMBER	X(24)	Identifies the unique subscriber ID; for example, the IMSI number. Derivation: Optional.
DIALED DIGITS	X(40)	The number dialed by the customer when establishing a call or the number to which the call is forwarded or transferred. Derivation: Optional.

Associated MMS Record (RECType 590)

This optional record is used to store MMS call data.

Table 1-44 lists the Associated MMS Record fields.

Table 1-44 Associated MMS Record Fields

Field Name	Data Format	Description
RECORD TYPE	String	Record type for Associated MMS Record. Value: 590 Derivation: Mandatory. Set by the first processor.
RECORD NUMBER	9(9)	Sequence number of record in file. Used to ensure a linear sequence order for all records; for example, as a sorting criteria. Values: Minimum: 000000002 Maximum: 999999998 Derivation: Mandatory. Auto-generated. Set by the first processor. **Important:** Record number may change; for example, if new record types are inserted.
ACCOUNT STATUS TYPE	X(2)	Indicator of the account type from which the message was sent. Derivation: Optional.
PRIORITY	X(2)	Indicator as to the priority of the message; for example, high, medium or low. Derivation: Optional.

Table 1-44 (Cont.) Associated MMS Record Fields

Field Name	Data Format	Description
MESSAGE CONTENT	X(255)	Content type; for example, image or plain text. Derivation: Optional.
MESSAGE ID	X(16)	Unique message group ID. If the message was sent as part of a group, an indicator as to which group it was sent from. Derivation: Optional.
STATION IDENTIFIER	X(255)	Station from which message was sent. Value: MMS identifier. Derivation: Optional.
FC INDICATOR	X(9)	Indicator as to whether the message was forwarded or copied. Derivation: Optional.
CORRELATION ID	X(16)	Correlation ID for the message if part of a group. Derivation: Optional.
DEVICE NUMBER	X(24)	Identifies the equipment used by the subscriber during the call; for example, the International Mobile Equipment Identity number (IMEI). Derivation: Optional.
PORT NUMBER	X(24)	Identifies the unique subscriber ID; for example, the IMSI number. Derivation: Optional.
DIALED DIGITS	X(40)	The number dialed by the customer when establishing a call or the number to which the call is forwarded or transferred. Derivation: Optional.
CELL ID	X(10)	Cell ID of the A party, or the cell from which the call originated. Derivation: Optional.
B CELL ID	X(10)	Cell ID of the B party, or the cell receiving the call. Derivation: Optional.
A TERM CELL ID	X(10)	Cell ID of the A party when the call terminated. Derivation: Optional.

Trailer Record (RECType 090)

Table 1-45 lists the fields in the Trailer Record.

Table 1-45 Trailer Record Fields

Name	Format	Description
RECORD_LENGTH	Integer	Optional for backward compatibility.
RECORD_TYPE	String	Extended to be 3 bytes long, first byte denotes market like GSM, ISDN. Value: 090: Trailer Record Derivation: Mandatory. Set by the first processor and left unchanged. Usage: Determination of the different record types.
RECORD_NUMBER	9(9)	Sequence number of record in file. Can be used to ensure a linear sequence order for all records; for example, as a sorting criteria. Derivation: Mandatory. Set by the first processor.
SENDER	X(10)	Identifies the PLMN or physical (network) operator, which is sending the file, used to determine the network, which is the sender of the data. The full list of mobile codes in use is given in MoU TADIG PRD TD. 13: PLMN Naming Conventions. Can be used to determine a unique NOSP_ID together with the RECIPIENT. Can also be used to determine the network operator responsible for the EDR. Derivation: Optional, but should be defaulted if not present on the input side, for example, by own NO-Id, for example, 'DTAG'. Set by the first processor and left unchanged.
RECIPIENT	X(10)	Identifies the PLMN or physical (network) operator to whom the file is being sent, used to determine the network, which is the recipient of the data. The full list of mobile codes in use is given in MoU TADIG PRD TD. 13: PLMN Naming Conventions. Can be used to determine a unique NOSP_ID together with the SENDER. Can also be used to determine the reseller or service provider who is responsible for billing these CDRs. Derivation: Optional, but should be defaulted if not present on the input side, for example, by own NO-Id, for example, 'DTAG'. Set by the first processor and left unchanged.

Table 1-45 (Cont.) Trailer Record Fields

Name	Format	Description
SEQUENCE_NUMBER	9(6)	Identifies each file sent by the VPLMN or logical sender to a particular HPLMN or logical recipient. It indicates the file number of the specific file type, starting at 1 and increments by one for each new file of that type sent. Separate sequence numbering must be used for Test- and Chargeable-Data. Having reached the maximum value (99999), the number must recycle to 1. **Note:** In the case of retransmission for any reason, this number does not increment. Range: 000001 - 999999 for Test Data 000001 - 999999 for Chargeable Data Derivation: Mandatory. Set by the first processor and could be changed by any following processor in case of recycling to assure a unique and linear sequence order to all following processors.
ORIGIN_SEQUENCE_NUMBER	9(6)	Original file sequence number as generated the first time. Identical content as for the SEQUENCE_NUMBER, but will never be changed. Used as a reference to the original file/stream, if any processor has changed the actual file sequence number. Derivation: Mandatory, defaulted by SEQUENCE_NUMBER. Set by the first processor and left unchanged.
TOTAL_NUMBER_OF_RECORDS	9(9)	The total number of Basic Record in the file, excluding header and trailer. Should be used as a check value, to determine that all records have been correctly transmitted and/or used. Condition: Not Present in a Notification File or if no Detail records are present. Maximum Number: 999999999 Derivation: Mandatory. Might be recalculated by any processor.
TAP_TOTAL_NUMBER_OF_RECORDS	Integer	Mandatory. Set by TAP input grammar.

Table 1-45 (Cont.) Trailer Record Fields

Name	Format	Description
FIRST_START_TIMESTAMP	YYYYMMDD HHMISS	The earliest start of charging timestamp on any Basic Detail Record. It is not necessarily the start of charging timestamp of the first charge record on the file. Should be used as a check value, to determine that all records have been correctly transmitted and/or used. Condition: Not present in a Notification File or if no Detail records are present. Format: YYYYMMDD HHMISS (see also 'Time-stamp') local time, and not UTC time, is used to determine the earliest call. Optional Field Usage: It is possible to read/write dates in number of seconds since 01.01.1970 00:00:00; for example, 12345. The internal representation is the format YYYYMMDDHHMISS anyway. This is just an optional input/output format conversion. Derivation: Mandatory.
FIRST_CHARGING_UTC_TIME_OFFSET	X(5)+/-HHMI	All timestamps are sender (VPLMN) local time. So that the time can be equated to time in the recipient (HPLMN) local time, the sender shall give the difference between local time and UTC time. UTC Time Offset = Local Time minus UTC Time. Can be used to translate the TRANSFER_CUTOFF_TIMESTAMP into a unified UTC time. This might be useful if a centralized rating and billing will take place. Example: Washington DC, USA 1000hrs 10/10/97 UTC Time 1500hrs 10/10/97 UTC Time Offset = 10 - 15 = -0500 Madrid, Spain 1600hrs 10/10/97 UTC Time 1500hrs 10/10/97 UTC Time Offset = 16 - 15 = +0100 Sydney, Australia 0100hrs 11/10/97 UTC Time 1500hrs 10/10/97 UTC Time Offset = (01 + 24) - 15 = +1000 **Note:** Where dates are different, 24 is added to the time of the greater date Derivation: Mandatory. Set by the first processor and left unchanged.

Table 1-45 (Cont.) Trailer Record Fields

Name	Format	Description
LAST_START_TIMESTAMP	YYYYMMDDHH MISS	The latest start of charging timestamp on any Basic Detail Record. It is not necessarily the start of charging timestamp of the last charge record on the file. Should be used as a check value, to determine that all records have been correctly transmitted and/or used. Might also be used to validate that all records are earlier then the given transfer cutoff timestamp (see header record). Condition: Not present in a Notification File or if no Detail records are present. Format: YYYYMMDD HHMISS (see also 'Time-stamp') local time, and not UTC time, is used to determine the earliest call. Optional Field Usage: It is possible to read/write dates in number of seconds since 01.01.1970 00:00:00; for example, 12345. The internal representation is the format YYYYMMDDHHMISS anyway. This is just an optional input/output format conversion. Derivation: Mandatory.
LAST_CHARGING_UTC_TIME_OFFSET	X(5)+/-HHMI	All timestamps are sender (VPLMN) local time. So that the time can be equated to time in the recipient (HPLMN) local time, the sender gives the difference between local time and UTC time. UTC Time Offset = Local Time minus UTC Time. Can be used to translate the TRANSFER_CUTOFF_TIMESTAMP into a unified UTC time. This might be useful if a centralized rating and billing will take place. Example: Washington DC, USA 1000hrs 10/10/97 UTC Time 1500hrs 10/10/97 UTC Time Offset = 10 - 15 = -0500 Madrid, Spain 1600hrs 10/10/97 UTC Time 1500hrs 10/10/97 UTC Time Offset = 16 - 15 = +0100 Sydney, Australia 0100hrs 11/10/97 UTC Time 1500hrs 10/10/97 UTC Time Offset = (01 + 24) - 15 = +1000 **Note:** Where dates are different, 24 is added to the time of the greater date. Derivation: Mandatory. Set by the first processor and left unchanged.

Table 1-45 (Cont.) Trailer Record Fields

Name	Format	Description
TOTAL_RETAIL_CHARGED_VALUE	9(15)	The sum of the Retail Charged Amount Value of any Basic Detail Record. The toll element of a charge is that portion related to the carrier and Destination. There will only be one toll charge present for all chained records. Should be used as a check value, to determine that all records have been correctly transmitted and/or used. Values: Space: No price given, like NULL in a database. Variable floating point format: Given value, might be 0.000. The floating decimal point must be set. Minimum: -99999999999999 Maximum: 999999999999999 Examples: '00000000125' for 125,00 '00000012.50' for 12,50 '-0012.56780' for -12,5678 Derivation: Mandatory.
TOTAL_WHOLESALE_CHARGED_VALUE	9(15)	The sum of the Wholesale Charged Amount Value of any Basic Detail Record contained in the batch. This is present for audit purposes only. Should be used as a check value, to determine that all records have been correctly transmitted and/or used. Values: Variable floating point format: Given value, might be 0.000. The floating decimal point must be set. Minimum: -99999999999999 Maximum: 999999999999999 Examples: '00000000125' for 125,00 '00000012.50' for 12,50 '-0012.56780' for -12,5678 Derivation: Mandatory.
TAP_TOTAL_CHARGE_VALUE	Decimal	Mandatory. Set by TAP input grammar.
TOTAL_TAX_VALUE	Decimal	Calculated. Auto-generated.
TAP_TOTAL_TAX_VALUE	Decimal	Mandatory. Set by TAP input grammar.
TAP_TOTAL_DISCOUNT_VALUE	Decimal	Mandatory. Set by TAP input grammar.
OPERATOR_SPECIFIC_INFO	String	Stores a key to identify the CDR used to generate a specific EDR. Useful for RAP or CIBER return. Optional. Default = ' ' Must be set by an iScript.
CIBER_FILLER	String	Optional.
CREATION_TIMESTAMP	Date	Optional. See the CIBER specification for usage.

Table 1-45 (Cont.) Trailer Record Fields

Name	Format	Description
CIBER_RECORD_TYPE	String	Optional. See the CIBER specification for usage.
SETTLEMENT_PERIOD	String	Optional. See the CIBER specification for usage.
CLEARINGHOUSE_ID	String	Optional. See the CIBER specification for usage.
CURRENCY	String	Optional. See the CIBER specification for usage.
SENDING_CLEARINGHOUSE_BID	String	Optional. See the CIBER specification for usage.
CIBER_R70 BATCH_TOTALS_SIGN	String	Optional. See the CIBER specification for usage.
CIBER_R70 ORIGINAL_TOTALS_SIGN	String	Optional. See the CIBER specification for usage.
ORIGINAL_SEQUENCE_NUMBER	Integer	Optional. See the CIBER specification for usage.
ORIGINAL_CREATION_TIMESTAMP	Date	Optional. See the CIBER specification for usage.
ORIGINAL_TOTAL_NUMBER_OF_RECORDS	Integer	Optional. See the CIBER specification for usage.
ORIGINAL_TOTAL_WHOLESALE_CHARGED_VALUE	Decimal	Optional. See the CIBER specification for usage.
NOTIFICATION_END_INDEX	Integer	Notification block end index.
AUDIT_CONTROL_INFO_START_INDEX	Integer	AuditControlInfo block start index.
AUDIT_CONTROL_INFO_END_INDEX	Integer	AuditControlInfo block end index.
DELAYED_ERROR_BLOCK	String	Stores the block name that has the fatal error.
TOTAL_CHARGE_VALUE_LIST	Block	*n* times. Mandatory. The TAP record is used by GSM operators and data clearinghouses to exchange roaming information.
TOTAL_CHARGE_VALUE	Decimal	Mandatory. Set by TAP grammar.
CHARGE_TYPE	String	Mandatory. Set by TAP grammar.
TOTAL_CHARGE_REFUND	String	Optional.
TOTAL_CHARGE_REFUND	Decimal	Optional.

Associated UTCOffset Record

Table 1-46 lists the fields in the Associated UTCOffset Record.

Table 1-46 Associated UTCOffset Record Fields

Name	Format	Description
UTCTIMEOFFSETCODE	Integer	Stores the UTC time offset code from TAP header.
UTCTIMEOFFSET	String	Stores the UTC time offset value from TAP header.

Associated Recentity Record

Table 1-47 lists the fields in the Associated Recentity Record.

Table 1-47 Associated Recentity Record Fields

Name	Format	Description
RECENTITYCODE	Integer	Stores the REC entity code from TAP header.
RECENTITYTYPE	Integer	Stores the REC entity type from TAP header.
RECENTITYID	String	Stores the REC entity ID from TAP header.

TAP Total Charge Value List

Mandatory. 0 .. *N* times.

Table 1-48 lists the TAP Total Charge Value List fields.

Table 1-48 TAP Total Charge Value List Fields

Name	Format	Description
TOTAL_CHARGE_VALUE	Decimal	Set by TAP grammar. Mandatory.
CHARGE_TYPE	String	Set by TAP grammar. Mandatory.
TOTAL_CHARGE_REFUND	Decimal	Optional.

Internal Service Control Container

This record is used internally by the framework.

Table 1-49 lists the fields in the Internal Service Control Container.

Table 1-49 Internal Service Controller Container Fields

Name	Format	Description
STREAM_NAME	String	Calculated.
OFFSET_GENERATION	Integer	Calculated.
SEQ_CHECK	Integer	Calculated.
SEQ_GENERATION	Integer	Calculated.
TRANSACTION_ID	Decimal	Calculated.
PROCESS_STATUS	Integer	Values: 0: Normal (default) 1: Recycling 2: Recycling test Mandatory. Calculated.

Customer Data Record

This record is used internally to save all customer-related attributes along with an event.

Table 1-50 lists the fields in the Customer Data Record.

Table 1-50 Customer Data Record Fields

Name	Format	Description
ACCOUNT_ID	String	Mandatory.
ACCOUNT_PARENT_ID	String	Optional.
ACCOUNT_NO	String	Mandatory.
CREATION_DATE	Date	Mandatory.
CURRENCY	String	Mandatory.
CUST_SEG_LIST	String	Mandatory.
RESIDENCE_TYPE	String	Optional.
SYSTEM_BRAND	String	Optional.
BILL_CYCLE	String	Values: 00-28 Mandatory.
BILL_FREQUENCY	Integer	Values: 1-12 Mandatory.
PAYMENT_TYPE	String	Optional.
BILL_STATE	Integer	Mandatory.
ACTG_LAST_DATE	Date	Mandatory.
ACTG_NEXT_DATE	Date	Mandatory.
ACTG_FUTURE_DATE	Date	Mandatory.
ACTG_USED_DATE	Date	Mandatory.
BILL_LAST_DATE	Date	Mandatory.
BILL_NEXT_DATE	Date	Mandatory.
BILL_FUTURE_DATE	Date	Mandatory.
RESOURCE_LIST	String	Optional.
LEAST_COST	Integer	Optional.
PROMOTIONAL_SAVING	Integer	Optional.
PROMOTION	Integer	Optional.

Purchased Charge Offers

Table 1-51 lists the fields in the Purchase Products block.

Table 1-51 Purchased Products Fields

Name	Format	Description
PRODUCT_NAME	String	Mandatory.
USAGE_START	Date	Optional.
USAGE_END	Date	Optional.

Table 1-51 (Cont.) Purchased Products Fields

Name	Format	Description
QUANTITY	Decimal	Optional.
OFFERING_POID	String	Optional.
OVERRIDDEN_OFFERING_POID	String	Optional.
NODE_LOCATION	String	Mandatory.
DEAL_NAME	String	Mandatory. **Important:** The DEAL_NAME value is not stored in memory nor retained in the EDR. Therefore, this value will not appear in the EDR dump.
PLAN_NAME	String	Mandatory.
PRODUCT_TYPE	Integer	Defines system or normal product; for example, 603 or 602.
RATEPLAN_NAME	String	Mandatory.
PRIORITY	Integer	Mandatory.
PRODUCT_ID	String	Optional.
SERVICE_TYPE	String	For example, **/service/telco/gsm/data**. Mandatory.
SERVICE_ID	String	Optional.
SERVICE_PROMO_CODE	String	Optional.
SERVICE_VENDOR	String	Optional.
SERVICE_SOURCE	String	Optional.
SERVICE_LOGIN	String	Mandatory.
SERVICE_MSISDN	String	Optional.
SERVICE_IMSI	String	Optional.
SERVICE_STATUS	String	Mandatory.
SERVICE_USED_ITEM_POID	String	Mandatory.
NETWORK_IDENT	String	Optional.
FIRST_USAGE_INDICATOR	Integer	Specifies whether the charge offer is configured to start when first used and the first-usage validity period status. Optional.

Extended Rating Attributes List

Table 1-52 lists the fields in the Extended Rating Attributes List.

Table 1-52 Extended Rating Attributes List Fields

Name	Format	Description
PROFILE	String	Mandatory.
LABEL	String	Optional.

Profile Attributes

Table 1-53 lists the Profile Attributes fields.

Table 1-53 Profile Attributes Fields

Name	Format	Description
KEY	String	Mandatory.
VALUE	String	Mandatory.

Alias List

Table 1-54 lists the Alias list field.

Table 1-54 Alias List Field

Name	Format	Description
ALIAS_NAME	String	Mandatory.

Discount List

Table 1-55 lists the Discount List fields.

Table 1-55 Discount List Fields

Name	Format	Description
BALANCE_GROUP_ID	String	Mandatory.
DISCOUNT_OWNER_ACCT_ID	String	Mandatory.
DISCOUNT_OWNER_ID	String	Mandatory.
DISCOUNT_OWNER_TYPE	String	Mandatory.

Purchased Discounts

Table 1-56 lists the Purchased Discounts fields.

Table 1-56 Purchased Discounts Fields

Name	Format	Description
DISCOUNT_ID	String	Mandatory.
DISCOUNT_MODEL	String	Mandatory.
PURCHASE_START	Date	Mandatory.
PURCHASE_END	Date	Mandatory.
USAGE_START	Date	Optional.
USAGE_END	Date	Optional.
PRIORITY	Integer	Mandatory.

Table 1-56 (Cont.) Purchased Discounts Fields

Name	Format	Description
MODE	Integer	Mandatory.
VALID_FLAG	Integer	Mandatory.
TYPE	Integer	Mandatory.
OFFERING_POID	String	Mandatory.
NODE_LOCATION	String	Mandatory.
STATUS	Integer	Mandatory.
QUANTITY	Integer	Mandatory.
FLAGS	Integer	Mandatory.
SCALE	Decimal	Mandatory.
FIRST_USAGE_INDICATOR	Integer	Specifies whether the charge offer is configured to start when first used and the first-usage validity period status. Optional.

Sponsor List

Table 1-57 shows the Sponsor List fields.

Table 1-57 Sponsor List Fields

Name	Format	Description
BALANCE_GROUP_ID	String	Mandatory.
SPONSOR_OWNER_ACCT_ID	String	Mandatory.
SPONSOR_OWNER_ID	String	Mandatory.
SPONSOR_OWNER_TYPE	String	Mandatory.

Sponsorship Details

Table 1-58 lists the Sponsorship Details fields.

Table 1-58 Sponsorship Details Fields

Name	Format	Description
SPONSORSHIP_ID	String	Mandatory.
DISCOUNT_MODEL	String	Mandatory.
VALID_FLAG	Integer	Mandatory.

Plan List

Table 1-59 lists the Plan List field.

Table 1-59 Plan List Field

Name	Format	Description
PLAN_ID	String	Mandatory.

Balance Group

Table 1-60 lists the Balance Group field.

Table 1-60 Balance Group Field

Name	Format	Description
BALANCE_GROUP_ID	String	Mandatory.

Balance Element

Table 1-61 lists the Balance Element fields.

Table 1-61 Balance Element Fields

Name	Format	Description
RESOURCE_ID	Integer	Mandatory.
CURR_BAL	Decimal	Mandatory.
CREDIT_FLOOR	Decimal	Mandatory.
CREDIT_LIMIT	Decimal	Mandatory.
RESERVED_AMOUNT	Decimal	Mandatory.
Name	Format	Description

Associated CIBER Extension Record

See the CIBER 2.5 specification for explanations of the fields listed in Table 1-62.

Table 1-62 Associated CIBER Extension Record Fields

Name	Format	Description
RECORD_TYPE	String	Default = 701 (this is not yet the final value). Mandatory.
RECORD_NUMBER	Integer	Auto-generated. Mandatory.
FILLER	String	The filler for CIBER optional fields at the end of a record. Optional.
NO_OCC	Integer	Flag to suppress OCC (type 50 or 52) record-creation process. Optional.

Table 1-62 (Cont.) Associated CIBER Extension Record Fields

Name	Format	Description
INTERN_MOBILE_ID_NO	String	None
INTERN_CALLED_NO	String	None
INTERN_MSISDN_MDN	String	None
INTERN_CALLER_ID	String	None
INTERN_ROUTING_NO	String	None
INTERN_TLDN_NO	String	None
CIBER_RECORD_TYPE	String	Optional.
RETURN_CODE	String	Optional.
RETURN_REASON_CODE	String	Optional.
INVALID_FIELD_ID	String	Optional.
HOME_CARRIER_SID	String	Optional.
MOBILE_ID_NO_LENGTH	Integer	Optional.
MOBILE_ID_NO	String	Optional.
MOBILE_ID_NO_OVERFLOW	String	Optional.
ELECTRONIC_SERIAL_NO	String	Optional.
CALL_DATE	Date	Optional.
SERVING_CARRIER_SID	String	Optional.
TOTAL_CHARGE_AND_TAX	Decimal	Optional.
TOTAL_STATE_TAX	Decimal	Optional.
TOTAL_LOCAL_TAX	Decimal	Optional.
CALL_DIRECTION	String	Optional.
CALL_COMPLETION_INDICATOR	String	Optional.
CALL_TERMINATION_INDICATOR	String	Optional.
CALLED_NO_LENGTH	Integer	Optional.
CALLED_NO	String	Optional.
CALLED_NO_OVERFLOW	String	Optional.
TEMP_LOCAL_DIRECTORY_NO	String	Optional.
CURRENCY_TYPE	String	Optional.
ORIG_BATCH_SEQ_NO	Integer	Optional.
INITIAL_CELL_SITE	String	Optional.
TIME_ZONE_INDICATOR	String	Optional.
DAYLIGHT_SAVINGS_INDICATOR	String	Optional.
MSG_ACCOUNTING_DIGITS	String	Optional.
SSU_CONNECT_TIME	Date	Optional.
SERVING_STATE	String	Optional.
RECV_CARRIER_SID	String	Optional.
TRANS_CODE1	String	Optional.

Table 1-62 (Cont.) Associated CIBER Extension Record Fields

Name	Format	Description
TRANS_CODE2	String	Optional.
SENDING_CARRIER_SID	String	Optional.
CHARGE_NO_1_IND	String	Optional.
CHARGE_NO_1_CONNECT_TIME	Date	Optional.
CHARGE_NO_1_CHARGEABLE_TIME	String	Optional.
CHARGE_NO_1_ELAPSED_TIME	String	Optional.
CHARGE_NO_1_RATE_PERIOD	String	Optional.
CHARGE_NO_1_MULTIRATE_PERIOD	String	Optional.
CHARGE_NO_1	Decimal	Optional.
CHARGE_NO_2_IND	String	Optional.
CHARGE_NO_2_CONNECT_TIME	Date	Optional.
CHARGE_NO_2_CHARGEABLE_TIME	String	Optional.
CHARGE_NO_2_ELAPSED_TIME	String	Optional.
CHARGE_NO_2_RATE_PERIOD	String	Optional.
CHARGE_NO_2_MULTIRATE_PERIOD	String	Optional.
CHARGE_NO_2	Decimal	Optional.
CHARGE_NO_3_IND	String	Optional.
CHARGE_NO_3_CONNECT_TIME	Date	Optional.
CHARGE_NO_3_CHARGEABLE_TIME	String	Optional.
CHARGE_NO_3_ELAPSED_TIME	String	Optional.
CHARGE_NO_3_RATE_PERIOD	String	Optional.
CHARGE_NO_3_MULTIRATE_PERIOD	String	Optional.
CHARGE_NO_3	Decimal	Optional.
CHARGE_NO_4_IND	String	Optional.
CHARGE_NO_4_CONNECT_TIME	Date	Optional.
CHARGE_NO_4_CHARGEABLE_TIME	String	Optional.
CHARGE_NO_4_ELAPSED_TIME	String	Optional.
CHARGE_NO_4_RATE_PERIOD	String	Optional.
CHARGE_NO_4_MULTIRATE_PERIOD	String	Optional.
CHARGE_NO_4	Decimal	Optional.
CHARGE_NO_1_SURCHARGE_IND	String	Optional.
CHARGE_NO_2_SURCHARGE_IND	String	Optional.
CHARGE_NO_3_SURCHARGE_IND	String	Optional.
CHARGE_NO_4_SURCHARGE_IND	String	Optional.
TOLL_CONNECT_TIME	Date	Optional.
TOLL_CHARGEABLE_TIME	String	Optional.
TOLL_ELAPSED_TIME	String	Optional.

Table 1-62 (Cont.) Associated CIBER Extension Record Fields

Name	Format	Description
TOLL_TARIFF_DESC	String	Optional.
TOLL_RATE_PERIOD	String	Optional.
TOLL_MULTIRATE_PERIOD	String	Optional.
TOLL_RATE_CLASS	String	Optional.
TOLL_FROM_RATING_NPA_NXX	String	Optional.
TOLL_CHARGE	Decimal	Optional.
TOLL_STATE_TAX	Decimal	Optional.
TOLL_LOCAL_TAX	Decimal	Optional.
TOLL_NETWORK_CARRIER_ID	String	Optional.
MSID_INDICATOR	String	Optional.
MSID	String	Optional.
MSISDN_MDN_LENGTH	Integer	Optional.
MSISDN_MDN	String	Optional.
ESN_IMEI_INDICATOR	String	Optional.
ESN_IMEI	String	Optional.
CALLER_ID_LENGTH	Integer	Optional.
CALLER_ID	String	Optional.
ROUTING_NO_LENGTH	Integer	Optional.
ROUTING_NO	String	Optional.
TLDN_NO_LENGTH	Integer	Optional.
TLDN_NO	String	Optional.
AIR_CONNECT_TIME	Date	Optional.
AIR_CHARGEABLE_TIME	String	Optional.
AIR_ELAPSED_TIME	String	Optional.
AIR_RATE_PERIOD	String	Optional.
AIR_MULTIRATE_PERIOD	String	Optional.
AIR_CHARGE	Decimal	Optional.
OTHER_CHARGE_1_INDICATOR	String	Optional.
OTHER_CHARGE_1	Decimal	Optional.
CALLED_COUNTRY	String	Optional.
SERVING_COUNTRY	String	Optional.
TOLL_RATING_POINT_LENGTH	Integer	Optional.
TOLL_RATING_POINT	String	Optional.
FEATURE_USED_AFTER_HO_IND	String	Optional.
OCC_START_DATE	Date	Optional.
OCC_CHARGE	Decimal	Optional.
FET_EXEMPT_INDICATOR	String	Optional.

Table 1-62 (Cont.) Associated CIBER Extension Record Fields

Name	Format	Description
PASS_THROUGH_CHARGE_IND	String	Optional.
CONNECT_TIME	Date	Optional.
RECORD_USE_INDICATOR	String	Optional.
OCC_DESCRIPTION	String	Optional.
OCC_END_DATE	Date	Optional.
RECORD_CREATE_DATE	Date	Optional.
SEQ_INDICATOR	String	Optional.
OCC_INTERVAL_INDICATOR	String	Optional.
EVENT_DATE	Date	Optional.
MIN_ESN_APP_INDICATOR	String	Optional.
R70_RECORD_USE_INDICATOR	String	Optional.
EVENT_TIME	Date	Optional.

Discount Balance Packet

Table 1-63 lists the Discount Balance Packet fields.

Table 1-63 Discount Balance Packet Fields

Name	Format	Description
DISCOUNT_KEY	String	Discount key (normally the account ID).
ACCOUNT_ID	Integer	Related account ID.
RESOURCE_ID	Integer	BRM mapped balance element ID.
DISCOUNT_STEP	Integer	Alternative key if balance element ID 0.
DISCOUNT_MASTER	Integer	Alternative key if balance element ID 0.
UPDATE_LEVEL	String	Always empty. Supports backward compatibility.
SERVICE_ID	Integer	Supports backward compatibility.

Aggregation Period

Table 1-64 lists the Aggregation Period fields.

Table 1-64 Aggregation Period Fields

Name	Format	Description
PERIOD	String	The accounting cycle. *YYYYMMDD*.
ITEM_POID	String	POID of the item that identifies the accounting cycle.
CREATED	String	Creation date.
TOTAL_CHARGE	Decimal	Total charge.

Table 1-64 (Cont.) Aggregation Period Fields

Name	Format	Description
TOTAL_QUANTITY	Decimal	Total quantity based on RUMs in the discount filter.
TOTAL_EVENT	Decimal	Sum of events based on charge packets.
GRANTED_CHARGE	Decimal	The discounted charge.
GRANTED_QUANTITY	Decimal	The discounted quantity.
FRAME_CHARGE	Decimal	Total charge of the discount frame, based on the frame.
FRAME_QUANTITY	Decimal	Total quantity in the discount frame based on RUMs in the discount filter.
FRAME_EVENT	Decimal	Total events of the discount frame, based on charge packets.

Discount Packet

Table 1-65 lists the Discount Packet fields.

Table 1-65 Discount Packet Fields

Name	Format	Description
RECORD_TYPE	String	Mandatory.
CREATED	String	Creation date.
OBJECT_ID	String	Discount/sponsor object ID.
OBJECT_TYPE	String	Discount/sponsor object that generated the event.
OBJECT_ACCOUNT	Integer	POID of the discount owner.
OBJECT_OWNER_ID	Integer	POID type of the discount owner.
OBJECT_OWNER_TYPE	String	POID type of the discount owner.
DISCOUNTMODEL	String	Discount.
DISCOUNTRULE	String	Discount rule.
DISCOUNTSTEPID	Integer	Discount step ID.
DISCOUNTBALIMPACTID	Integer	Discount balance impact ID.
TAX_CODE	String	Tax code.
GRANTED_AMOUNT	Decimal	Granted discount/sponsorship amount. Can be currency or noncurrency.
GRANTED_AMOUNT_ORIG	Decimal	Original granted discount/sponsorship amount. Used when exchange rate is configured.
GRANTED_QUANTITY	Decimal	The discount base value used to compute the granted amount.
AMOUNT	Decimal	Discounted currency amount.
PIN_PERCENT	Decimal	Percent value filled from charge packet
QUANTITY	Decimal	Discounted noncurrency amount.
QUANTITY_FROM	Decimal	Discounted quantity start value.
QUANTITY_TO	Decimal	Discounted quantity end value.
VALID_FROM	Date	Grant case, valid-from date.

Table 1-65 (Cont.) Discount Packet Fields

Name	Format	Description
VALID_TO	Date	Grant case, valid-to date.
VALID_FROM_DETAIL	Integer	First Usage Offset and Unit for valid-from date.
VALID_TO_DETAIL	Integer	First Usage Offset and Unit for valid-to date.
CYCLE_OFFSET	Integer	Charge offer cycle that identifies a grant's validity period.
BALANCE_GROUP_ID	Integer	Balance group ID.
SERVICE_CODE	String	Service code.
RESOURCE_ID	Integer	Balance Element ID.
RESOURCE_ID_ORIG	Integer	Original balance element ID. Used when exchange rate is configured.
ZONEMODEL_CODE	String	Zone model.
IMPACT_CATEGORY	String	Impact category.
TIMEZONE_CODE	String	Time-zone code.
TIMEMODEL_CODE	String	Time model code.
SERVICE_CLASS	String	Service class.
PRICEMODEL_CODE	String	Pricing code.
RUM	String	RUM.
RATETAG	String	Rate tag.
RATEPLAN	String	Charge.
GLID	String	G/L ID.
OFFERING_POID	String	Purchased discount offer POID.
NODE_LOCATION	String	Node location.
INTERN_PACKET_INDEX	Integer	Packet ID.
INTERN_SRC_PACKET_INDEX	Integer	Source packet ID.
INTERN_RUM_ID	Integer	RUM ID passed in and out, used in real-time pipeline only.
INTERN_DISC_MATCH_FACTOR	Decimal	Discount match factor (the percentage of usage discounted).
INTERN_TOTAL_MATCH_FACTOR	Decimal	Total discounted match factor (the total percentage of usage discounted).
DEFERRED_AMOUNT	Decimal	Deferred amount.

Discount Sub-Balance Packet

Table 1-66 lists the Discount Sub-Balance Packet fields.

Table 1-66 Discount Sub-Balance Packet Fields

Name	Format	Description
REC_ID	Integer	Record ID.
VALID_FROM	Date	Validity start time.
VALID_TO	Date	Validity end time.

Table 1-66 (Cont.) Discount Sub-Balance Packet Fields

Name	Format	Description
AMOUNT	Decimal	Amount.
CONTRIBUTOR	String	Contributor.
NEXT_BAL	Date	Next balance.
DELAYED_BAL	Decimal	Delayed balance.
GRANTOR	String	The charge offer or discount offer that granted this balance element.
VALID_FROM_DETAILS	Date	Sub-balance start time mode (such as first-usage or relative) and relative offset and unit.
VALID_TO_DETAILS	Date	Sub-balance end time mode (such as relative) and relative offset and unit.
GRANT_VALID_FROM	Date	Grant validity start time.
GRANT_VALID_TO	Date	Grant validity end time.

Associated SMS Extension Record

Table 1-67 lists the Associated SMS Extension Record fields.

Table 1-67 Associated SMS Extension Record Fields

Name	Format	Description
RECORD_TYPE	String	Mandatory. Default = 580.
RECORD_NUMBER	Integer	Mandatory. Auto-generated.
CONTENT_INDICATOR	String	Optional.
ORIGINATING_SWITCH_IDENTIFICATION	String	Optional.
DESTINATION_SWITCH_IDENTIFICATION	String	Optional.
PROVIDER_ID	String	Optional.
SERVICE_ID	String	Optional.
DEVICE_NUMBER	String	Optional.
PORT_NUMBER	String	Optional.
DIALED_DIGITS	String	Optional.

Associated MMS Extension Record

Table 1-68 lists the Associated MMS Extension Record fields.

Table 1-68 Associated MMS Extension Record Fields

Name	Format	Description
RECORD_TYPE	String	Mandatory. Default = 590.
RECORD_NUMBER	Integer	Mandatory. Auto-generated.
ACCOUNT_STATUS_TYPE	String	Optional.
PRIORITY	String	Optional.
MESSAGE_CONTENT	String	Optional.
MESSAGE_ID	String	Optional.
STATION_IDENTIFIER	String	Optional.
FC_INDICATOR	String	Optional.
CORRELATION_ID	String	Optional.
CELL_ID	String	Optional.
B_CELL_ID	String	Optional.
A_TERM_CELL_ID	String	Optional.
DEVICE_NUMBER	String	Optional.
PORT_NUMBER	String	Optional.

SGSN Information

Table 1-69 lists the SGSN Information field.

Table 1-69 SGSN Information Field

Name	Format	Description
SGSN_ADDRESS	String	Mandatory.

Profile Event Ordering

Table 1-70 lists the Profile Event Ordering fields.

Table 1-70 Profile Event Ordering Fields

Name	Format	Description
RECORD_TYPE	String	Mandatory. Must be set to 850.
RECORD_NUMBER	Integer	Mandatory.
BAL_GRP_POID	String	Mandatory.
CRITERIA_NAME	String	Mandatory.
PROFILE_POID	String	Mandatory.
BILLING_CYCLE_TIMESTAMP	Date.	Mandatory.

Associated Roaming Extension Record

Table 1-71 lists the Associated Roaming Extension Record fields.

Table 1-71 Associated Roaming Extension Record Fields

Name	Format	Description
RECORD_TYPE	String	None
RECORD_NUMBER	Integer	None
TAP_FILE_SEQ_NO	Integer	None
RAP_FILE_SEQ_NO	Integer	None
RAP_RECORD_TYPE	String	None
SENDER	String	None
RECIPIENT	String	None
TAP_FILE_PATH	String	None
START_MISSING_SEQ_NUM	Integer	None
END_MISSING_SEQ_NUM	Integer	None
SUSPENSION_TIME	Date	None
PORT_NUMBER	String	None
TOTAL_TAX_REFUND	Decimal	None
TOTAL_DISCOUNT_REFUND	Decimal	None
GUARANTEED_BIT_RATE	String	None
MAXIMUM_BIT_RATE	String	None
HSCSD_INDICATO	String	None
SMS_ORIGINATOR	String	None
SMS_DESTINATION_NUMBER	String	None
DISCOUNTABLE_AMOUNT	Decimal	None
DISCOUNT_CODE	Integer	None
NETWORKACCESS_IDENTIFIE	String	None
ISM_SIGNALLING_CONTEXT	Integer	None
IMSI	String	None
HOME_BID	String	None
HOMELOCATION_DESCRIPTION	String	None
MOBILE_ID_NUMBER	String	None
MOBILE_DIR_NUMBER	String	None
TOTAL_ADVISEDCHARGE	Decimal	None
TOTAL_ADVISEDCHARGE_REFUND	Decimal	None
TOTAL_COMMISSION	Decimal	None
TOTAL_COMMISSION_REFUND	Decimal	None
ITEM_OFFSET	Integer	None
ERROR_CODE	Integer	None

Table 1-71 (Cont.) Associated Roaming Extension Record Fields

Name	Format	Description
TOTAL_SEVERE_RETURN_VALUE	Decimal	None
RETURN_DETAILS_COUNT	Integer	None
CLIR_INDICATOR	String	None
TAP_CURRENCY	String	Currency used for TAP3 and TAP 311.

Associated RAP Extension

Table 1-72 lists the Associated RAP Extension fields.

Table 1-72 Associated RAP Extension Fields

Name	Format	Description
PATH_ITEMID	Integer	None
ITEM_OCCURRENCE	Integer	None
ITEM_LEVEL	Integer	None

Total Advised Charge Value List

Table 1-73 lists the Total Advised Charge Value List fields.

Table 1-73 Total Advised Charge Value List Fields

Name	Format	Description
TOTAL_ADVISEDCHARGE	Decimal	None
TOTAL_ADVISEDCHARGE_REFUND	Decimal	None
ADVISED_CHARGE_CURRENCY	String	Optional. AdvisedChargeCurrency item.
TOTAL_COMMISSION;	Decimal	None
TOTAL_COMMISSION_REFUND	Decimal	None

Field Usage

Roaming

The following conditions are checked to determine if the usage record is for roaming:

```
if [DETAIL.USAGE_DIRECTION] = 2
       then Roaming = TRUE; Roaming-Type = MOC
elseif [DETAIL.USAGE_DIRECTION] = 3
       then Roaming = TRUE; Roaming-Type = MTC
else   Roaming = FALSE
```

International-Call

The following conditions are checked to determine if the usage record is for an international call:

```
if [DETAIL.CONNECT_SUB_TYPE] = '04'
       then International-Call=TRUE
```

CLI Normalization

The result of the mapping operation (performed either within an input module or within an iScript) must be written/copied to the following internal container fields:

DETAIL.A_NUMBER -> normalize -> DETAIL.INTERN_A_NUMBER_ZONE

DETAIL.B_NUMBER -> normalize -> DETAIL.INTERN_B_NUMBER_ZONE

DETAIL.C_NUMBER -> normalize -> DETAIL.INTERN_C_NUMBER_ZONE

The original values within the Basic Detail Record are kept unchanged.

The following fields determine how and which a normalization function should be carried out:

DETAIL.A_NUMBERING_PLAN -> normalize DETAIL.A_NUMBER

DETAIL.B_NUMBERING_PLAN -> normalize DETAIL.B_NUMBER

DETAIL.C_NUMBERING_PLAN -> normalize DETAIL.C_NUMBER

if DETAIL.x_NUMBERING_PLAN between 1 and 9 -> use "ISDN, MSISDN"

if DETAIL.x_NUMBERING_PLAN between A and B -> use "Ipv4, IPv6"

if DETAIL.x_NUMBERING_PLAN = 0 -> no normalization

ISDN, MSISDN

The following rules normalize the A number and B number. All CLIs are normalized to match the international format: *International_access_codeCountry_codeNational_destination_codeSubscriber_number*; for example, '00491711234567' or '004980012345'.

To handle a flexible international format, the following parameters must be set prior to the normalization:

International_access_code (iac): international access code; for example, '00'

> **Note:**
>
> Multiple iacs might be defined.

International_access_code_sign (iacs): international access code sign; for example, '+'

Country_code (cc): country code of the home country; for example, '49'

National_destination_access_code (ndac): national destination access code for long distance; for example, '0'

National_destination_code (ndc): default national destination code; for example, '172' (only for special mobile calls)

Normal-Call

Table 1-74 lists the Normal-Call fields.

Table 1-74 Normal-Call Fields

Item	Description
A#:	[DETAIL.A_NUMBER, X(40)]
B#:	[DETAIL.B_NUMBER, X(40)]

1. If <empty>-> replace with '<iac><cc>' (break)
2. If Prefix = '<iacs>'-> replace with '<iac>' (continue)
3. If Prefix = '<iac>'-> do nothing (break)
4. If Prefix = '<ndac>'-> replace with '<iac><cc>' (break)
5. If [TYPE_OF_NUMBER] = 1-> prefixing '<iac>' (break)

Roaming-Call (Mapping only for Zone- and PrefixDesc.-Determination)

An Associated GSM/Wireline Extension Record must exist.

Table 1-75 lists the Roaming-Call fields.

Table 1-75 Roaming-Call Fields

Item	Type	Description
A#:	MOC	-> '0000' + Left([ASS_GSMW_SE.ORIGINATING_SWITCH_ID], 5)
A#:	MTC	-> Normalization as for normal calls
B#:	MOC	-> Normalization as for normal calls
B#:	MTC	-> '0000' + Left([ASS_GSMW_SE.TERMINATING_SWITCH_ID], 5)

If on the input side there is only one single MSC_ID or PLMN_ID available, the related input module has to map this single value into both fields (Originating and Terminating).

Special-Mobile-Call:

A/B-MODIFICATION_INDICATOR = '04'

Normalize to: <iac><cc>'0'<ndc><number>; for example, '0049017222255'

Table 1-76 lists the Special-Mobile-Call fields.

Table 1-76 Special-Mobile-Call Fields

Item	Description
A#:	[DETAIL.A_NUMBER, X(40)]
B#:	[DETAIL.B_NUMBER, X(40)]

1. If <empty>-> replace with '<iac><cc>' (break)
2. If Prefix = '<iacs>'-> replace with '<iac>' (continue)

3. If Prefix = '<iac><cc>'-> replace with '<iac><cc>0' (break)
4. If Prefix = '<ndac>'-> prefixing '<iac><cc>' (break)
5. If Prefix != '<cc>'-> prefixing '<iac><cc>0<ndc>' (break)
6. If [TYPE_OF_NUMBER] = 1-> prefixing '<iac>' (break)

IPv4, IPv6

If the A# (source ip) and B# (destin ip) are carrying ip-addresses, they are normalized to zero-leading-tokens without the dots or colon; for example, '192.168.10.1' is normalized to '192168010001'.

Notations are listed in Table 1-77.

Table 1-77 IPv4 and IPv6 Attributes

Item	Type	Description
IPv4:	nnn.nnn.nnn.nnn	n = (0..9)
IPv6:	hhhh:hhhh:hhhh:hhhh:hhhh:hhhh:hhhh:hhhh	h = (0..F)
IPv4 as v6:	0000:0000:0000:0000:0000:0000:nnn.nnn.nnn.nnn	None
or	0000:0000:0000:0000:0000:FFFF:nnn.nnn.nnn.nnn	None

IPv4 Record

Table 1-78 lists the IPv4 Record fields.

Table 1-78 IPv4 Record Fields

Item	Type	Description
A#:	[DETAIL.A_NUMBER, X(40)]	(source ip-address)
B#:	[DETAIL.B_NUMBER, X(40)]	(destination ip-address)

1. Determine the four decimal ip-tokens
2. Fill each token with leading zeros (up to 3-digits)
3. Remove all dots '.'

IPv6 Record

Table 1-79 lists the IPv6 Record fields.

Table 1-79 IPv6 Record Fields

Item	Type	Description
A#:	[DETAIL.A_NUMBER, X(40)]	(source ip-address)
B#:	[DETAIL.B_NUMBER, X(40)]	(destination ip-address)

1. Determine the eight hexadecimal ip-tokens
2. Fill each token with leading zeros (up to 4-digits)
3. Remove all colons ':'

IPv4 as IPv6 Record

Table 1-80 lists the IPv4 vs IPv6 Record fields.

Table 1-80 IPv4 as IPv6 Record Fields

Item	Type	Description
A#:	[DETAIL.A_NUMBER, X(40)]	(source ip-address)
B#:	[DETAIL.B_NUMBER, X(40)]	(destination ip-address)

1. Determine the six fix hexadecimal v6 ip-tokens
2. Fill each token with leading zeros (up to 4-digits)
3. Remove all colons ':'
4. Determine the four decimal ip-tokens at the end of the address
5. Take the first two v4 tokens and convert them to hexadecimal (for example, '192.168' = 192*256^1 + 168*256^0 = 49320 = 'C0A8')
6. Take the last two v4 tokens and convert them to hexadecimal
7. Replace the origin v4 address by the two calculated v6 equivalents filled with leading zeros

UsageClass (CallClass)

The result of the mapping operation (performed either within an input module or within an early iScript) must be written/copied to the following internal container fields:

DETAIL.USAGE_CLASS -> mapping -> DETAIL.INTERN_USAGE_CLASS

The original values within the Basic Detail Record are kept unchanged.

The following rules apply to determine the external UsageClass value:

Always: Value of the field [DETAIL.USAGE_CLASS]

ServiceCode / ServiceClass

The result of the mapping operation (performed either within an input module or within an early iScript) must be written/copied to the following internal container fields:

DETAIL.BASIC_SERVICE -> mapping -> DETAIL.INTERN_SERVICE_CODE

mapping -> DETAIL.INTERN_SERVICE_CLASS

The original values within the Basic Detail Record are kept unchanged.

The following rules apply to determine the external ServiceCode value:

Always: Value of the field [DETAIL.BASIC_SERVICE];

containing [SERVICE_TYPE, X(1)]+ [SERVICE_CODE, X(2)]

2

List of Pipeline Manager Modules, iScripts, and iRules

This chapter lists the Oracle Communications Billing and Revenue Management Pipeline Manager modules.

Topics in this document:

- Pipeline Manager Modules

For information about placement of modules in a pipeline, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Pipeline Manager Modules

Table 2-1 lists the Pipeline Manager modules with descriptions.

Table 2-1 Pipeline Manager Modules

Module	Description
Controller	Controls and monitors the pipeline framework.
Database Connect (DBC)	Provides database connections for other modules.
DAT_AccountBatch	Provides customer data from the BRM database.
DAT_AccountRealtime	Provides data to a real-time discounting pipeline. See "Configuring a Real-Time Discounting Pipeline" in *BRM Setting Up Pipeline Rating and Discounting*.
DAT_BalanceBatch	Maintains balance information in the Pipeline Manager memory. See "Configuring Discounting Modules and Components" in *BRM Setting Up Pipeline Rating and Discounting*.
DAT_BalanceRealtime	Retrieves current balance information from the BRM database and supplies the data to the real-time discounting pipeline. See "Configuring a Real-Time Discounting Pipeline" in *BRM Setting Up Pipeline Rating and Discounting*.
DAT_Calendar	Provides special day calendar data for the FCT_MainRating module.
DAT_Currency	Converts currency symbols to numeric values.
DAT_Dayrate	Provides special day rate data for the FCT_Dayrate module.
DAT_Discount	Provides data for the FCT_Discount module and the FCT_DiscountAnalysis module. See "Configuring Discounting Modules and Components" in *BRM Setting Up Pipeline Rating and Discounting*.
DAT_ExchangeRate	Provides currency exchange rate data for the FCT_ExchangeRate module.

Table 2-1 (Cont.) Pipeline Manager Modules

Module	Description
DAT_InterConnect	Provides network configuration data for the FCT_CarrierIcRating module.
DAT_ItemAssign	Returns the item POID for an item tag to the FCT_ItemAssign and FCT_Billing Record modules.
DAT_Listener	Listens to business events from BRM and provides data to the DAT_AccountBatch and DAT_Discount modules.
DAT_ModelSelector	Provides model selector rules to other modules. See "Configuring Pipeline Rating" and "Configuring Discounting Modules and Components" in *BRM Setting Up Pipeline Rating and Discounting*.
DAT_NOSP	Provides data for mapping network source and destinations to new values for the FCT_NOSP module. See "Identifying the Network Operator/Service Provider" in *BRM Setting Up Pipeline Rating and Discounting*.
DAT_NumberPortability	Provides number portability data to the FCT_NumberPortability module. See "Setting Up Number Portability" in *BRM Setting Up Pipeline Rating and Discounting*.
DAT_PortalConfig	Provides data for mapping phone number prefixes to descriptions, used by the FCT_PrefixDesc module. See "Creating Call Destination Descriptions" in *BRM Setting Up Pipeline Rating and Discounting*.
DAT_PriceModel	Provides price model data for the FCT_MainRating module. See "About Pipeline Rating" in *BRM Setting Up Pipeline Pricing*.
DAT_Rateplan	Provides charge data for the FCT_MainRating module. See "Configuring Pipeline Rating" in *BRM Setting Up Pipeline Rating and Discounting*.
DAT_Recycle	Used by standard recycling and Suspense Manager EDR to recycle EDRS. See "Configuring Standard Recycling" in *BRM Suspending and Recycling Pipeline EDRs*.
DAT_ScenarioReader	Provides aggregation scenario data for the FCT_AggreGate module. See "Setting Up Pipeline Aggregation" in *BRM Setting Up Pipeline Rating and Discounting*.
DAT_TimeModel	Provides time model, time zone, and day code data for the FCT_Mainrating module.
DAT_USC_Map	Provides usage scenario (USC) mapping data. It retrieves USC mapping data from the Pipeline Manager database or an ASCII file for the FCT_USC_Map module.
DAT_Zone	Provides zone data for the FCT_MainRating module.
EDR Factory	Generates and allocates memory to EDR Containers.
Event Handler	Starts external programs.

Table 2-1 (Cont.) Pipeline Manager Modules

Module	Description
EXT_InEasyDB	Handles pipeline input from a database. See "Configuring EDR Input Processing" in *BRM Setting Up Pipeline Rating and Discounting*. Configure this module as a submodule of the INP_GenericStream module. See "INP_GenericStream" for more information.
EXT_InFileManager	Performs file handling for pipeline input from files. See "Configuring EDR Input Processing" in *BRM Setting Up Pipeline Rating and Discounting*. Configure this module as a submodule of the INP_GenericStream module. See "INP_GenericStream" in *BRM Pipeline Manager Reference*.
EXT_OutFileManager	Handles files for the OUT_Generic_Stream and OUT_Reject modules. See "Configuring EDR Output Processing" in *BRM Setting Up Pipeline Rating and Discounting*.
Pipeline Dispatcher	Parses CDR files from a single input directory to multiple pipelines.
FCT_Account	Adds customer data to an EDR.
FCT_AccountRouter	For a multischema system, finds the database schema for the customer and routes the EDRs to the appropriate pipeline. See "Using Pipeline Manager with Multiple Database Schemas" in *BRM Setting Up Pipeline Rating and Discounting*.
FCT_AggreGate	Performs aggregation of data in EDR containers. See "Setting Up Pipeline Aggregation" in *BRM Setting Up Pipeline Rating and Discounting*.
FCT_APN_Map	**Before zoning**: Maps the access point name (APN) to a physical PDP address. **After zoning**: Enhances zone values to support enhanced zoning functionality.
FCT_APN_Map	Reads the discount packets added by DAT_Discount, adds the discounting sub-balance impact to the EDR, and updates the in-memory balance.
FCT_ApplyBalance	Consolidates balance impact data into an associated BRM billing record and one or more balance impact packets. This data is loaded into the BRM database by RE Loader. See "About Consolidation for BRM Billing" in *BRM Setting Up Pipeline Rating and Discounting*.
FCT_CallAssembling	Assembles EDRs that have been split into multiple EDRs. See "Assembling EDRs" in *BRM Setting Up Pipeline Rating and Discounting*.
FCT_CarrierIcRating	Adds roaming data to EDRs for rating by the FCT_PreRating and FCT_MainRating modules.
FCT_CiberOcc	The FCT_CiberOcc module creates a CIBER record for other charges and credits (OCC record), type 50 or 52.

Table 2-1 (Cont.) Pipeline Manager Modules

Module	Description
FCT_CliMapping	Maps multiple numbers to a single number for billing. See "Mapping Multiple Phone Numbers to a Single Number" in *BRM Setting Up Pipeline Rating and Discounting*.
FCT_CreditLimitCheck	Performs credit limit checking to determine whether the event owner has enough resources for the requested service.
FCT_CustomerRating	Supplies the charge for the FCT_MainRating module. See "About Customer Rating" in *BRM Setting Up Pipeline Rating and Discounting*.
FCT_Dayrate	Calculates charges for special day rates, for example, a discount for calls made on January 1.
FCT_Discard	Discards or skips EDRs based on configurable EDR properties. • Skipping an EDR removes it from the pipeline. • Discarding an EDR sends it to a different output stream. In both the cases the state of the EDR becomes invalid. See "Discarding and Skipping EDRs" in *BRM Setting Up Pipeline Rating and Discounting*.
FCT_Discount	Performs discounting functions. See "Configuring Discounting Modules and Components" in *BRM Setting Up Pipeline Rating and Discounting*.
FCT_DiscountAnalysis	Performs discounting analysis functions. See "Configuring Discounting Modules and Components" in *BRM Setting Up Pipeline Rating and Discounting*.
FCT_DroppedCall	Identifies dropped calls and continuation calls.
FCT_DuplicateCheck	Checks for duplicate EDRs. See "Handling Duplicate EDRs" in *BRM Setting Up Pipeline Rating and Discounting*.
FCT_EnhancedSplitting	Specifies different output streams for EDRs based on rules. For example: • You can split EDRs for different service types to different output streams. • You can split EDRs from roaming outcollects and incollects into different streams. See "Using Rules to Send EDRs to Different Output Streams" in *BRM Setting Up Pipeline Rating and Discounting*.
FCT_ExchangeRate	Converts the currency used for rating to the home (system) currency, and the customer's billing currency.
FCT_Filter_Set	Determines whether an EDR is eligible for the system charge offers and system discounts contained in a filter set, and if it is, adds those system charge offers and discounts to a customer's list of purchased charge offers. See "About Using Filter Sets to Apply System Products and Discounts" in *BRM Setting Up Pipeline Rating and Discounting*.
FCT_GlobalRating	Rates all EDRs with a default set of charges. See "About Global Rating" in *BRM Setting Up Pipeline Rating and Discounting*.

Table 2-1 (Cont.) Pipeline Manager Modules

Module	Description
FCT_IRules	Evaluates iRules. Those rules can be used for mapping functions for EDR data fields, splitting EDR containers to different output streams, and so forth.
FCT_IScript	Runs iScripts. The scripts are run in the order specified in the registry.
FCT_Reject	Retrieves an item POID for an item tag from the DAT_ItemAssign module and populates the EDR container with the item POID. See "DAT_ItemAssign".
FCT_MainRating	Performs the main Pipeline Manager rating functionality. See "About Main Rating" in *BRM Setting Up Pipeline Rating and Discounting*.
FCT_MainZoning	Performs zoning for multi-segment zoning.
FCT_NOSP	Maps network source and destination to new values. See "Identifying the Network Operator/Service Provider" in *BRM Setting Up Pipeline Rating and Discounting*.
FCT_NumberPortability	Specifies the new network operator for an existing phone number. See "Setting Up Number Portability" in *BRM Setting Up Pipeline Rating and Discounting*.
FCT_PrefixDesc	Maps phone number prefixes to destination descriptions. See "Creating Call Destination Descriptions" in *BRM Setting Up Pipeline Rating and Discounting*.
FCT_PreRating	Calculates zones and creates impact categories.
FCT_PreRecycle	Used for pipeline-only implementations. Gets the file of rejected EDRs from the reject stream output directory. The module puts the reject EDR file into the pipeline input directory for recycling. It uses the same input folder as the incoming CDR files. See "Recycling EDRs in Pipeline-Only Systems" in *BRM Suspending and Recycling Pipeline EDRs*.
FCT_PreSuspense	When used as part of BRM standard recycling, this module adds suspense-related information to EDRs. When used with Suspense Manager, this module also configures the queryable fields for EDRs suspended in a specific pipeline.
FCT_RateAdjust	Adjusts the charge for an EDR after rating has been performed. See "About Rate Adjustment" in *BRM Setting Up Pipeline Rating and Discounting*.

Table 2-1 (Cont.) Pipeline Manager Modules

Module	Description
FCT_Recycle	Used for pipeline-only implementations. Runs at the end of the pipeline It does either of the following: • When the FCT_PreRecycle module runs in test mode, the FCT_Recycle module creates a report about the processing, but does not send the EDRs to an output file. • When the FCT_PreRecycle module runs in recycle mode, the FCT_Recycle module sends the results to an output file, and attaches a sequence number to the output file. See "Recycling EDRs in Pipeline-Only Systems" in *BRM Suspending and Recycling Pipeline EDRs*.
FCT_Reject	The FCT_Reject module analyzes the errors in an EDR and, if necessary, moves the EDR to a reject file. See "About Standard Recycling" in *BRM Suspending and Recycling Pipeline EDRs*.
FCT_Rounding	Performs rounding for rating and discounting.
FCT_RSC_Map	Performs rate service class (RSC) mapping. See "About Rate-Service Class Mapping" in *BRM Setting Up Pipeline Rating and Discounting*.
FCT_SegZoneNoCust	Finds the segment using the source network information instead of using the customer information.
FCT_ServiceCodeMap	Maps external service codes to internal service codes.
FCT_SocialNo	Flags social numbers for special processing. See "Setting Up Social Numbers" in *BRM Setting Up Pipeline Rating and Discounting*.
FCT_Suspense	When used as part of BRM standard recycling, routes failed EDRs to appropriate output streams depending on their processing status (normal, recycling, or test recycling) and suspense status (succeeded or suspended). When used with Suspense Manager, also adds the suspense reason and subreason codes to EDRs.
FCT_TriggerBill	Sends EDRs to the billing-trigger output stream to trigger immediate billing for the associated accounts. It also sets a billing-trigger error code used to route the EDRs to the suspense output stream, and the **Trigger_Billing** recycle key used to retrieve the suspended EDRs for recycling.
FCT_UoM_Map	Converts the unit of measurement (UoM) of an incoming EDR to a UoM needed for rating a particular service.
FCT_UsageClassMap	The FCT_UsageClassMap module maps external codes for secondary services, such as call forwarding, to internal usage classes.
FCT_Discount	The FCT_USC_Map module performs usage scenario mapping.
FCT_Zone	The FCT_Zone module computes zones when you use Pipeline Manager only for zoning.

Table 2-1 (Cont.) Pipeline Manager Modules

Module	Description
INP_GenericStream	Provides the input interface to the pipeline. See "Configuring EDR Input Processing" in *BRM Setting Up Pipeline Rating and Discounting*.
INP_Realtime	Converts data in an flist to the EDR container format. See "Configuring a Real-Time Discounting Pipeline" in *BRM Setting Up Pipeline Rating and Discounting*.
INP_Recycle	Used by standard recycling and Suspense Manager in the pre-recycling pipeline. It reads suspended usage records from the BRM database, restores original EDRs, applies edits to them, and pushes EDRs into the pre-recycling pipeline.
IRL_EventTypeSplitting	Sends EDRs to separate output streams based on service codes. See "Sending EDRs to Pipeline Output Streams" in *BRM Setting Up Pipeline Rating and Discounting*.
IRL_LeastCostPerEDR	Flags all EDRs that satisfy the criteria for least cost rating. see "About Least Cost Rating" in *BRM Setting Up Pipeline Rating and Discounting*.
IRL_PipelineSplitting	Used in the pre-recycling pipeline to send EDRs to different output streams depending on their original pipeline names. The EDRs are then routed to their original pipelines for recycling.
IRL_LeastCostPerEDR	Flags all EDRs that satisfy the criteria for a promotional savings calculation. See "About Least Cost Rating" in *BRM Setting Up Pipeline Rating and Discounting*.
IRL_UsageType	Assigns usage types to EDRs.
ISC_AddCBD	Prepares EDRs for rerating in the back-out pipeline. **Important:** This is a deprecated module but remains in BRM for backward compatibility.
ISC_BACKOUTTypeSplitting	Used by the backout pipeline for back-out-only rerating. It determines if the EDRs are flagged for back-out-only rerating and sends the EDRs to different output streams based on the event types.
ISC_CiberInputValidation	Performs record-level validations of CIBER records.
ISC_CiberOutputMapping	Adds charge data to the ASSOCIATED_CIBER_EXTENSION block of the EDR. If the EDR does not contain an ASSOCIATED_CIBER_EXTENSION block, this iScript adds one.
ISC_CiberRejectReason	Sets a reason code in the CIBER extension block for records that are rejected.
ISC_EDRToTAPOUTMap	Populates standard values to fields in output TAP file based on its corresponding value in the EDR container.
ISC_GetCamelFlag	Retrieves the CAMEL flag information for a roaming partner. This iScript is used by roaming outcollect processing.

Table 2-1 (Cont.) Pipeline Manager Modules

Module	Description
ISC_LeastCost	Performs one of the following: • Calculates and finds the lowest charge for an EDR. • Calculates the total savings when using an overlay promotion. See "About Least Cost Rating" and "About Calculating the Promotional Savings" in *BRM Setting Up Pipeline Rating and Discounting*.
ISC_MapNetworkOperatorInfo	Maps the DETAIL.SOURCE_NETWORK field to the PIN_FLD_ORIGIN_NETWORK field and the DETAIL.DESTINATION_NETWORK field to the PIN_FLD_DESTINATION_NETWORK field of the opcode input block for the corresponding event. See "Setting Up Number Portability" in *BRM Setting Up Pipeline Rating and Discounting*.
ISC_NRTRDE_ErrorReport	Collects the validation errors in the EDRs and creates error records in the Pipeline Manager database. This iScript is used during roaming incollect processing by the NRTRDE (Near Real-Time Roaming Data Exchange) processing pipeline.
ISC_NRTRDE_EventSplit	Duplicates and routes EDRs to the corresponding roaming partner NRTRDE output streams based on the roaming partner's NRTRDE flag. This iScript is used by roaming outcollect processing.
ISC_NrtrdeHeaderValidation_v2_01	Validates the information in the header record of the TD35 file based on the TD35 specifications. This iScript is used during roaming incollect processing by the NRTRDE processing pipeline.
ISC_ObjectCacheTypeOutputSplitter	Creates two output CDRs from a single input EDR.
ISC_OverrideSuspenseParams	Populates the RATE_TAG field with the value of the NRTRDE flag in the balance impact. This iScript is used by the outcollect settlement pipelines.
ISC_ProfileAnalyzer	Analyzes friends and family extended rating attributes (ERAs) during pipeline rating.
ISC_ProfileLabel	Analyzes ERAs during pipeline rating to determine whether the ERA profiles specified in the **ProfileName** registry entry match the EDR field value.
ISC_PostRating	Adds all the retail and wholesale charges and puts them in DETAIL.RETAIL_CHARGED_AMOUNT_VALUE and DETAIL.WHOLESALE_CHARGED_AMOUNT_VALUE fields. See "Billing Consolidation with CIBER Roaming and Revenue Assurance" in *BRM Setting Up Pipeline Rating and Discounting*.
ISC_SetAndValidateBatchInfo	Populates and validates the batch related fields for the EDR container.
ISC_SetEDRStatus	Sets the EDR status to **Success**, **Suspense**, **Duplicate**, **Discard**, or **Skipped** for each EDR.
ISC_SetOutputStream	Sets the Output Stream to **TelOut**, **SMSOut**, **GPRSOut**, **RejectOut**, or **DuplicateOut** for each EDR.

Table 2-1 (Cont.) Pipeline Manager Modules

Module	Description
ISC_SetRevenueFigures	Collects the previous and current charged and discount amount for a configured Balance Element ID.
ISC_SetRevenueStream	Sets the Revenue Stream to **Retail**, **Wholesale**, **Roaming**, or **Unknown** for each EDR.
ISC_SetSvcCodeRTZoning	Finds the service type and updates the **DETAIL.INTERN_SERVICE_CODE** EDR field with the customized service code value for each EDR.
ISC_TapDetailValidation_v3_12	Validates that the fields present in the detail record of the EDR container contain valid data.
ISC_TapSplitting	Splits mobile originating and terminating EDRs when the CDR contains more than one basic service. ISC_TapSplitting creates a new EDR for each additional basic service. See "Generating Multiple TAP MOC and MTC Records" in *BRM Setting Up Pipeline Rating and Discounting*.
ISC_TaxCalc	Applies a flat tax to pipeline-rated events.
LOG	Logs error messages.
Memory Monitor	Monitors Pipeline Manager system memory during startup and while it is processing files.
NET_EM	The NET_EM module hosts a BRM External Module (EM). This allows the NET_EM module to use the BRM API opcodes to transfer data between real-time rating and Pipeline Manager. See "Configuring a Real-Time Discounting Pipeline" in *BRM Setting Up Pipeline Rating and Discounting*.
OUT_DB	Sends output to the database. See "Sending Output to a Database" in *BRM Setting Up Pipeline Rating and Discounting*.
OUT_DevNull	Removes EDRs that are not needed by Pipeline Manager. See "Configuring Output for Rejected or Duplicate EDRs" in *BRM Setting Up Pipeline Rating and Discounting*. All registry entries and error messages are handled by the Output Collection module. See "Output Collection". For more information, see "Discarding and Skipping EDRs" in *BRM Setting Up Pipeline Rating and Discounting*.
OUT_GenericStream	Handles the output stream for rated EDRs. See "Configuring EDR Output Processing" in *BRM Setting Up Pipeline Rating and Discounting*. When you configure the OUT_GenericStream module, you configure the EXT_OutFileManager module to specify file management options. See "EXT_OutFileManager".
OUT_Realtime	The OUT_Realtime module converts data in the pipeline EDR output to flist format. See "Configuring a Real-Time Discounting Pipeline" in *BRM Setting Up Pipeline Rating and Discounting*.

Table 2-1 (Cont.) Pipeline Manager Modules

Module	Description
OUT_Reject	Writes rejected EDRs to an output stream. The written record is exactly the same as the original input record. See "Configuring Output for Rejected or Duplicate EDRs" in *BRM Setting Up Pipeline Rating and Discounting*. All registry entries and error messages are handled by the Output Collection module. See "Output Controller". For more information, see the following documents: • "About Standard Recycling" in *BRM Suspending and Recycling Pipeline EDRs* • "Handling Duplicate EDRs" in *BRM Setting Up Pipeline Rating and Discounting*.
Sequencer	Checks for duplicate CDR input files and adds tracking numbers to output streams.
Input Controller	Manages incoming input streams for its associated pipeline. See "Configuring EDR Input Processing" in *BRM Setting Up Pipeline Rating and Discounting*.
Output Controller	Manages the output streams for its associated pipeline. See "Configuring EDR Output Processing" in *BRM Setting Up Pipeline Rating and Discounting*.
Output Collection	Handles output streams. See "Configuring EDR Output Processing" in *BRM Setting Up Pipeline Rating and Discounting*.
Pipeline Controller	Manages all processes in its associated pipeline. See "Pipeline Controller".
Transaction Manager	Coordinates the state of all transactional modules and components in a pipeline.
Transaction ID Controller	Generates transaction IDs for all pipelines.
Transaction ID Database Generator	Stores transaction IDs in a Pipeline Manager database table.
Transaction ID File Generator	Stores transaction IDs in a file.

3
Pipeline Manager Function Modules

Learn about the Oracle Communications Billing and Revenue Management (BRM) Pipeline Manager function modules.

Topics in this document:

- FCT_MainRating
- FCT_MainZoning
- FCT_NOSP
- FCT_NumberPortability
- FCT_Opcode
- FCT_PrefixDesc
- FCT_PreRating
- FCT_PreRecycle
- FCT_PreSuspense
- FCT_RateAdjust
- FCT_Recycle
- FCT_Reject
- FCT_Rounding
- FCT_RSC_Map
- FCT_SegZoneNoCust
- FCT_ServiceCodeMap
- FCT_SocialNo
- FCT_Suspense
- FCT_Timer
- FCT_TriggerBill
- FCT_UoM_Map
- FCT_UsageClassMap
- FCT_USC_Map
- FCT_Zone

FCT_Account

The FCT_Account module adds customer data to an EDR.

It also does the following:

- Flags incoming CDRs to be suspended when the account is being rerated by **pin_rerate**.
- For exclusion rules for usage discounts, retrieves package list information from the DAT_AccountBatch module and adds this information to the PLAN_LIST block in the CustomerData block in the EDR. The package list includes all plans for the subscription service, including plans owned by any member services in the subscription group.
- For Balance Monitoring, retrieves the balance monitor information for an event owner from the DAT_AccountBatch module and fills the CustomerData block in the EDR with the monitor list.

Dependencies

Requires a connection to the DAT_AccountBatch module.

This module must run before the zoning and rating modules.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-1 lists the FCT_Account registry entries.

Table 3-1 FCT_Account Registry Entries

Entry	Description	Mandatory
Active	Specifies whether the module is active or inactive. **True** = Active **False** = Inactive You can use this entry in a semaphore file.	Yes
DataModule	Specifies the connection to the DAT_AccountBatch module.	Yes
DisableRatingProductCheck	Specifies whether the module rejects any CDRs with no rating charge offers. **True** = FCT_Account does not reject CDRs, if the configured event type for batch rating is not found in any of the charge offers owned by the service or account. **False** = FCT_Account rejects CDRs if the configured event type for batch rating is not found in any of the charge offers owned by the service or account.	No

Sample Registry

```
Account
{
  ModuleName = FCT_Account
  Module
  {
    Active         = True
    DataModule     = ifw.DataPool.CustomerData
    Offset         = 5
  }
}
```

Semaphore File Entries

Table 3-2 lists the FCT_Account Semaphore file entry.

Table 3-2 FCT_Account Semaphore File Entries

Entry	Description
Active	Specifies whether the module is active or inactive. **True** = Active **False** = Inactive

Sample Semaphore File Entry

```
ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.CustomerSearch.Module.Active = False
```

EDR Container Fields

The FCT_Account module uses the EDR container fields listed in Table 3-3.

Table 3-3 FCT_Account EDR Container Fields

Alias field name Default field name	Type	Access	Description
INTERN_SERVICE_CODE DETAIL.INTERN_SERVICE_CODE	String	Read	Contains the internal service code that determines the EDR container fields that are used for the customer lookup. For example, a telephone service uses the A number to find the customer account.
BDR_CHARGING_START_TIMESTAMP DETAIL.CHARGING_START_TIMESTAMP	Date	Read	Contains the start timestamp of the event. The time zone information for this timestamp is stored in the field BDR_UTC_TIME_OFFSET.
BDR_UTC_TIME_OFFSET DETAIL.UTC_TIME_OFFSET	String	Read	Contains the UTC time offset that normalizes the charging start timestamp to the UTC time zone. All validity timestamps in the BRM customer data are stored in normalized UTC time. The format is +/- *HHMM*.
DETAIL.CUST_A.ACTG_LAST_DATE	Date	Read	Contains the date that the current monthly cycle began.
DETAIL.CUST_A.ACTG_NEXT_DATE	Date	Read	Contains the date that the current monthly cycle ends.
DETAIL.CUST_A.ACTG_USED_DATE	Date	Write	Contains the date used for this EDR.
RESOURCE_LIST DETAIL.CUST_A.RESOURCE_LIST	String	Write	Contains a list of the balance elements included in the A-number customer's balance.
RESOURCE_LIST DETAIL.CUST_B.RESOURCE_LIST	String	Write	Contains a list of the balance elements included in the B-number customer's balance.
DETAIL.CUST_A.INTERN_RATING_PRODUCTS	String	Create	Contains the charge offer rating indexes. This is a comma-separated list of all rating charge offers' indexes associated with the same service and event, and their priorities.
DETAIL.CUST_A.PRODUCT_PRIORITY	Integer	Read	Contains the priority for a charge offer.
DETAIL.CUST_A.PRODUCT.USAGE_START	Date	Read	Contains the start time for a charge offer.

Table 3-3 (Cont.) FCT_Account EDR Container Fields

Alias field name Default field name	Type	Access	Description
DETAIL.CUST_A.PRODUCT.FIRST_USAGE_INDICATOR	Integer	Write	Specifies whether the account's charge offer is configured to start when first used and the state of the validity period.
DETAIL.CUST_A.DL.PD.FIRST_USAGE_INDICATOR	Integer	Write	Specifies whether the account's discount is configured to start when first used and the state of the validity period.
BALANCE_GROUP_ID DETAIL.CUST_A.ML.BALANCE_GROUP_ID	String	Create	Contains the balance monitor group ID.
MONITOR_OWNER_ACCT_ID DETAIL.CUST_A.ML.MONITOR_OWNER_ACCT_ID	String	Create	Contains the monitor owner's account ID.
MONITOR_OWNER_ID DETAIL.CUST_A.ML.MONITOR_OWNER_ID	String	Create	Contains the monitor owner ID.
MONITOR_OWNER_TYPE DETAIL.CUST_A.ML.MONITOR_OWNER_TYPE	String	Create	Contains the monitor owner type.
DETAIL.CUST_A.SHARED_PROFILE_LIST. ERA.LABEL	String	Write	Contains the label associated with a shared service profile. For example, MYFAMILY.
DETAIL.CUST_A.PRODUCT.ERA. LABEL	String	Write	Contains the label associated with an owned service profile. For example. MYFAMILY.
DETAIL.CUST_A. SHARED_PROFILE_LIST	Block	Write	Contains all the shared profiles, which the service shares as a member of one or more profile sharing groups.
DETAIL.CUST_A. SHARED_PROFILE_LIST.ERA	Block	Write	Contains shared ERA information.

Database Interface for the FCT_Account Module

The FCT_Account modules uses the following database tables:

- The FCT_Account and FCT_AccountRouter modules use the data in the IFW_ALIAS_MAP table to link an internal service code the EDR container field used for identifying an account.
- The FCT_Account and FCT_AccountRouter module use the data in the IFW_EDRC_DESC table to look up the alias mapping data. This table contains all valid EDR container fields for different format descriptions.
- The FCT_Account module uses the data in the IFW_EDRC_FIELD table to look up the alias mapping data. This table contains all valid EDR container fields for different format descriptions.
- The FCT_Account and FCT_AccountRouter use the data in the IFW_PIPELINE table to look up the alias mapping data. The IFW_PIPELINE database table defines the EDR formats that can be used for each pipeline.

FCT_AccountLPRouter

The FCT_AccountLPRouter module looks up the logical partition in which the account resides and writes it to the LOGICAL_PARTITION_ID EDR field.

Dependencies

Requires a connection to the DAT_AccountBatch module.

Registry Entries

Table 3-4 lists the FCT_AccountPRouter registry entries.

Table 3-4 FCT_AccountLPRouter Registry Entries

Entry	Description	Mandatory
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive You can use this entry in a semaphore file.	Yes
DataModule	Specifies the connection to the DAT_AccountBatch module.	Yes

Sample Registry

```
#------------------------------------------------
# AccountLPRouter
#------------------------------------------------
AccountLPRouterModule
{
  ModuleName = FCT_AccountLPRouter
  Module
  {
    Active       = True
    DataModule   = ifw.DataPool.CustomerData
  } # end Module
} # end LPRouter
#------------------------------------------------
```

Semaphore File Entries

Table 3-5 lists the FCT_AccountPRouter Semaphore file entry.

Table 3-5 FCT_AccountPRouter Semaphore File Entry

Entry	Description
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive

Sample Semaphore File Entry

```
ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.AccountLPRouter.Module.Active = False
```

EDR Container Fields

The FCT_AccountLPRouter module uses the following EDR container fields:

Events

Table 3-6 lists the events for FCT_AccountPRouter.

Table 3-6 FCT_AccountPRouter Events

Alias Field Name Default Field Name	Type	Access	Description
LOGICAL_PARTITION_ID	String	Read	Contains the logical partition ID for the event.

FCT_AccountRouter

The FCT_AccountRouter module routes EDRs to the appropriate database schema in multischema systems. This module finds the customer's schema and routes the EDRs to the appropriate pipeline. See "Using Pipeline Manager with Multiple Database Schemas" in *BRM Setting Up Pipeline Rating and Discounting*.

When used with standard recycling or Suspense Manager, this module routes call records from the pre-recycling pipeline to the appropriate rating pipeline.

> **Note:**
>
> FCT_AccountRouter runs in its own instance of Pipeline Manager and should be configured with its own registry file. Create a registry file that includes entries for the Input, FCT_AccountRouter, and Output modules.

Dependencies

Requires a connection to the DAT_AccountBatch module.

For general use, this module must run after the FCT_ServiceCodeMap module and before the rating modules.

For use with standard recycling or Suspense Manager using multiple database schemas, this module must be run before OUT_GenericStream in a pre-recycling pipeline.

This module sends output to a separate pipeline for each BRM database schema.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-7 lists the FCT_AccountRouter registry entries.

Table 3-7 FCT_AccountRouter Registry Entries

Entry	Description	Mandatory
Active	Specifies whether the module is active or inactive. **True** = Active **False** = Inactive You can use this entry in a semaphore file.	Yes
DataModule	Specifies the connection to the DAT_AccountBatch module.	Yes
Mode	Specifies how this module routes EDRs to the appropriate pipeline. **ROUTER** = Routes EDRs based on the database schema ID. **RECYCLE** = Routes EDRs based on the pipeline name and schema ID. See "Configuring a Pre-Recycling Pipeline" in *BRM Suspending and Recycling Pipeline EDRs*.	Yes
Streams	Lists the target output streams. The syntax for this section depends on whether the module is operating in **ROUTER** mode or **RECYCLE** mode. See "Configuring EDR Output Processing" in *BRM Setting Up Pipeline Rating and Discounting*.	Yes

Sample Registries

Sample registry entries for the **ROUTER** mode:

```
AccountRouter
{
   ModuleName = FCT_AccountRouter
   Module
   {
      Active = True
      DataModule = ifw.DataPool.Account
      Mode=ROUTER
      Streams
      {
         1 = DB0001Stream
         2 = DB0002Stream
         3 = DB0003Stream
      }
   }
}
```

Sample registry entries for the **RECYCLE** mode:

```
AccountRouter
{
    ModuleName = FCT_AccountRouter
    Module
    {
        Active = True
        DataModule = ifw.DataPool.CustomerData
        Mode = RECYCLE

        Streams
```

```
            {
            # This section maps pipelines to their original stream
                1_Pipeline_A = StreamA_for_DB1
                1_Pipeline_B = StreamB_for_DB1
                2_Pipeline_C = StreamC_for_DB2
                2_Pipeline_D = StreamD_for_DB2

            # This section maps pipelines to their alternate stream
                1_Pipeline_C = StreamA_for_DB1
                1_Pipeline_D = StreamB_for_DB1
                2_Pipeline_A = StreamC_for_DB2
                2_Pipeline_B = StreamD_for_DB2

            # This section sends EDRs to the suspense stream
                0_* = SuspenseStream
            }
        }
}
```

Semaphore File Entries

Table 3-8 list the FCT_AccountRouter Semaphore file entry.

Table 3-8 FCT_AccountRouter Semaphore File Entry

Entry	Description
Active	Specifies whether the module is active or inactive. **True** = Active **False** = Inactive

Sample Semaphore File Entry

```
ifw.Pipelines.Rating.Module.AccountRouter.Module.Active = False
```

EDR Container Fields

The FCT_AccountRouter module uses the EDR container fields listed in Table 3-9.

Table 3-9 FCT_AccountRouter EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
INTERN_SERVICE_CODE DETAIL.INTERN_SERVICE_CODE	String	Read	Contains the internal service code that determines the EDR container fields used for the customer lookup.
BDR_CHARGING_START_TIMESTAMP DETAIL.CHARGING_START_TIMESTAMP	Date	Read	Contains the start timestamp of the event. The time zone information for this timestamp is stored in the BDR_UTC_TIME_OFFSET field.
BDR_UTC_TIME_OFFSET DETAIL.UTC_TIME_OFFSET	String	Read	Contains the UTC time offset that normalizes the charging start timestamp to the UTC time zone. All validity timestamps in the BRM customer data are stored in normalized UTC time. The format is +/- *HHMM*.

Database Tables

The FCT_Account module uses the following database tables:

- IFW_ALIAS_MAP. The FCT_Account and FCT_AccountRouter modules use the data in the IFW_ALIAS_MAP database table to link an internal service code the EDR container field used for identifying an account.
- IFW_EDRC_DESC. The FCT_Account and FCT_AccountRouter module use the data in the IFW_EDRC_DESC table to look up the alias mapping data. This table contains all valid EDR container fields for different format descriptions. See "Using Pipeline Manager with Multiple Database Schemas" in *BRM Setting Up Pipeline Rating and Discounting*.
- IFW_PIPELINE. The FCT_Account and FCT_AccountRouter use the data in this table to look up the alias mapping data. The IFW_PIPELINE database table defines the EDR formats that can be used for each pipeline. See "Using Pipeline Manager with Multiple Database Schemas" in *BRM Setting Up Pipeline Rating and Discounting*.

> **Note:**
>
> For information on compare patterns used in database values, see "About Using Regular Expressions when Specifying the Data to Extract" in *BRM Setting Up Pipeline Rating and Discounting*.

FCT_AggreGate

The FCT_AggreGate module performs aggregation of data in EDR containers. See "Setting Up Pipeline Aggregation" in *BRM Setting Up Pipeline Rating and Discounting*.

Dependencies

The FCT_AggreGate function module requires a connection to the DAT_ScenarioReader module.

This module runs after rating modules and can run in its own pipeline.

When you configure the FCT_CallAssembling function module to not drop EDRs from the pipeline, ensure that the FCT_AggreGate function module that counts them runs before the FCT_Reject function module. See "FCT_CallAssembling".

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Rating and Discounting*.

Registry Entries

Table 3-10 lists the FCT_AggreGate registry entries.

Table 3-10 FCT_AggreGate Registry Entries

Entry	Description	Mandatory
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive You can use this entry in a semaphore file.	Yes
BackOut	Specifies if the module is working in back-out mode, which is used for rerating. Possible values are **True** and **False**. Default = **False**	No
ControlFile	Specifies the definitions for the control file. The control files are used by the Database Loader utility to load the results into the database.	No
ControlFile.DataFilePath	Specifies whether the path to the data file is included in the control file. Possible values are **True** and **False**. Default = **True**	No
ControlFile.Suffix	Specifies the file name suffix for the control file. You can specify any suffix. Default = **.ctl**	No
IncludeCtlFile	Specifies whether to create control files. If **True**, control and data files are created. If **False**, only data files are created. Default = **True**	No
IncludeErrorEDRs	Specifies whether EDRs that include errors are included in the aggregation scenario. Possible values are **True** and **False**. Default = **False**	No
IncludeInvalidDetailEDRs	Specifies whether EDRs that are invalid are included in the aggregation scenario. Possible values are **True** and **False**. Default = **False**	No
ResultFile	Sub-entries define characteristics of the result file.	No
ResultFile.DoneSuffix	Specifies the file name suffix for processed files. You can specify any suffix. Default = **.dat**	No
ResultFile.TempSuffix	Specifies the file name suffix for temporary files created during processing. You can specify any suffix. Default = **.tmp**	No
ResultFile.WriteEmptyFile	Indicates whether to create an empty processed file if there are no processing results. Possible values are **True** and **False**. Default = **True**	No
ScenarioReaderDataModule	Specifies a connection to the DAT_ScenarioReader module.	Yes
Scenarios	Specifies the scenario that is processed and configured. See "Specifying Scenario Attributes" in *BRM Setting Up Pipeline Rating and Discounting*. If nothing is entered, no processing occurs.	No
Scenarios	No value. Subentries define the scenarios to be processed and how they are configured. See "Specifying Scenario Attributes" in *BRM Setting Up Pipeline Rating and Discounting*. If nothing is entered, no processing occurs.	No

Table 3-10 (Cont.) FCT_AggreGate Registry Entries

Entry	Description	Mandatory
Scenarios.*name*	Specifies the name of a scenario to be configured. The scenarios included with Revenue Assurance Manager have names such as **RA_01**, **RA_02** and so on.	Yes
Scenarios.*name***.ControlPointId**	Defines the Revenue Assurance control point that uses this scenario. Control point names must be unique system-wide. If a value is specified, the control point ID is included in the result file.	No
Scenarios.*name***.CtlDir**	Specifies the directory from which to read the control file. Use this entry when you want to override the default directory defined for the scenario.	No
Scenarios.*name***.DoneDir**	Specifies a path and name for processed files. Use this entry when you want to override the default directory.	No
Scenarios.*name***.FieldDelimiter**	Specifies the delimiter of result fields. The default is a semicolon (**;**), which is the value defined in the IFW_SCENARIO table.	Yes
Scenarios.*name***.IncludeErrorEDRs**	Speechifies whether EDRs that have errors are included in the aggregation processing. Possible values are **True** and **False**. Default value is **False**. **Note:** For Revenue Assurance, this parameter must be present and set to **True**.	No
Scenarios.*name***.IncludeInvalidDetailEDRs**	Specifies whether invalid EDRs are included in aggregation processing. Possible values are **True** and **False**. Default value is **False**. **Note:** For Revenue Assurance, this parameter must be present and set to **True**.	No
Scenarios.*name***.IncludeProcessingTimestamps**	Specifies whether transaction time range data is included in result files. Possible values are **True** and **False**. Default value is **False**.	No
Scenarios.*name***.TableName**	Specifies the table used when DB Loader stores aggregation results in the database. The table name is also used for files created by this scenario. Use this entry when you want to override the default table name defined for the scenario.	No
Scenarios.*name***.TempDir**	Specifies a path and name for a directory to use for temporary files created during processing. Use this entry when you want to override the default directory defined for the scenario.	No
Scenarios.*name***.Threshold**	Specifies the maximum number of aggregations stored in memory before writing to disk. In most cases, there aren't enough aggregations to make the threshold meaningful. A typical value is a relatively large number, such as **99999**. Use this entry when you want to override the default directory defined for the scenario.	No

Sample Registry

```
Aggregate
{
```

```
  ModuleName = FCT_AggreGate
  Module
  {
    Active = TRUE

    ScenarioReaderDataModule = ifw.DataPool.ScenarioReader

    Scenarios
      {
      PURCHASE
      {
        TableName = PURCHASE_RESULT
        Threshold = 100000
        TempDir = result/temp
        DoneDir = result/done
        CtlDir = result/ctl
        FieldDelimiter = |
      }
      STAT
      {
        TempDir = result/temp
        DoneDir = result/done
        CtlDir = result/ctl
        }
        DAY
        {
      }
   }

    ResultFile
    {
      TempSuffix = .tmp
      DoneSuffix = .dat
      WriteEmptyFile = FALSE
    }

    ControlFile
      {
      IncludeCtlFile = FALSE
      Suffix = .ctl
      DataFilePath = TRUE
    }
  }
}
```

Semaphore File Entries

Table 3-11 lists the FCT_AggreGate Semaphore file entry.

Table 3-11 FCT_AggreGate Semaphore File Entry

Entry	Description
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive

EDR Container Fields

The module reads EDR container fields defined by the DAT_ScenarioReader module. See "DAT_ScenarioReader".

Events

Table 3-12 lists the FCT_AggreGate events.

Table 3-12 FCT_AggreGate Events

Event name	Trigger	Sender	Parameter
EVT_CTL_FILE_CREATED	Control file for the DB Loader utility was created.	DAT_ScenarioReader	File name

FCT_APN_Map

Before zoning: The FCT_APN_Map module maps the access point name (APN) to a physical PDP address.

After zoning: The FCT_APN_Map module enhances zone values to support enhanced zoning functionality.

Dependencies

Requires a connection to the Pipeline Manager database.

This module can be run before or after the zoning modules (FCT_Zone and FCT_PreRating).

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-13 lists the FCT_APN_Map registry entries.

Table 3-13 FCT_APN_Map Registry Entries

Entry	Description	Mandatory
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive You can use this entry in a semaphore file.	Yes
APNGroup	Specifies the Access Point Name (APN) group value for the mapping. If you enter a group name, run the module before zoning. Otherwise, run it after zoning. You can use this entry in a semaphore file.	Yes
DataConnection	Specifies the connection to the Pipeline Manager database.	Yes

Sample Registry

```
APN_Map
{
  ModuleName = FCT_APN_Map
  Module
  {
    Active = True
    APNGroup = apn_group
    DataConnection = integrate.DataPool.Login
  }
}
```

Semaphore File Entries

Table 3-14 lists the FCT_APN_Map Semaphore file entries.

Table 3-14 FCT_APN_Map Semaphore File Entries

Entry	Description
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive
APNGroup	If you enter a group name, run the module before zoning. Otherwise, run it after zoning.
Reload	Reloads data from the database into memory.

Sample Semaphore File Entries

```
ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.APNMap.Module.Active = True
ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.APNMap.Module.APNGroup = apn_group
ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.APNMap.Module.Reload{}
```

EDR Container Fields

The FCT_APN_Map module uses the EDR container fields listed in Table 3-15.

Table 3-15 FCT_APN_Map EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
INTERN_SERVICE_CODE DETAIL.INTERN_SERVICE_CODE	String	Read	Contains the internal service code.
APN_ADDRESS DETAIL.ASS_GPRS_EXT.APN_ADDRESS	String	Read	Contains the access point name address.
ACTION_CODE DETAIL.ASS_GSMW_EXT.SS_PACKET.ACTION_CODE	String	Read	Contains the action code in supplementary service packet.

Table 3-15 (Cont.) FCT_APN_Map EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
SS_EVENT DETAIL.ASS_GSMW_EXT.SS_PACKET.SS_EVENT	String	Read	Contains the supplementary service event.
INTERN_B_NUMBER_ZONE DETAIL.INTERN_B_NUMBER_ZONE	String	Write	Contains the destination zone.
BDR_INTERN_APN_GROUP DETAIL.INTERN_APN_GROUP	String	Read	Contains the APN group.
WHOLESALE_IMPACT_CATEGORY DETAIL.WHOLESALE_IMPACT_CATEGORY	String	Read/Write	Contains the wholesale impact category.
RETAIL.IMPACT_CATEGORY DETAIL.RETAIL.IMPACT_CATEGORY	String	Read/Write	Contains the retail impact category.
ASS_CBD_IMPACT_CATEGORY DETAIL.ASS_CBD.CP.IMPACT_CATEGORY	String	Write	Contains the impact category.
ASS_CBD_INTERN_APN_GROUP DETAIL.ASS_CBD.CP.INTERN_APN_GROUP	String	Read	Contains the APN group in the charge packet.
RATEPLAN_TYPE DETAIL.ASS_CBD.CP.RATEPLAN_TYPE	String	Read	Contains the wholesale or retail charge type. The default is retail.
ASS_ZBD_INTERN_APN_GROUP DETAIL.ASS_ZBD.ZP.INTERN_APN_GROUP	String	Read	Contains the APN group in zone breakdown records.
ASS_ZBD_ZONE_RESULT_VALUE_WS DETAIL.ASS_ZBD.ZP.ZONE_RESULT_VALUE_WS	String	Write	Contains the wholesale zone result.
ASS_ZBD_ZONE_RESULT_VALUE_RT DETAIL.ASS_ZBD.ZP.ZONE_RESULT_VALUE_RT	String	Write	Contains the retail zone result.

Database Tables

The FCT_APN_Map module uses the following database tables:

- IFW_APN_MAP. This table stores the APN mapping rules.
- IFW_APN_GROUP. The IFW_APN_GROUP table stores the APN groups used for APN mapping.

> **Note:**
>
> For information on compare patterns used in database values, see "About Using Regular Expressions when Specifying the Data to Extract" in *BRM Setting Up Pipeline Rating and Discounting*.

FCT_ApplyBalance

The FCT_ApplyBalance module reads the discount packets added by DAT_Discount, adds the discounting sub-balance impact to the EDR, and updates the in-memory balance.

When the discount impacts a noncurrency balance element balance that starts on first usage, this module adds the validity period information to the EDR. See "About Setting the Validity of Balance Elements Impacted by Discounts" in *BRM Setting Up Pipeline Rating and Discounting*.

This module is mandatory when you configure batch discounting in Pipeline Manager. Add this module to the pipeline after the FCT_Rounding module.

Dependencies

Requires a connection to the DAT_BalanceBatch module.

This module should be in the same function pool as the FCT_Discount module for performance reasons and must run after that module.

This module also must run after FCT_Rounding.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-16 lists the FCT_ApplyBalance registry entries.

Table 3-16 FCT_ApplyBalance Registry Entries

Entry	Description	Mandatory
Active	Specifies if the module is active or inactive: **True** = Active **False** = Inactive	Yes
BalanceDataModule	Specifies the connection to the DAT_BalanceBatch module.	Yes
DiscountDataModule	Specifies the connection to the DAT_Discount module.	Yes
FirstUsageCreateStream	Specifies the output stream for balance element balance impacts that whose validity periods start on first usage. See "About Setting the Validity of Balance Elements Impacted by Discounts" in *BRM Setting Up Pipeline Rating and Discounting*. Default = **FirstUsageResourceOutput**	No
IgnoreEDROnDeadlock	Specifies what the module should do when it encounters a deadlock. **True** = Ignore the EDRs and continue processing the EDR file. **False** = Roll back already processed EDRs and start reprocessing the same file.	No
NumberOfNotificationLimit	Specifies the maximum number of notification events that can be written into the output XML file. Once the maximum number of notification events is reached, the module creates another XML file.	No
NotificationOutputDirectory	Specifies the directory in which to write the output XML files.	No
NotificationOutputPrefix	Specifies the prefix of the output XML file.	No

Table 3-16 (Cont.) FCT_ApplyBalance Registry Entries

Entry	Description	Mandatory
PortalConfigDataModule	Specifies the connection to the DAT_PortalConfig module.	Yes

Sample Registry

```
ApplyBalance
{
   ModuleName = FCT_ApplyBalance
   Module
   {
      Active                  = TRUE
      DiscountDataModule      = ifw.DataPool.DiscountModelDataModule
      BalanceDataModule       = ifw.DataPool.BalanceDataModule
      PortalConfigDataModule  = ifw.DataPool.PortalConfigDataModule
      IgnoreEDROnDeadlock     = False

     NumberOfNotificationLimit = ifw.DataPool.PortalConfigDataModule
     NotificationOutputDirectory          = ./data/out/notifybalancebreach
     NotificationOutputPrefix             = balancebreach_
   }
}
```

Semaphore File Entries

Table 3-17 lists the FCT_ApplyBalance Semaphore file entries.

Table 3-17 FCT_ApplyBalance Semaphore File Entries

Entry	Description
ReloadCreditThresholdParam	Reloads the value from the **CreditThresholdChecking** business parameter.

Sample Semaphore File Entry

```
ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.ApplyBalanceModule.Module.Reload
CreditThresholdParam{}
```

EDR Container Fields

Table 3-18 lists the FCT_ApplyBalance EDR container fields.

Table 3-18 FCT_ApplyBalance EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
DETAIL.EVENT_TYPE	String	Read	Specifies the BRM event type.
INTERNAL.TRANSACTION_ID	String	Read	Specifies the transaction ID. This is used for queuing.

Table 3-18 (Cont.) FCT_ApplyBalance EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
DETAIL.INTERN_PROCESS_STATUS	Integer	Read	Specifies the process status. If set to 2, a recycle test is in progress, and this container is skipped.
DETAIL.ASS_CBD.DP.OBJECT_ACCOUNT	Integer	Read	Specifies the POID of the discount offer owner.
DETAIL.ASS_CBD.DP.OFFERING_POID	String	Read	Specifies the POID of the account's purchased discount offer.
DETAIL.ASS_CBD.DP.RESOURCE_ID	Integer	Read	Specifies the ID of the balance element impacted by the discount.
DETAIL.ASS_CBD.DP.GRANTED_AMOUNT	Decimal	Read	Specifies the discount grant amount. This can be currency or noncurrency.
DETAIL.ASS_CBD.DP.BALANCE_GROUP_ID	String	Read	Specifies the POID of the account's balance group that is impacted by the discount.
DETAIL.ASS_CBD.DP.VALID_FROM	Date	Read	Specifies the date when the discount becomes valid.
DETAIL.ASS_CBD.DP.VALID_FROM_DETAIL	Integer	Read	Specifies the mode of the discounts' start time (such as first-usage or relative), the relative offset unit (such as minutes, months, or cycles) and the number of offset units.
DETAIL.ASS_CBD.DP.VALID_TO	Date	Read	Specifies the date when the discount is no longer valid.
DETAIL.ASS_CBD.DP.VALID_TO_DETAIL	Integer	Read	Specifies the mode of the discounts' end time (such as relative), the relative offset unit (such as minutes, months, or cycles) and the number of offset units.
DETAIL.ASS_CBD.DP.SUB_BALANCE	Packet	Write	Specifies the sub-balance that is impacted by the discount.
DETAIL.ASS_CBD.DP.SUB_BALANCE.REC_ID	Integer	Write	Specifies the ID of the sub-balance impacted by this discount.
DETAIL.ASS_CBD.DP.SUB_BALANCE.AMOUNT	Decimal	Write	Specifies the amount of the sub-balance impacted by this discount packet.
DETAIL.ASS_CBD.DP.SUB_BALANCE.GRANTOR	String	Write	Specifies the charge offer or discount that granted this balance element.
DETAIL.ASS_CBD.DP.SUB_BALANCE.VALID_FROM	Date	Write	Specifies the date when this sub-balance becomes valid.
DETAIL.ASS_CBD.DP.SUB_BALANCE.VALID_TO	Date	Write	Specifies the date when this sub-balance is no longer valid.
DETAIL.ASS_CBD.DP.SUB_BALANCE.VALID_FROM_DETAILS	Integer	Write	Specifies the mode of the sub-balance validity period and the relative offset start time details.

Table 3-18 (Cont.) FCT_ApplyBalance EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
DETAIL.ASS_CBD.DP.SUB_BALANCE.VALID_TO_DETAILS	Integer	Write	Specifies the mode of the sub-balance validity period and the relative offset end time details.
DETAIL.ASS_CBD.DP.SUB_BALANCE.CONTRIBUTOR	String	Write	Specifies the service or account that contributes to the sub-balance amount.
DETAIL.ASS_CBD.UBP	Packet	Write	The update balance packet. This packet contains validity period information for all the account's sub-balances that start on first usage. This packet is added when a sub-balance with a first-usage start time is consumed for the first time.
DETAIL.ASS_CBD.UBP.BALANCE_GROUP_ID	Integer	Write	Specifies the POID of the account's balance group associated with a balance element balance that starts on first usage.
DETAIL.ASS_CBD.UBP.RESOURCE_ID	Integer	Write	Specifies the ID of the associated balance element.
DETAIL.ASS_CBD.UBP.RECORD_NUMBER	Integer	Write	Specifies the sequence number of the record in the file.
DETAIL.ASS_CBD.UBP.VALID_FROM	Date	Write	Specifies the start time of the balance element balance.
DETAIL.ASS_CBD.UBP.VALID_TO	Date	Write	Specifies the end time of the balance element balance.
DETAIL.ASS_CBD.UBP.VALID_FROM_DETAIL	Integer	Write	Specifies the balance element balance start time details: the mode of the validity period (such as first-usage or relative), the relative offset unit (such as minutes, months, or cycles) and the number of offset units.
DETAIL.ASS_CBD.UBP.VALID_TO_DETAIL	Integer	Write	Specifies the balance element balance end time details: the mode of the validity period (such as relative), the relative offset unit (such as minutes, months, or cycles) and the number of offset units.

FCT_BatchSuspense

This module is used by the Suspense Manager service to handle file-level suspense operations. It generates the suspended batch create and update streams to be loaded into the BRM database for suspense batch files.

> **Note:**
>
> This module must be placed before all the validation modules/iScripts in a pipeline.

This module also adds suspense reason and suspense subreason codes to batches. See:

- "Setting Up Suspended Batch (SB) Loader for Suspense Manager" in *BRM Suspending and Recycling Pipeline EDRs*.
- "About the FCT_BatchSuspense Module" in *BRM Setting Up Pipeline Rating and Discounting*.
- "About Suspense Manager" in *BRM Setting Up Pipeline Rating and Discounting*.

Dependencies

Requires a connection to the BRM database.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-19 lists the FCT_BatchSuspense registry entries.

Table 3-19 FCT_BatchSuspense Registry Entries

Entry	Description	Mandatory
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive You can use this entry in a semaphore file.	Yes
ResubmitDataModule	Specifies a connection to the DAT_Resubmit module.	Yes
DataConnection	Specifies the connection to the BRM database.	Yes
PipelineCategory	Specifies the pipeline category. Default = **CDRPipeline**	Yes
StorableClass	Specifies the storable class used to store suspended batch records. Default =/suspended_batch/cdr See "Suspending CDR Files" in *BRM Suspending and Recycling Pipeline EDRs*.	Yes (for Batch file processing)
SuspenseFile	Specifies the batch suspense file this module generates: • **Path** Specifies the path where the data file will be written to. • **Prefix** Specifies the prefix for the resulting filename. • **Suffix** Specifies the suffix for the resulting filename. • **TempDataPrefix** Specifies the prefix for the temporary filename that is used while the file is being built.	Yes

Sample Registry

```
#---------------------------
# Batch Suspense FCT
#---------------------------
BatchSuspense
{
  ModuleName                = FCT_BatchSuspense
  Module
  {
    Active                  = TRUE
    ResubmitDataModule      = ifw.DataPool.ResubmitBatch
    DataConnection          = ifw.DataPool.LoginInfranet
    PipelineCategory        = CDRPipeline
    StorableClass           = /suspended_batch/cdr
    SuspenseFile
    {
       Path = ..
       Prefix = Framework
       Suffix = msg
       TempDataPrefix = rej_
          }
  }
}
```

EDR Container Fields

Table 3-20 lists the FCT_BatchSuspense EDR container fields.

Table 3-20 FCT_BatchSuspense EDR Container Fields

Alias Field Name Default Field Name	Type	Description
BATCH_ID HEADER.BATCH_ID	String	The unique identifier for the batch file. The batch file's BATCH_ID field is set in the HEADER block when it is received by the pipeline and this field receives its value from that field.
SEQUENCE_NUMBER HEADER.SEQUENCE_NUMBER	String	The suspense sequence number.
SENDER HEADER.SENDER	String	The sender.
TAP_PROCESSING_INFO HEADER.TAP_PROCESSING_INFO	String	Tap specific information (e.g TAP file name for a specific RAP etc). The format of this field is specific to TAP/RAP processing modules and this information created and interpreted by these modules
OVERRIDE_REASONS HEADER.OVERRIDE_REASONS	String	Contains the batch suspense reasons that are overridden from SMC while resubmitting the batch. Mapped from the error code. Used by Suspense Manager.

FCT_BillingRecord

The FCT_BillingRecord module consolidates balance impact data into an associated BRM billing record and one or more balance impact packets. See "About Consolidation for BRM Billing" in *BRM Setting Up Pipeline Rating and Discounting*.

> **Note:**
>
> - The FCT_ItemAssign module handles items assigned for usage events.
> - Don't use the FCT_BillingRecord module in a CIBER roaming revenue assurance environments. For more information, see "Billing Consolidation with CIBER Roaming and Revenue Assurance" in *BRM Setting Up Pipeline Rating and Discounting*.

Dependencies

Requires a connection to the following modules:

- DAT_AccountBatch
- DAT_BalanceBatch
- DAT_Currency
- DAT_ItemAssign

This module must run after the FCT_MainRating and FCT_Discount modules.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-21 lists the FCT_BillingRecord registry entries.

Table 3-21 FCT_BillingRecord Registry Entries

Entry	Description	Mandatory
Active	Specifies if the module is active or inactive. **TRUE** = Active **FALSE** = Inactive You can use this entry in a semaphore file.	Yes
AccountDataModule	Specifies a connection to the DAT_AccountBatch module.	Yes
BalanceDataModule	Specifies a connection to the DAT_BalanceBatch module.	Yes
ChargeBreakDownRecordType	By default, FCT_BillingRecord module processes Charge Breakdown records whose record type is 981. Use this entry to specify additional Charge Breakdown records. **Note:** You can specify multiple Charge Breakdown record types. Separate each record type by using a comma.	No

Table 3-21 (Cont.) FCT_BillingRecord Registry Entries

Entry	Description	Mandatory
CurrencyDataModule	Specifies a connection to the DAT_Currency module.	Yes
CurrencyType	Specifies the CHARGED_CURRENCY_TYPE value that the charge packets should contain. The possible values are: • **H** = Home currency • **B** = Billing currency • **R** = Rating currency Default = **H**.	No
ItemAssignDataModule	Specifies a connection to the DAT_ItemAssign module.	Yes
PortalConfigDataModule	Specifies the connection to the DAT_PortalConfig module.	Yes
RatingPipeline	Specifies whether the module is running in the rating or rerating pipeline: **FALSE** = Rerating **TRUE** = Rating	No
RatePlanType	By default, FCT_BillingRecord processes charge packets whose rateplan type is R. Use this entry to specify additional rateplan types.	No
SetTaxLocales	Specifies whether to view the resulting tax locales in the **PIN_TAX_LOCALES** field: **FALSE** = Viewing the resulting tax locales in the **PIN_TAX_LOCALES** field is disabled. **TRUE** = View the resulting tax locales in the **PIN_TAX_LOCALES** field; for example, phone numbers. Default = **FALSE**.	No

Sample Registry Entry

```
AddInfranetBillingRecord
{
  ModuleName = FCT_BillingRecord
  Module
  {
    Active = TRUE
    AccountDataModule  = ifw.DataPool.CustomerData
    PortalConfigDataModule   = ifw.DataPool.PortalConfigDataModule
    BalanceDataModule = ifw.DataPool.BalanceDataModule
    ChargeBreakDownRecordType = 981
    CurrencyType = R
    CurrencyDataModule = ifw.DataPool.CurrencyDataModule
    ItemAssignDataModule = ifw.DataPool.ItemAssignDataModule
    RatingPipeline = TRUE
    RatePlanType = W
    SetTaxLocales = FALSE
  }
}
```

Semaphore File Entries

Table 3-22 lists the FCT_BillingRecord Sempahore file entries.

Table 3-22 FCT_BillingRecord Semaphore File Entries

Entry	Description
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive

Sample Semaphore File Entry

```
ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.AddInfranetBillingRecord.
Module.Active = false
```

EDR Container Fields

Table 3-23 lists the FCT_BillingRecord EDR container fields.

Table 3-23 FCT_BillingRecord EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
A_NUMBER DETAIL.A_NUMBER	String	Read	Contains the normalized A number.
ACCOUNT_POID DETAIL.ASS_PIN.ACCOUNT_POID	String	Write	Contains the resulting account POID.
ACCOUNT_POID DETAIL.ASS_PIN.MP.ACCOUNT_POID	String	Create	POID of the account that the balance impact applies to. **Derivation:** Mandatory. Calculated.
ASS_CBD_IMPACT_CATEGORY DETAIL.ASS_CBD.CP.IMPACT_CATEGORY	String	Read	Contains the impact category.
ASS_CBD_RECORD_TYPE DETAIL.ASS_CBD.RECORD_TYPE	String	Read	Contains the record type of the associated charge record.
ASS_CBD_RUM_NAME DETAIL.ASS_CBD.RUM_NAME	String	Read	Contains the RUM name.
ASS_PIN_BALANCE_PACKET DETAIL.ASS_PIN.NUMBER_OF_BALANCE_PACKETS	Integer	Write	Contains the resulting number of balance packets.
B_NUMBER DETAIL.B_NUMBER	String	Read	Contains the normalized B number.
BAL_GRP_ID DETAIL.INTERN_BALANCE_GROUP_ID	String	Read	Contains the balance group POID of the service.
BAL_GRP_POID DETAIL.ASS_PIN.MP.BAL_GRP_POID	String	Create	Balance monitor group that the balance impact applies to. **Derivation:** Mandatory. Calculated.
BALANCE_GROUP_ID DETAIL.CUST_A.ML.BALANCE_GROUP_ID	String	Read	Balance monitor group ID.

Table 3-23 (Cont.) FCT_BillingRecord EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
BDR_UTC_TIME_OFFSET DETAIL.UTC_TIME_OFFSET	String	Read	Contains the UTC offset.
BP_ACCOUNT_POID DETAIL.ASS_PIN.BP.ACCOUNT_POID	String	Write	Contains account POID.
BP_RUM_ID DETAIL.ASS_PIN.BP.RUM_ID	Long	Read/Write	Contains RUM id of the balance packet.
CHARGED_AMOUNT_CURRENCY DETAIL.ASS_CBD.CP.CHARGED_AMOUNT_CURRENCY	String	Read	Contains the currency.
CHARGED_AMOUNT_VALUE_ORIG DETAIL.ASS_CBD.CP.CHARGED_AMOUNT_VALUE_ORIG	Decimal	Read	Contains the charged amount value.
CHARGED_AMOUNT_VALUE DETAIL.ASS_CBD.CP.CHARGED_AMOUNT_VALUE	Decimal	Read	Contains the charged amount value.
CHARGED_CURRENCY_TYPE DETAIL.ASS_CBD.CP.CHARGED_CURRENCY_TYPE	String	Read	Contains the currency type: • **H** = Home • **R** = Rating • **B** = Billing
CHARGED_TAX_CODE DETAIL.ASS_CBD.CP.CHARGED_TAX_CODE	String	Read	Contains the charged tax code.
CHARGING_END_TIMESTAMP DETAIL.CHARGING_END_TIMESTAMP	Date	Read	Contains charging end time.
CHARGING_START_TIMESTAMP DETAIL.CHARGING_START_TIMESTAMP	Date	Read	Contains charging start time.
CONNECT_SUBTYPE DETAIL.CONNECT_SUBTYPE	String	Read	Contains the connect subtype.
DETAIL.ASS_CBD.CP.ITEM_TAG	String	Read	Contains the name of the charge item for balance impacts.
DETAIL.ASS_CBD.CP.ITEM_POID	String	Read	Contains the POID of the associated charge item for balance impacts.
DETAIL.ASS_CBD.DP.ITEM_TAG	String	Read	Contains the name of the discount item for balance impacts.
DETAIL.ASS_CBD.DP.ITEM_POID	String	Read	Contains the POID of the associated discount item for balance impacts.
DETAIL.ASS_CBD.TP.ITEM_TAG	String	Read	Contains the name of the tax item for balance impacts.
DETAIL.ASS_CBD.TP.ITEM_POID	String	Read	Contains the POID of the associated tax item for balance impacts.
CP_RUM_ID DETAIL.ASS_CBD.CP.RUM_ID	Long	Read	Contains RUM id.

Table 3-23 (Cont.) FCT_BillingRecord EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
DETAIL.ASS_CBD.CP.RECORD_TYPE	String	Read	Contains charge packet record type.
DETAIL.ASS_CBD.DP	Record	Read	The discount packet record.
DETAIL.ASS_CBD.DP.BALANCE_GROUP_ID	Integer	Read	Contains the ID of the balance group that the discount applies to.
DETAIL.ASS_CBD.DP.GLID	String	Read	Contains the G/L ID.
DETAIL.ASS_CBD.DP.GRANTED_AMOUNT	Decimal	Read	Contains the discount amount granted. This can be currency or noncurrency.
DETAIL.ASS_CBD.DP.GRANTED_QUANTITY	Decimal	Read	Contains the discount base values used to calculate the amount granted.
DETAIL.ASS_CBD.DP.IMPACT_CATEGORY	String	Read	Contains the discount impact category.
DETAIL.ASS_CBD.DP.NODE_LOCATION	String	Read	Contains the node location of the discount.
DETAIL.ASS_CBD.DP.OBJECT_ACCOUNT	String	Read	Contains the POID of the discount owner.
DETAIL.ASS_CBD.DP.OBJECT_ID	String	Read	Contains the ID of the discount or sponsor object.
DETAIL.ASS_CBD.DP.OBJECT_OWNER_ID	String	Read	Contains the POID type of the discount owner.
DETAIL.ASS_CBD.DP.OBJECT_OWNER_TYPE	String	Read	Contains the POID type of the discount owner.
DETAIL.ASS_CBD.DP.OBJECT_TYPE	String	Read	Contains the discount or sponsor object that generated the discount.
DETAIL.ASS_CBD.DP.RATETAG	String	Read	Contains the rate tag for the discount.
DETAIL.ASS_CBD.DP.RESOURCE_ID	String	Read	Contains the ID of the balance element impacted.
DETAIL.ASS_CBD.DP.SUB_BALANCE.AMOUNT	Decimal	Read	Contains the amount applied to this sub-balance.
DETAIL.ASS_CBD.DP.SUB_BALANCE.CONTRIBUTOR	String	Read	Contains the sub-balance contributor field.
DETAIL.ASS_CBD.DP.SUB_BALANCE.REC_ID	Integer	Read	Contains the record ID of the sub-balance record.
DETAIL.ASS_CBD.DP.SUB_BALANCE.VALID_FROM	Date	Read	Contains the date the balance element is valid from.
DETAIL.ASS_CBD.DP.SUB_BALANCE.VALID_TO	Date	Read	Contains the date the balance element is valid to.
DETAIL.ASS_CBD.DP.TAX_CODE	String	Read	Contains the tax code for the discount.

Table 3-23 (Cont.) FCT_BillingRecord EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
DETAIL.ASS_PIN.BI.PIN_RATE_TAG	String	Write	Contains the name of the charge that provided a promotional savings.
DETAIL.ASS_PIN.BP.BAL_GRP_POID	String	Write	Contains the POID of the balance group that is impacted.
DETAIL.ASS_PIN.BP.ITEM_POID	String	Write	Contains the POID of the associated item.
DETAIL.ASS_PIN.SBI	Record	Write	The sub-balance impact record.
DETAIL.ASS_PIN.SBI.BAL_GRP_POID	String	Write	Contains the POID of the balance group that is impacted.
DETAIL.ASS_PIN.SBI.PIN_RESOURCE_ID	Integer	Write	Contains the balance element ID.
DETAIL.ASS_PIN.SBI.RECORD_NUMBER	Integer	Write	Contains the record number of the sub-balance impact record.
DETAIL.ASS_PIN.SBI.RECORD_TYPE	String	Write	Contains the record type of the sub-balance impact record.
DETAIL.ASS_PIN.SBI.SB	Record	Write	The sub-balance record.
DETAIL.ASS_PIN.SBI.SB.CONTRIBUTOR	String	Write	Contains the sub-balance contributor field.
DETAIL.ASS_PIN.SBI.SB.PIN_AMOUNT	Decimal	Write	Contains the amount of the sub-balance impact.
DETAIL.ASS_PIN.SBI.SB.RECORD_NUMBER	Integer	Write	Contains the record number of the sub-balance record.
DETAIL.ASS_PIN.SBI.SB.RECORD_TYPE	String	Write	Contains the record type of the sub-balance record.
DETAIL.ASS_PIN.SBI.SB.VALID_FROM	Date	Write	Contains the date the sub-balance balance element is valid.
DETAIL.ASS_PIN.SBI.SB.VALID_TO	Date	Write	Contains the date the sub-balance balance element is no longer valid.
DETAIL.CUST_A.ACCOUNT_PARENT_ID	String	Read	Contains the BRM account ID.
DETAIL.CUST_A.INTERN_FOUND_PP_INDEX	Long	Read	Contains the index of the purchased charge offer.
DETAIL.CUST_A.PRODUCT.OFFERING_POID	String	Read	Contains the charge offer's node location.
DETAIL.CUST_A.PRODUCT.PRODUCT_ID	String	Read	Contains the BRM charge offer ID.
DETAIL.CUST_A.PRODUCT.SERVICE_ID	String	Read	Contains the ID of the charge offer.

Table 3-23 (Cont.) FCT_BillingRecord EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
DETAIL.CUST_A.PRODUCT.SERVICE_TYPE	String	Read	Contains the service type of the charge offer.
DETAIL.CUST_A.PRODUCT.SERVICE_USED_ITEM_POID	String	Read	Contains the item POID.
DETAIL_OBJECT_CACHE_TYPE DETAIL.OBJECT_CACHE_TYPE	Long	Write	Contains the cache type of the associated object.
DP_AMOUNT_ORIG DETAIL.ASS_CBD.DP.GRANTED_AMOUNT_ORIG	Decimal	Read	Contains original discount amount
DP_INTERN_RUM_ID DETAIL.ASS_CBD.DP.INTERN_RUM_ID	Long	Read	Contains discount RUM id.
DP_OFFERING_POID DETAIL.ASS_CBD.DP.OFFERING_POID	String	Read	Contains discount offering POID.
DP_RESOURCE_ID_ORIG DETAIL.ASS_CBD.DP.RESOURCE_ID_ORIG	Long	Read	Contains original balance element ID.
DP_SB_GRANTOR DETAIL.ASS_CBD.DP.SUB_BALANCE.GRANTOR	String	Read	Contains discount sub-balance grantor.
DP_SB_VALID_FROM_DETAILS DETAIL.ASS_CBD.DP.SUB_BALANCE.VALID_FROM_DETAILS	Long	Read	Contains discount sub-balance valid from details.
DP_SB_VALID_TO_DETAILS DETAIL.ASS_CBD.DP.SUB_BALANCE.VALID_TO_DETAILS	Long	Read	Contains discount sub-balance valid to details.
EXCHANGE_CURRENCY DETAIL.ASS_CBD.CP.EXCHANGE_CURRENCY	String	Read	Contains currency for exchange.
EXPIRED_UNITS DETAIL.ASS_CBD.CP.DP. EXPIRED_UNITS	String	Write	Contains the number of units that exceed the maximum.
GRANTED_DISCOUNT_AMOUNT_VALUE DETAIL.ASS_CBD.CP.GRANTED_DISCOUNT_AMOUNT_VALUE	Decimal	Read	Contains the granted discount amount value.
GRANTOR DETAIL.ASS_PIN.SBI.SB.GRANTOR	String	Write	Contains grantor of sub-balance impact.
ITEM_POID DETAIL.ASS_PIN.ITEM_POID	String	Write	Contains the resulting item POID.
ITEM_TAG DETAIL.ITEM_TAG	String	Read	Contains the item tag.
MONITOR_OWNER_ACCT_ID DETAIL.CUST_A.ML.MONITOR_OWNER_ACCT_ID	String	Read	Monitor owner's account ID.
MONITOR_OWNER_ID DETAIL.CUST_A.ML.MONITOR_OWNER_ID	String	Read	Monitor owner ID.
MONITOR_OWNER_TYPE DETAIL.CUST_A.ML.MONITOR_OWNER_TYPE	String	Read	Monitor owner type.
MONITOR_SUB_BAL DETAIL.ASS_PIN.MSBI.MONITOR_SUB_BAL	SB	Create	Sub-balance monitored.

Table 3-23 (Cont.) FCT_BillingRecord EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
MSBI_BAL_GRP_POID DETAIL.ASS_PIN.MSBI.BAL_GRP_POID	String	create	Contains balance group POID of monitor sub-balance impact.
MSBI_MSB_PIN_AMOUNT DETAIL.ASS_PIN.MSBI.MSB.PIN_AMOUNT	Decimal	Write	Contains amount of monitor sub-balance.
MSBI_MSB_RECORD_NUMBER DETAIL.ASS_PIN.MSBI.MSB.RECORD_NUMBER	Integer	Write	Contains record number of monitor sub-balance.
MSBI_MSB_RECORD_TYPE DETAIL.ASS_PIN.MSBI.MSB.RECORD_TYPE	String	Write	Contains record type of monitor sub-balance.
MSBI_MSB_VALID_FROM DETAIL.ASS_PIN.MSBI.MSB.VALID_FROM	Date	Write	Contains valid from of monitor sub-balance.
MSBI_MSB_VALID_TO DETAIL.ASS_PIN.MSBI.MSB.VALID_TO	Date	Write	Contains valid to of monitor sub-balance.
MSBI_PIN_RESOURCE_ID DETAIL.ASS_PIN.MSBI.PIN_RESOURCE_ID	Long	Write	Contains balance element id of monitor sub-balance impact.
MSBI_RECORD_TYPE DETAIL.ASS_PIN.MSBI.RECORD_TYPE	String	Write	Contains record type of monitor sub-balance impact.
NET_QUANTITY DETAIL.NET_QUANTITY	Decimal	Write	Contains net quantity.
OBJECT_CACHE_TYPE DETAIL.ASS_PIN.BP.OBJECT_CACHE_TYPE	Integer	Write	Contains the cache type of the associated object. **Possible values:** • **2** (POSTPAID) • **0** (CONVERGENT).
ORIGINAL_EVENT_POID DETAIL.ASS_PIN.ORIGINAL_EVENT_POID	String	Read	Contains original event POID.
PIN_AMOUNT_DEFERRED DETAIL.ASS_PIN.BP.PIN_AMOUNT_DEFERRED	Integer	Write	Specifies whether an EDR contains tax data. This field is set to **0** when an EDR contains a tax packet and a PIN_AMOUNT when an EDR doesn't contain a tax packet.
PIN_AMOUNT_ORIG DETAIL.ASS_PIN.BP.PIN_AMOUNT_ORIG	Decimal	Read/Write	Contains original amount.
PIN_AMOUNT DETAIL.ASS_PIN.BP.PIN_AMOUNT	Decimal	Write	Contains the resulting amount used to update BRM balances.

Table 3-23 (Cont.) FCT_BillingRecord EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
PIN_AMOUNT DETAIL.ASS_PIN.MP.PIN_AMOUNT	9(11)	Create	Amount of one balance element impact for the account balance. The value can be either positive or negative. The value is added to the PIN_FLD_CURRENT_BAL field of the PIN_FLD_BALANCES array in the account object specified by PIN_FLD_ACCOUNT_OBJ. **Note:** In case of multiple-RUM rating, this value might be a total value. Possible values: Price. See below for minimum and maximum values.If no price is given, this is a space, for example, NULL in a database. The format is variable floating point. The floating decimal point must be set if the given value is not in the required format. Maximum value: 99999999999 Minimum value: -9999999999 Examples: • '00000000125' for 125,00 • '00000012.50' for 12,50 • '-0012.56780' for -12,5678 Derivation: Mandatory. Derived from the object, **/event/PIN_FLD_BAL_IMPACTS.PIN_FLD_AMOUNT**. The post-mapping processor decides what mapping rule applies to this attribute; for example, add multiple charge packet values. **Note:** This value does not include any granted discounts.
PIN_DISCOUNT DETAIL.ASS_PIN.BP.PIN_DISCOUNT	Decimal	Write	Contains the resulting discount values.
PIN_GL_ID DETAIL.ASS_PIN.BP.PIN_GL_ID	String	Write	Contains the resulting G/L ID.
PIN_IMPACT_CATEGORY DETAIL.ASS_PIN.BP.PIN_IMPACT_CATEGORY	String	Write	Contains the impact category from the charge packet or the discount packet, depending on which was passed.
PIN_IMPACT_TYP DETAIL.ASS_PIN.BP.PIN_IMPACT_TYPE	Long	Write	Contains impact type.

Table 3-23 (Cont.) FCT_BillingRecord EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
PIN_INFO_STRING DETAIL.ASS_PIN.BP.PIN_INFO_STRING	String	Read/Write	Contains information of balance packet.
PIN_INVOICE_DATA DETAIL.ASS_PIN.PIN_INVOICE_DATA	String	Write	Contains the resulting BRM invoice data.
PIN_OFFERING_POID DETAIL.ASS_PIN.BP.PIN_OFFERING_POID	String	Write	Uniquely identifies an account charge offer, discount, or sponsor.
PIN_PRODUCT_POID DETAIL.ASS_PIN.BP.PIN_PRODUCT_POID	String	Write	Contains the resulting charge offer object.
PIN_QUANTITY DETAIL.ASS_PIN.BP.PIN_QUANTITY	Decimal	Write	Contains the resulting quantity used to update BRM accounts.
PIN_RATE_TAG DETAIL.ASS_PIN.BP.PIN_RATE_TAG	String	Write	Contains the rate tag from the discount packet.
PIN_RESOURCE_ID_ORIG DETAIL.ASS_PIN.BP.PIN_RESOURCE_ID_ORIG	Long	Write	Contains original balance element id.
PIN_RESOURCE_ID DETAIL.ASS_PIN.BP.PIN_RESOURCE_ID	String	Write	Contains the resulting ID of the BRM balance element.
PIN_RESOURCE_ID DETAIL.ASS_PIN.MP.PIN_RESOURCE_ID	9(9)	Create	Numeric value of the balance element that is impacted; for example, 840 for US dollars. Possible value: Any configured BRM balance element ID. **Derivation:** Mandatory.
PIN_TAX_CODE DETAIL.ASS_PIN.BP.PIN_TAX_CODE	String	Write	Contains the resulting BRM tax code.
PRICEMODEL_TYPE DETAIL.ASS_CBD.CP.PRICEMODEL_TYPE	String	Read	Contains the pricing type.
RATEPLAN_CODE DETAIL.ASS_CBD.CP.RATEPLAN_CODE	String	Read	Contains rateplan code.
RATEPLAN_TYPE DETAIL.ASS_CBD.CP.RATEPLAN_TYPE	String	Read	Contains the charge type.
RECORD_TYPE DETAIL.ASS_PIN.MP.RECORD_TYPE	String	Create	Extended to be 3 bytes long. Possible values: • 800: monitor packet • 805: monitor sub-balance impact • 807: monitor sub-balance
RESOURCE_ID DETAIL.ASS_CBD.CP.RESOURCE_ID	Long	Read	Contains charge packet balance element ID.
RESOURCE_ORIG DETAIL.ASS_CBD.CP.RESOURCE_ID_ORIG	Long	Read	Contains charge packet original balance element ID.

Table 3-23 (Cont.) FCT_BillingRecord EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
RM_NET_QUANTITY DETAIL.ASS_CBD.RM.NET_QUANTITY	Decimal	Read/Write	Contains net quantity of RUM.
ROUNDED_QUANTITY_VALUE DETAIL.ASS_CBD.CP.ROUNDED_QUANTITY_VALUE	Decimal	Read	Contains the rounded quantity value that was used for the price calculations.
RUM_NAME DETAIL.ASS_PIN.RUM_NAME	String	Write	Contains RUM name.
SERVICE_POID DETAIL.ASS_PIN.SERVICE_POID	String	Write	Contains the resulting service POID.
TAX_LOCALES DETAIL.ASS_PIN.PIN_TAX_LOCALES	String	Write	Contains the resulting tax locales string.
TJ_PIN_AMOUNT DETAIL.ASS_PIN.BP.TJ.PIN_AMOUNT	Decimal	create	Contains amount of tax jurisdiction of balance packet.
TJ_PIN_TAX_RATE DETAIL.ASS_PIN.BP.TJ.PIN_TAX_RATE	String	Create	Contains tax rate of tax jurisdiction of balance packet.
TJ_PIN_TAX_TYPE DETAIL.ASS_PIN.BP.TJ.PIN_TAX_TYPE	String	Read/create	Contains tax type of tax jurisdiction of balance packet.
TJ_PIN_TAX_VALUE DETAIL.ASS_PIN.BP.TJ.PIN_TAX_VALUE	Decimal	Create	Contains tax value of tax jurisdiction of balance packet.
TJ_RECORD_TYPE DETAIL.ASS_PIN.BP.TJ.RECORD_TYPE	String	Create	Contains record type of tax jurisdiction of balance packet.
TP_RELATED_CHARGE_INFO_ID DETAIL.ASS_CBD.TP.RELATED_CHARGE_INFO_ID	Long	Read	Contains related charge info id.
TP_TAX_CODE DETAIL.ASS_CBD.TP.TAX_CODE	String	Read	Contains tax code.
TP_TAX_RATE DETAIL.ASS_CBD.TP.TAX_RAT	String	Read	Contains tax rate.
TP_TAX_TYPE DETAIL.ASS_CBD.TP.TAX_TYPE	String	Read	Contains tax type.
TP_TAX_VALUE_ORIG DETAIL.ASS_CBD.TP.TAX_VALUE_ORIG	Decimal	Read	Contains original tax value.
TP_TAX_VALUE DETAIL.ASS_CBD.TP.TAX_VALUE	Decimal	Read	Contains tax value.
TP_TAXABLE_AMOUNT DETAIL.ASS_CBD.TP.TAXABLE_AMOUNT	Decimal	Read	Contains taxable amount.
USAGE_GL_ACCOUNT_CODE DETAIL.ASS_CBD.CP.USAGE_GL_ACCOUNT_CODE	String	Read	Contains G/L account code.
VALID_FROM_DETAILS DETAIL.ASS_PIN.SBI.SB.VALID_FROM_DETAILS	Long	Write	Contains valid from details of sub-balance impact.

Table 3-23 (Cont.) FCT_BillingRecord EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
VALID_FROM DETAIL.ASS_PIN.MSB.VALID_FROM	Date	Create	Valid from date for this sub-balance.
VALID_TO_DETAILS DETAIL.ASS_PIN.SBI.SB.VALID_TO_DETAILS	Long	Write	Contains valid to details of sub-balance impact.
VALID_TO DETAIL.ASS_PIN.MSB.VALID_TO	Date	Create	Valid-to date for this sub-balance.

FCT_CallAssembling

The FCT_CallAssembling module assembles the multiple CDRs that comprise a single wireless call into a single EDR that Pipeline Manager can process. See "Assembling EDRs" in *BRM Setting Up Pipeline Rating and Discounting*.

Dependencies

You must run this module early in a pipeline to assemble EDRs. You must run it before FCT_Discard.

When you configure the FCT_CallAssembling function module to not drop EDRs from the pipeline, ensure that the FCT_AggreGate function module that counts them runs before the FCT_Reject function module.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-24 lists the FCT_CallAssembling registry entries.

Table 3-24 FCT_CallAssembling Registry Entries

Entry	Description	Mandatory
Active	Turns FCT_CallAssembling module processing on and off. **True** = Active **False** = Inactive You can use this entry in a semaphore file.	Yes
AssembledEDR	Specifies a list of fields that the EDR takes from the **L** call segment and appends it to the last EDR (if the two are different). No default entry. See "Capturing Fields From the Last Call Record" in *BRM Setting Up Pipeline Rating and Discounting*.	No

Table 3-24 (Cont.) FCT_CallAssembling Registry Entries

Entry	Description	Mandatory
AssembleSGSN	Turns SGSN data capture on and off. If **True**, this entry waits for all CDRs to arrive before rating a call. If your system does not process TAP records, leave this set to **False** to save system resources. **True** = SGSN data recorded **False** = SGSN data not recorded Default = **False** See "Rating Calls by Volume of Data Sent" in *BRM Setting Up Pipeline Rating and Discounting*.	No
AssembleVolume	Turns volume rating on and off. If **True**, this entry waits for all CDRs to arrive before rating a call. If your system does not require volume rating, leave this set to **False** to save system resources. **True** = volume rating on **False** = volume rating off Default = **False** See "Rating Calls by Volume of Data Sent" in *BRM Setting Up Pipeline Rating and Discounting*.	No
CallDurationTolerance	Specifies an allowable cumulative time error for a single call (in seconds). Used with **SplitAtGaps = True**. Default = **60** See "Specifying a Time Error" in *BRM Setting Up Pipeline Rating and Discounting*.	No
DropLateCDRs	Specifies how to handle the output of late EDRs: • **True** = Drop late EDRs from the pipeline. • **False** = Send late EDRs through the pipeline as non-valid. Default = **True** See "Dropping Late Calls" in *BRM Setting Up Pipeline Rating and Discounting*.	No
EmitPartialEDROnUpgrade	Specifies the results of the **UpgradeFlushLimit** semaphore. Results are one of the following: • Silently drops EDRs from the in-memory **.dat** file. • Emit partial EDRs for revenue assurance tracking. (Partial EDRs should be sent to the discard stream.) Default = **False** (disabled)	No
FileName	Specifies the base file name for the data files. The transaction ID and the suffix are appended. See "Managing the Call Assembling Data Files" in *BRM Setting Up Pipeline Rating and Discounting*.	Yes
MaxDuration	Directs FCT_CallAssembling to rate segments of a wireless call periodically. This entry specifies the maximum amount of time (in seconds) that a call can remain open before FCT_CallAssembling rates the segments that have arrived. This module recalculates the call duration for every call each time a new call segment arrives and compares it to the **MaxDuration** setting. If the new time duration equals or exceeds the setting for **MaxDuration,** FCT_CallAssembling emits an EDR to rate the existing portion of the call. For details and a comparison to **FlushLimit**, see "Rating Calls by Time Duration" in *BRM Setting Up Pipeline Rating and Discounting*. No default entry.	No

Table 3-24 (Cont.) FCT_CallAssembling Registry Entries

Entry	Description	Mandatory
Mode	Change this entry *only* if you are creating data upgrade pipelines that are used when changing an EDR container description. The possible values are: **Normal**. The default mode, and the most common use of this module. Directs FCT_CallAssembling to assemble CDRs into EDRs so Pipeline Manager can process them. **RestoreEDRs**. Directs FCT_CallAssembling to read serialized EDRs in sequence from data files and inserts them into the pipeline. **UpGradeData**. Directs FCT_CallAssembling to update data files based on the EDRs it receives. Default = **Normal**	Yes
Path	Specifies the directory for the data files. Default = **.** See "Managing the Call Assembling Data Files" in *BRM Setting Up Pipeline Rating and Discounting*.	No
RejectMissingChain	Specifies whether to report an error if a chain reference value is missing. Default = **False**	No
SplitAtGaps	Specifies whether a non-contiguous set of CDRs can be collected into a single EDR. For example, assume that CDRs F, I1, I2, and I4 have arrived. If set to **True**, this entry directs FCT_CallAssembling to emit an EDR for F, I1, and I2, and because I3 is missing, a separate EDR for I4. If set to **False**, all CDRs that have arrived are collected into a single EDR. If set to **True**, FCT_CallAssembling will emit multiple EDRs if a CDR is missing when the call. **True** = Active **False** = Inactive Default = **False** See "Rating Calls by Volume of Data Sent" in *BRM Setting Up Pipeline Rating and Discounting*.	No

Startup Registry Interdependencies

The Following Section Explains The Relationships Between Certain Startup Registry Entries.

Sample Registry

```
CallAssembling
{
   ModuleName = FCT_CallAssembling
   Module
    {
      Active = True
      Path = .
      FileName = calls
      RejectMissingChain = False
      AssembleVolume = TRUE
      AssembledEDR  {
          1 = Detail.custom_fields_from_last_edr1
          2 = Detail.custom_field_from_last_edr2...
```

```
                    }
                }
            }
```

Semaphore File Entries

Table 3-25 lists the FCT_CallAssembling Semaphore file entries.

Table 3-25 FCT_CallAssembling Semaphore File Entries

Entry	Description
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive
ExportDataToXml	Exports the call data in the existing data file to an XML file with the name specified by the **FileName** entry in the startup registry file. See "Migrating Call Assembling Data Between Releases and Pipelines" in *BRM Setting Up Pipeline Rating and Discounting*.
ExportDataToXML.CallsPerFile	If the number of calls exported is larger than the resources available in the host system, you can divide the call data into multiple files by using this option and specifying the number of calls per file. See "Migrating Call Assembling Data Between Releases and Pipelines" in *BRM Setting Up Pipeline Rating and Discounting*.
FlushLimit	Sets the maximum age (in days) an open (incomplete) EDR can have before being flushed from the work files. For example, a setting of **0** flushes all open calls; a setting of **1** flushes all calls that have been open for a day or more; a setting of **2** flushes all calls that have been open for two days or more, and so on. **Note:** The setting of **0** does not flush future-dated EDRs because the value of CHARGING_START_TIMESTAMP is greater than the system date. No default value. See "Rating Incomplete Time Duration Calls" in *BRM Setting Up Pipeline Rating and Discounting*.
FlushServiceCode	Used with **FlushLimit**. Specifies a service. When used, only the calls with the service that match the three-letter service code are flushed. Multiple entries are not allowed. No default value. See "Rating Partial Calls by Service" in *BRM Setting Up Pipeline Rating and Discounting*.
ImportDataFromXml	Imports the entire contents of the XML file created by the **ExportDataToXml** entry to the **.dat** file in the new format. **Values:** See "Migrating Call Assembling Data Between Releases and Pipelines" in *BRM Setting Up Pipeline Rating and Discounting*.
ImportDataFromXML.FileName	Specifies the XML file from which to import data. See "Migrating Call Assembling Data Between Releases and Pipelines" in *BRM Setting Up Pipeline Rating and Discounting*.

Table 3-25 (Cont.) FCT_CallAssembling Semaphore File Entries

Entry	Description
KeepCallOpen	Used with **FlushLimit**. Specifies whether to rate additional EDRs for a call that has already been flushed (**True**). Default = **False** See: • Rating Calls by Implied Time Duration in *BRM Setting Up Pipeline Rating and Discounting*. • Rating Continuous Data Calls by Segment in *BRM Setting Up Pipeline Rating and Discounting*. • Rating Partial Calls by Service in *BRM Setting Up Pipeline Rating and Discounting*.
RemoveLimit	Sets a time limit (in days) for removing EDRs in a **Closed** or **Timeout** state from the work files. No default value. See "Removing Incomplete Time Duration Calls" in *BRM Setting Up Pipeline Rating and Discounting*.
RemoveRejectedLimit	Sets a time limit (in days) for removing EDRs in a **Closed_Rejected** or **Timeout_Rejected** state from the work files. No default value. See "Removing Incomplete Time Duration Calls" in *BRM Setting Up Pipeline Rating and Discounting*.
UpgradeFlushLimit	Flushes partial EDRs that were closed as a result of a change to the EDR container. No default value (no limit).

Sample FlushLimit Semaphore Commands

```
ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.CallAssembling.Module.FlushLimit=1

ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.CallAssembling.
Module.FlushServiceCode = tel

ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.CallAssembling.
Module.KeepCallOpen = True
```

Semaphore Entries for a Call-Assembling Report

For information, see "Tracking the Status of Assembled Calls" in *BRM Setting Up Pipeline Rating and Discounting*.

Table 3-26 lists the semaphore entries for Call-Assembling Report.

Table 3-26 Semaphore Entries for Call-Assembling Report

Entry	Description	Mandatory
CreateReport	Command to create the report.	Yes

Table 3-26 (Cont.) Semaphore Entries for Call-Assembling Report

Entry	Description	Mandatory
EndTime	Specifies the end date and time for the report. EDRs created before this date and time are reported. The format is *YYYYMMDDhhmmss*. Default = **0** (Current time)	No
ReportPath	Specifies path of the report file. Default = **.**	No
ReportPrefix	Specifies the file name prefix of the report file. Default = **assembly**	No
StartTime	Specifies the start date and time for the report. EDRs created on or after this date and time are reported. The format is *YYYYMMDDhhmmss*.	No

Sample semaphore command for call assembling reports

```
ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.CallAssembling.Module
{
     CreateReport      = True
     StartTime         = 20111206000000
     Module.StartTime  = 20020315000000
     EndTime           = 20111212090000
     ReportPath        = ./
     ReportPrefix      = call_assembly
}
```

EDR Container Fields

Table 3-27 lists the EDR container fields for Call-Assembling Report.

Table 3-27 EDR Container Fields for Call-Assembling Report

Alias Field Name Default Field Name	Type	Access	Description
CHAIN_REFERENCE DETAIL.CHAIN_REFERENCE	String	Read	Contains the chain reference key.

Table 3-27 (Cont.) EDR Container Fields for Call-Assembling Report

Alias Field Name Default Field Name	Type	Access	Description
LONG_DURATION_INDICATOR DETAIL.LONG_DURATION_INDICATOR	String	Read/Write	Contains the long duration indicator. Arriving call segments have one of these: • **F** = First • **I** = Intermediate • **L** = Last Assembled call segments are given one of these: • **C** = Complete call. • **SL** = *Slice* (portion) of a call. • **P** = Partially assembled call. • **XC** = Late intermediate call segment. • **XO** = Late overlap segment. • **XP** =Late segment (any) of a call. See "How FCT_CallAssembling Classifies EDRs" in *BRM Setting Up Pipeline Rating and Discounting*.
TRANSACTION_ID	Decimal	Read	Contains the transaction ID.
PROCESS_STATUS	Long	Read	Contains the EDR status.
CHARGING_START_TIMESTAMP DETAIL.CHARGING_START_TIMESTAMP	Date	Read/Write	Contains the charging time stamp.
DURATION DETAIL.DURATION	Decimal	Read/Write	Contains the duration of the assembled EDR.
VOLUME_SENT DETAIL.VOLUME_SENT	Decimal	Read/Write	Contains the volume sent for the assembled EDR.
VOLUME_RECEIVED DETAIL.VOLUME_RECEIVED	Decimal	Read/Write	Contains the volume received for the assembled EDR.
NUMBER_OF_UNITS DETAIL.NUMBER_OF_UNITS	Decimal	Read/Write	Contains the number of units for the assembled EDR.
RETAIL_CHARGED_AMOUNT_VALUE DETAIL.RETAIL_CHARGED_AMOUNT_VALUE	Decimal	Read/Write	Contains the retail charged amount value for the assembled EDR.
WHOLESALE_CHARGED_AMOUNT_VALUE DETAIL.WHOLESALE_CHARGED_AMOUNT_VALUE	Decimal	Read/Write	Contains the wholesale charged amount value for the assembled EDR.
NUMBER_OF_CDRS DETAIL.NUMBER_OF_CDRS	Integer	Write	The number of CDRs assembled in the EDR.

FCT_CancelTimer

The FCT_CancelTimer module checks the TimerID to identify the EDR and the timeout flag to verify if the EDR is valid or timed out. If the time out flag is set to **False**, FCT_CancelTimer cancels the timeout flag in the EDR so that the EDR can be sent for further processing.

If the timeout flag is set to **True**, it means there is a duplicate EDR and the FCT_CancelTimer discards the EDR.

Dependencies

FCT_CancelTimer depends on the FCT_Timer in the Dispatcher pipeline for the TimerID and the timeout flag values.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-28 lists the FCT_CancelTimer registry entries.

Table 3-28 FCT_CancelTimer Registry Entries

Entry	Description	Mandatory
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive You can use this entry in a semaphore file.	Yes
StreamName	The output queue to which the timed out EDR is sent.	Yes

Sample Registry

```
CancelTimer
{
  ModuleName = FCT_CancelTimer
  Module
  {
    Active = TRUE
    StreamName = ExceptionOutput
  }
}
```

EDR Container Fields

FCT_CancelTimer uses the EDR container fields listed in Table 3-29:

Table 3-29 FCT_CancelTimer Container Fields

Alias Field Name Default Field Name	Type	Access	Description
TIMER_ID DETAIL.TIMER_ID	Integer	Read	Contains the timer ID needed to cancel the timer

FCT_CarrierIcRating

The FCT_CarrierIcRating module adds roaming/interconnect data to EDRs for rating by the FCT_PreRating and FCT_MainRating modules.

Dependencies

Requires a connection to the Pipeline Manager database.

All rating and mapping related modules should be placed in the pipeline.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-30 lists the FCT_CarrierIcRating registry entries.

Table 3-30 FCT_CarrierIcRating Registry Entries

Entry	Description	Mandatory
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive You can use this entry in a semaphore file.	Yes
EdrNetworkModel	Specifies the network model. This entry identifies your network as the home network. You can specify one home network per pipeline. You can use this entry in a semaphore file.	Yes
IcProductGroup	Specifies the IC charge offer group that contains the IC charge offers. This field is mandatory for all modes except CARRIER_IC mode.	Yes
InterConnectDataModule	Specifies a connection to the DAT_Interconnect module.	Yes
Mode	Specifies the evaluation path for finding the IC-charge offer. • If you specify ROAMING, IC charge offers are found using the **IcProductGroup** registry entry. • If you specify CARRIER_IC, the module assigns a charge by using trunk information from the EDR.	Yes

Table 3-30 (Cont.) FCT_CarrierIcRating Registry Entries

Entry	Description	Mandatory
RecordTypeField	Specifies the EDR field that contains the record type. The record type is used to search the IFW_ICPRODUCT_CNF database table for matching records. When processing CIBER record types, this entry is used to find the IC Charge Offers and the corresponding charge to use for rating the CIBER records.	No
UseRateplan	Specifies how the price is calculated: • **STANDARD**. The price is calculated using the specified charge. • **ALTERNATIVE**. The price is calculated using the alternative charge. If you entered an alternative charge when configuring the IC charge offer, you can specify whether to use the alternate charge. You can use this entry in a semaphore file.	Yes

Sample Registry

```
Module
{
  Active = TRUE
  InterConnectDataModule = integRate.DataPool.InterConnect
  EdrNetworkModel = OWN
  UseRateplan = STANDARD
  Mode = ROAMING
  IcProductGroup = PG_ROAM
  RecordTypeField = DETAIL.ASS_CIBER_EXT.CIBER_RECORD_TYPE
}
```

Semaphore File Entries

Table 3-31 lists the FCT_CarrierIcRating Semaphore file entries.

Table 3-31 FCT_CarrierIcRating Semaphore File Entries

Entry	Description
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive
EdrNetworkModel	Specifies the network model to be used (CODE from table IFW_NETWORKMODEL).
UseRateplan	• **STANDARD:** IC-Price will be calculated using the charge from IFW_ICPRODUCT_RATE. • **ALTERNATIVE:** IC-Price will be calculated using the alternative charge from IFW_ICPRODUCT_RATE.

Sample Semaphore File Entry

```
ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.
CarrierIcRating.Module.Active = TRUE

ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.
```

```
CarrierIcRating.Module.EdrNetworkModel = OTHER

ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.
CarrierIcRating.Module.UseRateplan = ALTERNATIVE
```

EDR Container Fields

Table 3-32 lists the FCT_CarrierIcRating EDR container fields.

Table 3-32 FCT_CarrierIcRating EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
ASS_CBD DETAIL.ASS_CBD	Block	Create	The associated charge breakdown record created to hold the mapping data.
ASS_CBD_CHARGE_PACKET DETAIL.ASS_CBD.CP	Block	Create	The charge packet created to hold the mapping data.
TRUNK_INPUT DETAIL.ASS_GSMW_EXT.TRUNK_INPUT	String	Read	Contains the input trunk search value from the IFW_TRUNK_CNF table.
TRUNK_OUTPUT DETAIL.ASS_GSMW_EXT.TRUNK_OUTPUT	String	Read	Contains the output trunk search value from the IFW_TRUNK_CNF table.
ASS_GSMW_ORIGINATING_SWITCH_IDENTIFICATION DETAIL.ASS_GSMW_EXT.ORIGINATING_SWITCH_IDENTIFICATION	String	Read	Contains the switch search value from the IFW_TRUNK_CNF table.
SOURCE_NETWORK DETAIL.SOURCE_NETWORK	String	Read	Contains the IC charge offer search value from the IFW_ICPRODUCT_CNF table.
DESTINATION_NETWORK DETAIL.DESTINATION_NETWORK	String	Read	Contains the destination network search value from the IFW_ICPRODUCT_CNF table.
A_NUMBER DETAIL.A_NUMBER	String	Read	Contains the A number search value from the IFW_ICPRODUCT_CNF table.
B_NUMBER DETAIL.B_NUMBER	String	Read	Contains the B number search value from the IFW_ICPRODUCT_CNF table.
C_NUMBER DETAIL.C_NUMBER	String	Read	Contains the C number search value from the IFW_ICPRODUCT_CNF table.
RECORD_TYPE DETAIL.RECORD_TYPE	String	Read	Contains the record type search value from the IFW_ICPRODUCT_CNF table.
INTERN_SERVICE_CODE DETAIL.INTERN_SERVICE_CODE	String	Read	Contains the internal service code search value from the IFW_ICPRODUCT_CNF table.
INTERN_SERVICE_CLASS DETAIL.INTERN_SERVICE_CLASS	String	Read	Contains the internal service class search value from the IFW_ICPRODUCT_CNF table.

Table 3-32 (Cont.) FCT_CarrierIcRating EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
INTERN_USAGE_CLASS DETAIL.INTERN_USAGE_CLASS	String	Read	Contains the internal usage class search value from the IFW_ICPRODUCT_CNF table.
BDR_CHARGING_START_TIMESTAMP DETAIL.CHARGING_START_TIMESTAMP	Date	Read	Contains the charging time stamp.
INTERN_A_NUMBER_ZONE DETAIL.INTERN_A_NUMBER_ZONE	String	Read	Contains the internal A number zone. This value sets the charge packet INTERN_ORIGIN_NUM_ZONE value.
INTERN_B_NUMBER_ZONE DETAIL.INTERN_B_NUMBER_ZONE	String	Read	Contains the internal B number zone. This value sets the charge packet INTERN_DESTIN_NUM_ZONE value.
ASS_CBD_RECORD_TYPE DETAIL.ASS_CBD.RECORD_TYPE	String	Write	Contains the record type: • **990** = CarrierIC • **991** = Roaming
INTERN_CALC_MODE DETAIL.ASS_CBD.INTERN_CALC_MODE	String	Write	Contains the calculation mode from the CALCMODE field in the IFW_NETWORKMODEL database table.
CHARGE_TYPE DETAIL.ASS_CBD.CP.CHARGE_TYPE	String	Write	Contains the charge type.
NETWORK_OPERATOR DETAIL.ASS_CBD.CP.NETWORK_OPERATOR_CODE	String	Write	Contains the network operator code from the CONNECTED_NO field in the IFW_TRUNK database table.
NETWORK_OPERATOR_BILLINGTYPE DETAIL.ASS_CBD.CP.NETWORK_OPERATOR_BILLINGTYPE	String	Write	Contains the billing type from the BILL_DIRECTION field in the IFW_ICPRODUCT_RATE database table.
PRODUCTCODE_USED DETAIL.ASS_CBD.CP.PRODUCTCODE_USED	String	Write	Contains the IC charge offer code from the ICPRODUCT field in the IFW_ICPRODUCT database table.
TRUNK_USED DETAIL.ASS_CBD.CP.TRUNK_USED	String	Write	Contains the trunk from the TRUNK field in the IFW_TRUNK database table.
POI_USED DETAIL.ASS_CBD.CP.POI_USED	String	Write	Contains the POI from the POI field in the IFW_POI database table.
ASS_CBD_CHARGING_START_TIMESTAMP DETAIL.ASS_CBD.CP.CHARGING_START_TIMESTAMP	Date	Write	Contains the charging time stamp.
INTERN_FIX_COST DETAIL.ASS_CBD.CP.INTERN_FIX_COST	Decimal	Write	Contains the internal fixed cost. Added to the charge by the FCT_MainRating module.

Table 3-32 (Cont.) FCT_CarrierIcRating EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
INTERN_RATEPLAN DETAIL.ASS_CBD.CP.INTERN_RATEPLAN	String	Write	Contains the charge. Used by the FCT_PreRating and FCT_MainRating modules.
INTERN_ORIGIN_NUM_ZONE DETAIL.ASS_CBD.CP.INTERN_ORIGIN_NUM_ZONE	String	Write	Contains the A number zone. Used by the FCT_PreRating module to find the impact category.
INTERN_DESTIN_NUM_ZONE DETAIL.ASS_CBD.CP.INTERN_DESTIN_NUM_ZONE	String	Write	Contains the B number zone. Used by the FCT_PreRating module to find the impact category.
INTERN_BILLING_CURRENCY DETAIL.ASS_CBD.CP.INTERN_BILLING_CURRENCY	String	Write	Contains the billing currency name
INTERN_HOME_CURRENCY DETAIL.ASS_CBD.CP.INTERN_HOME_CURRENCY	String	Write	Contains the home currency name

FCT_CiberOcc

The FCT_CiberOcc module creates a CIBER record for other charges and credits (OCC record), type 50 or 52.

Dependencies

This module requires a connection to the DAT_InterConnect module.

Must run after the FCT_DuplicateCheck module and before the FCT_CarrierIcRating module.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-33 lists the FCT_CiberOcc registry entries.

Table 3-33 FCT_CiberOcc Registry Entries

Entry	Description	Mandatory	Default
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive You can use this entry in a semaphore file.	Yes	N/A

Table 3-33 (Cont.) FCT_CiberOcc Registry Entries

Entry	Description	Mandatory	Default
CallRecordTypeField	Specifies the EDR field that indicates the CIBER record type. When processing CIBER record types, this entry is used to find the IC Charge Offers and the corresponding charge to use for rating the CIBER records.	No	DETAIL.ASS_CIBER_EXT.CIBER_RECORD_TYPE
EdrNetworkModel	Specifies the network model to use for CIBER_OCC searching. This identifies the home network You can specify one home network per pipeline. You can change this value by using a semaphore.	Yes	N/A
InterConnectDataModule	Specifies a connection to the DAT_InterConnect module.	Yes	N/A
NoOCCField	Specifies the field that indicates whether to generate an OCC record. This field must match the field name specified in the **DuplicateIndicatorField** entry of the FCT_DuplicateCheck registry. See "FCT_DuplicateCheck".	Yes	DETAIL.ASS_CIBER_EXT.NO_OCC
OCCDescription	Description of the service associated with the OCC. **Important:** This field must not contain spaces. If you require spaces in the description, write an iScript to populate this field.	No	" " (Empty string)
OCCIntervalIndicator	Specifies the interval at which the associated OCC record is generated.	No	**3**

Sample Registry

```
CiberOcc
{
   ModuleName = FCT_CiberOcc
   Module
   }
      Active = TRUE
      InterConnectDataModule = ifw.DataPool.InterConnect
      EdrNetworkModel = ROAMING
      CallRecordTypeField = DETAIL.ASS_CIBER_EXT.CIBER_RECORD_TYPE
      NoOCCField = DETAIL.ASS_CIBER_EXT.NO_OCC
      OCCIntervalIndicator = 3
      OCCDescription = DAILY_SURCHARGE
    }
}
```

Semaphore File Entries

Table 3-34 lists the FCT_CiberOcc Semaphore file entries.

Table 3-34 FCT_CiberOcc Semaphore File Entries

Entry	Description
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive
EdrNetworkModel	The network model to use for CIBER_OCC searching.

Sample Semaphore File Entry

```
ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.
CiberOcc.Module.Active = false

ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.
CiberOcc.Module.EdrNetworkModel = ROAMING
```

EDR Container Fields

The FCT_CiberOcc module uses the EDR container fields listed in Table 3-35:

Table 3-35 FCT_CiberOcc EDR Container Fields

Default Field Name	Type	Access	Description
DETAIL.ASS_CIBER_EXT.AIR_CONNECT_TIME	Date	Read	Used to specify the connection time for the OCC record.
DETAIL.CHARGING_START_TIMESTAMP	Date	Read	IFW_CIBER_OCC
DETAIL.ASS_CIBER_EXT.CHARGE_NO_1_CONNECT_TIME	Date	Read	Used to specify the connection time for the OCC record.
DETAIL.ASS_CIBER_EXT.CIBER_RECORD_TYPE	String	Read	This field is specified in the **CallRecordTypeField** entry in the registry. This is the default field used to determine the current EDR call record type.
DETAIL.ASS_CIBER_EXT.CIBER_RECORD_TYPE	String	Write	Specifies the type of record to create. If the current record type is: • 22 or 32, assign **52** to this field. • 10, 20, or 30, assign **50** to this field.
DETAIL.ASS_CIBER_EXT.CONNECT_TIME	Date	Write	This field is set to one of the following: • **AIR_CONNECT_TIME** if the call record type is 22. • **SSU_CONNECT_TIME** if the call record type is 10 or 20. • **CHARGE_NO_1_CONNECT_TIME** if the call record type is 30 or 32.

Table 3-35 (Cont.) FCT_CiberOcc EDR Container Fields

Default Field Name	Type	Access	Description
DETAIL.ASS_CIBER_EXT.NO_OCC	String	Read	This field is specified in the **NoOCCField** entry in the registry.
DETAIL.ASS_CIBER_EXT.OCC_DESCRIPTION	String	Write	The value of this field is specified in the **OCCDescription** entry in the registry. The default value is an empty string: " "
DETAIL.ASS_CIBER_EXT.OCC_END_DATE	Date	Write	The value of **CHARGING_START_TIMESTAMP** in the current EDR.
DETAIL.ASS_CIBER_EXT.OCC_INTERVAL_INDICATOR	String	Write	The value of this field is specified in the **OCCIntervalIndicator** entry in the registry. The default value is **3** (daily interval).
DETAIL.ASS_CIBER_EXT.OCC_START_DATE	Date	Write	The value of **CHARGING_START_TIMESTAMP** in the current EDR.
DETAIL.ASS_CIBER_EXT.RECORD_CREATE_DATE	Date	Write	This field is set to the system date.
DETAIL.ASS_CIBER_EXT.RECORD_USE_INDICATOR	String	Write	This field is set to **1**.
DETAIL.ASS_CIBER_EXT.SEQ_INDICATOR	String	Write	This field is set to **01**.
DETAIL.SOURCE_NETWORK	String	Read	Search value used for searching the IFW_CIBER_OCC database table.
DETAIL.ASS_CIBER_EXT.SSU_CONNECT_TIME	Date	Read	Used to specify the connection time for the OCC record.

Database Interface for the FCT_CiberOcc Module

The FCT_CiberOcc module uses the IFW_CIBER_OCC database table to determine whether OCC records are generated for the network operator.

FCT_CliMapping

The FCT_CliMapping module maps multiple numbers to a single number for billing. See "Mapping Multiple Phone Numbers to a Single Number" in *BRM Setting Up Pipeline Pricing*.

Dependencies

Must run before the rating modules.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-36 lists the FCT_CliMapping registry entries.

Table 3-36 FCT_CliMapping Registry Entries

Entry	Description	Mandatory
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive You can use this entry in a semaphore file.	Yes
MapFile	Specifies the path to the mapping file. You can use this entry in a semaphore file.	Yes

Sample Registry Entry

```
CliMapping
{
  ModuleName = FCT_CliMapping
  Module
   {
    Active = True
    MapFile = cli_map_1.dat
  }
}
```

Semaphore File Entries

Table 3-37 lists the FCT_CliMapping Semaphore file entries.

Table 3-37 FCT_CliMapping Semaphore File Entry

Entry	Description
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive

Sample Semaphore File Entry

```
ifw.Pipelines.ALL_RATE.Functions.Processing.FucntionPool.CliMapping.Module.Active = True
```

EDR Container Fields

Table 3-38 lists the FCT_CliMapping EDR container fields.

Table 3-38 FCT_CliMapping EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
A_NUMBER DETAIL.A_NUMBER	String	Read/Write	Contains the customer A number.
CUST_A_ACCOUNT_ID DETAIL.CUST_A.ACCOUNT_ID	Block	Read	Contains the customer account ID.

FCT_CreditLimitCheck

The FCT_CreditLimitCheck module determines whether event owners have enough balances in their account balance to cover the cost of usage. If the account does not have sufficient balances to authorize the entire request, this module determines how much usage can be authorized with the available balances.

> **Note:**
>
> The FCT_CreditLimitCheck module does not check the credit floor.

Dependencies

Use this module in a real-time discounting pipeline.

This module must run after all other discounting modules.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-39 lists the FCT_CreditLimitCheck registry entries.

Table 3-39 FCT_CreditLimitCheck Registry Entries

Entry	Description	Mandatory
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive	Yes
CLCTrace	Specifies whether to generate a credit limit check trace file. **True** = Generate a trace file. **False** = Do not generate a trace file (Default)	No
CurrencyDataModule	Specifies the connection to the DAT_Currency module.	Yes
RoundUpRequestQuantity	Determines whether authorized quantities are rounded up to remove fractional values. **True** = Round up **False** = No rounding (Default)	No
StepValue	Specifies the step value to be considered for the quantity during reverse rating and for rounding the prorated quantity.	No

Sample Registry Entry

```
#------------------------------------------------
# Credit Limit Check
#------------------------------------------------
CreditLimitCheckModule
{
```

```
  ModuleName = FCT_CreditLimitCheck
  Module
  {
    Active  = True
    RoundUpRequestQuantity = True
    CLCTrace = True
    CurrencyDataModule = ifw.DataPool.CurrencyDataModule
    StepValue = 0.1
  }
}
```

EDR Container Fields

The FCT_CreditLimitCheck module uses the EDR container fields listed in Table 3-40:

Table 3-40 FCT_CreditLimitCheck EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
CREDIT_LIMIT_CHECK DETAIL.CREDIT_LIMIT_CHECK	Integer	Read	Specifies whether to perform a credit limit check on the EDR: 1 = check; 0 = don't check.
CREDIT_LIMIT_CHECK_RESULT DETAIL.CREDIT_LIMIT_CHECK_RESULT	Integer	Write	Specifies whether the credit limit check passed or failed: 1 = passed; 0 = failed.
INTERN_BALANCE_GROUP_ID DETAIL.INTERN_BALANCE_GROUP_ID	String	Read	Account level balance group of the event owner account.
DETAIL.UNRATED_QUANTITY	Decimal	Write	The quantity that could not be rated.
RESOURCE_ID DETAIL.ASS_CBD.CP.RESOURCE_ID	Integer	Read	Balance Element ID for the charge packet.
CHARGED_AMOUNT_VALUE DETAIL.ASS_CBD.CP.CHARGED_AMOUNT_VALUE	Decimal	Read	The balance impact of the charge packet. This amount was computed by real-time rating.
QUANTITY_FROM DETAIL.ASS_CBD.CP.QUANTITY_FROM	Decimal	Read	Charge packet start quantity. If the charge packet is split by FCT_Discount, this module reads QUANTITY_FROM values from the DETAIL.ASS_CBD.CP.SPLIT_CP block.

Table 3-40 (Cont.) FCT_CreditLimitCheck EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
QUANTITY_TO DETAIL.ASS_CBD.CP.QUANTITY_TO	Decimal	Read	Charge packet end quantity. If the charge packet is split by FCT_Discount, this module reads QUANTITY_TO values from the DETAIL.ASS_CBD.CP.SPLIT_CP block.
DETAIL.ASS_CBD.CP.ROUNDED_QTY_VALUE	Decimal	Read	The quantity that could be authorized.
DP_DISCOUNT_BALANCE_GROUP_ID DETAIL.ASS_CBD.DP.BALANCE_GROUP_ID	Integer	Read	POID of the balance group impacted by this discount packet.
DP_DISCOUNT_GRANTED_AMOUNT DETAIL.ASS_CBD.DP.GRANTED_AMOUNT	Decimal	Read	Total amount of the discount packet. This discount amount is applied to the balance group.
DP_QUANTITY_FROM DETAIL.ASS_CBD.DP.QUANTITY_FROM	Decimal	Write	Discount packet start quantity. Aligns with the QUANTITY_FROM value in a charge packet or a split charge packet.
DP_QUANTITY_TO DETAIL.ASS_CBD.DP.QUANTITY_TO	Decimal	Write	Discount packet end quantity. Aligns with the QUANTITY_TO value in a charge packet or a split charge packet.
DP_DISCOUNT_RESOURCE_ID DETAIL.ASS_CBD.DP.RESOURCE_ID	Integer	Read	Balance Element ID for the discount packet.
BG_BG_ID DETAIL.CUST_A.BG.BALANCE_GROUP_ID	String	Read	Numeric ID for the balance group.
BG_BELEM_RESOURCE_ID DETAIL.CUST_A.BG.BAL_ELEM.RESOURCE_ID	Integer	Read	The balance element ID for the balance group element.
BG_BELEM_CURR_BAL DETAIL.CUST_A.BG.BAL_ELEM.CURR_BAL	Decimal	Read	The event owner's current balance for this balance group element.
BG_BELEM_CREDIT_LIMIT DETAIL.CUST_A.BG.BAL_ELEM.CREDIT_LIMIT	Decimal	Read	The credit limit for this balance group element.
BG_BELEM_RESERVED_AMOUNT DETAIL.CUST_A.BG.BAL_ELEM.RESERVED_AMOUNT	Decimal	Read	The amount already in reserve by the event owner.
DETAIL.RUM_MAP.RUM_NAME	String	Read	Name of the RUM. Used in multi-RUM checks.
DETAIL.RUM_MAP.NET_QUANTITY	Decimal	Write	The total requested quantity for a RUM. Used in multi-RUM checks.

Table 3-40 (Cont.) FCT_CreditLimitCheck EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
DETAIL.RUM_MAP.UNRATED_QUANTITY	Decimal	Write	The quantity of a RUM that could not be authorized. Used in multi-RUM checks.

FCT_CustomerRating

The FCT_CustomerRating module supplies charges for the FCT_MainRating module.

See the following documents:

- "About Customer Rating" in *BRM Setting Up Pipeline Rating and Discounting*
- FCT_MainRating

The FCT_CustomerRating module is also used for least-cost rating and promotional overlays. See:

- "About Least Cost Rating" in *BRM Setting Up Pipeline Rating and Discounting*
- "About Calculating the Promotional Savings" in *BRM Setting Up Pipeline Rating and Discounting*

Dependencies

Requires a connection to the Pipeline Manager database.

This module must run after FCT_Account.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-41 lists the FCT_CustomerRating registry entries.

Table 3-41 FCT_CustomerRating Registry Entries

Entry	Description	Mandatory
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive You can use this entry in a semaphore file.	Yes
DataConnection	Specifies the connection to the Pipeline Manager database.	Yes
DefaultRateplan	Specifies the charge code used as default if no customer data for the A number can be found. You can use this entry in a semaphore file. See "Assigning a Default Charge and Default Segment for Customer Rating" in *BRM Setting Up Pipeline Rating and Discounting*.	No

Table 3-41 (Cont.) FCT_CustomerRating Registry Entries

Entry	Description	Mandatory
DefaultSegment	Specifies the segment name used as default if no customer data for the A number can be found. You can use this entry in a semaphore file. See "Assigning a Default Charge and Default Segment for Customer Rating" in *BRM Setting Up Pipeline Rating and Discounting*.	No
LeastCostRating	Specifies whether least cost rating is active or inactive. For more information, see "About Least Cost Rating" in *BRM Setting Up Pipeline Rating and Discounting*. • **True** activates least cost rating. • **False** disables least cost rating. See "Configuring Least Cost Rating" in *BRM Setting Up Pipeline Rating and Discounting*.	No
Mode	Specifies that the module is run for customer rating (**CUSTOMER**). See "About Customer Rating" in *BRM Setting Up Pipeline Rating and Discounting*.	Yes
PromotionalSaving	Specifies whether to calculate the total savings to customers when rating a usage event with a promotional charge offer rather than a base charge offer. For more information, see "About Calculating the Promotional Savings" in *BRM Setting Up Pipeline Rating and Discounting*. • **True** specifies to calculate the savings amount. • **False** specifies to not calculate the savings amount. See "About Calculating the Promotional Savings" in *BRM Setting Up Pipeline Rating and Discounting*.	No

Sample Registry

```
CustomerRating
{
  ModuleName = FCT_CustomerRating
  Module
  {
    Active = True
    Mode = CUSTOMER
    DataConnection = ifw.DataPool.Database
  }
}
```

Semaphore File Entries

Table 3-42 lists the FCT_CustomerRating Semaphore file entries.

Table 3-42 FCT_CustomerRating Semaphore File Entries

Entry	Description
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive

Table 3-42 (Cont.) FCT_CustomerRating Semaphore File Entries

Entry	Description
DefaultRateplan	Specifies the charge code used as default if no customer data for the A number can be found. See "Assigning a Default Charge and Default Segment for Customer Rating" in *BRM Setting Up Pipeline Rating and Discounting*.
DefaultSegment	Specifies the segment name used as default if no customer data for the A number can be found. See "Assigning a Default Charge and Default Segment for Customer Rating" in *BRM Setting Up Pipeline Rating and Discounting*.

Sample Semaphore File Entry

```
ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.CustomerRating.Module.Active = False
```

EDR Container Fields

Table 3-43 lists the FCT_CustomerRating EDR container fields.

Table 3-43 FCT_CustomerRating EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
ASS_CBD DETAIL.ASS_CBD	Block	Create	The associated charge breakdown record created to hold the rating data.
ASS_CBD_CHARGE_PACKET DETAIL.ASS_CBD_CHARGE_PACKET	Block	Create	The charge packet created to hold the rating data.
BDR_CHARGING_START_TIMESTAMP DETAIL.CHARGING_START_TIMESTAMP	Date	Read	Contains the charging time stamp. Written to the DETAIL.ASS_CBD.CP.CHARGING_START_TIMESTAMP field.
INTERN_A_NUMBER_ZONE DETAIL.INTERN_A_NUMBER_ZONE	String	Read	Contains the A number zone. Written to the DETAIL.ASS_CBD.CP.INTERN_ORIGIN_NUM_ZONE field.
INTERN_B_NUMBER_ZONE DETAIL.INTERN_B_NUMBER_ZONE	String	Read	Contains the B number zone. Written to the DETAIL.ASS_CBD.CP.INTERN_DESTIN_NUM_ZONE field.
INTERN_SLA_RSC_GROUP DETAIL.INTERN_SLA_RSC_GROUP	String	Write	Contains the SLA RSC group.
INTERN_SLA_USC_GROUP DETAIL.INTERN_SLA_USC_GROUP	String	Write	Contains the SLA USC group code.
INTERN_SLA_IRULE_SET DETAIL.INTERN_SLA_IRULE_SET	String	Write	Contains the SLA iRule set.

Table 3-43 (Cont.) FCT_CustomerRating EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
ASS_CBD_RECORD_TYPE DETAIL.ASS_CBD.RECORD_TYPE	String	Write	Contains the record type for the associated charge breakdown record. • 981 = Customer rating
CHARGE_TYPE DETAIL.ASS_CBD.CP.CHARGE_TYPE	String	Write	Contains the charge type. This field is always set to **N**.
ASS_CBD_ACCOUNT_CODE DETAIL.ASS_CBD.ACCOUNT_CODE	String	Write	Contains the account code. Set with the value from the DETAIL.CUST_A.ACCOUNT_NO field.
ASS_CBD_SYSTEM_BRAND_CODE DETAIL.ASS_CBD.SYSTEM_BRAND_CODE	String	Write	Contains the system brand code. Set with the value from the DETAIL.CUST_A.SYSTEM_BRAND field.
ASS_CBD_CUSTOMER_BILLCYCLE DETAIL.ASS_CBD.CUSTOMER_BILLCYCLE	String	Write	Contains the customer's bill cycle code. Set with data from the DETAIL.CUST_A.BILL_CYCLE field.
ASS_CBD_CUSTOMER_CURRENCY DETAIL.ASS_CBD.CUSTOMER_CURRENCY	String	Write	Contains the customer's currency. Set with the value from the DETAIL.CUST_A.CURRENCY field.
ASS_CBD_CUSTOMER_RATEPLAN_CODE DETAIL.ASS_CBD.CUSTOMER_RATEPLAN_CODE	String	Write	A comma-separated list of charge codes for all rating charge offers. The list is arranged by charge offer priority, with highest priority first and lowest priority last.
INTERN_BILLING_CURRENCY DETAIL.ASS_CBD.CP.INTERN_BILLING_CURRENCY	String	Write	Contains the billing currency. Set with the value from the DETAIL.CUST_A.CURRENCY field.
ASS_CBD_SEGMENT_CODE DETAIL.ASS_CBD.SEGMENT_CODE	String	Write	Contains the segment code. Set with the value from the Data Warehouse ERA.
ASS_CBD_SLA_CODE DETAIL.ASS_CBD.SLA_CODE	String	Write	Contains the SLA code. Set with the value from the Service Level Agreement ERA.

Table 3-43 (Cont.) FCT_CustomerRating EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
INTERN_DISCOUNT_ACCOUNT DETAIL.ASS_CBD.CP.INTERN_DISCOUNT_ACCOUNT	String	Write	Contains the discount account. Set with the value from the Discount ERA.
RATEPLAN_CODE DETAIL.ASS_CBD.CP.RATEPLAN_CODE	String	Write	Contains the charge code to use for rating.
ASS_CBD_CHARGING_START_TIMESTAMP DETAIL.ASS_CBD.CP.CHARGING_START_TIMESTAMP	Date	Write	Contains the charging time stamp. Set with the value from the DETAIL.CHARGING_START_TIMESTAMP field.
INTERN_RATEPLAN DETAIL.ASS_CBD.CP.INTERN_RATEPLAN	String	Write	Contains the internal charge code. Set with values from the Data Warehouse ERA.
INTERN_ORIGIN_NUM_ZONE DETAIL.ASS_CBD.CP.INTERN_ORIGIN_NUM_ZONE	String	Write	Contains the zone for the A number. Set with DETAIL.INTERN_A_NUMBER_ZONE
INTERN_DESTIN_NUM_ZONE DETAIL.ASS_CBD.CP.INTERN_DESTIN_NUM_ZONE	String	Write	Contains the zone for the B number. Set with DETAIL.INTERN_B_NUMBER_ZONE
CUST_A_ACCOUNT_ID DETAIL.CUST_A.ACCOUNT_ID	String	Read	Contains the customer account ID. Write to DETAIL.ASS_CBD.ACCOUNT_CODE
CUST_A_ACCOUNT_NO DETAIL.CUST_A.ACCOUNT_NO	String	Read	Contains the customer account number.
CUST_A_SYSTEM_BRAND DETAIL.CUST_A.SYSTEM_BRAND	String	Read	Contains the system brand. Written to the DETAIL.ASS_CBD.SYSTEM_BRAND_CODE field.
CUST_A_BILL_CYCLE DETAIL.CUST_A.BILL_CYCLE	String	Read	Contains the customer's bill cycle. Written to the DETAIL.ASS_CBD.CUSTOMER_BILLCYCLE field.

Table 3-43 (Cont.) FCT_CustomerRating EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
CUST_A_CURRENCY DETAIL.CUST_A.CURRENCY	String	Read	Contains the customer's currency. Written to the DETAIL.ASS_CBD.CUSTOMER_CURRENCY and DETAIL.ASS_CBD.CP.INTERN_BILLING_CURRENCY field.
CUST_A_INTERN_PP_INDEX DETAIL.CUST_A.INTERN_FOUND_PP_INDEX	Integer	Read	Contains an index of the customer's purchased charge offers identified by the FCT_Account module.
CUST_A_RATEPLAN_NAME DETAIL.CUST_A.PRODUCT.RATEPLAN_NAME	String	Read	Contains the charge name. Written to the DETAIL.ASS_CBD.CP.RATEPLAN_CODE field.
CUST_A_PROFILE DETAIL.CUST_A.ERA.PROFILE	String	Read	Contains the customer's account-related ERA data.
CUST_A_KEY DETAIL.CUST_A.ERA.PA.KEY	String	Read	Contains the key for the account-related ERA data.
CUST_A_VALUE DETAIL.CUST_A.ERA.PA.VALUE	String	Read	Contains the value for the account-related ERA data.
CUST_A_PRODUCT_PROFILE DETAIL.CUST_A.PRODUCT.ERA.PROFILE	String	Read	Contains the customer's service-related ERA data.
CUST_A_PRODUCT_KEY DETAIL.CUST_A.PRODUCT.ERA.PA.KEY	String	Read	Contains the key for the service-related ERA data.
CUST_A_PRODUCT_KEY DETAIL.CUST_A.PRODUCT.ERA.PA.KEY	String	Read	Contains the value for the service-related ERA data.
DETAIL.CUST_A.INTERN_RATING_PRODUCTS	String	Read	Contains the charge offer rating indexes. This is a comma-separated list of all rating charge offers' indexes associated with the same service and event, and their priorities.
DETAIL.CUST_A.LEAST_COST	Integer	Write	Indicates whether to use least cost rating for an EDR. **1** turns it off; any other integer turns it on. This entry overrides the least cost rating entry in the registry file.
DETAIL.CUST_A.PRODUCT_PRIORITY	String	Read	Contains a list of the priorities for all charge offers that are associated with the same service and event.

Table 3-43 (Cont.) FCT_CustomerRating EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
DETAIL.CUST_A.PRODUCT.USAGE_START	Date	Read	Contains a list of the start times for all charge offers that are associated with the same service and event.
DETAIL.CUST_A.PROMOTIONAL_SAVING	Integer	Write	Indicates whether to calculate the promotional savings for an EDR. **1** turns off promotional savings; any other integer turns it on.
DETAIL.ASS_CBD.CP.RATEPLAN_CODE	String	Write	A comma-separated list of charge codes for all rating charge offers. The list is arranged by charge offer priority, with highest priority first and lowest priority last.

FCT_Dayrate

The FCT_Dayrate module calculates charges for special day rates, for example, a discount for calls made on January 1.

Dependencies

Requires a connection to the DAT_Dayrate module.

This module must run after the FCT_MainRating module to adjust the rate.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-44 lists the FCT_Dayrate registry entries.

Table 3-44 FCT_Dayrate Registry Entries

Entry	Description	Mandatory
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive You can use this entry in a semaphore file.	No
DayrateDataModule	Specifies the connection to the DAT_Dayrate module.	No

Sample Registry

```
Dayrate
{
  ModuleName = FCT_Dayrate
  Module
  {
    Active = True
    DayrateDataModule = DayrateData
  }
}
```

Semaphore File Entries

Table 3-45 lists the FCT_Dayrate Semaphore file entry.

Table 3-45 FCT_Dayrate Semaphore File Entry

Entry	Description
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive

Sample Semaphore File Entry

```
ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.SpecialDayRate.Module.Active = False
```

EDR Container Fields

Table 3-46 lists the FCT_Dayrate EDR container fields.

Table 3-46 FCT_Dayrate EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
CHARGING_START_TIMESTAMP DETAIL.CHARGING_START_TIMESTAMP	Date	Read	Contains the charging time stamp. The field is used to determine which discount to use.
ASS_CBD.CP.INTERN_RATEPLAN DETAIL.ASS_CBD.CP.INTERN_RATEPLAN	String	Read	Contains the charge code. This field is used to determine which discount to use.
ASS_CBD.CP.INTERN_RATEPLAN_VERSION DETAIL.ASS_CBD.CP.INTERN_RATEPLAN_VERSION	Block	Read	Contains the charge version. This field is used to determine which discount to use.
ASS_CBD.CP.CHARGED_AMOUNT_VALUE DETAIL.ASS_CBD.CP.CHARGED_AMOUNT_VALUE	Block	Read/Write	Contains the recalculated charged amount value.
ASS_CBD.CP DETAIL.ASS_CBD.CP	Block	Read	Contains the block index to charge packages.

FCT_Discard

The FCT_Discard module discards or skips EDRs based on configurable EDR properties.

- Skipping an EDR removes it from the pipeline.
- Discarding an EDR sends it to a different output stream.

In both the cases the state of the EDR becomes invalid.

See "Discarding and Skipping EDRs" in *BRM Setting Up Pipeline Rating and Discounting*.

Dependencies

Requires a connection to the Pipeline Manager database.

Because you can discard or split EDRs based on service codes, this module should run after the FCT_ServiceCodeMap module. Should be early in the function pool, but must be run after FCT_CallAssembling.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-47 lists the FCT_Discard registry entries.

Table 3-47 FCT_Discard Registry Entries

Entry	Description	Mandatory
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive You can use this entry in a semaphore file.	Yes
DataConnection	Specifies the connection to the Pipeline Manager database.	Yes
Function	Specifies whether to discard or skip the EDR. Default = **Discard** You can use this entry in a semaphore file.	No
StreamName	Specifies the output stream for discarded EDRs. You can use this entry in a semaphore file. Default = **DevNull** See "Configuring Output of Discarded EDRs" in *BRM Setting Up Pipeline Rating and Discounting*.	No

Sample Registry

```
Discard
{
   ModuleName = FCT_Discard
   Module
   {
      Active = True
```

```
            DataConnection = ifw.DataPool.Database
            Function = Discard
            StreamName = DevNull
        }
    }
```

Semaphore File Entries

Table 3-48 lists the FCT_Discard Semaphore file entries.

Table 3-48 FCT_Discard Semaphore File Entries

Entry	Description
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive
Function	Specifies whether to discard or skip the EDR. Default = **Discard**
Reload	Reloads the discard rules.

Sample Semaphore File Entry

```
ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.EventDiscarding.Module.Reload {}
```

EDR Container Fields

Table 3-49 lists the FCT_Discard EDR container fields.

Table 3-49 FCT_Discard EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
RECORD_TYPE DETAIL.RECORD_TYPE	String	Read	Contains the record type.
SOURCE_NETWORK DETAIL.SOURCE_NETWORK	String	Read	Contains the source network code. This could either be PLMN ID or any logical operator code.
DESTINATION_NETWORK DETAIL.DESTINATION_NETWORK	String	Read	Contains the destination network code.
CALL_COMPLETION_INDICATOR DETAIL.CALL_COMPLETION_INDICATOR	String	Read	Indicates if a call was successfully completed.
LONG_DURATION_INDICATOR DETAIL.LONG_DURATION_INDICATOR	String	Read	Contains the long duration indicator: • **F** = First • **I** = Intermediate • **L** = Last
USAGE_CLASS DETAIL.USAGE_CLASS	String	Read	Contains the external usage class.

Table 3-49 (Cont.) FCT_Discard EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
INTERN_SERVICE_CODE DETAIL.INTERN_SERVICE_CODE	String	Read	Contains the internal service code.
ASS_GSMW_EXT.ORIGINATING_SWITCH_IDENTIFICATION DETAIL.ASS_GPRS_EXT.ORIGINATING_SWITCH_IDENTIFICATION	String	Read	Contains the GSM MSC or Switch ID handling the origin of the call.
ASS_GPRS_EXT_ORIGINATING_SWITCH_IDENTIFICATION DETAIL.ASS_GPRS_EXT.ORIGINATING_SWITCH_IDENTIFICATION	String	Read	Contains the GPRS MSC or Switch ID handling the origin of the call.
TARIFF_CLASS DETAIL.TARIFF_CLASS	String	Read	Contains the tariff class.
TARIFF_SUB_CLASS DETAIL.TARIFF_SUB_CLASS	String	Read	Contains the tariff subclass.
CONNECT_SUB_TYPE DETAIL.CONNECT_SUB_TYPE	String	Read	Contains the connection subtype.
B_NUMBER DETAIL.B_NUMBER	String	Read	Contains the B number.
CHARGING_START_TIMESTAMP DETAIL.CHARGING_START_TIMESTAMP	Date	Read	Contains the charging time stamp.
WHOLESALE_CHARGED_AMOUNT_VALUE DETAIL.WHOLESALE_CHARGED_AMOUNT_VALUE	Decimal	Read	Contains the sum of the wholesale charged amount value.
DISCARDING DETAIL.DISCARDING	Integer	Write	Indicates if the EDR should be discarded.
DETAIL.DISCARD_REASON	String	Write	Specifies the name of the discarding rule that applies to the EDR. Discarding rules are defined in the ifw_discarding table. If any of the rules in the table applies to the EDR, the EDR is discarded or skipped.

Database Tables

The FCT_Discard module uses the data in the IFW_Discarding table to determine which EDRs should be discarded. See "Discarding and Skipping EDRs" in *BRM Setting Up Pipeline Rating and Discounting*.

> **Note:**
>
> For information on compare patterns used in database values, see "About Using Regular Expressions when Specifying the Data to Extract" in *BRM Setting Up Pipeline Rating and Discounting*.

FCT_Discount

The FCT_Discount module performs discounting operations. The module supports discounting for events rated in real time and events rated in batch by Pipeline Manager.

When the module is called for discount calculation as part of prepaid authorization or reauthorization, it splits charge packets if necessary to create linear segments to which a single net rate applies.

Dependencies

Requires a connection to the following:

- The DAT_Discount module. See "DAT_Discount".
- A balance data module, either DAT_BalanceRealtime or DAT_BalanceBatch depending on whether the module is being used for real-time or batch discounting. See "DAT_BalanceRealtime" or "DAT_BalanceBatch".
- An account data module, either DAT_AccountRealtime or DAT_AccountBatch depending on whether the module is being used for real-time or batch discounting. See "DAT_AccountRealtime" or "DAT_AccountBatch".
- The DAT_Currency module. This module converts balance element codes to balance element IDs. See "DAT_Currency".

For batch discounting, you must also configure the FCT_ApplyBalance module, which should be in the same function pool as this module for performance reasons and must run after this module.

This module must run after FCT_MainRating.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-50 lists the FCT_Discount registry entries.

Table 3-50 FCT_Discount Registry Entries

Entry	Description	Mandatory
AccountDataModule	Specifies the connection to the DAT_AccountRealtime or DAT_AccountBatch module.	Yes

Table 3-50 (Cont.) FCT_Discount Registry Entries

Entry	Description	Mandatory
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive You can use this entry as a semaphore command.	Yes
AvoidMatchFactorCalculation	Specifies whether to calculate the amount of usage that is discounted (the match factor) for parallel and sequential discounts. The match factor is typically used only for cascading discounts. See "Calculating the Match Factor of Parallel and Sequential Discounts" in *BRM Setting Up Pipeline Rating and Discounting*.	No
BackOut	Specifies if the module is used in a back-out pipeline for rerating. Default = **False**	No
BalanceDataModule	Specifies the connection to the DAT_BalanceRealtime or DAT_BalanceBatch module.	Yes
CurrencyDataModule	Specifies the connection to DAT_Currency module.	Yes
DiscountDataModule	Specifies the connection to the DAT_Discount module.	Yes
DiscountMoreThanPossible	Specifies if a discount can be higher than the real revenue. Default = **False**	No
DiscountTrace	Specifies whether to generate discount trace file. • **True** indicates that a discount trace file is generated. • **False** indicates that a discount trace file is not generated. Default = **False**	No
EvalMultipleBalImpact	Specifies to create a discount balance impact for each charge packet when the discount base expression is not a simple expression (StepC, StepQ, TotalC, or TotalQ). Default = **False**	No
IgnoreEDROnDeadlock	Specifies whether to ignore EDRs causing the deadlock. • **True** indicates that the module should ignore the EDRs and continue processing the EDR file. The module places the EDR causing the deadlock into the discountError directory. • **False** indicates that the module should roll back already processed EDRs and start reprocessing the same file.	No
IgnoreEDROnDiscountError	Specifies whether to ignore EDRs with discounting error. • **True** indicates that the module should ignore the EDRs with discounting error and continue processing the remaining EDRs in the EDR file. The module places the EDRs that failed discounting into the discountError directory. • **False** indicates that the module should roll back the EDRs with discounting error that has already been processed and reprocess them. Default = **False**	No

Table 3-50 (Cont.) FCT_Discount Registry Entries

Entry	Description	Mandatory
IgnoreEDROnLock	Specifies whether to ignore the EDR if the balance group object is locked by another transaction. The ignored or rejected EDRs with the locked balance group are placed into the **discountError** directory. • **True** indicates that the module ignores the EDRs and continues processing the EDR file. • **False** indicates that the module rolls back already processed EDRs and begins reprocessing the same file. Default = **False**	No
ShowZeroDiscount	Specifies whether Discount Packet are generated when granted discount amount is 0. • **True** indicates that Discount Packets are generated when the granted discount amount is 0. • **False** indicates that Discount Packets are not generated when the granted discount amount is 0. Default = **False**	No
SupportBundleERA	Specifies if the support for ERAs is needed. Default = **False**	No
TaxationMode	Used only when FCT_Discount is configured for discounting in the real-time pipeline. Specifies when events are taxed. The value of this entry should be the same as the value of **taxation_switch** entry in the Connection Managers (CM) configuration file. Possible values are: **0** - Taxation is disabled. **1** - Enable real-time tax calculation. **2** - Enable deferred (cycle-time) tax calculation. **3** - Enable real-time and deferred tax calculation. Default = **3**	No
ZeroValuePacketFilterDisabled	Used with the aggregation scenario to filter out charge packets where either charge or quantity is zero. Default = **True**	No
ProrateFixedDiscount	Specifies whether to prorate fixed cycle-event discounts: • **False** indicates that fixed cycle-event discounts are not prorated. This is the default. • **True** indicates that fixed cycle-event discounts are prorated. To prorate fixed cycle-event discounts, you must set this entry to **True** before starting the real-time pipeline.	No

Sample Registry

```
GeneralDiscounting
{
  ModuleName = FCT_Discount
  Module
  {
    Active                    = TRUE
    DiscountDataModule        = ifw.DataPool.DiscountModelDataModule
    BalanceDataModule         = ifw.DataPool.BalanceDataModule
    AccountDataModule         = ifw.DataPool.CustomerData
```

```
    CurrencyDataModule          = ifw.DataPool.CurrencyDataModule
    DiscountMoreThanPossible    = False
    ProrateFixedDiscount        = True
    TaxationMode                = 3
    EvalMultipleBalImpact       = True
  }
}
```

Semaphore File Entries

Table 3-51 lists the FCT_Discount Semaphore file entries.

Table 3-51 FCT_Discount Semaphore File Entries

Entry	Description
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive
IgnoreEDROnLock	Specifies whether to ignore the EDR if the balance group object is locked by another transaction. The ignored or rejected EDRs with the locked balance group are placed into the **discountError** directory. • **True** indicates that the module ignores the EDRs and continues processing the EDR file. • **False** indicates that the module rolls back already processed EDRs and begins reprocessing the same file. Default = **False**

Sample Semaphore File Entry

```
ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.ApolloDiscountModule.
Module.Active = False
```

EDR Container Fields

Table 3-52 lists the FCT_Discount EDR container fields.

Table 3-52 FCT_Discount EDR Container Fields

Entry	Format	Access	Description
BDR_CHARGING_END_TIMESTAMP DETAIL.CHARGING_END_TIMESTAMP	Date	Read	Used to calculate the duration.
BDR_CHARGING_START_TIMESTAMP DETAIL.CHARGING_START_TIMESTAMP	Date	Read	Used to calculate the duration.

Table 3-52 (Cont.) FCT_Discount EDR Container Fields

Entry	Format	Access	Description
CREDIT_LIMIT_CHECK DETAIL.CREDIT_LIMIT_CHECK	Integer	Read	Determines whether the module applies discounts normally or as part of a credit limit check. If set to **1**, the module applies discounts as part of a credit limit check. It set to **0** (or any value other that **1**), the module operates normally.
ERROR_REJECT_TYPE DETAIL.ERROR_REJECT_TYPE	String	Read	Used by FCT_Reject to reject the DETAIL to a stream other than the standard reject stream.
EVENT_TYPE DETAIL.EVENT_TYPE	String	Read	BRM event type.
EVENT_BALANCE_GROUP_ID DETAIL.INTERN_BALANCE_GROUP_ID	String	Read	POID of the balance group charged.
INTERN_PROCESS_STATUS DETAIL.INTERN_PROCESS_STATUS	Integer	Read	Process status. If set to **2**, a recycle test is in progress, and this container is skipped.
INTERN_USAGE_CLASS DETAIL.INTERN_USAGE_CLASS	Date	Read	Usage class. This is used for matching the usage class in the discount detail.
REFRESH_BALANCE DETAIL.REFRESH_BALANCE	Integer	Read	Specifies whether the latest balance information should be retrieved from the database. When this field is set, this module calls the balance module to get the latest balance information from the database, whether or not a balance packet is present in the EDR.
USAGE_TYPE DETAIL.USAGE_TYPE	Date	Read	Usage type. Used for matching the usage type in the discount detail.
BDR_UTC_TIME_OFFSET DETAIL.UTC_TIME_OFFSET	String	Read	UTC time offset, if the discount owner is not the A customer.
FU_DISCOUNT_OBJECTS DETAIL.ASS_CBD.FU_DISCOUNT_OBJECTS	String	Write	ID of the account's discounts that have first-usage start times which were used to discount the event.
ASS_CBD_RECORD_TYPE DETAIL.ASS_CBD.RECORD_TYPE	String	Read	Record type. Used for matching the record type in the discount detail.

Table 3-52 (Cont.) FCT_Discount EDR Container Fields

Entry	Format	Access	Description
CHARGED_CURRENCY DETAIL.ASS_CBD.CP.CHARGED_AMOUNT_CURRENCY	String	Read	Currency of the charge.
CHARGED_AMOUNT_VALUE DETAIL.ASS_CBD.CP.CHARGED_AMOUNT_VALUE	Decimal	Read	Amount of the charge. Used as the base value for discounting.
DISCOUNTMODEL_CODE DETAIL.ASS_CBD.CP.DISCOUNTMODEL_CODE	String	Read	Discount. Used for matching the discount in the discount detail.
ASS_CBD_IMPACT_CATEGORY DETAIL.ASS_CBD.CP.IMPACT_CATEGORY	String	Read	Impact category Used for matching the impact category in the discount detail.
CP_DISCOUNT_PACKET_INDEX DETAIL.ASS_CBD.CP.INTERN_PACKET_INDEX	Integer	Read	Packet ID.
PRICEMODEL_CODE DETAIL.ASS_CBD.CP.PRICEMODEL_CODE	String	Read	Pricing code. Used for matching the pricing in the discount detail.
QUANTITY_FROM DETAIL.ASS_CBD.CP.QUANTITY_FROM	Decimal	Read	Charge packet start quantity. Used to determine whether to split charge packets.
QUANTITY_TO DETAIL.ASS_CBD.CP.QUANTITY_TO	Decimal	Read	Charge packet end quantity. Used to determine whether to split charge packets.
RATEPLAN_CODE DETAIL.ASS_CBD.CP.RATEPLAN_CODE	String	Read	Charge code. Used for filtering in the discount detail.
RATETAG_CODE DETAIL.ASS_CBD.CP.RATETAG_CODE	String	Read	Concatenation of charge definition. Used for filtering in the discount detail.
RESOURCE DETAIL.ASS_CBD.CP.RESOURCE	String	Read	Balance Element code. Used for filtering in the discount detail.
RESOURCE_ID DETAIL.ASS_CBD.CP.RESOURCE_ID	Integer	Read	Numeric ID of the balance element. Used for filtering in the discount detail.
ROUNDED_QUANTITY_FROM DETAIL.ASS_CBD.CP.ROUNDED_QUANTITY_FROM	Decimal	Read	From quantity after rounding.
ROUNDED_QUANTITY_TO DETAIL.ASS_CBD.CP.ROUNDED_QUANTITY_TO	Decimal	Read	To quantity after rounding.
ROUNDED_QUANTITY_VALUE DETAIL.ASS_CBD.CP.ROUNDED_QUANTITY_VALUE	Decimal	Read	Rounded quantity. Used as the base value for discounting.
RUM DETAIL.ASS_CBD.CP.RUM	String	Read	RUM name. Used for matching the RUM in the discount detail.

Table 3-52 (Cont.) FCT_Discount EDR Container Fields

Entry	Format	Access	Description
SERVICE_CLASS_USED DETAIL.ASS_CBD.CP.SERVICE_CLASS_USED	String	Read	Service class. Used for matching the service class in the discount detail.
SERVICE_CODE_USED DETAIL.ASS_CBD.CP.SERVICE_CODE_USED	String	Read	Service code. Used for matching the service code in the discount detail.
TIMEMODEL_CODE DETAIL.ASS_CBD.CP.TIMEMODEL_CODE	String	Read	Time model code. Used for matching the time model in the discount detail.
TIMEZONE_CODE DETAIL.ASS_CBD.CP.TIMEZONE_CODE	String	Read	Time zone code. Used for matching the time zone in the discount detail.
USAGE_GL_ACCOUNT_CODE DETAIL.ASS_CBD.CP.USAGE_GL_ACCOUNT_CODE	String	Read	G/L code. Used for matching the G/L code in the discount detail.
ASS_CBD_ZONEMODEL_CODE DETAIL.ASS_CBD.CP.ZONEMODEL_CODE	String	Read	Zone model code. Used for matching the zone model in the discount detail.
DETAIL.ASS_CBD.CP.SPLIT_CP	Sub-block	Write	Optional sub-block added when charge packets are split.
DETAIL.ASS_CBD.CP.SPLIT_CP.RESOURCE_ID	Integer	Write	Numeric ID of the balance element. Used for filtering in the discount detail. Copied from the original charge packet.
DETAIL.ASS_CBD.CP.SPLIT_CP.RUM	String	Write	RUM name. Copied from the original charge packet.
DETAIL.ASS_CBD.CP.SPLIT_CP.QUANTITY_FROM	Decimal	Write	Split charge packet start quantity. Calculated by the module.
DETAIL.ASS_CBD.CP.SPLIT_CP.QUANTITY_TO	Decimal	Write	Split charge packet end quantity. Calculated by the module.
DETAIL.ASS_CBD.CP.SPLIT_CP.CHARGED_AMOUNT_VALUE	Decimal	Write	Amount of the charge for this split charge packet. Calculated by the module.
DETAIL.ASS_CBD.CP.SPLIT_CP.INTERN_PACKET_INDEX	Integer	Write	The index of the split charge packet.
DETAIL.ASS_CBD.CP.SPLIT_CP.INTERN_SRC_PACKET_INDEX	Integer	Write	The packet index of the charge packet from which this split charge packet was generated.
DP_DISCOUNT_BALANCE_GROUP_ID DETAIL.ASS_CBD.DP.BALANCE_GROUP_ID	String	Write	POID of the balance group impacted by this discount packet.
DP_CREATED DETAIL.ASS_CBD.DP.CREATED	String	Write	Creation date of the element.

Table 3-52 (Cont.) FCT_Discount EDR Container Fields

Entry	Format	Access	Description
DP_DISCOUNTBALIMPACTID DETAIL.ASS_CBD.DP.DISCOUNTBALIMPACTID	Integer	Write	Discount balance impact ID.
DP_DISCOUNTMODEL DETAIL.ASS_CBD.DP.DISCOUNTMODEL	String	Write	Discount used to create this discount packet.
DP_DISCOUNTRULE DETAIL.ASS_CBD.DP.DISCOUNTRULE	String	Write	Discount rule used to create this discount packet
DP_DISCOUNTSTEPID DETAIL.ASS_CBD.DP.DISCOUNTSTEPID	Integer	Write	Discount step used to create this discount packet
DP_DISCOUNT_GLID DETAIL.ASS_CBD.DP.GLID	String	Write	G/L ID as specified in the discount balance impact, otherwise copied from the charge packet.
DP_GRANTED_AMOUNT DETAIL.ASS_CBD.DP.GRANTED_AMOUNT	Decimal	Write	Granted discount/ sponsorship amount. Can be currency or noncurrency.
DP_GRANTED_QUANTITY DETAIL.ASS_CBD.DP.GRANTED_QUANTITY	Decimal	Write	Discount base value used to compute granted amount.
DP_DISCOUNT_IMPACT_CATEGORY DETAIL.ASS_CBD.DP.IMPACT_CATEGORY	String	Write	Impact category specified for this discount packet, otherwise copied from the charge packet.
DP_INTERN_DISC_MATCH_FACTOR DETAIL.ASS_CBD.DP.INTERN_DISC_MATCH_FACTOR	Decimal	Write	Discount match factor
DP_DISCOUNT_PACKET_INDEX DETAIL.ASS_CBD.DP.INTERN_PACKET_INDEX	Integer	Write	Packet ID.
DP_DISCOUNT_SRC_PACKET_INDEX DETAIL.ASS_CBD.DP.INTERN_SRC_PACKET_INDEX	Integer	Write	Source packet ID.
DP_INTERN_TOTAL_MATCH_FACTOR DETAIL.ASS_CBD.DP.INTERN_TOTAL_MATCH_FACTOR	Decimal	Write	Total discounted match factor.
DP_NODE_LOCATION DETAIL.ASS_CBD.DP.NODE_LOCATION	String	Write	Node location.
DP_OBJECT_ACCOUNT DETAIL.ASS_CBD.DP.OBJECT_ACCOUNT	Integer	Write	POID of discount owner.
DP_OBJECT_ID DETAIL.ASS_CBD.DP.OBJECT_ID	String	Write	Discount/sponsor object ID.
DP_OBJECT_OWNER_ID DETAIL.ASS_CBD.DP.OBJECT_OWNER_ID	Integer	Write	POID of the account or service that owns the discount.
DP_OBJECT_OWNER_TYPE DETAIL.ASS_CBD.DP.OBJECT_OWNER_TYPE	String	Write	POID type of discount owner, **/account** or **/ service**.

Table 3-52 (Cont.) FCT_Discount EDR Container Fields

Entry	Format	Access	Description
DP_OBJECT_TYPE DETAIL.ASS_CBD.DP.OBJECT_TYPE	String	Write	Discount/sponsor object that generated this discount.
DP_DISCOUNT_PRICEMODEL_CODE DETAIL.ASS_CBD.DP.PRICEMODEL_CODE	String	Write	Pricing used to generate this discount.
DP_DISCOUNT_QUANTITY DETAIL.ASS_CBD.DP.QUANTITY	Decimal	Write	Discounted noncurrency amount.
DP_QUANTITY_FROM DETAIL.ASS_CBD.DP.QUANTITY_FROM	Decimal	Write	Discount packet start quantity. Aligns with the QUANTITY_FROM value in a charge packet or a split charge packet.
DP_QUANTITY_TO DETAIL.ASS_CBD.DP.QUANTITY_TO	Decimal	Write	Discount packet end quantity. Aligns with the QUANTITY_TO value in a charge packet or a split charge packet.
DP_DISCOUNT_RATEPLAN DETAIL.ASS_CBD.DP.RATEPLAN	String	Write	Charge used to generate this discount.
DP_DISCOUNT_RATETAG DETAIL.ASS_CBD.DP.RATETAG	String	Write	Concatenation of the definition of the charge used to generate this discount.
DP_DISCOUNT_RESOURCE DETAIL.ASS_CBD.DP.RESOURCE_ID	Integer	Write	Balance Element ID of the balance element impacted by this discount.
DP_DISCOUNT_RUM DETAIL.ASS_CBD.DP.RUM	String	Write	The RUM entered for filtering in the discount filter.
DP_DISCOUNT_SERVICE_CLASS DETAIL.ASS_CBD.DP.SERVICE_CLASS	String	Write	Service class entered for filtering in the discount filter.
DP_DISCOUNT_SERVICE_CODE DETAIL.ASS_CBD.DP.SERVICE_CODE	String	Write	Service code entered for filtering in the discount filter.
DP_DISCOUNT_SUB_BALANCE.GRANTOR DETAIL.ASS_CBD.DP.SUB_BALANCE.GRANTOR	String	Read	ID of the charge offer or discount that granted this balance element.
DP_DISCOUNT_SUB_BALANCE.VALID_FROM_DETAILS DETAIL.ASS_CBD.DP.SUB_BALANCE.VALID_FROM_DETAILS	Integer	Read	The sub-balance start time mode (such as first-usage or relative) and relative offset details. This field is used in conjunction with SUB_BALANCE_VALID_FROM to determine the validity period start time.

Table 3-52 (Cont.) FCT_Discount EDR Container Fields

Entry	Format	Access	Description
DETAIL.ASS_CBD.DP.SUB_BALANCE.VALID_TO_DETAILS	Integer	Read	The sub-balance end time mode (such as relative) and relative offset details. This field is used in conjunction with SUB_BALANCE_VALID_TO to determine the validity period end time.
DP_DISCOUNT_TAX_CODE DETAIL.ASS_CBD.DP.TAX_CODE	String	Write	Tax code as specified in the discount balance impact. If not specified in the balance impact, copied from the charge packet for this discount packet.
DP_DISCOUNT_TIMEMODEL_CODE DETAIL.ASS_CBD.DP.TIMEMODEL_CODE	String	Write	Time model entered for filtering in the discount filter.
DP_DISCOUNT_TIMEZONE_CODE DETAIL.ASS_CBD.DP.TIMEZONE_CODE	String	Write	Time zone entered for filtering in the discount filter.
DP_DISCOUNT_VALID_FROM DETAIL.ASS_CBD.DP.VALID_FROM	Date	Write	Valid-from date for the grant in the discount packet.
DP_DISCOUNT_VALID_TO DETAIL.ASS_CBD.DP.VALID_TO	Date	Write	Valid-to date for the grant in the discount packet.
DP_DISCOUNT_VALID_FROM_DETAIL DETAIL.ASS_CBD.DP.VALID_FROM_DETAIL	Integer	Read	The valid-from mode (such as first usage or relative) and relative offset details. This field is used in conjunction with PIN_FLD_VALID_FROM to determine the validity period start time.
DP_DISCOUNT_VALID_TO_DETAIL DETAIL.ASS_CBD.DP.VALID_TO_DETAIL	Integer	Read	The valid-to mode (such as relative) and relative offset details. This field is used in conjunction with PIN_FLD_VALID_TO to determine the validity period end time.
DP_DISCOUNT_ZONEMODEL_CODE DETAIL.ASS_CBD.DP.ZONEMODEL_CODE	String	Write	Zone model code enter for filtering in the discount filter.
SUB_BAL_AMOUNT DETAIL.ASS_CBD.DP.SUB_BALANCE.AMOUNT	Decimal	Write	Amount of a sub-balance impacted by the discount packet.
SUB_BAL_CONTRIBUTOR DETAIL.ASS_CBD.DP.SUB_BALANCE.CONTRIBUTOR	String	Write	Contributor to a sub-balance impacted by the discount packet.

Table 3-52 (Cont.) FCT_Discount EDR Container Fields

Entry	Format	Access	Description
SUB_BAL_REC_ID DETAIL.ASS_CBD.DP.SUB_BALANCE.REC_ID	Integer	Write	ID of a sub-balance impacted by the discount packet.
SUB_BAL_VALID_FROM DETAIL.ASS_CBD.DP.SUB_BALANCE.VALID_FROM	Date	Write	Beginning validity date for a sub-balance impacted by the discount packet.
SUB_BAL_VALID_TO DETAIL.ASS_CBD.DP.SUB_BALANCE.VALID_TO	Date	Write	End validity date for a sub-balance impacted by the discount packet.
ACCOUNT_PARENT_ID DETAIL.CUST_A.ACCOUNT_PARENT_ID	String	Read	Customer account POID.
ACTG_LAST_DATE DETAIL.CUST_A.ACTG_LAST_DATE	Date	Read	The date that the current monthly cycle began.
ACTG_NEXT_DATE DETAIL.CUST_A.ACTG_NEXT_DATE	Date	Read	Date that the current monthly cycle ends.
ACTG_USED_DATE DETAIL.CUST_A.ACTG_USED_DATE	Date	Read	Date used for this EDR.
FIRST_USGAE_INDICATOR DETAIL.CUST_A.DL.PD.FIRST_USAGE_INDICATOR	Integer	Read	Specifies whether the discount's validity period is configured to start when first used.
DETAIL.CUST_A.DL.PD.USAGE_END		Write	
INTERN_FOUND_PP_INDEX DETAIL.CUST_A.INTERN_FOUND_PP_INDEX	Integer	Read	Purchased charge offer index of the charge offer or service.
SERVICE_ID DETAIL.CUST_A.PRODUCT.SERVICE_ID	String	Read	POID of the service instance.
BG_BG_ID DETAIL.CUST_A.BG.BALANCE_GROUP_ID	String	Read	POID of the balance group for the account.
BG_BELEM_CURR_BAL DETAIL.CUST_A.BG.BAL_ELEM.CURR_BAL	Decimal	Read	Current balance of the balance group element.
BG_BELEM_RESOURCE_ID DETAIL.CUST_A.BG.BAL_ELEM.RESOURCE_ID	Integer	Read	Balance Element ID of the balance group element.
DISCOUNT_BALANCE_GROUP_ID DETAIL.CUST_A.DL.BALANCE_GROUP_ID	String	Read	Balance group used to evaluate this discount instance.
DISCOUNT_OWNER_ACCT_ID DETAIL.CUST_A.DL.DISCOUNT_OWNER_ACCT_ID	String	Read	POID of the account that owns the discount, either directly or indirectly through a service.

Table 3-52 (Cont.) FCT_Discount EDR Container Fields

Entry	Format	Access	Description
DISOUNT_OWNER_ID DETAIL.CUST_A.DL.DISCOUNT_OWNER_ID	String	Read	POID of the account or service that owns the discount. Identical to DISCOUNT_OWNER_ACCT_ID if the discount is owned directly by an account.
DISOUNT_OWNER_TYPE DETAIL.CUST_A.DL.DISCOUNT_OWNER_TYPE	String	Read	Discount owner type, either **/account** or **/service**.
PD_DISCOUNTID DETAIL.CUST_A.DL.PD.DISCOUNT_ID	String	Read	POID of the discount object.
PD_DISCOUNTMODEL DETAIL.CUST_A.DL.PD.DISCOUNT_MODEL	String	Read	Code of a discount referenced in the discount object.
PD_FLAGS DETAIL.CUST_A.DL.PD.FLAGS	Integer	Read	Proration setting.
PD_MODE DETAIL.CUST_A.DL.PD.MODE	Integer	Read	Discount mode, either cascading or parallel.
PD_NODE_LOCATION DETAIL.CUST_A.DL.PD.NODE_LOCATION	String	Read	Unique ID of the discount object.
PD_QUANTITY DETAIL.CUST_A.DL.PD.QUANTITY	Integer	Read	Number of purchased discounts. This is multiplied by balance impact of this discount instance.
PD_SCALE DETAIL.CUST_A.DL.PD.SCALE	Decimal	Read	Proration scale.
PD_VALID_FLAG DETAIL.CUST_A.DL.PD.VALID_FLAG	Integer	Read	Indicates whether the discount is valid.
SPONSOR_BALANCE_GROUP_ID DETAIL.CUST_A.SL.BALANCE_GROUP_ID	String	Read	POID of the charge share group balance group.
SPONSOR_OWNER_ACCT_ID DETAIL.CUST_A.SL.SPONSOR_OWNER_ACCT_ID	String	Read	POID of the account that owns the chargeshare object, either directly or indirectly through a service.
SPONSOR_OWNER_ID DETAIL.CUST_A.SL.SPONSOR_OWNER_ID	String	Read	POID of the account or service that owns the chargeshare. Identical to SPONSOR_OWNER_ACCT_ID if the chargeshare is owned directly by an account.
SPONSOR_OWNER_TYPE DETAIL.CUST_A.SL.SPONSOR_OWNER_TYPE	String	Read	Chargeshare owner type, either **/account** or **/service**.

Table 3-52 (Cont.) FCT_Discount EDR Container Fields

Entry	Format	Access	Description
SD_DISCOUNTMODEL DETAIL.CUST_A.SL.SD.DISCOUNT_MODEL	String	Read	Discount used for this chargeshare.
SD_SPONSORID DETAIL.CUST_A.SL.SD.SPONSORSHIP_ID	String	Read	The POID of the chargeshare (*/sponsorship*) object.
SD_VALID_FLAG DETAIL.CUST_A.SL.SD.VALID_FLAG	Integer	Read	Indicates whether the chargeshare is valid.
DETAIL.CUST_A.SUBCYCLE_PL.DL.PD.USAGE_END	N/A	Read	N/A
DETAIL.CUST_A.SUBCYCLE_PL.DL.PD.FIRST_USAGE_INDICATOR	N/A	Read	N/A
TRANSACTION_ID INTERNAL.TRANSACTION_ID	Decimal	Read	The transaction ID. Needed for queuing.

FCT_DiscountAnalysis

The FCT_DiscountAnalysis module determines which discounts apply to a given event.

See "Configuring Discounting Modules and Components" in *BRM Setting Up Pipeline Rating and Discounting*.

Dependencies

The FCT_DiscountAnalysis module requires a connection to the following data modules:

- DAT_Discount
- DAT_ModelSelector

For pipeline rating, this module must run after the FCT_Account module and before the FCT_Discount module.

For real-time rating, this module must run before the FCT_Discount module.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-53 lists the FCT_DiscountAnalysis registry entries.

Table 3-53 FCT_DiscountAnalysis Registry Entries

Entry	Description	Mandatory
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive You can use this entry in a semaphore file.	Yes

Table 3-53 (Cont.) FCT_DiscountAnalysis Registry Entries

Entry	Description	Mandatory
DiscountModelDataModule	Specifies the connection to the DAT_Discount module.	Yes
Filter_SetModule	Specifies the connection to the FCT_Filter_Set module. Use this entry if the FCT_Filter_Set module is configured. See: • About Using Filter Sets to Apply System Products and Discounts • FCT_Filter_Set	No
ModelSelectorDataModule	Specifies the connection to the DAT_ModelSelector module.	Yes

Sample Registry

```
Discount
{
    ModuleName = FCT_DiscountAnalysis
    Module
    {
       Active = True
       DiscountModelDataModule = ifw.DataPool.DiscountModelDataModule
       Filter_SetModule = ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.Filter_Set
       ModelSelectorDataModule = ifw.DataPool.ModelSelectorDataModule
    }
}
```

Semaphore File Entries

Table 3-54 lists the FCT_DiscountAnalysis Semaphore file entry.

Table 3-54 FCT_DiscountAnalysis Semaphore File Entry

Entry	Description
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive

Sample Semaphore File Entry

```
ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.DiscountAnalysisModule.
Module.Active = False
```

EDR Container Fields

Table 3-55 lists the FCT_DiscountAnalysis EDR container fields.

Table 3-55 FCT_DiscountAnalysis EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
ASS_CBD DETAIL.ASS_CBD	Block	R	n/a
ASS_CBD_CHARGE_PACKET DETAIL.ASS_CBD.CP	Block	R	Charge packet; used to check for any discount ERAs set.
BDR_CHARGING_START_TIMESTAMP DETAIL.CHARGING_START_TIMESTAMP	Date	R	Event start time.
BDR_CHARGING_END_TIMESTAMP DETAIL.CHARGING_END_TIMESTAMP	Date	R	Event end time.
EVENT_TYPE DETAIL.EVENT_TYPE	String	R	The event type used to locate the event discount.
BDR_UTC_TIME_OFFSET DETAIL.UTC_TIME_OFFSET	String	R	The UTC offset used to adjust the start and end time.
INTERN_BALANCE_GROUP_ID DETAIL.INTERN_BALANCE_GROUP_ID	String	RW	Account level balance group of the event owner account.
INTERN_DISCOUNT_OWNER_ACCT_ID DETAIL.INTERN_DISCOUNT_OWNER_ACCT_ID	String	RW	Event owner account ID.
DISCOUNT_LIST DETAIL.CUST_A.DL	Block	R	Purchased discount offers that belong to an account or service, or are shared by an account or service.
BALANCE_GROUP_ID DETAIL.CUST_A.DL.BALANCE_GROUP_ID	String	RW	ID of the balance group whose balance elements are used for the discounts in the discount list.
DISCOUNT_OWNER_ACCT_ID DETAIL.CUST_A.DL.DISCOUNT_OWNER_ACCT_ID	String	RW	ID of the account that owns the set of purchased discount offers in the discount list.
PURCHASED_DISCOUNTS DETAIL.CUST_A.DL.PD	String	CW	Information about the discount.
DISCOUNT_ID DETAIL.CUST_A.DL.PD.DISCOUNT_ID	String	RW	Discount Ooffer object ID.
STATUS DETAIL.CUST_A.DL.PD.STATUS	String	RW	Discount offer state (active, inactive, or cancelled).
PURCHASE_START DETAIL.CUST_A.DL.PD.PURCHASE_START	Date	RW	Discount offer purchase start time.
PURCHASE_END DETAIL.CUST_A.DL.PD.PURCHASE_END	Date	RW	Discount offer purchase end time.
USAGE_START DETAIL.CUST_A.DL.PD.USAGE_START	Date	RW	Discount usage start time.

Table 3-55 (Cont.) FCT_DiscountAnalysis EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
USAGE_END DETAIL.CUST_A.DL.PD.USAGE_END	Date	RW	Discount usage end time.
PRIORITY DETAIL.CUST_A.DL.PD.PRIORITY	Integer	RW	Discount offer priority.
MODE DETAIL.CUST_A.DL.PD.MODE	Integer	RW	Discount mode (parallel or cascading).
VALID_FLAG DETAIL.CUST_A.DL.PD.VALID_FLAG	Integer	RW	A value indicating discount validity.
TYPE DETAIL.CUST_A.DL.PD.TYPE	Integer	RW	Discount type (system, subscription, or item).
QUANTITY DETAIL.CUST_A.DL.PD.QUANTITY	Decimal	RW	Purchase quantity.
SCALE DETAIL.CUST_A.DL.PD.SCALE	Decimal	RW	Proration scale.
OFFERING_POID DETAIL.CUST_A.DL.PD.OFFERING_POID	String	RW	A value that identifies the purchased discount associated with the account.
DISCOUNT_MODEL DETAIL.CUST_A.DL.PD.DISCOUNT_MODEL	String	RW	A discount.
SPONSOR_LIST DETAIL.CUST_A.SL	Block	R	The list of sponsors that split the charges with the event user.
SPONSOR_BALANCE_GROUP_ID DETAIL.CUST_A.SL.BALANCE_GROUP_ID	String	R	The balance group whose balance elements are used for the sponsorship list.
SPONSORSHIP_DETAILS DETAIL.CUST_A.SL.SD	Block	R	Sponsorships that belong to an account or service or are shared by the account or service.
SPONSORSHIP_ID DETAIL.CUST_A.SL.SD.SPONSORSHIP_ID	String	R	Sponsorship object ID.
SPONSOR_VALID_FLAG DETAIL.CUST_A.SL.SD.VALID_FLAG	Integer	RW	A value that indicates sponsorship validity.
SPONSOR_DISCOUNT_MODEL DETAIL.CUST_A.SL.SD.DISCOUNT_MODEL	String	RW	Sponsorship model.

FCT_DroppedCall

The FCT_DroppedCall module identifies phone calls that were terminated due to technical reasons and then resumed again through a customer's subsequent phone call.

Dependencies

Run the FCT_DroppedCall module *after* the FCT_Account module.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-56 lists the FCT_DroppedCall registry entries.

Table 3-56 FCT_DroppedCall Registry Entries

Entry	Description	Mandatory
Active	Specifies whether the module is active or inactive. **True** = Active **False** = Inactive You can use this entry in a semaphore file.	Yes
FilePath	Specifies the path to the dropped calls data file. The default is the data directory (**./data**).	No
FileName	Specifies the name of the dropped calls data file, which stores back-up information about dropped calls. You use this file to restore information in case of system error or system restart.	Yes
TempPrefix	Specifies the prefix for the temporary dropped calls data files. The default is **tmp**_.	No
CheckField	Section that specifies the EDR field and values used to identify a dropped call.	Yes
CheckField.Name	Specifies the EDR field that is used to identify a dropped call. **Note:** Only one EDR field can be used to identify a dropped call.	Yes
CheckField.Value	Specifies the values for identifying a dropped call. **Note:** If more than one value qualifies an EDR as a dropped call, enter multiple values separated by a comma (,) with no spaces; for example: 5,6,7. BRM interprets the comma as a Boolean OR value.	Yes
WrittenFields	Section that specifies the dropped call EDR fields that are written into memory and are used to detect continuation calls. You use dummy key values such as 1 and 2 to list the EDR fields, as shown below: **1 =** *EDR_field* **2 =** *EDR_field* **3 =** *EDR_field* **Note:** BRM automatically writes the DETAIL.A_NUMBER, DETAIL.B_NUMBER, DETAIL.CHARGING_END_TIMESTAMP, and DETAIL.CUST_A.BILL_NEXT_DATE EDR fields to memory, so you should not list these fields.	No
AddedFields	Section that specifies the dropped call EDR fields that are written into memory and then added to the continuation call EDR. Important: When you map a dropped call EDR field to a continuation call EDR field, both fields must have the same data type. You can find a field's data type by reading the container description file (**container.dsc**). By default, the module does not enrich the continuation call EDR.	No

Table 3-56 (Cont.) FCT_DroppedCall Registry Entries

Entry	Description	Mandatory
AddedFields.Field*x*	Section that maps one dropped call EDR field to one continuation call EDR field. You create a **Field***x* section for each pair of EDR fields that you want to map. For example, if you want to map three EDR pairs, create a **Field1** section, a **Field2** section, and a **Field3** section.	No
AddedFields.Field*x***.ContinuationCallField**	Specifies the continuation call EDR field. The module adds the value of the dropped call EDR field specified in **DroppedCallField** to the continuation call EDR field that you specify.	No
AddedFields.Field*x***.DroppedCallField**	Specifies the dropped call EDR field to write into memory. This field's value is added to the continuation call EDR field specified in **ContinuationCallField**.	No

Sample Registry

```
DroppedCall
{
  ModuleName = FCT_DroppedCall
  Module
  {
    Active = TRUE
    FilePath = ./data
    FileName = dropped_call
    TempPrefix = tmp_
    CheckField
    {
      Name = DETAIL.CALL_COMPLETION_INDICATOR
      Value = 5
    }
    WrittenFields
    {
      1 = DETAIL.RECORD_TYPE
    }
    AddedFields
    {
      Field1
      {
        ContinuationCallField = DETAIL.DROPPED_CALL_QUANTITY
        DroppedCallField = DETAIL.DURATION
      }
    }
  }
}
```

With this sample registry configuration, the FCT_DroppedCall module identifies:

- *Dropped calls* by finding all EDRs with a DETAIL.CALL_COMPLETION_INDICATOR EDR field set to **5**.
- *Continuation calls* by using the DETAIL.RECORD_TYPE EDR field.

> **Note:**
>
> By default, the module also automatically writes the DETAIL.A_NUMBER, DETAIL.B_NUMBER, DETAIL.CUST_A.BILL_NEXT_DATE, and DETAIL.CHARGING_END_TIMESTAMP EDR fields to memory.

When the module detects a continuation call, it adds the value of the dropped call's DETAIL.DURATION EDR field to the continuation call's DETAIL.DROPPED_CALL_QUANTITY EDR field.

Semaphore File Entries

Table 3-57 lists the FCT_DroppedCall Semaphore file entries.

Table 3-57 FCT_DroppedCall Semaphore File Entries

Entry	Description
Active	Specifies whether the module is active or inactive. **True** = Active **False** = Inactive
RemoveLimit	Specifies to remove all calls from the memory map and data file that are older than the specified number of days. For example, if you specify **7**, BRM removes from the memory map all entries older than 7 days. The time is calculated from the current system time. **Note:** Set the time to a large enough value to allow for late-arriving and recycled EDRs.

Sample Semaphore File Entries

```
ifw.Pipelines.ALL_RATE.Functions.FunctionPool.DroppedCall.Active = True
ifw.Pipelines.ALL_RATE.Functions.FunctionPool.DroppedCall.RemoveLimit = 7
```

EDR Container Fields

The FCT_DroppedCall module accesses the EDR container fields shown in Table 3-58. You can configure the module to access additional EDR container fields by using the module's **CheckField**, **WrittenFields**, and **AddedFields** registry entries.

Table 3-58 FCT_DroppedCall EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
A_NUMBER DETAIL.A_NUMBER	String	Read	Contains the customer A number.
B_NUMBER DETAIL.B_NUMBER	String	Read	Contains the customer B number.
CUST_A_BILL_NEXT_DATE DETAIL.CUST_A.BILL_NEXT_DATE	Date	Read	Contains the timestamp for the customer's next billing cycle.
CHARGING_END_TIMESTAMP DETAIL.CHARGING_END_TIMESTAMP	Date	Read	Contains the dropped call's ending timestamp.

Table 3-58 (Cont.) FCT_DroppedCall EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
DURATION DETAIL.DURATION	Decimal	Read	Contains the duration of the current call.
CUST_A_PROFILE DETAIL.CUST_A.ERA.PROFILE	String	Read	Contains the customer's account-related ERA data.
CUST_A_KEY DETAIL.CUST_A.ERA.PA.KEY	String	Read	Contains the key for the account-related ERA data.
CUST_A_VALUE DETAIL.CUST_A.ERA.PA.VALUE	String	Read	Contains the value for the account-related ERA data.
CALL_COMPLETION_INDICATOR DETAIL.CALL_COMPLETION_INDICATOR	String	Read	Contains the reason the current call session was terminated.
DROPPED_CALL_STATUS DETAIL.DROPPED_CALL_STATUS	Integer	Write	Flags the status of the call: **0** = Normal call **1** = Dropped call **2** = Continuation call **3** = Both a dropped call and a continuation call **4** = A call that was processed by FCT_DroppedCall but didn't qualify as a dropped call or a continuation call.
DROPPED_CALL_QUANTITY DETAIL.DROPPED_CALL_QUANTITY	Decimal	Write	When the EDR is flagged as a continuation call, this field contains the duration of the associated dropped call.

FCT_DuplicateCheck

The FCT_DuplicateCheck module checks for duplicate EDRs. See "Handling Duplicate EDRs" in *BRM Setting Up Pipeline Rating and Discounting*.

> **Note:**
>
> Before using the FCT_DuplicateCheck module, load the duplicate check stored procedures in the Pipeline Manager database.

Dependencies

To enable your system to check for duplicate EDRs without using excessive disk space, connect the FCT_DuplicateCheck module to the Pipeline Manager database.

The FCT_DuplicateCheck module is typically the second module in a pipeline, directly following the FCT_PreSuspense module. This ensures that no further processing is done on duplicate EDRs.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-59 lists the FCT_DuplicateCheck registry entries.

Table 3-59 FCT_DuplicateCheck Registry Entries

Entry	Description	Mandatory
Active	Specifies whether the module is active or inactive. **True** = Active **False** = Inactive You can use this entry in a semaphore file.	Yes
BufferLimit	Specifies the oldest date that previously processed EDRs can be stored in memory. The format is *YYYYMMDD*. Note: The **BufferLimit** date must be equal to or later than the **StoreLimit** date. For example, if the **StoreLimit** date is June 1, the **BufferLimit** must be June 1 or later. You can use this entry in a semaphore file. See "Setting Date Parameters for Storing Processed EDRs" in *BRM Setting Up Pipeline Rating and Discounting*.	Yes
BulkInsertArraySize	Specifies the maximum number of rows for bulk insert when the data in memory flushes to the database. Default = **10000**.	No
DataConnection	Specifies a connection to the Pipeline Manager database. **Important:** To avoid using excessive disk space when checking for duplicate EDRs, enable this entry.Default = **False**. See "About Storing EDRs in a Database Instead of Files" in *BRM Setting Up Pipeline Rating and Discounting*.	No
DuplicateIndicatorField	Specifies the EDR field to set if an EDR is a duplicate. The field specified must be an integer field. You can use any integer field in the EDR. This entry is used by the FCT_CiberOcc module to determine whether to create an OCC record. OCC records are not created for duplicate EDRs.	No
Fields	Specifies the EDR fields that are used for checking. **Important:** Do not use the DETAIL_CHARGING_START_TIMESTAMP field for duplicate checking. See "Specifying the Fields to Use for Duplicate Check" in *BRM Setting Up Pipeline Rating and Discounting*.	Yes
FileName	Specifies the base file name of the data files (the transaction ID and suffix are appended). Default = **.** See "Managing FCT_DuplicateCheck Data Files" in *BRM Setting Up Pipeline Rating and Discounting*.	Yes

Table 3-59 (Cont.) FCT_DuplicateCheck Registry Entries

Entry	Description	Mandatory
IndexSpaceName	Index space name where the run-time duplicate check index is created (database mode only). For example: `IndexSpaceName = INTEGRATE_TS_1_IDX`	Yes, if using a database connection.
Path	Specifies the directory for the data files that store EDRs. See "Managing FCT_DuplicateCheck Data Files" in *BRM Setting Up Pipeline Rating and Discountin*g.	No
SearchKey	Identifies duplicate EDRs. See "Specifying a Search Key for Duplicate Check" in *BRM Setting Up Pipeline Rating and Discounting*. **Important:** If you use the **SearchKey** registry entry, don't list the **SearchKey** value as a field in the **Fields** list.	No
StoreLimit	Specifies the oldest date that previously processed EDRs can be stored. If an EDR is dated earlier than the **StoreLimit** date, the EDR is not processed by the FCT_DuplicateCheck module. The format is *YYYYMMDD*. **Note:** The **StoreLimit** date must be equal to or earlier than the **BufferLimit** date. For example, if the **StoreLimit** date is June 1, the **BufferLimit** must be June 1 or later. You can use this entry in a semaphore file. See "Setting Date Parameters for Storing Processed EDRs" in *BRM Setting Up Pipeline Rating and Discounting*.	Yes
StreamName	Specifies the output stream for duplicate EDRs. See "Configuring Output for Rejected or Duplicate EDRs" in *BRM Setting Up Pipeline Rating and Discounting*.	Yes
TableSpaceName	Table space name where the run-time duplicate check table is created (database mode only). When a **StoreLimit** or **BufferLimit** semaphore is sent, FCT_DuplicateCheck needs to know where to store data. This entry, and the **IndexSpaceName** entry specify the location in the database. For example: `TableSpaceName = INTEGRATE_TS_1_DAT`	Yes, if using a database connection.
TableSuffix	Allows you to create multiple IFW_DUPLICATECHECK tables when you run multiple pipelines.	No

Sample Registry

> **Note:**
>
> When entering data in the **Fields** entry, use dummy key values such as **1**, **2**, and **3**, as shown in this example.

```
DuplicateCheck
{
   ModuleName  =  FCT_DuplicateCheck
   Module
   }
      Active  =  True
      DataConnection  =  ifw.DataPool.DupLogin1
      Path  =  ./data/dup
      Filename  =  call.duplicate
      StreamName  =  DuplicateOutput
      BufferLimit  =  20040105
      StoreLimit  =  20040101
      SearchKey  =  DETAIL.A_NUMBER
      Fields
      {
        1  =  DETAIL.BASIC_SERVICE
        2  =  DETAIL.B_NUMBER
      }
   }
}
```

With this sample registry configuration, the following occurs:

- EDRs dated January 5, 2004 (**BufferLimit**) or later are stored in the duplicate list in memory.
- EDRs dated January 1, 2004 (**StoreLimit**) through January 4, 2004 are stored in the IFW_DUPLICATECHECK database table.
- EDRs dated December 31, 2003 or earlier are ignored.

The following day, the module receives the following update registry:

```
DuplicateCheck
{
   ModuleName  =  FCT_DuplicateCheck
   Module
   {
      BufferLimit  =  20040106
      StoreLimit  =  20040102
   }
}
```

With this updated registry sample configuration, the following occurs:

- EDRs dated January 6, 2004 (new **BufferLimit**) or later are stored in the duplicate check list in memory.
- EDRs dated January 5, 2004 are moved to the IFW_DUPLICATECHECK database table.
- EDRs dated January 2, 2004 (new **StoreLimit**) through January 4, 2004 continue to be stored in the database table.

> **Note:**
>
> EDR data for duplicate checks is stored in the IFW_DUPLICATECHECK database table. This table can be hosted by any database. Normally, at the end of the day, all the EDR data in memory is flushed to the database.

Semaphore File Entries

Table 3-60 lists the FCT_DuplicateCheck Semaphore file entries.

Table 3-60 FCT_DuplicateCheck Semaphore File Entries

Entry	Description
Active	Specifies whether the module is active or inactive. **True** = Active **False** = Inactive
BufferLimit	Specifies the oldest date that previously processed EDRs can be stored in memory. The format is *YYYYMMDD*. **Note:** The **BufferLimit** date must be equal to or later than the **StoreLimit** date. For example, if the **StoreLimit** date is June 1, the **BufferLimit** must be June 1 or later. See "Setting Date Parameters for Storing Processed EDRs" in *BRM Setting Up Pipeline Rating and Discounting*.
StoreLimit	Specifies the oldest date that previously processed EDRs can be stored. If an EDR is dated earlier than the **StoreLimit** date, the EDR is not processed by the FCT_DuplicateCheck module. The format is *YYYYMMDD*. **Note:** The **BufferLimit** date must be equal to or later than the **StoreLimit** date. For example, if the **StoreLimit** date is June 1, the **BufferLimit** must be June 1 or later. See "Setting Date Parameters for Storing Processed EDRs" in *BRM Setting Up Pipeline Rating and Discounting*.

Sample Semaphore File Entry

```
ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.DuplicateCheck.
Module.StoreLimit = 20020101

ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.DuplicateCheck.
Module.BufferLimit = 20020125
```

Sample Output Configuration

You configure the output stream for the FCT_DuplicateCheck module in the Output section of the registry, for example:

```
# Output stream for duplicate events
DuplicateOutput
{
 ModuleName = OUT_Reject
 Module
 {
  OutputStream
  {
   ModuleName = EXT_OutFileManager
   Module
   {
    OutputPath = ./samples/wireless/data/rej
    OutputPrefix = test
    OutputSuffix = .dup
    TempPrefix   = tmp
```

```
    TempDataPath    = ./samples/wireless/data/rej
    TempDataPrefix  = dup.tmp.
    TempDataSuffix  = .data

    Replace = TRUE
    DeleteEmptyFile = TRUE
   }
  }
 }
} # end of DuplicateOutput
```

> **Note:**
>
> To ensure output file integrity, specify a unique combination of OutputPath, OutputSuffix, and OutputPrefix values for each output stream defined in the registry.

EDR Container Fields

You specify the EDR container fields in the FCT_DuplicateCheck module **Fields** startup registry entry.

Database Tables

The FCT_DuplicateCheck module uses the IFW_DUPLICATECHECK database table.

If you use the database instead of a file to define the data that the FCT_DuplicateCheck module uses for comparing EDRs, you need to create this table. The FCT_DuplicateCheck module uses the data in the IFW_DUPLICATECHECK table to check for duplicate EDRs. See "Handling Duplicate EDRs" in *BRM Setting Up Pipeline Rating and Discounting*.

The IFW_DUPLICATECHECK table should have a unique index. A duplicate EDR is detected by the database reporting a violation of the uniqueness when attempting to INSERT.

> **Note:**
>
> Oracle recommends that you have multiple partitions to increase the INSERT performance.

FCT_EnhancedSplitting

The FCT_EnhancedSplitting module specifies different output streams for EDRs based on rules. For example:

- You can split EDRs for different service types to different output streams.
- You can split EDRs from roaming outcollects and incollects into different streams.

See "Using Rules to Send EDRs to Different Output Streams" in *BRM Setting Up Pipeline Rating and Discounting*.

Dependencies

Requires a connection to the Pipeline Manager database.

Because you can split EDRs based on service codes, this module should run after the FCT_ServiceCodeMap and FCT_UsageClassMap modules.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-61 lists the FCT_EnhancedSplitting registry entries.

Table 3-61 FCT_EnhancedSplitting Registry Entries

Entry	Description	Mandatory
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive You can use this entry in a semaphore file.	Yes
DataConnection	Specifies the connection to the Pipeline Manager database.	Yes
DefaultOutput	Specifies the default output stream if no splitting rule matches the current EDR. If no default output stream is specified, the EDR receives the error message ERR_NO_SPLITTING_PERFORMED. Default = **.** See "Configuring EDR Output Processing" in *BRM Setting Up Pipeline Rating and Discounting*.	No
SystemBrands	Maps the system brand table to the output stream. Each entry in this section has the format SYSBRAND=OUTPUT-STREAM. See "Using Rules to Send EDRs to Different Output Streams" in *BRM Setting Up Pipeline Rating and Discounting*.	Yes

Sample Registry

```
Splitting
{
   ModuleName = FCT_EnhancedSplitting
   Module
      {
      Active = True
      DataConnection = Login
      DefaultOutput = EdrOutput0
      SystemBrands
         {
         1 = EdrOutput1
         2 = EdrOutput2
      }
   }
}
```

Semaphore File Entries

Table 3-62 lists the FCT_EnhancedSplitting Semaphore file entries.

Table 3-62 FCT_EnhancedSplitting Semaphore File Entries

Entry	Description
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive
Reload	Reloads data from the database.

Sample Semaphore File Entry

```
ifw.Pipelines.PRE_PRODCESS.Functions.PreProcessing.FunctionPool.
EnhancedSplitting.Module.Active = True
```

EDR Container Fields

Table 3-63 lists the FCT_EnhancedSplitting EDR container fields.

Table 3-63 FCT_EnhancedSplitting EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
RECORD_TYPE DETAIL.RECORD_TYPE	String	Read	Contains the record type.
INTERN_SERVICE_CODE DETAIL.INTERN_SERVICE_CODE	String	Read	Contains the internal service code.
INTERN_USAGE_CLASS DETAIL.INTERN_USAGE_CLASS	String	Read	Contains the internal usage class.
SOURCE_NETWORK DETAIL.SOURCE_NETWORK	String	Read	Contains the source network code. This could either be the PLMN ID or any logical operator code.
DESTINATION_NETWORK DETAIL.DESTINATION_NETWORK	String	Read	Contains the destination network code.
ASS_GSMW_EXT.ORIGINATING_SWITCH_IDENTIFICATION DETAIL.ASS_GSMW_EXT.ORIGINATING_SWITCH_IDENTIFICATION	String	Read	Contains the GSM MSC or Switch ID handling the origin of the call.
ASS_GPRS_EXT.ORIGINATING_SWITCH_IDENTIFICATION DETAIL.ASS_GPRS_EXT.ORIGINATING_SWITCH_IDENTIFICATION	String	Read	Contains the GPRS MSC or Switch ID handling the origin of the call.
ASS_GSMW_EXT.TRUNK_INPUT DETAIL.ASS_GSMW_EXT.TRUNK_INPUT	String	Read	Contains the trunk identification (in-route address in network switches).

Table 3-63 (Cont.) FCT_EnhancedSplitting EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
ASS_GSMW_EXT_TRUNK_OUTPUT DETAIL.ASS_GSMW_EXT.TRUNK_OUTPUT	String	Read	Contains the trunk identification (out-route address in network switches).
A_NUMBER DETAIL.A_NUMBER	String	Read	Contains the A number.
B_NUMBER DETAIL.B_NUMBER	String	Read	Contains the B number.
INTERN_C_NUMBER_ZONE DETAIL.INTERN_C_NUMBER_ZONE	String	Read	Contains the number where the call was first terminated if it was forwarded or routed.
CHARGING_START_TIMESTAMP DETAIL.CHARGING_START_TIMESTAMP	Date	Read	Contains the time-stamp used for start of charging.

Database Tables

The FCT_EnhancedSplitting module uses the following database tables:

- IFW_SPLITTING_TYPE. The FCT_EnhancedSplitting module uses the data in the IFW_SPLITTING_TYPE table to determine how to route EDRs to different output streams based on rules. See "Using Rules to Send EDRs to Different Output Streams" in *BRM Setting Up Pipeline Rating and Discounting*.
- IFW_SYSTEM_BRAND. The IFW_SYSTEM_BRAND table stores the system brand codes. See "Using Rules to Send EDRs to Different Output Streams" in *BRM Setting Up Pipeline Rating and Discounting*.

> **Note:**
>
> For information on compare patterns used in database values, see "About Using Regular Expressions when Specifying the Data to Extract" in *BRM Setting Up Pipeline Rating and Discounting*.

FCT_EventOrder

When an event is rated, the FCT_EventOrder module uses the criteria and the event timestamps to determine if the event needs to be rerated.

Dependencies

This module must run *after* the FCT_MainRating and FCT_Discount modules and *before* the FCT_Reject module.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-64 lists the FCT_EventOrder registry entries.

Table 3-64 FCT_EventOrder Registry Entries

Entry	Description	Mandatory
Active	Specifies whether the module is active or inactive: **True** = Active **False** = Inactive	Yes
AccountDataModule	Specifies the connection to the DAT_AccountBatch module.	Yes
PortalConfigDataModule	Specifies the connection to the DAT_PortalConfig module.	Yes
RerateDelayTolerance	Specifies a time in minutes that controls how much out-of-order EDR data FCT_EventOrder writes to a rerate-request file before creating a new file.	No
NumberOfAccountLimit	Specifies the number of accounts FCT_EventOrder assigns to each rerate job. This entry affects batch rerating throughput.	No
OutputDirectory	Specifies the output directory for out-of-order events. **Important:** You must create this directory. It is not created by BRM installation scripts.	Yes
OutputPrefix	Specifies the prefix of the rerate-request file name. The default is **ood**.	Yes
SkipPrevBillingCycle	Specifies whether to skip events that belong to previous billing cycles. **True** = Skip the events **False** = Do not skip the events (Default)	No

Sample Registry

```
EventOrder
{
  ModuleName                = FCT_EventOrder
  Module
  {
    Active                  = True
    AccountDataModule       = ifw.DataPool.CustomerData
    PortalConfigDataModule  = ifw.DataPool.PortalConfigDataModule

#delay tolerance in minutes for creating rerate jobs
    RerateDelayTolerance    =  180

#maximum number of accounts in the rerate-request file
#this should match the value of the pin_rerate "per_job"
#configuration entry
    NumberOfAccountLimit    = 1000

#output directory and prefix of the rerate-request file
    OutputDirectory         = Pipeline_home/data/out/ood
    OutputPrefix            = ood_

  }
}
```

FCT_ExchangeRate

The FCT_ExchangeRate module converts the currency in the charge packets, discount packets, and tax packets to the home (system) currency or the customer's billing currency.

Dependencies

Requires a connection to the DAT_ExchangeRate module and the DAT_Currency module.

This module must run *after* the "FCT_MainRating" module.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-65 lists the FCT_ExchangeRate registry entries.

Table 3-65 FCT_ExchangeRate Registry Entries

Entry	Description	Mandatory
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive You can use this entry in a semaphore file.	Yes
CurrencyDataModule	Specifies the connection to the DAT_Currency module. When currency module is specified, the resourceid field of the charge packet is updated to the new currency. The resourceid is needed only when loading the records into the BRM database by using Rated Event Loader.	No
ErrorMessages	Specifies whether error messages should be appended to the EDR container or should be suppressed.	Yes
ExchangeRateDataModule	Specifies the connection to the DAT_ExchangeRate module.	Yes
HomeCurrency	Specifies the local currency used by your company. The code must use three characters; for example, USD or DEM. You must set this value when the **RatingDateHome** registry value is set. This entry is used only if the **RatingDateHome** entry is used.	No
Mode	Specifies how to apply the exchange rate. Values are: • Normal - Converts the From Currency to the To Currency by multiplying the From Currency amount and exchange rate. • Reverse - Converts the To Currency to the From Currency by multiplying the To Currency amount and 1/exchange rate. Default mode is **Normal**.	No

Table 3-65 (Cont.) FCT_ExchangeRate Registry Entries

Entry	Description	Mandatory
RatingDateBilling	Specifies how to determine the validity date for the conversion from the rating currency to the billing currency. Values are: • SYSTEM • FILE • CDR • NONE Default = **NONE** **Note:** FCT_ExchangeRate module converts currency to home currency or billing currency. If you specify RatingDateBilling in addition to RatingDateHome and HomeCurrency parameters, it converts the currency to the home currency only.	No
RatingDateHome	Specifies how to determine the validity date for the conversion from the rating currency to the home currency. Values are: • SYSTEM • FILE • CDR • NONE Default = **NONE**	No

Sample Registry

```
ExchangRate
{
  ModuleName = FCT_ExchangeRate
  Module
  {
    Active = True
    ExchangeRateDataModule = ExchangeRateData
    RatingDateBilling  =  SYSTEM
    RatingDateHome = CDR
    HomeCurrency = DEM
    ErrorMessages = False
  }
}
```

Semaphore File Entries

Table 3-66 lists the FCT_ExchangeRate Semaphore file entries.

Table 3-66 FCT_ExchangeRate Semaphore File Entry

Entry	Description
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive

EDR Container Fields

Table 3-67 lists the FCT_ExchangeRate EDR container fields.

Table 3-67 FCT_ExchangeRate EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
CREATION_TIMESTAMP HEADER.CREATION_TIMESTAMP	Date	Read	Contains the EDR creation time stamp. This field is used if the exchange rate time (Home or Billing) is set to the file date.
BDR_CHARGING_START_TIMESTAMP DETAIL.CHARGING_START_TIMESTAMP	Date	Read	Contains the charging time stamp. This field is used if the exchange rate time (Home or Billing) is set to the CDR date.
ASS_CBD DETAIL.ASS_CBD	Block-Index	Read	Block used for iteration over all associated charge breakdown records.
ASS_CBD_CHARGE_PACKET DETAIL.ASS_CBD.CP	Block-Index	Read	Block used for iteration over all charge packages.
ASS_CBD_RECORD_TYPE DETAIL.ASS_CBD.RECORD_TYPE	Integer	Read	Contains the record type. The billing and/or home currency is calculated for these record types: • 981 • 982 • 983 • 984 • 990 • 991 For record type 980, only the home currency is calculated.
CHARGE_CURRENCY_TYPE DETAIL.ASS_CBD.CP.CHARGED_CURRENCY_TYPE	String	Read/Write	Contains the currency type. The type **R** is read and the types **B** and **H** are calculated if specified.
CHARGED_AMOUNT_VALUE DETAIL.ASS_CBD.CP.CHARGED_AMOUNT_VALUE	Integer	Read/Write	Contains the charge amount that needs to be converted.
CHARGED_AMOUNT_CURRENCY DETAIL.ASS_CBD.CP.CHARGED_AMOUNT_CURRENCY	String	Read/Write	Contains the amount in the charge currency for calculation.
INTERN_BILLING_CURRENCY DETAIL.ASS_CBD.CP.INTERN_BILLING_CURRENCY	String	Read	Contains the billing currency which is calculated and added with a charge package.

Table 3-67 (Cont.) FCT_ExchangeRate EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
INTERN_HOME_CURRENCY DETAIL.ASS_CBD.CP.INTERN_HOME_CURRENCY	String	Read	Contains the home currency which is calculated and added with an charge package. If no home currency is found the home currency from the registry is used for calculation.
EXCHANGERATE DETAIL.ASS_CBD.CP.EXCHANGERATE	String	Write	Contains the exchange rate recommended for TAP.
EXCHANGECURRENCY DETAIL.ASS_CBD.CP.EXCHANGECURRENCY	String	Read/Write	Contains the exchange currency recommended for TAP.
CHARGED_AMOUNT_VALUE_ORIG DETAIL.ASS_CBD.CP.CHARGED_AMOUNT_VALUE_ORIG	Decimal	Write	Contains the charge amount in rating currency.
RESOURCE DETAIL.ASS_CBD.CP.RESOURCE	String	Write	Contains the balance element name
RESOURCE_ID DETAIL.ASS_CBD.CP.RESOURCE_ID	Integer	Read/Write	Contains the balance element ID.
RESOURCE_ID_ORIG DETAIL.ASS_CBD.CP.RESOURCE_ID_ORIG	Integer	Read	Contains the rating balance element ID.
DISCOUNT_PACKET DETAIL.ASS_CBD.DP	Block	Read	Contains the discount related information for the event.
DISCOUNT_PACKET_GRANTED_AMOUNT DETAIL.ASS_CBD.DP.GRANTED_AMOUNT	Decimal	Read/Write	Contains the discounted amount.
DISCOUNT_PACKET_GRANTED_AMOUNT_ORIG DETAIL.ASS_CBD.DP.GRANTED_AMOUNT_ORIG	Decimal	Write	Contains the discounted amount in the initial rated currency.
DISCOUNT_PACKET_INTERN_SRC_PACKET_INDEX DETAIL.ASS_CBD.DP.INTERN_SRC_PACKET_INDEX	Integer	Read	Contains the charge packet number for which this discount is given.
DISCOUNT_PACKET_RESOURCE_ID DETAIL.ASS_CBD.DP.RESOURCE_ID	Integer	Read/Write	Contains the balance element ID.
DISCOUNT_PACKET_RESOURCE_ID_ORIG DETAIL.ASS_CBD.DP.RESOURCE_ID_ORIG	Integer	Write	Contains the currency for which discount has been given originally.
ASS_CBD_RECORD_NUMBER DETAIL.ASS_CBD.RECORD_NUMBER	Integer	Read	Contains the block creation number. These numbers (980, 981, 982, 983, 984, 990, 991) are only considered for exchange rate. Any other number of ASS_CBD will be ignored.

Table 3-67 (Cont.) FCT_ExchangeRate EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
ASS_CBD_CHARGE_PACKET_TAX_PACKET DETAIL.ASS_CBD.TP	Block	Read	Contains tax related information for the event.
CHARGED_INFO_ID DETAIL.ASS_CBD.TP.RELATED_CHARGE_INFO_ID	Integer	Read	Contains charge packet index corresponding to the tax.
TP_TAX_VALUE DETAIL.ASS_CBD.TP.TAX_VALUE	Decimal	Read/Write	Contains the tax amount
TP_TAX_VALUE_ORIG DETAIL.ASS_CBD.TP.TAX_VALUE_ORIG	Decimal	Write	Contains the tax amount in the originally rated currency.
CUST_A_CURRENCY DETAIL.CUST_A.CURRENCY	String	Read	Contains the billing currency of the customer.

FCT_Filter_Set

The FCT_Filter_Set module determines whether an event data record (EDR) is eligible for the system charge offers and system discounts contained in a filter set. If so, it adds those system charge offers and discounts to a customer's list of purchased products.

Dependencies

This module must run *after* the "FCT_Account" module.

FCT_Filter_Set requires the following connections:

- BRM database
- "DAT_Discount" module
- "DAT_AccountBatch" module

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-68 lists the FCT_Filter_Set registry entries.

Table 3-68 FCT_Filter_Set Registry Entries

Entry	Description	Mandatory
AccountDataModule	Specifies the connection to the DAT_AccountBatch module.	Yes
Active	Specifies if the module is active or inactive. • **True** = Active • **False** = Inactive **Note:** When the module is active, it takes over the function of applying system discounts from the FCT_DiscountAnalysis module.	Yes

Table 3-68 (Cont.) FCT_Filter_Set Registry Entries

Entry	Description	Mandatory
DiscountDataModule	Specifies the connection to the DAT_Discount module.	Yes
InfranetConnection	Specifies the connection to the BRM database.	Yes

Sample Registry

```
#------------------------
# Segment FCT
#------------------------
Segment
{
    ModuleName = FCT_Filter_Set
    Module
    {
        Active = True
        DiscountDataModule = ifw.DataPool.DiscountModelDataModule
        AccountDataModule = ifw.DataPool.CustomerData
        InfranetConnection = ifw.DataPool.LoginInfranet
    }
}
```

Semaphore File Entries

Table 3-69 lists the FCT_Filter_Set Semaphore file entries.

Table 3-69 FCT_Filter_Set Semaphore File Entry

Entry	Description
Reload	Reloads data from the database into FCT_Filter_Set.

Sample Semaphore File Entry

```
ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.Segment.Module.Reload{}
```

EDR Container Fields

This module can read any valid EDR fields (defined in **container.dsc**) for matching criteria (EDR field name and value).

> **Note:**
>
> EDR field names read by this module cannot exceed a depth of 6 levels. Exceeding this limit prevents the pipeline from starting and results in an error message.

FCT_FirstUsageNotify

The FCT_FirstUsageNotify module sets the batch rating output stream for charge offers and discounts that start on first usage and sets an error code in the EDR for suspending events that use those charge offers and discounts while their validity periods are being set.

For more information, see "About Suspending EDRs for Products and Discounts that Start on First Usage" in *BRM Setting Up Pipeline Rating and Discounting*.

> **Note:**
>
> To process first-usage events in the real-time rerating pipeline, use the "ISC_FirstProductRealtime" iScript.

Dependencies

Run this module before the FCT_ApplyBalance and FCT_Reject modules.

Requires a connection to DAT_AccountBatch and FCT_Reject.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-70 lists the FCT_FirstUsageNotify registry entries.

Table 3-70 FCT_FirstUsageNotify Registry Entries

Entry	Description	Mandatory
Active	Specifies whether the module is active or inactive. • **True** = Active • **False** = Inactive You can use this entry in a semaphore file.	Yes
CustomerDataModule	Specifies a connection to the DAT_AccountBatch module.	Yes
FirstUsageNotifyOutput	Specifies the output stream for the list of charge offers and discounts used that are set to start on first usage. Default = **FirstUsageNotifyOutput**	Yes
RejectModule	Specifies a connection to the FCT_Reject module. FCT_FirstUsageNotify uses FCT_Reject to determine whether an EDR will be rejected for reasons other than setting the validity. FCT_FirstUsageNotify does not set validity if the EDR will be rejected.	Yes

Sample Registry

```
FirstUsageNotify
{
    ModuleName = FCT_FirstUsageNotify
    Module
```

```
        {
            Active = True
            CustomerDataModule = ifw.DataPool.Account
            RejectModule = ifw.Pipelines.MyPipeline1.Reject
            FirstUsageNotifyOutput = FirstUsageNotifyOutput
        }
    }
```

Semaphore File Entries

Table 3-71 lists the FCT_FirstUsageNotify Semaphore file entry.

Table 3-71 FCT_FirstUsageNotify Semaphore File Entry

Entry	Description
Active	Specifies whether the module is active or inactive. • **True** = Active • **False** = Inactive

Sample Semaphore File Entry

```
ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.FirstUsageNotify.Module.Active = False
```

EDR Container Fields

Table 3-72 lists the FCT_FirstUsageNotify EDR container fields.

Table 3-72 FCT_FirstUsageNotify EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
TRANSACTION_ID INTERNAL.TRANSACTION_ID	String	Read	Specifies the transaction ID.
CUST_A_INTERN_PP_INDEX DETAIL.CUST_A.INTERN_FOUND_PP_INDEX	String	Read	Contains an index of the customer's purchased charge offers that were used for rating.
CUST_A_ACCOUNT_PARENT_ID DETAIL.CUST_A.ACCOUNT_PARENT_ID	String	Read	Specifies the customer account POID.
CUST_A_PRODUCT_ID DETAIL.CUST_A.PRODUCT.OFFERING_POID	String	Read	Specifies the POID of the account's charge offer used to rate the event.
CUST_A_PRODUCT_FIRST_USAGE_INDICATOR DETAIL.CUST_A.PRODUCT.FIRST_USAGE_INDICATOR	String	Read	Specifies whether the charge offer is configured to start when first used and the first-usage validity period status.
FU_DISCOUNT_OBJECTS DETAIL.ASS_CBD.FU_DISCOUNT_OBJECTS	String	Read	Specifies the account's first-usage discounts that were applied to the event.

Table 3-72 (Cont.) FCT_FirstUsageNotify EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
RECYCLE_KEY DETAIL.ASS_SUSPENSE_EXT.RECYCLE_KEY	String	Write	Specifies the recycle key. If the charge offer or discount starts on first usage and its validity period is not set, this field is set to **FirstUsageValidity**.
UTC_TIME_OFFSET DETAIL.UTC_TIME_OFFSET	String	Read	Specifies the difference between local time and UTC time.
ASSOCIATED_INFRANET_BILLING DETAIL.ASS_PIN	String	Write	The Associated BRM Billing record.
FIRST_USAGE_SET_VALIDITY DETAIL.ASS_PIN.FIRST_USAGE	String	Write	The First-usage data block.
FIRST_USAGE_ACCOUNT_POID DETAIL.ASS_PIN.FIRST_USAGE.ACCOUNT_POID_STR	String	Write	Specifies the customer account POID.
FIRST_USAGE_OFFERING_POID DETAIL.ASS_PIN.FIRST_USAGE.OFFERING_POID_STR	String	Write	Specifies the POID of the account's charge offer or discount used to rate or discount the event.
FIRST_USAGE_START_TIME DETAIL.ASS_PIN.FIRST_USAGE.START_T	String	Write	Specifies the validity period start time, which is based on the event start time.
FU_UTC_TIME_OFFSET DETAIL.ASS_PIN.FIRST_USAGE.UTC_TIME_OFFSET	String	Read	Specifies the difference between first-usage start time and UTC time.
CHARGING_START_TIMESTAMP DETAIL.CHARGING_START_TIMESTAMP	String	Read	Specifies the EDR start timestamp.
CUST_A_PRODUCT_SERVICE_ID DETAIL.CUST_A.PRODUCT.SERVICE_ID	String	Read	Specifies the POID of the service object
CUST_A_PRODUCT_SERVICE_TYPE DETAIL.CUST_A.PRODUCT.SERVICE_TYPE	String	Read	Specifies the POID type of the service object
CUST_A_PRODUCT_SERVICE_TYPE DETAIL.ASS_PIN.FIRST_USAGE.SERVICE_POID_STR	String	Read	Specifies the POID of the service object which has FU charge offer uninitialized

FCT_GlobalRating

The FCT_GlobalRating module rates all EDRs with a default set of charges. See "About Global Rating" in *BRM Setting Up Pipeline Rating and Discounting*.

Dependencies

This module must run after FCT_Account.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-73 lists the FCT_GlobalRating registry entries.

Table 3-73 FCT_GlobalRating Registry Entries

Entry	Description	Mandatory
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive You can use this entry in a semaphore file.	Yes
EdrRatePlans	Specifies a set of charges that are used for rating. You can use this entry in a semaphore file.	Yes

Sample Registry

```
GlobalRating
{
  ModuleName = FCT_GlobalRating
  Module
  {
    Active = True
    EdrRatePlans
    {
      RatePlan_1
      RatePlan_2
      RatePlan_3
    }
  }
}
```

Semaphore File Entries

Table 3-74 lists the FCT_GlobalRating Semaphore file entries.

Table 3-74 FCT_GlobalRating Semaphore File Entries

Entry	Description
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive
EdrRatePlans	Specifies a set of charges that are used for rating.

Sample Semaphore File Entry

```
ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.GlobalRating.Module.Active = True
```

EDR Container Fields

Table 3-75 lists the FCT_GlobalRating EDR container fields.

Table 3-75 FCT_GlobalRating EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
ASS_CBD DETAIL.ASS_CBD	Block	Create	Block to hold the rating data.
ASS_CBD_CHARGE_PACKET DETAIL.ASS_CBD.CP	Block	Create	Block created for each **EdrRatePlans** entry.
CHARGING_START_TIMESTAMP DETAIL.CHARGING_START_TIMESTAMP	Date	Read	Contains the EDR start time stamp. This value is used in the charge packet.
INTERN_A_NUMBER_ZONE DETAIL.INTERN_A_NUMBER_ZONE	String	Read	Contains the A number for zoning. This value is used in the charge packet.
INTERN_B_NUMBER_ZONE DETAIL.INTERN_B_NUMBER_ZONE	String	Read	Contains the B number for zoning. This value is used in the charge packet.
ASS_CBD_RECORD_TYPE DETAIL.ASS_CBD.RECORD_TYPE	String	Write	Contains the record type. This value is always set to **980**.
CHARGE_TYPE DETAIL.ASS_CBD.CP.CHARGE_TYPE	String	Write	Contains the charge type. This value is always set to **N**.
ASS_CBD_CHARGING_START_TIMESTAMP DETAIL.ASS_CBD.CP.CHARGING_START_TIMESTAMP	Date	Write	Contains the charging start timestamp from the DETAIL field.
RATEPLAN_CODE DETAIL.ASS_CBD.CP.RATEPLAN_CODE	String	Write	Contains the charge code that is used for zoning and rating.
INTERN_ORIGIN_NUM_ZONE DETAIL.ASS_CBD.CP.INTERN_ORIGIN_NUM_ZONE	String	Write	Contains the A number zoning information used by the FCT_PreRating module. See "FCT_PreRating".
INTERN_DESTIN_NUM_ZONE DETAIL.ASS_CBD.CP.INTERN_DESTIN_NUM_ZONE	String	Write	Contains the B number zoning information used by the FCT_PreRating module. See "FCT_PreRating".

FCT_IRules

The FCT_IRules module evaluates iRules. You use iRules to perform functions such as mapping EDR data fields and splitting EDR containers to different output streams. You group rules together in a rule set.

You can store rules in the database or in an ASCII file.

> **Note:**
>
> FCT_IRules cannot read files written in XML format. You can, however, use the **irules2db.pl** script to load iRules written in XML into the database.

Dependencies

If the data is read from the database, this module requires a connection to the Pipeline Manager database.

This module can run anywhere, depending on the data that is being processed.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-76 lists the FCT_IRules registry entries.

Table 3-76 FCT_IRules Registry Entries

Entry	Description	Mandatory
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive You can use this entry in a semaphore file.	Yes
DataConnection	Specifies the connection to the Pipeline Manager database if rules are stored in the database.	Yes
Descriptions	Specifies the rule set descriptions. The rule sets are evaluated in the specified order. You can use this entry in a semaphore file.	No
Rules	Specifies the rule sets that the module evaluates. You can use this entry in a semaphore file.	Yes, when using the database interface. No, when using a file interface.
Source	Specifies the source of the rules: • **File** • **Database**	Yes

Sample Registry Entry for the Database Interface

```
IRules
{
    ModuleName = FCT_IRules
    Module
    {
        Active = TRUE
        Source = Database
        DataConnection = ifw.DataPool.DataConnection
        Rules
        {
            CIBER_VAL
        }
    }
}
```

Sample Registry Entry for the File Interface

```
Router
{
    ModuleName = FCT_IRules
    Module
    {
        Active = TRUE
        Source = File
        Rules
        {
        }
        Descriptions
        {
            ServiceRequestRouter = ./iScriptLib/AAA/IRL_Router.irl
        }
    }
}
```

Semaphore File Entries

Table 3-77 lists the FCT_IRules Semaphore file entries.

Table 3-77 FCT_IRules Semaphore File Entries

Entry	Description
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive
Descriptions	Specifies the rule set descriptions. The rule sets are evaluated in the specified order.
Reload	Reloads rules from the database or an ASCII file.
Rules	Specifies the rule sets that the module evaluates.

Sample Semaphore File Entry

```
ifw.Pipelines.PRE_RECYCLE.Functions.PreProcessing.FunctionPool.PipelineSplit.Module.Reload {}
```

EDR Container Fields

The EDR container fields that are accessed by FCT_IRule are those that you specify in the rules.

Database Interface

FCT_IRules uses the following database tables to store the generic rules:

- **IFW_RULESET**. Specifies a rule set for linking a related set of rules.
- **IFW_RULESETLIST**. Links a rule set with its rules. Each rule has a rank that specifies the order in which it is evaluated.
- **IFW_RULE**. Stores the iRules.
- **IFW_RULEITEM**. Contains the conditions, the result and the rank for a rule item.

> **Note:**
>
> For information on compare patterns used in database values, see "About Using Regular Expressions when Specifying the Data to Extract" in *BRM Setting Up Pipeline Rating and Discounting*.

File Interface

You can store iRules in an ASCII file by using a syntax that is similar to XML:

- Each tag must start on a separate line.
- Blank lines are allowed.
- Comment lines must start with a pound symbol: #

The rules and rule items are ranked by their order in the file; the first having the highest rank.

Loading Rule Sets from the Database

You can load rule sets from the Pipeline Manager database. iRule components are stored in the following tables:

- **IFW_RULESET**
- **IFW_RULESETLIST**
- **IFW_RULE**
- **IFW_RULEITEM**

Loading Rule Sets from an ASCII File

The following example file shows the syntax of a rule set:

```
##############################################################################
# Example ruleset file
##############################################################################
```

```
<RULESET>
#==============================================================================
# The first rule of this ruleset
#==============================================================================
<RULE MapRule1>
<INIT_SCRIPT>
String code = edrString(DETAIL.SERVICE_CODE) + "_GK2";
</INIT_SCRIPT>
#------------------------------------------------------------------------------
# The first ruleitem of this rule
#------------------------------------------------------------------------------
<RULEITEM discount>
<CONDITION>
/* The text between <CONDITION> and </CONDITON> is directly passed to the
* the interpreter. Lines with # are no comments here!
*/
edrString(DETAIL.RECORD_TYPE ) =~ "02*"; // this is a pattern compare rule
edrLong(DETAIL.QUANTITY) > 30; // this is a normal condition
</CONDITION>
<RESULT>
/* This iScript is processed if the conditions specified in the <CONDITION>
* tag before are matching the current edr container */
edrDecimal(DETAIL.CHARGED_AMOUNT) = 2.50; // set the amount to DM 2.50
</RESULT>
</RULEITEM>
#------------------------------------------------------------------------------
# Here can be specified some more RULEITEMS
#------------------------------------------------------------------------------
</RULE>
#==============================================================================
# The second rule of this ruleset
#==============================================================================
<RULE MapRule2>
#------------------------------------------------------------------------------
# Put the ruleitems for MapRule2 here
#------------------------------------------------------------------------------
</RULE>
</RULESET>
```

FCT_IScript

The FCT_IScript module runs iScripts. The scripts are run in the order specified in the registry.

Dependencies

If the iScripts are stored in the database, this module requires a connection to the Pipeline Manager database.

This module can run anywhere, depending on the data that is being processed.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-78 lists the FCT_IScript registry entries.

Table 3-78 FCT_IScript Registry Entries

Entry	Description	Mandatory
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive You can use this entry in a semaphore file.	Yes
DataConnection	Specifies a connection to the Pipeline Manager database.	Yes, if the module gets data from the database.
Scripts	Specifies the scripts to run. • If the scripts are stored in the database, each value in the entry specifies the SCRIPTCODE field in the INT_ISCRIPT database table. • If the scripts are stored in a file, each section in the file must have a registry entry FileName = *file*. **Note:** The section for each script can store a number of entries that are passed as global constants to the interpreter.	Yes
Scripts.*ScriptName*	The name of the script as used in the registry.	N/A
Scripts.*ScriptName*.**FileName**	The name and path of the script.	N/A
Scripts.*ScriptName.Registry_Parameter*	Registry parameters used in the iScript. The example below uses the GL_Code parameter: GL_CODE = 1514	N/A
Source	Specifies the source of the iScripts. • **File** • **Database**	Yes

Sample Registry for the File Interface

```
ConsolidatedCP
{
   ModuleName = FCT_IScript
   Module
   {
      Active = True
      Source = File
      Scripts
      {
         ConsolidatedCPIScript
         {
            FileName = ./iScriptLib/iScriptLib_Roaming/ISC_ConsolidatedCP.isc
            GL_CODE = 1514
         }
      }
   }
}
```

Sample Registry for the Database Interface

```
IScript
{
```

```
  ModuleName = FCT_IScript
  Module
  {
    Active = True
    Source = Database
    DataConnection = ifw.DataPool.Login
    Scripts
    {
      Mapping
      {
        # can be used as reg.Arg1 in the IScript
        Arg1 = any_argument_value
        }
        Specials
        {
      }
    }
  }
}
```

Semaphore File Entries

Table 3-79 lists the FCT_IScript Semaphore file entries.

Table 3-79 FCT_IScript Semaphore File Entries

Entry	Description
Active	Activates or deactivates the module. **TRUE** = Activate. **FALSE** = Deactivate.
Reload	Reloads iScripts from the database or an ASCII file.
Scripts	Section with scripts to run. In case of Source = DATABASE each value in the section specifies the SCRIPTCODE of a script in database table INT_ISCRIPT. In case of Source = FILE each section within the section must have a registry entry FileName = *file*.

Sample Semaphore File Entry

```
ifw.Pipelines.ALL_RATE.Functions.Standard.FunctionPool.IScript.Module.Reload {}
```

Database Tables

The FCT_IScript module uses the IFW_ISCRIPT database table to store iScripts.

File Interface

The iScript source code is stored in a simple ASCII file. The file is loaded and compiled at startup.

FCT_ItemAssign

The FCT_ItemAssign module assigns bill items to events.

Dependencies

FCT_ItemAssign requires a connection to the DAT_ItemAssign module.

Must run after the FCT_Account, rating, and discounting modules and before the FCT_BillingRecord module.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-80 lists the FCT_ItemAssign registry entries.

Table 3-80 FCT_ItemAssign Registry Entries

Entry	Description	Mandatory
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive	Yes
DataModule	Specifies the connection to the DAT_ItemAssign module.	Yes

Sample Registry

```
Item Assignment
{
  ModuleName = FCT_ItemAssign
  Module
  {
    Active = True
    DataModule = ifw.DataPool.ItemAssignDataModule
  }
}
```

EDR Container Fields

The FCT_ItemAssign module uses the EDR container fields listed in Table 3-81:

Table 3-81 FCT_ItemAssign EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
DETAIL.ITEM_TAG	String	Read	Contains the item tag.
DETAIL.CUST_A. PRODUCT.SERVICE_USED_ITEM_POID	String	Write	Contains the item POID for the event.
DETAIL.CUST_A.INTERN_FOUND_PP_INDEX	Integer	Read	Contains the charge offer ID of the charge offer used for rating.
DETAIL.CUST_A.ACCOUNT_PARENT_ID	String	Read	Contains the customer's account number.

Table 3-81 (Cont.) FCT_ItemAssign EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
DETAIL.CUST_A.PRODUCT.SERVICE_ID	String	Read	Contains the ID of the charge offer.
DETAIL.UTC_TIME_OFFSET	String	Read	Contains the UTC time offset that normalizes the charging start timestamp to the UTC time zone. All validity timestamps in the BRM customer data are stored in normalized UTC time. The format is +/- HHMM.
DETAIL.CHARGING_START_TIMESTAMP	Date	Read	Contains the start timestamp of the event. The time zone information for this timestamp is stored in the BDR_UTC_TIME_OFFSET field.

FCT_MainRating

The FCT_MainRating module performs the main rating functionality in a pipeline. See "About Main Rating" in *BRM Setting Up Pipeline Rating and Discounting*.

Dependencies

The FCT_MainRating module requires a connection to the following data modules:

- DAT_Calendar
- DAT_Currency
- DAT_TimeModel
- DAT_Rateplan
- DAT_PriceModel
- DAT_ModelSelector

This module must run after at least one of the following modules:

- FCT_CarrierIcRating
- FCT_CustomerRating
- FCT_GlobalRating

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-82 lists the FCT_MainRating registry entries.

Table 3-82 FCT_MainRating Registry Entries

Entry	Description	Mandatory
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive You can use this entry in a semaphore file.	Yes
CalendarDataModule	Specifies the connection to the DAT_Calendar module.	Yes
CurrencyDataModule	Specifies the connection to the DAT_Currency module.	No
ModelSelectorDataModule	Specifies the connection to the DAT_ModelSelector module.	Yes
PriceDataModule	Specifies the connection to the DAT_PriceModel module.	Yes
RateplanDataModule	Specifies the connection to the DAT_Rateplan module.	Yes
TimeDataModule	Specifies the connection to the DAT_TimeModel module.	Yes

Sample Registry

```
MainRating
{
  ModuleStart = FCT_MainRating
  Module
  {
    Active = True
    RateplanDataModule = RateplanDataModule
    CalendarDataModule = CalendarDataModule
    TimeDataModule = TimeDataModule
    PriceDataModule = PriceDataModule
    CurrencyDataModule = ifw.DataPool.CurrencyDataModule
    ModelSelectorDataModule = Model selector module
  }
}
```

Semaphore File Entries

Table 3-83 lists the FCT_MainRating Semaphore file entry.

Table 3-83 FCT_MainRating Semaphore File Entry

Entry	Description
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive

Sample Semaphore File Entry

```
ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.MainRating.Module.Active = False
```

EDR Container Fields

Table 3-84 lists the FCT_MainRating EDR container fields.

Table 3-84 FCT_MainRating EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
ASS_CBD DETAIL.ASS_CBD	Block	Read	Data block for rate data.
ASS_CBD_CHARGE_PACKET DETAIL.ASS_CBD.CP	Block	Read	Charge packet for rate data.
BDR_CHARGING_START_TIMESTAMP DETAIL.CHARGING_START_TIMESTAMP	Date	Read	Contains the charging time stamp.
WHOLESALE_CHARGED_AMOUNT_VALUE DETAIL.WHOLESALE_CHARGED_AMOUNT_VALUE	Decimal	Read	Contains the wholesale charge amount value.
ASS_CBD_RECORD_NUMBER DETAIL.ASS_CBD.RECORD_NUMBER	Integer	Read	Contains the record number.
ASS_CBD_IMPACT_CATEGORY DETAIL.ASS_CBD.CP.IMPACT_CATEGORY	String	Read	Contains the impact category.
RATEPLAN_CODE DETAIL.ASS_CBD.CP.RATEPLAN_CODE	String	Read/Write	Contains the charge code.
RATEPLAN_TYPE DETAIL.ASS_CBD.CP.RATEPLAN_TYPE	String	Write	Contains the charge type.
TIMEMODEL_CODE DETAIL.ASS_CBD.CP.TIMEMODEL_CODE	String	Write	Contains the time model code.
TIMEZONE_CODE DETAIL.ASS_CBD.CP.TIMEZONE_CODE	String	Write	Contains the time zone code.
DAY_CODE DETAIL.ASS_CBD.CP.DAY_CODE	String	Write	Contains the special day code.
TIME_INTERVAL_CODE DETAIL.ASS_CBD.CP.TIME_INTERVAL_CODE	String	Write	Contains the time interval code.
PRICEMODEL_CODE DETAIL.ASS_CBD.CP.PRICEMODEL_CODE	String	Write	Contains the pricing code.
PRICEMODEL_TYPE DETAIL.ASS_CBD.CP.PRICEMODEL_TYPE	String	Write	Contains the pricing type.
SERVICE_CODE_USED DETAIL.ASS_CBD.CP.SERVICE_CODE_USED	String	Write	Contains the service code.
SERVICE_CLASS_USED DETAIL.ASS_CBD.CP.SERVICE_CLASS_USED	String	Write	Contains the service class.
CHARGED_AMOUNT_VALUE DETAIL.ASS_CBD.CP.CHARGED_AMOUNT_VALUE	Decimal	Write	Contains the charge amount value for the event.
CHARGED_AMOUNT_CURRENCY DETAIL.ASS_CBD.CP.CHARGED_AMOUNT_CURRENCY	String	Write	Contains the currency amount.
CHARGED_CURRENCY_TYPE DETAIL.ASS_CBD.CP.CHARGED_CURRENCY_TYPE	String	Write	Contains the currency type.
CHARGED_TAX_TREATMENT DETAIL.ASS_CBD.CP.CHARGED_TAX_TREATMENT	String	Write	Contains the tax treatment.

Table 3-84 (Cont.) FCT_MainRating EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
ASS_CBD_CHARGING_START_TIMESTAMP DETAIL.ASS_CBD.CP.CHARGING_START_TIMESTAMP	Date	Write	Contains the charging time stamp.
ROUNDED_QUANTITY_VALUE DETAIL.ASS_CBD.CP.ROUNDED_QUANTITY_VALUE	Decimal	Write	Contains the rounded quantity value.
ROUNDED_QUANTITY_UOM DETAIL.ASS_CBD.CP.ROUNDED_QUANTITY_UOM	String	Write	Contains the rounded quantity UoM.
USAGE_GL_ACCOUNT_CODE DETAIL.ASS_CBD.CP.USAGE_GL_ACCOUNT_CODE	String	Write	Contains the G/L code.
REVENUE_GROUP_CODE DETAIL.ASS_CBD.CP.REVENUE_GROUP_CODE	String	Write	Contains the revenue group.
RUMGROUP DETAIL.ASS_CBD.CP.RUMGROUP	String	Write	Contains the RUM group.
RUM DETAIL.ASS_CBD.CP.RUM	String	Write	Contains the RUM.
RESSOURCE DETAIL.ASS_CBD.CP.RESOURCE	String	Write	Contains the balance element.
INTERN_DISCOUNT_MODEL DETAIL.ASS_CBD.CP.INTERN_DISCOUNT_MODEL	Integer	Write	Contains the discount.
INTERN_RATEPLAN DETAIL.ASS_CBD.CP.INTERN_RATEPLAN	String	Write	Contains the internal charge.
INTERN_RATEPLAN_VERSION DETAIL.ASS_CBD.CP.INTERN_RATEPLAN_VERSION	Integer	Write	Contains the charge version.
INTERN_FIX_COST DETAIL.ASS_CBD.CP.INTERN_FIX_COST	Decimal	Read	Contains the fixed cost amount.
INTERN_PRICE_MDL_STEP_INFO DETAIL.ASS_CBD.CP.INTERN_PRICE_MDL_STEP_INFO	String	Write	Contains information about the pricing steps used to calculate the charge in the charge packet.
DETAIL.CUST_A.PRODUCT.RATEPLAN_CODE	String	Read	Contains the charge code of the charge offer to match with the charge in the charge breakdown record.
DETAIL.CUST_A.INTERN_RATING_PRODUCTS	String	Read	Contains the rating charge offer indexes. This is a comma-separated list of all rating charge offers' indexes associated with the same service and event, and their priorities.

Table 3-84 (Cont.) FCT_MainRating EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
DETAIL.CUST_A.INTERN_FOUND_PP_INDEX	String	Write	Contains the index of the highest priority rating charge offer. This is the charge offer with the highest rate priority for an event.
DETAIL.CUST_A.LEAST_COST	Integer	Read	Indicates whether to use least cost rating for an EDR. **1** turns it off; **2** turns it on.
DETAIL.CUST_A.PROMOTIONAL_SAVING	Integer	Read	Indicates whether to use promotional savings for an EDR. **1** turns it off; **2** turns it on.
DETAIL.CUST_A.PROMOTION	Integer	Read	Indicates whether to use overlay promotions for an EDR. **1** turns it off; **2** turns it on.

FCT_MainZoning

The FCT_MainZoning module performs zoning for multi-segment zoning.

Dependencies

Requires a connection to the DAT_Zone module.

This module must run after the FCT_SegZoneNoCust module.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-85 lists the FCT_MainZoning registry entries.

Table 3-85 FCT_MainZoning Registry Entries

Entry	Description	Mandatory
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive You can use this entry in a semaphore file.	Yes
ZoneDataModule	Specifies the connection to the DAT_Zone module.	Yes

Sample Registry

```
MainZoning
{
  ModuleName = FCT_MainZoning
  Module
  {
    Active = True
    ZoneDataModule = integRate.DataPool.ZoneDataModule
  }
}
```

Semaphore File Entries

Table 3-86 lists the FCT_MainZoning Semaphore file entries.

Table 3-86 FCT_MainZoning Semaphore File Entries

Entry	Description
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive

Sample Semaphore File Entry

```
ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.MainZoning.Module.Active = False
```

EDR Container Fields

Table 3-87 lists the FCT_MainZoning EDR container fields.

Table 3-87 FCT_MainZoning EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
BDR_CHARGING_START_TIMESTAMP DETAIL.CHARGING_START_TIMESTAMP	Date	Read	Contains the charging time stamp.
INTERN_SERVICE_CODE DETAIL.INTERN_SERVICE_CODE	String	Read	Contains the internal service code.
INTERN_A_NUMBER_ZONE DETAIL.INTERN_A_NUMBER_ZONE	String	Read	Contains the zone for the A number.
INTERN_B_NUMBER_ZONE DETAIL.INTERN_B_NUMBER_ZONE	String	Read	Contains the zone for the B number.
ASS_ZBD_ZONE_DESCRIPTION DETAIL.ASS_ZBD.ZP.ZONE_DESCRIPTION	String	Write	Contains the zone description for displaying on invoices.
ASS_ZBD_RECORD_NUMBER DETAIL.ASS_ZBD.RECORD_NUMBER	Integer	Read	Contains the record number.

Table 3-87 (Cont.) FCT_MainZoning EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
ASS_ZBD_ZONE_RESULT_VALUE_WS DETAIL.ASS_ZBD.ZP.ZONE_RESULT_VALUE_WS	String	Write	Contains the wholesale zone.
ASS_ZBD_ZONE_RESULT_VALUE_RT DETAIL.ASS_ZBD.ZP.ZONE_RESULT_VALUE_RT	String	Write	Contains the retail zone.
ASS_ZBD_ZONEMODEL_CODE DETAIL.ASS_ZBD.ZP.ZONEMODEL_CODE	String	Write	Contains the zone model code.
ASS_ZBD_INTERN_ZONE_MODEL DETAIL.ASS_ZBD.ZP.INTERN_ZONE_MODEL	Integer	Read	Contains the zone model.
ASS_ZBD_INTERN_APN_GROUP DETAIL.ASS_ZBD.ZP.INTERN_APN_GROUP	String	Write	Contains the APN group.
ASS_ZBD_ZONE_ENTRY_NAME DETAIL.ASS_ZBD.ZP.ZONE_ENTRY_NAME	String	Write	Contains the destination description for this combination of service code and impact category.

FCT_NOSP

The FCT_NOSP module maps network source and destination to new values. See "Identifying the Network Operator/Service Provider" in *BRM Setting Up Pipeline Rating and Discounting*.

Dependencies

Requires a connection to the DAT_NOSP module.

This module must be run before segment rating is performed.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-88 lists the FCT_NOSP registry entries.

Table 3-88 FCT_NOSP Registry Entries

Entry	Description	Mandatory
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive You can use this entry in a semaphore file.	Yes
DataModule	Specifies the connection to the DAT_NOSP module. See "DAT_NOSP".	Yes
MapGroup	Specifies the map group that the NOSP mappings belong to. You can use this entry in a semaphore file.	Yes

Sample Registry

```
Zoning
{
  ModuleName = FCT_NOSP
  Module
  {
    Active = True
    DataModule = ifw.DataPool.NospData
    MapGroup = MOBILE
  }
}
```

Semaphore File Entries

Table 3-89 lists the FCT_NOSP Semaphore file entries.

Table 3-89 FCT_NOSP Semaphore File Entries

Entry	Description
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive
MapGroup	Specifies the map group.

Sample Semaphore File Entry

```
ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.NOSP.Module.Active = False
```

EDR Container Fields

Table 3-90 lists the FCT_NOSP EDR container fields.

Table 3-90 FCT_NOSP EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
SOURCE_NETWORK DETAIL.SOURCE_NETWORK	String	Read/Write	Contains the source network.
DESTINATION_NETWORK DETAIL.DESTINATION_NETWORK	String	Read/Write	Contains the destination network.
A_NUMBER DETAIL.A_NUMBER	String	Read	Contains the A number.

FCT_NumberPortability

The FCT_NumberPortability module specifies the new network operator for an existing phone number.

Dependencies

Requires a connection to the DAT_NumberPortability module.

This module must be run before the zoning and rating modules.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-91 lists the FCT_NumberPortability registry entries.

Table 3-91 FCT_NumberPortability Registry Entries

Entry	Description	Mandatory
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive Default = **False**. You can use this entry in a semaphore file.	Yes
DataModule	Specifies the connection to the DAT_NumberPortability module.	Yes
DefaultSourceNetwork	Specifies the default network operator ID of the source network. This ID is used if there is no ID present in the data retrieved from the DAT_NumberPortability module.	Yes
DefaultDestinationNetwork	Specifies the default network operator ID of the destination network. This ID is used if there is no ID present in the data retrieved from the DAT_NumberPortability module.	Yes
OverwriteNetwork	If the **DefaultSourceNetwork** and **DefaultDestinationNetwork** fields are empty, overwrites the source and destination network with the value configured in the DAT_NumberPortability module. Default = **True**.	No
OverwriteNetworkType	If the SOURCE_NETWORK_TYPE and DESTINATION_NETWORK_TYPE fields are empty, overwrites the SOURCE_NETWORK_TYPE and DESTINATION_NETWORK_TYPE fields with the type of network that is populated in the source and destination network. Default = **True**.	No

Sample Registry

```
NumberPortability
{
  ModuleName = FCT_NumberPortability
  Module
  {
    Active = True
    DataModule = integrate.DataPool.NPortData
    DefaultSourceNetwork = D030
    DefaultDestinationNetwork = D017
  }
}
```

Semaphore File Entries

Table 3-92 lists the FCT_NumberPortability Semaphore file entry.

Table 3-92 FCT_NumberPortability Semaphore File Entry

Entry	Description
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive

Sample Semaphore File Entry

```
ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.NumberPortability.
Module.Active = False
```

EDR Container Fields

Table 3-93 lists the FCT_NumberPortability EDR container fields.

Table 3-93 FCT_NumberPortability EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
A_NUMBER DETAIL.A_NUMBER	String	Read	Specifies the event originator.
B_NUMBER DETAIL.B_NUMBER	String	Read	Specifies the event receiver.
BDR_CHARGING_START_TIMESTAMP DETAIL.CHARGING_START_TIMESTAMP	Date	Read	Specifies the charging timestamp. The format is: *YYYYMMDDhhmmss*
IGNORE_NP DETAIL.IGNORE_NP	Integer	Read/ Write	Specifies whether FCT_NumberPortability should look for network operator IDs for A and B number. Default is **0** (cleared).
SOURCE_NETWORK DETAIL.SOURCE_NETWORK	String	Write	Specifies the source network. This can either be the PLMN ID or any logical operator code.
SOURCE_NETWORK_TYPE DETAIL.SOURCE_NETWORK_TYPE	String	Write	Specifies the source network type, for example GSM 900.
DESTINATION_NETWORK DETAIL.DESTINATION_NETWORK	String	Write	Specifies the network to which an event is routed.

Table 3-93 (Cont.) FCT_NumberPortability EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
DESTINATION_NETWORK_TYPE DETAIL.DESTINATION_NETWORK_TYPE	String	Write	Specifies the destination network type, for example GSM 900.

FCT_Opcode

The FCT_Opcode module uses the DAT_ConnectionPool module to connect to the CM and calls the appropriate opcode for the request.

Dependencies

The FCT_Opcode module requires a connection to the "DAT_ConnectionPool" module.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-94 lists the FCT_Opcode registry entries.

Table 3-94 FCT_Opcode Registry Entries

Entry	Description	Mandatory
Active	Specifies whether the module is active or inactive **True** = Active (default) **False** = Inactive	Yes
Retries	Specifies the number of times to try the request on the CM Default = **2**	Yes
Logging	Logs each opcode called from the processing pipeline Default = **False**	Yes
ConnectionPoolDataModule	Specifies a connection to the DAT_ConnectionPool module.	Yes

Sample Registry

```
EdrOpcodeCall
{
  ModuleName = FCT_Opcode
  Module
  {
    Active = True
    Retries = 2
    Logging = True
    ConnectionPoolDataModule = ifw.DataPool.CMConnectionPool.Module
  }
}
```

EDR Container Fields

FCT_Opcode uses the EDR container fields listed in Table 3-95:

Table 3-95 FCT_Opcode Container Fields

Alias Field Name Default Field Name	Type	Access	Description
OPCODE_FLAG DETAIL.OPCODE_FLAG	Integer	Read	The flag that specifies the behavior of the opcode
OPCODE_NODE DETAIL.OPCODE_NODE	String	Read	Name of the opcode
PCM_OP_EBUF DETAIL.PCM_OP_EBUF	pin_ebuf_t	Read	Error buffer

FCT_PrefixDesc

The FCT_PrefixDesc module maps phone number prefixes to destination descriptions. See "Creating Call Destination Descriptions" in *BRM Setting Up Pipeline Rating and Discounting*.

Dependencies

Requires a connection to the DAT_PrefixDesc module.

This module can run from anywhere.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-96 lists the FCT_PrefixDesc registry entries.

Table 3-96 FCT_PrefixDesc Registry Entries

Entry	Description	Mandatory
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive You can use this entry in a semaphore file.	Yes
PrefixDataModule	Specifies the connection to the DAT_PrefixDesc module.	Yes

Sample Registry

```
{
  ModuleName = FCT_PrefixDesc
  Module
  {
    Active = True
```

```
        PrefixDataModule = PrefixDescData
    }
}
```

Semaphore File Entries

Table 3-97 lists the FCT_PrefixDesc Semaphore file entry.

Table 3-97 FCT_PrefixDesc Semaphore File Entry

Entry	Description
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive

Sample Semaphore File Entry

```
ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.PrefixDesc.Module.Active = False
```

EDR Container Fields

Table 3-98 lists the FCT_PrefixDesc EDR container fields.

Table 3-98 FCT_PrefixDesc EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
B_NUMBER DETAIL.B_NUMBER	String	Read	Contains the B number.
DESCRIPTION DETAIL.DESCRIPTION	String	Write	Contains the call destination description.

FCT_PreRating

The FCT_PreRating module calculates zones and creates impact category.

Dependencies

Requires a connection to the DAT_Rateplan and the DAT_Zone module.

This module must be run before the FCT_MainRating module.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-99 lists the FCT_PreRating registry entries.

Table 3-99 FCT_PreRating Registry Entries

Entry	Description	Mandatory
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive You can use this entry in a semaphore file.	Yes
RateplanDataModule	Specifies the connection to the DAT_Rateplan module.	Yes
ZoneDataModule	Specifies the connection to the DAT_Zone module.	Yes

Sample Startup Registry

```
PreRating
{
  ModuleStart = FCT_PreRating
  Module
  {
    Active = True
    RateplanDataModule = ifw.DataPool.RateplanDataModule
    ZoneDataModule = ifw.DataPool.ZoneDataModule
  }
}
```

Semaphore File Entries

Table 3-100 lists the FCT_PreRating Semaphore file entry.

Table 3-100 FCT_PreRating Semaphore File Entry

Entry	Description
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive

Sample Semaphore File Entry

```
ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.PreRatingZone.Module.Active = False
```

EDR Container Fields

Table 3-101 lists the FCT_PreRating EDR container fields.

Table 3-101 FCT_PreRating EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
BDR_CHARGING_START_TIMESTAMP DETAIL.CHARGING_START_TIMESTAMP	Date	Read	Contains the charging time stamp. The time stamp is used for calculating the impact category by comparing it to the dates in the VALID_FROM and VALID_TO fields in the **IFW_STANDARD_ZONE** database table.
INTERN_SERVICE_CODE DETAIL.INTERN_SERVICE_CODE	String	Read	Contains the service code. The service code is used for calculating the impact category by comparing it with the value in the SERVICECODE field in the **IFW_STANDARD_ZONE** database table. The module writes the service code to the ASS_CBD_SERVICE_CODE and SERVICE_CLASS_USED fields.
INTERN_SERVICE_CLASS DETAIL.INTERN_SERVICE_CLASS	String	Read	Contains the service class. The module writes the service class to the ASS_CBD_SERVICE_CODE and SERVICE_CLASS_USED fields.
ASS_CBD_RECORD_NUMBER DETAIL.ASS_CBD.RECORD_NUMBER	String	Read	Contains the charge breakdown record number. Charge breakdown records are processed only if the record number = 0 (newly created).
ASS_CBD_SERVICE_CODE DETAIL.ASS_CBD.SERVICE_CODE	String	Write	Contains the service code. Set with the value of the INTERN_SERVICE_CODE field.
ASS_CBD_ZONEMODEL_CODE DETAIL.ASS_CBD.CP.ZONEMODEL_CODE	String	Write	Contains the zone model code. Set with the value of the INTERN_SERVICE_CLASS field.

Table 3-101 (Cont.) FCT_PreRating EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
ASS_CBD_IMPACT_CATEGORY DETAIL.ASS_CBD.CP.IMPACT_CATEGORY	String	Write	Contains the impact category. Set with the zoning results by using the value from either the ZONE_WS or ZONE_RT in the **IFW_STANDARD_ZONE** database table, depending on charge type.
RATEPLAN_CODE DETAIL.ASS_CBD.CP.RATEPLAN_CODE	String	Write	Contains a comma-separated list of charge codes for all rating charge offers. This list arranged by charge offer priority, with the highest priority first and the lowest priority last. Set with the value of the CODE field in the **IFW_RATEPLAN** database table.
RATEPLAN_TYPE DETAIL.ASS_CBD.CP.RATEPLAN_TYPE	String	Write	Contains the charge type. Set with the value of the TYPE field in the **IFW_RATEPLAN** database table.
SERVICE_CODE_USED DETAIL.ASS_CBD.CP.SERVICE_CODE_USED	String	Write	Contains the service code. Set with the value from the INTERN_SERVICE_CODE field.
SERVICE_CLASS_USED DETAIL.ASS_CBD.CP.SERVICE_CLASS_USED	String	Write	Contains the service class. Set with the value from the INTERN_SERVICE_CLAS S field.
INTERN_ORIGIN_NUM_ZONE DETAIL.ASS_CBD.CP.INTERN_ORIGIN_NUM_ZONE	String	Read	Contains the area code for the A number. The area code is used for calculating the impact category by comparing it to ORIGIN_AREACODE field in the **IFW_STANDARD_ZONE** database table.
INTERN_DESTIN_NUM_ZONE DETAIL.ASS_CBD.CP.INTERN_DESTIN_NUM_ZONE	String	Read	Contains the area code for the B number. The area code is used for calculating the impact category by comparing it to DESTIN_AREACODE field in the **IFW_STANDARD_ZONE** database table.

Table 3-101 (Cont.) FCT_PreRating EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
INTERN_RATEPLAN DETAIL.ASS_CBD.CP.INTERN_RATEPLAN	String	Read	Contains the charge. Set with the value from the RATEPLAN field in the **IFW_RATEPLAN** database table.
INTERN_RATEPLAN_VERSION DETAIL.ASS_CBD.CP.INTERN_RATEPLAN_VERSION	Integer	Write	Contains the charge version. Set with the value from the VERSION field in the **IFW_RATEPLAN_VER** database table.
ASS_CBD_INTERN_ZONE_MODEL DETAIL.ASS_CBD.CP.INTERN_ZONE_MODEL	Integer	Write	Contains the zone model. Set with the value from the ZONEMODEL field in the **IFW_RATEPLAN_VER** database table.
ASS_CBD_INTERN_APN_GROUP DETAIL.ASS_CBD.CP.INTERN_APN_GROUP	String	Write	Contains the APN group. Set with the value from the APN_GROUP field in the **IFW_ZONEMODEL** database table.
ASS_CBD_INTERN_GEOMODEL DETAIL.ASS_CBD.CP.INTERN_GEOMODEL	Integer	Write	Contains the geographical model, if the MODELTYPE field in the **IFW_ZONEMODEL** database table is set to **L**. Set with the value from the GEOMODEL field in the **IFW_ZONEMODEL** database table.
ASS_CBD_INTERN_RULESET DETAIL.ASS_CBD.CP.INTERN_RULESET	String	Write	Contains the rule set, if the MODELTYPE field in the **IFW_ZONEMODEL** database table is set to **L**. Set with the value from the RULESET field in the **IFW_GEOGRAPHICAL_MODEL** database table.
ASS_CBD_ZONE_DESCRIPTION DETAIL.ASS_CBD.CP.ZONE_DESCRIPTION	String	Write	Contains the zone description for displaying on invoices.
ASS_ZBD_ZONE_ENTRY_NAME DETAIL.ASS_ZBD.ZP.ZONE_ENTRY_NAME	String	Write	Contains the destination description for displaying on invoices.
RATE_PLAN_NAME DETAIL.CUST_A.PRODUCT.RATEPLAN_NAME	String	Write	Contains the charge name for the purchased charge offer.
INTERN_RATING_PRODUCTS DETAIL.CUST_A.INTERN_RATING_PRODUCTS	String	Write	Contains the indexes of the candidate charge offers that can be used for rating.

Table 3-101 (Cont.) FCT_PreRating EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
INTERN_FOUND_PP_INDEX DETAIL.CUST_A.INTERN_FOUND_PP_INDEX	String	Write	Contains the purchased charge offer index of the charge offer or service used.
CUST_A_LEAST_COST_RATING DETAIL.CUST_A.LEAST_COST	String	Write	Specifies if least cost rating is to be used for rating the EDR.

FCT_PreRecycle

The FCT_PreRecycle module gets the file of rejected EDRs from the reject stream output directory. The module puts the reject EDR file into the pipeline input directory for recycling. It uses the same input folder as the incoming CDR files.

See:

- Configuring Standard Recycling
- Recycling EDRs in Pipeline-Only Systems

Registry Entries

Table 3-102 lists the FCT_PreCycle registry entries.

Table 3-102 FCT_PreCycle Registry Entries

Entry	Description	Mandatory
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive	Yes
RecycleSuffix	Specifies the suffix for the file that contains the EDRs that need recycling. The suffix is automatically appended when the file is moved from the reject directory to the input directory. If it is empty, no suffix is added. Default = **_Recy**	No
RecyFileName	Specifies the file name and path for the file that contains the EDRs that need recycling.	Yes

Sample Registry

```
PreRecycle
{
   ModuleName = FCT_PreRecycle
   Module
   {
      Active = True
      RecycleSuffix = RecycleFile
      RecyFileName = ./recycle.dat
```

```
    }
}
```

Semaphore File Entries

When you update the registry, you must select one of the following entries listed in Table 3-103:

Table 3-103 FCT_PreRecycle Semaphore File Entries

Entry	Description
Recycle	The module runs in real processing mode You can specify a list of files to recycle. If this entry is empty, all files from the reject directory are recycled.
RecycleTest	The module runs in test mode You can specify a list of files to test recycling with. If this entry is empty, all files from the reject directory are recycled.

Sample Semaphore File Entries

Recycle all files:

```
ifw.Pipelines.PRE_RECYCLE.Functions.Processing.FunctionPool.PreRecycle.Module.Recycle {}
```

Recycle only specific files:

```
ifw.Pipelines.PRE_RECYCLE.Functions.Processing.FunctionPool.PreRecycle.Module.Recycle.
File = ./format_a/abc.cdr

ifw.Pipelines.PRE_RECYCLE.Functions.Processing.FunctionPool.PreRecycle.Module.Recycle.
File = ./format_a/xyz.cdr
```

Test recycle all files:

```
ifw.Pipelines.PRE_RECYCLE.Functions.Processing.FunctionPool.PreRecycle.Module.RecycleTest {}
```

Test recycle only specific files:

```
ifw.Pipelines.PRE_RECYCLE.Functions.Processing.FunctionPool.PreRecycle.Module.RecycleTest.
File = ./format_a/abc.cdr

ifw.Pipelines.PRE_RECYCLE.Functions.Processing.FunctionPool.PreRecycle.Module.RecycleTest.
File = ./format_a/xyz.cdr
```

EDR Container Fields

Table 3-104 lists the FCT_PreRecycle EDR container fields.

Table 3-104 FCT_PreRecycle EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
INTERN_PROCESS_STATUS DETAIL.INTERN_PROCESS_STATUS	Integer	Write	Contains the internal process status.

Table 3-104 (Cont.) FCT_PreRecycle EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
TRANSACTION_ID INTERNAL.TRANSACTION_ID	Decimal	Read	Contains the transaction ID.
STREAM_NAME INTERNAL.STREAM_NAME	String	Read	Contains the stream name.
PROCESS_STATUS INTERNAL.PROCESS_STATUS	Integer	Write	Contains the process status.

FCT_PreSuspense

This module is used both by the standard recycling mechanism and by the Suspense Manager service integration component that you purchase separately. Both implementations are described below.

> **Note:**
>
> This module stores the contents of the EDR before any other modules change it. This module must take the original version of an EDR as input, so that it can be recycled after being suspended.

Standard Recycling Implementation

The BRM FCT_PreSuspense module adds suspense-related information to EDRs. It adds the DETAIL.ASS_SUSPENSE_EXT data block to the EDR if that data block does not already exist.

Suspense Manager Implementation - Adding Queryable Fields

When used with Suspense Manager, FCT_PreSuspense configures the queryable fields for EDRs suspended in a specific pipeline. You must enter the table and field names from the ***/ suspended_usage*** object, as well as the corresponding EDR container fields. See "Registry Entries" for syntax and formatting information.

If no **QueryableFields** registry entry is present, the HEADER.QUERYABLE_FIELDS_MAPPING and DETAIL.ASS_SUSPENSE_EXT.QUERYABLE_FIELDS_MAPPING are set to empty strings.

> **Note:**
>
> Each table listed in the FCT_PreSuspense registry must also be configured in the RE Loader **Infranet.properties** file so that RE Loader can load into these tables.

This module adds queryable field mapping information to the HEADER.QUERYABLE_FIELDS_MAPPING field of the EDR.This information is passed to the Suspended Event (SE) Loader to generate control files for loading suspended usage records.

> **Note:**
>
> You add one set of queryable fields representing one **/suspended_usage** subclass *per pipeline*. For example, for a single pipeline that accepts **/suspended_usage/telco/gsm** records, you can pick queryable fields from the **/suspended_usage/telco** and **/suspended_usage/telco/gsm** subclasses. You could not pick queryable fields from **/suspended_usage/telco/gprs**, because it requires a separate pipeline.

FCT_PreSuspense serializes the original EDR container and stores it in DETAIL.ASS_SUSPENSE_EXT.EDR_BUF. It also stores the EDR size in DETAIL.ASS_SUSPENSE_EXT.EDR_SIZE.

When call records are being recycled, this module sets values for:

- HEADER.BATCH_ID, based on value already set by INP_Recycle in pre-recycle pipeline. This ID is appended with information to ensure that it remains unique.
- DETAIL.BATCH_ID with the batch ID value from the header record.

Changing the Way Batch IDs Are Set

You use the **KeepExistingBatchIds** registry entry to determine whether an EDR's batch ID is preserved as it is processed by the pipeline. A value of **False** directs the pipeline to change the batch ID; a value of **True** preserves it.

Set **KeepExistingBatchIds** to **False** (the default value) if the current pipeline is not part of a chain. Also set this entry to **False** if the current pipeline is the first pipeline in a chain of pipelines that includes the FCT_PreSuspense module.

Set this entry to **True** if the current pipeline is part of a chain of pipelines, and it is not the first pipeline in the chain that includes FCT_PreSuspense.

Dependencies

This module must be the first preprocessing module in a pipeline.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-105 lists the FCT_PreSuspense registry entries.

Table 3-105 FCT_PreSuspense Registry Entries

Entry	Description	Mandatory
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive You can use this entry in a semaphore file.	Yes
KeepExistingBatchIds	A value of **True** preserves the Batch ID in the detail record of each EDR in an EDR file. A value of **False** sets the Batch ID of each EDR to the Batch ID contained in the header record of the batch input file. The default is **False**.	No

QueryableFields
(Suspense Manager only)
	Specifies which fields in which tables are queryable from the Suspense Management Center. Includes the EDR container fields that correspond to the database fields.
	This entry is only useful to customers who have purchased Suspense Manager.
	Format:
	`QueryableFields`
	`{`
	`  table_name_1`
	`  {`
	`  database_column_name_1 = edr_container_field_1`
	`  database_column_name_2 = edr_container_field_2`
	`  }`
	`  table_name_2`
	`  {`
	`  database_column_name_3 = edr_container_field_3`
	`  database_column_name_4 = edr_container_field_4`
	`  }`
	`}`
	If this entry is not present, this module sets HEADER.QUERYABLE_FIELDS_MAPPING and DETAIL.ASS_SUSPENSE_EXT.QUERYABLE_FIELDS to empty strings.
		Yes

Sample Registry

```
#---------------------------
# PreSuspense FCT
#---------------------------
PreSuspense
{
  ModuleName                = FCT_PreSuspense
  Module
  {
    Active                  = True
    QueryableFields
    {
      # table name. If more than one table, use a separate block
      SUSP_USAGE_TELCO_INFO_T
      {
        # format : <database_column_name> = <edr_conatiner_field_name>
        BYTES_IN = DETAIL.VOLUME_RECEIVED
```

```
        BYTES_OUT = DETAIL.VOLUME_SENT
        CALLED_TO = DETAIL.B_NUMBER
        #CALLING_FROM = DETAIL.B_NUMBER
        CALL_DURATION = DETAIL.DURATION
        PRIMARY_MSID = DETAIL.A_NUMBER
        SERVICE_TYPE = DETAIL.BASIC_SERVICE
        START_TIME = DETAIL.CHARGING_START_TIMESTAMP
        USAGE_TYPE = DETAIL.USAGE_TYPE
      }
      #Examples for GPRS calls
      #SUSP_USAGE_TELCO_GPRS_INFO_T
      #{
        # format : <database_column_name> = <edr_conatiner_field_name>
        #APN = DETAIL.ASS_GPRS_EXT.APN_ADDRESS
        #GGSN_ADDRESS = DETAIL.ASS_GPRS_EXT.GGSN_ADDRESS
        #NODE_ID = DETAIL.ASS_GPRS_EXT.NODE_ID
        #SECONDARY_MSID = DETAIL.ASS_GPRS_EXT.PORT_NUMBER
        #SGSN_ADDRESS = DETAIL.ASS_GPRS_EXT.SGSN_ADDRESS
      #}
      #SUSP_USAGE_TELCO_GSM_INFO_T
      #{
        #APN = DETAIL.ASS_GSMW_EXT.APN_ADDRESS
        #CELL_ID = DETAIL.ASS_GSMW_EXT.CELL_ID
        #DESTINATION_SID = DETAIL.ASS_GSMW_EXT.TERMINATING_SWITCH_IDENTIFICATION
        #DIALED_NUMBER = DETAIL.ASS_GSMW_EXT.DIALED_DIGITS
        #ORIGIN_SID = DETAIL.ASS_GSMW_EXT.ORIGINATING_SWITCH_IDENTIFICATION
        #SECONDARY_MSID = DETAIL.ASS_GSMW_EXT.PORT_NUMBER
      #}
    }
  }
}
```

Semaphore File Entries

Table 3-106 lists the FCT_PreSuspense Semaphore file entry.

Table 3-106 FCT_PreSuspense Semaphore File Entry

Entry	Description
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive

Sample Semaphore File Entry

```
ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.PreSuspense.Module.Active = False
```

EDR Container Fields

Table 3-107 lists the FCT_PreSuspense EDR container fields.

Table 3-107 FCT_PreSuspense EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
QUERYABLE_FIELDS_MAPPING HEADER.QUERYABLE_FIELDS_MAPPING	String	Write	Database column names and data types that map queryable fields. This field is used by Suspense Manager only. This is an empty string for standard recycling implementations.
PIPELINE_NAME DETAIL.ASS_SUSPENSE_EXT.PIPELINE_NAME	String	Write	The name of the pipeline, derived from the pipeline registry.
SOURCE_FILENAME DETAIL.ASS_SUSPENSE_EXT.SOURCE_FILENAME	String	Write	The source file name. The same as INTERNAL.STREAM_NAME.
EDR_BUF DETAIL.ASS_SUSPENSE_EXT.EDR_BUF	String	Write	A stored representation of the EDR container with its original field values.
EDR_SIZE DETAIL.ASS_SUSPENSE_EXT.EDR_SIZE	Integer	Write	The size of DETAIL.ASS_SUSPENSE_EXT.EDR_BUF.
QUERYABLE_FIELDS DETAIL.ASS_SUSPENSE_EXT.QUERYABLE_FIELDS	String	Write	The queryable field values defined in the registry. This field is used by Suspense Manager only. This is an empty string for standard recycling implementations.
BATCH_ID DETAIL.BATCH_ID	String	Write	At recycling, the value is set from HEADER.BATCH_ID.
INTERN_PROCESS_STATUS DETAIL.INTERN_PROCESS_STATUS	Integer	Read	Indicates whether the EDR is being recycled or test recycled.
ORIGINAL_BATCH_ID DETAIL.ORIGINAL_BATCH_ID	String	Write	Set the first time EDRs go through the pipeline (not being recycled or rerated.) Sets the value from HEADER.BATCH_ID in the header record.
BATCH_ID HEADER.BATCH_ID	String	Read	At recycling, modifies this value to ensure that it is unique.

FCT_RateAdjust

The FCT_RateAdjust module adjusts the charge for an EDR after rating has been performed.

Dependencies

If the rate adjustment data is stored in the database, the module requires a connection to the Pipeline Manager database.

This module must run after the FCT_MainRating module to adjust the rate.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-108 lists the FCT_RateAdjust registry entries.

Table 3-108 FCT_RateAdjust Registry Entries

Entry	Description	Mandatory
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive You can use this entry in a semaphore file.	Yes
DataConnection	Specifies the connection to the Pipeline Manager database.	Yes, if the data is stored in the database. Otherwise not used.
RateAdjustFile	Specifies file name that contains the rate adjustment data. See "Creating a Rate Adjustment Rules File" in *BRM Setting Up Rating and Discounting*. You can use this entry in a semaphore file.	Yes, if the data is stored in a file. Otherwise not used.
Source	Specifies where the rate adjustment data is stored: • **File** • **Database**	Yes

Sample Registry for the Database Interface

```
RateAdjust
{
  ModuleName = FCT_RateAdjust
  Module
  {
    Active = True
    Source = Database
    DataConnection = ifw.DataPool.DataConnection
  }
}
```

Sample Registry for the File Interface

```
RateAdjust
{
  ModuleName = FCT_RateAdjust
  Module
  {
    Active = True
    Source = File
```

```
      RateAdjustFile = /data/etc/discount.dat
    }
  }
```

Semaphore File Entries

Table 3-109 lists the FCT_RateAdjust Semaphore file entries.

Table 3-109 FCT_RateAdjust Semaphore File Entries

Entry	Description
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive
RateAdjustFile	Specifies file name that contains the rate adjustment data.
Reload	Reloads data from the database.

Sample Semaphore File Entry

```
ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.RateAdjustment.Module.Active = False
```

EDR Container Fields

Table 3-110 lists the FCT_RateAdjust EDR container fields.

Table 3-110 FCT_RateAdjust EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
ASS_CBD DETAIL.ASS_CBD	Block-Index	Read	Data block.
ASS_CBD_CHARGE_PACKET DETAIL.ASS_CBD.CP	Block-Index	Read	Data block.
ASS_CBD_RECORD_NUM DETAIL.ASS_CBD.RECORD_NUMBER	Integer	Read	Contains the record number.
USAGE_CLASS DETAIL.USAGE_CLASS	String	Read	Contains the usage class.
USAGE_TYPE DETAIL.USAGE_TYPE	String	Read	Contains the usage type.
INTERN_SERVICE_CODE DETAIL.INTERN_SERVICE_CODE	String	Read	Contains the internal service code.
INTERN_SERVICE_CLASS DETAIL.INTERN_SERVICE_CLASS	String	Read	Contains the internal service class.
BDR_START_TIMESTAMP DETAIL.CHARGING_START_TIMESTAMP	Date	Read	Contains the charging time stamp.
DURATION DETAIL.DURATION	Decimal	Read	Contains the duration of the event.

Table 3-110 (Cont.) FCT_RateAdjust EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
SOURCE_NETWORK DETAIL.SOURCE_NETWORK	String	Read	Contains the source network.
DESTINATION_NETWORK DETAIL.DESTINATION_NETWORK	String	Read	Contains the destination network.
INTERN_RATEPLAN DETAIL.ASS_CBD.CP.INTERN_RATEPLAN	String	Read	Contains the charge code.
INTERN_RATEPLAN_VERSION DETAIL.ASS_CBD.CP.INTERN_RATEPLAN_VERSION	Integer	Read	Contains the charge version.
ASS_CBD_IMPACT_CATEGORY DETAIL.ASS_CBD.CP.IMPACT_CATEGORY	String	Read	Contains the impact category.
CHARGED_AMOUNT_VALUE DETAIL.ASS_CBD.CP.CHARGED_AMOUNT_VALUE	Decimal	Read/Write	Contains the charge amount value.

Database Tables

The FCT_RateAdjust module uses the IFW_RATEADJUST table to set the rate adjustment rules.

> **Note:**
>
> For information on compare patterns used in database values, see "About Using Regular Expressions when Specifying the Data to Extract" in *BRM Setting Up Pipeline Rating and Discounting*.

FCT_Recycle

The FCT_Recycle module runs at the end of the pipeline It does either of the following:

- When the FCT_PreRecycle module runs in test mode, the FCT_Recycle module creates a report about the processing, but does not send the EDRs to an output file.
- When the FCT_PreRecycle module runs in recycle mode, the FCT_Recycle module sends the results to an output file, and attaches a sequence number to the output file.

See:

- Configuring Standard Recycling
- Recycling EDRs in Pipeline-Only Systems

Dependencies

> **Note:**
>
> You must configure the FCT_Recycle module as the last module of all function modules in the pipeline.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-111 lists the FCT_Recycle registry entries.

Table 3-111 FCT_Recycle Registry Entries

Entry	Description	Mandatory
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive You can use this entry in a semaphore file.	Yes
RecycleLog	Specifies the log file parameters: • **MessageFilePath** Specifies the path where the log file can find the message database. • **MessageFilePrefix** Specifies the prefix for collecting the files from the message file path. • **MessageFileSuffix** Specifies the suffix for collecting the files from the message file path. • **FilePath** Specifies the path in which the log file is written. • **FilePrefix** Specifies the prefix for the log file. • **FileSuffix** Specifies the suffix for the log file.	Yes

Sample Registry

```
Recycle
{
   ModuleName = FCT_Recycle
   Module
   {
      Active = True
      RecycleLog
      {
         MessageFilePath = ..
         MessageFilePrefix = Framework
         MessageFileSuffix = msg
         FilePath = ../tmp/log01
```

```
            FilePrefix = rej_
            FileSuffix = .log
        }
    }
}
```

Semaphore File Entries

Table 3-112 lists the FCT_Recycle Semaphore file entry.

Table 3-112 FCT_Recycle Semaphore File Entry

Entry	Description
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive

Sample Semaphore File Entry

```
ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.Recycle.Module.Active = True
```

EDR Container Fields

Table 3-113 lists the FCT_Recycle EDR container fields.

Table 3-113 FCT_Recycle EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
STREAM_NAME INTERNAL.STREAM_NAME	String	Read	Contains the stream name.
SEQ_CHECK INTERNAL.SEQ_CHECK	Integer	Write	Specifies the sequence check.
SEQ_GENERATION INTERNAL.SEQ_GENERATION	Integer	Write	Specifies the sequence generation.
OFFSET_GENERATION INTERNAL.OFFSET_GENERATION	Integer	Write	Specifies the offset generation.
PROCESS_STATUS INTERNAL.PROCESS_STATUS	Integer	Read	Contains the internal process status.

FCT_Reject

The FCT_Reject module analyzes the errors in an EDR and, if necessary, moves the EDR to a reject file.

See:

- Configuring Standard Recycling
- Recycling EDRs in Pipeline-Only Systems

Dependencies

This module must run after the rating and discount modules.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-114 lists the FCT_Reject registry entries.

Table 3-114 FCT_Reject Registry Entries

Entry	Description	Mandatory
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive You can use this entry in a semaphore file.	Yes
CallAssemblingModule	Provides data to the FCT_CallAssembling module to ensure that assembled calls are processed completely. See "Recycling Assembled EDRs" in *BRM Setting Up Pipeline Rating and Discounting*.	No
MinErrorSeverity	Specifies to reject EDRs that have a specified severity. To allow warning and normal messages without rejecting the EDR, set this entry to 3. Default = Not used. See "Processing EDRs with Errors" in *BRM Setting Up Pipeline Rating and Discounting*.	Yes
NotifyonReject	Specifies if other modules should be notified if an EDR is rejected. Default = **True**	No
StreamMap	Specifies a list of error types mapped to output streams. Important: A **UseRejectStream** entry is required to use **StreamMap**. See "Specifying Multiple Reject Streams" in *BRM Setting Up Pipeline Rating and Discounting*.	No
UseRejectStream	Specifies whether to use the reject output stream: • **True**. Rejected EDRs are sent to the reject stream. • **False**. Rejected EDRs are sent to the normal output stream, but flagged as discarded. Important: A **StreamMap** entry is required to use **UseRejectStream**. See "Using a Reject Output Stream" in *BRM Setting Up Pipeline Rating and Discounting*.	No

Sample Registry

```
Reject
{
   ModuleName = FCT_Reject
   {
         Active = True
         NotifyOnReject = True
         UseRejectStream = True
```

```
        CallAssemblingModule = ifw.Pipelines.Pipe.Functions.Standard.FunctionPool.CallAssembling
        StreamMap
              {
              error_type_1 = output_stream_1
              error_type_2 = output_stream_2
              intern = RejectStream
              logic = ReturnStream
              }
   }
}
```

Semaphore File Entries

Table 3-115 lists the FCT_Reject Semaphore file entry.

Table 3-115 FCT_Reject Semaphore File Entry

Entry	Description
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive

Sample Semaphore File Entry

```
ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.Rejection.Module.Active = False
```

Sample Output Configuration

You configure the reject stream in the registry in two places:

- In the pipeline configuration, for example:

```
Pipelines
{
ALL_RATE
{
Active   =   TRUE
.
.
.
RejectStream   =   RejectOutput
```

- In the Output configuration, for example:

```
# Output stream for rejected events
RejectOutput
{
 ModuleName = OUT_Reject
 Module
 {
  OutputStream
  {
   ModuleName = EXT_OutFileManager
   Module
   {
    OutputPath = ./samples/wireless/data/rej
    OutputPrefix = test
    OutputSuffix = .rej
```

```
    TempPrefix   = tmp

    TempDataPath    = ./samples/wireless/data/rej
    TempDataPrefix  = rej.tmp.
    TempDataSuffix  = .data

    Replace = TRUE
    DeleteEmptyFile = TRUE
   }
  }
 }
} # end of Reject
```

See "Configuring Output for Rejected or Duplicate EDRs" in *BRM Setting Up Pipeline Rating and Discounting*.

> **Note:**
>
> To ensure output file integrity, specify a unique combination of OutputPath, OutputSuffix, and OutputPrefix values for each output stream defined in the registry.

EDR Container Fields

Table 3-116 lists the FCT_Reject container fields.

Table 3-116 FCT_Reject Container Fields

Field name	Access	Description
DISCARDING DETAIL.DISCARDING	Read/Write	**Read:** This field is used to detect pre-rejected EDRs. If this field is 1, the EDR is always rejected. **Write:** If this field is 0 and the EDR is rejected, the field is set to 1.
ERROR_REJECT_TYPE DETAIL.ERROR_REJECT_TYPE	Read	Specifies the type of error in the EDR. This determines which stream the EDR is directed to.

FCT_Rounding

The FCT_Rounding module performs rounding for rating and discounting. Add this module to the pipeline after the processing module for which it is rounding.

The FCT_Rounding pipeline module converts the rounding mode in the **/config/beid** object into the corresponding rounding mode used by the BAS rounding method in Pipeline Manager. If you use the BAS Decimal rounding method in custom pipeline modules and iScripts, you should include the FCT_Rounding module in your pipeline. Otherwise, you will need to convert the rounding mode in the **/config/beid** object into the BAS rounding mode before calling the BAS rounding method.

Dependencies

Requires a connection to the DAT_Currency module.

This module must run after the FCT_RateAdjust module if you want rating results to be rounded and after FCT_Discount module if you want discount results to be rounded. FCT_Rounding must come after each module for which rounding should occur. For batch rating, it must come before the FCT_ApplyBalance module.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-117 lists the FCT_Rounding registry entries.

Table 3-117 FCT_Rounding Registry Entries

Entry	Description	Mandatory
Active	Specifies if the module is active or inactive: **True** = Active **False** = Inactive	Yes
CurrencyDataModule	Specifies the connection to the DAT_Currency module.	Yes
Mode	Specifies the process for which rounding is applied: **Rating** = Round the balance impact of rating. **Taxation** = Round the balance impact of taxation. **Discounting** = Round the balance impact of discounting.	Yes

Sample Registry

```
Rounding
{
   ModuleName = FCT_Rounding
   Module
   {
      Active = TRUE
      Mode   = Rating
      CurrencyDataModule = ifw.DataPool.CurrencyDataModule
   }
}
```

EDR Container Fields

Table 3-118 lists the FCT_Rounding EDR container fields.

Table 3-118 FCT_Rounding EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
EVENT_TYPE DETAIL.EVENT_TYPE	String	Read	Specifies the event type.
ASS_CBD_RECORD_NUMBER DETAIL.ASS_CBD.RECORD_NUMBER	Integer	Read	Specifies the record number.
ASS_CBD_RECORD_TYPE DETAIL.ASS_CBD.RECORD_TYPE	String	Read	Specifies the record type.

Table 3-118 (Cont.) FCT_Rounding EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
RESOURCE_ID DETAIL.ASS_CBD.DP.RESOURCE_ID	Decimal	Read	Specifies the balance element ID.
CHARGED_AMOUNT_CURRENCY DETAIL.ASS_CBD.CP.CHARGED_AMOUNT_CURRENCY	Decimal	Read	Specifies the currency of the charged amount.
CHARGED_AMOUNT_VALUE DETAIL.ASS_CBD.CP.CHARGED_AMOUNT_VALUE	Integer	Write	Specifies the rounded charged amount.
TAX_VALUE DETAIL.ASS_CBD.TP.TAX_VALUE	Decimal	Write	Specifies the rounded tax value.
GRANTED_AMOUNT DETAIL.ASS_CBD.DP.GRANTED_AMOUNT	Integer	Write	Specifies the rounded noncurrency discount amount.
RELATED_CHARGE_INFO_ID DETAIL.ASS_CBD.TP.RELATED_CHARGE_INFO_ID	Integer	Read	Specifies the index of the corresponding charge packet for which the tax was calculated.

FCT_RSC_Map

The FCT_RSC_Map module performs rate service class (RSC) mapping.

See "About Rate-Service Class Mapping" in *BRM Setting Up Pipeline Rating and Discounting*.

Dependencies

Requires a connection to the Pipeline Manager database.

This module must run before FCT_MainRating module to change the Service Code and Service Class fields of the Charge Packet. These fields are used by the main rating module to find out the rate to be applied for a particular call.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-119 lists the FCT_RSC_Map registry entries.

Table 3-119 FCT_RSC_Map Registry Entries

Entry	Description	Mandatory
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive You can use this entry in a semaphore file.	Yes
DataConnection	Specifies the connection to the Pipeline Manager database.	Yes

Table 3-119 (Cont.) FCT_RSC_Map Registry Entries

Entry	Description	Mandatory
DefaultRSCGroup	Specifies the default RSC group to use when the RSC group is not specified in the EDR.	Yes

Sample Registry

```
RSC_Mapping
{
  ModuleName = FCT_RSC_Map
  Module
  {
    Active = True
    DataConnection = integrate.DataPool.DataConnection
    DefaultRscGroup = testGroup
  }
}
```

Semaphore File Entries

Table 3-120 lists the FCT_RSC_Map Semaphore file entries.

Table 3-120 FCT_RSC_Map Semaphore File Entries

Entry	Description
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive
Reload	Reloads data from the database into memory.

Sample Semaphore File Entry

```
ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.RateServiceClassMap.
Module.Active = False
```

EDR Container Fields

Table 3-121 lists the FCT_RSC_Map EDR container fields.

Table 3-121 FCT_RSC_Map EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
BDR_CHARGING_START_TIMESTAMP DETAIL.CHARGING_START_TIMESTAMP	Date	Read	Contains the charging start time.
INTERN_SLA_RSC_GROUP DETAIL.INTERN_SLA_RSC_GROUP	String	Read	Contains the internal RSC group.

Table 3-121 (Cont.) FCT_RSC_Map EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
QOS_REQUESTED DETAIL.QOS_REQUESTED	String	Read	Contains the quality of service requested.
QOS_USED DETAIL.QOS_USED	String	Read	Contains the quality of service used.
USAGE_TYPE DETAIL.USAGE_TYPE	String	Read	Contains the usage type.
INTERN_SERVICE_CODE DETAIL.INTERN_SERVICE_CODE	String	Read	Contains the internal service code.
INTERN_SERVICE_CLASS DETAIL.INTERN_SERVICE_CLASS	String	Read	Contains the internal service class.
INTERN_USAGE_CLASS DETAIL.INTERN_USAGE_CLASS	String	Read	Contains the internal usage class.
INTERN_RATEPLAN DETAIL.ASS_CBD.CP.INTERN_RATEPLAN	String	Read	Contains the internal charge.
ASS_CBD_IMPACT_CATEGORY DETAIL.ASS_CBD.CP.IMPACT_CATEGORY	String	Read	Contains the impact category.
SERVICE_CODE_USED DETAIL.ASS_CBD.CP.SERVICE_CODE_USED	String	Write	Contains the service code used.
SERVICE_CLASS_USED DETAIL.ASS_CBD.CP.SERVICE_CLASS_USED	String	Write	Contains the service class used.

Database Interface

FCT_RSC_Map uses the following database tables:

- **IFW_RSC_MAP**. Stores the mapping rules. See "About Rate-Service Class Mapping" in *BRM Setting Up Pipeline Rating and Discounting*.
- The **IFW_RSC_GROUP**. Stores the RSC groups used for RSC mapping.
- The **IFW_SERVICECLASS**. Stores the service class codes used when defining service codes.

> **Note:**
>
> For information on compare patterns used in database values, see "About Using Regular Expressions when Specifying the Data to Extract" in *BRM Setting Up Pipeline Rating and Discounting*.

FCT_SegZoneNoCust

The FCT_SegZoneNoCust module finds the segment using the source network information instead of using the customer information.

Dependencies

Requires a connection to the Pipeline Manager database.

This module must run before the FCT_MainZoning module.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-122 lists the FCT_SegZoneNoCust registry entries.

Table 3-122 FCT_SegZoneNoCust Registry Entries

Entry	Description	Mandatory
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive You can use this entry in a semaphore file.	Yes
DataConnection	Specifies the connection to the Pipeline Manager database.	Yes
Segments	Specifies a list of mapping rules. Each rule defines the connection between the source network and the segment.	Yes

Sample Registry

```
SegZoneNoCust
{
  ModuleName = FCT_SegZoneNoCust
  Module
  {
    Active = True
    DataConnection = ifw.DataPool.Database
    Segments
    {
       26201 = SegmentD1
       26202 = SegmentD2
    }
  }
}
```

Semaphore File Entries

Table 3-123 lists the FCT_SegZoneNoCust Semaphore file entries.

Table 3-123 FCT_SegZoneNoCust Semaphore File Entries

Entry	Description
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive

Table 3-123 (Cont.) FCT_SegZoneNoCust Semaphore File Entries

Entry	Description
Reload	Reloads data from the database into memory.

Sample Semaphore File Entry

```
ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.SegZoneNoCust.Active = False
ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.SegZoneNoCust.Module.Reload { }
```

EDR Container Fields

Table 3-124 lists the FCT_SegZoneNoCust EDR container fields.

Table 3-124 FCT_SegZoneNoCust EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
ASS_ZBD DETAIL.ASS_ZBD	Block	Create	Data block.
ASS_ZBD_ZONE_PACKET DETAIL.ASS_ZBD.ZP	Block	Create	Data block.
SOURCE_NETWORK DETAIL.SOURCE_NETWORK	String	Read	Contains the source network.
BDR_CHARGING_START_TIMESTAMP DETAIL.CHARGING_START_TIMESTAMP	Date	Read	Contains the charging time stamp.
INTERN_SERVICE_CODE DETAIL.INTERN_SERVICE_CODE	String	Read	Contains the internal service code.
ASS_ZBD_RECORD_TYPE DETAIL.ASS_ZBD.RECORD_TYPE	String	Write	Contains the record type.
ASS_ZBD_SEGMENT_CODE DETAIL.ASS_ZBD.SEGMENT_CODE	String	Write	Contains the segment code.
ASS_ZBD_SERVICE_CODE DETAIL.ASS_ZBD.SERVICE_CODE	String	Write	Contains the resulting service code.
ASS_ZBD_INTERN_ZONE_MODEL DETAIL.ASS_ZBD.ZP.INTERN_ZONE_MODEL	Integer	Write	Contains the zone model.

Database Tables

The FCT_SegZoneNoCust module uses the **IFW_SEGZONE_LNK** database table. This table stores segments used for multi-segment zoning.

FCT_ServiceCodeMap

The FCT_ServiceCodeMap module maps external service codes to internal service codes.

Dependencies

Requires a connection to the Pipeline Manager database.

Some modules require an internal service code, so this module should run near the front of a pipeline.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-125 lists the FCT_ServiceCodeMap registry entries.

Table 3-125 FCT_ServiceCodeMap Registry Entries

Entry	Description	Mandatory
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive You can use this entry in a semaphore file.	Yes
DataConnection	Specifies the connection to the Pipeline Manager database.	Yes
MapGroup	Specifies the map group that the service code map belongs to. You can use this entry in a semaphore file.	Yes

Sample Registry

```
ServiceCodeMapping
{
  ModuleName = FCT_ServiceCodeMap
  Module
  {
     Active = True
     DataConnection = integrate.DataPool.Database
     MapGroup = serviceMapGroup
  }
}
```

Semaphore File Entries

Table 3-126 lists the FCT_ServiceCodeMap Semaphore file entries.

Table 3-126 FCT_ServiceCodeMap Semaphore File Entries

Entry	Description
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive
MapGroup	Specifies the mapping rule set.
Reload	Reloads data from the database into memory.

Sample Semaphore File Entry

```
ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.ServiceCodeMap.Module.Reload {}
ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.ServiceCodeMap.Module.MapGroup =
ALL_RATE
```

EDR Container Fields

The FCT_ServiceCodeMap module reads data from the EDR to map the external service code to the internal service code. The module then writes the internal service code and service class to the EDR.

Table 3-127 lists the FCT_ServiceCodeMap EDR container fields.

Table 3-127 FCT_ServiceCodeMap EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
BASIC_SERVICE DETAIL.BASIC_SERVICE	String	Read	Contains the external service code.
INTERN_SERVICE_CODE DETAIL.INTERN_SERVICE_CODE	String	Write	Contains the internal service code.
INTERN_SERVICE_CLASS DETAIL.INTERN_SERVICE_CLASS	String	Write	Contains the internal service class.
INTERN_USAGE_CLASS DETAIL.INTERN_USAGE_CLASS	String	Read	Contains the internal usage class.
ASS_GSMW_LOCATION_AREA_INDICATOR DETAIL.ASS_GSMW_EXT.LOCATION_AREA_INDICATOR	String	Read	Contains the GSMW extension location area.
ASS_GPRS_LOCATION_AREA_INDICATOR DETAIL.ASS_GPRS_EXT.LOCATION_AREA_INDICATOR	String	Read	Contains the GPRS extension location area.
QOS_REQUESTED DETAIL.QOS_REQUESTED	String	Read	Contains the quality of service requested.
QOS_USED DETAIL.QOS_USED	String	Read	Contains the quality of service used.
BDR_RECORD_TYPE DETAIL.RECORD_TYPE	String	Read	Contains the record type.

Database Tables

The FCT_ServiceCodeMap module uses the following database tables:

- **IFW_SERVICE_MAP**. Maps external service codes to internal service codes.
- **IFW_MAP_GROUP**. Stores the map groups used for service code mapping.

> **Note:**
>
> For information on compare patterns used in database values, see "About Using Regular Expressions when Specifying the Data to Extract" in *BRM Setting Up Pipeline Rating and Discounting*.

FCT_SocialNo

The FCT_SocialNo module flags social numbers for special processing. See "Setting Up Social Numbers" in *BRM Setting Up Pipeline Rating and Discounting*.

Dependencies

If the social number data is stored in the database, the FCT_SocialNo module requires a connection to the Pipeline Manager database.

This module can run anywhere.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-128 lists the FCT_SocialNo registry entries.

Table 3-128 FCT_SocialNo Registry Entries

Entry	Description	Mandatory
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive You can use this entry in a semaphore file.	No
DataConnection	Specifies the connection to the Pipeline Manager database.	Yes, if the data is stored in the database. Otherwise not used.
FileName	Specifies file that contains the social number data. See "Creating a Social Number Data File" in *BRM Setting Up Pipeline Rating and Discounting*.	Yes, if the data is stored in a file.
ReuseOnFailure	Specifies if the module should continue to use the old data if the **Reload** command fails. If **True**, the old data is used. The default is **False**.	Yes
SocialNoMapSize	Specifies the size of the in-memory map that stores social numbers. For a large set of social numbers to be loaded, specifying this parameter will enhance loading performance.	No
Source	Specifies where the social number data is stored: • **File** • **Database**	Yes

Sample Registry for the Database Interface

```
SocialNo
{
  ModuleName = FCT_SocialNo
  Module
  {
    ReuseOnFailure = false
    Active = true
    Source = database
    DataConnection = dataPool
  }
}
```

Sample Registry for the File Interface

```
SocialNo
{
  ModuleName = FCT_SocialNo
  Module
  {
    Active = True
    ReuseOnFailure = False
    Source = File
    FileName = ../daten/socialno.dat

}
```

Semaphore File Entries

Table 3-129 lists the FCT_SocialNo Semaphore file entries.

Table 3-129 FCT_SocialNo Semaphore File Entries

Entry	Description
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive
FileName	Reloads data from the specified file if **Source** parameter is set to **File**.
Reload	Reloads data from the database if **Source** parameter is set to **Database**.
ReuseOnFailure	Specifies if the module should continue to use the old data if the **Reload** command fails.

Sample Semaphore File Entry

```
ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.SocialNo.Module.Active = False
```

EDR Container Fields

The FCT_SocialNo module uses the following EDR container fields listed in Table 3-130 to flag social numbers for further processing.

Table 3-130 FCT_SocialNo EDR Container Fields.

Alias Field Name Default Field Name	Type	Access	Description
B_MODIFICATION_INDICATOR DETAIL.B_MODIFICATION_INDICATOR	String	Read/Write	Contains the modification indicator.
B_NUMBER_ZONE DETAIL.INTERN_B_NUMBER_ZONE	String	Read	Contains the B number.

Database Interface

The FCT_SocialNo module uses the **IFW_SOCIALNUMBER** database table. This table stores social numbers. See "Setting Up Social Numbers" in *BRM Setting Up Pipeline Rating and Discounting*.

> **Note:**
>
> For information on compare patterns used in database values, see "About Using Regular Expressions when Specifying the Data to Extract" in *BRM Setting Up Pipeline Rating and Discounting*.

FCT_Suspense

This module is used both by the standard recycling mechanism and by the Suspense Manager service integration component that you purchase separately. Both implementations are described below.

> **Note:**
>
> With one exception, FCT_Suspense must be the last function module in the pipeline. This ensures that it processes the final EDR container, including overwritten field values and enrichment field values. However, if FCT_AggreGate is used, it can be after FCT_Suspense.

Standard Recycling Implementation

As part of the standard recycling mechanism, the BRM FCT_Suspense function module:

- Routes EDRs being recycled from SuspenseCreateOutput to suspenseUpdateOutput.
- Logs the results of test recycling (if the **LogTestResults** registry entry is set).

Suspense reason and subreason codes are not supported with standard recycling, and these codes are all set to **O** (other).

Suspense Manager Implementation

As part of Suspense Manager, this module adds suspense reason and suspense subreason codes to EDRs. The specific errors that it adds are based on the error codes assigned to the EDR by the pipeline and the mapping information stored in the **/config/suspense_reason_code** object. If no **/config/suspense_reason_code** object is present, this module sets the suspense reason to **O** (other).

Dependencies

Requires a connection to the BRM database.

This module must be the last one in the pipeline.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-131 lists the FCT_Suspense registry entries.

Table 3-131 FCT_Suspense Registry Entries

Entry	Description	Mandatory
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive You can use this entry in a semaphore file.	Yes
DataConnection	Specifies the connection to the BRM database.	Yes
LogTestResults	For standard recycling only. Determines whether the results of test recycling are logged. If this entry is not present, the results are not logged. If set to **True**, the **RecycleLog** entry must also be present in the FCT_Suspense registry. Default = **False** See "pin_recycle".	No
RecycleLog	Specifies the log file parameters.	Yes, when **LogTestResults** is set to **True**.
RecycleLog.MessageFilePath	Specifies the path where the log file can find the message database.	Yes, when **LogTestResults** is set to **True**.
RecycleLog.MessageFilePrefix	Specifies the prefix for collecting the files from the message file path.	Yes, when **LogTestResults** is set to **True**.
RecycleLog.MessageFileSuffix	Specifies the suffix for collecting the files from the message file path.	Yes, when **LogTestResults** is set to **True**.
RecycleLog.FilePath	Specifies the path in which the log file is written.	Yes, when **LogTestResults** is set to **True**.

Table 3-131 (Cont.) FCT_Suspense Registry Entries

Entry	Description	Mandatory
RecycleLog.FilePrefix	Specifies the prefix for the log file.	Yes, when **LogTestResults** is set to **True**.
RecycleLog.FileSuffix	Specifies the suffix for the log file.	Yes, when **LogTestResults** is set to **True**.
SuspenseCreateStream	Specifies the output stream for newly suspended EDRs.	Yes
SuspenseUpdateStream	Specifies the output stream for recycled EDRs.	Yes

Sample Registry

```
#---------------------------
# Suspense FCT
#---------------------------
Suspense
{
  ModuleName                = FCT_Suspense
  Module
  {
    Active                  = True
    SuspenseCreateStream    = SuspenseCreateOutput
    SuspenseUpdateStream    = SuspenseUpdateOutput
    EdrFieldMap             = DETAIL.ASS_GSMW_EXT.PORT_NUMBER
    DataConnection          = ifw.DataPool.LoginInfranet
    LogTestResults          = True
    RecycleLog
    {
       MessageFilePath = ..
       MessageFilePrefix = Framework
       MessageFileSuffix = msg
       FilePath = ../tmp/log01
       FilePrefix = rej_
       FileSuffix = .log
    }
  }
}
```

Semaphore File Entries

Table 3-132 lists the FCT_Suspense file entry.

Table 3-132 FCT_Suspense Semaphore File Entry

Entry	Description
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive

Sample Semaphore File Entry

```
ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.Suspense.Module.Active = False
```

EDR Container Fields

Table 3-133 lists the FCT_Suspense EDR container fields.

Table 3-133 FCT_Suspense EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
ERROR_CODE DETAIL.ASS_SUSPENSE_EXT.ERROR_CODE	String	Write	The error code for the most severe error reported by the pipeline for the EDR.
SUSPENSE_REASON DETAIL.ASS_SUSPENSE_EXT.SUSPENSE_REASON	String	Write	The suspense reason. Mapped from the error code. Used by Suspense Manager only. This field is set to **0** for standard recycling implementations.
SUSPENSE_SUBREASON DETAIL.ASS_SUSPENSE_EXT.SUSPENSE_SUBREASON	String	Write	The suspense subreason. Mapped from the error code. Used by Suspense Manager only. This field is set to **0** for standard recycling implementations.
EDR_BUF DETAIL.ASS_SUSPENSE_EXT.EDR_BUF	String	Write	A stored representation of the EDR container including fields overwritten and enriched by the pipeline.
EDR_SIZE DETAIL.ASS_SUSPENSE_EXT.EDR_SIZE	Integer	Write	The size of DETAIL.ASS_SUSPENSE_EXT.EDR_BUF.
PROCESS_STATUS DETAIL.INTERN_PROCESS_STATUS	Integer	Read	Indicates whether the EDR is being recycled or test recycled.
SUSPENSE_STATUS DETAIL.ASS_SUSPENSE_EXT.SUSPENSE_STATUS	Integer	Write	Indicates whether the EDR is suspended or successfully recycled.
BATCH_ID DETAIL.BATCH_ID	String	Write	Writes the batch ID from the header record, except during recycling.
BATCH_ID HEADER.BATCH_ID	String	Read	Written during recycling.

FCT_Timer

The FCT_Timer module sets the timer ID for an EDR and stores a copy of the EDR when the original EDR is sent to the processing pipeline.

If an EDR times out, FCT_Timer sets the timeout flag to **True** and sends:

- The original EDR with a timeout flag to the exception pipeline, which is the pipeline to which the original EDR is sent when it times out.
- The duplicate EDR to the timeout pipeline, which is the pipeline to which the duplicate EDR is sent when it times out.

If the EDR is processed within the time limit, FCT_Timer sets the timeout flag to **False**.

FCT_Timer also handles heartbeat and keep-alive messages by automatically resetting the timer at the interval specified in the **KeepAliveInterval** registry entry when there is message traffic between the CM and the Intelligent Network (IN).

Dependencies

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-134 lists the FCT_Timer registry entries.

Table 3-134 FCT_Timer Registry Entries

Entry	Description	Mandatory
Active	Specifies whether the module is active or inactive **True** = Active (default). **False** = Inactive.	Yes
Threads	Specifies the number of threads running this module. Default = **1**.	Yes
Reactors	Specifies the number of reactor objects to use. The reactors detect timeout events and then dispatch the events to the timer handler. The recommended number of reactors is from 1 to 5.	No
TimeoutQueue	Specifies the pipeline queue to which the duplicate EDR is sent when the EDR times out.	Yes
KeepAliveInterval	Specifies the idle timeout value in milliseconds, which specifies how long to wait before sending the original EDR to the exception pipeline. Default = **3000**.	Yes
Timeout	Specifies the idle timeout value in microseconds, which specifies how long to wait before sending the duplicate EDR to the timeout pipeline. Default = **100000**.	Yes
NoOpcodeNumbers	Specifies the number of the BRM opcode that prevents duplicate EDRs from being created. When the OPCODE_NUM EDR field is set to the specified number, the module does not create a duplicate EDR for the Timeout pipeline. Opcode numbers are defined in header (***.h**) files in the *BRM_home***/include/ops** directory. *BRM_home* is the directory where you installed BRM components.	No

Sample Registry

```
Timer
{
```

```
  ModuleName = FCT_Timer
   Module
  {
   Active = TRUE
   Threads = 1
   Reactors = 3
   TimeoutQueue = ifw.IPCQueues.TimeoutQueue
   KeepAliveInterval = 3000 ### milliseconds
   Timeout = 100000 ### microseconds
   NoOpcodeNumbers = 5000, 50001
   }
}
```

EDR Container Fields

FCT_Timer uses the EDR container fields listed in Table 3-135:

Table 3-135 FCT_Timer EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
TIMER_ID DETAIL.TIMER_ID	Integer	Write	ID assigned to the EDR container's timer when FCT_SetTimer schedules it. This ID is required to cancel the timer.
CURRENT_TIME DETAIL.CURRENT_TIME	Integer	Read	Current system time
TIMEOUT_OFFSET DETAIL.TIMEOUT_OFFSET	Integer	Write	Offset from the current system time
OPCODE_NUM DETAIL.OPCODE_NUM	Integer	Read	Specifies the number of the BRM opcode that performs the requested action. Opcode numbers are defined in header (***.h**) files in the *BRM_home*/**include/ops** directory.
TIMEOUT_FLAG DETAIL.TIMEOUT_FLAG	Integer	Write	Specifies whether the EDR has timed out: • **0** is False. • **1** is True.
SESSION_ID DETAIL.SESSION_ID	String	Write	ID required to cancel the timer.
MILLISEC_TIME DETAIL.MILLISEC_TIME	Integer	Read	The latency time in milliseconds.

FCT_TriggerBill

The FCT_TriggerBill module sends EDRs to the billing-trigger output stream to trigger immediate billing for the associated accounts. It also sets a billing-trigger error code used to route the EDRs to the suspense output stream, and the **Trigger_Billing** recycle key used to retrieve the suspended EDRs for recycling.

Dependencies

Configure the FCT_TriggerBill module to run before the FCT_MainRating module.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-136 lists the FCT_TriggerBill registry entries.

Table 3-136 FCT_TriggerBill Registry Entries

Entry	Description	Mandatory
Active	Specifies if the module is active or inactive: **True** = Active **False** = Inactive	Yes
TriggerBillCreateStream	Specifies the billing-trigger output stream module.	Yes

Sample Registry

```
TriggerBill
{
   ModuleName = FCT_TriggerBill
   Module
   {
      Active = TRUE
      TriggerBillCreateStream = TriggerBillCreateOutput
   }
}
```

Semaphore File Entries

Table 3-137 lists the FCT_TriggerBill Semaphore file entry.

Table 3-137 FCT_TriggerBill Semaphore File Entry

Entry	Description
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive

EDR Container Fields

Table 3-138 lists the FCT_TriggerBill EDR container fields.

Table 3-138 FCT_TriggerBill EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
ACTG_NEXT_DATE DETAIL.CUST.A.ACTG_NEXT_DATE	String	Read	The date that the current monthly cycle ends. Used to determine the accounting cycle to which the EDR belongs.

Table 3-138 (Cont.) FCT_TriggerBill EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
BILL_STATE DETAIL.CUST.A.BILL_STATE	String	Read	The billing state. Possible values are: • **PIN_ACTG_CYCLE_OPEN** (unbilled) • **PIN_ACTG_CYCLE_CLOSED** (billed)
CHARGING_START_TIMESTAMP DETAIL.CHARGING_START_TIMESTAMP	String	Read	The timestamp when the call started. Used to determine the accounting cycle to which the EDR belongs.
ERROR_CODE DETAIL.ASS_SUSPENSE_EXT.ERROR_CODE	String	Write	Specifies the billing-trigger error code. Used to send the EDR to the suspense output stream.
RECYCLE_KEY DETAIL.ASS_SUSPENSE_EXT.RECYCLE_KEY	String	Write	The key value that identifies the EDRs suspended for pipeline-triggered billing. Set to **Trigger_Billing** when BILL_STATE is unbilled and ACTG_NEXT_DATE is passed.
UTC_TIME_OFFSET DETAIL.UTC_TIME_OFFSET	X(5)	Read	The UTC time offset.

FCT_UoM_Map

The FCT_UoM_Map module converts the unit of measurement (UoM) of an incoming EDR to a UoM needed for rating a particular service.

Dependencies

Requires a connection to the Pipeline Manager database.

Must run after the FCT_ServiceCodeMap module, and before the rating modules.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-139 lists the FCT_UoM_Map registry entries.

Table 3-139 FCT_UoM_Map Registry Entries

Entry	Description	Mandatory
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive You can use this entry in a semaphore file.	Yes
DataConnection	Specifies the connection to the Pipeline Manager database.	Yes

Table 3-139 (Cont.) FCT_UoM_Map Registry Entries

Entry	Description	Mandatory
Mapping	Specifies the mapping rules.	Yes
Mapping.AssCBDServiceCode	Specifies the service code field in the associated charge breakdown records that is used for the mapping.	No
Mapping.InternServiceCode	Specifies the service code field in the basic detail block that is used for the mapping.	No

Sample Registry

```
UoM_Map
{
  ModuleName = FCT_UoM_Map
  Module
  {
    Active = True
    DataConnection = integrate.DataPool.Login
    Mapping
    {
      InternServiceCode = INTERN_SERVICE_CODE
      AssCBDServiceCode = ASS_CBD_SERVICE_CODE
    }
  }
}
```

Semaphore File Entries

Table 3-140 lists the FCT_UoM_Map Semaphore file entries.

Table 3-140 FCT_UoM_Map Semaphore File Entries

Entry	Description
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive
Reload	Reloads data from the database into memory.

Sample Semaphore File Entry

```
ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.UoM_Map.Module.Active = False
```

EDR Container Fields

Table 3-141 lists the FCT_UoM_Map EDR container fields.

Table 3-141 FCT_UoM_Map EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
INTERN_SERVICE_CODE DETAIL.INTERN_SERVICE_CODE	String	Read	Contains the internal service code.
ASS_CBD_SERVICE_CODE DETAIL.ASS_CBD.SERVICE_CODE	String	Read	Contains the charge breakdown service code.
DETAIL.ASSOCIATED_CHARGE.CHARGE_PACKET ASS_CBD_CHARGE_PACKET	Struct	Read/Write	Contains charge packet data.

Database Interface

FCT_UoM_Map accesses the following database tables:

- **IFW_SERVICE**. This table stores data about services and associated RUMs.
- **IFW_RUMGROUP_LNK**. This table defines a list of RUM/UOM pairs with the RUM group value obtained from the **IFW_SERVICE** table.
- **IFW_UOM_MAP**. This table maps a UoM to a basic detail or to an associated charge packet.
- **IFW_UOM**. This table stores the UoMs for pipeline rating.
- **IFW_ALIAS_MAP**. This table stores an alias name for each RUM and UoM.

FCT_UsageClassMap

The FCT_UsageClassMap module maps external codes for supplementary services, such as call forwarding, to internal usage classes.

Dependencies

Requires a connection to the Pipeline Manager database.

The FCT_UsageClassMap module is run before the zoning and rating modules.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-142 lists the FCT_UsageClassMap registry entries.

Table 3-142 FCT_UsageClassMap Registry Entries

Entry	Description	Mandatory
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive You can use this entry in a semaphore file.	Yes

Table 3-142 (Cont.) FCT_UsageClassMap Registry Entries

Entry	Description	Mandatory
DataConnection	Specifies the connection to the Pipeline Manager database.	Yes
MapGroup	Specifies the map group. You can use this entry in a semaphore file.	Yes
OverwriteUsageClass	Specifies if the external usage class should be overwritten by the internal one. The default is to not overwrite the external usage class; if you map usage codes, you should enable this entry. Default = **False**	No
OptimizeFor	Specifies if the module is configured to optimize memory consumption as well as pipeline startup time. **Memory** = Optimizes memory consumption and pipeline startup time. **No memory optimization** = Does not optimize memory consumption and pipeline startup time (the default). **Note** • Enabling this entry might have an adverse impact on the number of call detail records (CDRs) processed in a specific time interval. • This entry is read only at pipeline start up. Its value cannot be changed by using a semaphore.	No

Sample Registry

```
UsageClassMapping
{
  ModuleName = FCT_UsageClassMap
  Module
  {
    Active  = True
    DataConnection  =  ifw.DataPool.Database
    OverwriteUsageClass = False
    MapGroup  = mapGroup0
    OptimizeFor = Memory
  }
}
```

Semaphore File Entries

Table 3-143 lists the FCT_UsageClassMap Semaphore file entries.

Table 3-143 FCT_UsageClassMap Semaphore File Entries

Entry	Description
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive
MapGroup	Specifies the mapping rule set.
Reload	Reloads data from the database into memory.

Sample Semaphore File Entry

```
ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.UsageClassMap.Module.Reload {}
ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.UsageClassMap.Module.MapGroup =
ALL_RATE
```

EDR Container Fields

The FCT_UsageClassMap module adds the internal usage class to the EDR. All other fields in Table 3-144 are used for mapping the usage class.

Table 3-144 FCT_UsageClassMap EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
BDR_RECORD_TYPE DETAIL.RECORD_TYPE	String	Read	Contains the event record type.
USAGE_CLASS DETAIL.USAGE_CLASS	String	Read	Contains the external usage class.
USAGE_TYPE DETAIL.USAGE_CLASS	String	Read	Contains the external usage type.
WHOLESALE_IMPACT_CATEGORY DETAIL.WHOLESALE_IMPACT_CATEGORY	String	Read	Contains the wholesale impact category.
TARIFF_CLASS DETAIL.TARIFF_CLASS	String	Read	Contains the tariff class.
TARIFF_SUB_CLASS DETAIL.TARIFF_SUB_CLASS	String	Read	Contains the tariff subclass.
INTERN_USAGE_CLASS DETAIL.INTERN_USAGE_CLASS	String	Write	Contains the internal usage class.
CONNECT_TYPE DETAIL.CONNECT_TYPE	String	Read	Contains the connection type.
CONNECT_SUB_TYPE DETAIL.CONNECT_SUB_TYPE	String	Read	Contains the connection subtype.
APN_ADDRESS DETAIL.ASS_GPRS_EXT.APN_ADDRESS	String	Read	Contains the GPRS APN type.
ACTION_CODE DETAIL.ASS_GSMW_EXT.SS_PACKET.ACTION_CODE	String	Read	Contains the GSM SS packet action code.
SS_EVENT DETAIL.ASS_GSMW_EXT.SS_PACKET.SS_EVENT	String	Read	Contains the GSM SS packet action event.
INTERN_C_NUMBER_ZONE DETAIL.INTERN_C_NUMBER_ZONE	String	Read	Contains the internal normalized C number.
DETAIL.ASS_GPRS_EXT ASS_GPRS	String	Read	Contains the GPRS extension.
DETAIL.ASS_GSMW_EXT.SS_PACKET ASS_GSMW_SS_PACKET	String	Read	Contains the GSMW SS packet.

Database Tables

The FCT_UsageClassMap module uses the following database tables:

- The **IFW_USAGECLASS_MAP** table maps external supplementary service codes in the EDR to internal usage classes.
- The **IFW_USAGECLASS** table stores the usage classes that can be used as a result of usage class mapping.
- The **IFW_MAP_GROUP** table stores the map groups used for usage class mapping.

> **Note:**
>
> For information on compare patterns used in database values, see "About Using Regular Expressions when Specifying the Data to Extract" in *BRM Setting Up Pipeline Rating and Discounting*.

FCT_USC_Map

The FCT_USC_Map module performs usage scenario mapping.

Dependencies

This module needs a connection to the DAT_USC_Map module.

This module must run after the following:

- FCT_UsageClassMap
- ISC_UsageType
- FCT_PreRating

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-145 lists the FCT_USC_Map registry entries.

Table 3-145 FCT_USC_Map Registry Entries

Entry	Description	Mandatory
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive You can use this entry in a semaphore file.	Yes
DataModule	Specifies the connection to the DAT_USC_Map data module.	Yes

Table 3-145 (Cont.) FCT_USC_Map Registry Entries

Entry	Description	Mandatory
DefaultUSCGroup	Specifies the USC group that contains the mapping rules. If no matching rule is found, the FCT_USC_Map module uses the rule in the default USC map group. You can use this entry in a semaphore file.	Yes
LogZoneModelNotFoundEntries	Specifies, if set to **True**, that all log entries in INF_NO_USC_MAPPING_ENTRY are logged into the Stream log. The default value is **False**.	No
Mode	Specifies the mode in which USC mapping is done. The **Rating** mode (the default) specifies that USC mapping is done using the zone model from the charge packets. Mapping in this mode provides the impact category for charge packets. The **Zoning** mode specifies that USC mapping is done using the zone model from the EDR detail block. Mapping in this mode provides impact categories for the detail block. Using the **Zoning** mode requires that the DETAIL.RETAIL_IMPACT_CATEGORY and DETAIL.WHOLESALE_IMPACT_CATEGORY fields are populated.	No

Sample Registry

```
USC_Mapping
{
  ModuleName = FCT_USC_Map
  Module
  {
    Active = True
    DefaultUSCGroup = usc_group
    DataModule = ifw.DataPool.USCDataModule
  }
}
```

Semaphore File Entries

Table 3-146 lists the FCT_USC_Map Semaphore file entry.

Table 3-146 FCT_USC_Map Semaphore File Entry

Entry	Description
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive

Sample Semaphore File Entry

```
ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.UsageScenarioMap.
Module.Active = True

ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.UsageScenarioMap.
Module. DefaultUSCGroup  =  usc_group
```

EDR Container Fields

Table 3-147 lists the FCT_USC_Map EDR container fields.

Table 3-147 FCT_USC_Map EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
BDR_CHARGING_START_TIMESTAMP DETAIL.CHARGING_START_TIMESTAMP	String	Read	Contains the charging time stamp.
INTERN_SLA_USC_GROUP DETAIL.INTERN_SLA_USC_GROUP	String	Read	Contains the internal USC group.
USAGE_TYPE DETAIL.USAGE_TYPE	String	Read/Write	Contains the usage type code. This field is updated only when the **Mode** registry entry is set to **Zoning**. It is not updated when the **Mode** entry is set to **Rating**.
RETAIL_IMPACT_CATEGORY DETAIL.RETAIL_IMPACT_CATEGORY	String	Read/Write	Contains the retail impact category.
WHOLESALE_IMPACT_CATEGORY DETAIL.WHOLESALE_IMPACT_CATEGORY	String	Read/Write	Contains the wholesale impact category.
WHOLESALE_CHARGED_AMOUNT_VALUE DETAIL.WHOLESALE_CHARGED_AMOUNT_VALUE	String	Read	Contains the wholesale charged amount value.
IC_DESCRIPTION DETAIL.IC_DESCRIPTION	String	Write	Contains the zone description for displaying on invoices.
IC_DESCRIPTION DETAIL.ASS_CBD.CP.IC_DESCRIPTION	String	Write	Contains the zone description for displaying on invoices.
INTERN_SERVICE_CODE DETAIL.INTERN_SERVICE_CODE	String	Read	Contains the internal service code.
INTERN_SERVICE_CLASS DETAIL.INTERN_SERVICE_CLASS	String	Read	Contains the internal service class.
DURATION DETAIL.DURATION	String	Read	Contains the duration of the event.
INTERN_USAGE_CLASS DETAIL.INTERN_USAGE_CLASS	String	Read	Contains the internal usage class code.
BDR_INTERN_ZONE_MODEL DETAIL.INTERN_ZONE_MODEL	String	Read	Contains the zone model.
RATEPLAN_TYPE DETAIL.ASS_CBD.CP.RATEPLAN_TYPE	String	Read	Contains the charge breakdown record charge type.
ASS_CBD_INTERN_ZONE_MODEL DETAIL.ASS_CBD.CP.INTERN_ZONE_MODEL	String	Read	Contains the charge breakdown record zone model.

Table 3-147 (Cont.) FCT_USC_Map EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
ASS_CBD_IMPACT_CATEGORY DETAIL.ASS_CBD.CP.IMPACT_CATEGORY	String	Read/Write	Contains the charge breakdown record impact category.
ASS_CBD_ZONE_ENTRY_NAME DETAIL.ASS_CBD.CP.ZONE_ENTRY_NAME	String	Read	Contains the zone name used for the charge packet.
ZONE_ENTRY_NAME DETAIL.ZONE_ENTRY_NAME	String	Read	Contains the zone name of the event.

Database Interface

The FCT_USC_Map module uses the following database tables:

- IFW_USC_MAP. This table stores mapping rules for usage scenario maps.
- IFW_USC_GROUP. This table stores USC group codes used for usage scenario mapping.
- IFW_USAGETYPE. This table stores usage type codes used for usage scenario mapping.

> **Note:**
>
> For information on compare patterns used in database values, see "About Using Regular Expressions when Specifying the Data to Extract" in *BRM Setting Up Pipeline Rating and Discounting*.

FCT_Zone

The FCT_Zone module computes zones when you use Pipeline Manager only for zoning.

Dependencies

The FCT_Zone module requires a connection to the DAT_Zone module.

This module must run after FCT_Account.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 3-148 lists the FCT_Zone registry entries.

Table 3-148 FCT_Zone Registry Entries

Entry	Description	Mandatory
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive You can use this entry in a semaphore file.	Yes
DataModule	Specifies the connection to the DAT_Zone module.	Yes
EdrZoneModel	Specifies the zone model which should be used for the zoning. You can use this entry in a semaphore file.	Yes

Sample Registry

```
Zoning
{
  ModuleName = FCT_Zone
  Module
  {
    Active = True
    DataModule = ifw.DataPool.ZoneDataModule
    EdrZoneModel = ZM_ADD
  }
}
```

Semaphore File Entries

Table 3-149 lists the FCT_Zone Semaphore file entries.

Table 3-149 FCT_Zone Semaphore File Entries

Entry	Description
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive
EdrZoneModel	Specifies the zone model which should be used for the zoning.

Sample Semaphore File Entry

```
ifw.Pipelines.Functions.Processing.FunctionPool.Zoning.Module.EdrZoneModel = D2_FUN
```

EDR Container Fields

Table 3-150 lists the FCT_Zone EDR container fields.

Table 3-150 FCT_Zone EDR Container Fields

Alias Field Name Default Field Name	Type	Access	Description
BDR_CHARGING_START_TIMESTAMP DETAIL.CHARGING_START_TIMESTAMP	Date	Read	Contains the charging time stamp.
RETAIL_IMPACT_CATEGORY DETAIL.RETAIL_IMPACT_CATEGORY	String	Write	Contains the retail impact category.
WHOLESALE_IMPACT_CATEGORY DETAIL.WHOLESALE_IMPACT_CATEGORY	String	Write	Contains the wholesale impact category.
ZONE_DESCRIPTION DETAIL.ZONE_DESCRIPTION	String	Write	Contains the zone description for displaying on invoices.
ZONE_ENTRY_NAME DETAIL.ZONE_ENTRY_NAME	String	Write	Contains the destination description for displaying on invoices.
INTERN_SERVICE_CODE DETAIL.INTERN_SERVICE_CODE	String	Read	Contains the internal service code.
INTERN_A_NUMBER_ZONE DETAIL.INTERN_A_NUMBER_ZONE	String	Read	Contains the A number.
INTERN_B_NUMBER_ZONE DETAIL.INTERN_B_NUMBER_ZONE	String	Read	Contains the B number.
BDR_INTERN_ZONE_MODEL DETAIL.INTERN_ZONE_MODEL	Integer	Write	Contains the resulting zone model ID.
BDR_INTERN_APN_GROUP DETAIL.INTERN_APN_GROUP	String	Write	Contains the zone model related APN_GROUP for use by the FCT_APN_Map module.

4

Pipeline Manager Data Modules

Learn about the Oracle Communications Billing and Revenue Management (BRM) Pipeline Manager data modules.

Topics in this document:

DAT_AccountBatch

DAT_AccountBatch retrieves account data from the BRM database for the "DAT_ItemAssign", "FCT_Account", and "FCT_AccountRouter" modules.

This module also maintains a list of the accounts that are being rerated by the **pin_rerate** utility. This information is used by the batch rating pipeline to suspend incoming call detail records (CDRs) for those accounts while rerating is in progress.

Dependencies

This module requires connections to the following:

- BRM database.
- Pipeline Manager database.
- DAT_Listener module. See "DAT_Listener".
- DAT_PortalConfig module. See "DAT_PortalConfig".

Registry Entries

Table 4-1 lists the DAT_AccountBatch registry entries.

Table 4-1 DAT_AccountBatch Registry Entries

Entry	Description	Mandatory
AcceptLoginSearchFailure	If set to **True,** when a customer login number is not found in memory, the EDR will be accepted, the pipeline will continue processing the EDR, and a warning will be reported in the stream log. If set to **False** (the default), when a customer login number is not found in memory, the EDR will be set as invalid and rejected and a major error will be reported. **Important:** If the **UseAsRouter** registry entry is set to **True**, this registry entry is not used.	No
AccountLocks	Use this entry to tune performance by managing thread contention. Default = **10**	No
AddAliasList	Specifies whether all alias names and logins are added to the EDR. Default = **False** **Important:** If the **UseAsRouter** registry entry is set to **True**, this registry entry is not used.	No
ClosedAccountDelay	Specifies to not load closed accounts. Also specifies the number of days prior to the current date for which closed accounts are not loaded. For example, if **ClosedAccountDelay** is set to **10** and the current date it June 20, accounts that were closed prior to June 10 are not loaded into memory. Default = **0**	No
Connections	Specifies the number of connections to the database. This value must be at least the number of threads plus 1. Default = **5**	No

Table 4-1 (Cont.) DAT_AccountBatch Registry Entries

Entry	Description	Mandatory
EnrichRatingProductOnly	If set to **True**, only the purchased charge offers whose service ID matches the service ID in the EDR being rated are added to the EDR. If set to **False**, the purchased charge offers whose service type matches the service type in the EDR being rated are added to the EDR. **Note:** This entry is present for backward compatibility.	No
InfranetConnection	Specifies the connection to the BRM database.	Yes
InitialLoading	Specifies whether the initial loading of service and account data is performed. Otherwise, loading occurs while processing. Login objects are always loaded. Setting this entry to **False** enables the system to start faster. Default = **True** **Important:** If the **UseAsRouter** registry entry is set to **True**, this registry entry is not used.	No
IntegrateConnection	Specifies the connection to the Pipeline Manager database.	Yes
Listener	Specifies the connection to the DAT_Listener module.	Yes
LoadAccountForSharingOnly	Specifies whether Pipeline Manager can load serviceless accounts that are owners of resource sharing groups. Default = **False**	No
LoadLogins	Specifies whether the login is loaded in case of an existing alias list. When set to **True**, logins are loaded from both the PIN_FLD_LOGIN field and the PIN_FLD_ALIAS_LIST array. When set to **False**, logins are only loaded from the PIN_FLD_ALIAS_LIST array. Default = **False** when **UseAsRouter** is disabled. When **UseAsRouter** is enabled, then **LoadLogins** is always **True**.	No
LoadPercentage	Indicates the percentage of account POIDs to store locally when determining the account blocks for which each thread is responsible. Values must be greater than 0.000000 and less than or equal to 100.0. Default = **10.0**	Yes
LogEvents	Specifies whether received events should be written to a log file. Use this entry to troubleshoot Pipeline Manager event handling. Default = **False**	No
LoginLocks	Use this entry to tune performance by managing thread contention. Default = **10**	No
PerThreadJobsCount	Specifies the number of jobs per thread. **Important:** Setting the number of jobs per thread to a large number can decrease performance because of the system overhead associated with creating too many jobs. (Typically, three to eight jobs per thread is optimal). If you want to adjust the number of accounts or balances per job, increase or decrease the number of threads.	Yes

Table 4-1 (Cont.) DAT_AccountBatch Registry Entries

Entry	Description	Mandatory
PortalConfigDataModule	Specifies the connection to the DAT_PortalConfig module. This enables DAT_AccountBatch to retrieve business parameter settings from the DAT_PortalConfig module.	Yes
ProfilesNotLoadedIntoEdr	When you have accounts that own or share a large number of ERA profiles, use this registry entry to specify the profiles to be filtered. When a CDR is rated, if the value in the EDR field matches a value in one or more of the specified ERA profiles, the ISC_ProfileLabel iScript loads only the ERA labels into the EDR container. See "Improving Pipeline Rating Performance for Events with ERAs" in *BRM Setting Up Pipeline Pricing*.	No
ReadAccountBalances	Specifies whether to load account resource data. The data includes resource IDs, such as 840. If enabled, the RESOURCE_LIST field in the CUSTOMER_DATA block is populated with the resource IDs for that account. Default = **False** **Important:** If the **UseAsRouter** registry entry is set to **True**, this registry entry is not used.	No
ReadAllProducts	If set to **True,** all the purchased charge offers for the account are added to the EDR. If set to **False**, only those purchased charge offers matching the service types and event types of the CDR processed are added to the EDR.	No
ReadPlans	If set to **True**, the module loads plan IDs into memory when loading purchased charge offers. During EDR processing, the list of plan IDs for an account is returned to the FCT_Account module.	No
ReadSystemProductFromMain	If set to **True,** the module retrieves the latest system charge offers from the main tables and uses the start and end dates to validate when the charge offer is in effect. If set to **False** (the default), the module retrieves the system charge offer information from the audit tables and uses the purchase creation date to validate when the charge offer is in effect. **Important:** If the **UseAsRouter** registry entry is set to **True**, this registry entry is not used.	No
RejectClosedAccounts	Specifies whether to reject CDRs for accounts that are closed. If set to **True**, all closed account information is loaded from the database. Any CDR with a timestamp later than the account's closed date is rejected. Default = **False**	No
RowFetchSize	Specifies the number of rows of data to retrieve from the BRM database. Use this entry for performance tuning. Default = **1000**	No
ServiceLocks	Tunes performance by managing thread contention. Default = **10**	No

Table 4-1 (Cont.) DAT_AccountBatch Registry Entries

Entry	Description	Mandatory
ThreadAccountHashMapSize	Controls the size of the temporary hash map built by each thread for accounts. **Important:** • Changing the default system-calculated values for this entry is not recommended. • If the **UseAsRouter** registry entry is set to **True,** this registry entry is not used.	No
ThreadGroupSharingChargesHashMapSize	Controls the size of the temporary hash map built by each thread for loading charge share group data. The system-calculated default value might not be appropriate. **Important:** If the **UseAsRouter** registry entry is set to **True**, this registry entry is not used.	No
ThreadGroupSharingDiscountsHashMapSize	Controls the size of the temporary hash map built by each thread for loading discount sharing group data. The system-calculated default value might not be appropriate. **Important:** If the **UseAsRouter** registry entry is set to **True**, this registry entry is not used.	No
ThreadGroupSharingMonitorsHashMapSize	Controls the size of a temporary hash map constructed for GroupSharingProfile object storage during multi-thread DAT_Account initialization. Default = (TotalAccounts / NumThreads) * 0.10. The default value for this entry is appropriate in most cases. However, the value should be increased if you exceed an average of 4 GroupSharingMonitors for every 10 accounts.	No
ThreadGroupSharingProfilesHashMapSizes	Controls the size of the temporary hash map built by each thread for loading profile sharing group data. The system-calculated default value might not be appropriate. **Important:** If the **UseAsRouter** registry entry is set to **True**, this registry entry is not used.	No
ThreadLoginHashMapSize	Controls the size of the temporary hash map built by each thread for loading logins. The system-calculated default value is appropriate for most BRM implementations.	No
Threads	Specifies the number of threads. Set this value to at least the number of CPUs in the system. Increasing the number of threads increases performance, up to a point. Specifying too many threads decreases performance. Default = **4**	Yes
ThreadServiceHashMapSize	Controls the size of the temporary hash map built by each thread for loading services. The system-calculated default value is appropriate for most BRM implementations. **Important:** If the **UseAsRouter** registry entry is set to **True**, this registry entry is not used.	No
TimesTenEnabled	Set this to **False**.	No

Table 4-1 (Cont.) DAT_AccountBatch Registry Entries

Entry	Description	Mandatory
UseAsRouter	If set to **True,** the module is used by the FCT_AccountRouter module to route EDRs to separate Pipeline Manager instances. See "Using Pipeline Manager with Multiple Database Schemas" in *BRM Setting Up Pipeline Rating and Discounting* and "FCT_AccountRouter". If set to **False** (the default), the module is used by the FCT_Account module. **Important:** If set to **True**, the following registry entries are not used: • AcceptLoginSearchFailure • AddAliasList • InitialLoading • ReadAccountBalances • ReadSystemProductFromMain • ThreadAccountHashMapSize • ThreadGroupSharingChargesHashMapSize • ThreadGroupSharingDiscountsHashMapSize • ThreadGroupSharingProfilesHashMapSizes • ThreadServiceHashMapSize • UseProductCreatedTime • UseLatestProductAndDiscount	No
UseLatestProductAndDiscount	If set to **True**, the module retrieves the latest purchased charge offer and discount offer information from the main tables and uses the start and end dates to validate when the charge offer is in effect. If set to **False** (the default), the module retrieves the purchased charge offer and discount offer information from the audit tables and uses the purchase creation date to validate when the charge offer is in effect. **Important:** If the **UseAsRouter** registry entry is set to **True**, this registry entry is not used.	No
UseProductCreatedTime	If set to **True** (the default), the charge offer is selected only if an event occurs after the charge offer's created time (PIN_FLD_CREATED_T) and between its start and end times. If set to **False**, charge offer validity is checked based only on the start and end times (PIN_FLD_START_T and PIN_FLD_END_T) of the charge offer. **Important:** If the **UseAsRouter** registry entry is set to **True**, this registry entry is not used.	No
UseProfileEffectiveTime	If set to **True** (the default), the module uses EFFECTIVE_T to determine the validity of the profile objects. If set to **False**, the module uses CREATED_T to determine the validity of the profile objects.	No

Sample Registry Entry

```
CustomerData
  {
    ModuleName = DAT_AccountBatch
    Module
    {
```

```
    IntegrateConnection  = ifw.DataPool.Login
    InfranetConnection   = ifw.DataPool.LoginInfranet
    LogEvents            = True
    Listener             = ifw.DataPool.Listener
    TimesTenEnabled      = False
    ReadAccountBalances  = True
    Threads              = 4
    Connections          =  5
    LoadPercentage       = 10.0
  }
}
```

Semaphore File Entries

Table 4-2 lists the DAT_AccountBatch Semaphore file entries.

Table 4-2 DAT_AccountBatch Semaphore File Entries

Entry	Description
LogEvents	Specifies whether events should be stored in a log file. You can also use this entry in the startup registry.
PrintData	Reports the account data for all accounts.
PrintDataLogin	Reports the account data for a single account identified by the BRM login ID (usually the phone number).
PrintDataSamples	Reports the account data for a specified number of accounts, chosen randomly.
PrintAmtData	Prints in-memory data about the Account Migration Manager (AMM) to the specified log file.
PrintAmtJobData	Prints in-memory data about one account migration job to the specified log file.
RejectClosedAccounts	Rejects CDRs with a timestamp later than the account's closed date.
Reload	Reloads data from the Pipeline Manager database.

Sample Semaphore File Entry

```
ifw.DataPool.CustomerData.Module.Reload {}
```

Database Tables

The DAT_AccountBatch module uses the following database tables:

- IFW_CURRENCY
- IFW_REF_MAP
- IFW_SERVICE

DAT_AccountRealtime

The DAT_AccountRealtime module provides customer data from the BRM database in a real-time discounting pipeline.

> **Note:**
>
> Unlike the DAT_AccountBatch module, the DAT_AccountRealtime module does not load account data in memory when you start Pipeline Manager. Instead, it gets account data in real time from the BRM database by using the NET_EM module.

The DAT_AccountRealtime module gets data for the FCT_Discount module. For information about the FCT_Discount module, see "FCT_Discount".

Dependencies

The DAT_AccountRealtime requires the NET_EM module. It makes a connection to the NET_EM module automatically; you don't need to configure the connection.

Registry Entries

There are no registry entries for the DAT_AccountRealtime module. You only need to enter the module in the registry DataPool section.

Sample Registry Entry

```
CustomerData
  {
    ModuleName = Dat_AccountRealtime
    Module
    {
      #
    }
  }
```

Semaphore File Entries

DAT_AccountRealtime does not support semaphore updates.

DAT_BalanceBatch

The DAT_BalanceBatch module maintains balance information in the Pipeline Manager memory. It uses account synchronization to retrieve balance information from the BRM database. Data is stored in memory only, not in the database or in a file.

When reading balances and sub-balances from the database, the DAT_BalanceBatch module ignores balance monitor impacts.

See the following documents:

- Configuring Discounting Modules and Components
- FCT_Discount

Dependencies

Requires the following connections:

- Pipeline Manager database.
- BRM database.
- DAT_AccountBatch module. See "DAT_AccountBatch".
- DAT_Listener module. See "DAT_Listener".
- DAT_Discount module. See "DAT_Discount".
- DAT_PortalConfig module. See "DAT_PortalConfig".

Registry Entries

Table 4-3 lists the DAT_BalanceBatch registry entries.

Table 4-3 DAT_BalanceBatch Registry Entries

Entry	Description	Mandatory
AccountDataModule	Specifies the connection to the DAT_AccountBatch module.	Yes
BalanceDirectory	Specifies the directory that contains data and transaction files.	No
BalanceLocks	Specifies the number of locks that can be acquired during processing. Must be a positive integer. Default = **100** **Important**: Setting this value too low may decrease pipeline throughput performance.	No
BalanceLockStatusLog	Specifies that when an event transaction is locked by an EDR transaction, it is logged to the process logger. Default = **False**	No
BalancesPerThreadJobsCount	Specifies the number of jobs per thread. **Important:** Setting the number of jobs per thread to a large number can decrease performance because of the system overhead associated with creating too many jobs. (Typically, three to eight jobs per thread is optimal). If you want to adjust the number of accounts or balances per job, you can do this by increasing or decreasing the number of threads.	Yes
BalanceTrace	Specifies whether to generate a balance trace file. **True** indicates that a balance trace file is generated. **False** indicates that a balance trace file is not generated. Default = **False**	No
CustomEvents	Lists custom business events that include balance data needed by Pipeline Manager. Custom events are defined in the payload configuration file (**payloadconfig_ifw_sync.xml**). See "Configuring Custom Business Events for Pipeline Discounting" in *BRM Setting Up Pipeline Rating and Discounting*.	No
DiscountDataModule	Specifies the connection to the DAT_Discount module.	Yes
InfranetConnection	Specifies the database connection to the BRM database.	Yes
IntegrateConnection	Specifies the database connection to the Pipeline Manager database.	Yes
ListenerDataModule	Specifies the connection to the DAT_Listener module.	Yes

Table 4-3 (Cont.) DAT_BalanceBatch Registry Entries

Entry	Description	Mandatory
LoadPercentage	Specifies how much data to load from the BRM database before the process log outputs status information. For example, to output status after every 10% of the data is loaded, enter **10**. Default = **10**	No
LogEvents	Specifies whether received events should be written to a log file. Use this entry to troubleshoot Pipeline Manager event handling. Default = **False**	No
LogTransactions	Specifies if the balances affected during the CDR processing are logged. Default = **False**	No
PortalConfigDataModule	Specifies the connection to the DAT_PortalConfig module. This enables DAT_BalanceBatch to retrieve business parameter settings from the DAT_PortalConfig module.	Yes
RowFetchSize	Specifies the number of rows of balance data to load from the BRM database for each database retrieving. Default = **50**	No
SelectiveSubBalLoad	Specifies whether to selectively load noncurrency sub-balances at Pipeline Manager startup. Default = **True**	No
Synchronized	Specifies whether to allow the first transaction to process and to make other transactions wait in the queue. Default = **False**	No
ThreadHashMapSize	Specifies the size of the hash map in each thread used for loading balance data from the BRM database. Default = **1024**	No
Threads	Specifies the number of threads for loading the balance data from the BRM database. The number of threads must be smaller than or equal to the number of connections. Default = **4**	No
UseFlexibleConsumptionRule	Specifies whether to use the resource consumption rules defined at the package level. **True**: Uses the consumption rules defined for each resource in a balance group. If a consumption rule is not defined, this module uses the rules defined in the **/config/beid** object. If a consumption rule isn't defined in a balance group or the **/config/beid** object, this module uses the rule defined in the **multi_bal** instance of the **/config/business_params** object. **False**: Uses the system-wide consumption rule defined in the **multi_bal** instance of the **/config/business_params** object only. Default = **True**	No

Table 4-3 (Cont.) DAT_BalanceBatch Registry Entries

Entry	Description	Mandatory
VirtualTime	Specifies whether this module uses system time or virtual time. Default = **False** Set to **True** *only* if you are performing tests and have used the **pin_virtual_time** utility to set a virtual time. If you set this entry to **True**, make sure you copy the **pin.conf** file from the *BRM_home***/sys/test** directory to the *Pipeline_home* directory. *BRM_home* is the directory where you installed BRM components. The **pin.conf** file contains this entry: **- - pin_virtual_time pin_virtual_time_file**	No

Sample Registry Entry

```
BalanceDataModule
  {
    ModuleName = DAT_BalanceBatch
    Module
    {
      IntegrateConnection = ifw.DataPool.Login
      InfranetConnection  = ifw.DataPool.LoginInfranet
      AccountDataModule   = ifw.DataPool.CustomerData
      ListenerDataModule  = ifw.DataPool.Listener
      DiscountDataModule  = ifw.DataPool.DiscountData
      BalanceDirectory    = ./samples/wireless/data/balance
      UseFlexibleConsumptionRule = True
      CustomEvents
      {
        CycleRollover20days
      }
    }
  }
```

Semaphore File Entries

Table 4-4 lists the DAT_BalanceBatch Semaphore file entries.

Table 4-4 DAT_BalanceBatch Semaphore File Entries

Entry	Description
BalanceGroupId	Specifies the ID field of the balance group POID entry. The balance data referenced by **BalanceGroupId** is written into the file specified by the **DataFileName** entry.
DataFileName	Specifies the file name that contains balance data. If the **BalanceGroupId** entry is not present, DAT_BalanceBatch writes all balance data in memory into the file.
LogEvents	Specifies whether events should be stored in a log file. You can also use this entry in the startup registry.
ReloadCreditThresholdParam	Reloads the value from the **CreditThresholdChecking** business parameter.

Sample Semaphore File Entry

```
ifw.DataPool.BalanceDataModule.Module.DataFileName = balData.txt
ifw.DataPool.BalanceDataModule.Module.BalanceGroupId = 70015
ifw.DataPool.BalanceDataModule.Module.LogEvents = True
ifw.DataPool.BalanceDataModule.Module.ReloadCreditThresholdParam{}
```

DAT_BalanceRealtime

The DAT_BalanceRealtime module runs in a real-time discounting pipeline. It retrieves the current balance from the BRM database and supplies the data for real-time discounting.

> **Note:**
>
> Unlike the DAT_BalanceBatch module, the DAT_BalanceRealtime module does not load balance data in memory when you start Pipeline Manager. Instead, it gets balance data in real time from the BRM database by using the NET_EM module.

See the following documents:

- Configuring a Real-Time Discounting Pipeline
- FCT_Discount

Dependencies

The DAT_BalanceRealtime module requires the NET_EM module. It makes a connection to the NET_EM module automatically; you don't need to configure the connection.

Registry Entries

There are no registry entries for the DAT_BalanceRealtime module. You only need to enter the module in the registry DataPool section.

Sample Registry Entry

```
BalanceDataModule
  {
    ModuleName = DAT_BalanceRealtime
    Module
    {
      #
    }
  }
```

Semaphore File Entries

DAT_BalanceRealtime does not support semaphore updates.

DAT_Calendar

The DAT_Calendar module provides special day calendar data for the FCT_MainRating module.

Dependencies

Requires a connection to the Database Connect (DBC) module. See "Database Connect (DBC)".

Registry Entries

Table 4-5 lists the DAT_Calendar registry entries.

Table 4-5 DAT_Calendar Registry Entries

Entry	Description	Mandatory
DataConnection	Specifies the database connection to the Pipeline Manager database.	Yes

Sample Registry Entry

```
Calendar
{
  ModuleName = DAT_Calendar
  Module
  {
    DataConnection = ifw.DataPool.Login
  }
}
```

Semaphore File Entries

Table 4-6 lists the DAT_Calendar Semaphore file entries.

Table 4-6 DAT_Calendar Semaphore File Entries

Entry	Description
Reload	Reloads data from the Pipeline Manager database.

Sample Semaphore File Entry

```
ifw.DataPool.CalendarDataModule.Module.Reload {}
```

Events

Table 4-7 lists the DAT_Calendar events.

Table 4-7 DAT_Calendar Events

Event Name	Trigger	Sender	Parameter
EVT_RELOAD_SUCCESSFUL	Data reload was successful.	DAT_Calendar	None
EVT_RELOAD_FAILED	Data reload failed.	DAT_Calendar	None

Database Tables

The DAT_Calendar module uses the following database tables:

- IFW_CALENDAR
- IFW_HOLIDAY

DAT_ConnectionMonitor

This module creates and monitors the idle timeout period for each connection and maintains the state for each client.

Registry Entries

Table 4-8 lists the DAT_ConnectionMonitor registry entries.

Table 4-8 DAT_ConnectionMonitor Registry Entries

Entry	Description	Mandatory
KeepAliveInterval	The idle timeout value in milliseconds, which specifies how long to wait for a message from the client before sending a Device Watchdog Request (DWR) message to the client. Default is **30000**.	Yes
KeepAliveQueue	Specifies the pipeline queue to which the dummy EDR for the DWR message is sent.	Yes
ShutdownInterval	The idle timeout value in milliseconds, which specifies how long to wait before shutting down after sending a Disconnect-Peer-Request (DPR) message to the client. Default is **1000**.	No
Threads	Number of threads in the pool. Default is **1**.	Yes

Sample Registry Entry

```
ConnectionMonitor
{
  ModuleName = DAT_ConnectionMonitor
  Module
  {
    Threads = 1
    KeepAliveInterval = 30000
    ShutdownInterval = 1000
    KeepAliveQueue = ifw.IPCQueues.INOutputQueue
```

```
    }
}
```

Semaphore File Entries

DAT_ConnectionMonitor does not support semaphore updates.

DAT_ConnectionPool

DAT_ConnectionPool module has a set of configured Connection Manager (CM) connections, which the "FCT_Opcode" module uses to connect to the CM and call the appropriate opcode.

For each CM, the DAT_ConnectionPool module maintains a queue for spare connections, determined by the size of the queue.

The DAT_ConnectionPool module balances the load by distributing the required load among the pipelines using the FCT_Opcode module and when there is a CM failure, redistributes the load among the active CMs.

> **Note:**
>
> If a connection fails, the processing pipeline connects to the spare connection of the CM to which it initially connected; if the CM is down, it tries to use a connection from a different CM queue.

If an idle processing pipeline is connected to an inactive CM, The DAT_ConnectionPool module sets the connection status to inactive and updates the pipeline recycle flag, so that when the pipeline starts processing a request it can connect to an active CM.

The DAT_ConnectionPool module initializes the Global Data Dictionary (GDD) during startup by accessing the database.

> **Note:**
>
> If CMs are not available, it uses a file containing the data dictionary flist to initialize the GDD.

Registry Entries

Table 4-9 lists the DAT_ConnectionPool registry entries.

Table 4-9 DAT_ConnectionPool Registry Entries

Entry	Description	Mandatory
InfranetDataDictionaryFileName	The file containing the data dictionary. If a CM is not available, DAT_ConnectionPool uses this file to start and initialize the GDD. If you don't specify a file name, the DAT_ConnectionPool uses the default file, **./gddDataFile.dat**, where it stored the data dictionary flists at the initial startup.	No

Table 4-9 (Cont.) DAT_ConnectionPool Registry Entries

Entry	Description	Mandatory
IdleConnectionBuffer	Size of the queue for spare connections.	Yes
FullQueueTimeout	The interval, in seconds, in which the worker thread pings the queue to check if there is space available in the queue for a connection, when the queue is full. Default is 10 seconds.	No
EmptyQueueTimeout	The interval, in seconds, in which the pipeline thread pings the queue to check if there is a connection available in the queue, when the queue is empty. It can happen during startup of the pipeline or when the CM connection is not working as expected, for example, the CM times out. Default is 1 second.	No
InfranetPool	Specifies the CMs in the connection pool. For each CM in the pool, define the following entries: • Host name (**Host =** *CM1_host)* • Port number (**Port =***CM1_port)* • Login name and password for logging into BRM (**LoginName =** *root.0.0.0.1* and • **LoginPassword =***password)* • Whether to log debug messages (**Logging =** *False*. Values are **True** and **False**. The default is **False**.) • CM response timeout in milliseconds (**SocketTimeOut =** *30000*.) • Whether to enable SSL (**SSLEnabled =** *false*. Values are **True** and **False**. The default is **False**.)	No

Sample Registry Entry

```
DataPool
{

   CMConnectionPool
  {
    ModuleName = DAT_ConnectionPool
    Module
    {
      InfranetDataDictionaryFileName = File_with_DD_objects
      IdleConnectionBuffer
      {
        Size = 2
       }
      FullQueueTimeout = number_of_seconds
      EmptyQueueTimeout = number_of_seconds
      InfranetPool
     {
       CM1
         {
           Host = CM1_host
           Port = CM1_port
           LoginName = root.0.0.0.1
           LoginPassword = password
           # Default is false
           SSLEnabled = TRUE
```

```
            # For logging login flist
            # Default is false
            Logging = True
            SocketTimeOut = 30000
          }
       CM2
          {
            Host = CM2_host
            Port = CM2_port
            LoginName = root.0.0.0.1
            LoginPassword = password
            # Default is false
            SSLEnabled = True
            # For logging login flist
            # Default is false
            Logging = True
            SocketTimeOut = 30000
           }
       }
    }
}
```

Semaphore File Entries

DAT_ConnectionPool does not support semaphore updates.

DAT_Currency

The DAT_Currency module converts currency symbols to numeric values and retreives resource rounding rules, using data from **/config/beid** objects in the BRM database.

Dependencies

Requires a connection to the BRM database.

Registry Entries

Table 4-10 lists the DAT_Currency registry entries.

Table 4-10 DAT_Currency Registry Entries

Entry	Description	Mandatory
DataConnection	Specifies the database connection to the Pipeline Manager database.	Yes
InfranetConnection	Specifies the database connection to the BRM database.	Yes
ReuseOnFailure	Specifies whether the module should continue to use the old data if the **Reload** command fails. If **True**, the old data is used. If the entry is not used, the default is **False**.	No

Sample Registry Entry

```
DAT_Currency
{
```

```
  ModuleName = DAT_Currency
  Module
  {
    ReuseOnFailure = TRUE
    InfranetConnection  = ifw.DataPool.LoginInfranet
    DataConnection = ifw.DataPool.Login
  }
}
```

Semaphore File Entries

Table 4-11 lists the DAT_Currency Semaphore file entries.

Table 4-11 DAT_Currency Semaphore File Entries

Entry	Description
Reload	Reloads data from the Pipeline Manager database.

Sample Semaphore File Entry

```
ifw.DataPool.CurrencyDataModule.Module.Reload ()
```

DAT_Dayrate

The DAT_Dayrate module provides special day rate data for the FCT_Dayrate module.

Dependencies

Requires a connection to the Pipeline Manager database.

Registry Entries

Table 4-12 lists the DAT_Dayrate registry entries.

Table 4-12 DAT_Dayrate Registry Entries

Entry	Description	Mandatory
Buffer	Specifies the size of the internal data buffer.	Yes
DataConnection	Specifies the database connection to the Pipeline Manager database.	Yes

Sample Registry Entry

```
Dayrate
{
  ModuleName = DAT_Dayrate
  Module
  {
    DataConnection = ifw.DataPool.Login
    Buffer = 5000
  }
}
```

Semaphore File Entries

Table 4-13 lists the DAT_Dayrate Semaphore file entries.

Table 4-13 DAT_Dayrate Semaphore File Entries

Entry	Description
Reload	Reloads data from the Pipeline Manager database.

Sample Semaphore File Entry

```
ifw.DataPool.DayRateDataModule.Module.Reload {}
```

Events

Table 4-14 lists the DAT_Dayrate events.

Table 4-14 DAT_Dayrate Events

Event Name	Trigger	Sender	Parameter
EVT_RELOAD_SUCCESSFUL	Data reload was successful.	DAT_Dayrate	None
EVT_RELOAD_FAILED	Data reload failed.	DAT_Dayrate	None

Database Tables

The DAT_Dayrate module uses the following database tables:

- IFW_SPECIALDAYRATE
- IFW_SPECIALDAY_LNK

DAT_Discount

The DAT_Discount module provides discount data for the FCT_Discount module. See "FCT_Discount".

Dependencies

Requires a connection to:

- The Pipeline Manager database.
- The BRM database.
- The DAT_AccountRealtime or DAT_AccountBatch module.
- The DAT_ModelSelector module.
- The DAT_PortalConfig module.

Registry Entries

Table 4-15 lists the DAT_Discount registry entries.

Table 4-15 DAT_Discount Registry Entries

Entry	Description	Mandatory
AccountDataModule	Specifies the connection to the DAT_AccountRealtime or DAT_AccountBatch module.	Yes
InfranetConnection	Specifies the database connection to the BRM database.	Yes
EvalScriptFiles	Specify name-value pairs for one or more iScript files. The name is a unique string used to identify the script if there is an error. The value is the relative or absolute path of the file. The iScript files specified in this entry contain functions that can be referenced via **EVAL** tokens in discount expressions. Any number of files can be specified.	No
IntegrateConnection	Specifies the database connection to the Pipeline Manager database.	Yes
PortalConfigDataModule	Specifies the connection to the DAT_PortalConfig module. This enables DAT_Discount to retrieve business parameter settings from the DAT_PortalConfig module.	Yes

Sample Registry Entry

```
ifw
{ ....
  DataPool
  { ....
    DiscountModelDataModule
    {
      ModuleName = DAT_Discount
      Module
      {
        IntegrateConnection = ifw.DataPool.Login
        InfranetConnection = ifw.DataPool.LoginInfranet
        AccountDataModule = ifw.DataPool.CustomerData
        #Customizable iScript files supporting EVAL function
        EvalScriptFiles
        {
           scriptFile1 = ../iScriptLib/iScriptLib_Samples/ISC_GetMostCalledInfo.isc
           scriptFile2 = ../iScriptLib/iScriptLib_Samples/ISC_GetLastSixMonthCharge.isc
        }
      }
    }
  }
}
```

Semaphore File Entries

Table 4-16 lists the DAT_Discount Semaphore file entries.

Table 4-16 DAT_Discount Semaphore File Entries

Entry	Description
Reload	Reloads data from the Pipeline Manager database. **Note:** Discounting data is reloaded only when the discount configuration contains a new noncurrency resource.
ReloadEvalScripts	Reloads and recompiles the iScript files specified in the **EvalScriptFiles** registry entry.
DiscountModel	The discounts whose data you want written to the output. This entry is used for troubleshooting purposes and must be used in conjunction with the **DataFileName** semaphore. • **ALL**: Writes all discount codes (such as model codes, rule codes, step codes, and so on) and related configuration information for all discounts in your system. • *Discount_model_code*: Writes the discount codes and related configuration information associated with the specified discount code. **Note:** You cannot specify multiple discount codes. You can specify only a single code or ALL.
DataFileName	Where discount information should be written. This entry is used for troubleshooting purposes and must be used in conjunction with the **DiscountModel** semaphore. • To write the information to a file, specify a file name. By default, the file is created in the *Pipeline_home* directory. • To write the information to the terminal, leave the value of this entry blank.

Sample Semaphore File Entry

- To reload data from the database and to reload and recompile iScript files:

```
ifw.DataPool.DiscountModelDataModule.Module.Reload {}
ifw.DataPool.DiscountModelDataModule.Module.ReloadEvalScripts=True
```

- To write all discount configuration information to a file named **DiscountConfig.log**:

```
ifw.DataPool.DiscountModelDataModule.Module.DiscountCode = ALL
ifw.DataPool.DiscountModelDataModule.Module.DataFileName = DiscountConfig.log
```

- To write the configuration information for a discount with the code **DM10%off** to the terminal:

```
ifw.DataPool.DiscountModelDataModule.Module.DiscountCode = DM10%OFF
ifw.DataPool.DiscountModelDataModule.Module.DataFileName {}
```

Database Tables

The DAT_Discount module uses the following database tables:

- IFW_DISCOUNTMODEL
- IFW_DSCMDL_VER
- IFW_DSCMDL_CNF
- IFW_DSCTRIGGER
- IFW_DSCCONDITION
- IFW_DISCOUNTMASTER
- IFW_DISCOUNTDETAIL

- IFW_DISCOUNTRULE
- IFW_DISCOUNTSTEP
- IFW_DSCBALIMPACT

> **Note:**
>
> For information on compare patterns used in database values, see "About Using Regular Expressions when Specifying the Data to Extract" in *BRM Setting Up Pipeline Rating and Discounting*.

DAT_ExchangeRate

The DAT_ExchangeRate module provides currency exchange rate data for the FCT_ExchangeRate module.

Dependencies

Requires a connection to the Pipeline Manager database.

Registry Entries

Table 4-17 lists the DAT_ExchangeRate registry entries.

Table 4-17 DAT_ExchangeRate Registry Entries

Entry	Description	Mandatory
DataConnection	Specifies the database connection to the Pipeline Manager database.	Yes
ReuseOnFailure	Specifies if the module should continue to use the old data if the **Reload** command fails. If **True**, the old data is used. If the entry is not used, the default is **False**.	No

Sample Registry Entry

```
ExchangeRateData
{
    ModuleName = DAT_ExchangeRate
    Module
    {
      DataConnection = ifw.DataPool.Login
      ReuseOnFailure = True
    }
}
```

Semaphore File Entries

Table 4-18 lists the DAT_ExchangeRate Semaphore file entries.

Table 4-18 DAT_ExchangeRate Semaphore File Entries

Entry	Description
Reload	Reloads data from the Pipeline Manager database.

Sample Semaphore File Entry

```
ifw.DataPool.ExchangeRateDataModule.Module.Reload {}
```

Events

Table 4-19 lists the DAT_ExchangeRate events.

Table 4-19 DAT_ExchangeRate Events

Event Name	Trigger	Sender	Parameter
EVT_RELOAD_SUCCESSFUL	Data reload was successful.	DAT_ExchangeRate	None
EVT_RELOAD_FAILED	Data reload failed.	DAT_ExchangeRate	None

Database Tables

The DAT_ExchangeRate module uses the IFW_EXCHANGE_RATE database table.

> **Note:**
>
> For information on compare patterns used in database values, see "About Using Regular Expressions when Specifying the Data to Extract" in *BRM Setting Up Pipeline Rating and Discounting*.

DAT_InterConnect

The DAT_InterConnect module caches InterConnect and roaming related configuration data. This information is used by FCT_CarrierIcRating module.

> **Note:**
>
> Use the OpFlagExt iScript extension in iScripts to call this module directly to obtain the network operator taxation value flag.

Dependencies

Requires a connection to the Pipeline Manager database.

Registry Entries

Table 4-20 lists the DAT_InterConnect registry entries.

Table 4-20 DAT_InterConnect Registry Entries

Entry	Description	Mandatory
DataConnection	Specifies the database connection to the Pipeline Manager database.	Yes
LoadPoiAreas	If **True**, load data from the IFW_POIAREA_LNK table for reseller interconnection network models.	No
ReuseOnFailure	Specifies if the module should continue to use the old data if the **Reload** command fails. If **True**, the old data is used. If the entry is not used, the default is **False**.	Yes

Sample Registry Entry

```
InterConnect
{
  ModuleName = DAT_InterConnect
  Module
  {
    DataConnection = ifw.DataPool.Login
    ReuseOnFailure = False
  }
}
```

Semaphore File Entries

Table 4-21 lists the DAT_InterConnect Semaphore file entries.

Table 4-21 DAT_InterConnect Semaphore File Entries

Entry	Description
Reload	Reloads data from the Pipeline Manager database.
LoadPoiAreas	Specifies if the module should load data from the IFW_POIAREA_LNK table for reseller interconnection network models.
ReuseOnFailure	Specifies if the module should continue to use the old data if the **Reload** command fails. If **True**, the old data is used. If the entry is not used, the default is **False**.

Sample Semaphore File Entry

```
ifw.DataPool.InterConnect.Module.Reload {}
ifw.DataPool.InterConnectDataModule.Module.ReuseOnFailure = True
ifw.DataPool.InterConnectDataModule.Module.LoadPoiAreas = True
```

Database Tables

The DAT_InterConnect module uses the following database tables:

- IFW_NETWORKOPER

- IFW_NETWORKMODEL
- IFW_ICPRODUCT
- IFW_ICPRODUCT_RATE
- IFW_ICPRODUCT_GRP
- IFW_ICPRODUCT_CNF

Data for interconnect rating is stored in the following tables:

- IFW_SWITCH
- IFW_POI
- IFW_TRUNK
- IFW_TRUNK_CNF
- IFW_POIAREA_LNK

> **Note:**
>
> For information on compare patterns used in database values, see "About Using Regular Expressions when Specifying the Data to Extract" in *BRM Setting Up Pipeline Rating and Discounting*.

DAT_ItemAssign

The DAT_ItemAssign module returns the item POID for an item tag to the FCT_ItemAssign and FCT_Billing Record modules.

Dependencies

Requires a connection to the BRM database and the "DAT_AccountBatch" module.

Registry Entries

Table 4-22 lists the DAT_ItemAssign registry entries.

Table 4-22 DAT_ItemAssign Registry Entries

Entry	Description	Mandatory
AccountDataModule	Specifies the connection to the DAT_AccountBatch module.	Yes
InfranetConnection	Specifies the connection to the BRM database.	Yes
ItemPoidReservedRangeUnitSize	Specifies the maximum number of POIDs to be reserved. Default =**10000**	No

Sample Registry Entry

```
Flexible item assignment data module

ItemAssignDataModule
```

```
{
  ModuleName = DAT_ItemAssign
  Module
  {
   InfranetConnection = ifw.DataPool.LoginInfranet
   AccountDataModule = ifw.DataPool.CustomerData
   ItemPoidReservedUnitSize = 10000
  }
}
```

Semaphore File Entries

Table 4-23 lists the DAT_ItemAssign Semaphore file entries.

Table 4-23 DAT_ItemAssign Semaphore File Entries

Entry	Description
PrintData	Generates a log file that contains the item tag-to-type mapping information.
Reload	Reloads data from the Pipeline Manager database.

Sample Semaphore File Entry

```
ifw.DataPool.ItemAssignDataModule.Module.PrintData=TagTypeMap.txt
ifw.dataPool.ItemAssignDataModule.Module.Reload {}
```

DAT_Listener

The DAT_Listener module dequeues business events from a database queue and then sends the data to the DAT_AccountBatch and DAT_Discount modules.

The DAT_Listener module also posts acknowledgment events to the acknowledgment queue in response to business events sent by **pin_rerate**.

DAT_Listener module controls whether the Pipeline Manager processes business events or CDRs by interleaving the two processes. You can configure the DAT_Listener module for concurrent or interleaved processing.

Dependencies

Requires a connection to the database containing the Oracle Advanced Queuing (AQ) database queue.

The Listener section of the registry file must be listed after the Pipeline Manager and BRM database connection sections. Otherwise, the Pipeline Manager fails to start.

Registry Entries

Table 4-24 lists the DAT_Listener registry entries.

Table 4-24 DAT_Listener Registry Entries

Entry	Description	Mandatory
AckQueueNameAMM	Specifies the name of the acknowledgment queue for posting AMM-related acknowledgment events.	Yes This entry is mandatory for AMM operations.
AckQueueName	Specifies the name of the acknowledgment queue for posting rerating related acknowledgment events. Default = **RERATING_ACK_QUEUE**	Yes This entry is mandatory for rerating using **pin_rerate**.
BatchSize	The maximum number of events to be dequeued in each dequeuing operation. Default = 10	No
EventsPath	Specifies the directory location of the file that stores the event information retrieved by DAT_Listener if the LogEvents entry is set to **True**. Default location is the directory where the ifw is launched.	No
EventsPrefix	Specifies the prefix of the name of the file that stores the event information retrieved by DAT_Listener if the LogEvents entry is set to **True**. Default =**listenerLog**	No

EventThreadAllocation	Defines the number of threads (in addition to one default thread) to use for dequeuing specific business events. For example:
	`EventThreadAllocation`
	`{`
	`  RecycleRequest = 1`
	`  OpenNewActgCycle = 2`
	`}`
	uses 4 threads: one for RecycleRequest business events, two for OpenNewActgCycle business events, and one default thread for dequeuing all other types of business events.
		No

InfranetConnection	Specifies the connection to a database with the Oracle Advanced Queuing (AQ) database queue. Default = **ifw.DataPool.LoginInfranet**	Yes
LogEvents	Specifies whether received events should be written to a log file. Default = **False**	No
NumOfRetries	Specifies the number of times the DAT_Listener module retries to connect to the database queue. Default = 10	No
QueueLibrary	Specifies whether the DAT_Listener module is configured for Oracle AQ. Set QueueLibrary to **OracleQueue**.	Yes
QueueName	Specifies the name of the database queue from which the DAT_Listener module retrieves events. Default = **IFW_SYNC_QUEUE**	Yes

Table 4-24 (Cont.) DAT_Listener Registry Entries

Entry	Description	Mandatory
RetryInterval	Specifies the time in seconds that the DAT_Listener module waits before trying to reconnect to the database specified in **InfranetConnection**. Default = **5**	No

Registry Entries for Interleaved Processing

The following are registry entries used for interleaved processing.

> **Note:**
>
> The default values for interleaved processing are also the minimum required values. If you specify a value less than the default for any entry, that value is ignored and the minimum default value is used.

Table 4-25 lists the DAT_Listener registry entries for interleaved processing.

Table 4-25 DAT_Listener Registry Entries for Interleaved Processing

Entry	Description	Mandatory
CheckInterval	Specifies (in seconds) how frequently the DAT_Listener module checks the number of events waiting in the queue. If this entry is not present, the default frequency check is used. **Important:** This entry takes precedence over **MaxNumEvents**, **MinNumEvents**, **MaxEventProcessTime**, and **MaxCDRProcessTime**. For example, if **MaxEventProcessTime** is set to 3600 seconds, and **CheckInterval** is set to 7200 seconds, events are processed for 7200 seconds. Default = **60**	No
EnableInterLeaving Statistics	Specifies whether to log only interleaving statistical data. If set to **False**, all processing messages are logged. Default = **False**	No
InterleavingReqd	Specifies whether interleaved processing is enabled: **True** = Enabled **False** = Not enabled When set to **False** or not specified, interleaved processing is not performed; CDRs and events are processed simultaneously. Default = **False**	No
MaxCDRProcessTime	Specifies the maximum number of seconds that CDRs are processed. When the pipeline has been processing CDRs for this amount of time, the DAT_Listener module stops CDR processing and starts business event processing regardless of how many business events are in the queue. Default and minimum allowed = **300**	If **MaxEventProcessTime** is specified and **InterleavingReqd** is set to TRUE, yes. Otherwise, no.

Table 4-25 (Cont.) DAT_Listener Registry Entries for Interleaved Processing

Entry	Description	Mandatory
MaxEventProcessTime	Specifies the maximum number of seconds that business events are processed. When the pipeline has been processing business events for this amount of time, the DAT_Listener module stops business event processing and starts CDR processing regardless of how many business events are in the queue. Default and minimum allowed = **60**	If **MaxCDRProcessTime** is specified and **InterleavingReqd** is set to TRUE, yes. Otherwise, no.
MaxNumEvents	Specifies the maximum number of business events allowed in the queue. When the number of events in the queue reaches or exceeds this amount, DAT_Listener stops pipeline CDR processing and starts business event processing. Default and minimum allowed = **900**	Yes, if **InterleavingReqd** is set to **True**. Otherwise, no. Requires that you also specify **MinNumEvents** and **MaxNumEvents** is greater than **MinNumEvents**.
MinNumEvents	Specifies the minimum number of business events allowed in the queue. When the number of events in the queue reaches or drops below this amount, the DAT_Listener stops business event processing and starts CDR processing. Default and minimum allowed = **300**	Yes, if **InterleavingReqd** is set to **True**. Otherwise, no.
ProcessAllEvents	Specifies whether to process all business events in the queue when Pipeline Manager is started: **True** = Processes all business events in the queue before activating interleaved processing. **False** = Interleaved processing is activated at startup. Business events are processed according to the interleaving settings. If set to **True** at startup, after processing all business events, this entry is reset to **False**. To process all business events at startup, you must reset this entry to **True** each time you restart Pipeline Manager. Default = **False**	No

Sample Registry Entry

```
Listener
{
    ModuleName = DAT_Listener
    Module
    {
       InfranetConnection = ifw.DataPool.LoginInfranet
       QueueLibrary = OracleQueue
       QueueName = IFW_SYNC_QUEUE
       NumOfRetries = 1
       RetryInterval = 5
       LogEvents = TRUE

       InterleavingReqd = true
       MaxNumEvents = 900
       MinNumEvents = 300
       CheckInterval = 60
       EnableInterLeavingStatistics = false
       ProcessAllEvents = true
       MaxEventProcessTime = 60
```

```
            MaxCDRProcessTime = 300
        }
    }
```

Semaphore File Entries

Table 4-26 lists the DAT_Listener Semaphore file entries.

Table 4-26 DAT_Listener Semaphore File Entries

Entry	Description
AckQueueNameAMM	Specifies the name of the Acknowledgment queue for posting AMM-related acknowledgment events. To add or modify this entry without having to stop the pipeline, use the **Disconnect** semaphore to disconnect the Listener before setting it. After specifying the queue name, use the **Connect** semaphore to reconnect the Listener to the pipeline for the new value to take effect.
CheckInterval	Specifies (in seconds) how frequently the DAT_Listener module checks the number of events waiting in the queue. If this entry is not present, the default frequency check is used.
Connect{}	Reconnects the DAT_Listener module event dequeuing threads to the AQ database queue.
Disconnect{}	Disconnects the DAT_Listener module event dequeuing threads from the AQ database queue. The module checks the dequeuing threads before disconnecting them: • If a thread is in the middle of processing an event, the module waits until the pipeline finishes processing events, and then suspends and disconnects the thread from the queuing database. • If a thread *is not* in the middle of processing an event, the module suspends and disconnects the thread from the queuing database.
EnableDequeueStatistics	Specifies whether to log dequeue statistics in the process log. **TRUE** = Log dequeue statistics **FALSE** = Do not log dequeue statistics (Default) **Note:** When set to **TRUE**, the size of the process log increases. Additionally, there may be a performance impact due to file input and output processing. Use this entry for diagnostic purposes only and should not be used otherwise.
EnableInterLeavingStatistics	Specifies whether to log only interleaving statistical data. If set to **False**, all processing messages are logged.
LogEvents	Specifies whether received events should be written to a log file. Default = **False**
MaxCDRProcessTime	Specifies the maximum number of seconds that CDRs are processed. When the pipeline has been processing CDRs for this amount of time, the DAT_Listener module stops CDR processing and starts business event processing regardless of how many business events are in the queue. Required when **MaxEventProcessTime** is specified. Requires that you also specify **MaxNumEvents**, **MinNumEvents**, and **MaxEventProcessTime**.

Table 4-26 (Cont.) DAT_Listener Semaphore File Entries

Entry	Description
MaxEventProcessTime	Specifies the maximum number of seconds that business events are processed. When the pipeline has been processing business events for this amount of time, the DAT_Listener module stops business event processing and starts CDR processing regardless of how many business events are in the queue. Required when **MaxCDRProcessTime** is specified. Requires that you also specify **MaxNumEvents**, **MinNumEvents**, and **MaxCDRProcessTime**.
MaxNumEvents	Specifies the maximum number of business events allowed in the queue. When the number of events in the queue reaches this amount, the DAT_Listener module stops pipeline CDR processing and starts business event processing. Required when **MinNumEvents**, **MaxEventProcessTime**, or **MaxCDRProcessTime** is specified. Requires that you also specify **MinNumEvents**.
MinNumEvents	Specifies the minimum number of business events allowed in the queue. When the number of events in the queue reaches this amount, the DAT_Listener module stops business event processing and starts CDR processing. Required when **MaxNumEvents**, **MaxEventProcessTime**, or **MaxCDRProcessTime** is specified. Requires that you also specify **MaxNumEvents**.

Sample Semaphore File Entry

```
ifw.DataPool.Listener.Module.CheckInterval=180
ifw.DataPool.Listener.Module.EnableInterLeavingStatistics=true
ifw.DataPool.Listener.Module.Disconnect{}
ifw.DataPool.Listener.Module.Connect{}
```

DAT_ModelSelector

When a model selector is used to rate or discount an EDR, the DAT_ModelSelector module evaluates the model selector rules to determine the correct price or discount. The rules are evaluated in the order they are ranked in the model selector.

The following rating and discounting modules get the model information from DAT_ModelSelector:

- FCT_MainRating gets the pricing from DAT_ModelSelector. See "FCT_MainRating".
- FCT_DiscountAnalysis gets the discount from DAT_ModelSelector. See "FCT_DiscountAnalysis".

Dependencies

Requires a connection to the Database Connect (DBC) module. See "Database Connect (DBC)".

The module uses event notification to refresh customized charge offer data. You must configure a connection to DAT_Listener if you plan to use this feature. See "DAT_Listener".

Registry Entries

Table 4-27 lists the DAT_ModelSelector registry entries.

Table 4-27 DAT_ModelSelector Registry Entries

Entry	Description	Mandatory
IntegrateConnection	Specifies the connection to the Pipeline Manager database. This typically points to the login registry section. For example: `IntegrateConnection = ifw.DataPool.Login`	Yes
ListenerDataModule	Specifies the connection to the DAT_Listener module.	No

Sample Registry Entry

```
ModelSelectorDataModule
    {
      ModuleName = DAT_ModelSelector
      Module
      {
     ListenerDataModule  = ifw.DataPool.Listener
        IntegrateConnection = ifw.DataPool.Login
        LogEvents           = True
      }
```

Semaphore File Entries

Table 4-28 lists the DAT_ModelSelector Semaphore file entries.

Table 4-28 DAT_ModelSelector Semaphore File Entries

Entry	Description
Reload	Reloads data from the Pipeline Manager database.
DiscountModel	The discounts whose data you want written to the output. This entry is used for troubleshooting purposes and must be used in conjunction with the **DataFileName** semaphore. • **ALL**: Writes all discount codes (such as model codes, rule codes, step codes, and so on) and related configuration information for all discounts in your system. • *Discount_model_code*: Writes the discount codes and related configuration information associated with the specified discount code. **Note:** You cannot specify multiple discount codes. You can specify only a single code or **ALL**.
DataFileName	Where discount information should be written. This entry is used for troubleshooting purposes and must be used in conjunction with the **DiscountModel** semaphore. • To write the information to a file, specify a file name. By default, the file is created in the *Pipeline_home* directory. • To write the information to the terminal, leave the value of this entry blank.

Sample Semaphore File Entry

To reload data from the database and to reload and recompile iScript files:

```
ifw.DataPool.ModelSelectorDataModule.Module.Reload {}
```

To write all model selector configuration information to a file named **ModelSelectorConfig.log**:

```
ifw.DataPool.ModelSelectorDataModule.Module.ModelSelectorCode = ALL

ifw.DataPool.ModelSelectorDataModule.Module.DataFileName = ModelSelectorConfig.log
```

To write the configuration information for a model selector with the code **DMS10%off** to the terminal:

```
ifw.DataPool.ModelSelectorDataModule.Module.ModelSelectorCode = DMS10%off
ifw.DataPool.ModelSelectorDataModule.Module.DataFileName {}
```

Database Tables

The DAT_ModelSelector module uses the following database tables:

- IFW_MODEL_SELECTOR. This table stores all model selector information in the Pipeline Manager database. It has a type field to indicate whether a model selector is for discounting or rating.
- IFW_SELECTOR_RULESET. This table maps model selector rules to specific model selectors. Rules associated with a model selector are ranked in order of priority.
- IFW_SELECTOR_RULE. This table stores information for each model selector rule, including the code, name, and rule links.
- IFW_SELECTOR_RULE_LNK. This table maps a model selector rule to its detail or block.
- IFW_SELECTOR_DETAIL. This table stores each model selector's rule details; the EDR field and value.
- IFW_SELECTOR_BLOCK. This table stores block information for a model selector rule. This table is for future use.
- IFW_SELECTOR_BLOCK_LNK. This table maps a block to a selector detail or to another block. This table is for future use.

> **Note:**
>
> For information on compare patterns used in database values, see "About Using Regular Expressions when Specifying the Data to Extract" in *BRM Setting Up Pipeline Rating and Discounting*.

DAT_NOSP

The DAT_NOSP module provides data for mapping network source and destinations to new values for the FCT_NOSP module.

See the following documents:

- Identifying the Network Operator/Service Provider

- FCT_NOSP

Dependencies

DAT_NOSP module supports both file and database option.

If configured to get data from the database, the DAT_NOSP module requires a connection to the Pipeline Manager database.

Registry Entries

Table 4-29 lists the DAT_NOSP registry entries.

Table 4-29 DAT_NOSP Registry Entries

Entry	Description	Mandatory
DataConnection	Specifies the database connection to the Pipeline Manager database.	Yes, if the data is stored in the database. Otherwise is not used.
FileName	Specifies the path and file name of the initialization file. See "Creating an NO/SP Data File" in *BRM Setting Up Pipeline Rating and Discounting*. You can use this entry in a semaphore file.	Yes, if the data is stored in a file. Otherwise is not used.
ReuseOnFailure	Specifies if the module should continue to use the old data if the **Reload** command fails. If **True**, the old data is used. If the entry is not used, the default is **False**.	Yes
Source	Specifies where the data is stored: • **File** • **Database**	Yes

Sample Registry Entry for the Database Interface

```
NospData
{
  ModuleName = DAT_NOSP
  Module
  {
    Source = Database
    DataConnection = ifw.DataPool.Login
    ReuseOnFailure = FALSE
  }
}
```

Sample Registry Entry for the File Interface

```
NOSP
{
  ModuleName = DAT_NOSP
  Module
  {
    ReuseOnFailure = FALSE
    Source = File
    FileName = ./cfg/NOSP_Config1.dat
```

```
  }
}
```

Format of the file:

```
RANK;OLD_SOURCE;OLD_DESTINATION;A_PREFIX;NEW_SOURCE;NEW_DESTINATION;
```

For example,

```
ALL_RATE;1;ABC;BCD;0987;XYZ;YZA;
```

Semaphore File Entries

Table 4-30 lists the DAT_NOSP Semaphore file entries.

Table 4-30 DAT_NOSP Semaphore File Entries

Entry	Description
FileName	Specifies the path and file name of the initialization file.
Reload	Reloads data from the Pipeline Manager database. Only used if data is stored in the database.

Sample Semaphore File Entry for the Database Interface

```
ifw.DataPool.NOSP.Module.Reload {}
```

Sample Semaphore File Entry for the File Interface

```
ifw.DataPool.NOSP.Module.FileName = ./cfg/NOSP_Config2.dat
```

Database Tables

The DAT_NOSP module uses the following tables:

- IFW_NOSP
- IFW_GROUP

DAT_NumberPortability

The DAT_NumberPortability module provides number portability data to the FCT_NumberPortability module.

Registry Entries

Table 4-31 lists the DAT_NumberPortability registry entries.

Table 4-31 DAT_NumberPortability Registry Entries

Entry	Description	Mandatory
CountryCode	Specifies the country code, for example **49** for Germany. This is needed for normalization of CLIs. See "Configuring Normalization for Number Portability" in *BRM Setting Up Pipeline Rating and Discounting*.	Yes
FileName	Specifies the name of the number portability file. See "Creating a Number Portability Data File" in *BRM Setting Up Pipeline Rating and Discounting*.	Yes
InternationalAccessCode	Specifies the international access code. This is needed for normalization of CLIs. Default = 00 See "Configuring Normalization for Number Portability" in *BRM Setting Up Pipeline Rating and Discounting*.	No
InternationalAccessCodeSign	Specifies the international access code sign. This is needed for normalization of CLIs. Default = + See "Configuring Normalization for Number Portability" in *BRM Setting Up Pipeline Rating and Discounting*.	No
NationalAccessCode	Specifies the national access code, for example **0** for Germany. This is needed for normalization of CLIs. See "Configuring Normalization for Number Portability" in *BRM Setting Up Pipeline Rating and Discounting*.	Yes
ReuseOnFailure	Specifies if the module should continue to use the old data if the **Reload** command fails. If **True**, the old data is used. If the entry is not used, the default is **False**.	No
SearchMethod	Specifies which search method to use. • **0** specifies to use best match. • **1** specifies to use exact match. • **2** specifies to use first prefix match. Default = **0** See "Configuring Number Portability Search" in *BRM Setting Up Pipeline Rating and Discounting*.	No

Sample Registry Entry

```
NumberPortabilityData
{
  ModuleName = DAT_NumberPortability
  Module
  {
    FileName = ./data/primary_np.data
    CountryCode = 49
    NationalAccessCode = 0
    SearchMethod = 0
  }
}
```

Semaphore File Entries

Table 4-32 lists the DAT_NumberPortability Semaphore file entries.

Table 4-32 DAT_NumberPortability Semaphore File Entries

Entry	Description
AdditionalNumPortData	Specifies the name of the ASCII file that contains the newly ported numbers to reload. **Important:** This parameter must not be used with the Reload semaphore entry. Otherwise, an error message is logged and nothing is updated.
PrintData	Specifies the name of the ASCII file in which to print all newly ported numbers. Important: If this entry is specified with the **Reload** or **AdditionalNumPortData** semaphore entries, the **PrintData** entry is processed last.
Reload	Reloads data from the number portability data file. Important: You cannot use this entry at the same time as the **AdditionalNumPortData** semaphore entry.

Sample Semaphore File Entry

```
ifw.DataPool.NumberPortability.Module.Reload {}
ifw.DataPool.NumberPortability.Module.AdditionalNumPortData=./data/primary_np.data
ifw.DataPool.NumberPortability.Module.PrintData=./data/records.nport.dump
```

Events

Table 4-33 lists the DAT_NumberPortability Events.

Table 4-33 DAT_NumberPortability Events

Event Name	Trigger	Sender	Parameter
EVT_ADD_NUM_PORT_DATA_SUCCESSFUL	Adding new number Portability data succeeded.	DAT_NumberPortability	None
EVT_ADD_NUM_PORT_DATA_FAILED	Adding new number portability data failed.	DAT_NumberPortability	None

DAT_PortalConfig

The DAT_PortalConfig module loads data from the **/config/event_order_criteria**, **/config/business_params**, and **/config/credit_profile** objects in the BRM database.

Dependencies

This module requires a connection to the Database Connect (DBC) module. See "Database Connect (DBC)".

> **Note:**
>
> Due to the dependency of other data modules on DAT_PortalConfig, the DAT_PortalConfig registry entries must appear before all other data module entries in the registry file.

Registry Entries

Table 4-34 lists the DAT_PortalConfig registry entries.

Table 4-34 DAT_PortalConfig Registry Entries

Entry	Description	Mandatory
InfranetConnection	Specifies the connection to the DBC module.	Yes

Sample Registry Entry

```
PortalConfigDataModule
{
  ModuleName       = DAT_PortalConfig
  Module
  {
    InfranetConnection = ifw.DataPool.LoginInfranet
  }
}
```

Semaphore File Entries

Table 4-35 lists the DAT_PortalConfig Semaphore file entries.

Table 4-35 DAT_PortalConfig Semaphore File Entries

Entry	Description
CBPPrintData	Prints the /config/business_params data stored in the DAT_PortalConfig module's memory. If a filename is not provided, the module dumps the data into a file named **DefaultCBPDataFile**_*Timestamp*.**lst**.
CreditProfilePrintData	Prints the **/config/credit_profile** data stored in the DAT_PortalConfig module's memory. If a filename is not provided, the module dumps the data into a file named **DefaultConfigCreditProfileDataFile**_*Timestamp*.**lst**.
CBPReload	Reloads /config/business_params data.
CreditProfileReload	Reloads **/config/credit_profile** data.
PrintData	Prints all the data stored in the DAT_PortalConfig module's memory. If a filename is not provided, the module dumps the data to the standard output console.
OODReload	Reloads **/config/event_order_criteria** data.

Table 4-35 (Cont.) DAT_PortalConfig Semaphore File Entries

Entry	Description
OODPrintData	Prints the **/config/event_order_criteria** data stored in the DAT_PortalConfig module's memory. If a filename is not provided, the module dumps the data into a file named **DefaultOODDataFile**_*Timestamp*.**lst**.

Sample Semaphore File Entry

```
ifw.DataPool.PortalConfigDataModule.Module.CreditProfile.Reload{}
ifw.DataPool.PortalConfigDataModule.Module.Reload{}
ifw.DataPool.PortalConfig.Module.CBPPrintData=BRM/config/prntCBPdata
ifw.DataPool.PortalConfig.Module.OODPrintData=BRM/config/prntOODdata
```

Events

Table 4-36 lists the DAT_PortalConfig events.

Table 4-36 DAT_PortalConfig Events

Event Name	Trigger	Sender	Parameter
EVT_RELOAD_SUCCESSFUL	Reload was successful.	DAT_PortalConfig	None
EVT_RELOAD_FAILED	Reload failed.	DAT_PortalConfig	None

Database Tables

The DAT_PortalConfig module uses the following database tables:

- CONFIG_T
- CONFIG_EVENT_ORDER_CRITERIA_T

DAT_PrefixDesc

The DAT_PrefixDesc module provides data for mapping phone number prefixes to descriptions, used by the FCT_PrefixDesc module.

See the following documents:

- Creating Call Destination Descriptions
- FCT_PrefixDesc

Dependencies

If data is stored in the database, the DAT_PrefixDesc module requires a connection to the Pipeline Manager database.

Registry Entries

Table 4-37 lists the DAT_PrefixDesc registry entries.

Table 4-37 DAT_PrefixDesc Registry Entries

Entry	Description	Mandatory
CLIBase	Specifies if the zone tree values should be hexadecimal or decimal. You can use this entry in a semaphore file.	Yes
DataConnection	Specifies the connection to the Pipeline Manager database.	Yes, if the data is stored in the database. Otherwise is not used.
PrefixDesc.File	Specifies the file prefix for the files that contain prefix descriptions. See "Creating a Prefix/Description Data File" in *BRM Setting Up Pipeline Rating and Discounting*. You can use this entry in a semaphore file.	Yes, if the data is stored in a file. Otherwise is not used.
ReuseOnFailure	Specifies if the module should continue to use the old data if the **Reload** command fails. If **True**, the old data is used. If the entry is not used, the default is **False**.	Yes
Source	Specifies where the module gets data: • **File** • **Database**	Yes

Sample Registry Entry for the Database Interface

```
PrefixDescDataModule
{
  ModuleName = DAT_PrefixDesc
  Module
  {
    Source = Database
    DataConnection = ifw.DataPool.Login
    ReuseOnFailure = false
    CLIBase = 10
  }
}
```

Sample Registry Entry for the File Interface

```
PrefixDescData
{
  ModuleName = DAT_PrefixDesc
  Module
  {
    Source = File
    ReuseOnFailure = false
    CLIBase = 10
    PrefixDesc
    {
      File = ../daten/forgn_names.dat
      File = ../daten/onkz_names.dat
    }
  }
}
```

Semaphore File Entries

Table 4-38 lists the DAT_PrefixDesc Semaphore file entries.

Table 4-38 DAT_PrefixDesc Semaphore File Entries

Entry	Description
CLIBase	Specifies if the zone tree values should be hexadecimal or decimal. Valid values are 10(DEC) and 16(HEX).
PrefixDesc.File	Specifies the file prefix for the files that contain prefix descriptions.
Reload	Reloads the data.

Sample Semaphore File Entry

```
ifw.DataPool.PrefixDescDataModule.Module.Reload {}
```

Events

Table 4-39 lists the DAT_PrefixDesc events.

Table 4-39 DAT_PrefixDesc Events

Event name	Trigger	Sender	Parameter
EVT_RELOAD_FAILED	Update semaphore	DAT_PrefixDesc	None
EVT_RELOAD_SUCCESSFUL	Update semaphore	DAT_PrefixDesc	None

Database Tables

The DAT_PrefixDesc module uses the IFW_DESTINDESC database table.

See "Creating Call Destination Descriptions" in *BRM Setting Up Pipeline Rating and Discounting*.

DAT_PriceModel

The DAT_PriceModel module provides pricing data for the FCT_MainRating module.

See the following documents:

- About Pipeline Rating
- FCT_MainRating

Dependencies

Requires a connection to the Pipeline Manager database.

This module uses event notification to refresh customized charge offer data. You must configure a connection to "DAT_Listener" if you plan to use this feature.

This module uses the **TailormadeProductsSearch** business parameter to skip lock on pricing. If you do not use tailor-made charge offers, and intend to skip lock on pricing, configure a connection to the DAT_PortalConfig module.

Registry Entries

Table 4-40 lists the DAT_PriceModel registry entries.

Table 4-40 DAT_PriceModel Registry Entries

Entry	Description	Mandatory
DataConnection	Specifies the connection to the Pipeline Manager database.	Yes
Listener	Specifies the connection to the DAT_Listener module.	No
LogEvents	Specifies whether notification events received by the module are written to the process log file. Default = **False**	No
PortalConfigDataModule	Specifies the connection to the DAT_PortalConfig module and enables DAT_PriceModel to retrieve business parameter settings from the DAT_PortalConfig module.	No

Sample Registry Entry

```
PriceModel
{
  ModuleName = DAT_PriceModel
  Module
  {
    DataConnection = ifw.DataPool.Login
    Listener = ifw.DataPool.Listener
  }
}
```

Semaphore File Entries

Table 4-41 lists the DAT_PriceModel Semaphore file entries.

Table 4-41 DAT_PriceModel Semaphore File Entries

Entry	Description
Reload	Reloads the data from the database.
PrintAllPriceModels	Prints all pricings in the configuration.
PrintOnePriceModel <PriceModel ID>	Prints the pricing ID.
PrintRangeOfPriceModels <PriceModel fromID> <PriceModel toID>	Prints all the pricings where ID is in the range.

Sample Semaphore File Entry

```
ifw.DataPool.PriceDataModule.Module.Reload {}
```

Events

Table 4-42 lists the DAT_PriceModel events.

Table 4-42 DAT_PriceModel Events

Event Name	Trigger	Sender	Parameter
EVT_RELOAD_SUCCESSFUL	Reload was successful.	DAT_PriceModel	None
EVT_RELOAD_FAILED	Reload failed.	DAT_PriceModel	None

Database Tables

The DAT_PriceModel module uses the following database tables:

- IFW_PRICEMODEL. This table stores pricing data.
- IFW_PRICEMDL_STEP. This table stores pricing step data.
- IFW_RESOURCE. This table stores resource configuration data.

> **Note:**
>
> For information on compare patterns used in database values, see "About Using Regular Expressions when Specifying the Data to Extract" in *BRM Setting Up Pipeline Rating and Discounting*.

DAT_Rateplan

The DAT_Rateplan module provides charge data for the FCT_MainRating module.

Dependencies

Requires a connection to the Pipeline Manager database.

This module uses event notification to refresh customized charge offer data. You must configure a connection to "DAT_Listener" if you plan to use this feature.

This module uses the **TailormadeProductsSearch** business parameter to skip lock on charges. If you do not use tailor-made charge offers, and intend to skip lock on charges, configure a connection to the DAT_PortalConfig module.

Registry Entries

Table 4-43 lists the DAT_Rateplan registry entries.

Table 4-43 DAT_Rateplan Registry Entries

Entry	Description	Mandatory
DataConnection	Specifies the connection to the Pipeline Manager database.	Yes

Table 4-43 (Cont.) DAT_Rateplan Registry Entries

Entry	Description	Mandatory
Listener	Specifies the connection to the DAT_Listener module.	No
LogEvents	Specifies whether notification events received by the module are written to the process log file. Default = **False**	No
PortalConfigDataModule	Specifies the connection to the DAT_PortalConfig module and enables DAT_RatePlan to retrieve business parameter settings from the DAT_PortalConfig module.	No
RowFetchSize	Specifies the number of rows of data to retrieve from the BRM database. Default = **1000**	**RowFetchSize**

Sample Registry Entry

```
Rateplan
{
  ModuleName = DAT_Rateplan
  Module
  {
    DataConnection = ifw.DataPool.Login
    Listener = ifw.DataPool.Listener
  }
}
```

Semaphore File Entries

Table 4-44 lists the DAT_Rateplan Semaphore file entries.

Table 4-44 DAT_Rateplan Semaphore File Entries

Entry	Description
Reload	Reloads the rating configuration data.
PrintAllRateplans	Prints all charges in the configuration.
PrintOneRateplan <RatePlan ID>	Prints the charge ID.
PrintRangeOfRateplans <RatePlan fromID> <RatePlan toID>	Prints all the charges where ID is in the range.

Sample Semaphore File Entry

```
ifw.DataPool.RateplanDataModule.Module.Reload {}
```

Events

Table 4-45 lists the DAT_Rateplan events.

Table 4-45 DAT_Rateplan Events

Event Name	Trigger	Sender	Parameter
EVT_RELOAD_SUCCESSFUL	Reload was successful.	DAT_Rateplan	None
EVT_RELOAD_FAILED	Reload failed.	DAT_Rateplan	None

Database Tables

The DAT_Rateplan module uses charge data from the following database tables:

- IFW_RATEPLAN
- IFW_RATEPLAN_VER
- IFW_RATEPLAN_CNF

The DAT_Rateplan module uses RUM data from the following database tables:

- IFW_RUM
- IFW_RUMGROUP
- IFW_RUMGROUP_LNK

> **Note:**
>
> For information on compare patterns used in database values, see "About Using Regular Expressions when Specifying the Data to Extract" in *BRM Setting Up Pipeline Rating and Discounting*.

DAT_Recycle

The DAT_Recycle module is used by standard recycling and Suspense Manager EDR to recycle EDRs. It connects to the DAT_Listener module and waits for business events that call for EDRs to be recycled.

This module creates a parameter file that allows the "EXT_InEasyDB" module to read suspended usage records associated with a recycle job. It also provides an interface for the "INP_Recycle" module to provide status updates about the EDR stream.

Dependencies

Requires a connection to the "DAT_Listener" module.

Registry Entries

Table 4-46 lists the DAT_Recycle registry entries.

Table 4-46 DAT_Recycle Registry Entries

Entry	Description	Mandatory
ControlPath	Specifies the path for SQL, parameter, job and restart files. **./database/Oracle/Scripts/Suspense**	Yes
Listener	Specifies the connection to the DAT_Listener module.	Yes
LogEvents	Specifies whether notification events received by the module are written to the process log file. Default = **False**	Yes
ParameterFile	Specifies the name of the parameter file which contains optional key/value entries.	Yes
ProcessCount	Specifies the threshold job count in the **QueueFileName** file. When the threshold is reached, the DAT_Recycle cleans up the queue. If not specified, the default value is **50**.	No
QueueFileName	Specifies the name of the file that stores recycle job IDs to be processed.	Yes
QueueFilePath	The path to the queue file specified for **QueueFileName**.	Yes

Sample Registry Entry

```
#------------------------------------------------------------
# Recycling Data
#------------------------------------------------------------
RecyclingData
{
  ModuleName               = DAT_Recycle
  Module
  {
    Listener = ifw.DataPool.Listener
    LogEvents = True
    ControlPath = ./database/Oracle/Scripts/Suspense
    ParameterFile = parameter.isc
    QueueFileName = RecycleJobIds_wireless.dat
    QueueFilePath = ./data
    ProcessCount = 50
  }
}
```

Semaphore File Entries

DAT_Recycle does not support semaphore updates.

DAT_ResubmitBatch

The DAT_ResubmitBatch module supports batch suspension and resubmission.

DAT_ResubmitBatch subscribes to "DAT_Listener" for ResubmitBatchRequest event. Upon receiving this, it gets information for all the batches corresponding to this ResubmitBatchRequest from the BRM database. It then moves all these batches to their respective pipeline input directories.

A ResubmitBatchRequest is propagated to the ifw through ifw_sync when a user resubmits a suspended batch with the SMC. During resubmission, a notification event (**/event/notification/**

suspense/batch_resubmit) is generated with the admin action job id. This notification event is propagated to the ifw through ifw_sync in form of ResubmitBatchRequest.

Dependencies

This module requires connections to the following:

- BRM database.
- Pipeline Manager database.
- DAT_Listener module. See "DAT_Listener".

Registry Entries

Table 4-47 lists the DAT_ResubmitBatch registry entries.

Table 4-47 DAT_ResubmitBatch Registry Entries

Entry	Description	Mandatory
DataConnection	Specifies the connection to the Pipeline Manager database.	Yes
Listener	Specifies the connection to the DAT_Listener module.	Yes
LogEvents	Logs each request received from the listener when true. Default = **True**	No
QueueFileName	Name of file for file based queue. For example: **QueueFileName** = **ResubmitJobIds.dat**	No
QueueFilePath	Path of file for file based queue	No
QueueFileCleanupThreshold	For already processed events cleanup.	No
PipelineCategory	The Pipeline Category for the records. The module should only process records of its own pipeline category For example: **PipelineCategory= CDRPipeline**	Yes
TempDirectoryPath	Temporary directory path, used as a staging directory for resubmitted batches. It should not be used for any other purpose. For example: TempDirectoryPath = **./data/tmp**	No

Sample Registry Entry

```
ResubmitBatch
  {
    ModuleName = DAT_AccountBatch
    Module
    {
      DataConnection       = ifw.DataPool.LoginInfranet
      Listener             = ifw.DataPool.Listener
      LogEvents            = True
      QueueFileName        = ResubmitJobIds.dat
      QueueFilePath        = ./dataQueue
      FileCleanupThreshold = 50
      PipelineCategory     = CDRPipeline
      TmpDirectoryPath     = ./data/tmp
```

```
    }
  }
```

Semaphore File Entries

DAT_ResubmitBatch does not support semaphore updates.

DAT_ScenarioReader

The DAT_ScenarioReader module provides aggregation scenario data for the FCT_AggreGate module.

See the following documents:

- Setting Up Pipeline Aggregation
- FCT_AggreGate

Dependencies

Requires a connection to the Pipeline Manager database.

Registry Entries

Table 4-48 lists the DAT_ScenarioReader registry entries.

Table 4-48 DAT_ScenarioReader Registry Entries

Entry	Description	Mandatory
Calendar	Specifies the calendar that is used for special day evaluation. Default = No calendar	No
DataCollection	Specifies a connection to the Pipeline Manager database.	Yes

Sample Registry Entry

```
ScenarioReader
{
  ModuleName = DAT_ScenarioReader
  Module
  {
    DataConnection = ifw.DataPool.Database
    Calendar = 2
  }
}
```

Semaphore File Entries

Table 4-49 lists the DAT_ScenarioReader Semaphore file entries.

Table 4-49 DAT_ScenarioReader Semaphore File Entries

Entry	Description
Reload	Reloads aggregation scenarios.

Sample Semaphore File Entry

```
ifw.DataPool.ScenarioReader.Module.Reload {}
```

Messages and Requests

Table 4-50 lists the DAT_ScenarioReader messages and requests.

Table 4-50 DAT_ScenarioReader Messages and Requests

Message/Request	Description	Sending/Receiving
REQ_EVENTHANDLER_NAME	Get the Event Handler.	Send to controller.

Events

Table 4-51 lists the DAT_ScenarioReader events.

Table 4-51 DAT_ScenarioReader Events

Event Name	Trigger	Sender	Parameter
EVT_RELOAD_SUCCESSFUL	Data reload was successful.	DAT_ScenarioReader	None
EVT_RELOAD_FAILED	Data reload failed.	DAT_ScenarioReader	None

Database Tables

The DAT_ScenarioReader module uses the following database tables:

- IFW_SCENARIO. This table stores the aggregation scenario parameters. Some values can be overwritten by using the FCT_AggreGate registry.
- IFW_EDRC_FIELD. This table defines the EDR container fields for the aggregation scenarios. Each scenario uses exactly one EDR container description.
- IFW_CONDITION. This table stores the conditions that exclude an EDR or parts of an EDR from the aggregation process.
- IFW_GROUPING. This tables stores the scenario groupings that group aggregated results into subgroups. You can summarize the values within a grouping into subclasses.
- IFW_AGGREGATION. This tables stores aggregation functions and specifies how to handle the results.
- IFW_GROUPING_CNF. This table links data classes to a grouping.
- IFW_CLASS. This table defines the grouping classes. Each class consists of several class items.

- IFW_CLASSITEM. This table defines class items. All grouping values matching a class item are summarized and the class item code is added to the result.
- IFW_CLASS_LNK. This table links items and classes.
- IFW_CLASSCON. This table defines the class conditions. They determine which class to use when more than one class is associated to a grouping. A class condition specifies the dependency between a class from one grouping and a class item from another grouping.
- IFW_CLASSCON_LNK. This table links class conditions to one or more class items.

> **Note:**
>
> For information on compare patterns used in database values, see "About Using Regular Expressions when Specifying the Data to Extract" in *BRM Setting Up Pipeline Rating and Discounting*.

DAT_TimeModel

The DAT_TimeModel module provides time model, time zone, and day code data for the FCT_Mainrating module.

Dependencies

Requires a connection to the Pipeline Manager database.

Registry Entries

Table 4-52 lists the DAT_TimeModel registry entries.

Table 4-52 DAT_TimeModel Registry Entries

Entry	Description	Mandatory
DataConnection	Specifies a connection to the Pipeline Manager database.	Yes

Sample Registry Entry

```
TimeModel
{
  ModuleName = DAT_TimeModel
  Module
  {
    DataConnection = ifw.DataPool.Login
  }
}
```

Semaphore File Entries

Table 4-53 lists the DAT_TimeModel Semaphore file entries.

Table 4-53 DAT_TimeModel Semaphore File Entries

Entry	Description
Reload	Reloads data from the Pipeline Manager database.

Sample Semaphore File Entry

```
ifw.DataPool.TimeDataModule.Module.Reload {}
```

Events

Table 4-54 lists the DAT_TimeModel events.

Table 4-54 DAT_TimeModel Events

Event Name	Trigger	Sender	Parameter
EVT_RELOAD_SUCCESSFUL	Data reload was successful.	DAT_TimeModel	None
EVT_RELOAD_FAILED	Data reload failed.	DAT_TimeModel	None

Database Tables

The DAT_TimeModel module uses the following database tables:

- IFW_TIMEMODEL
- IFW_TIMEMODEL_LNK
- IFW_DAYCODE
- IFW_TIMEINTERVAL
- IFW_TIMEZONE

> **Note:**
>
> For information on compare patterns used in database values, see "About Using Regular Expressions when Specifying the Data to Extract" in *BRM Setting Up Pipeline Rating and Discounting*.

DAT_USC_Map

The DAT_USC_Map module provides usage scenario mapping data.

The DAT_USC_Map module retrieves mapping data from an ASCII file or from the Pipeline Manager database. This data is used by the FCT_USC_Map module to perform usage scenario mapping. See "FCT_USC_Map".

Dependencies

If the usage scenario mapping is stored in the database, this module requires a connection to the Pipeline Manager database.

Registry Entries

Table 4-55 lists the DAT_USC_Map registry entries.

Table 4-55 DAT_USC_Map Registry Entries

Entry	Description	Mandatory
DataConnection	Specifies a connection to the Pipeline Manager database.	Yes, if the data is stored in the database. Otherwise not used.
LoadZoneDescription	Specifies whether to load the zone description into memory. Default = **False**	No
LoadZoneEntryName	Specifies whether to load the zone name into memory. Default = **False**	Yes
OptimizeFor	Specifies whether mapping should be optimized for **Speed** (the default) or **Memory**.	No
Source	Specifies where the USC mapping data is stored. The possible values are a **File** or **Database**.	Yes
USCMapFile	If **Source = File**, specifies file name and path that contains the USC mapping data. You can use this entry in a semaphore file.	Yes, if the data is stored in a file. Otherwise not used.
PreCompiledDataDir	Compiles USC mapping data and saves the data to the specified directory. Default = ./**compiled_usc_data** Files are stored in the format *USCzoneModelName*.**pc**. Make sure that the directory exists under the specified path. The compiled files are created in the first run of the pipeline. Before each subsequent run, they are validated and recompiled if necessary.	No
NumberOfThreads	Specifies the number of threads to use when loading and saving the precompiled mapping data. Default = **1**	No
UscGroups	Specifies the USC groups for which to load rules. Enclose the values in curly braces. For example: `UscGroups {TEL TEL_ROAMING TEL_INTL}` The default is to load all USC groups in the system. Use the semaphore when mapping rules are stored in the database (Source = **Database**).	No

Sample Registry Entry

```
USCDataModule
{
  ModuleName = DAT_USC_Map
```

```
  Module
  {
     Source               = DataBase
     DataConnection       = ifw.DataPool.Login
     LoadZoneDescription  = False
     LoadZoneEntryName    = False
     OptimizeFor          = MEMORY
     PreCompiledDataDir   = ./compiled_usc_data
     NumberOfThreads      = 5
     UscGroups            = {TEL TEL_ROAMING}
   }
}
```

Semaphore File Entries

Table 4-56 lists the DAT_USC_Map Semaphore file entries.

Table 4-56 DAT_USC_Map Semaphore File Entries

Entry	Description
LoadZoneDescription	Specifies whether to load the zone description into memory. Default = **False**
LoadZoneEntryName	Specifies whether to load the zone name into memory. Valid values are **True** and **False**.
PrintAllUscMapData	Prints all the USC map data.
PrintUscMapDataForZoneModel	Prints the data for a given zone model ID.
PreCompiledDataDir	Compiles USC mapping data and saves the data to the specified directory. Default = **./compiled_usc_data** Files are stored in the format *USCzoneModelName*.**pc**. Make sure that the directory exists under the specified path. The compiled files are created in the first run of the pipeline. Before each subsequent run, they are validated and recompiled if necessary.
NumberOfThreads	Specifies the number of threads to use when loading and saving the precompiled mapping data. Default = **1**
UscGroups	Specifies the USC groups for which to load rules. If not set, all USC groups are loaded.
Reload	Command used to reload data into memory from the database.
USCMapFile	If **Source = File**, specifies file name and path that contains the USC mapping data.

Sample Semaphore File Entry

```
ifw.DataPool.USCDataModule.Module.UscGroups {TEL TEL_ROAMING}
ifw.DataPool.USCDataModule.Module.Reload {}
```

Database Tables

If the mapping data is stored in the Pipeline Manager database, The DAT_USC_Map module uses the IFW_USC_MAP database table. This table stores mapping rules for usage scenario maps.

For information on compare patterns used in database values, see "About Using Regular Expressions when Specifying the Data to Extract" in *BRM Setting Up Pipeline Rating and Discounting*.

DAT_Zone

The DAT_Zone module provides zone data for the FCT_MainRating and the FCT_Zone modules. This module stores the real-time service class to Pipeline Manager service code mapping information in memory. When it processes realtime data, it returns the service code for a given service class.

Dependencies

Requires a connection to the Pipeline Manager database.

Registry Entries

Table 4-57 lists the DAT_Zone registry entries.

Table 4-57 DAT_Zone Registry Entries

Entry	Description	Mandatory
CliMode	Specifies if the zoning tree should be created in decimal (DEC) or hexadecimal (HEX) mode. Default = **DEC**	No
DataConnection	Specifies a connection to the Pipeline Manager database.	Yes, if the module gets data from the database.
DistCalcMode	Specifies the mode for calculating the distance between two area codes: • **DISC**. The configured coordinates are Cartesian coordinates. The distance is calculated using the Pythagorean theorem. • **GLOBE**. The coordinates are global. The distance is calculated using slower goniometric functions. You can use this entry in a semaphore file. Default = **DISC**	No
FileName	Specifies the path and file name for the zone model configuration file, if the module gets data from a file. You can use this entry in a semaphore file.	Yes, if the module gets data from a file
GeoFileName	Specifies the path and file name for the area code coordinate link file, if the module gets data from a file. You can use this entry in a semaphore file.	Yes, if the module gets data from a file
LoadZoneDescription	Specifies whether to load the zone descriptions into memory. Default = **False** You can use this entry in a semaphore file.	No
LoadZoneEntryName	Specifies whether to load the zone names into memory. Default = **False** You can use this entry in a semaphore file.	No

Table 4-57 (Cont.) DAT_Zone Registry Entries

Entry	Description	Mandatory
MaxAge	Specifies the maximum age of zone entries. If the value is **0** or null, all zone entries are loaded. You can use this entry in a semaphore file. Default = **0**	No
RealTime	Specifies whether the module should process real-time events. If **True**, the module processes real-time events, if **False**, the module processes batch events.	Yes
ReuseOnFailure	Specifies if the module should continue to use the old data if the **Reload** command fails: **True** = use the old data. **False** = do not use the old data. Default = **False**	Yes
Source	Specifies whether the module gets data from a file or the database.	Yes
ZoneModels	Specifies the source of the zone model codes: • If the source is a file, contains a list of zone model codes with the corresponding path and file name for the configuration file. • If the source is the database, contains a list of zone model codes that shall be used. You can use this entry in a semaphore file.	No, if the module gets data from the database.

Sample Registry for the Database Interface

```
Module
{
  ReuseOnFailure = FALSE
  MaxAge = 0
  Source = Database
  DataConnection = ifw.DataPool.Login
  LoadZoneDescription = False
  LoadZoneDescription = False
  ZoneModels
  {
    BASIC
    PROFI
    SPECIAL
  }
}
```

Sample Registry for the File Interface

Standard zone:

```
Module
{
   ReuseOnFailure = FALSE
   MaxAge = 90
   Source = File
   FileName = ./cfg/ZoneModelConfig.dat
   ZoneModels
   {
```

```
      ZM_ADD = /data9/INTEGRATE/TEST/config/ZM_ADD.dat
   }
}
```

Geographical zone:

```
Module
{
   ReuseOnFailure = FALSE
   MaxAge = 90
   Source = File
   FileName = ./cfg/ZoneModelConfig.dat
   GeoFileName = ./cfg/GeoAreaLink.dat
   ZoneModels
   {
      ZM_GEO = /data9/INTEGRATE/TEST/config/ZM_GEO.dat
   }
}
```

Sample Registry for Real-Time Zoning

```
Module
{
  ReuseOnFailure = FALSE
  Source = DataBase
  MaxAge = 0
  DistCalcMode = DISC
  DataConnection = ifw.DataPool.Login
  LoadZoneDescription = False
  LoadZoneEntryName = False
  RealTime = True
  ZoneModels
  {

  }
}
```

Semaphore File Entries

Table 4-58 lists the DAT_Zone Semaphore file entries.

Table 4-58 DAT_Zone Semaphore File Entries

Entry	Description
DistCalcMode	Specifies the mode for calculating the distance between two area codes: • **DISC**. The configured coordinates are cartesian coordinates. The distance is calculated using the Pythagorean theorem. • **GLOBE**. The coordinates are global. The distance is calculated using slower goniometric functions.
FileName	Specifies the path and file name for the zone model master file, if the module gets data from a file.
GeoFileName	Specifies the path and file name for the area code coordinate link file, if the module gets data from a file.
LoadZoneDescription	Specifies whether to load the zone descriptions into memory. **Note:** When this entry is updated through a semaphore, the reload semaphore must also be passed to reload the zone descriptions.

Table 4-58 (Cont.) DAT_Zone Semaphore File Entries

Entry	Description
LoadZoneEntryName	Specifies whether to load the zone names into memory. **Note:** When this entry is updated through a semaphore, the reload semaphore must also be passed to reload the zone names.
MaxAge	Specifies the maximum age of zone entries. If the value is 0 or null, all zone entries are loaded.
Reload	Reloads the zoning data.
ZoneModels	Specifies the source of the zone model codes: • If the source is a file, contains a list of zone model codes with the corresponding path and file name for the configuration file. • If the source is the database, contains a list of zone model codes that shall be used.

Sample Semaphore File Entry for the Database Interface

```
ifw.DataPool.ZoneDataModule.Module.ZoneModels.BASIC.Reload {}
ifw.DataPool.ZoneDataModule.Module.ZoneModels.SPECIAL.Reload {}
```

Sample Semaphore File Entry for the File Interface

```
ifw.DataPool.ZoneDataModule.Module.ZoneModels.
ZM_ADD = /data9/INTEGRATE/test/config/ZM_ADD-new.dat

ifw.DataPool.ZoneDataModule.Module.ZoneModels.
ZM_MOBILE = /data9/INTEGRATE/test/config/ZM_MOBILE-new.dat
```

Events

Table 4-59 lists the DAT_Zone events.

Table 4-59 DAT_Zone Events

Event Name	Trigger	Sender	Parameter
EVT_RELOAD_SUCCESSFUL	Data reload was successful.	DAT_Zone	None
EVT_RELOAD_FAILED	Data reload failed.	DAT_Zone	None

Database Tables

The DAT_Zone module uses data from the following database tables:

- IFW_ZONEMODEL
- IFW_IMPACT_CAT
- IFE_STANDARD_ZONE
- IFW_GEO_MODEL
- IFW_GEO_ZONE
- IFW_GEOAREA_LNK

> **Note:**
>
> For information on compare patterns used in database values, see "About Using Regular Expressions when Specifying the Data to Extract" in *BRM Setting Up Pipeline Rating and Discounting*.

5

Pipeline Manager iRules

Learn about the Oracle Communications Billing and Revenue Management (BRM) Pipeline Manager iRules.

Topics in this document:

IRL_EventTypeSplitting

The IRL_EventTypeSplitting iRule sends EDRs to separate output streams based on service codes.

See "Sending EDRs to Pipeline Output Streams" in *BRM Setting Up Pipeline Rating and Discounting*.

Dependencies

This module must run after the FCT_ServiceCodeMap module and before the FCT_Reject module.

This is typically the last module before the FCT_Reject module.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Sample Registry

```
EventSplitting
{
 ModuleName = FCT_IRules
 Module
 {
    Active = True
    Source = Database
    DataConnection = ifw.DataPool.DataConnection
    Rules {}
 }
}
```

EDR Container Fields

Table 5-1 lists the IRL_EventTypeSplitting EDR Container fields.

Table 5-1 IRL_EventTypeSplitting EDR Container Fields

Alias field name Default field name	Type	Access	Description
INTERN_SERVICE_CODE DETAIL.INTERN_SERVICE_CODE	String	Read	Internal service code.

IRL_EventTypeSplitting_tt

The IRL_EventTypeSplitting_tt iRule sends EDRs to separate output streams.

See "Sending EDRs to Pipeline Output Streams" in *BRM Setting Up Pipeline Rating and Discounting*.

Dependencies

This module must run after the FCT_ServiceCodeMap module and before the FCT_Reject module.

This is typically the last module before the FCT_Reject module.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Sample Registry

```
ServiceOutputSplit
{
 ModuleName = FCT_IRules
 Module
 {
    Active = True
    Source = Files
    Rules
    {
    }
    Descriptions
    {
        ServiceCodeSplit = ./iScriptLib/iScriptLib_Standard/IRL_EventTypeSplitting_tt.irl
    }
 }
}
```

EDR Container Fields

Table 5-2 lists the IRL_EventTypeSplitting_tt EDR Container fields.

Table 5-2 IRL_EventTypeSplitting_tt EDR Container Fields

Alias field name Default field name	Type	Access	Description
LOGICAL_PARTITION_ID DETAIL.LOGICAL_PARTITION_ID	String	Read	Logical Partition ID.
INTERN_SERVICE_CODE DETAIL.INTERN_SERVICE_CODE	String	Read	Internal service code.

IRL_LeastCostPerEDR

The IRL_LeastCostPerEDR iRule flags all EDRs that satisfy the criteria for BRM least=1 cost rating. For more information, see "About Least Cost Rating" in *BRM Setting Up Pipeline Rating and Discounting*.

You set up the criteria that an EDR must meet to qualify for least=1 cost rating in the **IRL_LeastCostPerEDR.irl** and **IRL_LeastCostPerEDR.data** files. See "Specifying the Rules to Qualify for Least Cost Rating" in *BRM Setting Up Pipeline Rating and Discounting*.

Dependencies

This module must run:

- Before the "FCT_CustomerRating" module, because FCT_CustomerRating uses the flag set by this module to decide whether to create charge packets for all charge offers.
- After the "FCT_Filter_Set" module, because the rules you set up in **IRL_LeastCostRating.data** frequently use filter sets as one criteria for least=1 cost rating.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 5-3 lists the IRL_LeastCostEDR registry entries.

Table 5-3 IRL_LeastCostEDR Registry Entries

Entry	Description	Mandatory
Active	Specifies if the module is active or inactive. • **True** = Active • **False** = Inactive	Yes
LeastCostCheck	Specifies the path to the **IRL_LeastCostPerEDR.irl** file. See "Specifying the Rules to Qualify for Least Cost Rating" in *BRM Setting Up Pipeline Rating and Discounting*.	Yes
Source	Specifies whether the least= cost rating data is stored in a file or in a database table. • **File** = The iRules data is stored in a file. • **Database** = The iRules data is stored in a database table.	Yes

Sample Registry

```
LeastCostPerEDR
{
   ModuleName = FCT_IRules
   Module
   {
      Active = False
      Source = Database
      DataConnection = ifw.DataPool.DataConnection
              Rules {}
   }
   }
```

EDR Container Fields

Table 5-4 lists the IRL_LeastCostEDR EDR Container fields.

Table 5-4 IRL_LeastCostEDR EDR Container Fields

Alias field name Default field name	Type	Access	Description
DETAIL.CUST_A.LEAST_COST	Integer	Write	Toggles least= cost rating on and off. A value of **1** means do not apply least cost rating; a value of **2** specifies to apply least= cost rating.
DETAIL.CUST_A.MARKET_SEGMENT	String	Read	Specifies the filter set associated with the EDR.
DETAIL.CUST_A.PRODUCT_PRIORITY	String	Read	Contains a list of the priorities for all charge offers that are associated with the same service and event.
DETAIL.CUST_A.INTERN_FOUND_PP_INDEX	String	Write	Contains the index of the highest priority rating charge offer. This is the charge offer with the highest rate priority for an event. In the case of two charge offers with matching priorities, the charge offer with the first start time is selected.

IRL_PipelineSplitting

The IRL_PipelineSplitting iRule is used in the pre-recycling pipeline to send EDRs to different output streams depending on their original pipeline names. The EDRs are then routed to their original pipelines for recycling.

The **PipelineSplitting.irl** file specified in the registry references a data file called **PipelineSplitting.data**, which you must modify based on your pipeline names. The default contents of the file are:

```
ALL_RATE;PreRecycleOutput
ALL_RATE_2;PreRecycleOutput_2
.*;PreRecylceOutput
```

See "Configuring a Pre-Recycling Pipeline" in *BRM Suspending and Recycling Pipeline EDRs*.

Sample Registry

```
PipelineSplit
{
  ModuleName = FCT_IRules
  Module
  {
    Active = True
      Source = Database
      DataConnection = ifw.DataPool.DataConnection
      Rules
      {
      }
  }
}
```

IRL_PromotionalSavingPerEDR

The IRL_PromotionalSavingPerEDR iRule flags all EDRs that satisfy the criteria for a promotional savings calculation.

You set up the rules to qualify for the promotional savings calculation in the promotional savings iRules files (**IRL_PormotionalSavingPerEDR.irl** and **IRL_PromotionalSavingPerEDR.data**).

For more information, see "About Calculating the Promotional Savings" in *BRM Setting Up Pipeline Rating and Discounting*.

Dependencies

This module must run before the IRL_LeastCost module and the FCT_CustomerRating module.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 5-5 lists the IRL_PromotionalSavingPerEDR registry entries.

Table 5-5 IRL_PromotionalSavingPerEDR Registry Entries

Entry	Description	Mandatory
Active	Specifies if the module is active or inactive. • **True** = Active • **False** = Inactive	Yes
PromotionalSaving	Specifies the path to the **IRL_PromotionalSavingPerEDR.irl** file. See "Specifying the Rules to Qualify for Promotional Savings" in *BRM Setting Up Pipeline Rating and Discounting*.	No
Source	Specifies whether the iRules data is stored in a file or database table. • **File** = The iRules data is stored in a file. • **Database** = The iRules data is stored in a database table.	Yes

Sample Registry

```
PromotionalSavingPerEDR
{
   ModuleName = FCT_IRules
   Module
   {
      Active = True
      Source = Database
      DataConnection = ifw.DataPool.DataConnection
          Rules {}
   }
}
```

EDR Container Fields

Table 5-6 lists the IRL_PromotionalSavingPerEDR EDR Container fields.

Table 5-6 IRL_PromotionalSavingPerEDR EDR Container Fields.

Alias field name Default field name	Type	Access	Description
DETAIL.CUST_A.PROMOTIONAL_SAVING	Integer	Write	Toggles promotional savings on and off. A value of **1** means do not apply promotional savings. A value of **2** applies promotional savings.
DETAIL.CUST_A.MARKET_SEGMENT	String	Read	Specifies the filter set associated with an EDR.
DETAIL.CUST_A.PRODUCT_PRIORITY	String	Read	Contains a list of the priorities for all charge offers that are associated with the same service and event.
DETAIL.CUST_A.INTER_RATING_PRODUCTS	String	Write	Contains the charge offer rating indexes. This is a comma-separated list of all rating charge offers' indexes associated with the same service and event, and their priorities.

IRL_UsageType

The IRL_UsageType iRule assigns usage types to EDRs.

Dependencies

This module must be run after the FCT_Account module and before the FCT_USC_Map module.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Sample Registry

```
IRules
{
  ModuleName = FCT_IRules
  Module
  {
     Active = True
     Source = Database
     DataConnection = integrate.DataPool.DataConnection
     Rules {}
  }
}
```

EDR Container Fields

Table 5-7 lists the IRL_UsageType EDR Container fields.

Table 5-7 IRL_UsageType EDR Container Fields

Alias field name Default field name	Type	Access	Description
DETAIL.CUST_A.ERA.PROFILE	String	Read	Contains the profile for customer A
DETAIL.CUST_B.ERA.PROFILE	String	Read	Contains the profile for customer B
DETAIL.CUST_A.PRODUCT.ERA.PROFILE	String	Read	Contains the charge offer profile for customer A
DETAIL.CUST_B.PRODUCT.ERA.PROFILE	String	Read	Contains the charge offer profile for customer B
DETAIL.CUST_A.ERA.PA.KEY	String	Read	Contains the customer A profile attribute key
DETAIL.CUST_A.ERA.PA.VALUE	String	Read	Contains the customer A profile attribute value
DETAIL.CUST_B.ERA.PA.KEY	String	Read	Contains the customer B profile attribute key
DETAIL.CUST_B.ERA.PA.VALUE	String	Read	Contains the customer B profile attribute value
DETAIL.CUST_A.PRODUCT.ERA.PA.KEY	String	Read	Contains the customer A charge offer profile attribute key
DETAIL.CUST_A.PRODUCT.ERA.PA.VALUE	String	Read	Contains the customer A charge offer profile attribute key
DETAIL.CUST_B.PRODUCT.ERA.PA.KEY	String	Read	Contains the customer B charge offer profile attribute key
DETAIL.CUST_B.PRODUCT.ERA.PA.VALUE	String	Read	Contains the customer B charge offer profile attribute key
DETAIL.CUST_A.INTERN_FOUND_PP_INDEX	String	Read	Contains the internal found purchases charge offer index.
DETAIL.USAGE_DIRECTION	String	Read	Contains the usage direction.
DETAIL.CONNECT_SUB_TYPE	String	Read	Contains the connection subtype
DETAIL.CUST_A.ACCOUNT_NO	String	Read	Contains the customer A account number.

Table 5-7 (Cont.) IRL_UsageType EDR Container Fields

Alias field name Default field name	Type	Access	Description
DETAIL.CUST_B.ACCOUNT_NO	String	Read	Contains the customer B account number
DETAIL.CUST_A.SYSTEM_BRAND	String	Read	Contains the system brand for customer A
DETAIL.CUST_B.SYSTEM_BRAND	String	Read	Contains the system brand for customer B
DETAIL.CUST_A.BILL_CYCLE	String	Read	Contains the customer A bill cycle
DETAIL.B_NUMBER	String	Read	Contains the B number for call
DETAIL.ASS_GSMW_EXT.CELL_ID	String	Read	Contains the cell ID for GSM call
DETAIL.CHARGING_START_TIMESTAMP	String	Read	Contains the start time for call
DETAIL.CUST_A.PRODUCT.RATEPLAN_NAME	String	Read	Contains the charge for customer A charge offer
DETAIL.CUST_B.PRODUCT.RATEPLAN_NAME	String	Read	Contains the charge for customer B charge offer

iRuleValidation

iRuleValidation is an instance of the FCT_IRules module used to validate the data in individual CIBER fields in the EDR container. iRuleValidation uses the **CIBER_VAL.xml** file that specifies the rules and rule items for validating CIBER fields.

You must load the rules in the **CIBER_VAL.xml** file into the Pipeline Manager database before starting Pipeline Manager.

Dependencies

Run this iRule before the ISC_TapSplitting module.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Sample Registry

```
iRuleValidation
{
  ModuleName = FCT_IRules
  Module
  {
     Active = True
     Source = Database
     DataConnection = ifw.DataPool.DataConnection
     Rules
     {
        CIBER_VAL
     }
  }
}
```

6

Pipeline Manager iScripts

Learn about the Oracle Communications Billing and Revenue Management (BRM) Pipeline Manager iScripts.

Topics in this document:

- ISC_SetAndValidateBatchInfo
- ISC_SetEDRStatus
- ISC_SetOutputStream
- ISC_SetRevenueFigures
- ISC_SetRevenueStream
- ISC_SetSvcCodeRTZoning
- ISC_StartTime
- ISC_TapDetailValidation_v3_12
- ISC_TapHeaderTrailerValidation_v3_12
- ISC_TapSplitting
- ISC_TaxCalc
- ISC_TAP_0312_Include
- ISC_TAP_0312_InMap
- ISC_TAP_0312_Validations
- ISC_UsageClassSetting
- UpdateTapInfo_StopRapout
- UpdateTapInfo_Tapin

ISC_AddCBD

The ISC_AddCBD iScript prepares EDRs for rerating in the backout pipeline.

> **Note:**
>
> The ISC_AddCBD iScript is a deprecated module but remains in BRM for backward compatibility.

Dependencies

This module runs in its own backout pipeline for rerating.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Sample Registry

```
AddCBD
{
  ModuleName = FCT_IScript
  Module
  {
    Active = TRUE
    Source = FILE
    Scripts
    {
```

```
      AddCBD
      {
        FileName = ./iScriptLib/iScriptLib_Standard/ISC_AddCBD.isc
      }
    }

}
```

Modified Output Container Fields

The ISC_AddCBD iScript creates one associated charge breakdown record of type 981 with charge packets of type 680.

EDR Container Fields

Table 6-1 lists the ISC_AddCBD EDR container fields.

Table 6-1 ISC_AddCBD EDR Container Fields

Alias field name Default field name	Type	Access	Description
DETAIL.ASS_PIN.ACCOUNT_POID	String	Read	Contains the BRM account POID.
DETAIL.ASS_PIN.BP.PIN_INFO_STRING	String	Read	Contains the string that stores aggregated balance packet information.
DETAIL.ASS_PIN.BP.PIN_AMOUNT	Decimal	Read/Write	Contains the monetary balance impact.
DETAIL.ASS_PIN.BP.PIN_DISCOUNT	Decimal	Read/Write	Contains the discount value
DETAIL.ASS_CBD.ACCOUNT_CODE	String	Write	Contains the account code related to charge packet being constructed.
DETAIL.ASS_CBD.RECORD_TYPE	String	Write	Contains the record type of charge packet.
DETAIL.ASS_CBD.CP	Data Block	Write	Charge packet data block.
DETAIL.DISCOUNT_KEY	String	Write	Contains the discount key.
DETAIL.ASS_CBD.CP.DP	Data Block	Write	Charge breakdown discount packet data block.

Required Input EDR Container Fields

The associated BRM billing record type 900 and the balance impacts of type 600 must be present.

ISC_ApplyTax

The ISC_ApplyTax iScript is used in the reprice pipeline during Incollect processing. This iScript retrieves the taxation flag value for the specific network operator from the in-memory cache. If the taxation flag is set to **on**, the tax amount is passed to the subscriber; otherwise, the tax amount is ignored.

Dependencies

This iScript requires the DAT_InterConnect module and the iScript extension IXT_OpFlag, which provide network operator configuration information that is accessed by ISC_ApplyTax.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Sample Registry

```
ApplyTaxIScript
{
   ModuleName = FCT_IScript
   Module
   {
      Active = True
      Source = File
      Scripts
      {
         ApplyTaxIScript
         {
            FileName = ./iScriptLib/iScriptLib_Roaming/ISC_ApplyTax.isc
         }
      }
   }
}
```

EDR Container Fields

Table 6-2 lists the ISC_ApplyTax EDR container fields.

Table 6-2 ISC_ApplyTax EDR Container Fields

Alias field name Default field name	Type	Access	Description
DETAIL.ASS_CBD.CP.CHARGED_AMOUNT_VALUE	Decimal	Read/Write	Contains the charge amount value.
DETAIL.ASS_ROAMING_EXT.SENDER	String	Read	Contains the sender PLMN of the Transferred Account Procedure (TAP) file.

ISC_BACKOUTTypeSplitting

The ISC_BACKOUTTypeSplitting iScript is used by the backout pipeline for backout-only rerating. It determines if the EDRs are flagged for backout-only rerating and sends the EDRs to different output streams based on the event types.

Sample Registry

```
BackOutputSplit
{
     ModuleName = FCT_IScript
     Module
     {
          Active = TRUE
          Source = File
          Scripts
          {
               SplitScript
               {
                    FileName = ./iScriptLib/iScriptLib_Standard/ISC_BACKOUTTypeSplitting.isc
               }
```

```
          }
     }
}
```

ISC_CiberInputValidation

The ISC_CiberInputValidation iScript performs record-level validations of CIBER records.

Dependencies

Because erroneous CIBER records can be discarded, this module must run before the FCT_Discard module.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Sample Registry

```
ISC_CiberInputValidation
{
  ModuleName = FCT_IScript
  Module
  {
     Active = TRUE
     Source = FILE
     Scripts
     {
       CiberInputValidation
       {
            FileName = ./iScriptLib/iScriptLib_Standard/ISC_CiberInputValidation.isc
       }
    }
  }
}
```

ISC_CiberOutputMapping

The ISC_CiberOutputMapping iScript adds charge data to the ASSOCIATED_CIBER_EXTENSION block of the EDR. If the EDR does not contain an ASSOCIATED_CIBER_EXTENSION block, this iScript adds one.

Dependencies

This module must run after the FCT_MainRating module and the ISC_PostRating iScript.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Sample Registry

```
ISC_CiberOutputMapping
{
  ModuleName = FCT_IScript
  Module
  {
    Active = TRUE
```

```
    Source = FILE
    Scripts
    {
       CiberOutputMapping
       {
          AirRum = AIR
          TollRum = TOL
          OCCRum = OCC
          FileName = ./iScriptLib/iScriptLib_Standard/ISC_CiberOutputMapping.isc
       }
    }
  }
}
```

EDR Container Fields

Table 6-3 lists the ISC_CiberOutputMapping EDR container fields.

Table 6-3 ISC_CiberOutputMapping EDR Container Fields

Alias field name Default field name	Type	Access	Description
DETAIL.ASS_CBD.CP.CHARGED_AMOUNT_CURRENCY	String	Read	Specifies the currency type.
CURRENCY_TYPE	String	Write	Specifies the currency type.
DETAIL.ASS_CBD.CP.RUM	String	Read	Checked to see if the EDR contains a toll charge.
TOLL_NETWORK_CARRIER_ID	String	Write	If the EDR contains a toll charge, this is set to **99001**, otherwise it is set to **00000**.
AIR_CHARGE Set only for CIBER record type 22 (specified in DETAIL.ASS_CIBER_EXT.CIBER_RECORD_TYPE) .	String	Write	If the value of AIR_CHARGABLE_TIME in the CIBER extension block is **0**, AIR_CHARGE is also set to **0**. Otherwise, it is set to the sum of the DETAIL.ASS_CBD.CP.CHARGED_AMOUNT_VALUE in each charge packet for which all of the following statements are true: • RUM equals the value specified in the **AirRum** parameter in the registry • CHARGED_CURRENCY_TYPE equals "R" (rated) or "H" (home) • PRICEMODEL_TYPE equals "S" or "A" **Note:** If the value of DETAIL.ASS_CIBER_EXT.SPECIAL_FEATURES_USED in the CIBER extension block is F, then AIR_CHARGE is always set to **0.0**. The Special Features Used value is set in the CIBER input grammar.

Table 6-3 (Cont.) ISC_CiberOutputMapping EDR Container Fields

Alias field name Default field name	Type	Access	Description
AIR_RATE_PERIOD Set only for CIBER record type 22 (specified in DETAIL.ASS_CIBER_EXT.CIBER_RECORD_TYPE) .	String	Write	If the value of AIR_CHARGE is **0**, this field is set to **00**. Otherwise, this field is set to **01**. **Note:** If the value of DETAIL.ASS_CIBER_EXT.SPECIAL_FEATURES_USED in the CIBER extension block is F, then AIR_CHARGE is always set to **0.0**, and therefore AIR_RATE_PERIOD is set to **00**. The value of AIR_RATE_PERIOD is generally derived from the time interval code in the charge packet where RUM equals the value of the **AirRum** registry parameter. The time interval code is part of the time model that you define. Therefore, the fields used to derive the CIBER extension AIR_RATE_PERIOD field is specific to your customization.
AIR_MULTIRATE_PERIOD Set only for CIBER record type 22 (specified in DETAIL.ASS_CIBER_EXT.CIBER_RECORD_TYPE) .	N/A	Write	If the value of AIR_CHARGE is **0**, this field is set to **0**. Otherwise, this field is set to **1**. **Note:** If the value of DETAIL.ASS_CIBER_EXT.SPECIAL_FEATURES_USED in the CIBER extension block is F, then AIR_CHARGE is always set to **0.0**, and therefore AIR_MULTIRATE_PERIOD is set to **0**. The value of AIR_MULTIRATE_PERIOD is generally derived from the time interval code in the charge packet where RUM equals the value of the **AirRum** registry parameter. The time interval code is part of the time model that you define. Therefore, the fields used to derive the CIBER extension AIR_RATE_PERIOD field is specific to your customization.
TOLL_CHARGE Set only for CIBER record type 22 (specified in DETAIL.ASS_CIBER_EXT.CIBER_RECORD_TYPE) .	N/A	Write	If the value of AIR_CHARGABLE_TIME in the CIBER extension block is **0**, TOLL_CHARGE is also set to **0**. Otherwise, it is set to the sum of the DETAIL.ASS_CBD.CHARGED_AMOUNT_VALUE in each charge packet for which all of the following statements are true: • RUM equals the value specified in the **AirRum** parameter in the registry • CHARGED_CURRENCY_TYPE equals "R" (rated) or "H" (home) • PRICEMODEL_TYPE equals "S" or "A"

Table 6-3 (Cont.) ISC_CiberOutputMapping EDR Container Fields

Alias field name Default field name	Type	Access	Description
TOLL_RATE_PERIOD (for CIBER record type 22 only)	N/A	Write	If the value of TOLL_CHARGE is **0**, this field is set to **00**. Otherwise, this field is set to **01**. The value of TOLL_RATE_PERIOD is generally derived from the time interval code in the charge packet where RUM equals the value of the **TollRum** registry parameter. The time interval code is part of the time model that you define. Therefore, the fields used to derive the CIBER extension TOLL_RATE_PERIOD field are specific to your customization.
TOLL_MULTIRATE_PERIOD (for CIBER record type 22 only)	N/A	Write	If the value of TOLL_CHARGE is **0**, this field is set to **0**. Otherwise, this field is set to **1**. The value of TOLL_MULTIRATE_PERIOD is generally derived from the time interval code in the charge packet where RUM equals the value of the **TollRum** registry parameter. The time interval code is part of the time model that you define. Therefore, the fields used to derive the CIBER extension TOLL_MULTIRATE_PERIOD field are specific to your customization.
TOLL_RATE_PERIOD and TOLL_MULTIRATE_PERIOD (for CIBER record type 32 and 42 only) Sets these Charge NO 1 related fields: • **Charge NO 1** • **Charge NO 1 Indicator** • **Charge NO 1 Rate Period** • **Charge NO 1 Multi-Rate Period**	N/A	Write	**Charge NO 1** is set to the value of DETAIL.ASS_CBD.CP.CHARGED_AMOUNT_VALUE. The values for the other Charge NO 1 related fields are based on the CIBER record type: Record type 32: • **Indicator** is set to **17** • **Rate Period** is set to **01** • **Multi-Rate Period** is set to **1** Record type 42: • **Indicator** is set to **12** • **Rate Period** is set to **04** • **Multi-Rate Period** is set to **1**

ISC_CiberRejectReason

The ISC_CiberRejectReason iScript sets a reason code in the CIBER extension block for records that are rejected for one of the following reasons:

- They are duplicates.
- There is no roaming agreement with the network operator.

Sample Registry

```
ISC_CiberRejectReason
{
  ModuleName = FCT_IScript
  Module
  {
     Active = TRUE
     Source = FILE
     Scripts
     {
        CiberRejectReason
        {
           FileName = ./iScriptLib/iScriptLib_CiberValidation/ISC_CiberRejectReason.isc
        }
     }
  }
}
```

ISC_ConsolidatedCP

The ISC_ConsolidatedCP iScript is used in the Incollect settlement pipeline and the Outcollect settlement pipeline. This iScript removes all non '00' impact category charge packets.

The TAP input grammar creates individual charge packets as well as consolidated charge packets. However, the FCT_BillingRecord module considers all charge packets for creating balance packets. For this reason, the individual charge packets (non '00' impact category charge packets) are removed and only consolidated charge packets are considered so that the balance amounts in the balance packets are correct.

This iScript also assigns the G/L ID to each consolidated charge packet based on the **GL_CODE** registry entry.

Registry Parameters

Table 6-4 lists the ISC_ConsolidatedCP registry parameter.

Table 6-4 ISC_ConsolidatedCP Registry Parameter

Registry Parameter	Description	Mandatory
GL_CODE	Specifies the G/L ID used for tracking the balance impacts of roaming usage events.	Yes

Sample Registry

```
ConsolidatedCP
{
   ModuleName = FCT_IScript
   Module
   {
      Active = True
      Source = File
      Scripts
      {
         ConsolidatedCPIScript
```

```
            {
               FileName = ./iScriptLib/iScriptLib_Roaming/ISC_ConsolidatedCP.isc
               GL_CODE = 1514
            }
         }
      }
}
```

EDR Container Fields

Table 6-5 lists the ISC_ConsolidatedCP EDR container fields.

Table 6-5 ISC_ConsolidatedCP EDR Container Fields

Alias field name Default field name	Type	Access	Description
DETAIL.ASS_CBD.CP.IMPACT_CATEGORY	String	Read	Contains the impact category of the charge packet.
DETAIL.ASS_CBD.CP.RUM_ID	Long	Write	Contains the RUM ID.
DETAIL.ASS_CBD.CP.USAGE_GL_ACCOUNT_CODE	String	Write	Contains the G/L account.
DETAIL.ASS_CBD.TP.RELATED_CHARGE_INFO_ID	Integer	Write	Contains the corresponding charge packet index.

ISC_DupRAPRecords

The ISC_DupRAPRecords iScript is used in the RAP processing pipeline. It duplicates severe and fatal TAP records so that the records can be routed to multiple output streams.

Registry Parameters

Table 6-6 lists the ISC_DupRAPRecords registry parameter.

Table 6-6 ISC_DupRAPRecords Registry Parameter

Registry Parameter	Description	Mandatory
NULLStream	Specifies the DevNull output stream. See "OUT_DevNull".	Yes

Sample Registry

```
DupRAPRecords
{
   ModuleName = FCT_IScript
   Module
   {
      Active = True
      Source = File
      Scripts
      {
         DupRAPRecords
         {
```

```
            FileName = ./iScriptLib/iScriptLib_Roaming/ISC_DupRAPRecords.isc
            NULLStream = DevNull
         }
      }
   }
}
```

EDR Container Fields

Table 6-7 lists the ISC_DupRAPRecords EDR container fields.

Table 6-7 ISC_DupRAPRecords EDR Container Fields

Alias field name Default field name	Type	Access	Description
DETAIL.ERROR_REJECT_TYPE	String	Write	FCT_Reject uses this to reject the detail to a stream other than the standard reject stream.
DETAIL.ASS_ROAMING_EXT.RAP_RECORD_TYPE	String	Read	RAP record type.

ISC_EDRToTAPOUTMap

The ISC_EDRToTAPOUTMap iScript is used to populate standard values to fields in the output TAP file based on its corresponding value in the EDR container. This is because for some fields EDR might have different internal representations and the TAP specification may ask for different representation for the same fields.

This iScript has to be customized by the user for mapping fields and specifying what is the internal EDR representation for the TAP fields specified value given in the standards document.

> **Note:**
>
> The ISC_EDRToTAPOUTMap iScript is a deprecated module but remains in BRM for backward compatibility.

Sample Registry

```
ISC_EDRToTAPOUTMap
{
  ModuleName = FCT_IScript
  Module
  {
     Active = TRUE
     Source = FILE
     Scripts
     {
       EDRToTAPOUTMap
       {
            FileName = ./iScriptLib/iScriptLib_Standard/ISC_EDRToTAPOUTMap.isc
       }
     }
```

```
    }
}
```

EDR Container Fields

Table 6-8 lists the ISC_EDRToTAPOUTMap EDR container fields.

Table 6-8 ISC_EDRToTAPOUTMap EDR Container Fields

Alias field name Default field name	Type	Access	Description
DETAIL.ASS_CBD.CP.DAY_CODE	String	Read / Write	External Day Code as estimated and used by the rating process. Sample mapping given in the script: (Can be customized as per user's requirement) WEEKDAY => N WEEKEND => P ALLDAYS => I
DETAIL.ASS_CBD.CP.TIME_INTERVAL_CODE	String	Read / Write	External Time Interval Code as estimated and used by the rating process. Sample mapping given in the script (Can be customized as per user's requirement): 20012 - O 20021 - I 20031 - P 20032 - O 20033 - O 20001 - P 0002 - S 20003 - S 20004 - S 20011 - P
DETAIL.ASS_CBD.CP.RUM	String	Read / Write	Classifies the charging part of a call in an intercarrier relationship, for interconnection or roaming. Sample mapping given in the script (Can be customized as per user's requirement): DUR - D SND - V REC - W EVT - E AIR - D OCC - F

ISC_FirstProductRealtime

This iScript sets the validity period of charge offers that start on first usage when they are used for the first time to rate an event in the real-time rerating pipeline.

> **Note:**
>
> To process first-usage events in the batch rating pipeline, use the "FCT_FirstUsageNotify" module.

Dependencies

Place this iScript in the real-time rerating pipeline before the FCT_DiscountAnalysis module.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Sample Registry

```
FirstProductRealtime
{
    ModuleName = FCT_IScript
    Module
    {
        Active = True
        Source = File
        Scripts
        {
            FirstProductRealtime
            {
                 FileName = ./iScriptLib/iScriptLib_Standard/ISC_FirstProductRealtime.isc
            }
        }
    }
}
```

EDR Container Fields

Table 6-9 lists the ISC_FirstProductRealtime EDR container fields.

Table 6-9 ISC_FirstProductRealtime EDR Container Fields

Alias field name Default field name	Type	Access	Description
DETAIL.CUST_A.INTERN_FOUND_PP_INDEX	Decimal	Read	Contains an index of the customer's purchased charge offers that were used for rating.
DETAIL.CUST_A.PRODUCT.FIRST_USAGE_INDICATOR	Decimal	Read	Specifies whether the charge offer is configured to start when first used and the first-usage validity period status.
DETAIL.CUST_A.ACCOUNT_PARENT_ID	String	Read	Specifies the customer account POID.
DETAIL.CUST_A.PRODUCT.OFFERING_POID	String	Read	Specifies the account's charge offer POID.
DETAIL.CUST_A.PRODUCT.SERVICE_ID	String	Read	Specifies the charge offer's service POID.
DETAIL.CUST_A.PRODUCT.SERVICE_TYPE	String	Read	Specifies the charge offer's service type.
DETAIL.CHARGING_START_TIMESTAMP	Date	Read	Specifies the EDR's start timestamp.

Table 6-9 (Cont.) ISC_FirstProductRealtime EDR Container Fields

Alias field name Default field name	Type	Access	Description
DETAIL.EVENT_ID	Decimal	Read	Specifies the event POID.
DETAIL.EVENT_TYPE	String	Read	Specifies the event type.
DETAIL.UTC_TIME_OFFSET	String	Read	Specifies the UTC time offset.
DETAIL.REFRESH_BALANCE	Decimal	Write	Specifies whether the account's charge offer validity period has been updated in the BRM database. If set, the discount module retrieves the updated balance information before evaluating discounts for the event.

ISC_GetCamelFlag

The ISC_GetCamelFlag iScript is used in the roaming outcollect processing to retrieve the CAMEL flag information for a roaming partner. The CAMEL flag information is used during EDR processing.

Dependencies

The ISC_GetCamelFlag iScript must run before the FCT_PreRating module in the outcollect rating pipeline.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Sample Registry

```
GetCamelFlag
{
     ModuleName = FCT_IScript
     Module
     {
          Active = True
          Source = FILE
          Scripts
          {
               CamelFlagIScript
               {
                    FileName =./iScriptLib/iScriptLib_Roaming/ISC_GetCamelFlag.isc
               }
          }
     }
}
```

ISC_GetResourceBalance

The ISC_GetResourceBalance iScript is used to get the memory balance of a balance element. This iScript returns either the balance amount or '0' if the balance retrieval is successfull. If a failure occurs, it returns '-1'.

The iScript should be in the same function pool and added after FCT_ApplyBalance in the batch pipeline.

Sample Registry

```
NewBalanceAPI
{
     ModuleName = FCT_IScript
     Module
     {
          Active = True
          Source = FILE
          Scripts
          {
               NewBalanceAPI
               {
                    FileName =./iScriptLib/iScriptLib_Standard/ISC_RetrieveBalance.isc
               }
          }
     }
}
```

ISC_LeastCost

The ISC_LeastCost iScript performs the following:

- Calculates and finds the lowest charge for an EDR. See "About Least Cost Rating" in *BRM Setting Up Pipeline Rating and Discounting*.
- Calculates the total savings between the charge for a promotional charge offer and the charge for the lowest priority (base) charge offer. See "About Calculating the Promotional Savings" in *BRM Setting Up Pipeline Rating and Discounting*.

Dependencies

This module must run after FCT_CustomerRating.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Parameters

Table 6-10 lists the ISC_LeastCost registry parameters.

Table 6-10 ISC_LeastCost Registry Parameters

Registry Parameter	Description	Mandatory
Resource	Specifies the balance element type that this module uses to calculate any savings amount.	Yes
Resource_ID	Specifies the balance element IDs used to identify the balance element used when calculating the savings amount.	Yes

Sample Registry

```
FCT_LeastCostRating
{
    ModuleName = FCT_IScript
    Module
    {
        Active = True
        Source = File
        Scripts
        {
            myScript
            {
                FileName = ./ISC_LeastCost.isc
                Resource = "Saving Charge Resource"
                Resource_ID = "1000100"
            }
        }
    }
}
```

EDR Container Fields

Table 6-11 lists the ISC_LeastCost EDR container fields.

Table 6-11 ISC_LeastCost EDR Container Fields

Alias field name Default field name	Type	Access	Description
DETAIL.ASS_CBD.CP.CHARGED_AMOUNT_VALUE	Integer	Read/write	The amount charged to the customer.
DETAIL.ASS_CBD.DP.AMOUNT	Integer	Read	The amount of discount between the amount charged to the customer, and a greater amount that could have been charged.
DETAIL.ASS_CBD.CP.RATEPLAN_CODE	String	Read	The code identifying the least cost rating charge.
DETAIL.CUST_A.PRODUCT.RATEPLAN_NAME	String	Read	A description of the least cost rating charge code.
DETAIL.CUST_A.PRODUCT_PRIORITY	String	Read	Contains a list of the priorities for all charge offers that are associated with the same service and event.
DETAIL.CUST_A.INTERN_RATING_PRODUCTS	String	Write	Contains the charge offer rating indexes. This is a comma-separated list of all rating charge offers' indexes associated with the same service and event, and their priorities.
DETAIL.ASS_CBD.CP.RESOURCE	String	Write	The balance element to impact for reporting promotional savings.
DETAIL.ASS_CBD.CP.RESOURCE_ID	Integer	Write	The ID of the balance element to impact for promotional savings.
DETAIL.ASS_CBD.CP.RESOURCE_TYPE	Integer	Write	The savings charge packet to impact. This is **992** by default.

ISC_MapNetworkOperatorInfo

The ISC_MapNetworkOperatorInfo iScript maps the DETAIL.SOURCE_NETWORK field to the PIN_FLD_ORIGIN_NETWORK field and the DETAIL.DESTINATION_NETWORK field to the PIN_FLD_DESTINATION_NETWORK field of the opcode input block for the corresponding event.

Dependencies

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Sample Registry

```
MapNetworkOperatorInfo
  {
   ModuleName = FCT_IScript
   Module
   {
     Active = TRUE
     Source = File
     Scripts
     {
       PreOpcode
       {
         FileName = ./iScriptLib/AAA/ISC_MapNetworkOperatorInfo.isc
       }
     }
   }
  }
```

ISC_Migration

Use the ISC_Migration iScript during account migration to automatically flag EDRs for suspension. The FCT_Suspense module can then route the EDRs to a separate suspense output stream.

Sample Registry

```
MigrationPlugIn
{
  ModuleName = FCT_IScript
  Module
  {
    Active = True
    Source = File
    Scripts
    {
      MigrationIScript
      {
        FileName = ./samples/wireless_splitter/ISC_Migration.isc
      }
    }
  }
}
```

ISC_MiscOutcollect

The ISC_MiscOutcollect iScript is used in the Outcollect rating pipeline. This module adds BASIC_SERVICE and SUPPLEMENTARY_SERVICE blocks to the EDR container for GSM services. This is done to ensure that the TAP file generated by the pipeline contains all the required information.

For GPRS services, this module adjusts the record number field in the GPRS extension block to **0**.

It also modifies the RUM field of the charge packets as follows:

- DUR is replaced by D.
- SND is replaced by V.
- REC is replaced by W.

Sample Registry

```
MiscAddInfo
{
   ModuleName = FCT_IScript
   Module
   {
      Active = True
      Source = File
      Scripts
      {
         MiscAddInfoIScript
         {
            FileName = ./iScriptLib/iScriptLib_Roaming/ISC_MiscOutCollect.isc
         }
      }
   }
}
```

EDR Container Fields

Table 6-12 lists the ISC_MiscOutcollect EDR container fields.

Table 6-12 ISC_MiscOutcollect EDR Container Fields

Alias field name Default field name	Type	Access	Description
DETAIL.ASS_GSMW_EXT.BS_PACKET.RECORD_TYPE	String	Write	Contains the record type field.
DETAIL.ASS_GSMW_EXT.BS_PACKET.RECORD_NUMBER	Integer	Write	Contains the record number field.
DETAIL.ASS_GSMW_EXT.BS_PACKET.CHAIN_REFERENCE	String	Write	Contains the call reference.
DETAIL.ASS_GSMW_EXT.BS_PACKET.LONG_DURATION_INDICATOR	String	Write	Contains whether the call is a long duration or not.
DETAIL.ASS_GSMW_EXT.BS_PACKET.BASIC_SERVICE	String	Write	Contains the basic service of the call.

Table 6-12 (Cont.) ISC_MiscOutcollect EDR Container Fields

Alias field name Default field name	Type	Access	Description
DETAIL.ASS_GSMW_EXT.BS_PACKET.QOS_REQUESTED	String	Write	Contains the QOS requested.
DETAIL.ASS_GSMW_EXT.BS_PACKET.QOS_USED	String	Write	Contains the QOS used.
DETAIL.ASS_GSMW_EXT.BS_PACKET.CHARGING_START_TIMESTAMP	Date	Write	Contains the call start time.
DETAIL.ASS_GSMW_EXT.BS_PACKET.CHARGING_END_TIMESTAMP	Date	Write	Contains the call end time.
DETAIL.ASS_GSMW_EXT.BS_PACKET.WHOLESALE_CHARGED_AMOUNT_VALUE	Decimal	Write	Contains the total charged amount value of the call.
DETAIL.ASS_GSMW_EXT.BS_PACKET.WHOLESALE_CHARGED_TAX_RATE	Decimal	Write	Contains the tax rate.
DETAIL.ASS_GSMW_EXT.BS_PACKET.SPEECH_VERSION_REQUESTED	String	Write	Contains the speech version requested.
DETAIL.ASS_GSMW_EXT.BS_PACKET.SPEECH_VERSION_USED,	String	Write	Contains the speech version used.
DETAIL.ASS_GSMW_EXT.SS_PACKET.RECORD_TYPE	String	Write	Contains the record type.
DETAIL.ASS_GSMW_EXT.SS_PACKET.RECORD_NUMBER	Integer	Write	Contains the record number.
DETAIL.ASS_GSMW_EXT.SS_PACKET.SS_EVENT	String	Write	Contains the event type.
DETAIL.ASS_GSMW_EXT.SS_PACKET.ACTION_CODE	String	Write	Contains the connect type.
DETAIL.ASS_CBD.CP.RUM	String	Read/Write	Contains the ratable unit of measurement.
DETAIL.ASS_GPRS_EXT.RECORD_NUMBER	Integer	Write	Contains the record number.

ISC_Monitoring

The ISC_Monitoring iScript records latencies for authentication, authorization, and accounting requests.

Dependencies

The ISC_Monitoring iScript depends on the ISC_StartTime iScript.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Parameters

Table 6-13 lists the ISC_Monitoring registry parameters.

Table 6-13 ISC_Monitoring Registry Parameters

Registry Parameter	Description	Mandatory
FileName	Specifies the location of the ISC_Monitoring iScript.	Yes
recordsPerFile	Specifies the number of records per event log file.	Yes
recordsPerWrite	Specifies the number of records per write operation.	Yes
eventLogDir	Specifies the directory for the event log file.	Yes

Sample Registry

```
MonitoringPlugIn
{
   ModuleName = FCT_IScript
   Module
     {
       Active = True
       Source = FILE
       Scripts
       {
          ReadIScript
          {
             FileName = ./iScriptLib/AAA/ISC_Monitoring.isc
             recordsPerFile = 1000 # number of records per event log file
             recordsPerWrite = 10 # number of records per write operation
             eventLogDir = ./log/dump # directory for series of event log file
          }
       }
   }
}
```

EDR Container Fields

Table 6-14 lists the ISC_Monitoring EDR container fields.

Table 6-14 ISC_Monitoring EDR Container Fields

Alias field name Default field name	Type	Access	Description
DETAIL.MILLISEC_TIME	Integer	Read	Specifies the latency time in milliseconds.
DETAIL.OPCODE_NUM	Integer	Read	Number of the BRM opcode that performs the requested action.

ISC_NRTRDE_ErrorReport

The ISC_NRTRDE_ErrorReport iScript is used during roaming incollect processing by the NRTRDE processing pipeline. This iScript collects the validation errors in the EDRs and creates error records in the Pipeline Manager database. It also collects NRTRDE file processing information and creates file processing records in the Pipeline Manager database. Information stored in the validation and file processing records in the database are used for generating NRTRDE reports.

Dependencies

The ISC_NRTRDE_ErrorReport iScript must run after the ISC_NrtrdeHeaderValidation iScript.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Parameters

The registry parameter is **FileName**. It is mandatory. **FileName** specifies the location of the iScript. The default location is:

`./iScriptLib/iScriptLib_Roaming/ISC_NRTRDE_ErrorReport.isc`

Sample Registry

```
NRTRDE_ErrorReport
{
   ModuleName = FCT_IScript
   Module
   {
      Active = True
      Source = FILE
      Scripts
      {
         NRTRDE_ErrorReport
        {
            FileName = ./iScriptLib/iScriptLib_Roaming/ISC_NRTRDE_ErrorReport.isc
            DatabaseConnection = ifw.DataPool.Login
        }
      }
   }
}
```

ISC_NRTRDE_EventSplit

The ISC_NRTRDE_EventSplit iScript is used by roaming outcollect processing to duplicate and route EDRs to the corresponding roaming partner's NRTRDE output streams based on the roaming partner's NRTRDE flag.

Dependencies

The ISC_NRTRDE_EventSplit iScript must run after the FCT_EnhancedSplitting module in the outcollect rating pipeline.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Parameters

Table 6-15 lists the ISC_NRTRDE_EventSplit registry parameters.

Table 6-15 ISC_NRTRDE_EventSplit Registry Parameters

Registry Parameter	Description	Mandatory
FileName	Specifies the location of the ISC_NRTRDE_EventSplit iScript. The default location is: `./iScriptLib/iScriptLib_Roaming/ISC_NRTRDE_EventSplit.isc`	Yes
NRTRDE_STREAM_PATTERN	Specifies the name of the NRTRDE stream pattern. This parameter appends the PLMN to the NRTRDE stream pattern to retrieve the output stream name.	Yes

Sample Registry

```
NRTRDESplit
{
   ModuleName = FCT_IScript
   Module
   {
      Active = True
      Source = File
      Scripts
      {
         NRTRDESplit
         {
            FileName = ./iScriptLib/iScriptLib_Roaming/ISC_NRTRDE_EventSplit.isc
            NRTRDE_STREAM_PATTERN = NRTRDEOutput_
         }
      }
    }
}
```

ISC_NrtrdeHeaderValidation_v2_01

The ISC_NrtrdeHeaderValidation_v2_01 iScript is used during roaming incollect processing by the NRTRDE processing pipeline. This iScript validates the information in the header record of the TD35 file based on the TD35 specifications.

Dependencies

The ISC_NrtrdeHeaderValidation_v2_01 iScript must run before any other modules in the NRTRDE processing pipeline.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Parameters

The registry parameter is **FileName**. It is mandatory. **FileName** specifies the location of the iScript. The default location is:

`./iScriptLib/iScriptLib_Tap3Validation/ISC_NrtrdeHeaderValidation_v2_01.isc`

Sample Registry

```
TapValidationIScripts
{
   ModuleName = FCT_IScript
   Module
   {
      Active = True
      Source = FILE
      Scripts
      {
         NrtrdeHeaderValidation
         {
             FileName = ./iScriptLib/iScriptLib_Tap3Validation/ISC_NrtrdeHeaderValidation_v2_01.isc
         }
      }
   }
}
```

ISC_ObjectCacheTypeOutputSplitter

Use the ISC_ObjectCacheTypeOutputSplitter iScript to enter a value in an EDR to create two identical output files from a single input EDR and write them to separate output streams.

Dependencies

To use this iScript, you must have object cache residency distinction enabled in your system.

Requires a connection to the DAT_BalanceBatch module.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Sample Registry

```
ObjectCacheTypeOutputSplitter_Script
{
  ModuleName = FCT_IScript
  Module
  {
    Active = True
    Source = File

    Scripts
    {
      ObjectCacheTypeOutputSplit
      {
        FileName = ./iScriptLib/iScriptLib_Standard/ISC_ObjectCacheTypeOutputSplitter.isc
      }
    }
  }
}
```

ISC_OverrideRateTag

The ISC_OverrideRateTag iScript is used in the outcollect settlement pipelines to populate the RATE_TAG field with the value of the NRTRDE flag in the balance impact.

The value of the RATE_TAG field is used by high-usage reports to filter out NRTRDE operator data from the report.

Dependencies

The ISC_OverrideRateTag iScript must run after the FCT_BillingRecord module in the outcollect settlement pipeline.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Sample Registry

```
OverrideRateTag
{
     ModuleName = FCT_IScript
     Module
     {
          Active = True
          Source = FILE
          Scripts
          {
               OverrideRateTagIScript
               {
                    FileName =./iScriptLib/iScriptLib_Roaming/ISC_OverrideRateTag.isc
               }
          }
     }
}
```

ISC_OverrideSuspenseParams

The ISC_OverrideSuspenseParams iScript overrides some fields in the Suspense Extension block of the EDR container that is set by Suspense Manager.

During Outcollect processing, this iScript in the RAP Processing pipeline overrides the PIPELINE_NAME suspense field. For severe TAP records, PIPELINE_NAME is set to the name of the outcollect rating pipeline. For fatal TAP records, PIPELINE_NAME is set to the Suspense Batch output stream.

Registry Parameters

Table 6-16 lists the ISC_OverrideSuspenseParams registry parameters.

Table 6-16 ISC_OverrideSuspenseParams Registry Parameters

Registry Parameter	Description	Mandatory
TAPFilePrefix	Specifies the prefix of the TAP file. • **CD** for files containing chargeable data • **TD** for files containing test data	Yes
TAPOutCollectPipeline	Specifies the name of the outcollect rating pipeline.	Yes
TAPCorrectionPipeline	Specifies the path of the directory from which the original TAP file was sent to the network operator.	Yes
TAPSentArchivePath	Specifies the path of the directory from which the original TAP file was sent to the network operator.	Yes
SBRStream	Specifies the output stream that generates the suspense batch file.	Yes

Sample Registry

```
OverrideSuspenseParams
{
   ModuleName = FCT_IScript
   Module
   {
      Active = True
      Source = File
      Scripts
      {
         OverrideSuspenseParams
         {
            FileName = ./iScriptLib/iScriptLib_Roaming/ISC_OverrideSuspenseParams.isc
            TAPFilePrefix = CD
            TAPOutCollectPipeline = TAPOutCollectPipeline
            TAPCorrectionPipeline = TAPCorrectionPipeline

            #the following path must be absolute
            TAPSentArchivePath = ./data/outcollect/tapout/sent
            SBRStream = SBROutput
         }
      }
   }
}
```

EDR Container Fields

Table 6-17 lists the ISC_OverrideSuspenseParams EDR container fields.

Table 6-17 ISC_OverrideSuspenseParams EDR Container Fields

Alias field name Default field name	Type	Access	Description
DETAIL.ASS_ROAMING_EXT.TAP_FILE_SEQ_NO	Integer	Read	Sequence number of the TAP file.
DETAIL.ASS_ROAMING_EXT.RAP_RECORD_TYPE	String	Read	RAP record type.
DETAIL.BATCH_ID	String	Write	Records the TAP batch ID.
DETAIL.ORIGINAL_BATCH_ID	String	Write	Records the TAP batch ID.

Table 6-17 (Cont.) ISC_OverrideSuspenseParams EDR Container Fields

Alias field name Default field name	Type	Access	Description
DETAIL.ASS_SUSPENSE_EXT.SUSPENDED_FROM_BATCH_ID	String	Write	Records the TAP batch ID.
DETAIL.ASS_SUSPENSE_EXT.SOURCE_FILENAME	String	Write	Records the TAP file name.
DETAIL.ASS_SUSPENSE_EXT.PIPELINE_NAME	String	Write	For a severe error, records the TAP outcollect rating pipeline name. For a fatal error, records the TAP correction pipeline name.
DETAIL.ASS_SUSPENSE_EXT.RECYCLE_KEY	String	Write	Records the RAP file sequence number.
DETAIL.ASS_ROAMING_EXT.TAP_FILE_PATH	String	Write	The path of the directory from which the original TAP file was sent to the network operator (Only for fatal errors).
DETAIL.ASS_ROAMING_EXT.SUSPENSION_TIME	Date	Write	(Only for fatal errors) Suspension time of the TAP file.

ISC_PopulateOpcodeandUtilBlock_Diameter

The ISC_PopulateOpcodeandUtilBlock_Diameter iScript adds an opcode block in the EDR. In the Dispatcher pipeline, the EDR is duplicated to handle failover. The ISC_PopulateOpcodeandUtilBlock_Diameter iScript improves the performance by reducing the time taken to duplicate the EDR. Also, it provides the flexibility of performing minor validations. This iScript populates the opcode block and other relevant fields like OPCODE_NODE and OPCODE_NUMBER, which are required by the TimeoutRouter pipeline.

Dependencies

This iScript must be called before calling FCT_Opcode.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Sample Registry

```
ProcessPipeline
  {
    PopulateOpcodeAndUtilBlock
  {
    ModuleName? = FCT_IScript
    Module
     {
       Active = TRUE
       Source = File
       Scripts
       {
         PreOpcode?
         {
           FileName? = ./iScriptLib/AAA/ISC_PopulateOpcodeAndUtilBlock.isc
     }
```

```
    }
}
```

ISC_PostRating

The ISC_PostRating iScript adds all the retail and wholesale charges and puts them in the DETAIL.RETAIL_CHARGED_AMOUNT_VALUE and DETAIL.WHOLESALE_CHARGED_AMOUNT_VALUE fields.

See "Billing Consolidation with CIBER Roaming and Revenue Assurance" in *BRM Setting Up Pipeline Rating and Discounting*.

Dependencies

This module must run:

- After rating modules FCT_CustomerRating, FCT_PreRating, and FCT_MainRating

  or

- After the FCT_ExchangeRate module

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Sample Registry

```
PostRating
{
    ModuleName = FCT_IScript
    Module
    {
        Active = True
        Source = File

        Scripts
        {
           PostRating
           {
              FileName = ./ISC_PostRating.isc

              RetailRecordType = 981
              RetailResource = DEM
              RetailPricemodelType = S
              RetailCurrencyType = R

              WholesaleRecordType = 990
              WholesaleResource = DEM
              WholesalePricemodelType = S
              WholesaleCurrencyType = R
           }
        }
    }
}
```

EDR Container Fields

Table 6-18 lists the ISC_PostRating input EDR container fields.

Table 6-18 ISC_PostRating Input EDR Container Fields

Alias field name Default field name	Type	Access	Description
DETAIL.ASS_CBD.RECORD_TYPE	String	N/A	Contains the charge breakdown record type (retail or wholesale).
DETAIL.ASS_CBD.CP.RESOURCE	String	N/A	Contains the charge breakdown balance element type.
DETAIL.ASS_CBD.CP.PRICEMODEL_TYPE	String	N/A	Contains the charge breakdown pricing.
DETAIL.ASS_CBD.CP.CHARGED_CURRENCY_TYPE	String	N/A	Contains the charge breakdown currency type for charged amount.
DETAIL.ASS_CBD.CP.IMPACT_CATEGORY	String	N/A	Contains the charge breakdown impact category.
DETAIL.ASS_CBD.CP.CHARGED_AMOUNT_CURRENCY	String	N/A	Contains the charge breakdown currency type for charged amount.
DETAIL.ASS_CBD.CP.CHARGED_TAX_TREATMENT	String	N/A	Contains the charge breakdown tax treatment type.
DETAIL.ASS_CBD.CP.CHARGED_AMOUNT_VALUE	Decimal	N/A	Contains the charge breakdown charged amount.

Table 6-19 lists the ISC_PostRating output EDR container fields.

Table 6-19 ISC_PostRating Output EDR Container Fields

Alias field name Default field name	Type	Access	Description
DETAIL.RETAIL_IMPACT_CATEGORY	String	N/A	Contains the retail impact category.
DETAIL.RETAIL_CHARGED_AMOUNT_VALUE	Decimal	N/A	Contains the retail charged amount value.
DETAIL.RETAIL_CHARGED_AMOUNT_CURRENCY	String	N/A	Contains the retail charged amount currency.
DETAIL.RETAIL_CHARGED_TAX_TREATMENT	String	N/A	Contains the retail charged amount tax treatment.
DETAIL.WHOLESALE_IMPACT_CATEGORY	String	N/A	Contains the wholesale impact category.
DETAIL.WHOLESALE_CHARGED_AMOUNT_VALUE	Decimal	N/A	Contains the wholesale charged amount value.
DETAIL.WHOLESALE_CHARGED_AMOUNT_CURRENCY	String	N/A	Contains the wholesale charged amount currency.
DETAIL.WHOLESALE_CHARGED_TAX_TREATMENT	String	N/A	Contains the wholesale charged amount tax treatment.

ISC_ProfileAnalyzer

For each EDR, the ISC_ProfileAnalyzer iScript compares the telephone numbers or other relevant data in EDRs with the ERAs that the customer's service owns. ISC_ProfileAnalyzer stores each ERA label that matches the relevant EDR container field.

The iScript adds the label names to the EDR container field DETAIL.PROFILE_LABEL_LIST. If there are multiple names, the names are separated by a comma by default. You can change

the separator character in the registry. If a label name contains a comma, you need to change the separator character.

Dependencies

ISC_ProfileAnalyzer depends on the ERA values populated by FCT_Account. It must be run after FCT_Account and before any rating modules in the pipeline.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Sample Registry

```
ProfileAnalyzer
{
   ModuleName = FCT_IScript
   Module
   {
      Active = True
      Source = FILE
      Scripts
      {
        ProfileAnalyzer
        {
        ServiceType = TEL
        ProfileName = FRIENDS_FAMILY
        LabelSeparator = ,
        FileName = ./iScriptLib/iScriptLib_Standard/ISC_ProfileAnalyzer.isc
        }
     }
   }
}
```

EDR Container Fields

The ISC_ProfileAnalyzer iScript uses the EDR container fields listed in Table 6-20:

Table 6-20 ISC_ProfileAnalyzer EDR Container Fields

Alias field name Default field name	Type	Access	Description
PROFILE_LABEL_LIST DETAIL. PROFILE_LABEL_LIST	String	Write	Contains unique profile labels when the profile attributes match an EDR container field used for comparison to find F&F list.
DETAIL.CUST_A.PRODUCT. ERA.LABEL	String	Read	Contains a label associated with the service: for example, MYFAMILY.
DETAIL.CUST_A.SHARED_PROFILE_LIST. ERA.LABEL	String	Read	Contains a label associated with the service profile from a shared profile.
ERA.PROFILE	String	Read	Contains the profile name: for example, "Friends&Family."
CUST_A. SHARED_PROFILE_LIST	Block	Read	Contain the profiles that the service owns or shares as a member of a profile sharing group.

Table 6-20 (Cont.) ISC_ProfileAnalyzer EDR Container Fields

Alias field name Default field name	Type	Access	Description
CUST_B.SHARED_PROFILE_LIST	Block	Read	Contain the profiles that the service owns or shares as a member of a profile sharing group.
CUST_A. SHARED_PROFILE_LIST .ERA	Block	Read	Contains shared ERA information.
CUST_B.SHARED_PROFILE_LIST.ERA	Block	Read	N/A
ERA.NAME	String	Read	Contains the profile attribute name.
ERA.VALUE	String	Read	Contains the profile attribute value.

ISC_ProfileLabel

The ISC_ProfileLabel iScript is used when rating CDRs based on ERAs. It determines whether the ERA profiles specified in the **ProfileName** registry entry match the EDR field value and populates the DETAIL.PROFILE_LABEL_LIST field with the ERA labels of the matching ERAs, and the DETAIL.USAGE_TYPE field with appropriate usage type for the ERA.

Dependencies

The ISC_ProfileLabel iScript must run after the FCT_Account module and before any rating modules.

Requires a connection to DAT_AccountBatch. This iScript works only in batch rating.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Parameters

Table 6-21 lists the ISC_ProfileLabel registry parameters.

Table 6-21 ISC_ProfileLabel Registry Parameters

Registry Parameter	Description	Mandatory
ProfileName	Specifies the name of the ERA profile to analyze.	Yes
LabelSeparator	ERA labels in the DETAIL.PROFILE_LABEL_LIST EDR field are separated using this delimiter. The default is a comma (**,**).	No
FileName	Specifies the location of the ISC_ProfileLabel iScript. The default location is **./iScriptLib/iScriptLib_Standard/ISC_ProfileLabel.isc**.	Yes

Sample Registry

```
ProfileLabel
{
  ModuleName = FCT_IScript
  Module
  {
    Active = True
```

```
    Source = FILE
    Scripts
    {
      ProfileLabel
      {
      ProfileName = FRIENDS_FAMILY
      LabelSeparator = ,
      FileName = ./iScriptLib/iScriptLib_Standard/ISC_ProfileLabel.isc
      }
    }
  }
}
```

EDR Container Fields

Table 6-22 list the ISC_ProfileLabel EDR container fields.

Table 6-22 ISC_ProfileLabel EDR Container Fields

Alias field name Default field name	Type	Access	Description
CUST_A_INTERN_PP_INDEX DETAIL.CUST_A.INTERN_FOUND_PP_INDEX	Integer	Read	Contains an index of the customer's purchased charge offers identified by the FCT_Account module.
CUST_A.ACCOUNT_PARENT_ID DETAIL.CUST_A.ACCOUNT_PARENT_ID	String	Read	Account ID of the service for which usage is getting rated.
CUST_A.PRODUCT.SERVICE_ID DETAIL.CUST_A.PRODUCT.SERVICE_ID	String	Read	Service ID for which usage is getting rated.
B_NUMBER DETAIL.B_NUMBER	String	Read	Called number of the event.
BDR_CHARGING_START_TIMESTAMP DETAIL.CHARGING_START_TIMESTAMP	Date	Read	Contains the event's starting timestamp. The timezone information of this timestamp is stored in the BDR_UTC_TIME_OFFSET field.
BDR_UTC_TIME_OFFSET DETAIL.UTC_TIME_OFFSET	String	Read	Contains the UTC time offset that normalizes the charging start timestamp to the UTC tim zone. All validity timestamps in the BRM customer data are stored in normalized UTC time. The format is +/– *HHMM*.
PROFILE_LABEL_LIST DETAIL.PROFILE_LABEL_LIST	String	Write	Contains ERA labels that contain a value that matches the EDR field value.
USAGE_TYPE DETAIL.USAGE_TYPE	String	Write	Contains the usage type for the ERA.

ISC_RAP_0105_InMap

The ISC_RAP_0105_InMap iScript copies TAP data from staging fields in the EDR container to business fields in the EDR container. This iScript is used during roaming outcollect processing.

Dependencies

This should be the first module in the **FunctionPool**.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Sample Registry

```
BusinessMapping
{
  ModuleName = FCT_IScript
  Module
  {
     Active = True
     Source = File
     Scripts
     {
       BusinessMapping
       {
           FileName = ./iScriptLib/iScriptLib_Roaming/ISC_RAP_0105_InMap.isc
       }
     }
  }
}
```

ISC_RemoveASSCBD

The ISC_RemoveASSCBD iScript is used in the Outcollect rating pipeline to remove associated charge breakdown packets associated with RAP records that are recycled to the pipeline during RAP file processing.

Sample Registry

```
RemoveASSCBD
{
   ModuleName = FCT_IScript
   Module
   {
      Active = True
      Source = File
      Scripts
      {
         RemoveASSCBDIScript
         {
            FileName = ./iScriptLib/iScriptLib_Roaming/ISC_RemoveASSCBD.isc
         }
      }
   }
}
```

EDR Container Fields

Table 6-23 lists the ISC_RemoveASSCBD EDR container fields.

Table 6-23 ISC_RemoveASSCBD EDR Container Fields

Alias field name Default field name	Type	Access	Description
DETAIL.ASS_CBD.RECORD_NUMBER	Integer	Read/Write	Contains the record number.

ISC_RollbackSettlement

During roaming incollect and outcollect settlement processing, the ISC_RollbackSettlement iScript checks for errors in the EDR. When there is an error, it notifies the Transaction Manager to roll back the transactions in the settlement pipeline. The Transaction Manager then notifies the FCT_BatchSuspense module to suspend the entire input file.

Sample Registry

```
RollbackSettlement
{
   ModuleName = FCT_IScript
   Module
   {
      Active = True
      Source = File
      Scripts
      {
         RollbackIScript
         {
            FileName = ./iScriptLib/iScriptLib_Roaming/ISC_RollbackSettlement.isc
         }
      }
   }
}
```

ISC_SetAndValidateBatchInfo

The ISC_SetAndValidateBatchInfo iScript populates and validates the batch related fields for the EDR container.

The iScript validates the HEADER.BATCH_ID. If it does not exist, the entire batch is rejected. If it exists, then it copies the HEADER.BATCH_ID to DETAIL.BATCH_ID.

If DETAIL.EVENT_ID is missing in an EDR, the EDR is rejected.

Dependencies

This iScript must be placed at the beginning of the pipeline so that the batch ID is inserted before any further processing of the mediation batches. It should be used only if you do not use Suspense Manager.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Entries

Table 6-24 lists the ISC_SetAndValidateBatchInfo registry entries.

Table 6-24 ISC_SetAndValidateBatchInfo Registry Entries

Entry	Description	Mandatory
ValidateOriginalBatchId	• If **True**, and if DETAIL.ORIGINAL_BATCH_ID is missing, the EDR is rejected. • If **False**, it copies the HEADER.BATCH_ID in DETAIL.ORIGINAL_BATCH_ID.	Yes
KeepBatchIds	• If **True**, this iScript does not modify the values in DETAIL.ORIGINAL_BATCH_ID and DETAIL.BATCH_ID. • If **False**, and if **ValidateOriginalBatchId** is **True**, this iScript assigns the values in HEADER.BATCH_ID to DETAIL.BATCH_ID. • If **False**, and if **ValidateOriginalBatchId** is **False**, this iScript assigns the value in HEADER.BATCH_ID to DETAIL.ORIGINAL_BATCH_ID and DETAIL.BATCH_ID.	Yes

Sample Registry

```
SetAndValidateBatchInfo
   {
    ModuleName = FCT_IScript
     Module
     {
       Active = True
       Source = FILE
       Scripts
        {
          SetAndValidateBatchInfo
        {
          FileName = ./iScriptLib/iScriptLib_Standard/ISC_SetAndValidateBatchInfo.isc
          ValidateOriginalBatchId = TRUE
          KeepBatchIds = TRUE
        }
      }
    }
}
```

ISC_SetEDRStatus

The ISC_SetEDRStatus iScript sets the EDR status to **Success**, **Suspense**, **Duplicate**, **Discard**, or **Skipped** for each EDR.

Dependencies

This iScript must be used before FCT_AggreGate and before the scenario that collects audit data grouped on the EDRStatus field.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Sample Registry

```
SetEDRStatus
  {
    ModuleName = FCT_IScript
     Module
```

```
      {
       Active = True
        Source = FILE
         Scripts
      {
        SetEDRStatus
      {
        FileName = ./iScriptLib/iScriptLib_Standard/ISC_SetEDRStatus.isc
      }
    }
  }
}
```

ISC_SetOutputStream

The ISC_SetOutputStream iScript sets the Output Stream to **TelOut**, **SMSOut**, **GPRSOut**, **RejectOut**, or **DuplicateOut** for each EDR.

Dependencies

This iScript must be used after FCT_Reject, because it is dependent on the fields set in FCT_Reject, and before the scenario that collects audit data grouped on the OutputStream field.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Sample Registry

```
SetOutputStream
  {
    ModuleName = FCT_IScript
    Module
   {
     Active = True
     Source = FILE
     Scripts
   {
       SetOutputStream
   {
        FileName = ./iScriptLib/iScriptLib_RevenueAssurance/ISC_SetOutputStream.isc
     }
   }
  }
}
```

ISC_SetRevenueFigures

The ISC_SetRevenueFigures iScript collects the previous and current charged and discount amount for a configured balance element ID.

Dependencies

This iScript must run after rating and discounting and before FCT_AggreGate.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Parameters

Table 6-25 lists the ISC_SetRevenueFigures registry parameter.

Table 6-25 ISC_SetRevenueFigures Registry Parameter

Registry Parameter	Description	Mandatory
ResourceId	Specifies the balance element for which you want to calculate the discount value.	Yes

Sample Registry

```
SetRevenueFigures
{
 ModuleName = FCT_IScript
 Module
 {
  Active = True
  Source = FILE
  Scripts
  {
   SetRevenueFigures
   {
   FileName = ./iScriptLib/iScriptLib_Standard/ISC_SetRevenueFigures.isc
   ResourceId = 978
   }
  }
 }
}
```

ISC_SetRevenueStream

The ISC_SetRevenueStream iScript sets the Revenue Stream to **Retail**, **Wholesale**, **Roaming**, or **Unknown** for each EDR.

Dependencies

This iScript must be used before FCT_AggreGate and after post rating (after the EDRs are rated).

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Sample Registry

```
SetRevenueStream
{
  ModuleName = FCT_IScript
  Module
 {
  Active = True
  Source = FILE
  Scripts
  {
```

```
   SetRevenueStream
   {
    FileName = ./iScriptLib/iScriptLib_Standard/ISC_SetRevenueStream.isc
   }
  }
 }
}
```

ISC_SetSvcCodeRTZoning

For each EDR, the ISC_SetSvcCodeRTZoning iScript finds the service type and updates DETAIL.INTERN_SERVICE_CODE EDR field with the customized service code value.

Sample Registry

```
SetSvcCodeRTZoning
{
   ModuleName = FCT_IScript
   Module
   {
      Active = True
      Source = FILE
      Scripts
      {
         SetSvcCodeRTZoning
         {
            FileName = ./iScriptLib/iScriptLib_Standard/ISC_SetSvcCodeRTZoning.isc
         }
      }
   }
}
```

EDR Container Fields

Table 6-26 lists the ISC_SetSvcCodeRTZoning EDR container fields.

Table 6-26 ISC_SetSvcCodeRTZoning EDR Container Field

Alias field name Default field name	Type	Access	Description
DETAIL.INTERN_SERVICE_CODE	String	Write	Contains the service type of the current EDR.

ISC_StartTime

The ISC_StartTime iScript is used to request the start time.

Sample Registry

```
StartTime
{
  ModuleName = FCT_IScript
  Module
  {
    Active = True
    Source = FILE
```

```
    Scripts
    {
      StartTime
      {
      FileName = ./iScriptLib/iScriptLib_Standard/ISC_StartTime.isc
      }
    }
  }
}
```

EDR Container Fields

Table 6-27 lists the ISC_StartTime EDR container fields.

Table 6-27 ISC_StartTime EDR Container Field

Alias field name Default field name	Type	Access	Description
DETAIL.MILLISEC_TIME	Integer	Read	Specifies the latency time in milliseconds.

ISC_TapDetailValidation_v3_12

The ISC_TapDetailValidation_v3_12 iScript validates that the fields present in the detail record of the EDR container contain valid data.

> **Note:**
>
> The ISC_TapDetailValidation_v3_12 iScript is a deprecated module but remains in BRM for backward compatibility.

You run the ISC_TapDetailValidation_v3_12 iScript when incoming files that you receive from your roaming partner use the TAP format. When processing the incoming TAP files, BRM converts input data from TAP fields into corresponding fields of the EDR container description file.

Dependencies

Because an erroneous TAP record can be discarded before the record is split into multiple records, this module must run before the FCT_Discard and ISC_TapSplitting modules.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Sample Registry

```
ISC_TapDetailValidation_v3_12
{
  ModuleName = FCT_IScript
  Module
  {
     Active = True
     Source = File
```

```
    Scripts
    {
      TapDetailValidation_v3_12
      {
          FileName = ./iScriptLib/iScriptLib/ISC_TapDetailValidation_v3_12.isc
      }
    }
  }
}
```

EDR Container Fields

Table 6-28 lists the ISC_TapDetailValidation_v3_12 EDR container fields.

Table 6-28 ISC_TapDetailValidation_v3_12 EDR Container Fields

Alias field name Default field name	Type	Access	Description
SERVER_TYPE_OF_NUMBER DETAIL.ASS_CAMEL_EXT.SERVER_ TYPE_OF_NUMBER	Integer	Read	CAMEL Invocation Fee TD57 item. Performs number-normalization.
CHARGEABLE_QUANTITY_VALUE DETAIL.ASS_CBD.CP.CHARGEABLE_ QUANTITY_VALUE	Decimal	Read	Chargeable Units TD57 item. Indicates the number of chargeable units. The value should be equal to or greater than zero.
IMPACT_CATEGORY DETAIL.ASS_CBD.CP.IMPACT_CATEGORY	String	Read	Charge Type TD57 item. Identifies the type of charge. Possible values: • **00** Total charge for Charge Information (the invoiceable value) • **01** Airtime charge • **02** reserved • **03** Toll charge • **04** Directory assistance • **05** – **20** reserved • **21** VPMN surcharge • **69** – **99** reserved
TAX_TYPE DETAIL.ASS_CBD.TP.TAX_TYPE	String	Read	Tax Type TD57 item. Indicates the type of tax. Possible values: • **01** National • **02** Regional • **03** County • **04** Local/City
VOLUME_RECEIVED DETAIL.VOLUME_RECEIVED	Decimal	Read	Data Volume Incoming TD57 item. Identifies the number of incoming octets (bytes) within an occurrence of GPRS Service Used or Content Service Used. The value should be equal to or greater than zero.

Table 6-28 (Cont.) ISC_TapDetailValidation_v3_12 EDR Container Fields

Alias field name Default field name	Type	Access	Description
VOLUME_SENT DETAIL.VOLUME_SENT	Decimal	Read	Data Volume Outgoing TD57 item. Identifies the number of outgoing octets (bytes) within an occurrence of GPRS Service Used or Content Service Used. The value should be equal to or greater than zero.
WHOLESALE_IMPACT_CATEGORY DETAIL.WHOLESALE_IMPACT_CATEGORY	String	Read	Charge Type TD57 item. Identifies the wholesale type of charge.

ISC_TapHeaderTrailerValidation_v3_12

The ISC_TapHeaderTrailerValidation_v3_12 iScript validates that fields present in the header and trailer records of the EDR contain valid data.

> **Note:**
>
> The ISC_TapHeaderTrailerValidation_v3_12 iScript is a deprecated module but remains in BRM for backward compatibility.

You run the ISC_TapHeaderTrailerValidation_v3_12 iScript when incoming files that you receive from your roaming partner use the TAP format. When processing the incoming TAP files, BRM converts input data from TAP fields into corresponding fields of the EDR container description file.

Dependencies

Because an erroneous TAP record can be discarded before the record is split into multiple records, this module must run before the FCT_Discard and ISC_TapSplitting modules.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Sample Registry

```
ISC_TapHeaderTrailerValidation_v3_12
{
  ModuleName = FCT_IScript
  Module
  {
     Active = True
     Source = File
     Scripts
     {
       TapHeaderTrailerValidation_v3_12
       {
            FileName = ./iScriptLib/iScriptLib/ISC_TapHeaderTrailerValidation_v3_12.isc
       }
```

```
      }
    }
}
```

EDR Container Fields

Table 6-29 lists the ISC_TapHeaderTrailerValidation_v3_12 EDR container fields.

Table 6-29 ISC_TapHeaderTrailerValidation_v3_12 EDR Container Fields

Alias field name Default field name	Type	Access	Description
CHARGE_REFUND_INDICATOR DETAIL.ASS_CONT_EXT.SERVICE_USED_ LIST.CHARGE_REFUND_INDICATOR	Integer	Read	Charge Refund Indicator TD57 item. Indicates that Content Transaction is a refund. When present, changes the signs of any revenue within Content Service Used. **Value:** **1** Refund
TOTAL_CHARGE_REFUND TRAILER.TOTAL_CHARGE_VALUE_ LIST.TOTAL_CHARGE_REFUND	String	Read	Total Charge Refund TD57 item. Specifies the sum of all the charges associated with Charge Type when Charge Type represents a refund. The value must be greater than zero.
TOTAL_NUMBER_OF_RECORDS TRAILER.TOTAL_NUMBER_OF_RECORDS	Integer	Read	Specifies the total number of Basic Records in the file, excluding header and trailer. Used as a check value to determine that all records have been correctly transmitted or used.

ISC_TapSplitting

The ISC_TapSplitting iScript splits mobile originating and terminating EDRs when the CDR contains more than one basic service. ISC_TapSplitting creates a new EDR for each additional basic service.

For information about splitting MOC and MTC records, see "Generating Multiple TAP MOC and MTC Records" in *BRM Setting Up Pipeline Rating and Discounting*.

Dependencies

Must run after the following modules:

- FCT_DuplicateCheck
- FCT_Discard

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Sample Registry

```
ISC_TapSplitting
{
  ModuleName = FCT_IScript
```

```
  Module
  {
     Active = True
     Source = File
     Scripts
     {
       TapSplitting
       {
            FileName = ./iScriptLib/iScriptLib/ISC_TapSplitting.isc
       }
      }
  }
}
```

ISC_TaxCalc

The ISC_TaxCalc iScript applies a flat tax to pipeline-rated events.

Dependencies

Run this module after the FCT_MainRating module, but before the FCT_BillingRecord module.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Registry Parameters

Table 6-30 lists the ISC_TaxCalc registry parameters.

Table 6-30 ISC_TaxCalc Registry Parameters

Registry Parameter	Description	Mandatory
FlatTaxRate	Specifies the default flat tax percentage. Pipeline Manager applies this tax rate when an event does not match any criteria in the **taxcodes_map** file. For example, set **FlatTaxRate** to **5** to apply a 5% tax on the charged amount.	Yes
TaxCode	Specifies the default tax code. Pipeline Manager uses this tax code when an event does not match any criteria in the **taxcodes_map** file.	Yes
TaxCodeMapFilePath	Specifies the path to the **taxcodes_map** file. If the parameter is missing, the module retrieves tax code map data from the **/config/taxcodes_map** object.	No
TaxType	Specifies the default tax type. Pipeline Manager applies this tax type when an event does not match any criteria in the **taxcodes_map** file.	Yes

Sample Registry

```
TaxPlugin
{
  ModuleName = FCT_IScript
  Module
  {
     Active = True
     Source = File
     Scripts
     {
         Tax
            FlatTaxRate = 5
```

```
            TaxCode = FLAT
            TaxType = 16
            TaxCodeMapFilePath = BRM_home/sys/cm/taxcodes_map
            Filename = ./iScriptLib/ISC_TaxationLib/ISC_TaxCalc.isc
      }
   }
}
```

Semaphore File Entries

Table 6-31 lists the ISC_TaxCalc Semaphore file entries.

Table 6-31 ISC_TaxCalc Semaphore File Entries

Entry	Description
TaxCodeMapReload	Reloads the tax code map data to pipeline memory.
TaxCodeMapPrintData	Prints the tax code map data to the console.
TaxCodeMapPrintData =*filename*	Prints the tax code map data into a log file, where *filename* is the name of the log file.

Sample Semaphore File Entry

```
ifw.DataPool.PortalConfigDataModule.Module.TaxCodeMapReload {}
ifw.DataPool.PortalConfigDataModule.Module.TaxCodeMapPrintData {}
ifw.DataPool.PortalConfigDataModule.Module.TaxCodeMapPrintData = taxCodeMapData.log
```

ISC_TAP_0312_Include

The ISC_TAP_0312_Include iScript copies TAP data from staging fields in the EDR container to business fields in the EDR container.

This iScript is part of the following iScripts:

- ISC_TAP_0312_InMap
- ISC_RAP_0105_InMap

Sample Registry

```
BusinessMapping
{
  ModuleName = FCT_IScript
  Module
  {
     Active = True
     Source = File
     Scripts
     {
       BusinessMapping
       {
           FileName = ./iScriptLib/iScriptLib_Roaming/ISC_TAP_0312_Include.isc
       }
     }
  }
}
```

EDR Container Fields

Table 6-32 lists the ISC_TAP_0312_Include EDR container fields.

Table 6-32 ISC_TAP_0312_Include EDR Container Fields

Alias field name Default field name	Type	Access	Description
DETAIL.ASS_LTE_EXT.REQUESTEDNUMBER	String	Read	Contains the number (TEL URI of the original destination) to which the customer requested to be connected.
DETAIL.ASS_LTE_EXT.REQUESTEDPUBLICUSERID	String	Read	Contains the public user ID (SIP URI of the original destination) to which the customer requested to be connected.

ISC_TAP_0312_InMap

The ISC_TAP_0312_InMap iScript copies TAP data from staging fields in the EDR container to business fields in the EDR container.

Registry Entries

Table 6-33 lists the ISC_TAP_0312_inMap registry entries.

Table 6-33 ISC_TAP_0312_InMap Registry Entries

Entry	Description	Mandatory
Active	Specifies if the module is active or inactive: **True** = Active **False** = Inactive	Yes
Source	Specifies the source of the iScripts: • **File** • **Database**	Yes
FileName	Specifies the location of the iScript. The default location is **./iScriptLib/iScriptLib_Roaming/ISC_TAP_0312_InMap.isc**.	Yes

Sample Registry

```
BusinessMapping
{
  ModuleName = FCT_IScript
  Module
  {
     Active = True
     Source = File
     Scripts
     {
       BusinessMapping
       {
           FileName = ./iScriptLib/iScriptLib_Roaming/ISC_TAP_0312_InMap.isc
       }
```

```
      }
   }
}
```

ISC_TAP_0312_Validations

The ISC_TAP_0312_Validations iScript validates TAP input data.

Registry Entries

Table 6-34 lists the ISC_TAP_0312_Validations registry entries.

Table 6-34 ISC_TAP_0312_Validations Registry Entries

Entry	Description	Mandatory
Active	Specifies if the module is active or inactive. **True** = Active **False** = Inactive	Yes
Source	Specifies the source of the iScripts. • **File** • **Database**	Yes
FileName	Specifies the location of the iScript. The default location is **./iScriptLib/iScriptLib_Roaming/ISC_TAP_0312_Validations.isc**.	Yes

Sample Registry

```
TAP3Validations
{
  ModuleName = FCT_IScript
  Module
  {
     Active = True
     Source = File
     Scripts
     {
       TAP3Validations
       {
           FileName = ./iScriptLib/iScriptLib_Roaming/ISC_TAP_0312_Validations.isc       }
     }
  }
}
```

ISC_UsageClassSetting

The ISC_UsageClassSetting iScript is used in the Incollect and Outcollect settlement pipelines to populate the DETAIL.USAGE_CLASS field with the value of DETAIL.CONNECT_TYPE in the EDR container, which specifies the type of call. The value of DETAIL.USAGE_CLASS is stored in the **event_dlay_sess_tlcs** table and is later used by high-usage reports to determine premium calls.

Sample Registry

```
UsageClassMap
{
   ModuleName = FCT_IScript
   Module
   {
      Active = True
      Source = File
      Scripts
      {
         UsageClassSettingIScript
         {
            FileName = ./iScriptLib/iScriptLib_Roaming/ISC_UsageClassSetting.isc
         }
      }
   }
}
```

EDR Container Fields

Table 6-35 lists the ISC_UsageClassSetting EDR container fields.

Table 6-35 ISC_UsageClassSetting EDR Container Fields

Alias field name Default field name	Type	Access	Description
DETAIL.USAGE_CLASS	String	Write	Internal usage class.
DETAIL.CONNECT_TYPE	String	Read	Contains the connect type of the call.

UpdateTapInfo_StopRapout

The UpdateTapInfo_StopRapout iScript updates the database with information on the Stop Return RAP file sent to the Visited Public Mobile Network (VPMN) operator.

Dependencies

None

Sample Registry

```
UpdateTapInfo_StopRapout
{
  ModuleName = FCT_IScript
  Module
  {
    Active = True
    Source = FILE
    Scripts
    {
      UpdateTapInfo_StopRapout
      {
        FileName = ./iScriptLib/iScriptLib_Roaming/ISC_UpdateTapInfo_StopRapout.isc
        DatabaseConnection = ifw.DataPool.Login
```

```
      }
    }
  }
}
```

UpdateTapInfo_Tapin

The UpdateTapInfo_Tapin iScript updates the information in the database on incoming TAP files for use in generating Stop Return RAP files.

Dependencies

This iScript should be configured before the FCT_Reject module, and after any iScripts that populate the mandatory fields in the header of an EDR.

For more information, see "Function Module Dependencies" in *BRM Setting Up Pipeline Pricing*.

Sample Registry

```
UpdateTapInfo_Tapin
{
  ModuleName = FCT_IScript
  Module
  {
    Active = True
    Source = FILE
    Scripts
    {
      UpdateTapInfo_Tapin
      {
        FileName = ./iScriptLib/iScriptLib_Roaming/ISC_UpdateTapInfo_Tapin.isc
        DatabaseConnection = ifw.DataPool.Login
      }
    }
  }
}
```

7

Pipeline Manager iScript Functions

Learn about the Oracle Communications Billing and Revenue Management (BRM) Pipeline Manager iScript functions.

Topics in this document:

Arithmetic Functions

Table 7-1 contains the arithmetic functions.

Table 7-1 Arithmetic Functions

Function	Description
decimalAbs	Derives an absolute value from a decimal value.
decimalToLong	Converts the integer portion of a decimal value to a Long value.
longAbs	Derives an absolute value from a Long value.
longToDecimal	Converts a Long value to a decimal value.
round	Rounds a decimal value to a specified number of decimal places.
sqrt	Calculates the square root of the input value.
trunc	Truncates a decimal value to a specified number of decimal places.

decimalAbs

This function derives an absolute value from a decimal value.

Syntax

```
Decimal decimalAbs(Decimal source);
```

Parameter

source
The decimal value from which to derive the absolute value.

Return Values

Returns the derived absolute value.

Example

```
if ( x == decimalAbs( x ) )
{
  logFormat( "x is a positive value" );
}
```

decimalToLong

This function converts the integer portion of a decimal value to a Long value.

Syntax

```
Long decimalToLong(Decimal source);
```

Parameter

source
The decimal value to convert to a Long value.

Return Values

Returns the Long value of the integer portion of the decimal value.

Example

```
Long p = decimalToLong( 3.1415 );
```

longAbs

This function derives an absolute value from a Long value.

Syntax

```
Long longAbs(Long source);
```

Parameter

source
The Long value from which to derive the absolute value.

Return Values

Returns the derived absolute value.

Example

```
if ( x == longAbs( x ) )
{
  logFormat( "x is a positive value" );
}
```

longToDecimal

This function converts a Long value to a decimal value.

Syntax

```
Decimal longToDecimal(Long value);
```

Parameter

value
The Long value to convert to a decimal value.

Return Values

Returns the converted decimal value.

Example

```
Decimal bytesPerSecond = longToDecimal( bytes ) / \
longToDecimal( seconds );
```

round

This function rounds a decimal value to a specified number of decimal places.

Syntax

```
Decimal round(Decimal value [, Long places] [, String mode]);
```

Parameters

value
The value to round.

places
The number of decimal places to achieve when rounding, also known as the number of significant digits (the default is **0**).

mode
The rounding mode, or method of rounding. Possible values:

- **ROUND_PLAIN:** (Default) If the digit following the last significant digit is 5 or greater, round up. If the digit following the last significant digit is less than 5, round down.
- **ROUND_UP:** Always round up if the digit following the last significant digit is greater than 0.
- **ROUND_DOWN:** Always round down. This is the same as truncating all digits following the last significant digit.
- **ROUND_BANKERS:** This mode rounds in one of the following ways, depending on the value of the digit following the last significant digit:
  - If it is less than 5, truncate all digits following the last significant digit.
  - If it is greater than 5, round up.
  - If it is 5, round to the nearest even digit. For example, if the precision is 2, 10.155 and 10.165 both round to 10.16 because 6 is an even number.

Return Values

Returns the value rounded to the specified decimal place.

Example

```
Decimal r = round( 3.1415, 3 ); // r now is 3.142
```

sqrt

This function calculates the square root of the input value.

Syntax

```
Decimal sqrt(Decimal value);
```

Parameter

value
The value for which to calculate the square root.

Return Values

Returns the square root of the input value.

Example

```
Decimal c = sqrt( a*a + b*b );
```

trunc

This function truncates a decimal value to a specified number of decimal places.

Syntax

```
Decimal trunc(Decimal value [, Long places]);
```

Parameters

value
The value to truncate.

places
The number of decimal places by which the value should be truncated (the default is **0**).

Return Values

Returns the value truncated to the specified decimal place.

Example

```
Decimal t = trunc( 3.1415, 3 ); // t now is 3.141
```

ASN.1 Functions

Table 7-2 contains the ASN.1 functions.

Table 7-2 ASN.1 Functions

Function	Description
asnTreeAddInteger	Adds an integer object to the current active node of the ASN tree.
asnTreeAddString	Adds a string object to the current active node of the ASN tree.
asnTreeCreate	Creates a tree in memory to hold an ASN.1 file structure.
asnTreeDelete	Deletes the last created or used ASN.1 tree.
asnTreeDeleteNodeByIndex	Deletes a node from the ASN.1 tree.
asnTreeFlush	Flushes the content of the ASN.1 tree to the output.
asnTreeGetStoredNode	Gets the active (working) node from a list of created and store
asnTreePop	Backs up one level in the ASN.1 tree hierarchy.
asnTreePushTag	Adds a new block to the current active node of the ASN.1 tree and sets this new block as an active node of the tree.
asnTreeStoreNode	Stores an index to a constructed block node in the ASN.1 tree, when the block position in the tree is fixed.

asnTreeAddInteger

This function adds an integer object to the current active node of the ASN.1 tree.

Syntax

```
Bool asnTreeAddInteger(String blockName, Long value);
```

Parameters

blockName
The name of the block to add (exact type from the block description file).

value
The integer to insert as the value.

Return Values

Returns **True** if successful; otherwise, returns **False**.

Example

```
...
asnTreeAddInteger("TAP3.DataVolume.DATA_VOLUME", 2512);
...
```

asnTreeAddString

This function adds a string object to the current active node of the ASN.1 tree.

Syntax

```
Bool asnTreeAddString(String blockName, String value)
```

Parameters

blockName
The name of the block to add. This must exactly match the type from the block description file.

value
The string to insert as the value.

Return Values

Returns **True** if successful; otherwise, returns **False**.

Example

```
...
asnTreeAddString("TAP3.CalledPlace.CALLED_PLACE", "Freephone");
...
```

asnTreeCreate

This function creates a tree in memory to hold an ASN.1 file structure, where the Length field of the objects can be calculated in the end, just before writing on the output.

Parameters

None.

Return

True on success, otherwise, **False**

Only one tree can be in use at a time.

Example

```
...
if ( asnTreeCreate() == false )
{
logFormat( "asnTreeCreate() failed.");
}
...
```

asnTreeDelete

This function deletes the last created or used ASN.1 tree.

Syntax

```
Bool asnTreeDelete();
```

Parameters

None.

Return Values

Returns **True** if successful; otherwise returns **False**.

Example

```
...
if ( asnTreeDelete() == false )
{
logFormat( "asnTreeDelete() failed.");
}
...
```

asnTreeDeleteNodeByIndex

This function deletes a node from the ASN.1 tree, by recursively deleting all contained blocks and values.

Syntax

```
Bool asnTreeDeleteNodeByIndex(Long nodeIdx);
```

Parameter

nodeIdx
Node index in the ASN.1 tree as returned by asnTreeStoreNode().

Return Values

Returns **True** if successful; otherwise, returns **False**.

Example

```
...
//there is no need for this optional block (no data to store
//in it), so delete it
asnTreeDeleteNodeByIndex(networkInfoIdx);
...
```

asnTreeFlush

This function flushes the content of the ASN.1 tree to the output.

Syntax

```
Bool asnTreeFlush();
```

Parameters

None.

Return Values

Returns **True** if successful; otherwise, returns **False**.

Example

```
...
if ( asnTreeFlush() == false )
{
logFormat( "asnTreeFlush() failed.");
}
...
```

asnTreeGetStoredNode

This function gets the active (working) node from a list of created and stored, but not filled, constructed blocks.

Syntax

```
Bool asnTreeGetStoredNode(Long nodeIdx);
```

Parameter

nodeIdx
Node index in the ASN.1 tree as returned by asnTreeStoreNode().

Return Values

Returns **True** if successful; otherwise, returns **False**.

Example

```
...
asnTreeGetStoredNode(networkInfoIdx);
//use asnTreeAddString() and asnTreeAddInteger() to update
//the TAP3.NetworkInfo block.
...
```

asnTreePop

This function backs up one level in the ASN.1 tree hierarchy. Every asnTreePushTag(xxxx) should have an associated asnTreePop(); it is like opening and closing brackets.

Syntax

```
Bool asnTreePop();
```

Parameters

None.

Return Values

Returns **True** if successful; otherwise, returns **False**.

Example

```
...
asnTreePushTag("TAP3.AuditControlInfo");
...
asnTreePop(); //asnTreePushTag("TAP3.AuditControlInfo");
...
```

asnTreePushTag

This function adds a new block to the current active node of the ASN.1 tree and sets this new block as an active node of the tree. Use this function to create constructed ASN.1 objects, for example, **SEQUENCE** or **CHOICE**.

If the **isIndefiniteLength** parameter is set to true, the Length field of the ASN.1 object is set to 0x80 and 2 null bytes are appended to the Value field of the ASN.1 object.

Syntax

```
Bool asnTreePushTag(String blockName [, Bool isIndefiniteLength=false] );
```

Parameters

blockName
The name of the structured block to add (exact type from the block description file).

isIndefiniteLength
Flag to indicate that the generated block has to use indefinite lengths. The default is false, that is, it stores the exact size of the value field in the objects header.

Return Values

Returns **True** if successful; otherwise, returns **False**.

Example

```
...
asnTreePushTag("TAP3.AuditControlInfo");
...
```

asnTreeStoreNode

This function stores an index to a constructed block node in the ASN.1 tree, when for example, the data values that should be put in this block are unknown, but the block position in the tree is fixed.

Syntax

```
Long asnTreeStoreNode();
```

Parameters

None.

Return Values

Returns a node index that can be used with asnTreeGetStoredNode(nodeIdx) or asnTreeDeleteNodeByIndex(nodeIdx).

Example

```
...
asnTreePushTag("TAP3.NetworkInfo");
Long networkInfoIdx = asnTreeStoreNode();
//Nothing to do now, node will be updated after all //details are processed
asnTreePop(); //for asnTreePushTag("TAP3.NetworkInfo");
...
```

The following example iScript demonstrates how to create an output file in ASN.1 containing only a list of QoS requested objects (one per event data record (EDR)), with all field values set to **3**.

This is the content of the **OutGrammar.dsc** file. There should be an associated file describing the block structure that is here used, for example, **TAP3.QoSRequestedList**.

```
// The initial iScript code
iScript
{
  use EXT_AsnTree; // iScript extension to build a Tree of ASN.1 object
                   // used to fill the Length value of the ASN.1 bloc,
                   // before printing on output stream
}
// The definition of the grammar
Grammar
{
  edr_stream:
    header
    details
    trailer
  ;
  header:
    HEADER
    {
      asnTreeCreate();
      asnTreePushTag("TAP3.QoSRequestedList");
    }
  ;
  trailer:
    TRAILER
    {
      asnTreePop(); //for asnTreePushTag("TAP3.QoSRequestedList");
      asnTreeFlush();
      asnTreeDelete();
    }
  ;
  details:
    details
    DETAIL
    {
      asnTreePushTag("TAP3.QoSRequested");
      asnTreeAddInteger("TAP3.QoSDelay.QOS_DELAY", 3);
      asnTreeAddInteger("TAP3.QoSMeanThroughput.QOS_MEAN_THROUGHPUT", 3);
      asnTreeAddInteger("TAP3.QoSPeakThroughput.QOS_PEAK_THROUGHPUT", 3);
      asnTreeAddInteger("TAP3.QoSPrecedence.QOS_PRECEDENCE", 3);
      asnTreeAddInteger("TAP3.QoSReliability.QOS_RELIABILITY", 3);
      asnTreePop(); //for asnTreePushTag("TAP3.QoSRequested");
    }
  | /*EMPTY*/
```

```
  ;
}
```

Database Connection Functions

Table 7-3 contains database connection functions.

Table 7-3 Database Connection Functions

Function	Description
dbBeginTransaction	Starts a new transaction using the specified connection.
dbCloseConnection	Closes a connection to the Pipeline Manager database.
dbCloseResult	Closes a result handle after processing the result data.
dbCommitTransaction	Commits a transaction to a specific connection.
dbConnection	Establishes a connection to the Pipeline Manager database. The handle returned by this function should be used in future calls to the **dbExecute** function.
dbDataConnection	Connects the extension to a DBC_Database module.
dbError	Retrieves a description for the last error. This description is not reset after a valid call to one of the other database connection functions. Therefore, **dbError** should only be called directly after one of the other database connection functions fails.
dbExecute	Runs an SQL statement against the Pipeline Manager database.
dbNextResult	Switches the cursor to the next result for the result handle you specify.
dbNextRow	Switches the cursor to the next row in the current result.
dbRollbackTransaction	Rolls the current transaction back for the specified connection.

dbBeginTransaction

This function starts a new transaction using the specified connection.

Syntax

```
Bool dbBeginTransaction(Long conHandle);
```

Parameter

conHandle
The connection you want to use for the new transaction.

Return Values

Returns **true** if the transaction was successfully started. Returns **false** if the function fails.

Example

```
if ( dbBeginTransaction( conHandle ) == false )
{
  logFormat( "ERROR: failed to begin a new transaction: " \
  + dbError() );
}
```

dbCloseConnection

This function closes a connection to the Pipeline Manager database.

Syntax

```
Bool dbCloseConnection(Long conHandle);
```

Parameter

conHandle
The connection you want to close.

Return Values

Returns **true** if the connection was successfully closed. Returns **false** if the function fails.

Example

```
if ( dbCloseConnection( conHandle ) == false )
{
  logFormat( "ERROR: failed to close a connection: " + \
  dbError() );
}
```

dbCloseResult

This function closes a result handle after processing the result data.

Syntax

```
Bool dbCloseResult(Long resHandle);
```

Parameter

resHandle
The result handle you want to close.

Return Values

Returns **true** if the result handle was successfully closed. Returns **false** if the function fails.

Example

```
resHandle = dbExecute( "SELECT * FROM INT_SUBS_CLI" );
if ( resHandle == INVALID_RESULT )
{
  logFormat( "ERROR: dbExecute() failed: " + dbError() );
}
...

// Process the result data

...
dbCloseResult( resHandle );
```

dbCommitTransaction

This function commits a transaction to a specific connection.

Syntax

```
Bool dbCommitTransaction(Long conHandle);
```

Parameter

conHandle
The connection you want to use for the transaction.

Return Values

Returns **true** if the transaction was successfully committed to the connection. Returns **false** if the function fails.

Example

```
if ( dbCommitTransaction( conHandle ) == false )
{
  logFormat( "ERROR: failed to commit the transaction: " + dbError() );
}
```

dbConnection

This function establishes a connection to the Pipeline Manager database. The handle returned by this function should be used in future calls to the **dbExecute** function.

> **Note:**
>
> Before calling **dbConnection**, connect to DBC_Database module using **dbDataConnection**.

Syntax

```
Long dbConnection();
```

Parameters

None.

Return Values

Returns the handle for the new connection (the handle is a value greater than or equal to 0) if the function is successful. Returns **INVALID_CONNECTION** if the function fails.

Example

```
conHandle = dbConnection();
if ( conHandle == INVALID_CONNECTION )
{
  logFormat( "ERROR: dbConnection() failed: " + dbError() );
}
```

dbDataConnection

This function connects the extension to a DBC_Database module. This connection is valid for the whole extension; you cannot connect the extension to two different DBC_Database modules.

> **Note:**
>
> Before calling **dbConnection**, connect to DBC_Database module using **dbDataConnection**.

Syntax

```
Bool dbDataConnection(String dbcModule);
```

Parameter

dbcModule
The registry name for the DBC_Database module.

Return Values

Returns **true** if the extension was successfully connected to the module. Returns **false** if the function fails.

Example

```
use IXT_Db;

if ( dbDataConnection( "integrate.DataPool.Login.Module" ) == \
true )
{
  logFormat( "Connection to DBC module established" );
}
else
{
  logFormat( "ERROR: failed to establish the connection \
  to DBC module" );
}
```

dbError

This function retrieves a description for the last error. This description is not reset after a valid call to one of the other database connection functions. Therefore, **dbError** should only be called directly after one of the other database connection functions fails.

Syntax

```
String dbError();
```

Parameters

None.

Return Values

Returns a description of the error.

Example

```
resHandle = dbExecute( conHandle, "SELECT * FROM INT_SUBS_CLI" );
if ( resHandle == INVALID_RESULT )
{
  logFormat( "ERROR: dbExecute() failed: " + dbError() );
}
```

dbExecute

This function runs an SQL statement against the Pipeline Manager database. The handle this function returns should be used to access the result of the SQL statement in the **dbNextResult** and **dbNextRow** calls that follow. After processing the result data, free the handle by calling **dbCloseResult**.

Syntax

```
Long dbExecute(Long conHandle, String sqlStatement);
```

Parameters

conHandle
The connection you want to use.

sqlStatement
The SQL statement to run.

Return Values

Returns the result handle (the handle is a value greater than or equal to 0) if the function is successful. Returns **INVALID_RESULT** if the function fails.

Example

```
resHandle = dbExecute( conHandle, "SELECT * FROM INT_SUBS_CLI" );
if ( resHandle == INVALID_RESULT )
{
  logFormat( "ERROR: dbExecute() failed: " + dbError() );
}
```

dbNextResult

This function switches the cursor to the next result for the result handle you specify.

> **Note:**
>
> This function is specific to results, not rows. The return generated by **dbExecute** can consist of a list of results in table form, with each result containing one or more data rows. Using **dbNextResult** moves the cursor from result to result, not from data row to data row within a result.

Syntax

```
Long dbNextResult(Long resHandle);
```

Parameter

resHandle
The result handle you want to process.

Return Values

Returns the next result in the result handle if the function is successful. Returns **NO_MORE_RESULTS** if the function reaches the last result. Returns a value less than 0 if the function fails.

Example

```
resHandle = dbExecute( conHandle, "SELECT * FROM INT_SUBS_CLI" );

// loop for all results
do
{
  // process the rows of the current result
  while ( (ret = dbNextResult( resHandle )) == NEXT_RESULT );

if ( ret != NO_MORE_RESULTS )
{
  logFormat( "ERROR: dbNextResult() failed: " + dbError() );
}
```

dbNextRow

This function switches the cursor to the next row in the current result.

> **Note:**
>
> This function is specific to rows, not results. The return generated by **dbExecute** can consist of a list of results in table form, with each result containing one or more data rows. Using **dbNextRow** moves the cursor from row to row within a result, not from result to result.

Syntax

```
Long dbNextRow(Long resHandle, ...);
```

Parameters

resHandle
The handle for the result you want to process.
A list of bound variables

Return Values

Returns the next row in the result if the function is successful. Returns **NO_MORE_ROWS** if the function reaches the last row. Returns a value less than 0 if the function fails.

Example

```
resHandle = dbExecute( conHandle, "SELECT * FROM INT_SUBS_CLI" );

// loop for all rows
while ( (rowRet = dbNextRow( resHandle, cli, validFrom validTo )) > 0 )
{
  ...
}

if ( rowRet != NO_MORE_ROWS )
{
  logFormat( "ERROR: dbNextRow() failed: " + dbError() );
}
```

dbRollbackTransaction

This function rolls the current transaction back for the specified connection.

Syntax

```
Bool dbRollbackTransaction(Long conHandle);
```

Parameter

conHandle
The connection whose transaction you want rolled back.

Return Values

Returns **true** if the rollback is successful. Returns **false** if the function fails.

Example

```
if ( dbRollbackTransaction( conHandle ) == false )
{
  logFormat( "ERROR: failed to rollback current transaction: " \
  + dbError() );
}
```

Data Normalizing Functions

Table 7-4 contains data normalizing functions.

Table 7-4 Data Normalizing Functions

Function	Description
convertCli	Normalizes wireline and wireless command-line interfaces (CLIs). Static class function: "EXT_ConvertCli::convert "
convertIPv4	Normalizes IPv4 addresses. Static class function: "EXT_ConvertIPv4::convert "
String convertIPv6	Normalizes IPv6 addresses. Static class function: "EXT_ConvertIPv6::convert "

Table 7-4 (Cont.) Data Normalizing Functions

Function	Description
convertIPv4onv6	Normalizes IPv4 over IPv6 addresses. Static class function: "EXT_ConvertIPv4onv6::convert "

convertCli

This function normalizes wireless and wireline CLIs into international format.

Syntax

```
String convertCli( String cli,
                   String modInd,
                   Long typeOfNumber,
                   String natAccessCode,
                   StringArray intAccessCode,
                   StringArray countryCode,
                   String intAccessCodeSign,
                   String natDestinCode )
```

Parameters

cli
CLI to normalize.

modInd
Modification Indicator, for example, "00".

typeOfNumber
Type Of Number, for example, 0.

natAccessCode
National Access Code, for example, "0".

intAccessCode
International Access Code, for example, "00".

countryCode
Country Code, for example, "49".

intAccessCodeSign
International Access Code Sign, for example, "+".

natDestinCode
National Destination Code, for example, "172".

Return Values

Returns a CLI in international normalized format: <iac>< cc><ndc>extension.

Example

```
...
use EXT_Converter;

String normCli;
```

```
String cli = "01721234567";

normCli = convertCli( cli, "00", 0, "0", "00", "49", "+", "172" );

// normCli now contains: 00491721234567

...
```

convertIPv4

This function normalizes IPv4 addresses.

Syntax

```
String convertIPv4( String ip );
```

Parameter

ip
The IP address to normalize.

Return Values

Returns an IP address in normalized format.

Dots (.) are skipped. Tokens are left-padded to 3 digits with zeroes.

Example

```
....
use EXT_Converter;

String normIp;
String ip = "192.168.1.253";

normIp = convertIPv4( ip );

// normIp now contains: 192168001253

...
```

String convertIPv6

This function normalizes IPv6 addresses.

Syntax

```
String convertIPv6(String ip;
```

Parameter

ip
The IP address to normalize

Return Values

Returns an IP address in normalized format.

Dots (.) are skipped. Tokens are left-padded to 4 digits with zeroes.

Example

```
....
use EXT_Converter;

String normIp;
String ip = "0:0:0:AF:E:0:1:FE";

normIp = convertIPv6( ip );

// normIp now contains: 00000000000000AF000E0000000100FE

...
```

convertIPv4onv6

This function normalizes IPv4 over IPv6 addresses. The decimal IPv4 address is converted into hexadecimal representation.

Syntax

```
String convertIPv4onv6(String ip);
```

Parameter

ip
The IP address to normalize

Return Values

Returns an IPv6 address in normalized format.

Dots (.) are skipped. Tokens are left-padded to 4 digits with zeroes.

Example

```
....
use EXT_Converter;

String normIp;
String ip = "0:0:0:0:0:0:192.168.10.1";

normIp = convertIPv4onv6( ip );

// normIp now contains: 00000000000000000000000C0A80A01

...
```

Date Functions

Table 7-5 contains date functions.

Table 7-5 Date Functions

Function	Description
dateAdd	Adds date and time values.

Table 7-5 (Cont.) Date Functions

Function	Description
dateDiff	Calculates the difference between two dates.
dateIsValid	Checks a date for validity; for example, after initialization from a string.
dateToStr	Converts a date value to a string.
strToDate	Converts a string into a date value.
sysdate	Retrieves the current system date.

dateAdd

This function manipulates date and time values.

Syntax

```
Date dateAdd(Date source [, Long years [, Long months [, days [, Long hours [, Long
mins [, Long secs]]]]]]);
```

Parameters

source
The source date for the addition.

years
The number of years to add. This parameter can be positive or negative.

months
The number of months to add. This parameter can be positive or negative.

days
The number of days to add. This parameter can be positive or negative.

hours
The number of hours to add. This parameter can be positive or negative.

mins
The number of minutes to add. This parameter can be positive or negative.

secs
The number of seconds to add. This parameter can be positive or negative.

Return Values

Returns the manipulated source date.

> **Note:**
>
> The variable source itself is not manipulated; only the result is returned.

Example

```
Date now = sysdate();
Date later = dateAdd( now, 1, 2, 0, 5 );

logStdout( "Date now is " + dateToStr(now) + "\n" );
logStdout( "In 1 year, 2 months and 5 hours it is " + dateToStr(later) + "\n" );
```

dateDiff

This function calculates the difference between two dates. The difference is returned in seconds.

Syntax

```
Long dateDiff(Date date1, Date date2);
```

Parameters

date1
The first date used for calculating the difference. This is the minuend.

date2
The second date used for calculating the difference. This is the subtrahend.

Return Values

Returns the difference between the first and second date, in seconds.

Example

```
if ( dateDiff( sysdate(), date ) < 0 )
{
  logFormat( "the date is a future date" );
}
```

dateIsValid

This function checks a date for validity; for example, after initialization from a string.

Syntax

```
Bool dateIsValid(Date date);
```

Parameter

date
The date to validate.

Return Values

Returns **true** if the date is valid. Returns **false** if the date is not valid.

Example

```
Date timeStamp = strToDate( timeString );
if ( dateIsValid( timeStamp ) == false )
{
```

```
  logFormat( timeString + " is no valid date string" );
}
```

dateToStr

This function converts a date value to a string.

Syntax

```
String dateToStr(Date date);
```

Parameters

%a
The abbreviated week day name; for example, Sun for Sunday. This is from tm::tm_wday.

%A
The full weekday name; for example, Sunday. This is from tm::tm_wday.

%b
The abbreviated month name; for example, Feb for February.

%B
The full month name; for example, February.

%c
The date and time; for example, Feb 29 14:34:56 2004. This may use all members.

%d
The day of the month; for example, 29.

%H
The hour of the 24-hour day; for example, 14.

%I
The hour of the 12-hour day; for example, 02.

%j
The day of the year starting from 001; for example, 060. This is from tm::tm_yday.

%m
The month of the year, from 01; for example, 02.

%M
The minutes after the hour; for example, 34.

%p
The AM/PM indicator, if any; for example, AM.

%S
The seconds after the minute; for example, 56.

%U
The week of the year, starting from 00; for example, 45. This is from tm::tm_yday and tm::tm_wday. The week is defined as starting on Sunday.

%w
The day of the week, with 0 for Sunday; for example, 2 for Tuesday.

%W
The week of the year, from 00; for example, 33. This is from tm::tm_yday and tm::tm_wday. In this case, the week is defined as starting on Monday.

%x
The date; for example, Feb 29 2004. This uses tm::tm_yday in some locales.

%X
The time; for example, 14:34:56.

%y
The year of the century, from 00; for example, 04 for 2004. In most cases, you should avoid this parameter; to ensure correct handling of the past century, use **%Y** instead.

%Y
The year including the century; for example, 1994.

%Z
The time zone name; for example, PST or PDT. This is from tm::tm_isdst.

Return Values

Returns the date as a string using the format defined by the function parameters if the function is successful. Returns an empty string if the date is invalid.

Example

```
dateToString("%a %d. %B %Y")
```

will result in:

```
"Mon 24. June 2002"
```

strToDate

This function converts a string into a date value. The only supported string format is *YYYYMMDDHHMMSS*.

Syntax

```
Date strToDate(String dateStr);
```

Parameters

%%
The literal **%** character.

%d
The day of the month; for example, 29. The range is 00-31.

%H
The hour of the 24-hour day; for example, 14. The range is 00-23.

%m
The month of the year, from 01; for example, 02. The range is 01-12.

%M
The minutes after the hour; for example, 34. The range is 00-59.

%S
The seconds after the minute; for example, 56. The range is 00-59.

%y
The year of the century, from 00; for example, 04 for 2004. The range is 01-99. In most cases, you should avoid this parameter.

%Y
The year including the century; for example, 1994.

Return Values

Returns a valid date if the input string is in the right format. Returns an invalid date if the format is not correct.

Example

```
edrDate(DETAIL.CHARGING_START_TIMESTAMP) = \
strToDate("24.12.2002", "%d. %m. %Y");
```

sysdate

This function retrieves the current system date.

Syntax

```
Date sysdate();
```

Parameters

None.

Return Values

Returns the current system date.

Example

```
Date now;
now = sysdate();
```

EDR Container Functions

Table 7-6 contains EDR container functions.

Table 7-6 EDR Container Functions

Function	Description
edrAddAdditionalStream	Adds additional output streams to each EDR.
edrAddDatablock	Adds a new data block to the current EDR container.
edrAddDatablockEx	Adds a new data block to the current EDR container.
edrAddError	Adds a new error to the current EDR container.

Table 7-6 (Cont.) EDR Container Functions

Function	Description
edrArrayIndex	Accesses the array index values in EDR container.
edrClearErrors	Clears the list of errors that the pipeline modules add to the EDR container.
edrConnectToken	Associates an EDR field with an input token and is identical to calling a block mapping with **edrInputMap**, except that it is accomplished using only one field. This function calls the **edrMissingInput** and **edrEmptyInput** state-setting functions, which indicate the reason for missing fields.
edrConnectTokenEx	Associates an EDR field with an input token and is identical to calling a block mapping with **edrInputMap**, except that it is accomplished using only one field. This function calls the **edrMissingInput** and **edrEmptyInput** state-setting functions, which indicate the reason for missing fields.
edrContainsAdditionalStream	Determines whether an EDR has an additional output stream with the name you pass in.
edrCurrentTokenIndex	Provides the index of the token parsed from the stream. Valid only in input grammar.
edrDate	Retrieves and sets date values in the current EDR container. This function is usually used to retrieve date values.
edrDateEx	Retrieves and sets date values in the current EDR container. This function is usually used to retrieve date values.
edrDecimal	Retrieves and sets decimal values in the current EDR container. This function is used usually to retrieve decimal values.
edrDecimalEx	Retrieves and sets decimal values in the current EDR container. This function is used usually to retrieve decimal values.
edrDelete	Deletes the current EDR container, changing the current pointer to the EDR container directly in front of the deleted EDR.
edrDeleteDatablock	Deletes a data block from the current EDR container. The function is not supported for nested transactions.
edrDeleteField	Clears the contents of a field in an EDR container. The function is not supported for nested transactions.
edrDuplicate	Duplicates the current EDR container.
edrEmptyInput	Sets the state of a field to **EDR_INPUT_EMPTY** when the field is present in the CDR but contains no value.
edrFieldConnectInfo	Retrieves the Info string associated with the token for the corresponding EDR field. By default, the Info string contains the description of the token type. The function works only when the EDR field is associated with a token through either the **edrInputMap** or **edrConnectToken** function.
edrFieldTokenBytePos	Calculates the position of the token associated with the corresponding EDR field. The function works only when the EDR field is associated with a token through either the **edrInputMap** or **edrConnectToken** function.

Table 7-6 (Cont.) EDR Container Functions

Function	Description
edrGetAdditionalStream	Gets the name of an additional EDR output stream given an array index number.
edrGetError	Retrieves the names of the attached error messages.
edrGetErrorParameters	Retrieves the parameters associated to a specified error.
edrGetErrorSeverity	Retrieves the severity for each of the associated errors.
edrGetStream	Gets the output stream for an EDR.
edrHasError	Retrieves the names of the attached error messages.
edrInputState	Retrieves the input state of an EDR field.
edrInternalState	Returns the internal state of an EDR field.
edrInternalStateEx	Returns the internal state of an EDR field.
edrIsValidDetail	Determines whether the current EDR container is a valid detail container.
edrLong	Retrieves and sets Long values in the current EDR container. This function is usually used to retrieve Long values.
edrLongEx	Retrieves and sets Long values in the current EDR container. This function is usually used to retrieve Long values.
edrMaxSeverity	Finds the maximum severity of the errors added to the current EDR container.
edrMissingInput	Sets the state of a field to **EDR_INPUT_MISSING** when the field is not present in the CDR.
edrNumDatablocks	Determines the number of data blocks of the specified type.
edrNumDatablocksEx	Determines the number of data blocks of the specified type.
edrNumErrors	Accesses the number of error messages attached to the current EDR container.
edrNumTokens	Accesses the number of tokens attached to the current EDR container.
edrRemoveAdditionalStream	Removes additional output streams from an EDR.
edrSetContentType	Sets the content type of the current EDR container.
edrSetCurrent	Sets the current EDR container.
edrSetIsValidDetail	Sets the EDR container's valid detail flag. The valid detail flag specifies whether the EDR container is to be discarded.
edrSetStream	Sets the output stream for an EDR.
edrString	Retrieves and sets string values in the current EDR container. This function is usually used to retrieve string values.
edrStringEx	Retrieves and sets string values in the current EDR container. This function is usually used to retrieve string values.

Table 7-6 (Cont.) EDR Container Functions

Function	Description
edrTokenString	Used to retrieve the content of each token, as identified by their indexes. When the index is not available, as for a function call with no argument, this function returns the complete byte string attached to the EDR. The byte string corresponds to the original input string that generated the EDR. The function works only when the EDR field is associated with a token through either the **edrInputMap** or **edrConnectToken** function.
iRulesModeOn	Enables the iRules mode.
iRulesModeOff	Disables the iRules mode.
pipelineName	Retrieves the name of the pipeline in which the script is running.
stopPipeline	Stops the pipeline from which it is called.

edrAddAdditionalStream

This function adds additional output streams to each EDR.

Each Out_GenericStream pipeline module has a default output stream for EDRs. You use this function to add additional output streams to direct the output to additional locations.

Output stream characteristics (output path, record prefix, and record suffix) are set in the registry file.

If the stream name sent in with this function already exists, **edrAddAdditionalStream** returns **true** but does not create the stream again.

Syntax

```
Bool edrAddAdditionalStream(String output_stream_name);
```

Parameter

output_stream_name
The name of the new output stream that you are adding.

Return Values

Returns **true** if the function is successful. Returns **false** for all other conditions.

Example

This iScript example adds two additional output module streams:

```
addoutmod.isc
------
function onDetailEdr
{
      if (edrAddAdditionalStream( "TElOut1" ) == true)
   {
      logStdout("Stream TelOut1 added ");
   }
      if (edrAddAdditionalStream( "TELOut2" ) == true)
   {
```

```
      logStdout("Stream TElOut2 added ");
   }
} // end onDetailEdr + end iScript ----
```

This registry fragment shows the two example iScript files, **addoutmod.isc** and **removeoutmod.isc**, defined in the FunctionPool section. These iScripts add and remove output module streams. The new iScripts are shown in **bold**.

```
   FunctionPool
   {
      Iscript
      {
         ModuleName = FCT_IScript
         Module
         {
            Active = True
            Source = FILE
            Scripts
            {
               addoutmod
               {
                  FileName = ./samples/simple/addoutmod.isc
               }
            }
         }
      }
      Iscript2
      {
         ModuleName = FCT_IScript
         Module
         {
            Active = True
            Source = FILE
            Scripts
            {
               removeoutmod
               {
                  FileName = ./samples/simple/removeoutmod.isc
               }
            }
         }
      }
```

This output registry section defines the **TELOut1** output section:

```
TELOut1
{
   ModuleName = OUT_GenericStream
   Module
   {
      Grammar = ./formatDesc/Formats/Solution42/SOL42_V430_OutGrammar.dsc
      DeleteEmptyStream = TRUE
      OutputStream
      {
         ModuleName = EXT_OutFileManager
         Module
         {
            OutputPath = ./samples/simple/data/out2
            OutputPrefix = Sol42_
            OutputSuffix = .out
```

```
            TempPrefix = tmp
            TempDataPath = ./samples/simple/data/out2
            TempDataPrefix = out.tmp.
            TempDataSuffix = .data

            Replace = TRUE
        }
    }
  }
}
```

> **Note:**
>
> To ensure output file integrity, specify a unique combination of OutputPath, OutputPrefix, and OutputSuffix values for each output stream defined in the registry.

edrAddDatablock

This function adds a new data block to the current EDR container.

Syntax

```
Bool edrAddDatablock(EdrField block [, Long idx1 [, Long idx2 ...]]);
```

Parameters

block
The name of the EDR block you want to add.

idxN
Additional index values specifying the path through the EDR tree structure.

Return Values

Returns **true** if the function is successful. Returns **false** if the function fails.

Example

```
if ( edrAddDatablock( DETAIL.ASS_CBD ) == false )
{
  logFormat( "ERROR: failed to add ASSOCIATED_CHARGE \
  datablock" );
}
```

edrAddDatablockEx

This function adds a new data block to the current EDR container.

Syntax

```
Bool edrAddDatablockEx(String block, Long indicesArray, Long numIndices);
```

Parameters

block
The name of the EDR block you want to add.

indicesArray
Array of additional index values specifying the path through the EDR tree structure.

numIndices
Number of indices.

Return Values

Returns **true** if the function is successful. Returns **false** if the function fails.

Example

```
Long indicesArray [ ];
Long numberOfIndices;
String edrFieldName;

edrFieldName = "DETAIL.ASS_CBD";
numberOfIndices = 0;

if ( edrAddDatablockEx(edrFieldName, indicesArray, numberOfIndices) == false )
{
  logFormat( "ERROR: failed to add ASSOCIATED_CHARGE \
  datablock" );
}
```

edrAddError

This function adds a new error to the current EDR container.

Syntax

```
Bool edrAddError(String error, Long severity [, String paramX...]);
```

Parameters

error
The name of the error you want to add to the EDR container.

severity
The severity of the error:

- **0** = Debug
- **1** = Normal
- **2** = Warning
- **3** = Minor error
- **4** = Major error
- **5** = Critical error

Return Values

Returns **true** if the function is successful. Returns **false** if the function fails.

Example

```
if ( edrString( DETAIL.SERVICE_CODE ) != "Tel" and \
edrString( DETAIL.SERVICE_CODE ) != "Fax" )
{
  edrAddError( "ERR_UNKNOWN_SERVICE_CODE", 3, edrString\
  ( DETAIL.SERVICE_CODE ) );
}
```

edrArrayIndex

This function accesses array index values in EDR container.

Syntax

```
Long edrArrayIndex(EdrField block [, Long idx1 [, Long idx2 ...]]);
```

Parameters

block
The array block of the EDR container whose index you want to access.

idxN
Additional index values specifying the path through the EDR tree structure.

Return Values

Returns the index of the EDR container.

Example

```
edrArrayIndex( DETAIL.ASS_TCF_AAA_DETAIL.PCM_OP_TCF_AAA_AUTHORIZE.INPUT.PIN_FLD_BALANCES,
 0, 0, 0, 0) = 1;
edrIndex =
edrArrayIndex( DETAIL.ASS_TCF_AAA_DETAIL.PCM_OP_TCF_AAA_AUTHORIZE.OUTPUT.PIN_FLD_BALANCES
, 0, 0, 0, 0);
```

edrClearErrors

This function clears the list of errors that the pipeline modules add to the EDR container.

Each pipeline module error has a name, severity level, and optional parameters that you can use for debugging or constructing an error message. The error list is a collection of all the errors that the pipeline modules have added to an EDR, the number of errors in the list, and the maximum severity of the errors. You can use the errors to reject an EDR or to instruct the pipeline module to process an EDR differently or to not process an EDR.

However, if an EDR does not have errors severe enough to be rejected or processed differently, you can use this function to remove the errors from the list. This function resets the error count to 0 and the maximum severity level to normal.

> **Note:**
>
> Before clearing the errors, analyze all the errors in the EDR to ensure they can be safely ignored.

Syntax

```
Void edrClearErrors();
```

Parameters

None.

Return Values

Returns nothing.

Example

```
function onInvalidDetailEdr
{
  if(edrNumErrors() > 0)
   {
   logStdout(" Current Edr contains" + longToStr(edrNumErrors()) +      "Errors");
   edrClearErrors();
   logStdout(" Current Edr contains" + longToStr(edrNumErrors()) +       "Errors after
clearErrors");
   }
  else
   {
   logStdout(" Current Edr contains no Errors");
   }
}
```

edrConnectToken

This function associates an EDR field with an input token and is identical to calling a block mapping with **edrInputMap**, except that it is accomplished using only one field.

This function calls the **edrMissingInput** and **edrEmptyInput** state-setting functions, which indicate the reason for missing fields.

Syntax

```
Bool edrConnectToken(EdrField field [, Long idx1 [, Long idx2 ...]], const String
tokenName);
```

Parameters

field
The name of the EDR field you want to access.

idxN
Additional index values specifying the path through the EDR tree structure.

tokenName
The name of the token field to access (stream record field).

Return Values

Returns **true** if the EDR field is successfully associated with the input token. Returns **false** if the function fails.

Example

```
Bool success = edrConnectToken(DETAIL.RECORD_TYPE, "SOL42.DETAIL.RECORD_NUMBER");
```

edrConnectTokenEx

This function associates an EDR field with an input token and is identical to calling a block mapping with **edrInputMap**, except that it is accomplished using only one field.

This function calls the **edrMissingInput** and **edrEmptyInput** state-setting functions, which indicate the reason for missing fields.

Syntax

```
Bool edrConnectTokenEx(String field, Long indicesArray, Long numIndices, String
tokenName);
```

Parameters

field
The name of the EDR field you want to access.

indicesArray
Array of additional index values specifying the path through the EDR tree structure.

numIndices
Number of indices.

tokenName
The name of the token field to access (stream record field).

Return Values

Returns **true** if the EDR field is successfully associated with the input token. Returns **false** if the function fails.

Example

```
Long indicesArray [ ];
Long numberOfIndices;
String edrFieldName;

edrFieldName = "DETAIL.RECORD_TYPE";
numberOfIndices = 0;

Bool success = edrConnectTokenEx(edrFieldName, indicesArray, numberOfIndices,
"SOL42.DETAIL.RECORD_NUMBER");
```

edrContainsAdditionalStream

This function determines whether an EDR has an additional output stream with the name you pass in. EDRs contain one default stream and any number of additional output streams.

Syntax

```
Bool edrContainsAdditionalStream(String output_stream_name);
```

Parameter

output_stream_name
The name of the output stream you want to confirm exists in the EDR.

Return Values

Returns **true** if the stream exists. Returns **false** if it does not.

Example

```
if ( edrContainsAdditionalStream( "TELOut3" ) == false )
{
logStdout( "ERROR: EDR does not contain additonal stream: TELOut1\n" );
}
```

edrCurrentTokenIndex

This function returns the index of the token parsed from the stream. It is valid only in input grammar.

Syntax

```
Long edrCurrentTokenIndex();
```

Parameters

None.

Return Values

Returns the token index if the token exists. Returns **-1** if the function fails.

Example

```
Long index = edrCurrentTokenIndex();
logStdout("Currently processing: " + edrTokenString(index0 + "\n");
```

edrDate

This function retrieves and sets date values in the current EDR container. This function is usually used to retrieve date values. When setting date values, use the function as the left-hand value in an assignment statement.

Syntax

```
Date edrDate(EdrField field [, Long idx1 [, Long idx2 ... ]]);
```

Parameters

field
The name of the EDR field you want to access.

idxN
Additional index values specifying the path through the EDR tree structure.

Return Values

Returns the date value of the EDR field if the function is successful. Returns **INVALID_DATE** if the data type for this EDR is not **Date** or if the path through the EDR tree structure is not valid.

Example

```
Date timeStamp;

timeStamp = edrDate( DETAIL.CHARGING_START_TIMESTAMP ); \
edrDate( DETAIL.CHARGING_START_TIMESTAMP ) = sysdate();
```

edrDateEx

This function retrieves and sets date values in the current EDR container. This function is usually used to retrieve date values. When setting date values, use the function as the left-hand value in an assignment statement.

Syntax

```
Date edrDateEx(String field, Long indicesArray, Long numIndices);
```

Parameters

field
The name of the EDR field you want to access.

indicesArray
Array of additional index values specifying the path through the EDR tree structure.

numIndices
Number of indices

Return Values

Returns the date value of the EDR field if the function is successful. Returns **INVALID_DATE** if the data type for this EDR is not **Date** or if the path through the EDR tree structure is not valid.

Example

```
Long indicesArray [ ];
Long numberOfIndices;
String edrFieldName;

edrFieldName = "DETAIL.CHARGING_START_TIMESTAMP";
numberOfIndices = 0;

Date timeStamp;

timeStamp = edrDateEx( edrFieldName, indicesArray, numberOfIndices); \
edrDateEx( edrField, indicesArray, numberOfIndices) = sysdate();
```

edrDecimal

This function retrieves and sets decimal values in the current EDR container. This function is used usually to retrieve decimal values. When used to set decimal values, use the function as the left-hand value in an assignment statement.

Syntax

```
Decimal edrDecimal(EdrField field [, Long idx1 [, Long idx2 ...]]);
```

Parameters

field
The name of the EDR field you want to access.

idxN
Additional index values specifying the path through the EDR tree structure.

Return Values

Returns the decimal value of the EDR field if the function is successful. Returns an invalid decimal value if the data type for this EDR is not decimal or if the path through the EDR tree structure is not valid (for example, an index number is wrong).

Example

```
Decimal oldAmount;

oldAmount = edrDecimal( DETAIL.CHARGED_AMOUNT_VALUE ); \
edrDecimal( DETAIL.CHARGED_AMOUNT_VALUE ) = oldAmount + 1.0;
```

edrDecimalEx

This function retrieves and sets decimal values in the current EDR container. This function is used usually to retrieve decimal values. When used to set decimal values, use the function as the left-hand value in an assignment statement.

Syntax

```
Decimal edrDecimalEx(String field, Long indicesArray, Long numIndices);
```

Parameters

field
The name of the EDR field you want to access.

indicesArray
Array of additional index values specifying the path through the EDR tree structure.

level
Number of indices.

Return Values

Returns the decimal value of the EDR field if the function is successful. Returns an invalid decimal value if the data type for this EDR is not decimal or if the path through the EDR tree structure is not valid (for example, an index number is wrong).

Example

```
Long indicesArray [ ];
Long numberOfIndices;
String edrFieldName;

edrFieldName = "DETAIL.CHARGED_AMOUNT_VALUE";
```

```
numberOfIndices = 0;

Decimal oldAmount;

oldAmount = edrDecimalEx(edrFieldName, indicesArray, numberOfIndices); \
edrDecimalEx(edrFieldName, indicesArray, numberOfIndices) = oldAmount + 1.0;
```

edrDelete

This function deletes the current EDR container, changing the current pointer to the EDR container directly in front of the deleted EDR.

Syntax

```
Bool edrDelete();
```

Parameters

None.

Return Values

Returns **true** if the current EDR container is deleted successfully. Returns **false** if there was no current EDR container.

Example

```
if ( edrDelete() )
{
  logStdout( "EDR container deleted" );
}
```

edrDeleteDatablock

This function deletes a data block from the current EDR container. The function is not supported for nested transactions (for example, transactions contained within transactions).

Syntax

```
Bool edrDeleteDatablock(EdrField block, Long idx1 [, Long idx2 ...]);
```

Parameters

block
The name of the data block you want to delete.

idxN
Additional index values specifying the path through the EDR tree structure.

Return Values

Returns **true** if the data block is successfully deleted. Returns **false** if the operation fails.

Example

```
if edrDeleteDatablock( DETAIL.ASS_GSMW_EXT, 0 ) == false )
{
  logStdout("Error: failed to delete datablock");
}
```

edrDeleteField

This function clears the contents of a field in an EDR container. The function is not supported for nested transactions (for example, transactions contained within transactions).

Syntax

```
Bool edrDeleteField(EdrField field, Long idx1 [, Long idx2 ...]);
```

Parameters

field
The name of the EDR field you want to delete.

idxN
Additional index values specifying the path through the EDR tree structure.

Return Values

Returns **true** if the EDR field content is successfully deleted. Returns **false** if the operation fails.

Example

```
if edrDeleteField( DETAIL.ASS_GSMW_EXT.RECORD_NUMBER ) == false )
{
  logStdout("ERROR: failed to delete field");
}
```

edrDuplicate

This function duplicates the current EDR container. The returned index is used as a parameter for the **edrSetCurrent** function to access the newly created EDR container.

Syntax

```
Long edrDuplicate();
```

Parameters

None.

Return Values

Returns the index of the duplicate EDR container (the index is greater than or equal to 0) if the function is successful. Returns a value less than 0 if the function fails.

Example

```
Long index = edrDuplicate();
if ( index < 0 )
{
  logFormat( "ERROR: duplication of edr failed" );
}
else
{
  if ( edrSetCurrent( index ) == true )
  {
    // send new edr to duplicate output
```

```
    edrSetStream( "DuplicateOutput" );
  }
}
```

edrEmptyInput

This function sets the state of a field to **EDR_INPUT_EMPTY** when the field is present in the CDR but contains no value.

Syntax

```
Bool edrEmptyInput(EdrField field, Long idx1 [, Long idx2 ... ]);
```

Parameters

field
The name of the empty EDR field.

idxN
Additional index values specifying the path through the EDR tree structure.

Return Values

Returns **true** if the function is successful. Returns **false** if the function fails.

Example

```
Bool success = edrEmptyInput(DETAIL.BASIC_SERVICE);
```

edrFieldConnectInfo

This function retrieves the Info string associated with the token for the corresponding EDR field. By default, the Info string contains the description of the token type. This is the default for ASCII object types.

The function works only when the EDR field is associated with a token through either the **edrInputMap** or **edrConnectToken** function.

Syntax

```
String edrFieldConnectInfo(EdrField field [, Long idx1 [, Long idx2 ...]]);
```

Parameters

field
The name of the EDR field you want to access.

idxN
Additional index values specifying the path through the EDR tree structure.

Return Values

Returns the Info string associated with the token for the EDR field if the function is successful. Returns an empty string if the path through the EDR tree structure is not valid.

Example

```
logStdout("This field is of type: "+ edrFieldConnectInfo\
( DETAIL.RECORD_TYPE ) +"\n" );
```

edrFieldTokenBytePos

This function calculates the position of the token associated with the corresponding EDR field. The calculation is in bytes starting from the beginning of the input file.

The function works only when the EDR field is associated with a token through either the **edrInputMap** or **edrConnectToken** function.

Syntax

```
Long edrFieldTokenBytePos(EdrField field [, Long idx1 [, Long idx2 ...]]);
```

Parameters

field
The name of the EDR field you want to access.

idxN
Additional index values specifying the path through the EDR tree structure.

Return Values

Returns the position (in bytes) of the token associated with the EDR field if the function is successful. Returns **-1** if the EDR field is not associated with a token.

Example

```
if ( edrString( DETAIL.RECORD_TYPE ) != "020" )
{
  logStdout("Error, unexpected value at bytePosition= \
  "+ longToStr(edrFieldTokenBytePos( DETAIL.RECORD_TYPE )) + \
  "\n" );
}
```

edrGetAdditionalStream

This function gets the name of an additional EDR output stream given an array index number.

Each EDR contains a default output stream and any number of additional output streams.

Syntax

```
String edrGetAdditionalStream(Long index_number);
```

Parameter

index_number
The array index of the output stream that you need the name of.

Return Values

Returns the name of the stream if the function is successful. Returns an empty string for all other conditions.

Example

```
String streamName = edrGetAdditionalStream( 5)
```

```
if ( streamName == "" )
{
logStdout( "ERROR: no additional stream set at index: 5\n" );
}
```

edrGetError

This function retrieves the names of the attached error messages.

Syntax

```
String edrGetError(Long idx);
```

Parameter

idx
The index of the error to be retrieved.

Return Values

Returns the name of the attached error if the function is successful. Returns an empty string if the function fails.

Example

```
for ( i = 0; i < edrNumErrors(); i = i+1 )
{
  logStdout( "ERROR " + longToStr(i) + ": " + \
  edrGetError(i) + "\n" );
}
```

edrGetErrorParameters

This function retrieves the parameters associated to a specified error.

Syntax

```
Long edrGetErrorParameters(Long idx, Array params);
```

Parameters

idx
The index of the error that you want to retrieve, where 0 <= *idx* < **edrNumErrors**.

params
The string array where the parameters can be stored. This is a return parameter.

Return Values

Returns the number of parameters in the array. Returns **0** if this function fails or if there are no parameters in the array.

Example

```
String paramList[];
Long paramCount;
Long Tap3MaxParamCount = 7;
long i;
for ( i = 0; i < edrNumErrors(); i = i+1 )
```

```
{
  if (edrGetError(i) == "ERR_TAP3_RET")
  {
    // get parameter list
    paramCount = edrGetErrorParameters(i, paramList);
    // check if enough parameters
    if (paramCount != Tap3MaxParamCount)
    {
      logStdout( "ERROR " + longToStr(i) + ": " + edrGetError(i) \
      + ", has missing parameters\n" );
    }
  }
}
```

edrGetErrorSeverity

This function retrieves the severity for each of the associated errors.

Syntax

```
Long edrGetErrorSeverity(Long idx);
```

Parameter

idx
The index of the error whose severity is being retrieved.

Return Values

Returns **0** if the severity of the attached error is Normal. Returns **1** if the severity of the attached error is Warning. Returns **2** if the severity of the attached error is Minor. Returns **3** if the severity of the attached error is Major. Returns **4** if the severity of the attached error is Critical. Returns **-1** if the function fails.

Example

```
for ( i = 0; i < edrNumErrors(); i = i+1 )
{
  logStdout( "ERROR " + longToStr(i) + " Severity: " + \
  longToStr(edrGetErrorSeverity(i)) + "\n" );
}
```

edrGetStream

This function gets the output stream for an EDR.

Syntax

```
String edrGetStream();
```

Parameters

None.

Return Values

Returns the name of the actual string.

Example

```
String streamName = edrGetStream();
```

edrHasError

This function retrieves the names of the attached error messages.

Syntax

```
Bool edrHasError(String error);
```

Parameter

error
The name of the error to be retrieved.

Return Values

Returns the name of the attached error if the function is successful. Returns an empty string if the function fails.

Example

```
for ( i = 0; i < edrNumErrors(); i = i+1 )
{
  logStdout( "ERROR " + longToStr(i) + ": " + \
  edrGetError(i) + "\n" );
}
```

edrInputState

This function retrieves the input state of an EDR field.

Syntax

```
Long edrInputState(EdrField field, Long idx1 [, Long idx2...]);
```

Parameters

field
The name of the EDR field for which to return the input state.

idxN
Additional index values specifying the path through the EDR tree structure.

Return Values

Returns **1** if the EDR field contains a default value that was added due to missing input data in the CDR. Returns **2** if the EDR field contains a default value that was added due to empty input data in the CDR. Returns **3** if the EDR field is not populated or contains data that came from the CDR.

Example

```
Bool boolvar;
boolvar = edrEmptyInput(DETAIL.BASIC_SERVICE);
boolvar = edrMissingInput(DETAIL.QOS_USED);
```

```
switch(edrInputState(DETAIL.BASIC_SERVICE))
{
  case EDR_INPUT_MISSING:
    logStdout("DETAIL.BASIC_SERVICE: MISSING\n");
    break;
  case EDR_INPUT_EMPTY:
    logStdout("DETAIL.BASIC_SERVICE: EMPTY\n");
    break;
  default:  // "uninteresting" values
    logStdout("DETAIL.BASIC_SERVICE: OTHER\n");
    break;
}
```

edrInternalState

This function returns the internal state of an EDR field.

Syntax

```
Long edrInternalState(EdrField field, Long idx1 [, Long idx2...]);
```

Parameters

field
The name of the EDR field for which to return the internal state.

idxN
Additional index values specifying the path through the EDR tree structure.

Return Values

Returns 0 if cleared. Returns 1 if connected. Returns 2 if initialized. Returns 3 if set. Returns 4 if restored. Returns 5 if restored asset. Returns -1 if the function fails.

Example

```
Long state = edrInternalState(DETAIL.ASS_CBD.CP.CHARGE);
```

edrInternalStateEx

This function returns the internal state of an EDR field.

Syntax

```
Long edrInternalStateEx(String field, Long indicesArray, Long numIndices);
```

Parameters

field
The name of the EDR field you want to access.

indicesArray
Array of additional index values specifying the path through the EDR tree structure.

numIndices
Number of indices.

Return Values

Returns 0 if cleared. Returns 1 if connected. Returns 2 if initialized. Returns 3 if set. Returns 4 if restored. Returns 5 if restored asset. Returns -1 if the function fails.

Example

```
Long indicesArray [ ];
Long numberOfIndices;
String edrFieldName;

edrFieldName = "DETAIL.ASS_CBD.CP.CHARGE";
indicesArray[0]=0;
indicesArray[1]=0;
numberOfIndices=2;

Long state = edrInternalStateEx(edrFieldName, indicesArray, numberOfIndices);
```

edrIsValidDetail

This function determines whether the current EDR container is a valid detail container. This helps you avoid processing of EDR containers that will be discarded.

Syntax

```
Bool edrIsValidDetail();
```

Parameter

None.

Return Values

Returns **true** if the current EDR container is a valid detail container. Returns **false** if it is not a valid detail container.

Example

```
if ( edrIsValidDetail() == true )
{
  // process the edr
}
```

edrLong

This function retrieves and sets Long values in the current EDR container. This function is usually used to retrieve Long values. When setting Long values, use the function as left-hand value in an assignment statement.

Syntax

```
Long edrLong(EdrField field [, Long idx1 [, Long idx2 ...]]);
```

Parameters

field
The name of the EDR field you want to access.

idxN
Additional index values specifying the path through the EDR tree structure.

Return Values

Returns the Long value of the EDR field if the function is successful. Returns **0** if the EDR has no Long field or if the path through the EDR tree structure is not valid.

Example

```
edrLong( DETAIL.CHARGED_TAX_RATE ) = 1600;
```

edrLongEx

This function retrieves and sets Long values in the current EDR container. This function is usually used to retrieve Long values. When setting Long values, use the function as left-hand value in an assignment statement.

Syntax

```
Long edrLongEx(String field, Long indicesArray, Long numIndices);
```

Parameters

field
The name of the EDR field you want to access.

indicesArray
Array of additional index values specifying the path through the EDR tree structure.

numIndices
Number of indices.

Return Values

Returns the Long value of the EDR field if the function is successful. Returns **0** if the EDR has no Long field or if the path through the EDR tree structure is not valid.

Example

```
Long indicesArray [ ];
Long numberOfIndices;
String edrFieldName;

edrFieldName = "DETAIL.CHARGED_TAX_RATE";
numberOfIndices=0;
edrLongEx(edrFieldName, indicesArray, numberOfIndices) = 1600;
```

edrMaxSeverity

This function finds the maximum severity of the errors added to the current EDR container.

Syntax

```
Long edrMaxSeverity();
```

Parameters

None.

Return Values

Returns the maximum severity of the errors of the EDR container if the function is successful. Returns **0** if there are no errors. Returns **-1** if the function fails.

Example

```
if ( edrMaxSeverity() == 0 )
{
  // The edr has no errors with severity > 0
}
```

edrMissingInput

This function sets the state of a field to **EDR_INPUT_MISSING** when the field is not present in the CDR.

Syntax

```
Bool edrMissingInput(EdrField field, Long idx1 [, Long idx2...]);
```

Parameters

field
The name of the missing EDR field.

idxN
Additional index values specifying the path through the EDR tree structure.

Return Values

Returns **true** if the function is successful. Returns **false** if the function fails.

Example

```
Bool success = edrMissingInput(DETAIL.QOS_USED);
```

edrNumDatablocks

This function determines the number of data blocks of the specified type.

Syntax

```
Long edrNumDatablocks(EdrField block [, Long idx1 [, Long idx2 ...]]);
```

Parameters

block
The name of the data block you want to access.

idxN
Additional index values specifying the path through the EDR tree structure.

Return Values

Returns the number of data blocks (the number is greater than or equal to 0) if the function is successful. Returns a value less than 0 if the function fails.

Example

```
for ( i = 0; i < edrNumDatablocks( DETAIL.ASS_CBD ); i = i + 1 )
{
  String recordType = edrString( DETAIL.ASS_CBD.RECORD_TYPE, i );
}
```

edrNumDatablocksEx

This function determines the number of data blocks of the specified type.

Syntax

```
Long edrNumDatablocksEx(String block, Long indicesArray, Long numIndices);
```

Parameters

block
The name of the data block you want to access.

indicesArray
Array of additional index values specifying the path through the EDR tree structure.

numIndices
Number of indices.

Return Values

Returns the number of data blocks (the number is greater than or equal to 0) if the function is successful. Returns a value less than 0 if the function fails.

Example

```
Long indicesArray [ ];
Long numberOfIndices;
String edrFieldName;

edrFieldName = "DETAIL.ASS_CBD";
numberOfIndices=0;

for ( i = 0; i < edrNumDatablocksEx(edrFieldName, indicesArray, numberOfIndices); i = i
+ 1 )
{
  String recordType = edrString( DETAIL.ASS_CBD.RECORD_TYPE, i );
}
```

edrNumErrors

This function accesses the number of error messages attached to the current EDR container.

Syntax

```
Long edrNumErrors();
```

Parameters

None.

Return Values

Returns the number of attached error messages (this number will be greater than or equal to 0) if the function is successful. Returns **-1** if the function fails.

Example

```
for ( i = 0; i < edrNumErrors(); i = i+1 )
{
  logStdout( "ERROR " + longToStr(i) + ": " + \
  edrGetError(i) + "\n" );
}
```

edrNumTokens

This function accesses the number of tokens attached to the current EDR container.

Syntax

```
Long edrNumTokens();
```

Parameters

None.

Return Values

Returns the number of attached tokens (this number will be greater than or equal to 0) if the function is successful. Returns **-1** if the function fails.

Example

```
for ( i = 0; i < edrNumTokens(); i = i+1 )
{
  logStdout( "Token " + longToStr(i) + ": " + \
  edrGetToken(i) + "\n" );
}
```

edrRemoveAdditionalStream

This function removes additional output streams from an EDR. Each EDR has a default output stream and any number of additional output streams.

> **Note:**
>
> This function will not remove the default output stream.

Syntax

```
Bool edrRemoveAdditionalStream(String output_stream_name);
```

Parameter

output_stream_name
The name of the output stream that you are removing from the EDR.

Return Values

Returns **true** if the function is successful or if the named stream does not exist. Returns **false** for all other conditions.

Example

This example shows how to use **edrRemoveAdditionalStream** to remove an output stream.

```
if ( edrRemoveAdditionalStream( "TELOut1" ) == false
{
logStdout( "ERROR: failed to remove additional stream: TELOut1\n" );
}
```

Example 7-1 Example removeoutmod.isc file

This example removes output module streams:

```
removeoutmod.isc
---------------
function onDetailEdr
   {
      if (edrRemoveAdditionalStream( "TelOut1" ) == true)
   {
      logStdout("Stream TelOut1 removed ");
   }
      if (edrRemoveAdditionalStream( "TelOut2" ) == true)
      {
         logStdout("Stream TelOut2 removed ");
      }
   } // end onDetailEdr + end iScript
```

edrSetContentType

This function sets the content type of the current EDR container.

Syntax

```
Bool edrSetContentType(Long content);
```

Parameter

content
The content type to be assigned to the EDR container:

- EDR_UNKNOW_CONT
- EDR_HEADER
- EDR_DETAIL
- EDR_TRAILER
- EDR_START
- EDR_STOP

- EDR_BEGIN
- EDR_END
- EDR_BEGIN_TRANSACTION
- EDR_END_TRANSACTION

Return Values

Returns **true** if the content type is valid. Returns **false** if the container type is not valid.

Example

```
if ( edrSetContentType( EDR_TRAILER ) == false )
{
  logFormat( "ERROR: edrSetContentType() failed" );
}
```

edrSetCurrent

This function sets the current EDR container. All EDR container functions only access the current EDR container.

Syntax

```
Bool edrSetCurrent(Long index);
```

Parameter

index
The index of the EDR container you want to set. This is the return value from **edrDuplicate**.

Return Values

Returns **true** if there is an EDR container with the specified index. Returns **false** if there is no EDR container with that index.

Example

```
Long index = edrDuplicate();
if ( index < 0 )
{
  logFormat( "ERROR: duplication of edr failed" );
}
else
{
  // Set the output stream for the old container
  edrSetStream( "OrigOutput" );

  // Set the output stream for the new container
  if ( edrSetCurrent( index ) == true )
  {
    edrSetStream( "NewOutput" );
  }
}
```

edrSetIsValidDetail

This function sets the EDR container's valid detail flag. The valid detail flag specifies whether the EDR container is to be discarded.

Syntax

```
Void edrSetIsValidDetail(Bool flag);
```

Parameter

flag
The valid detail flag for the EDR container.

Return Values

Returns nothing.

Example

```
if ( ... )
{
  // record shall be discarded
  edrSetIsValidDetail( false );
}
```

edrSetStream

This function sets the output stream for an EDR. Internally, Pipeline Manager uses stream numbers instead of stream names. For this reason, the name specified must be converted to a number. If you use a constant as the stream name, the conversion can be performed at compile time, resulting in quicker performance than using a stream name that is not a constant. The second advantage of using a constant is that the existence of the stream can be checked at compile time.

> **Caution:**
>
> Illegal stream names lead to compilation errors.

Syntax

```
Bool edrSetStream(String streamName);
```

Parameter

streamName
The name of the output stream for the EDR container.

Return Values

Returns **true** if the output stream is successfully set. Returns **false** if the output stream does not exist.

Example

```
// This is the FAST method: The stream number can be evaluated \
   at compile time.
// There is also a check if the stream exists at compile time.
if ( edrSetStream( "NationalOutput" ) == false )
{
  logFormat( "ERROR: edrSetStream() failed" );
```

```
}

// This is the SLOW method and should be avoided.
String nationalOutput = "NationalOutput"

if ( edrSetStream( nationalOutput ) == false )
{
  logFormat( "ERROR: no stream " + nationalOutput );
}
```

edrString

This function retrieves and sets string values in the current EDR container. This function is usually used to retrieve string values. When setting string values, use this function as the left-hand value in an assignment statement.

Syntax

```
String edrString(EdrField field [, Long idx1 [, Long idx2 ...]]);
```

Parameters

field
The name of the EDR field you want to access.

indicesArray
Array of additional index values specifying the path through the EDR tree structure.

numIndices
Number of indices.

Return Values

Returns the string value of the EDR field if the function is successful. Returns an empty string if the path through the EDR tree structure is not valid.

Example

```
if ( edrString( DETAIL.RECORD_TYPE) == "020" )
edrString(DETAIL.RECORD_TYPE) = "021";
```

edrStringEx

This function retrieves and sets string values in the current EDR container. This function is usually used to retrieve string values. When setting string values, use this function as the left-hand value in an assignment statement.

Syntax

```
String edrStringEx(String field, Long indicesArray, Long numIndices);
```

Parameters

field
The name of the EDR field you want to access.

indicesArray
Array of additional index values specifying the path through the EDR tree structure.

numIndices
Number of indices.

Return Values

Returns the string value of the EDR field if the function is successful. Returns an empty string if the path through the EDR tree structure is not valid.

Example

```
Long indicesArray [ ];
Long numberOfIndices;
String edrFieldName;

edrFieldName = "DETAIL.RECORD_TYPE";
numberOfIndices=0;

if ( edrStringEx(edrFieldName, indicesArray, numberOfIndices) == "020" )
edrStringEx(edrFieldName, indicesArray, numberOfIndices) = "021";
```

edrTokenString

This function retrieves the content of each token, as identified by their indexes. When the index is not available, as for a function call with no argument, this function returns the complete byte string attached to the EDR. The byte string corresponds to the original input string that generated the EDR.

The function works only when the EDR field is associated with a token through either the **edrInputMap** or **edrConnectToken** function.

Syntax

```
String edrTokenString([Long idx]);
```

Parameter

idx
The index of the token whose index you want to retrieve, where 0 <= *idx* < **edrNumTokens**.

Return Values

Returns the contents of the tokens if the function is successful. Returns an empty string if the index is invalid or there are no tokens associated with the EDR.

Example

```
logStdout( "The original (input) record corresponding to this \
EDR is \n" + edrTokenString() );
```

iRulesModeOn

This function enables the iRules mode. In the iRules mode, the init section does not consider the specified indices for an EDR field.

Syntax

```
iRulesModeOn();
```

Parameters

None.

Return Values

Returns nothing.

Example

```
INIT_SCRIPT:
function testPrint
{
iRulesModeOff();
logFormat("hyewons era hardc
-->"+edrString(DETAIL.CUST_A.PRODUCT.ERA.PA.KEY,0,0,0,1));
logFormat("hyewons era hardc
-->"+edrString(DETAIL.CUST_A.PRODUCT.ERA.PA.KEY,0,0,0,2));
iRulesModeOn();
}
```

iRulesModeOff

This function disables the iRules mode. Disabling iRules mode ensures that the INIT takes the specified indices.

Syntax

```
iRulesModeOff();
```

Parameters

None.

Return Values

Returns nothing.

Example

```
INIT_SCRIPT:
function testPrint
{
iRulesModeOff();
logFormat("hyewons era hardc
-->"+edrString(DETAIL.CUST_A.PRODUCT.ERA.PA.KEY,0,0,0,1));
logFormat("hyewons era hardc
-->"+edrString(DETAIL.CUST_A.PRODUCT.ERA.PA.KEY,0,0,0,2));
iRulesModeOn();
}
```

pipelineName

This function retrieves the name of the pipeline in which the script is running.

Syntax

```
String pipelineName();
```

Parameters

None.

Return Values

Returns the pipeline name.

Example

```
logPipeline("This script runs in pipeline " + pipelineName());
```

stopPipeline

This function stops the pipeline from which it is called. After the pipeline is stopped, the operator must restart the pipeline using the **ifw** command.

> **Note:**
>
> This function does not work within the BEGIN function because the pipeline object instantiation is not completed when the BEGIN function processing is in progress.

> **Note:**
>
> Use this function only when there is an unrecoverable error that requires operation intervention.

Syntax

```
Void stopPipeline();
```

Parameters

None.

Return Values

Returns nothing.

Example

```
if (unrecoverableError())
{
stopPipeline();
}
```

File Manipulation Functions

Table 7-7 contains file manipulation functions.

Table 7-7 File Manipulation Functions

Function	Description
fileClose	Closes a file that was opened earlier using the **fileOpen** function.
fileCopy	Copies a file.
fileDelete	Deletes a file.
fileEof	Checks to see whether the end of file has been reached.
fileFlush	Flushes the contents of the file buffer to disk.
fileIsOpen	Determines whether a file is currently open.
fileOpen	Opens a file for reading or writing. If the file is already open, the old file will be closed and the new file will be opened. The open mode is equivalent to the **fopen** C function.
fileReadLine	Reads a line from the input file. The line is read until the function encounters an end-of-line or end-of-file character or until *maxLen* is reached.
fileRename	Renames a file.
fileSeek	Sets the read/write pointer on a specific position (in bytes from the beginning of the file) in an opened file.
fileTell	Retrieves the position (measured in bytes from the start of the file) of the read/write pointer in an opened file.
fileWriteLong	Writes a Long value, as a string and not in binary mode, to the output file.
fileWriteStr	Writes a string to the output file.

fileClose

This function closes a file that was opened earlier using the **fileOpen** function.

Syntax

```
Void fileClose(File file);
```

Parameter

file
The file you want to close.

Return Values

Returns nothing.

Example

```
File out;
if ( fileOpen( out, "test.txt", "w" ) == true )
{
  fileWriteStr( out, "Hello World!" );
  fileClose( out );
}
```

fileCopy

This function copies a file.

Syntax

```
Bool fileCopy(String old, String new);
```

Parameters

old
The file name of the file to be copied.

new
The file name of the copy.

Return Values

Returns **true** when a file has been copied. Returns **false** when it has not been copied.

Example

```
if ( fileCopy(tempName, realname ) == false )
{
logStdout( "Failed to copy" + tempName + " to " + realName );
}
```

fileDelete

This function deletes a file.

Syntax

```
Void fileDelete(String file);
```

Parameter

file
The name of the file you want to delete.

Return Values

Returns **true** if the file was successfully deleted. Returns **false** if the function failed.

Example

```
if ( fileDelete( "test.txt" ) == false )
{
  logFormat( "ERROR: failed to delete 'test.txt'" );
}
```

fileEof

This function checks to see whether the end of file has been reached.

Syntax

```
Bool fileEof(File file);
```

Parameter

file
The file you want to check.

Return Values

Returns **true** if the end of the file was reached or if no file was open. Returns **false** if it does not reach the end of the file.

Example

```
while ( fileReadLine( in, line, 2048 ) == true )
{
  ...
}
if ( fileEof( in ) == false )
{
  logFormat( "ERROR: read error()" );
}
```

fileFlush

This function flushes the contents of the file buffer to disk.

Syntax

```
Bool fileFlush(File file);
```

Parameter

file
The file you want to flush.

Return Values

Returns **true** if the file was successfully flushed. Returns **false** if the function failed.

Example

```
fileWriteStr( out, "Price is " + price );
if ( fileFlush( out ) == false )
{
  logFormat( "ERROR: fileFlush() failed" );
}
```

fileIsOpen

This function determines whether a file is currently open.

Syntax

```
Bool fileIsOpen(File file);
```

Parameter

file
The name of file you want to check.

Return Values

Returns **true** if the file is open. Returns **false** if the function failed.

Example

```
if ( fileIsOpen( in ) == false )
{
  logFormat( "ERROR: file is not open" );
}
```

fileOpen

This function opens a file for reading or writing. If the file is already open, the old file will be closed and the new file will be opened. The open mode is equivalent to the **fopen** C function.

Syntax

```
Bool fileOpen(File file, String fileName, String openMode);
```

Parameters

file
The file you want to open.

fileName
The name of the file you want to open.

openMode
The string specifying the open mode. Specify this parameter as you would for the **fopen** C function. The following description of open mode is from the Linux ManPage:

- **r:** Open text file for reading. The stream is positioned at the beginning of the file.
- **r+:** Open for reading and writing. The stream is positioned at the beginning of the file.
- **w:** Truncate file to zero length or create text file for writing. The stream is positioned at the beginning of the file.
- **w+:** Open for reading and writing. The file is created if it does not exist; otherwise it is truncated. The stream is positioned at the beginning of the file.
- **a:** Open for writing. The file is created if it does not exist. The stream is positioned at the end of the file.
- **a+:** Open for reading and writing. The file is created if it does not exist. The stream is positioned at the end of the file.

Return Values

Returns **true** if the file was opened successfully. Returns **false** if the function failed.

Example

```
File out;

if ( fileOpen( out, "test.txt", "w" ) == false )
{
  logFormat( "ERROR: fileOpen() failed" );
}
```

fileReadLine

This function reads a line from the input file. The line is read until the function encounters an end-of-line or end-of-file character or until *maxLen* is reached.

Syntax

```
Bool fileReadLine(File file, String line, Long maxLen);
```

Parameters

file
The name of file you want to read.

line
The string that specifies the line to be read. This must be a left-hand value.

maxLen
The maximum length for the line.

Return Values

Returns **true** if the line is successfully read. Returns **false** if the function failed.

Example

```
File in;
String line;

if ( fileOpen( in, "test.txt", "r" ) == true )
{
  fileReadLine( in, line, 100 );
}
```

fileRename

This function renames a file. The new name can specify a different directory, but both the old and new file must be in the same file system.

Syntax

```
Bool fileRename(String old, String new);
```

Parameters

old
The old file name.

new
The new file name.

Return Values

Returns **true** if the file is successfully renamed. Returns **false** if the function failed.

Example

```
if ( fileRename( tempName, realName ) == false )
{
  logStdout( "Failed to rename " + tempName + " to " + realName );
}
```

fileSeek

This function sets the read/write pointer on a specific position (in bytes from the beginning of the file) in an opened file.

Syntax

```
Bool fileSeek(File file, Long offset);
```

Parameters

file
The file in which you want to set a read/write pointer.

offset
The position where you want to set the read/write pointer.

Return Values

Returns **true** when setting the read/write pointer in an opened file is successful. Returns **false** when it has not been successful.

Example

```
long offset = fileTell( myfile );
if ( fileSeek(myfile, offset) == false )
{
logStdout( "could not set the file read/write pointer to " + longToStr(offset) );
}
```

fileTell

This function retrieves the position (measured in bytes from the start of the file) of the read/write pointer in an opened file.

Syntax

```
Long fileTell(File file);
```

Parameter

file
The file to check.

Return Values

Returns the position of the read/write pointer when successful. Returns **(-1)** when an error occurs.

Example

```
long offset = fileTell( Myfile );
if ( offset != (-1) )
{
logStdout( "the read pointer is currently on position " + longToStr() + " to " +
realName );
}
```

fileWriteLong

This function writes a Long value to the output file. The Long value is written as a string and not in binary mode.

Syntax

```
Bool fileWriteLong(File file, Long value [, Long len [, Bool leading [, String
pad]]]);
```

Parameters

file
The file you want to write the Long value to.

value
The Long value to write.

len
The length of the output.

leading
Specifies whether to add leading or trailing characters: **true** adds leading characters, **false** adds trailing characters.

pad
The padding character to use as the first character of the string.

Return Values

Returns **true** if the Long value is successfully written. Returns **false** if the function failed.

Example

```
File out;

if ( fileOpen( out, "test.txt", "w" ) == true )
{
  fileWriteLong( out, 100, 14, true, "0" );
}
```

fileWriteStr

This function writes a string to the output file. The string is not automatically terminated by an end-of-line character.

Syntax

```
Bool fileWriteStr(File file, String string);
```

Parameters

file
The file you want to write the string to.

string
The string to write.

len
The length of the output. This parameter is optional.

leading
Specifies whether to add leading or trailing characters: **true** adds leading characters, **false** adds trailing characters.

pad
The padding character to use as the first character of the string.

Return Values

Returns **true** if the string is successfully written. Returns **false** if the function failed.

Example

```
File out;

if ( fileOpen( out, "test.txt", "w" ) == true )
{
  fileWriteStr( out, "Hello World!\n" );
}
```

Flist Manipulation Functions

Table 7-8 contains flist manipulation functions.

Table 7-8 Flist Manipulation Functions

Function	Description
fListToString	Returns the content of the current flist in string format.
fListFromString	Replaces the current flist with an flist based on the input string.
fListCount	Counts the number of elements at the top level of the current flist.
fListCreateNew	Replaces the current flist with an empty flist.
fListDate	Retrieves the date value from the current flist.
fListDecimal	Retrieves the decimal value from the current flist.

Table 7-8 (Cont.) Flist Manipulation Functions

Function	Description
fListDropElem	Removes an array from the current flist.
fListDropFld	Deletes a field from the current flist.
fListElemid	Retrieves the array element ID from the specified array field.
fListGetErrorText	Puts the field name from the flist into *string1* and the error text into *string2.*
fListLong	Retrieves the long value from the current flist.
fListNumElem	Counts the number of elements in an array in the current flist.
fListPopElem	Resets the array to the previous value.
fListPushElem	Creates and sets the array element into which other functions set field values.
fListSetDate	Sets a date field in the current flist.
fListSetDecimal	Sets a decimal field in the current flist.
fListSetLong	Sets a long value within a PIN_FLDT_INT or PIN_FLDT_EMUN field in the current flist.
fListSetPoid	Sets a POID field in the current flist.
fListSetString	Sets a string field in the current flist.
fListString	Retrieves the string value from the current flist.
opcodeExecuteInternal	Calls the opcode specified in the parameter.

fListToString

This function returns the content of the current flist in string format. The function calls PIN_FLIST_TO_STR.

Syntax

```
String fListToString();
```

Parameters

None.

Return Values

Returns the content of the current flist in string format. Returns an empty string on failure.

Example

```
logStdout(fListToString());
fListCreateNew();
```

fListFromString

This function removes the current flist and replaces it with an flist based on a string that you pass in as a parameter. The function calls PIN_STR_TO_FLIST.

Syntax

```
Bool fListFromString(const String flist_str);
```

Parameter

flist_str
The contents of the flist to be created, in string format.

Return Values

Returns **true** on success and **false** on failure.

Example

```
String flistStr =

"0    PIN_FLD_ARRAY          ARRAY [0] allocated 13, used 1" +
"1    PIN_FLD_STRING         STR [0] \"testing\"" +
"1    PIN_FLD_DECIMAL        DECIMAL [0] 0.000" +
"1    PIN_FLD_INT            INT [0] 60";

if(!fListFromString(flistStr))
{
// flist could not be parsed
}
```

fListCount

This function counts the number of elements at the top level of the current flist by calling PIN_FLIST_COUNT.

Syntax

```
Long fListCount();
```

Parameters

None.

Return Values

Returns the number of elements at the top level of the current flist. Returns **-1** on failure.

Example

```
Long resultCounts = fListCount();
```

fListCreateNew

This function removes the current flist and replaces it with an empty flist.

Syntax

```
Bool fListCreateNew();
```

Parameters

None.

Return Values

Returns **true** on success and **false** on failure.

Example

```
fListCreateNew();
```

fListDate

This function retrieves the date value from a PIN_FLDT_TSTAMP field in the current flist. If the field is stored in substructs or arrays, you must specify the path. You must include element IDs for all arrays.

Syntax

```
Date fListDate([const String path_field [, Long elem_id]] [,const String path_field2
[, Long elem_id] ... , ] const String field);
```

Parameters

path_field
A substruct or array field that is part of the path to the target field. The parameter is repeated in the case of nested fields.

elem_id
The element ID of an array.

field
The name of the field from which the date is retrieved.

Return Values

Returns the date value from the specified PIN_FLDT_TSTAMP field. Returns INVALID_DATETIME on failure.

Example

```
fListDate("PIN_FLD_RESULTS",1,"PIN_FLD_CREATED_T");
```

fListDecimal

This function retrieves the decimal value from a PIN_FLDT_DECIMAL field in the current flist. If the field is stored in substructs or arrays, you must specify the path. You must include element IDs for all arrays.

Syntax

```
Decimal fListDecimal([const String path_field [, Long elem_id]] [,const String
path_field2 [, Long elem_id] ... , ] const String field);
```

Parameters

path_field
A substruct or array field that is part of the path to the target field. The parameter is repeated in the case of nested fields.

elem_id
The element ID of an array.

field
The name of the field from which the decimal value is retrieved.

Return Values

Returns the decimal value from the specified PIN_FLDT_DECIMAL field. Returns INVALID_DECIMAL on failure.

Example

```
fListDecimal("PIN_FLD_OBJ_DESC", 0, "PIN_FLD_OBJ_ELEM", 6, "PIN_FLD_ORDER");
```

fListDropElem

This function removes an array from the current flist by calling PIN_FLIST_ELEM_DROP.

Syntax

```
Bool fListDropElem(const String array_field [,Long = 0 elem_id]);
```

Parameters

array_field
The name of the array.

elem_id
The array's element ID. The default is **0**.

Return Values

Returns **true** on success and **false** on failure.

Example

```
fListDropElem("PIN_FLD_ARGS", 2);
```

fListDropFld

This function deletes a field from the current flist by calling PIN_FLIST_FLD_DROP.

Syntax

```
Bool fListDropFld(const String field)
```

Parameter

field
The name of the field to be deleted.

Return Values

Returns **true** on success and **false** on failure.

Example

```
fListDropFld("PIN_FLD_LABEL");
```

fListElemid

This function retrieves the array element ID from the specified array field using a 0-n index in the array.

Syntax

```
Decimal fListElemid([const String path_field [, Long elemid]]
                  [,const String path_field2 [, Long elemid]
                   ... , ] const String array_field, Long index);
```

Parameters

path_field
A parent substruct or array field that is part of the path to the target array. The parameter is repeated in the case of nested arrays.

elem_id
The element ID of a parent array or substruct.

field
The name of the array from which the element ID is retrieved.

index
The 0-n index of the exact array element, the ID of which to return.

Return Values

Returns the *elem_id* value of the array element specified by 0-n index. Returns **INVALID_ARRAY** on failure.

Example

```
fListElemid("PIN_FLD_OBJ_DESC", 0, "PIN_FLD_OBJ_ELEM", 0);
```

fListGetErrorText

This function puts the field name from the flist into *string1* and the error text into *string2.* You can use the error information for logging or other purposes.

Syntax

```
Void fListGetErrorText(String string1, String string2);
```

Parameters

string1
String field into which the field name is placed.

string2
String field into which the error text is placed.

Return Values

Returns nothing.

Example

```
// Opcode failed
   String s1;
   String s2;
   fListGetErrorText(s1, s2);
```

fListLong

This function retrieves the long value from a PIN_FLDT_INT or PIN_FLDT_ENUM field in the current flist. If the field is stored in substructs or arrays, you must specify the path. You must include element IDs for all arrays.

Syntax

```
Long fListLong([const String path_field [, Long elem_id]] [,const String path_field2
[, Long elem_id] ... , ]const String field) ;
```

Parameters

path_field
A substruct or array field that is part of the path to the target field. The parameter is repeated in the case of nested fields.

elem_id
The element ID of an array.

field
The name of the field from which the long value is retrieved.

Return Values

Returns the long value from the specified PIN_FLDT_INT or PIN_FLDT_ENUM field. Returns 0 on error.

Example

```
fListLong("PIN_FLD_OBJ_DESC", 0, "PIN_FLD_OBJ_ELEM", 6, "PIN_FLD_LENGTH")
```

fListNumElem

This function counts the number of elements in a PIN_FLD_ARRAY field by calling PIN_FLIST_ELEM_COUNT. If the array is stored in substructs or other arrays, you must specify the path. You must include element IDs for all arrays.

Syntax

```
Long fListNumElem([const String path_field [, Long elem_id]] [,const String
path_field2 [, Long elem_id] ... , ] const String array_field, Long elem_id);
```

Parameters

path_field
A substruct or array field that is part of the path to the target array. The parameter is repeated in the case of nested fields.

elem_id
The element ID of an array.

array_field
The name of the array.

Return Values

Returns the number of elements in the specified array. Returns **-1** on failure.

Example

```
Long resultCounts = fListNumElem("PIN_FLD_OBJ_DESC", 0, "PIN_FLD_OBJ_ELEM", 6);
```

fListPopElem

This function resets the array to the previous value.

Syntax

```
Void fListPopElem();
```

Parameters

None.

Return Values

Returns nothing.

Example

```
fListPopElem();
```

fListPushElem

This function creates and sets the array element into which other functions set field values. The function calls PIN_FLIST_ELEM_ADD.

Syntax

```
Bool fListPushElem(const String array_field [,Long = 0 element]);
```

Parameters

array_field
The name of the array to set.

element
The array's element ID. The default is **0**.

Return Values

Returns **true** on success and **false** on failure.

Example

```
fListPushElem("PIN_FLD_ARGS", 2);
```

fListSetDate

This function sets a date field in the current flist.

Syntax

```
Bool fListSetDate(const String field, Date value);
```

Parameters

field
The name of the date field to set.

value
The value to set for the field.

Return Values

Returns **true** on success and **false** on failure.

Example

```
Date d = strToDate("20060402143600"); // Apr 2, 2006 2:36 pm
fListSetDate("PIN_FLD_EFFECTIVE_T", d);
```

fListSetDecimal

This function sets a decimal field in the current flist.

Syntax

```
Bool fListSetDecimal(const String field, Decimal value);
```

Parameters

field
The name of the decimal field to set.

value
The value to set for the field.

Return Values

Returns **true** on success and **false** on failure.

Example

```
fListSetDecimal("PIN_FLD_DECIMAL",edrDecimal(DETAIL.ASS_DATA.VALUE,1));
```

fListSetLong

This function sets a long value in a PIN_FLDT_INT or PIN_FLDT_ENUM field in the current flist.

Syntax

```
Bool fListSetLong(const String field, Long value);
```

Parameters

field
The name of the long field to set.

value
The value to set for the field.

Return Values

Returns **true** on success and **false** on failure.

Example

```
fListSetLong("PIN_FLD_INT",edrLong(DETAIL.ASS_DATA.QUANTITY, 1));
```

fListSetPoid

This function sets a POID field in the current flist.

Syntax

```
Bool fListSetPoid(String field, String poid);
```

Parameters

field
The name of the POID field to set.

poid
The POID string to be set in the field.

Return Values

Returns **true** on success and **false** on failure.

Example

```
Bool success = fListSetPoid( "PIN_FLD_POID", "0.0.0.1 /account 1099832 0" );
```

fListSetString

This function sets a string field in the current flist.

Syntax

```
Bool fListSetString(const String field, String value);
```

Parameters

field
The name of the string field to set.

value
The value to set for the field.

Return Values

Returns **true** on success and **false** on failure.

Example

```
fListSetString("PIN_FLD_USAGE_TYPE", usageClass);
```

fListString

This function retrieves the string value from a PIN_FLDT_STR or PIN_FLDT_POID field in the current flist. If the field is stored in substructs or arrays, you must specify the path. You must include element IDs for all arrays.

Syntax

```
String fListString([const String path_field [, Long elem_id]] [,const String
path_field2 [, Long elem_id] ... , ] const String field);
```

Parameters

path_field
A substruct or array field that is part of the path to the target field. The parameter is repeated in the case of nested fields.

elem_id
The element ID of an array.

field
The name of the field from which the string value is retrieved.

Return Values

Returns the string value from the specified PIN_FLDT_STR or PIN_FLDT_POID field. Returns NULL_STRING on failure.

Example

```
fListString("PIN_FLD_OBJ_DESC", 0, "PIN_FLD_OBJ_ELEM", 6, "PIN_FLD_DESCR")
```

opcodeExecuteInternal

This function calls the opcode specified in the parameter. You can call any opcode.

You use this function in iScripts that run in a real-time pipeline. The function uses the Connection Manager (CM) context information in the EDR to call the opcode through the existing connection.

See "opcodeExecute" for information about calling opcodes in batch pipelines.

Before calling **opcodeExecuteInternal**, you compose the input flist by using the flist extension functions. The input flist is stored and used internally by the opcode call.

The output flist of the opcode call is also stored internally and replaces the input flist. It can be retrieved by using the flist extension functions again.

If there is an error in the opcode call, an error buffer will be set. The error text can be retrieved with the **fListGetErrorText**function. The error text can then be logged.

Syntax

```
Bool opcodeExecuteInternal(Long opcode, Long flags);
```

Parameters

opcode
The opcode number of the opcode to be run.

flags
The opcode flag value. Flag values differ from opcode to opcode. Some opcodes do not expect a flag value. Use **0** for opcodes that do not expect a flag value.

Return Values

Returns **true** on success and **false** on failure.

Example

```
Long PCM_OP_SEARCH = 7;
...
if ( opcodeExecuteInternal(PCM_OP_SEARCH, 0) == false )
....
```

Hash and Array Functions

Table 7-9 contains hash and array functions.

Table 7-9 Hash and Array Functions

Function	Description
arrayClear	Clears an array.
arraySize	Determines the size of an array.
hashClear	Clears a hash.
hashContains	Checks to determine whether a hash-array contains a specific value.
hashKeys	Retrieves all keys used in an associative array.
hashRemove	Removes an entry from an associative array.

arrayClear

This function clears an array.

Syntax

```
Void arrayClear(Array array);
```

Parameter

array
The array you want to clear.

Return Values

Returns nothing.

Example

```
if ( arraySize( array ) > 0 )
{
  // Cleanup the array
  arrayClear( array );
}
```

arraySize

This function determines the size of an array.

Syntax

```
Long arraySize(Array array);
```

Parameter

array
The array whose size you want to determine.

Return Values

Returns the size of the array.

Example

```
for ( i = 0; i < arraySize( array ); i = i + 1 )
{
  logStdout( "array[" + longToStr(i) + "] = " + array[i] );
}
```

hashClear

This function clears a hash.

Syntax

```
Void hashClear(Hash hash);
```

Parameter

hash
The hash you want to clear.

Return Values

Returns nothing.

Example

```
// Cleanup the hash
hashClear( hash );
```

hashContains

This function checks to determine whether a hash-array contains a specific value.

Syntax

```
Void hashContains(Hash hash, String key);
```

Parameters

hash
The hash you want to search.

key
The value you want to search for.

Return Values

Returns **true** if the hash contains the value specified by *key*. Returns **false** if the hash does not contain this value.

Example

```
if ( hashContains( hash, "Hamburg" ) == true )
{
  logStdout( "The hash contains a value for 'Hamburg'" );
}
```

hashKeys

This function retrieves all keys used in an associative array.

Syntax

```
Long hashKeys(Hash hash, Array key);
```

Parameters

hash
The hash you want to search, looking for the key.

key
The string array as a return buffer for the keys.

Return Values

Returns the number of elements in the hash.

Example

```
String keys[];
Long age{};
Long i;

age{"Mary"} = 23;
age{"John"} = 18;

Long entries = hashKeys( age, keys );
for ( i = 0; i < entries; i = i+1 )
{
  logStdout( "Age of " + keys[i] + " is " + \
  longToStr( age{keys[i]} ) + "\n" );
}
```

hashRemove

This function removes an entry from an associative array.

Syntax

```
Bool hashRemove(Hash hash, String key);
```

Parameters

hash
The hash from which you want to remove the entry.

key
The entry to remove.

Return Values

Returns **true** if the element was removed successfully. Returns **false** if the function failed.

Example

```
if ( hashRemove( hash, "Hamburg" ) == true )
{
  logStdout( "The entry 'Hamburg' was removed from the hash\n" );
}
```

Mapping Functions

Table 7-10 contains mapping functions.

Table 7-10 Mapping Functions

Function	Description
longDecode	Maps Long values to other Long values.
strDecode	Maps string values to other string values.

longDecode

This function maps Long values to other Long values.

Syntax

```
Long longDecode(Long toMap, Long default [[, const Long src1, const Long dest1] ...]);
```

Parameters

toMap
The Long value to map.

default
The default return value if no valid mapping entry exists.

src1
The source value of the first mapping entry; this value must be a constant.

dest1
The destination value of the first mapping entry; this value must be a constant.

Return Values

Returns the matching destination value if the destination exists. Returns the value you specified in the *default* parameter if there is no destination.

Example

```
newRecordType = longDecode( oldRecordType, C_defaultRecordType,
C_oldDetail, C_newDetail,
C_oldHeader, C_newHeader,
C_oldTrailer, C_newTrailer );
```

strDecode

This function maps string values to other string values.

Syntax

```
String strDecode(String toMap, String default [[, const String src1, const String
dest1] ...]);
```

Parameters

toMap
The string value to map.

default
The default return value if no valid mapping entry exists.

src1
The source value of the first mapping entry; this value must be a constant.

dest1
The destination value of the first mapping entry; this value must be a constant.

Return Values

Returns the matching destination value if the destination exists. Returns the value you specified in the *default* parameter if there is no destination.

Example

```
newRecordType = strDecode( oldRecordType, C_defaultRecordType,
C_oldDetail, C_newDetail,
C_oldHeader, C_newHeader,
C_oldTrailer, C_newTrailer );
```

Opcode Calling Functions

Table 7-11 contains opcode calling functions.

Table 7-11 Opcode Calling Functions

Function	Description
opcodeExecute	Calls the specified opcode.
opcodeGetConnection	Obtains a connection from the specified connection pool.
pcmOpCatch	Calls PCM_OP, which performs the operation of the specified opcode and then returns the contents of the error buffer (**ebuf**) produced by the operation.

opcodeExecute

This function calls the opcode specified in the parameter. You can call any opcode.

You use this function to call opcodes in batch pipelines. See "opcodeExecuteInternal" for information about calling opcodes from real-time pipelines.

Before calling **opcodeExecute** the first time in an iScript, you must call **opcodeGetConnection** to get the connection from the connection pool. If the CM restarts or if the existing connection is broken, an error results. To get a new connection, add more conditional checks for **opcodeExecute** and then call **opcodeGetConnection**.

For example:

```
.....
Bool connectionOpened;
Long PCM_OP_NUMBER = 200;
function onBeginEdr
{
  connectionOpened = false;
}
function getCMConnection
{
 if (connectionOpened == false)
 {
 {String connectionPool = "ifw.DataPool.CMConnectionPool.Module";   connectionOpened =
opcodeGetConnection(connectionPool);
 }
}
function Bool callOpcode
{
   Long retryCount;
   Bool success;
   Long numberOfRetries = 10;
   String fldName;
   String errMsg;

getCMConnection();
success = opcodeExecute(PCM_OP_NUMBER, 0);
if (success == false)
{
   fListGetErrorText (fldName, errMsg);
   if (errMsg == "PIN_ERR_CONNECTION_LOST")
   {
    connectionOpened = false;
    for (retryCount = 0; ((retryCount < numberOfRetries) and (connectionOpened ==
false)); retryCount = retryCount + 1)
    {
        connectionOpened = false
```

```
        getCMConnection();
        if(connectionOpened == true)
        {
           success = opcodeExecute(PCM_OP_NUMBER,0);
         }
    if ((connectionOpened == false) and (retryCount >= numberOfRetries))
    {
     logStdout("Error executing opcode PCM_OP_GET_PIN_VIRTUAL_TIME due to lost
connection with CM\n");
    }
     if ((success == false)
      {
        logStdout("Error: "+ errMsg + "while executing opcode
PCM_OP_GET_PIN_VIRTUAL_TIME\n");
       }
      return success;
    function  onDetailEdr()
    {
      Bool success = callOpcode()
    }
.....
```

Before calling **opcodeExecute**, you compose the input flist by using the flist extension functions. The input flist is stored and used internally by the opcode call.

The output flist of the opcode call is also stored internally and replaces the input flist. It can be retrieved by using the flist extension functions again.

If there is an error in the opcode call, an error buffer will be set. The error text can be retrieved with the **fListGetErrorText** function. The error text can then be logged.

Syntax

```
Bool opcodeExecute(Long opcode, Long flags);
```

Parameters

opcode
The opcode number of the opcode to be run.

flags
The opcode flag value. Flag values differ from opcode to opcode. Some opcodes do not expect a flag value. Use **0** for opcodes that do not expect a flag value.

Return Values

Returns **true** on success and **false** on failure.

Example

```
...
Long PCM_OP_SEARCH = 7;
Bool success = opcodeExecute(PCM_OP_SEARCH, 0)
...
```

opcodeGetConnection

This function obtains a connection to the CM from the specified connection pool in a batch pipeline. You must configure a connection pool in the pipeline before using this function. See

DAT_ConnectionPool in the BRM documentation for information about configuring a connection pool.

In an iScript, you must call **opcodeGetConnection** before calling **opcodeExecute** the first time. You do not need to call **opcodeGetConnection** again for subsequent opcode calls in the same script. Adding more conditional checks ensures that **opcodeGetConnection** is not called every time a CDR is processed.

> **Note:**
>
> This function is required in iScripts used in batch pipelines only. It is not necessary in real-time pipelines.

For example:

```
........
Bool connectionOpened;
function onBeginEdr
{
connectionOpened = false;
}
function getCMConnection
{
  if(connectionOpened == false)
   {
   String connectionPool = "ifw.DataPool.CMConnectionPool.Module";
   connectionOpened = opcodeGetConnection(connectionPool);
   }
  if(connectionOpened == false)
   {
      logStdout("Unable to get connection to CM\n");
   }
}
.........
```

Syntax

```
Bool opcodeGetConnection(String connectionPool);
```

Parameter

connectionPool
The full registry name of the connection pool used for the pipeline.

Return Values

Returns **true** on success and **false** on failure.

Example

```
...
String connectionPool = "ifw.DataPool.CMConnectionPool.Module";
Bool success = opcodeGetConnection(connectionPool);
...
```

pcmOpCatch

This function calls the PCM_OP opcode, which performs the operation of the specified opcode and then returns the contents of the error buffer (**ebuf**) produced by the operation. This enables the calling iScript to resolve any errors that occur during the operation without exiting the logic the iScript is performing.

Syntax

```
pcmOpCatch(opcode, flags, in_flistp, ebufp);
```

Parameters

opcode
The name or number of the opcode whose operation PCM_OP is to perform.

> **Note:**
>
> Opcode numbers are listed in the **pcm_ops.h** file in the *BRM_home***/include** directory, where *BRM_home* is the directory in which you installed the BRM components.

flags
The flags supported by the opcode being called. See the opcode description for information on the flags it takes.

- If the opcode takes no flags, enter **0**.
- To specify multiple flags, separate the flag names with a vertical bar (|).

in_flistp
A pointer to the input flist of the opcode being called. See the individual opcode flist reference pages for the input flist specifications.

ebufp
A pointer to the error buffer that stores any errors that occur during the operation.

Return Values

This function returns nothing.

Errors are passed back to the calling iScript through the specified error buffer.

Example

```
pcmOpCatch(7, SRCH_DISTINCT, search, PINERR);
```

Pipeline System Functions

Table 7-12 contains Pipeline system functions.

Table 7-12 Pipeline System Functions

Function	Description
formatName	Determines the name of the format the script is running in.
logFormat	Writes messages to the pipeline log
logPipeline	Writes messages to the pipeline log.
msgArg	Deprecated.
msgName	Deprecated.
msgNumArgs	Deprecated.
registryNodeName	Returns the name of the registry node in which the script (iScript or input/output grammar) is running.
regString	Retrieves values from the registry.
reqSend	Sends a request to a registered object and waits for an answer (i.e., synchronous messaging).
scriptUsable	Sets the *usable* flag for the script. If the *usable* flag is set to **false** in the BEGIN function during Pipeline Manager startup, Pipeline Manager will not start to process CDRs. The **false** setting can be useful if the iScript initialization fails.
sendEvent	Sends an event to the event handler.
stopFormat	Stops the format; for example, after critical errors.

formatName

This function determines the name of the format the script is running in.

Syntax

```
String formatName();
```

Parameters

None.

Return Values

Returns the format name.

Example

```
logFormat( "This script runs in format " + formatName() );
```

logFormat

This function writes messages to the pipeline log.

> **Note:**
>
> This function is obsolete and should be replaced by the **logPipeline** function.

Syntax

```
Void logFormat(String msg);
```

Parameter

msg
The message to write to the pipeline log.

Return Values

Returns nothing.

Example

```
logFormat( "Hello World!" );
```

logPipeline

This function writes messages to the pipeline log.

Syntax

```
Void logPipeline(String msg [, Long severity]);
```

Parameters

msg
The message to write to the pipeline log.

severity
The severity of the message:

- **0** = Debug
- **1** = Normal
- **2** = Warning
- **3** = Minor error
- **4** = Major error
- **5** = Critical error

  The default is **0**.

Return Values

Returns nothing.

Example

```
logPipeline( "ERROR: critical database error occurred", 4 );
```

registryNodeName

This function returns the name of the registry node in which the script (iScript or input/output grammar) is running.

Syntax

```
String registryNodeName();
```

Parameters

None.

Return Values

Returns the name of the registry node in which the script (iScript or input/output grammar) is running.

Example

```
logFormat( "This script is located at registry: " + registryNodeName () );
//this will return the following result,
//This script is located at registry:
ifw.Pipelines.ciber25.Functions.Thread1.FunctionPool.myIScript.Module.Scripts.retrieve
```

regString

This function retrieves values from the registry.

Syntax

```
String regString(String name);
```

Parameter

name
The name of the registry entry.

Return Values

Returns the specified registry entry if it exists. Returns an empty string if there is no registry entry with that name.

Example

```
if ( regString( "IntegRate.DataPool.Customer.Module.Source" ) ==\
"FILE" )
{
  logFormat( "Customers are read from file" );
}
```

reqSend

This function sends a request to a registered object and waits for an answer (i.e., synchronous messaging).

Syntax

```
Bool reqSend(String reqDestination, String reqName, Array inParams, Array outParams);
```

Parameters

reqDestination
The registry name of the request's destination.

reqName
The name of the request.

inParams
A string array containing the input parameter expected by the destination to be able to process the request.

outParams
A string array to contain the reply to the request.

Request Names

REQ_NEWSEQUENCENUMBER
(Sequencer) Returns the new sequence number.

REQ_CC
(Pipeline) Returns the country code defined in the registry for this pipeline.

REQ_MCC
(Pipeline) Returns the mobile country code defined in the registry for this pipeline.

REQ_NAC
(Pipeline) Returns the national access code value defined in the registry for this pipeline.

REQ_IAC
(Pipeline) Returns the international access code defined in the registry for this pipeline.

REQ_IAC_SIGN
(Pipeline) Returns the international access code sign value defined in the registry for this pipeline.

REQ_NDC
(Pipeline) Returns the national destination code value defined in the registry for this pipeline.

REQ_REJECT_STREAM_NAME
(Pipeline) Returns the reject stream name defined in the registry for this pipeline.

REQ_REJECT_STREAM
(Pipeline) Returns the reject stream number defined in the registry for this pipeline.

REQ_EVENTHANDLER_NAME
(ifw) Returns the event handler name.

REQ_ERROR_FILENAME
(Input) Returns the name and path of the error file.

REQ_INPUT_FILENAME
(Input) Returns the name and path of the input file.

REQ_INPUT_TEMP_FILENAME
(Input) Returns the name and path of the temporary input file.

REQ_DONE_FILENAME
(Input) Returns the name and path of the done file.

REQ_RETURN_FILENAME
(Input) Returns the name and path of the return file.

REQ_OUTPUT_FILENAME
(Output) Returns the name and path of the output file.

REQ_OUTPUT_TEMP_FILENAME
(Output) Returns the name and path of the temporary output file.

Return Values

Returns **true** if the request has been sent and an answer received successfully. Returns **false** if sending the request has failed.

Example

```
sendArray [0] = "abcdefg.so142" ;
if ( reqSend( reg_InputStream, "REQ_ERROR_FILENAME",sendArray, receiveArray)==true )
{
String errFileName = receiveArray[0]; // the fully qualified filename (including path)
}
```

scriptUsable

This function sets the *usable* flag for the script. If the *usable* flag is set to **false** in the BEGIN function during Pipeline Manager startup, Pipeline Manager will not start to process CDRs. The **false** setting can be useful if the iScript initialization fails.

> **Note:**
>
> You can use this function only in the iScript modules and not in the input grammar.

Syntax

```
Void scriptUsable(Bool usable);
```

Parameter

usable
The flag indicating whether the script is usable.

Return Values

Returns nothing.

Example

```
function BEGIN
{
  ...
  if ( fileOpen( inFile, "data.txt", "r" ) == false )
  {
    logFormat( "failed to open data file 'data.txt'" );
    scriptUsable( false );
```

```
  }
}
```

sendEvent

This function sends an event to the event handler.

Syntax

```
Bool sendEvent(String event [, String arg1 [, String arg2 ...]]);
```

Parameters

event
The name of the event to send.

argX
A comma-delimited number of argument strings used as parameters for the event.

Return Values

Returns **true** if the event was successfully sent. Returns **false** if the function failed.

Example

```
if ( sendEvent( EVT_FILE_PROCESSED, filename ) == false )
{
  logFormat( "ERROR: sendEvent() failed" );
};
```

stopFormat

This function stops the format; for example, after critical errors.

Syntax

```
Void stopFormat();
```

Parameters

None.

Return Values

Returns nothing.

Example

```
if ( fileWriteString( out, data ) == false )
{
  logFormat( "ERROR: fileWriteString() failed" );
  stopFormat();
};
```

Static Functions

This section describes static functions.

EXT_ConvertCli::convert

This function normalizes wireless and wireline CLIs into international format.

Syntax

```
const BAS_String EXT_ConvertCli::convert(const BAS_String& cli,
                                         const BAS_String& modInd,
                                         const long typeOfNumber,
                                         const BAS_String& natAccessCode,
                                         const BAS_String& intAccessCode,
                                         const BAS_String& countryCode,
                                         const BAS_String& intAccessCodeSign,
                                         const BAS_String& natDestinCode );
```

Parameters

cli
CLI to normalize.

modInd
Modification Indicator, for example, "00".

typeOfNumber
Type Of Number, for example, 0.

natAccessCode
National Access Code, for example, "0".

intAccessCode
International Access Code, for example, "00".

countryCode
Country Code, for example, "49".

intAccessCodeSign
International Access Code Sign, for example, "+".

natDestinCode
National Destination Code, for example, "172".

Return Values

Returns a CLI in international normalized format: <iac>< cc><ndc>extension.

Example

```
...
#include "EXT_ConverterExt.hpp"
#include "EXT_CliConverter.hpp"

BAS_String normCli;
BAS_String cli = "01721234567";

normCli = EXT_ConvertCli::convert( cli, "00", 0, "0", "00", "49", "+", "172" );

// normCli now contains: 00491721234567
...
```

EXT_ConvertIPv4::convert

This function normalizes IPv4 addresses.

Syntax

```
const BAS_String EXT_ConvertIPv4::convert( const BAS_String& ip );
```

Parameter

ip
The IP address to normalize.

Return Values

Returns an IP address in normalized format.

Dots (.) are skipped. Tokens are left-padded to 3 digits with zeroes.

Example

```
....
#include "EXT_ConverterExt.hpp"
#include "EXT_CliConverter.hpp"

BAS_String normIp;
BAS_String ip = "192.168.1.253";

normIp = EXT_ConvertIPv4::convert( ip );

// normIp now contains: 192168001253

...
```

EXT_ConvertIPv6::convert

This function normalizes IPv6 addresses.

Syntax

```
const BAS_String EXT_ConvertIPv6::convert( const BAS_String& ip );
```

Parameter

ip
The IP address to normalize

Return Values

Returns an IP address in normalized format.

Dots (.) are skipped. Tokens are left-padded to 4 digits with zeroes.

Example

```
....
#include "EXT_ConverterExt.hpp"
#include "EXT_CliConverter.hpp"
```

```
BAS_String normIp;
BAS_String ip = "0:0:0:AF:E:0:1:FE";

normIp = EXT_ConvertIPv6::convert( ip );

// normIp now contains: 00000000000000AF000E0000000100FE

...
```

EXT_ConvertIPv4onv6::convert

This function normalizes IPv4 over IPv6 addresses. The decimal IPv4 address is converted into hexadecimal representation.

Syntax

```
const BAS_String EXT_ConvertIPv4onv6::convert( const BAS_String& ip );
```

Parameter

ip
The IP address to normalize.

Return Values

Returns an IPv6 address in normalized format.

Dots (.) are skipped. Tokens are left-padded to 4 digits with zeroes.

Example

```
....
#include "EXT_ConverterExt.hpp"
#include "EXT_CliConverter.hpp"

BAS_String normIp;
BAS_String ip = "0:0:0:0:0:0:192.168.10.1";

normIp = EXT_ConvertIPv4onv6::convert( ip );

// normIp now contains: 00000000000000000000000C0A80A01

...
```

Standard Functions

Table 7-13 contains standard functions.

Table 7-13 Standard Functions

Function	Description
closeClientConnection	Closes the connection to the Diameter client
currentTimeInMillis	Gets the current system time in milliseconds.
getClientState	Gets the state of a Diameter client.

Table 7-13 (Cont.) Standard Functions

Function	Description
mutexAcquire	Acquires the mutex specified by the handle (a number that identifies the mutex). When the mutex specified by the handle is already acquired by another thread, the function call is blocked unless the other thread releases the mutex by calling the **mutexRelease** function.
mutexCreate	Creates a mutex that can later be accessed by its handle.
mutexDestroy	Used to destroy a mutex that is no longer needed.
mutexRelease	Releases a mutex that has been acquired. It unblocks a functional call by another thread that has been trying to acquire the mutex using the **mutexAcquire** function.
sleep	Makes the process sleep.
startTimer	Starts the timer.
sysExecute	Runs a command line in a file.
sysGetEnv	Gets an environment variable.

closeClientConnection

This function closes the connection to the Diameter client.

Syntax

Void closeClientConnection(Socket Num);

Parameter

Num
Socket Id of the Diameter client.

Return Values

Returns nothing.

Example

```
if( (commandCode == DIA_DP_REQUEST) and (commandFlag == 0) )
{
  logPipeline("CommandCode: DIA_DP_REQUEST. Closing the connection.",0);
  closeClientConnection(edrLong(DETAIL.ASS_PROTOCOL_INFO.ASS_DIAMETER_INFO.SOCKETID,0,0));
}
```

currentTimeInMillis

This function gets the current system time in milliseconds.

You can use this function in your custom iScript to record the time when a pipeline or a pipeline module starts processing an EDR and when it finishes processing the EDR. You can then

calculate the difference between the start and end times to determine the latency of the EDR processing in a pipeline or module.

You can include the iScript at any point in a pipeline to determine the latency of an EDR processing between two points in a pipeline.

Syntax

```
Long currentTimeInMillis();
```

Parameters

None.

Return Values

Returns the current system time as a long value.

Example

This example gets the current system time and logs a message:

```
logStdout("The Time in milliseconds is = " + longToStr(currentTimeInMillis()) + "\n");
```

getClientState

This function gets the state of a Diameter client.

Syntax

Long getClientState(Socket Num);

Parameter

Num
Socket Id of the Diameter client.

Return Values

Returns one of the following state values:

- 0 = STATE_INITIAL
- 1 = STATE_OKAY
- 2 = STATE_DOWN

Example

state = getClientState(edrLong(DETAIL.ASS_PROTOCOL_INFO.ASS_DIAMETER_INFO.SOCKETID, 0,0));

mutexAcquire

This function acquires the mutex specified by the handle (a number that identifies the mutex). When the mutex specified by the handle is already acquired by another thread, the function call is blocked unless the other thread releases the mutex by calling the **mutexRelease** function.

Syntax

```
Bool mutexAcquire(Long handle);
```

Parameter

handle
The handle of the mutex to acquire.

Return Values

Returns **true** if a valid handle is used and the mutex is acquired. Returns **false** if an invalid handle is used and the mutex is not acquired.

Example

```
// enter the protected area
mutexAcquire (handle)

// protected area

//leave the protected area
mutexRelease(handle)
```

mutexCreate

This function creates a mutex that can later be accessed by its handle.

Syntax

```
Long mutexCreate();
```

Parameters

None.

Return Values

Returns a handle (**>0**) if the mutex was created successfully. Returns **<0** if the mutex was not created successfully.

Example

```
long handle; function BEGIN
{
handle = mutexCreate ( )
if (handle < 0)
{
 logStdout("Mutex creation failed\n");
}
}
```

mutexDestroy

This function destroys a mutex that is no longer needed.

Syntax

```
Bool mutexDestroy(Long handle);
```

Parameter

handle
The handle of the mutex to be destroyed.

Return Values

Returns **true** when destroying the mutex is successful. Returns **false** when destroying the mutex has not been successful.

Example

```
if ( mutexDestroy (handle) == false )
{
logStdout( "Illegal mutex handle\n");
}
```

mutexRelease

This function releases a mutex that has been acquired. It unblocks a functional call by another thread that has been trying to acquire the mutex using the **mutexAcquire** function.

Syntax

```
Bool mutexRelease(Long handle);
```

Parameter

handle
The handle of the mutex you want to release.

Return Values

Returns **true** when a valid handle was used and the mutex is released successfully. Returns **false** when the handle used is invalid and the mutex is not released.

Example

```
// enter the protected area
mutexAcquire (handle)

// protected area

// leave the protected area
mutexRelease(handle)
```

sleep

This function makes the process sleep.

Syntax

```
Void sleep(Long seconds);
```

Parameter

seconds
The number of seconds you want the process to sleep.

Return Values

Returns nothing.

Example

```
sleep (10)
```

startTimer

This function starts the timer.

Syntax

Void startTimer(Socket Num);

Parameter

Num
Socket Id of the Diameter client.

Return Values

Returns nothing.

Example

startTimer(edrLong(DETAIL.ASS_PROTOCOL_INFO.ASS_DIAMETER_INFO.SOCKETID,0,0)
);

sysExecute

This function runs a command line in a file. When you call this function in an iScript, you must configure an EventHandler in the pipeline registry file. For example:

```
EventHandler
{
  ModuleName = EVT
  Module
  {
   Events
    {
    }
   Buffer
    {
     Size = 1000
    }
  }
}
```

Syntax

```
Long sysExecute(String commandLine [String returnBuffer, Long timeToWait]);
```

Parameters

commandLine
The command line to execute. The value must be the path to an executable, followed by any arguments.

returnBuffer
A string to collect the output produced on stdout by *commandLine*. The stdin and stderr for *commandLine* will be the terminal.

timeToWait
The maximum time (in seconds) to wait for the response from the event handler. Command processing is terminated when *timeToWait* expires.

Return Values

Returns a Long value greater than 0 if the function is successful. Returns **-1** if the specified path points to a file that is either not readable or not processable.

Example

```
// list the contents of the /data/input directory

String cmdline = "/usr/bin/ls -l /data/input";
String retbuf;
Long timeToWait = 10; // 10 seconds
Long retval = sysExecute( cmdline, retbuf, timeToWait );
if ( retval != -1 )
{
  // code to process retbuf
  logStdout( retbuf );
}
```

sysGetEnv

This function specifies an environment variable you want returned.

Syntax

```
String sysGetEnv(String envVariable);
```

Parameter

envVariable
The name of the environment variable you want returned.

Return Values

Returns the specified environment variable and its settings.

Example

```
logStdout("******************-\n");
logStdout( "PATH=" + sysGetEnv( "PATH") +"\n");directory \n");
```

String Functions

Table 7-14 contains string functions.

Table 7-14 String Functions

Function	Description
decimalToStr	Converts a decimal value into a string.
decimalToStrHex	Converts a decimal value into a hexadecimal string. Use **round(***value***)** or **trunc(***value***)** to remove the decimal portion if you do not want it to be coded in hexadecimal.
longToHexStr	Converts a Long value into a hexadecimal string.
longToStr	Converts a Long value into a string.
strByteValue	Converts the first character in the input string to its byte value.
strDecode	Maps string values to other string values.
strStartsWith	Checks to see if a string ends with a special suffix.
strHexStrToStr	Converts each pair of characters in a given hexadecimal string into the equivalent single-byte ASCII character in a new string. The returned string is half the size of the original. Only ASCII values from 0 through 255 can be handled by this function. Characters from multi-byte character sets will cause unexpected results. The function fails if memory cannot be allocated for the string to be returned.
strHexToDecimal	Converts a hexadecimal string to a decimal value.
strHexToLong	Converts a hexadecimal string into a Long value.
strLength	Determines the length of a string.
strMatch	Compares a regular expression to a string, looking for a match.
strPad	Pads a string to a specific length. The padding character and the justification can be selected.
strReplace	Replaces substrings in a string.
strSearch	Searches for a substring inside another string.
strSearchRegExpr	Searches for a regular expression to a string.
strSplit	Splits a string according to a specific separator character and stores the resulting tokens in a string array.
strStartsWith	Checks to see if a string starts with a specified prefix.
strStrip	Removes special leading or trailing characters from a string.
strStrToHexStr	Converts each character in a given string into its two-character hexadecimal equivalent in a new string. The returned string is twice the size of the original. Only ASCII values from 0 through 255 can be handled by this function. Characters from multi-byte character sets cause unexpected results. The function fails if memory cannot be allocated for the string to be returned.
strSubstr	Extracts a substring from a string.
strToDate	Converts a string into a date value.
strToDecimal	Converts string values to decimal values.
strToLong	Converts a string value to a Long value.
strToLower	Converts a string to lowercase characters.
strToUpper	Converts a string to uppercase characters.

decimalToStr

This function converts a decimal value into a string.

Syntax

```
String decimalToStr(Decimal value [, Long precision]);
```

Parameters

value
The value to convert into a string.

precision
The number of digits after the decimal point.

Return Values

Returns the value as a string.

Example

```
logFormat( "Pi = " + decimalToStr(pi) );
logFormat( "Pi = " + decimalToStr(pi,2) );
```

decimalToStrHex

This function converts a decimal value into a hexadecimal string.

> **Note:**
>
> Use **round(***value***)** or **trunc(***value***)** to remove the decimal portion if you do not want it to be coded in hexadecimal. For example, use **round(0)** to omit the .000 if you want only integer values returned.

Syntax

```
String decimalToStrHex(Decimal value [, String separator [, Long precision]]);
```

Parameters

value
The decimal value to convert into a hexadecimal string. Code this in readable ASCII.

separator
The character you want to use as a decimal separator (the default is **.**).

precision
The precision of the decimal value to use when generating the hexadecimal string (the default is **0**).

Return Values

Returns the decimal value as a hexadecimal string.

Example

```
logFormat( "X = " + decimalToStr(x) + "(" + decimalToStrHex(x) + \
" hexadecimal)" );
```

longToHexStr

This function converts a Long value into a hexadecimal string.

Syntax

```
String longToHexStr(Long value);
```

Parameter

value
The Long value to convert into a hexadecimal string.

Return Values

Returns the value as a hexadecimal string.

Example

```
logFormat( "X = " + longToStr(x) + "(" + longToHexStr(x) + \
" hexadecimal)" );
```

longToStr

This function converts a Long value into a string.

Syntax

```
String longToStr(Long value);
```

Parameter

value
The Long value to convert into a string.

Return Values

Returns the value as a string.

Example

```
logFormat( "X = " + longToStr(x) );
```

strByteValue

This function converts the first character in the input string to its byte value.

Syntax

```
Long strByteValue(String string);
```

Parameter

string
The string whose first character you want to convert.

Return Values

Returns the byte value of the first character if the function is successful. Returns **0** if the string is empty.

Example

```
Long ascA = strByteValue( "A" );
logStdout( "ASCII(A) = " + longToStr( ascA ) + "\n" );
```

strDecode

This function maps string values to other string values.

Syntax

```
String strDecode(String toMap, String default [[, const String src1, const String
dest1] ...]);
```

Parameters

toMap
The string value to map.

default
The default return value if no valid mapping entry exists.

src1
The source value of the first mapping entry; this value must be a constant.

dest1
The destination value of the first mapping entry; this value must be a constant.

Return Values

Returns the matching destination value if the destination exists. Returns the value you specified in the *default* parameter if there is no destination.

Example

```
newRecordType = strDecode( oldRecordType, C_defaultRecordType,
C_oldDetail, C_newDetail,
C_oldHeader, C_newHeader,
C_oldTrailer, C_newTrailer );
```

strEndsWith

This function checks to see if a string ends with a special suffix.

Syntax

```
Bool strEndsWith(String string, String suffix);
```

Parameters

string
The string to check the suffix for.

suffix
The suffix to check.

Return Values

Returns **true** if the string ends with the specified suffix. Returns **false** if the string does not end with the suffix.

Example

```
if ( strEndsWith( filename, ".txt" ) )
{
  logFormat( "file suffix is .txt" );
}
```

strHexStrToStr

This function converts each pair of characters in a given hexadecimal string into the equivalent single-byte ASCII character in a new string. The returned string is half the size of the original. For example, if you pass the string **58595A373839** to **strHexStrToStr**, it returns the string **XYZ789**.

Only ASCII values from 0 through 255 can be handled by this function. Characters from multi-byte character sets will cause unexpected results. The function fails if memory cannot be allocated for the string to be returned.

Syntax

```
String strHexStrToStr(source);
```

Parameter

source
The hexadecimal string to convert to ASCII:

- It must have an even number of characters.
- Only numeric characters and A through F are permitted.
- It cannot be empty.

Return Values

Returns the string converted to ASCII if the function is successful.

If *source* has hexadecimal representations for embedded nulls, the returned string contains embedded nulls. The caller must interpret such strings correctly.

Example

```
String source = ""58595A373839";
String result = strHexStrToStr(source);
logStdout(result);
```

strHexToDecimal

This function converts a hexadecimal string to a decimal value.

Syntax

```
Decimal strHexToDecimal(String string [, String separator [, Long precision]]);
```

Parameters

string
The hexadecimal string (coded in readable ASCII) to convert into a decimal value.

separator
The character you want to use as decimal separator (the default is **.**).

precision
The precision of the decimal value to be generated (the default is **0**).

Return Values

Returns a decimal value when the value entered for *string* is successfully converted to a decimal value. Returns **0.0** if *string* is not a valid hexadecimal decimal/Long value and is therefore not converted to a decimal value.

Example

```
logStdout ( "1FF hex is " + decimalToStr ( strHexToDecimal ( "1FF" ) ) + " decimal\n" );
```

strHexToLong

This function converts a hexadecimal string into a Long value.

Syntax

```
Long strHexToLong(String string);
```

Parameter

string
The hexadecimal string to convert into a Long value.

Return Values

Returns the hexadecimal string as a Long value.

Example

```
logStdout( "1FF hex is " + strHexToLong( "1FF" ) + " decimal\n" );
```

strLength

This function determines the length of a string.

Syntax

```
Long strLength(String string);
```

Parameter

string
The string whose length you want to determine.

Return Values

Returns the string length in characters if the function is successful.

Example

```
if ( strLength( edrString( DETAIL.RECORD_TYPE ) ) != 3 )
{
  logFormat( "WARNING: illegal RECORD_TYPE" );
};
```

strMatch

This function compares a regular expression to a string, looking for a match.

Syntax

```
String strMatch(String string, String regExp [, Long index]);
```

Parameters

string
The string that you want to search for the regular expression.

regExp
The regular expression to match against the string.

index
The starting index for the search; the beginning of the string has an index of 0 (the default is **0**).

Return Values

Returns the matching part of the string if the function is successful. Returns **0** if the function does not find a match.

Example

```
if ( strMatch( filename, ".*\\.edr" ) != "" ) // IMPORTANT: the first \ is removed by
the compiler!!!!
{
  logFormat( filename + " is a *.edr file" );
}
```

strPad

This function pads a string to a specific length. The padding character and the justification can be selected.

> **Note:**
>
> The original string you started with will be truncated. If the original string is greater in length than the string you set up to result from applying the **String strPad** function.

Syntax

```
String strPad(String string, String padChar, Long length, Bool isLeftJustified);
```

Parameters

string
The string to pad (or truncate) to a specified length.

padChar
The pad character to use (the first of the string is used if empty).

length
The desired length of the returned string. If *length* is less than or equal to 0, an empty string is returned.

isLeftJustified
If set to **true**, it specifies that the string be left justified. If set to **false**, it specifies that the string be right justified.

Return Values

Returns the padded or truncated string.

Example

```
String resString;
resString = strPad ("hello", " ", 2, true); // -> resString = "he";
resString = strPad ("hello", " ", 2, false); // -> resString = "he";
resString = strPad ("hello", " ", 10, true); // -> resString = "hello ";
resString = strPad ("hello", " ", 10, false); // -> resString = "  hello";
resString = strPad ("hello", "0", 10, false); // -> resString = "00000hello";
resString = strPad ("hello", " ", -2, true); // -> resString = "";
```

strReplace

This function replaces substrings in a string.

Syntax

```
String strReplace(String toReplace, Long pos, Long len, String replace);
```

Parameters

toReplace
The string in which you want the substring replaced.

> **Note:**
>
> The input string in *toReplace* is not changed.

pos
The start position of the substring to replace. Positions start with 0.

len
The length of the substring to replace.

replace
The replacement string.

Return Values

Returns a string with the replacement string in the correct position. Returns an empty string if *pos* and *len* do not specify a valid substring.

Example

```
logFormat( strReplace( "Hello !", 5, 1, "World " ) );
```

strSearch

This function searches for a substring inside another string.

Syntax

```
Long strSearch(String string, String search [, Long index]);
```

Parameters

string
The string that you want to search.

search
The string that you want to search for.

index
The starting index for the search; the beginning of the string has an index of 0 (the default is **0**).

Return Values

Returns the starting index (this should be a value greater than or equal to 0) for the search within the string. Returns a value less than 0 if the function does not find the string.

Example

```
if ( strSearch( edrString( DETAIL.B_NUMBER ), "0049", 0 ) >= 0 )
{
  logFormat( "B-Number contains '0049'" );
}
```

strSearchRegExpr

This function searches for a regular expression to a string.

Syntax

```
Long strSearchRegExpr(String string, const String regExp [, Long index]);
```

Parameters

string
The string that you want to search.

regExp
The regular expression to look for in the string.

> **Note:**
>
> The strSearchRegExpr function does not support braces ({ }); for example, A{1,3}.

index
The starting index for the search; the beginning of the string has an index of 0 (the default is **0**).

Return Values

Returns the position index (this should be a value greater than or equal to 0) of the string if the function is successful. Returns a value less than 0 if the function does not find the string.

Example

```
if ( strSearchRegExpr( filename, ".*\\.doc", 0 ) >= 0 )
// IMPORTANT: the first \ is removed by the compiler
{
  logFormat( filename + " is a *.doc file" );
}
```

strSplit

This function splits a string according to a specific separator character and stores the resulting tokens in a string array.

Syntax

```
Long strSplit(Array res, String string, String sep);
```

Parameters

res
The resulting array to fill.

string
The input string to split.

sep
The separator to use for splitting. If the separator you specify is longer than one character, the function uses only the first character.

Return Values

Returns the number of elements in the resulting array.

Example

```
String ListArray[];
String ListString;
ListArray="first,second,third"
Long nbElem = strSplit( ListArray, ListString, "," );
for (Long i=0 ; i<nbElem ; i=i+1)
{
  logStdout( "Element " + ListArray[i] + "\n");
}
```

strStartsWith

This function checks to see if a string starts with a specified prefix.

Syntax

```
Bool strStartsWith(String string, String prefix);
```

Parameters

string
The string in which to check for the specified prefix.

prefix
The specified prefix being checked for in the string.

Return Values

Returns **true** if the string starts with the specified prefix. Returns **false** if the string does not start with the specified prefix.

Example

```
if ( strStartsWith( edrString( DETAIL.B_NUMBER ), "0049" ))
{
isNationalCall = true;
}
```

strStrip

This function removes special leading or trailing characters from a string.

Syntax

```
Bool strStrip(String string [, Long stripMode [, String stripChar]]);
```

Parameters

string
The string from which you want to remove leading or trailing characters.

stripMode
The strip mode:

- STRIP_LEADING
- STRIP_TRAILING
- STRIP_BOTH

The default is **STRIP_LEADING**.

stripChar
The character to be removed, which is the first or last character of the string (the default is the space character).

Return Values

Returns the stripped string.

Example

```
String test = "--------Hello-------------";
if ( strStrip( test, STRIP_BOTH, "-" ) == "Hello" )
{
  logStdout( "strStrip() works correct" );
}
```

strStrToHexStr

This function converts each character in a given string into its two-character hexadecimal equivalent in a new string. The returned string is twice the size of the original. For example, if you pass the string **XYZ789** to **strStrToHexStr**, it returns the string **58595A373839**.

Only ASCII values from 0 through 255 can be handled by this function. Characters from multi-byte character sets cause unexpected results. The function fails if memory cannot be allocated for the string to be returned.

Syntax

```
String strStrToHexStr(source);
```

Parameter

source
The ASCII string to convert to hexadecimal. It cannot be empty. Embedded nulls are permitted and handled correctly.

Return Values

Returns the string converted to hexadecimal if the function is successful.

Example

```
String source = "XYZ789";
String result = strStrToHexStr(source);
logStdout(result);
```

strSubstr

This function extracts a substring from a string.

Syntax

```
String strSubstr(String string, Long pos, Long len);
```

Parameters

string
The string from which you want to extract the substring.

pos
The start position of the substring to extract. Positions start with 0.

len
The length of the substring to extract.

Return Values

Returns the specified string if the function is successful. Returns an empty string if *pos* and *len* do not specify a valid substring.

Example

```
if ( strLength( string ) > 6 )
{
  string = strSubstr( string, 0, 6 );
}
```

strToDate

This function converts a string into a date value. The only supported string format is YYYYMMDDHHMMSS.

Syntax

```
Date strToDate(String dateStr);
```

Parameters

%%
The literal **%** character.

%d
The day of the month; for example, 29. The range is 00-31.

%H
The hour of the 24-hour day; for example, 14. The range is 00-23.

%m
The month of the year, from 01; for example, 02. The range is 01-12.

%M
The minutes after the hour; for example, 34. The range is 00-59.

%S
The seconds after the minute; for example, 56. The range is 00-59.

%y
The year of the century, from 00; for example, 04 for 2004. The range is 01-99. In most cases, you should avoid this parameter.

%Y
The year including the century; for example, 1994.

Return Values

Returns a valid date if the input string is in the right format. Returns an invalid date if the format is not correct.

Example

```
edrDate(DETAIL.CHARGING_START_TIMESTAMP) = \
strToDate("24.12.2002", "%d. %m. %Y");
```

strToDecimal

This function converts string values to decimal values.

Syntax

```
Decimal strToDecimal(String string);
```

Parameter

string
The string to convert to a decimal value.

Return Values

Returns the string converted to a decimal value if the function is successful. Returns **0** if the string is not a valid decimal value.

Example

```
x = x + strToDecimal( "13.32" );
```

strToLong

This function converts a numeric string value to a Long value. An alphanumeric string is returned as **0**.

Syntax

```
Long strToLong(String string);
```

Parameter

string
The string to convert to a Long value.

Return Values

Returns the string converted to a Long value if the function is successful. Returns **0** if the string is not a valid Long value.

Example

```
if ( strToLong( edrString(DETAIL.RECORD_TYPE) ) == 20 )
{
  // Basic detail record
}
```

strToLower

This function converts a string to lowercase characters.

Syntax

```
String strToLower(String string);
```

Parameter

string
The string to convert to lowercase characters.

Return Values

Returns the string converted to lowercase characters if the function is successful.

Example

```
if ( strToLower( "HELLO" ) == "hello" )
{
  ...
}
```

strToUpper

This function converts a string to uppercase characters.

Syntax

```
String strToUpper(String string);
```

Parameter

string
The string to convert to uppercase characters.

Return Values

Returns the string converted to uppercase characters if the function is successful.

Example

```
if ( strToUpper( "Hello" ) == "HELLO" )
{
  ...
}
```

Transaction Management Functions

Table 7-15 contains transaction management functions.

Table 7-15 Transaction Management Functions

Function	Description
edrDemandCancel	Sends a request to the Transaction Manager to cancel the current transaction.
edrDemandRollback	Sends a request to the Transaction Manager to roll back the current transaction.
edrRollbackReason	Allows the iScript module to request the reason for the rollback in the onRollback function.
tamItemType	Returns the type of an item in the currently processed transaction.
tamNumTransItems	Returns the number of items processed in the currently processed transaction.
tamStreamExtension	Used to access the extension value of each item in the current transaction.
tamStreamName	Used to access the stream name of each item in the current transaction.
tamTransId	Returns the transaction ID of the transaction currently being processed.

edrDemandCancel

This function sends a request to the Transaction Manager to cancel the current transaction.

Syntax

```
Bool edrDemandCancel();
```

Parameters

None.

Return Values

Returns **true** if the function is successful. Returns **false** if the function fails.

Example

```
if ( edrDemandCancel() == false )
{
  logStdout( "ERROR: failed to demand cancel" );
}
```

edrDemandRollback

This function sends a request to the Transaction Manager to roll back the current transaction.

Syntax

```
Bool edrDemandRollback([rollbackReason]);
```

Parameter

rollbackReason
The reason for the rollback.

Return Values

Returns **true** if the function is successful. Returns **false** if the function fails.

Example

Request for rollback success status:

```
if ( edrDemandRollback() == false )
{
  logStdout( "ERROR: failed to demand rollback" );
}
```

Request for rollback with a reason:

```
edrDemandRollback("Invalid Input file")
```

edrRollbackReason

This function allows the iScript module to request the reason for the rollback in the onRollback function.

Syntax

```
String edrRollbackReason();
```

Parameters

None.

Return Values

Returns a string indicating the reason for the rollback.

Example

```
function Bool onRollback
{
  rollbackReason = edrRollbackReason();
  logStdout( "rollback reason= " + rollbackReason + "\n");
  return true;
}
```

tamItemType

This function returns the type of an item in the currently processed transaction. These items are only accessible for the functions dealing with transactions like onCancel, onCommit, onRollback, and so forth.

Syntax

```
Long tamItemType(Long idx);
```

Parameter

idx
The index of the transaction item you want to access.

Return Values

Returns the type of the specified item:

- TAM_NORMAL
- TAM_RECYCLE
- TAM_RECYCLE_TEST

Returns a value of <0 if there is no current transaction in all other functions or the index is out of range.

Example

```
function onCancel
{
  Long i;
  for ( i=0; i<tamNumTransItems(); i=i+1 )
  {
    if ( tamItemType(i) == TAM_NORMAL )
    {
      ...
    }
  }
}
```

tamNumTransItems

This function returns the number of items processed in the currently processed transaction. The count includes only items accessible for the functions dealing with transactions like onCancel, onCommit, onRollback, and so forth.

Syntax

```
Long tamNumTransItems();
```

Parameters

None.

Return Values

Returns the number of items in the transaction currently being processed. Returns **0** if there is no current transaction in all other functions or there are no items in the current transaction.

Example

```
function onCancel
{
  Long i;
  for ( i=0; i<tamNumTransItems(); i=i+1 )
  {
    ...
  }
}
```

tamStreamExtension

This function accesses the extension value of each item in the current transaction. The index should be between 0 and **tamNumTransItems()–1**. Usually, the extension value contains the sequence number of the currently processed stream.

Syntax

```
String tamStreamExtension(Long idx);
```

Parameter

idx
The index of the transaction item you want to access.

Return Values

Returns the stream extension string if the function is successful. Returns an empty string if the function fails.

Example

```
function onCommit
{
  Long i;
  for ( i=0; i<tamNumTransItems(); i=i+1 )
  {
    logFormat( "commiting " + tamStreamName(i) + \
    " with extension " + tamStreamExtension(i) );
  }
}
```

tamStreamName

This function accesses the stream name of each item in the current transaction. The index should be between 0 and **tamNumTransItems( )–1**.

Syntax

```
String tamStreamName(Long idx);
```

Parameter

idx
The index of the transaction item you want to access.

Return Values

Returns the stream name if the function is successful. Returns an empty string if the function fails.

Example

```
function onCommit
{
  Long i;
  for ( i=0; i<tamNumTransItems(); i=i+1 )
  {
```

```
    logFormat( "committing " + tamStreamName(i) );
  }
}
```

tamTransId

This function returns the transaction ID of the transaction currently being processed. This function should only be used with functions dealing with transactions like onCancel, onCommit, onRollback, and so forth.

Syntax

```
Decimal tamTransId();
```

Parameters

None.

Return Values

Returns the current transaction ID. Returns **0.0** if there is no current transaction in the other functions.

Example

```
function onCancel
{
  Decimal transId = tamTransId();
  ...
}
```

8

Pipeline Manager Input and Output Modules

Learn about the Oracle Communications Billing and Revenue Management (BRM) Pipeline Manager input and output modules.

Topics in this document:

EXT_InEasyDB

The EXT_InEasyDB module handles pipeline input from a database. See:

- "About Getting Pipeline Input from a Database" in *BRM Setting Up Pipeline Rating and Discounting*.
- "Configuring EDR Input Processing" in *BRM Setting Up Pipeline Rating and Discounting*.

This module runs automatically when you start Pipeline Manager.

Configure this module as a submodule of the INP_GenericStream module. See "INP_GenericStream".

To configure input from files, see "EXT_InFileManager".

Dependencies

Requires a connection to the Pipeline Manager database.

Registry Entries

Table 8-1 lists the EXT_InEasyDB registry entries.

Table 8-1 EXT_InEasyDB Registry Entries

Entry	Description	Mandatory
ControlPath	Specifies the path for SQL, parameter, job and restart files. **./database/Oracle/Scripts/Suspense**	Yes
DataConnection	Specifies the connection to the Pipeline Manager database.	Yes
FieldDelimiter	Specifies the character that separates the EDR fields.	Yes
FileName	Specifies the job and restart file name prefix.	Yes
InputDirEmptyTime	Specifies the time period (in seconds), the input directory must be empty before the EVT_INPUT_DIR_EMPTY event is sent.	No
InputPrefix	Specifies the prefix of the stream/output file name.	No
InputSuffix	Specifies the flag which tells the stream to generate date and time information in the stream/output file name.	Yes
NumberOfRows	Specifies the array fetch size.	Yes
ParameterFile	Specifies the name of a file that contains iRule parameter values which can be used as placeholders in the SQL statements. They can be used in the update or startup registry.	Yes
Replace	Specifies the prefix/suffix should be replaced (**True**) or appended (**False**).	No
SqlDetail	Specifies the name of a file that contains the SQL select statement for generating an EDR detail record. The choices are: • **StdRecycleDetail.sql** - used with the standard recycling feature. Used without changes. • **RecycleDetail.sql** - used with the Suspense Manager service integration component. You need to customize this file for your implementation. For details, see "Installing and Configuring Suspense Manager" in *BRM Suspending and Recycling Pipeline EDRs*.	Yes
SqlHeader	Specifies the name of a file that contains the SQL select statement for generating an EDR header (result must be exactly one row).	No
SqlOnFailure	Specifies the name of a file that contains the SQL statement which is run if the output file is incorrect.	Yes
SqlOnSucess	Specifies the name of a file that contains the SQL statement that is run if the output file is correct.	Yes
SqlTrailer	Specifies the name of a file that contains the SQL select statement for generating an EDR trailer (result must be exactly one row).	No

Sample Registry

```
Stream
{
   ControlPath = ./database/Oracle/Scripts/Suspense
   DataConnection = IntegRate.DataPool.Login
   FileName = DB
   FileNameExtension = true
  inputPrefix = sol42_
  inputSuffix = .dat
   FieldDelimiter = ;
   ParameterFile = parameter.isc
   SqlHeader = header.sql
   SqlDetail = detail.sql
```

```
    SqlTrailer = trailer.sql
    SqlOnSuccess = success.sql
    SqlOnFailure = failure.sql
    Replace = true
    SynchronizeWithOutput = true
    NumberOfRows = 1000
}
```

Event Messages

Table 8-2 lists the EXT_InEasyDB event messages.

Table 8-2 EXT_InEasyDB Event Messages

Message	Description	Send/Recv
MSG_STREAM_START	The database input stream is started.	Send: Input module Send: Format
MSG_STREAM_END	The database input stream is stopped.	Send: Input module Send: Format
MSG_STREAM_BEGIN	The database stream starts the processing of a new input file.	Send: Input module
MSG_STREAM_END	The current file has been completely processed.	Send: Input module
MSG_STREAM_STOP	The database input stream is stopped (inactive).	Send: Input module
CMD_RENAME_INPUT_STREAM	The input file is renamed.	Receive: Output module

Events

Table 8-3 lists the EXT_InEasyDB events.

Table 8-3 EXT_InEasyDB Events

Event	Trigger	Parameter
EVT_CURSOR_OPENED	Starts the processing of a restart/ job file.	File name
EVT_INPUT_DIR_EMPTY	Control directory is empty.	Name of the control directory

EXT_InFileManager

The EXT_InFileManager module performs file handling for pipeline input from files. See:

- "About Getting Pipeline Input from Files" in *BRM Setting Up Pipeline Rating and Discounting*.
- "Configuring EDR Input Processing" in *BRM Setting Up Pipeline Rating and Discounting*.

This module runs automatically when you start Pipeline Manager.

Configure this module as a submodule of the INP_GenericStream module. See "INP_GenericStream".

To configure input from a database, see "EXT_InEasyDB".

Registry Entries

Table 8-4 lists the EXT_InFileManager registry entries.

Table 8-4 EXT_InFileManager Registry Entries

Entry	Description	Mandatory
DonePath	Specifies the path for the processed files.	Yes
DonePrefix	Specifies the prefix for the processed files.	No
DoneSuffix	Specifies the suffix for the processed files.	No
ErrorPath	Specifies the path for incorrect files.	Yes
ErrorPrefix	Specifies the prefix for incorrect files.	No
ErrorSuffix	Specifies the suffix for incorrect files.	No
InputDirEmptyTimeout	Specifies the time period (in seconds), the input directory must be empty before the EVT_INPUT_DIR_EMPTY event is sent.	No
InputPath	Specifies the path for the input files.	Yes
InputPrefix	Specifies the prefix for the input files.	No
InputSuffix	Specifies the suffix for the input files.	No
Replace	Specifies the prefix and or suffix can be replaced or appended. Default = **True**.	No
TempPrefix	Specifies the prefix for temporary files.	No

Sample Registry Section

```
InputStream
  {
    ModuleName = EXT_InFileManager
    Module
    {
      InputDirEmptyTimeout = 10
      InputPath    = ./samples/wireless/data/in
      InputPrefix = test
      InputSuffix = .edr
      DonePath     = ./samples/wireless/data/done
      DonePrefix   = test
      DoneSuffix   = .done
      ErrorPath    = ./samples/wireless/data/err
      ErrorPrefix = test
      ErrorSuffix = .err
      TempPrefix   = tmp
      Replace      = TRUE
    }
  }
```

Event Messages

Table 8-5 lists the EXT_InFileManager event messages.

Table 8-5 EXT_InFileManager Event Messages

Message	Description	Send/Receive
REQ_INPUT_FILENAME	Request to send back the complete input file name (including path) corresponding to a specific stream name (as known by the TAM).	Receive
REQ_INPUT_TEMP_FILENAME	Request to send back the temporary input file name (including path) corresponding to a specific stream name (as known by the TAM).	Receive
REQ_DONE_FILENAME	Request to send back the final input file name (including path) corresponding to a specific stream name (as known by the TAM), after successful processing.	Receive
REQ_ERROR_FILENAME	Request to send back the final input file name (including path) corresponding to a specific stream name (as known by the TAM), after an unsuccessful processing.	Receive
REQ_RETURN_FILENAME	Request to send back the return file name (including path) corresponding to a specific stream name (as known by the TAM), when batch reject was requested.	Receive
REQ_RETURN_TEMP_FILENAME	Request to send back the temporary return file name (including path) corresponding to a specific stream name (as known by the TAM), when batch reject was requested.	Receive

EXT_OutFileManager

The EXT_OutFileManager module handles files for the OUT_Generic_Stream and OUT_Reject modules. See:

- "Sending Output to a File" in *BRM Setting Up Pipeline Rating and Discounting*.
- "Configuring EDR Output Processing" in *BRM Setting Up Pipeline Rating and Discounting*.

This module runs automatically when you start Pipeline Manager.

Registry Entries

> **Note:**
>
> To ensure output file integrity, specify a unique combination of OutputPath, OutputSuffix, and OutputPrefix values for each output stream defined in the registry.

Table 8-6 lists the EXT_OutFileManager registry entries.

Table 8-6 EXT_OutFileManager Registry Entries

Entry	Description	Mandatory
AppendSequenceNumber	Specifies if the sequence number should be appended to the output file name or not. See "Applying a Prefix to the Sequence Number" in *BRM Setting Up Pipeline Rating and Discounting*.	No
DeleteEmptyFile	Deletes the output file if only the header and trailer are written to the stream. **Note:** By default, this entry is set to **True**. Configure any processes that manipulate output files to wait approximately one minute before acting on a file. This delay allows the module to delete empty files.	No
Replace	Specifies if the prefix/suffix should be replaced (**TRUE**) or appended (**FALSE**). Default = **True**.	No
SequencerPrefix	This entry is used to specify a prefix to the sequence number before it gets appended to the generated output file name. This entry is used only when **AppendSequencerNumber** is set to **True**. See "Applying a Prefix to the Sequence Number" in *BRM Setting Up Pipeline Rating and Discounting*.	No
TempPrefix	Specifies the prefix for the output stream's temporary data file. See "Configuring the Temporary File Name" in *BRM Setting Up Pipeline Rating and Discounting*. Default = **.**	No
TempDataPath	Specifies the path for the internal temporary file list. **Important:** Do not change this registry entry. It is used by the pipeline for internal data processing.	No
TempDataPrefix	Specifies the prefix for the internal temporary file list. **Important:** Do not change this registry entry. It is used by the pipeline for internal data processing.	No
TempDataSuffix	Specifies the suffix for the internal temporary file list. **Important:** Do not change this registry entry. It is used by the pipeline for internal data processing.	No
OutputPath	Specifies the path for the output files. See "Configuring File Prefixes and Suffixes" in *BRM Setting Up Pipeline Rating and Discounting*.	Yes
OutputPrefix	Specifies the prefix for the output files. See "Configuring File Prefixes and Suffixes" in *BRM Setting Up Pipeline Rating and Discounting*.	No
OutputSuffix	Specifies the suffix for the output files. See "Configuring File Prefixes and Suffixes" in *BRM Setting Up Pipeline Rating and Discounting*.	No
UseInputStreamName	Specifies to use the input file name to build the output file name. See "Creating an Output File Name from the Input File Name" in *BRM Setting Up Pipeline Rating and Discounting*.	No

Sample Registry

```
#------------------------------------------------------------------
# The /service/telco/gsm/telephony output stream
#------------------------------------------------------------------
TELOutput
{
  ModuleName = OUT_GenericStream
  Module
  {
    Grammar = ./formatDesc/Formats/Solution42/SOL42_V430_REL_OutGrammar.dsc
    DeleteEmptyStream = True
    OutputStream
      {
      ModuleName = EXT_OutFileManager
      Module
      {
        OutputPath = ./samples/wireless/data/telout
        OutputPrefix = test
        OutputSuffix = .out
        UseInputStreamName = [2,4;4,6;8,&]
        TempPrefix   = tmp.

        TempDataPath    = ./samples/wireless/data/telout
        TempDataPrefix  = tel.tmp.
        TempDataSuffix  = .data
        Replace = TRUE
        SequencerPrefix = "+"
        }
     }
  }
} # end of TELOutput
```

Messages and Requests

Table 8-7 lists the EXT_OutFileManager messages and requests.

Table 8-7 EXT_OutFileManager Messages and Requests

Message	Description	Send/Receive
REQ_EVENTHANDLER_NAME	Get the event handler.	Send

Events

Table 8-8 lists the EXT_OutFileManager events.

Table 8-8 EXT_OutFileManager Events

Event	Trigger	Parameter
EVT_OUTPUT_FILE_READY	The renaming from the temporary file to the output file.	Target file name

INP_GenericStream

The INP_GenericStream module provides the input interface to pipelines. See "Configuring EDR Input Processing" in *BRM Setting Up Pipeline Rating and Discounting*.

Registry Entries

Table 8-9 lists the INP_GenericStream registry entries.

Table 8-9 INP_GenericStream Registry Entries

Entry	Description	Mandatory
DefaultOutput	The default output stream.	
Grammar	Path to the input grammar description file.	No
InputStream	The input submodule: • EXT_InEasyDB • EXT_InFileManager	

Sample Registry for INP_GenericStream

```
InputModule
{
  ModuleName  = INP_GenericStream
  Module
  {
    DefaultOutput = EdrOutput
    Grammar       = ./FMD/Formats/Solution42/SOL42_V430_InGrammar.dsc
    InputStream
      ModuleName = EXT_InFileManager
      Module
      {
        InputPath   = ../input/maxitel/in
        InputPrefix = Sol42
        InputSuffix = .edr
        DonePath    = ../input/maxitel/done
        DonePrefix  = Sol42
        DoneSuffix  = .done
        ErrorPath   = ../input/maxitel/err
        ErrorPrefix = Sol42
        ErrorSuffix = .err
        TempPrefix  = tmp
        Replace     = TRUE
      }
    }
  }
}
```

INP_Realtime

The INP_Realtime module converts data in flist format to the EDR container format. See "Configuring a Real-Time Discounting Pipeline" in *BRM Setting Up Pipeline Rating and Discounting*.

You can use an iScript to overwrite the mappings from flist fields to EDR fields, and to add custom mappings.

For more information, see "About Customizing Mapping of Flist Fields to Rating EDR Container Fields" in *BRM Setting Up Pipeline Rating and Discounting*.

Registry Entries

Table 8-10 lists the INP_Realtime registry entries.

Table 8-10 INP_Realtime Registry Entries

Entry	Description	Mandatory
DefaultOutput	Specifies the default output module. This is always the OUT_Realtime module.	Yes
MappingScript	iScript file name.	No
OpcodeMapping	Specifies the XML file that describes the input flist field to EDR container field mapping. BRM provides the following default flist-to-EDR mappings: • Discounting: **discount_event.xml** • Rerating: **rate_event.xml** • Zoning: **zonemap_event.xml** You can customize these files to change how flists are mapped to EDR container fields. See "About Customizing Mapping of Flist Fields to Rating EDR Container Fields" in *BRM Setting Up Pipeline Rating and Discounting*.	Yes

Sample Registry Entry

In this example, **rate_event.xml** maps rerate requests in flist format to EDR container fields.

```
InputModule
{
  ModuleName = INP_Realtime
  Module
  {
     DefaultOutput = PcmOutput
     OpcodeMapping = ./formatDesc/Formats/Realtime/rate_event.xml
     MappingScript = ./inflist.isc
  }
```

INP_Recycle

The INP_Recycle module is used by standard recycling and Suspense Manager in the pre-recycling pipeline. It reads suspended usage records from the BRM database, restores original EDRs, applies edits to them, and pushes EDRs into the pre-recycling pipeline.

- For standard recycling, see "Configuring a Pre-Recycling Pipeline" in *BRM Suspending and Recycling Pipeline EDRs*.
- For Suspense Manager, see "Configuring a Pre-Recycling Pipeline" in *BRM Suspending and Recycling Pipeline EDRs*.

Dependencies

Requires connections to:

- DAT_Recycling module
- BRM database

Registry Entries

Table 8-11 lists the INP_Recycle registry entries.

Table 8-11 INP_Recycle Registry Entries

Entry	Description	Mandatory
DefaultOutput	Specifies the default output stream.	Yes
InfranetConnection	Specifies a connection to the BRM database.	Yes
InputStream	Configures the EXT_InEasyDB module. See "EXT_InEasyDB".	Yes
RecyclingDataModule	Specifies a connection to the DAT_Recycling module.	Yes

Sample Registry

```
InputModule
{
 ModuleName            = INP_Recycle
  RecyclingDataModule   = ifw.DataPool.RecyclingData
  Module
  {
    DefaultOutput = TELOutput
    RecyclingDataModule = ifw.DataPool.RecyclingData
    InfranetConnection = ifw.DataPool.LoginInfranet
    InputStream
    {
      ModuleName = EXT_InEasyDB
      Module
      {
        ControlPath           = ./data/input/db
        DataConnection        = ifw.DataPool.LoginInfranet
        FileName              = DB
        FileNameExtension     = true
        InputPrefix           = sol42_
        InputSuffix           = .dat
        FieldDelimiter        = \t
        RecordDelimiter       = \n
        ParameterFile         = parameter.isc
        # optional parameter:
        SqlHeader             = RecycleHeader.sql
        SqlDetail             = StdRecycleDetail.sql
        # optional parameter:
        # SqlTrailer            = trailer.sql
        SqlOnSuccess          = success.sql
        SqlOnFailure          = failure.sql
        Replace               = true
        SynchronizeWithOutput = true
```

```
        NumberOfRows          = 1000
      }
    } # end of InputStream
  }
} # end of InputModule
```

EDR Container Fields

Table 8-12 lists the INP_Recycle EDR container fields.

Table 8-12 INP_Recycle EDR Container Fields

Alias field name Default field name	Type	Access	Description
OVERRIDE_REASONS DETAIL.ASS_SUSPENSE_OVERRIDE_REASONS	String	Write	The suspense reasons to ignore during recycling. Used by the other pipeline modules for rating call records in spite of the pipeline validation rules they violate.
PIPELINE_NAME DETAIL.ASS_SUSPENSE_EXT.PIPELINE_NAME	String	Write	Name of the pipeline originally used for the EDR. Read from the database. Used by the IRL_PipelineSplitting module.
SUSPENSE_ID DETAIL.ASS_SUSPENSE_EXT.SUSPENSE_ID	Integer	Write	The POID ID of the */* **suspended_usage** object for this EDR. Used by Suspended Event Loader when updating suspended usage records.
BATCH_ID DETAIL.ORIGINAL_BATCH_ID	String	Write	The original routing switch batch ID. Used for revenue assurance.
PROCESS_STATUS DETAIL.INTERN_PROCESS_STATUS	Integer	Write	Indicates whether the EDR is being recycled (**1**) or test recycled (**2**).

INP_Restore

The INP_Restore module reads serialized EDRs from the file output by the OUT_Serialize module. It restores EDRs to normal format and pushes them into a pipeline.

Dependencies

Requires the use of the OUT_Serialize module to produce serialized EDRs in the correct format.

When you configure this module, you also configure the EXT_InFileManager module, which manages input, temporary, and done files. See "EXT_InFileManager" in the Infranet documentation for more information about this module.

Registry Entries

Table 8-13 lists the INP_Restore registry entries.

Table 8-13 INP_Restore Registry Entries

Entry	Description	Mandatory
DefaultOutput	Specifies the default output stream.	Yes
InputStream	Configures the EXT_InFileManager module.	Yes

Sample Registry

```
#---------------------------------
# Input section
#---------------------------------
Input
{
  UnitsPerTransaction             = 1

  InputModule
  {
    ModuleName                    = INP_Restore
    Module
    {
      DefaultOutput               = EdrOutput

      InputStream
      {
        ModuleName                = EXT_InFileManager
        Module
        {
          InputPath               = $DATA/in
          InputPrefix             = testpipeline
          InputSuffix             = .edr

          DonePath                = $DATA/done
          DonePrefix              = testpipeline
          DoneSuffix              = .done

          ErrorPath               = $DATA/err
          ErrorPrefix             = testpipeline
          ErrorSuffix             = .err

          TempPrefix              = tmp

          Replace                 = True

          InputDirEmptyTimeout    = 60
        }
      }
    }
  }
}
```

OUT_DB

The OUT_DB module sends output to the Pipeline Manager database.

See "Sending Output to a Database" in *BRM Setting Up Pipeline Rating and Discounting*.

Dependencies

Requires a connection to the Pipeline Manager database.

Registry Entries

Table 8-14 lists the OUT_DB registry entries.

Table 8-14 OUT_DB Registry Entries

Entry	Description	Mandatory
ControlPath	Specifies the name of the control/configuration directory. See "About the OUT_DB Module Configuration Files" in *BRM Setting Up Pipeline Rating and Discounting*.	Yes
DataConnection	Specifies the connection to the Pipeline Manager database.	Yes
DeleteWithoutDetails	Specifies if an empty output stream should force a rollback of all actions. Default = **True** See "Handling Empty Output Streams" in *BRM Setting Up Pipeline Rating and Discounting*.	No
Destination	Specifies the value for the destination field in the header record. See "Specifying the Destination" in *BRM Setting Up Pipeline Rating and Discounting*.	No
DetailTableDefinition	Specifies the name of the file that contains the description of the destination detail table. See "About the OUT_DB Module Configuration Files" in *BRM Setting Up Pipeline Rating and Discounting*.	Yes
FieldDelimiter	Specifies the delimiter between each field needed by the tokenizer. See "About the OUT_DB Module Configuration Files" in *BRM Setting Up Pipeline Rating and Discounting*.	Yes
HeaderTableDefinition	Specifies the name of the file that contains the description of the destination header table. See "About the OUT_DB Module Configuration Files" in *BRM Setting Up Pipeline Rating and Discounting*.	No
NumberOfRows	Specifies the array size for the bulk inserter. A good value is **500**. See "About the OUT_DB Module Configuration Files" in *BRM Setting Up Pipeline Rating and Discounting*.	Yes
ParameterFile	Specifies the name of parameter file which contains optional key/value entries See "Parameter File" in *BRM Setting Up Pipeline Rating and Discounting*.	Yes
RowNumAlias	Specifies the alias that is replaced by the inserted rows. See "About the OUT_DB Module Configuration Files" in *BRM Setting Up Pipeline Rating and Discounting*.	Yes
SaveConfigurationFile	Specifies whether to delete the configuration file of each stream. Default = **False** See "About the OUT_DB Module Configuration Files" in *BRM Setting Up Pipeline Rating and Discounting*.	No

Table 8-14 (Cont.) OUT_DB Registry Entries

Entry	Description	Mandatory
Source	Specifies the value for the source field in the header record. See "Specifying the Source" in *BRM Setting Up Pipeline Rating and Discounting*.	No
SqlBeginStream	Specifies the name of SQL file that contains an SQL statement. See "SqlBeginStream" in *BRM Setting Up Pipeline Rating and Discounting*.	No
SqlEndStream	Specifies the name of SQL file that contains an SQL statement. See "SqlEndStream" in *BRM Setting Up Pipeline Rating and Discounting*.	No
StreamNameAlias	Specifies the alias that is replaced by the internal stream name value. See "About the OUT_DB Module Configuration Files" in *BRM Setting Up Pipeline Rating and Discounting*.	No
TrailerTableDefinition	Specifies the name of the file that contains the description of the destination trailer table. See "About the OUT_DB Module Configuration Files" in *BRM Setting Up Pipeline Rating and Discounting*.	No
WriteDefaultEdr	Specifies if a default EDR is written in empty data streams. Default = **False** See "Handling Empty Output Streams" in *BRM Setting Up Pipeline Rating and Discounting*.	No

Sample Registry

```
Streams
{
  EdrOutput
  {
    StreamDestination       = Database
    Destination             = CBC21
    Source                  = D1
    DataConnection          = integrate.DataPool.Login
    NumberOfRows            = 500
    ControlPath             = control
    FieldDelimiter          = ;
    RowNumAlias             = __RowNum__
    StreamNameAlias         = __StreamName__
    SqlBeginStream          = beginStream.sql
    SqlEndStream            = endStreamNeu.sql
    ParameterFile           = parameter_out.isc
    HeaderTableDefinition   = headerTable.dat
    DetailTableDefinition   = detailTable.dat
    TrailerTableDefinition  = trailerTable.dat
    WriteDefaultEdr         = false
    DeleteWithoutDetails    = true
    SaveConfigurationFile   = true
  }
  Reject
  {
    StreamDestination       = File
    OutputPath              = /data/reject
    OutputSuffix            = .rej
```

```
    Replace                 = True
  }
}
```

> **Note:**
>
> To ensure output file integrity, specify a unique combination of OutputPath, OutputSuffix, and OutputPrefix values for each output stream defined in the registry.

OUT_DevNull

The OUT_DevNull module removes EDRs that are not needed by Pipeline Manager. See "Configuring Output of Discarded EDRs" in *BRM Setting Up Pipeline Rating and Discounting*.

For more information, see "Discarding and Skipping EDRs" in *BRM Setting Up Pipeline Rating and Discounting*.

Sample Registry

```
OutputCollection
    {
    DevNull
    {
        ModuleName = OUT_DevNull
        Module
        {
        }
    }
```

OUT_GenericStream

The OUT_GenericStream module handles the output stream for rated EDRs. See "Configuring EDR Output Processing" in *BRM Setting Up Pipeline Rating and Discounting*.

When you configure the OUT_GenericStream module, you configure the EXT_OutFileManager module to specify file management options. See "EXT_OutFileManager".

Registry Entries

Table 8-15 lists the OUT_GenericStream registry entries.

Table 8-15 OUT_GenericStream Registry Entries

Entry	Description	Mandatory
AddInvoiceData	When set to True, the output module adds invoice data to each BRM billing record. Default = **False** See "Adding Pipeline Rating Data to an Invoice" in *BRM Setting Up Pipeline Rating and Discounting*.	No

Table 8-15 (Cont.) OUT_GenericStream Registry Entries

Entry	Description	Mandatory
DeleteEmptyStream	Specifies to delete empty output streams. If you run multiple RE Loader processes in parallel, set this option to **True**. Otherwise, RE Loader attempts to load the empty files. Default = **True**	No
EventType	Specifies the BRM event type that the output file contains, such as **/event/delayed/session/gsm**.	Mandatory only for RE Loader-related pipelines
Grammar	Specifies the output grammar description file.	Yes
OutputStream	Contains the configuration for the EXT_OutFileManager.	Yes
ProcessType	Specifies which process created the output file and can be set to one of the following values: • **RATING_PIPELINE** • **EVENT_EXTRACTION_TOOL** • **BACKOUT_PIPELINE** • **RERATING_PIPELINE** • **PIN_TRANSFORM_CDR**	Mandatory only for RE Loader-related pipelines
Sequencer	Specifies the Sequencer for performing sequence generation. This Sequencer must be defined in the **SequencerPool** section of the registry file.	No

Sample Registry

```
#------------------------------------------------------------------------
# The /service/telco/gsm/telephony output stream
#------------------------------------------------------------------------
TELOutput
{
  ModuleName = OUT_GenericStream
  ProcessType = RATING_PIPELINE
  EventType = /event/delayed/session/gsm

  Module
  {
    Grammar = ./formatDesc/Formats/Solution42/SOL42_V430_REL_OutGrammar.dsc
    DeleteEmptyStream = True
    Sequencer = SequenceGenerator_1
    OutputStream
      {
      ModuleName = EXT_OutFileManager
      Module
      {
        OutputPath = ./samples/wireless/data/telout
        OutputPrefix = test
        OutputSuffix = .out
        TempPrefix   = tmp.
        TempDataPath    = ./samples/wireless/data/telout
        TempDataPrefix  = tel.tmp.
        TempDataSuffix  = .data
        Replace = TRUE
        }
     }
```

```
  }
} # end of TELOutput
```

> **Note:**
>
> To ensure output file integrity, specify a unique combination of OutputPath, OutputSuffix, and OutputPrefix values for each output stream defined in the registry.

OUT_Realtime

The OUT_Realtime module converts data in the pipeline EDR output to flist format. It sends the output to the NET_EM module automatically. You don't need to configure a pointer to the NET_EM module.

You can use an iScript to overwrite the mappings and to add custom mappings. You use the registry file to specify the iScript.

For more information, see "Configuring a Real-Time Discounting Pipeline" in *BRM Setting Up Pipeline Rating and Discounting*.

Registry Entries

Table 8-16 lists the OUT_Realtime registry entries.

Table 8-16 OUT_Realtime Registry Entries

Entry	Description	Mandatory
MappingScript	iScript file name.	No
DoRating	Specifies the pipeline to be used. Set to **False** for Discounting/Zoning pipeline. Set to **True** for Rerating pipeline.	Yes

Sample Registry Entry

```
OutputCollection
{
  PcmOutput
  {
    ModuleName = OUT_Realtime
    Module
    {
       DoRating = false
       #MappingScript = <Optional Output IScript>
    }
  }
}
```

OUT_Reject

The OUT_Reject module writes rejected EDRs to an output stream. The written record is exactly the same as the original input record. See "Configuring Output for Rejected or Duplicate EDRs" in *BRM Setting Up Pipeline Rating and Discounting*.

When you configure the OUT_Reject module, you configure the EXT_OutFileManager module to specify file management options. See "EXT_OutFileManager".

Sample Registry

```
Reject
{
  ModuleName = OUT_Reject
  Module
  {
    OutputStream
    {
      ModuleName = EXT_OutFileManager
      Module
      {
        OutputPath = ../output/tel/rej
        OutputPrefix = Sol42
        OutputSuffix = rej
        TempPrefix   = tmp
        Replace = TRUE
        DeleteEmptyFile = FALSE
      }
    }
  }
}
```

> **Note:**
>
> To ensure output file integrity, specify a unique combination of OutputPath, OutputSuffix, and OutputPrefix values for each output stream defined in the registry.

OUT_Serialize

The OUT_Serialize module creates serialized EDR records with base64 encoding.

Dependencies

The INP_Restore module is required to read and restore EDRs serialized by this module.

When you configure this module, you also configure the EXT_OutFileManager module, which manages output files. See "EXT_OutFileManager" in the Infranet documentation for more information about this module.

Registry Entries

Table 8-17 lists the OUT_Serialize registry entries.

Table 8-17 OUT_Serialize Registry Entries

Entry	Description	Mandatory
DeleteEmptyStream	Specifies whether empty streams should be deleted. Set to **True** to prevent SEL from attempting to load empty files. Default = **True**	Yes
ProcessType	Specifies which process created the output file. If used, sets the HEADER.CREATION_PROCESS field in the EDR to the value specified.	No
EventType	Specifies the Infranet event type that the output file contains, such as **/event/delayed/session/gprs**. If used, sets the HEADER.EVENT_TYPE field to the values specified.	No
OutputStream	Configures the EXT_OutFileManager.	Yes

Sample Registry

```
#------------------------------------------------------------
# The serialized EDR output stream
#------------------------------------------------------------
SerEdrOutput
{
  ModuleName = OUT_Serialize
  Module
  {
    DeleteEmptyStream = True
    ProcessType = RECYCLING_PIPELINE
    EventType = /event/delayed/session/gprs

    OutputStream
    {
      ModuleName = EXT_OutFileManager
      Module
      {
        OutputPath = $DATA/out
        OutputPrefix = test
        OutputSuffix = .out

        TempPrefix   = .
        TempDataPath    = $DATA/out
        TempDataPrefix  = tel.tmp.
        TempDataSuffix  = .data

        Replace = TRUE
      }
    }
  }
} # end of SerEdrOutput
```

Pipeline Dispatcher

Parses CDR files to multiple identical pipelines. This module routes files with a specified filename prefix and suffix from a single input directory to multiple pipelines in round-robin fashion.

Registry Entries

Table 8-18 lists the Pipeline Dispatcher registry entries.

Table 8-18 Pipeline Dispatcher Registry Entries

Entry	Value	Description	Mandatory
DispatcherName	N/A	Specifies the name of the dispatcher. You can use any name, for example, **Dispatcher1**. If your system requires multiple dispatchers, create a set of entries for each dispatcher.	N/A
DispatcherName.**InputPath**	String	Specifies the path of the CDR input directory. For example: **InputPath = ./samples/wireless/data/input**	N/A
DispatcherName.**InputPrefix**	String	Specifies the prefix of all CDR files you want routed. The dispatcher only routes files with the specified prefix and suffix to your pipelines. For example: **InputPrefix = test**	N/A
DispatcherName.**InputSuffix**	String	Specifies the suffix of all CDR files you want routed. The dispatcher only routes files with the specified prefix and suffix to your pipelines. For example: **InputSuffix = .edr**	N/A
DispatcherName.**TargetPipelines**	N/A	Subgroup that lists which pipelines to route your CDR files.	N/A
DispatcherName.**TargetPipelines.DestinationPipeline**	String	Specifies to which pipelines to route CDR files. Add an entry for each pipeline. For example: **DestinationPipeline = ifw.Pipelines.W_SAMPLE** **DestinationPipeline = ifw.Pipelines.W_SAMPLE_2** **Important:** The pipelines must use a separate input directory from the CDR input files.	N/A
DispatcherName.**TargetPipelines.Routing**	ROUND_ROBIN	Specifies to use round-robin routing. **Note:** The dispatcher uses round-robin routing only.	Yes

Sample Registry

```
ifw
{
    PipelineDispatcher
    {
        ModuleName = EXT_PipelineDispatcher
        Module
        {
            Dispatcher1
            {
                InputPath = ./samples/wireless/input
                InputPrefix = TAP3
                InputSuffix = .edr
```

```
                    TargetPipelines
                    {
                        DestinationPipeline = ifw.Pipelines.Pipeline_1
                        DestinationPipeline = ifw.Pipelines.Pipeline_2
                        Routing = ROUND_ROBIN
                    }
                }
            }
        }
    }
```

9

Pipeline Manager Framework Modules

Learn about the Oracle Communications Billing and Revenue Management (BRM) Pipeline Manager framework modules.

Topics in this document:

Controller

Use the Pipeline Manager Controller to control and monitor components in the Pipeline Manager framework. The Controller also generates the log message table that is used by the LOG module to generate the Process log file, the Pipeline Manager log file, and the Stream log file. For information, see "LOG".

Registry Entries

Table 9-1 lists the Controller registry entries.

Table 9-1 Controller Registry Entries

Entry	Value	Description	Mandatory?
Active	**True** **False**	Activates or deactivates the Pipeline Manager framework. • **True** activates the Pipeline Manager framework. • **False** deactivates the Pipeline Manager framework.	Yes
DataPool	N/A	Subgroup that contains entries for all data modules in the Pipeline Manager framework.	Yes
DiagnosticDataHandler	N/A	Subgroup that configures diagnostic data collection.	No
EventHandler	N/A	Subgroup that contains the Event Handler entries.	No
Instrumentation	N/A	Subgroup that configures Operations Management Framework (OMF) instrumentation data collection.	Yes
Instrumentation.ProbeBroker	String	Specifies the path to the OMF instrumentation folder.	Yes
Instrumentation.HttpServer	String	Specifies configuration data for the OMF HTTP server.	No
LogMessageTable.MessageFilePath	String	Specifies the path to your error message files. By default, the Pipeline Manager installer installs your error message files in the *Pipeline_home***/etc** directory. *Pipeline_home* is the directory where you installed Pipeline Manager.	No
LogMessageTable.MessageFilePrefix	String	Specifies the prefix for your error message files.	No
LogMessageTable.MessageFileSuffix	String	Specifies the suffix for your error message files.	No
MemoryMonitor	N/A	Subgroup that configures memory monitoring. See "Memory Monitor".	No
Pipelines	N/A	Subgroup that contains the entries for each pipeline.	Yes
ProcessLog	N/A	Subgroup that contains entries for the main processing log.	Yes
ProcessLoopTimeout	Integer	Specifies the interval, in seconds, between polling for a new semaphore file. This parameter controls the overall event loop, which includes looking for semaphore files. The value must be greater than **0**.	Yes
QueueRequestTimeout	Integer	Specifies the interval for logging buffer sizes, in seconds. A value of **0** turns off logging.	Yes
Registry	N/A	Subgroup that contains registry processing entries.	Yes
Registry.Buffer	N/A	Subgroup that specifies the registry buffer's size and type. The registry entries in this subgroup depend on the buffer type.	Yes
Registry.FileName	String	Specifies the name of a file in dot-separated format that contains current updated registry settings, including semaphore updates.	Yes
Registry.FilePath	String	Specifies the path to the registry file.	Yes

Table 9-1 (Cont.) Controller Registry Entries

Entry	Value	Description	Mandatory?
Registry.NiceFormatFileName	String	Specifies the name of a file that contains current updated registry settings, including semaphore updates. This file format is easier to read than **.FileName** format and can be used to debug a registry file or create a new one.	Yes
Semaphore	N/A	Subgroup that contains semaphore processing entries.	Yes
Semaphore.FileName	String	Specifies the name of the semaphore file.	Yes
Semaphore.FilePath	String	Specifies the path to the semaphore file.	Yes
Semaphore.RetainFiles	**True** **False**	Specifies whether semaphore files are deleted or saved after they are processed. • **True** specifies to save semaphore files. The Controller renames the file by appending the current timestamp to the file name in the format *YYYYMMDD_hhmmss* and logs the semaphore file's new name in the **process.log** file. For example, the **semaphore.reg** file is renamed **semaphore_20031022_120803.reg**. • **False** specifies to delete semaphore files immediately after they are processed. The default is **False**.	No
TransactionIDController	N/A	Subgroup that contains the Transaction ID Controller entries.	Yes

Sample Registry File

```
ifw
{
    Active = TRUE
    ProcessLoopTimeout = 10
    QueueRequestTimeout = 0
    Semaphore
    {
        FilePath = ./semaphore
        FileName = semaphore.reg
        RetainFiles = False
    }
    Registry
    {
        FilePath = ./info
        FileName = Sample.reg
        NiceFormatFileName = niceSample.reg
        Buffer
        {
            Size = 1000
        }
    }
    ProcessLog
    {
        ModuleName = LOG
        Module
        {
```

```
        }
    }
    LogMessageTable
    {
        MessageFilePath= ./etc
        MessageFileSuffix= .msg
    }
    EventHandler
    {
        ModuleName = EVT
        Module
        {
        }
    }
    DataPool
    {
        Login
        {
            ModuleName = DBC
            Module
            {
                UserName = TEST
                PassWord = password
                DatabaseName = TE01
                AccessLib = oci231
                Connections = 5
            }
        }
    }
    TransactionIdController
    {
        Source = Database
        Generator
        {
            DataConnection = integrate.DataPool.Login
        }
    }
    Pipelines
    {
    }
}
```

Semaphore File Entries

Table 9-2 lists the Controller Semaphore file entries.

Table 9-2 Controller Semaphore File Entries

Entry	Description
Active	Starts and stops the Pipeline Manager framework.
LogTimeStatistic	Specifies whether to write time statistics into the process log file.
QueueRequestTimeout	Specifies the interval for logging buffer sizes, in seconds. A value of **0** turns off logging.

Sample Semaphore Entry

```
ifw.Pipelines.ALL_RATE.Active = True
```

Events

Table 9-3 lists the Controller Events.

Table 9-3 Controller Events

Event	Trigger
EVT_INTEGRATE_START	Pipeline Manager starts processing.
EVT_INTEGRATE_STOP	Pipeline Manager terminates. No more files are processed.

Database Connect (DBC)

Connects the Pipeline Manager framework to the Pipeline Manager database.

Registry Entries

Table 9-4 lists the Database Connect registry entries.

Table 9-4 Database Connect Registry Entries

Entry	Value	Description	Mandatory?
AccessLib	**oci10g72** **oci231**	Specifies the name of the database access library, without the **lib** prefix and **.so** suffix. • Use **oci10g72** for Oracle10g databases. • Use **oci231** for appropriate Oracle databases.	Yes
Connections	Integer	Specifies the number of database connections that a database module holds open in a connection pool. The default is **1**.	No
DatabaseName	String	Specifies the database name.	Yes
PassWord	String	Specifies the encrypted database password in hexadecimal format.	Yes
ServerName	String	Specifies the server name. The default is **''**.	No
UserName	String	Specifies the database user name.	Yes
ConvertToUpperCase	True False	Specifies whether to convert the user name used by the password decryption to uppercase. The default is **True**.	No

Sample Registry for Oracle Databases

```
ifw
{
    DataPool
    {
    ...
        #------------------------------------------------------------
        # Database Connection Pipeline
        #------------------------------------------------------------
        Login
        {
```

```
                ModuleName = DBC
                Module
                {
                    UserName = USER
                    PassWord = 574B93D1CBF21D1161611E0A07
                    DatabaseName = ORA_DB
                    AccessLib = oci231
                    Connections = 5
                }
            }
        ...
        }
}
```

Semaphore Entries

Table 9-5 lists the Database Connect Semaphore entry.

Table 9-5 Database Connect Semaphore Entry

Parameter	Description	Mandatory
Reconnect	Closes all open database connections and reconnects to the database.	No

Sample Semaphore

```
ifw.DataPool.Login.Module.Reconnect {}
```

EDR Factory

Use the EDR Factory to generate and allocate memory to EDR containers. The EDR Factory uses a container description file to generate each container.

Registry Entries

Table 9-6 lists the EDR Factory registry entries.

Table 9-6 EDR Factory Registry Entries

Entry	Value	Description	Mandatory?
DataConnection	String	Specifies the registry name of the database connection module (DBC). When you use this entry, the EDR Factory connects to the IFW_ALIAS_MAP database table to retrieve alias names. **Note:** Use this field when you use aliases to describe EDR container fields.	No
Description	String	Specifies the path to the container description file. See "About the Container Description File" in *BRM Setting Up Pipeline Pricing*.	Yes
EdrVersionDataConnection	String	Specifies the registry name of the database connection module (DBC). When you use this entry, the EDR Factory connects to the EDR_FIELD_MAPPING_T database table to retrieve the EDR field mapping.	No

Table 9-6 (Cont.) EDR Factory Registry Entries

Entry	Value	Description	Mandatory?
UsageStatistic	String	Specifies the name of the usage statistic file. This file lists all modules that are using EDR container fields together with the fields that are accessed by each module.	No

Sample Registry

```
EdrFactory
{
    DataConnection = integrate.DataPool.Login
    Description = ./FMD/Portal/containerDesc.dsc
    UsageStatistic = edrStatistic.txt
    EdrVersionDataConnection = ifw.DataPool.LoginInfranet
}
```

Event Handler

Use the Event Handler to start external programs when triggered by internal Pipeline Manager events.

Registry Entries

Table 9-7 lists the Event Handler registry entries.

Table 9-7 Event Handler Registry Entries

Entry	Description	Mandatory?
Buffer	Subgroup that specifies the size and type of the Event Handler's internal queue buffer. The registry entries in this subgroup depend on the buffer type.	Yes
Events	Subgroup that contains trigger entries.	Yes
Event.*EventName*	Specifies the event that triggers an external program. Add an entry for each event that triggers an external program.	Yes
Event.*ModuleSendingEvent*	Specifies the registry name of the module that sends the event to the Event Handler. Add an entry for each module that can trigger an external program.	Yes
TimeToWait	Specifies the time in seconds that the Event Handler waits for the external program to terminate. By default, no **TimeToWait** value is assumed.	No

Sample Registry

```
EventHandler
{
    ModuleName = EVT
    Module
    {
        Events
```

```
            {
                ifw.DataPool.Customer.Module
                {
                    EVT_ReloadSuccess = ./script/script_1
                    EVT_ReloadFailed = ./script/script_2
                    TimeToWait = 30
                }
                ifw.Pipelines.*
                {
                    EVT_PIPELINE_START = ./script/script_3
                    TimeToWait = 10
                }
            }
            Buffer
            {
                Size = 10
            }
        }
}
```

Instances

Use the Instances module to configure multiple instances of sequencers, output streams, or system brands for multiple roaming partners. This module is optional.

> **Note:**
>
> This module does not configure multiple instances of pipelines. To do that, use the **ifw.Pipelines.Instances** subsection.

Registry Entries

Table 9-8 lists the Instances registry entries.

Table 9-8 Instances Registry Entries

Entry	Description	Mandatory?
InstantiationName	Specifies the descriptive name of the instantiation, for example, TAPOutputStreamsInstantiation.	Yes, if the Instances module is used.
InstantiationName.**BlockName**	Specifies the template section or entry in the roaming registry file that is used to instantiate multiple registry sections or entries. The template section or entry contains variables for the section name, entry name, or the value of the entry that must be changed in each new instance created.	Yes
InstantiationName.**DataFile**	Specifies the path to the data file generated by the **RoamingConfigGen64** utility. See "RoamingConfigGen64" for more information.	Yes
InstantiationName.**InstanceSpecificEntries**	Subgroup that specifies the entries that must be changed in each new instance created, such as the section name, entry name, the value of an entry, the change mode, and so on.	Yes

Table 9-8 (Cont.) Instances Registry Entries

Entry	Description	Mandatory?
InstantiationName.**InstanceSpecificEntries**.*InstanceChangeName*	Specifies the descriptive name of the change required in each instance; for example, ModifyBlockName.	Yes
InstantiationName.**InstanceSpecificEntries**.*InstanceChangeName*.**Instance**	Specifies whether to change the section name, entry name, or the value of the entry in each new instance created. Valid values are: • **[BlockName]** specifies that the section name or entry name must be changed in each new instance. For example, to change the section name of the **SequencerPool.SEQ_GEN_TAPOUT_XXX** subsection in each new instance (such as **SEQ_GEN_TAPOUT_OPR01**, **SEQ_GEN_TAPOUT_OPR02**, and so on), set the **Instance** entry to **[BlockName]**. • **[BlockValue]** specifies that the value of the entry must be changed in each new instance. **Note:** Use this value only if *InstantiationName*.**BlockName** is a template entry: do not use this value if it is a template section. For example, to change the value of the **SystemBrands.XXX** entry in each new instance (such as TAPOutput_OPR01, TAPOutput_OPR02, and so on), set the **Instance** entry to **[BlockValue]**. • *RegistryEntry* specifies the entry in the template registry section for which the value must be changed in each new instance. **Note:** Use this value only if *InstantiationName*.**BlockName** is a template section: do not use this value if it is a template entry. For example, to change the value of the **Module.Recipient** entry in the **TAPOutput_XXX** section, set the **Instance** entry to **Module.Recipient**.	Yes
InstantiationName.**InstanceSpecificEntries**.*InstanceChangeName*.**UseColumn**	Specifies the column in the data file generated by the **RoamingConfigGen64** utility. This column is used to change the section name, entry name, or the value of the entry in each instance according to the change mode. For example, the TAPOUT_SEQUENCER column is used to change the section name in each instance of the **SequencerPool.SEQ_GEN_TAPOUT_XXX** subsection. See "Sample Registry for Multiple Instances of Sequencers".	Yes

Table 9-8 (Cont.) Instances Registry Entries

Entry	Description	Mandatory?
InstantiationName.**InstanceSpecificEntries**.*InstanceChangeName*.**Mode**	Specifies the mode of changing the section name, entry name, or the value of the entry in each instance using the column values in the data file generated by the **RoamingConfigGen64** utility. Valid values are: • **REPLACE** specifies that the section name, entry name, or the value of the entry is replaced with the corresponding column value from the data file. For example, if the entry name is *XXX* and the corresponding column value is **OPR01**, *XXX* is replaced with **OPR01** in the newly created instance. • **PREFIX** specifies that the corresponding column value is prefixed with the section name, entry name, or the value of the entry in each instance. For example, if the value of the entry is **.tmp** and the corresponding column value is **OPR01**, **OPR01** is prefixed with **.tmp** and the value is added as **OPR01.tmp** in the newly created instance. • **SUFFIX** specifies that the corresponding column value is suffixed with the section name, entry name, or the value of the entry in each instance. For example, if the value of the entry is **NREUR01** and the corresponding column value is **OPR01**, **OPR01** is suffixed with **NREUR01** and the value is added as **NREUR01OPR01** in the newly created instance. The default mode is **REPLACE**.	No

Sample Registry for Multiple Instances of Sequencers

```
Instances
{
   SEQ_GEN_TAPOUT
    {
     BlockName  = SequencerPool.SEQ_GEN_TAPOUT_XXX
     DataFile = ./RoamingPartnerConf.dat
     InstanceSpecificEntries
     {
         ModifyBlockName
         {
           Instance = [BlockName]
           UseColumn = TAPOUT_SEQUENCER
          }
      }
    }
}
SequencerPool
{
SEQ_GEN_TAPOUT_XXX
{
  Source = Database
  Controller
  {
   SequencerType = Generation
   ReuseGap = True
   SequenceLength = 5
   DatabaseConnection = ifw.DataPool.Login
  }
}
```

Sample Registry for Multiple Instances of System Brands

```
Instances
{
 EventSplitting
  {
   BlockName  =
Pipelines.TAPOutCollectPipeline.Functions.Processing.FunctionPool.RoamPartner_EventSplitt
ing.Module.SystemBrands.XXX
   DataFile = ./RoamingPartnerConf.dat
   InstanceSpecificEntries
   {
     ModifyBlockName
     {
       Instance = [BlockName]
       UseColumn = VPLMN
     }
     ModifyBlockValue
     {
      Instance = [BlockValue]
      UseColumn = TAPOUT_STREAM
     }
   }
}
RoamPartner_EventSplitting
{
   ModuleName = FCT_EnhancedSplitting
   Module
   {
     Active          = True
     DataConnection  = ifw.DataPool.Login
     DefaultOutput   = SuspenseCreateOutput
     SystemBrands
     {
     XXX         = TAPOutput_XXX
      SUSP        = SuspenseCreateOutput
     }
   }
}
```

Sample Registry for Multiple Instances of Output Streams

```
Instances
{
TAPOutputStreaminstantiation
    {
    BlockName = Pipelines.TAPOutCollectPipeline.Output.OutputCollection.TAPOutput_XXX
    DataFile = ./RoamingPartnerConf.dat
    InstanceSpecificEntries
    {
      ModifyBlockName
      {
        Instance = [BlockName]
        UseColumn = TAPOUT_STREAM
       }
      ModifyOutputStreamSequencer
      {
          Instance = Module.Sequencer
          UseColumn = TAPOUT_SEQUENCER
      }
```

```
        ModifyRecipient
        {
            Instance = Module.Recipient
            UseColumn = VPLMN
         }
        ModifyCountryCode
        {
            Instance = Module.CountryCode
            UseColumn = COUNTRYCODE
         }
        ModifyDecimalPlaces
        {
            Instance = Module.DecimalPlaces
            UseColumn = DECIMALPLACES
         }
         ModifyOutputPath
         {
            Instance = Module.OutputStream.Module.OutputPath
            UseColumn = TAPOUT_PATH
         }
         ModifyOutputPrefix
         {
            Instance = Module.OutputStream.Module.OutputPrefix
            UseColumn = TAPOUT_PREFIX
         }
         ModifyTempPrefix
         {
            Instance = Module.OutputStream.Module.TempPrefix
            UseColumn = TMP_PREFIX
         }
         ModifyTempDataPath
         {
             Instance = Module.OutputStream.Module.TempDataPath
             UseColumn = TAPOUT_PATH
          }
          ModifyTempDataPrefix
          {
              Instance = Module.OutputStream.Module.TempDataPrefix
              UseColumn = TMP_DATA_PREFIX
           }
    }
}
TAPOutput_XXX
{
ModuleName  = OUT_GenericStream
ProcessType = TAPOUTCOLLECT_PIPELINE
EventType   = /event/delayed/session/telco/gsm
Module
{
   Grammar = ./formatDesc/Formats/TAP3-NG/TAP_0312_OutGrammar.dsc
   DeleteEmptyStream = False
   Sequencer = SEQ_GEN_TAPOUT_XXX
   Sender = PORTL
   Recipient = XXX
   CountryCode = XXX
   DecimalPlaces = XXX
   OutputStream
   {
     ModuleName = EXT_OutFileManager
     Module
     { OutputPath            = ./data/outcollect/tapout/XXX
       OutputPrefix          = CDPORTLXXX
```

```
        TempPrefix            = tmptest_XXX_
        TempDataPath          = ./data/outcollect/tapout/XXX
        TempDataPrefix        = test.XXX.tmp.
        TempDataSuffix        = .data
        UseInputStreamName    = [0,0]
        SequencerPrefix       = ""
        AppendSequenceNumber  = True
      }
    }
}

NRTRDEOutput_XXX
          {
            ModuleName  = XXX
            ProcessType = TAPOUTCOLLECT_PIPELINE
            EventType   = /event/delayed/session/telco/gsm

            Module
            {
              Grammar = ./formatDesc/Formats/TAP3/NRTRDE2_v01_OutGrammar.dsc
              DeleteEmptyStream = False
              Sequencer = SEQ_GEN_NRTRDEOUT_XXX

              Sender = EUR01
              Recipient = XXX

              OutputStream
              {
                ModuleName = EXT_OutFileManager
                Module
                {
                  OutputPath            = ./data/outcollect/nrtrdeout/XXX
                  OutputPrefix          = NREUR01XXX
                  TempPrefix            = tmptest_XXX_
                  TempDataPath          = ./data/outcollect/nrtrdeout/XXX
                  TempDataPrefix        = test.XXX.tmp.
                  TempDataSuffix        = .data
                  UseInputStreamName    = [0,0]
                  SequencerPrefix       = ""
                  AppendSequenceNumber  = True
                }
              }
            }
          }

NRTRDEOutputStreams
   {
   BlockName=Pipelines.TAPOutCollectPipeline.Output.OutputCollection.NRTRDEOutput_XXX
      DataFile = ./conf/RoamingPartnerConf.dat
      InstanceSpecificEntries
      {
        ModifyBlockName
        {
          Instance = [BlockName]
          UseColumn = NRTRDEOUT_STREAM
        }
        ModifyModuleName
        {
          Instance = ModuleName
          UseColumn = NRTDEOUTPUTSTREAMMODULE
        }
        ModifyOutputStreamSequencer
```

```
            {
              Instance = Module.Sequencer
              UseColumn = NRTRDEOUT_SEQUENCER
            }
            ModifyRecipient
            {
              Instance = Module.Recipient
              UseColumn = VPLMN
            }
            ModifyOutputPath
            {
             Instance = Module.OutputStream.Module.OutputPath
             UseColumn = NRTRDEOUT_PATH
            }
            ModifyOutputPrefix
            {
              Instance = Module.OutputStream.Module.OutputPrefix
              UseColumn = NRTRDEOUT_PREFIX
            }
            ModifyTempPrefix
            {
              Instance = Module.OutputStream.Module.TempPrefix
              UseColumn = TMP_PREFIX
            }
            ModifyTempDataPath
            {
              Instance = Module.OutputStream.Module.TempDataPath
              UseColumn = NRTRDEOUT_PATH
            }
            ModifyTempDataPrefix
           {
              Instance = Module.OutputStream.Module.TempDataPrefix
              UseColumn = TMP_DATA_PREFIX
            }
        }
}
```

LOG

Use the LOG module to manage and create your system log files:

Dependencies

The LOG module needs access to the log message table generated by the Controller in order to create the system log files. For information, see "Controller".

Registry Entries

Table 9-9 lists the LOG registry entries.

Table 9-9 LOG Registry Entries

Entry	Description	Mandatory?
FileName	Specifies the name of your system log file. If empty, the name will be built by the date.	No
FilePath	Specifies the path to your system log file.	No

Table 9-9 (Cont.) LOG Registry Entries

Entry	Description	Mandatory?
FilePrefix	Specifies the log file prefix.	No
FileSuffix	Specifies the log file suffix.	No
LogLevel	Specifies the minimum severity limit. All messages greater or equal to the limit are logged. For example, enter **major** to log only major and critical error messages. Values are: • **critical** • **major** • **minor** • **warning** • **normal** • **debug** The default is **normal**, which means that all messages are logged. Using the **debug** setting returns additional debugging data.	No
MessageGroup	Specifies the message group name.	No
ProcessName	Specifies the process name.	No
ShowOriginator	Specifies whether to write the name of the module that emitted the message to the log file. **True** specifies to log the module name. This helps Oracle Technical support troubleshoot any problems. **False** specifies to not log the module name. The default is **False**.	No
SuppressErrors	Specifies any error messages to exclude from log files. For example, enter ERR_INSERTING_CLI to prevent those error messages from being logged.	No
WriteMessageKey	Specifies whether the module logs error codes. For example: ERR_FILE_NOT_FOUND. **True** specifies to write both the error code and error message to the log file. This helps technical support troubleshoot any problems. **False** specifies to write only the error message to the log file. The default is **False**.	No

Sample Registry Entry for the Process Log

```
ProcessLog
{
    ModuleName = LOG
    Module
    {
        ITO
        {
            FilePath = /ifw/log/process
            FileName = process
            FilePrefix = log_
            FileSuffix = .log
            LogLevel = normal
            ProcessName = ifw
            SuppressErrors
            {
```

```
                INF_IGNORE_CLI
                ERR_INSERTING_CLI
            }
        }
    }
}
```

Sample Registry Entry for the Pipeline Log

```
PipelineLog
{
    ModuleName = LOG
    Module
    {
        ITO
        {
            FilePath = /ifw/log/pipeline
            FileName = pipe2
            FileSuffix = .log
            LogLevel = minor
            SuppressErrors
            {
                INF_IGNORE_CLI
                ERR_INSERTING_CLI
            }
        }
    }
}
```

Sample Registry Entry for the Stream Log

```
OutputLog
{
    ModuleName = LOG
    Module
    {
        ITO
        {
            FilePath = /ifw/log/stream
            FilePrefix = stream_
            FileSuffix = .log
            LogLevel = normal
            SuppressErrors
            {
                ERR_SPEC_VERSION_INVALID
                ERR_RELEASE_VERSION_INVALID
            }
        }
    }
}
```

Semaphores

Table 9-10 lists the LOG Semaphores.

Table 9-10 LOG Semaphores

Entry	Description
FileName	Specifies the name of the log file. When you change the file name, the current log file is closed and renamed to file name plus timestamp. For example, the **process.log** file would be renamed **process_20030916130000.log**.
LogLevel	Specifies the minimum severity limit. The module logs all messages greater or equal to the limit. For example, enter **minor** to log only minor, major, and critical error messages. Values are: • **critical** • **major** • **minor** • **warning** • **normal** • **debug**
ShowOriginator	Specifies whether to write the name of the module that emitted the message to the log file. **True** specifies to log the module name. This helps Oracle Technical Support troubleshoot any problems. **False** specifies to not log the module name.
SuppressErrors	Specifies any error messages to exclude from the log file. For example, enter ERR_GETTING_DATADESCR to prevent those error messages from being logged.
WriteMessageKey	Specifies whether the module logs error codes. For example: ERR_FILE_NOT_FOUND. **True** specifies to write both the error code and error message to the log file. This helps Oracle Technical Support troubleshoot any problems. **False** specifies to write only the error message to the log file.

Sample Semaphores

```
ifw.ProcessLog.Module.ITO.FileName = process
ifw.ProcessLog.Module.ITO.LogLevel = minor
ifw.ProcessLog.Module.ITO.SupressErrors = ERR_GETTING_DATADESCR
```

Input Controller

Use the Input Controller to manage the input streams for its associated pipeline.

The Input Controller performs the following functions:

- Combines multiple CDR files into one transaction when configured to do so.
- Notifies the Transaction Manager (TAM) when a transaction begins.

You configure the Input Controller by editing the **Input** section of the registry file. For more information, see "Configuring the Input Section in the Registry" in *BRM Setting Up Pipeline Rating and Discounting*.

Registry Entries

Table 9-11 lists the Input Controller registry entries.

Table 9-11 Input Controller Registry Entries

Entry	Description	Mandatory
InputModule	Subgroup for the Input module. See "INP_GenericStream".	Yes
UnitsPerTransaction	Specifies the number of CDR input files that make up a transaction. By default, each CDR file forms its own transaction. This parameter only affects processing within the pipeline, and the number of output files match the number of CDR input files. The default is **1**.	No

Sample Registry

```
Input
{
    UnitsPerTransaction = 2
    InputModule
    {
        ModuleName = INP_GenericStream
        Module
        {
            ...
        }
    }
}
```

NET_EM

The NET_EM module hosts an External Module (EM). This allows the NET_EM module to use the BRM API opcodes to transfer data between real-time rating opcodes and the real-time rerating, discounting, and zoning pipelines.

Registry Entries

Table 9-12 lists the NET_EM registry entries.

Table 9-12 NET_EM Registry Entries

Entry	Description	Mandatory
FieldName	Use this entry for real-time rerating. Specifies the field in the event flist to be used to route the event. By using the "**.**" notation, you can specify a field at any level in the flist. The field identified by FieldName must be of type POID or String.	No
FieldValue	Use this entry for real-time rerating. Specifies the value of the field identified by **FieldName**.	No
NumberOfRTPipelines	Number of real-time pipelines. **Note:** This number must match the **NumberOfInstances** entry.	Yes

Table 9-12 (Cont.) NET_EM Registry Entries

Entry	Description	Mandatory
OpcodeName	Specifies the opcode sending the event to NET_EM. For real-time discounting, use: PCM_OP_RATE_DISCOUNT_EVENT For real-time zoning, use: PCM_OP_RATE_GET_ZONEMAP_INFO For real-time rerating, use: PCM_OP_RATE_PIPELINE_EVENT	Yes
PipelineName	The pipeline to route the input to. Each real-time pipeline must have a unique name.	Yes
Port	Specifies the port number of the host machine running the NET_EM module.	Yes
Threads	Set this entry to the number of threads being managed by the NET_EM module. For example, if you have two pipelines, set this entry to 2.	Yes
UnixSockFile	Specifies the UNIX Sock file when the CM and the Pipeline Manager instance are running on the same machine.	Yes

Sample Registry Entry

```
DataPool
{
     RealtimePipeline
     {
          ModuleName = NET_EM
          Module
          {
               ThreadPool
               {
                    Port = 14579
                    Threads = 3
               }
               ReratingOpcode
               {
                    OpcodeName = PCM_OP_RATE_PIPELINE_EVENT
                    PipelineName = RealtimeReratingPipeline
                    NumberOfRTPipelines = 3
               }
          }
     }
}
```

Output Collection

Use the Output Collection module to handle output streams. See "About Configuring the Output Section in the Registry" in *BRM Setting Up Pipeline Rating and Discounting* and "Configuring EDR Output Processing" in *BRM Setting Up Pipeline Rating and Discounting*.

Registry Entries

The only registry entries for the Output Collection configuration are the sections for each output stream, for example, OUT_DevNull, OUT_Reject, and OUT_GenericStream.

See the following:

- OUT_Reject
- OUT_DevNull
- OUT_GenericStream
- OUT_DB
- OUT_Realtime

Sample Registry

```
#------------------------------------------------
# Output Section
#------------------------------------------------
Output
{
   WriteDefaultEdr    = False
   DeleteEmptyFile    = True
     MaxErrorRates
     {
     }
     OutputCollection
     {
     #-----------------------------------------
     # The DevNull stream
     #-----------------------------------------
     DevNull
    ...{
```

Event Messages

Table 9-13 lists the Output Collection event messages.

Table 9-13 Output Collection Event Messages

Message	Description	Send/Receive
CMD_WRITE_LOG	An entry to the pipeline log has to be created.	Receive
REQ_STREAM_NUMBER	Determination of a specific stream number chosen by the first argument Name.	Receive from Output Collection. See "Output Collection".

Output Controller

Use the Output Controller to manage the output streams for its associated pipeline.

For more information, see "About Configuring the Output Section in the Registry" in *BRM Setting Up Pipeline Rating and Discounting*.

Registry Entries

Table 9-14 lists the Output Controller registry entries.

Table 9-14 Output Controller Registry Entries

Entry	Description	Mandatory
MaxErrorRates	Subgroup for the maximum error rate entries. This section should list all error codes to monitor and their threshold values. For more information, see "Specifying the Maximum Errors Allowed in an Input File" in *BRM Setting Up Pipeline Rating and Discounting*.	Yes
MultiThreading	Subgroup to configure multithreading in output processing.	No
MultiThreading.Active	Specifies whether to enable multithreading in output processing: • **True** enables multithreading. • **False** disables multithreading. This is the default.	Yes
MultiThreading.NumberOfThreads	Specifies the number of threads the Output Controller creates to manage the output streams for its associated pipeline. For optimum results, Oracle suggests that you set the number of threads to twice the average number of streams associated with an input EDR.	Yes
MultiThreading.BatchSize	Specifies the size of the batch in terms of number of EDRs: • **0** indicates that the Output Controller does not operate in a batch mode. • A value greater than **0** indicates that the Output Controller operates in the batch mode with the batch size equal to the specified value. Oracle suggests that **BatchSize** be greater than or equal to **BlockSize** if **BlockTransfer** is set to **True**; otherwise, **BatchSize** should be equal to the size of the output buffer. The **BatchSize** value should be directly proportional to **NumberOfThreads** and inversely proportional to the EDR enrichment rate.	Yes
OutputCollection	Subgroup for the Output Collection module entries. See: • About Configuring the Output Section in the Registry in *BRM Setting Up Pipeline Rating and Discounting* • Output Collection	Yes
OutputLog	Subgroup for the stream log entries.	Yes
SequenceGeneration	Specifies whether the pipeline generates one output file per CDR input file or one output file for an entire transaction. • **Unit** generates one output file per CDR input file. • **Transaction** generates one output file per transaction. For example, if you combine 5 CDR input files into one transaction, the pipeline creates only 1 output file. The default is **Unit**.	No
Sequencer	Specifies the name of the Sequencer for performing sequence checking. This Sequencer must be defined in the **SequencerPool** section of the registry file.	No

Table 9-14 (Cont.) Output Controller Registry Entries

Entry	Description	Mandatory
Statistic	Subgroup to control the statistics related to Pipeline Manager's EDR processing rate. You can view these statistics in the output logs and HTTP browser. See "About Configuring Statistics Information in the Output Section" in *BRM Setting Up Pipeline Rating and Discounting*.	No

Sample Registry Entry for the Multithreaded Mode

```
Output
{
    MaxErrorRates
    {
        ERR_CUST_NOT_FOUND = 10
    }

   MultiThreading
   {
       Active = True
       NumberOfThreads = 5
       BatchSize = 500
    }
    Statistic
    {
        EdrCountCriteria = ALL
    }...
    OutputCollection
      ...
    OutputLog
    {
         ...
    }
    SequenceGeneration = Unit
    Sequencer = SequenceCheck1
    ...
}
```

Sample Registry Entry for the Single-Threaded Mode

```
Output
{
    MaxErrorRates
    {
        ERR_CUST_NOT_FOUND = 10
    }

    Statistic
    {
        EdrCountCriteria = ALL
    }...
    OutputCollection
      ...
    OutputLog
    {
         ...
```

```
    }
    SequenceGeneration = Unit
    Sequencer = SequenceCheck1
    ...
}
```

ParallelLoadManager

Use the ParallelLoadManager module to load your pipelines, data modules, and function modules in parallel.

Registry Entries

Table 9-15 lists the ParallelLoadManager registry entries.

Table 9-15 ParallelLoadManager Registry Entries

Entry	Description	Mandatory
Active	Specifies whether to load the pipelines, data modules, and function modules in parallel. • **TRUE** specifies to use parallel loading. • **FALSE** specifies to use sequential loading. If the entry is missing, parallel loading is disabled.	Yes
NumberOfThreads	Specifies the number of threads Pipeline Manager uses to load your pipelines, data modules, and function modules.	Yes

Sample Registry

```
ifw
{
   ...
   ParallelLoadManager
   {
      Active = TRUE
      NumberOfThreads = 4
   }
   …
}
```

Pipeline Controller

Use the Pipeline Controller to control its associated pipeline.

Registry Entries

Table 9-16 lists the Pipeline Controller registry entries.

Table 9-16 Pipeline Controller Registry Entries

Entry	Description	Mandatory
Active	Activates or deactivates processing in the pipeline. • **True** activates pipeline processing. • **False** deactivates pipeline processing.	Yes
CountryCode	Specifies the valid country code for this pipeline. The default is **49** for Germany.	No
DataDescription	Speicifes the stream format description and mapping files See: • Configuring the Input DataDescription Registry Section in *Setting Up Pipeline Rating and Discounting*. • Configuring the Output DataDescription Registry Section in *Setting Up Pipeline Rating and Discounting*.	Yes
EdrFactory	Subgroup for the EDR Factory.	Yes
Functions	Subgroup for the function pool entries.	Yes
Input	Subgroup for the Input Controller. See: • Configuring the Input Section in the Registry in *Setting Up Pipeline Rating and Discounting*. • Input Controller.	Yes
Instances	Specifies multiple instances of a specific pipeline.	No
InternationalAccessCode	Specifies the international dial prefix. The default is **00** for Germany. **Note:** You can list multiple access codes by using a comma as a delimiter. For example: **00,001,002**.	No
InternationalAccessCodeSign	Specifies the international access code sign. The default is **+**.	No
InputBuffer	Subgroup that contains the entries for the buffer between the input module and function modules.	Yes
MobileCountryCode	Specifies the valid mobile country code for this pipeline. The default is **262** for Germany.	No
MultiThreaded	Specifies whether the pipeline uses multithreaded or single-threaded processing. The default is **True**. • **True** specifies multithreaded processing. • **False** specifies single-threaded processing.	No
NationalAccessCode	Specifies the dial prefix for national calls. The default is **0** for Germany.	No
NetworkDestinationCode	Specifies the network destination code, which identifies the home network for roaming calls. The default is **172** for D2.	No
NoOutputUsed	Specifies whether to load the Output module. Set this entry to **True** only when you are using single-threaded processing and the Input and Output modules are combined into one module. The default is **False**. • **True** specifies to *not* load the Output module. • **False** specifies to load the Output module.	No
Output	Subgroup that contains Output Controller entries. See: • About Configuring Output Processing in *Setting Up Pipeline Rating and Discounting* • Output Controller.	Yes

Table 9-16 (Cont.) Pipeline Controller Registry Entries

Entry	Description	Mandatory
OutputBuffer	Subgroup that contains the entries for the buffer between the function modules and the output module.	Yes
Pipeline_Name	Name of the pipeline.	Yes
PipelineLog	Subgroup that contains pipeline log entries.	Yes
RejectStream	Specifies the name of the rejection stream. This stream must be defined in the output module. See: • Configuring Standard Recycling in *BRM Suspending and Recycling Pipeline EDRs*. • Recycling EDRs in Pipeline-Only Systems in *BRM Suspending and Recycling Pipeline EDRs*.	Yes
TransactionManager	Specifies the subsection for the Transaction Manager.	N/A

Sample Registry

```
Pipelines
{
    Pipeline01
    {
        Active = TRUE
        MultiThreaded = TRUE
        CountryCode = 49
        MobileCountryCode = 262
        NationalAccessCode = 0
        InternationalAccessCode = 00
        InternationalAccessCodeSign = +
        NetworkDestinationCode = 171
        RejectStream = XXX
        PipelineLog
        {
            ModuleName = LOG
            Module
            {
            }
        }
        InputBuffer
        {
            Size = 1000
        }
        OutputBuffer
        {
            Size = 1000
        }
        Input
        {
            InputModule
            {
                ModuleName = XXX
                ModuleStart = XXX
                Module
                {
                    ...
                }
```

```
    }
} // Input
Functions
{
    FCI
    {
        FunctionPool
        {
            Function01
            {
                ...
            }
            Function02
            {
                ...
            }
        }
    }
} // Functions
Output
{
    ...
    Outputcollection
    {
        Output1
        {
            ModuleName = XXX
            ModuleStart = XXX
            Module
            {
                ...
            }
        }
        RejectOutput
        {
            ModuleName = XXX
            ModuleStart = XXX
            Module
            {
                ...
            }
        }
        Output2
        {
            ModuleName = XXX
            ModuleStart = XXX
            Module
            {
                ...
            }
        }
        DevNull
        {
            ModuleName = XXX
            ModuleStart = XXX
            Module
            {
                ...
            }
        }
        ...
    }
```

```
                        OutputLog
                        {
                            ...
                        }
                        Sequencer
                        {
                            ...
                        }
                    } #Output
                } #Pipeline01
            } #Pipelines
```

Sample Registry for Multiple Instances of a Pipeline

This sample shows how to configure multiple instances of a pipeline.

```
ifw
{
...
  Pipelines
  {
    Instances
    {
      RealtimeReratingPipelineGPRS
      {
        NumberOfInstances = 3
        InstanceSpecificRegistries
        {
          Entry1 = TransactionManager.BinaryLogFileName
          Entry2 = PipelineLog.Module.ITO.FileName
          Entry3 = OutputLog.FileName
          Entry4 = Functions.Standard.FunctionPool.EdrDump_Initial.Module.FileName
          Entry4 = Functions.Standard.FunctionPool.EdrDump_PostDiscounting.Module.FileName
          Entry5 = Functions.Standard.FunctionPool.EdrDump_PreDiscounting.Module.FileName
          Entry6 = Functions.Standard.FunctionPool.EdrDump_PostRating.Module.FileName
          Entry7 = Functions.Standard.FunctionPool.EdrDump_PreRating.Module.FileName
        }
...
```

Semaphore Entries

Table 9-17 lists the Pipeline Controller Semaphore entries.

Table 9-17 Pipeline Controller Semaphore Entries

Entry	Description
Active	Activates or deactivates processing in the pipeline.

Sample Semaphore Entry

```
ifw.Pipelines.ALL_RATE.Active = True
```

Event Messages

Table 9-18 lists the Pipeline Controller event messages.

Table 9-18 Pipeline Controller Event Messages

Message	Trigger	Parameter
EVT_PIPELINE_START	The pipeline was started.	Pipeline name from the registry.
EVT_PIPELINE_STOP	The pipeline was stopped.	Pipeline name from the registry.

Sequencer

Use the Sequencer to prevent Pipeline Manager from processing the same CDR file twice and to add tracking information to output streams.

Dependencies

When you configure the Sequencer to store state and log data in database tables, this module requires a connection to the Pipeline Manager database.

To assign a Sequencer to a pipeline, you must also configure the **Output** section of the registry file:

- To assign a sequence checker to a pipeline, use the **Sequencer** registry entry in the Output Controller module. For information, see "Output Controller".
- To assign a sequence generator to a pipeline, use the **Sequencer** registry entry in the output module. For information, see "OUT_GenericStream".

Registry Entries

Table 9-19 lists the Sequencer registry entries.

Table 9-19 Sequencer Registry Entries

Entry	Description	Mandatory
SequencerInstance	Specifies the name of the Sequencer instance.	Yes
Source	Specifies whether Sequencer state and log data are stored in files or in database tables. Values are: • **File** • **Database**	Yes
Controller	Subgroup that contains Controller entries	Yes
Controller.SequencerType	Specifies whether the Sequencer performs sequence checking or sequence generation: • **Check** configures the Sequencer to perform sequence checking. • **Generation** configures the Sequencer to perform sequence generation.	Yes
Controller.DatabaseConnection	Specifies a connection to the Pipeline Manager database.	Yes, only if **Source** = **Database**.

Table 9-19 (Cont.) Sequencer Registry Entries

Entry	Description	Mandatory
Controller.ReuseGap	Specifies whether the Sequencer assigns skipped sequence numbers to output files. • **True** directs the Sequencer to reuse skipped sequence numbers by assigning the skipped sequence numbers to other CDRs. • **False** directs the Sequencer to never reuse skipped sequence numbers. The default is **False**.	No
Controller.SequenceLength	Specifies the length of the incoming CDR file's sequence number. The default is **6**.	No
Controller.FileName	Specifies the name of the Sequencer state file. This file stores state information for one Sequencer instance. **Important:** You must create one state file for each Sequencer instance. Otherwise, the Sequencer fails.	Yes, only if **Source** = **File**
Controller.FilePath	Specifies the path to the Sequencer state file.	Yes, only if **Source** = **File**.
Controller.Log	Subgroup that contains Sequencer log file entries.	Yes
Controller.Log.FileName	Specifies the name of the Sequencer log file. **Important:** You must create one Sequencer log file for each Sequencer instance. Otherwise, the Sequencer fails.	Yes, only if **Source** = **File**
Controller.Log.FilePath	Specifies the path to the Sequencer log file.	Yes, only if **Source** = **File**
Controller.UseGapAtStartup	Specifies whether to add a gap for the skipped sequence numbers starting from 0. This entry is required only when the **SequencerType** field is **Check** and the **ReuseGap** field is **True**. You can use this entry even if you have set the **Seq Original Number** field to **0**. • **True**. This value directs the Sequencer to add a gap for the skipped sequence numbers starting from 0. • **False**. The default value. This value directs the Sequencer to never add a gap for the skipped sequence numbers.	No

Sample Registry for File Storage

```
SequencerPool
{
    SequencerInstance
    {
        Source = File

        Controller
        {
            SequencerType = Check
            ReuseGap = True
            SequenceLength = 7
            FileName = sequence.dat
            FilePath = /opt/portal/ifw/sequencer

            Log
            {
                FileName = sequence.log
```

```
                FilePath = /opt/portal/ifw/logs
            }
        }
    }
}
```

Sample Registry Entry for Database Storage

```
SequencerPool
{
    SequencerInstance
    {
        Source = Database

        Controller
        {
            SequencerType = Generation
            DatabaseConnection = DatabaseModule
            ReuseGap = False
            SequenceLength = 10
        }
    }
}
```

Database Tables

The Sequencer uses the following tables:

- IFW_PIPELINE
- IFW_SEQCHECK
- IFW_SEQLOG_IN
- IFW_SEQLOG_OUT

Transaction Manager

Use the Transaction Manager to coordinate the state of all transactional modules and components in one pipeline.

Dependencies

Requires a reference to the Transaction ID Controller. For information, see "Transaction ID Controller".

Registry Entries

Table 9-20 lists the Transaction Manager registry entries.

Table 9-20 Transaction Manager Registry Entries

Entry	Description	Mandatory?
BinaryLogFileName	Specifies the path and file name of the binary log file, which is used to persist and restore open transactions. **Important:** If you use multiple pipelines, you cannot use the same file for different pipelines.	Yes
RedoEnabled	Specifies whether the redo mechanism is enabled. **True** enables the redo mechanism. **False** disables the redo mechanism.	Yes
SingleTransaction	Specifies whether only one pipeline transaction is allowed at a time. **True** specifies that only one pipeline transaction can be active at one time. The TAM blocks any new transactions from starting while a transaction is in progress. **False** specifies that multiple pipeline transactions can be active at one time.	Yes
WriteToLogEnabled	Specifies whether the Transaction Manager writes status information to the pipeline log file. **True** enables writing to the pipeline log file. **False** disables writing to the pipeline log file.	No

Sample Registry

```
Pipelines
{
    PipelineName
    {
        TransactionManager
        {
            RedoEnabled = True
            SingleTransaction = True
            BinaryLogFileName = ./
            WriteToLogEnabled = False
        }
    }
}
```

Transaction ID Database Generator

Use the Transaction ID Database Generator to store transaction IDs in database tables.

Dependencies

Requires a connection to the Pipeline Manager database.

Registry Entries

Table 9-21 lists the Transaction ID Database Generator registry entries.

Table 9-21 Transaction ID Database Generator Registry Entry

Entry	Description	Mandatory?
DataConnection	Specifies a connection to the Pipeline Manager database.	Yes

Sample Registry

```
TransactionIdController
{
    Source = Database
    Generator
    {
        DataConnection = ifw.DataPool.Login
    }
}
```

Database Tables

The TAM_TransIdDbGenerator module uses the IFW_TAM database table.

Transaction ID File Generator

Use the Transaction ID File Generator to store transaction IDs in a file.

Registry Entries

Table 9-22 lists the Transaction ID File Generator registry entries.

Table 9-22 Transaction ID File Generator Registry Entries

Entry	Description	Mandatory?
FileName	Specifies the path and file name of the Transaction ID Controller state file.	Yes
Increment	Specifies the number of transaction IDs that are cached.	Yes

Sample Registry

```
TransactionIdController
{
    Source = File
    Generator
    {
        FileName = /data/system/info/transIdInfo.dat
        Increment = 10
    }
}
```

Transaction ID Controller

Use the Transaction ID Controller to generate transaction IDs for all pipelines.

Registry Entries

Table 9-23 lists the Transaction ID Controller registry entries.

Table 9-23 Transaction ID Controller Registry Entries

Entry	Description	Mandatory?
Generator	Subgroup for the generator entries. See: • Transaction ID File Generator • Transaction ID Database Generator	Yes
Source	Specifies whether the Transaction ID Controller stores transaction IDs in files or database tables. Values are: • **File** • **Database**	Yes

Sample Registry for File Storage

```
TransactionIdController
{
    Source = File
    Generator
    {
        FileName = /data/system/info/transIdInfo.dat
        Increment = 10
    }
}
```

Sample Registry for Database Storage

```
TransactionIdController
{
    Source = Database
    Generator
    {
        DataConnection = ifw.DataPool.Login
    }
}
```

10

Pipeline Manager Utilities

Learn about the Oracle Communications Billing and Revenue Management (BRM) Pipeline Manager utilities.

Topics in this document:

Database Loader

The **Database Loader** utility loads and unloads aggregation data into and from a database.

For information about aggregation, see "Setting Up Pipeline Aggregation" in *BRM Setting Up Pipeline Rating and Discounting*.

Dependencies

This utility needs a connection to the DBC database module and the DBL library (**libDBLXXX.so**). See "Database Connect (DBC)".

Location

*Pipeline_home***/tools**

where *Pipeline_home* is the directory in which you installed Pipeline Manager.

Syntax

```
dbLoader -r registry [-f files] [-u]
```

Parameters

-r *registry*
Defines the registry file.

-f *files*
Defines the file pattern (regular expression).

-u
Undo mode.

Registry Entries

Table 10-1 lists the Database Loader registry entries.

Table 10-1 Database Loader Registry Entries

Entry	Description	Mandatory
BULKSIZE	Specifies the Oracle array size for bulk inserts (loadmode 2 and 3).	Yes
DIRECTIONMODE	Defines the selection order of the control files (1 file name, 2 sequence).	Yes
FILES.ARCHIVE.PATH	Specifies the path where the successfully loaded files are stored.	Yes
FILES.ARCHIVE.SUFFIX	Specifies the suffix of the successfully loaded data files.	Yes
FILES.BAD.PATH	Specifies the path where the bad files are stored.	Yes
FILES.BAD.SUFFIX	Specifies the suffix of the bad data files.	Yes
FILES.CONTROL.PATH	Specifies the path for the input aggregate control files.	Yes
FILES.CONTROL.SUFFIX	Specifies the suffix of the input aggregate control files.	Yes
FILES.DATA.PATH	Specifies the path for the input aggregate data files.	Yes
FILES.DATA.SUFFIX	Specifies the suffix of the input aggregate data files.	Yes
FILES.MERGE.PATH	Specifies the path where the source merge data files are stored.	Yes
FILES.MERGE.SUFFIX	Specifies the suffix of the source data files before merging/ sorting.	Yes
FILES.REJECT.PATH	Specifies the path where the rejected files are stored.	Yes
FILES.REJECT.SUFFIX	Specifies the suffix of the rejected data files.	Yes
LOADMODE	Specifies how to load data: • **1**: Single row updates and inserts. • **2**: Single row updates and bulk inserts. • **3**: Single row updates and bulk inserts. Before loading, the files can be merged or sorted and split into smaller pieces. Undo mode is always 1.	Yes
MAXSPLITLINES	Specifies the maximum number of lines per data file after splitting (loadmode 3).	Yes

Table 10-1 (Cont.) Database Loader Registry Entries

Entry	Description	Mandatory
ROLLBACKSEGMENT	Specifies which Oracle rollback segment to use when loading the database. How to set this entry depends on your database software setup. If your Oracle9i database uses automatic undo management, comment out or remove this registry entry. If your database does not use undo management, specify a rollback segment. The Oracle9i software provides an automatic undo management feature, which creates undo tablespaces rather than rollback segments for undo information. If you use this undo management feature *and* specify a rollback segment for the Pipeline Manager **Database Loader** utility, the utility fails when it attempts to load the database. To prevent this problem, don't specify a rollback segment.	No
SORTCMD	Specifies the external sort command (loadmode 3).	Yes
SORTING	Specifies a flag if files of identical structure should me merged and sorted (loadmode 3).	Yes
SORTMAXFILESIZE	Specifies the maximum destination size of the merged and sorted files (loadmode 3).	Yes
SORTTMPDIR	Specifies the path where sort stores temporary files (loadmode 3).	Yes
SPLITTING	Specifies whether to split data files before loading (reduce transaction size) (loadmode 3).	Yes

Sample Registry

```
DBLOADER
  {
    Active              = TRUE
    ProcessLoopTimeout  = 10
    QueueRequestTimeout = 0
    Instrumentation
    {
      #------------------------------------------------------------
      # ProbeBroker registry entries.
      # ProbeInfoFilePath - The path that contains all probe
      # info files used by instrumented objects.
      #------------------------------------------------------------
      ProbeBroker
      {
        ProbeInfoFilePath = ./instrumentation
      }
    }
    LogMessageTable
    {
      MessageFilePath   = ./etc
      MessageFileSuffix = .msg
    }
    DiagnosticDataHandler
    {
      DiagnosticFilePath = ./log
      DiagnosticFileName = diagnostic.dat
    }
    #
```

```
# main parameter
#
DIRECTIONMODE   = 2
LOADMODE        = 2
BULKSIZE        = 100
ROLLBACKSEGMENT = R04
SORTING         = true
SORTCMD         = sort
SORTTMPDIR      = .
SORTMAXFILESIZE = 2000000000
SPLITTING       = true
MAXSPLITLINES   = 40000
#
# database section
#
DataPool
{
  Database
  {
    ModuleName = DBC
    Module
    {
      DatabaseName = $ORACLE_SID
      UserName     = AGGREGATOR
      PassWord     = password
      AccessLib    = oci231
      Connections  = 1
    }
  }
}
#
# File Section
#
FILES
{
  CONTROL
  {
    PATH    = ./data/aggregate/cntl
    SUFFIX  = .ctl
  }
  DATA
  {
    PATH    = ./data/aggregate/done
    SUFFIX  = .dat
  }
  REJECT
  {
    PATH      = ./data/aggregate/reject
    SUFFIX    = .rej
    THRESHOLD = 85
  }
  REJECT_HANDLE
  {
    PATH      = ./data/aggregate/reject
    SUFFIX    = .rej
    THRESHOLD = 85
  }
  ARCHIVE
  {
    PATH    = ./data/aggregate/archive
    SUFFIX  = .arc
  }
```

```
    BAD
    {
      PATH    = ./data/aggregate/bad
      SUFFIX  = .bad
    }
    MERGE
    {
      PATH    = ./data/aggregate/merge
      SUFFIX  = .mrg
    }
  }
  #
  # log section
  #
  ProcessLog
  {
    ModuleName = LOG
    Module
    {
      ITO
      {
        MessageFilePath = etc
        MessageFilePrefix = error
        MessageFileSuffix = error.msg
        FilePath = ./data/aggregate/log
        FileName = process
        FilePrefix = DBL_
        FileSuffix = .log
        ProcessName = dbLoader
        MessageGroup = DBLOADER
      }
      Buffer
      {
        Size = 1000
      }
    }
  }
}
```

db2irules.pl

Use the **db2irules.pl** script to extract rule sets from the Pipeline Manager database into the Rule Set XML file.

> **Note:**
>
> This utility uses DBI and DBD drivers that are not part of the Pipeline Manager installation. You download these drivers from https://www.cpan.org and compile and install them separately.

Location

Pipeline_home/**tools/IRules2Db/db2irules.pl**

> **Note:**
>
> Since there are dependencies between the **db2irules.pl** script and the **PerlParser.pm** XML library located in the same directory as the script, always run the script from this location.

Syntax

```
db2irules.pl [-d] [-u] dbi:dcs file_path rule_set_id
```

Parameters

If you start the **db2irules.pl** script without any parameters, a usage description and an example for each parameter are displayed.

-d
Deletes the specified rule sets from the database after you have extracted them. If you use this parameter, a transaction is opened with the database. If any of the rule set deletes fail, the entire delete sequence is rolled back to preserve database integrity. If all rule set tables are deleted successfully, the transaction is committed to the database.

-u
Creates a unique file name for the rule set, based on date and time. It uses the following format: *RULESET_yyyy-mm-dd_hh-mm-ss.xml*. Use this parameter to ensure that you do not override an existing XML file when extracting rule sets. If the file name for a rule set contains spaces, replace them with the underscore character (_).
Example:

```
db2irules.pl -u dbi:Oracle:orcl TAP3_VAL
```

dbi*:*dcs
Specifies the database connection string. This required parameter enables the script to access the database. The string is different for each database type. Example dcs for Oracle:

```
Oracle:orcl
```

> **Note:**
>
> The database connection string is the standard database access module for Perl scripts. It defines a set of methods, variables, and conventions that provide a consistent database interface, independent of the actual database being used.

file_path
Specifies where you want to export the rule set. If you want to use the same directory in which the rule set is stored, use **./** as file path. If you don't set this parameter, the rule set is exported automatically to the current directory.

rule_set_id
Extracts only one specific rule set, which is identified by its unique ID. If you don't set this parameter, the **db2irules.pl** script will extract all rule sets from the database. If you use this parameter, you must use the *file_path* parameter. This *rule_set_id* refers to the IFW_RULESET.RULESET database field.

Diagnostic Data Handler

Use Diagnostic Data Handler to get data about Pipeline Manager after a crash, exception, critical error, or while it is running.

Registry Entries

Table 10-2 lists the Diagnostic Data Handler registry entries.

Table 10-2 Diagnostic Data Handler Registry Entries

Entry	Description	Mandatory
DiagnosticFilePath	Path to the log file that is created by Diagnostic Data Handler.	Yes
DiagnosticFileName	File name of the log file that is created by Diagnostic Data Handler.	Yes

Sample Registry

```
DiagnosticDataHandler
{
   DiagnosticFilePath = ./log
   DiagnosticFileName = diagnostic.dat
}
```

irules2db.pl

Use the **irules2db.pl** script to insert a rule set from the Validation Rules XML file into the Pipeline Manager database.

> **Note:**
>
> This utility uses DBI and DBD drivers which are not part of the Pipeline Manager installation. You download these drivers from https://www.cpan.org and compile and install them separately.

Location

*Pipeline_home/***tools/IRules2Db/irules2db.pl**

> **Note:**
>
> Since there are dependencies between the **irules2db.pl** script and the **PerlParser.pm** XML library which is located in the same directory as the script. Always run the script from this location.

Syntax

```
irules2db.pl [-f] dbi:dcs rule_set_name backup_file_path
```

Parameters

If you start the **irules2db.pl** script without any parameters, a usage description and an example for each parameter are displayed.

dcs
The database connection string. This required parameter enables the script to access the database. The string is different for each database type. Example dcs for Oracle:

```
Oracle:orcl
```

> **Note:**
>
> The database connection string is the standard database access module for Perl scripts. It defines a set of methods, variables, and conventions that provide a consistent database interface, independent of the actual database being used.

rule_set_name
Use this parameter to specify the name of the Rule Set XML file that you want to import to the database. This parameter supports fully qualified and relative path names.
Examples:

- **./tap3_val.xml**
- **/home/data/tap3_val.xml**
- **/../files/tap3_val.xml**
- **tap3_val.xml**

backup_file_path
Use this parameter to specify the path for storing the extracted rule set before it is deleted from the database and then after modification inserted from the Rule Set XML file into the database. Use this parameter with the **-f** parameter.

-f
This parameter forces the rule set into the database. The **irules2db.pl** script connects to the database and starts parsing the Rule Set XML file. When it finds the name of the rule set, it calls the export script that contains the **-u** and **-d** parameters. If the **db2irules.pl** script finished successfully, the **irules2db.pl** script continues parsing the XML file and imports the rule set to the database. If any of the rule set columns fail to be inserted, the **irules2db.pl** script rolls back the transaction and exits. If all columns are inserted into the database successfully, the rule set for the transaction is committed.

load_notification_event

Loads the notification XML file from Pipeline Manager into the BRM database. This allows BRM to notify customers when their balance has reached a threshold value during the batch rating process.

You must configure the Batch Controller to run this utility.

Location

BRM_home/**bin**

where, *BRM_home* is the directory in which you installed the BRM software.

Syntax

```
load_notification_event [-d] [-v] [-h] XML_file
```

Parameters

-d
Sets the log level to debug and outputs debug information into the log file for this process. If not set, only error-level information is output.

-v
Displays information about failed or successful processing as the utility runs.

-h
Displays syntax and parameters for this utility.

XML_file
The name and location of the XML file to load into the BRM database. This must be the last parameter listed on the command line.

Results

This utility notifies you when it successfully loads the XML file.

If the utility does not notify you that it was successful, look in the utility log file (**default.pinlog**) to find any errors. The log file is either in the directory from which the utility was started or in a directory specified in the configuration file.

load_pin_rtp_trim_flist

Use this utility to specify account and service object fields to be included in the flist sent to a real-time rerating, discounting, or zoning pipeline. The main uses for this utility include:

- Improving system efficiency by removing (trimming) fields that Pipeline Manager doesn't use.
- Supporting custom iScripts and iRules in the real-time pipeline by adding fields to flists which are not included by default.

You can configure a different set of fields to be included in the flist based on event type.

Account object fields are included in the PIN_FLD_INHERITED_INFO substruct in the flist. Service object fields are included in the PIN_FLD_INHERITED_INFO.PIN_FLD_SERVICE_INFO substruct.

> **Note:**
>
> - You cannot load separate **/config/rtp/trim_flist** objects for each brand. All brands use the same object.
> - You can't remove fields from the PIN_FLD_INHERITED_INFO substruct or the subordinate PIN_FLD_INHERITED_INFO.PIN_FLD_SERVICE_INFO substruct.

You specify the list of required fields in an XML file (*field_list*.**xml**) and then load the file using the utility.

> **Note:**
>
> - If you use the utility to add new fields to the flist, you must update the input modules of the all pipelines to add the fields to the EDR container.
> - After you use the utility, you must restart BRM.

Location

BRM_home/**bin**

Syntax

```
load_pin_rtp_trim_flist -f field_list.xml [-v] [-d]
```

Parameters

-f *field_list*.xml
Specifies the XML file that describes which fields should be read. For a sample flist, see *BRM_home*/**sys/data/config/pin_config_rtp_trim_flist.xml**.

-v
Displays information about successful or failed processing as the utility runs.

-d
Creates a log file for debugging purposes. Use this parameter for debugging when the utility appears to have run with no errors, but the data has not been loaded into the database.

LoadIfwConfig

Use this utility to extract data from or load data into the Pipeline Manager database. This enables you to:

- Migrate data from a legacy database to the Pipeline Manager database.
- Transfer data between Pipeline Manager databases; for example, from a test database to a production database. See "Transferring Data Between Pipeline Manager Databases" in *BRM Setting Up Pipeline Rating and Discounting*.

> **Note:**
>
> The 7.4 version of the **LoadIfwConfig** utility is not backwards-compatible with previous versions of the utility. Any data exported by a previous version of the utility must also be loaded with that same version. In addition, any custom scripts or procedures that are dependent on the utility's functionality might need to be modified to work with the 7.4 version.

The **LoadIfwConfig** utility can run in these modes:

- *Non-interactive mode*: You use commands that batch several related parts of the extracting or loading process. You must enter a full command, including the utility name for each set of actions.
- *Interactive mode*: You issue a command for each step in the process of extracting or loading. After you enter interactive mode, the prompt changes to an angle bracket and commands are single words for performing particular actions. You can view a list of the change sets that will be extracted or loaded.

Location

Pipeline_home/**bin**

Syntax: Non-Interactive Mode

```
LoadIfwConfig  {-rall [-t Modifidate] | -r [-t Modifidate] | -p [f] | -u | -I}
               [-c] [-nodep] -i InputFile [-o OutputFile] [-h] [-v]
```

Parameters: Non-Interactive Mode

-rall [-t *Modifidate*]
Extracts all objects from the Pipeline Manager database. This parameter does not require an input XML file.
Using **-t** *Modifidate* retrieves only pricing objects that were modified after the specified timestamp. Enter the time in the ISO-8601 format: *YYYY*-*MM*-*DD*T*hh*:*mm*:*ss* or *YYYY*-*MM*-*DD* with the server time zone as the default.

-r [-t *Modifidate*]
Extracts from the database the objects listed in *InputFile*.
Using **-t** *Modifidate* retrieves only pricing objects that were modified after the specified time. Enter the time in the ISO-8601 format: *YYYY*-*MM*-*DD*T*hh*:*mm*:*ss* or *YYYY*-*MM*-*DD* with the server time zone as the default.

-p [f]
Deletes objects from the database.
Using the **f** parameter turns off the delete confirmation.

-u
Updates the Pipeline Manager database. Data is not actually updated in the database until it is committed with the **-c** parameter.

-I
Inserts data into the Pipeline Manager database. Data is not actually inserted into the database until it is committed with the **-c** parameter.

-c
Commits the data to the database. You use this command in conjunction with the **-u** and **-l** parameters.

-nodep
Suppresses any object dependency relationships that you configured in the *Pipeline_home*/**tools/XmlLoader/CustomConfig.xml** file. This allows the utility to extract from the database only those objects that meet your criteria and to ignore any dependent objects. For more information about object dependencies, see "About Specifying to Extract Child and Dependent Objects" in *BRM Setting Up Pipeline Rating and Discounting*.

-i *InputFile*
When extracting pipeline data by using the **-r** or **-rall** parameter, this is the name of the XML file that specifies the list of objects to extract from the source Pipeline Manager database.
When loading pipeline data by using the **-u** or **-l** parameter, this is the name of the XML file that contains the data you are loading into the destination Pipeline Manager database.
When deleting pipeline data by using the **-p** parameter, this is the name of the XML file that specifies the list of objects to delete from the Pipeline Manager database.

-o *OutputFile*
Specifies the output file to which the Pipeline Manager data is extracted. By default, the utility writes the output to a file named **default.out** in the current directory.

-h
Displays help about using the utility.

-v
Displays information about successful or failed processing as the utility runs.

Syntax: Interactive Mode

```
LoadIfwConfig  [read InputFile] [write OutputFile] [retrieve_all [-t Modifidate]]
               [fetch [-t Modifidate]] [list] [delete] [commit] [update] [insert]
               [help] [nodep] [verbose on|off] [quit]
```

Parameters: Interactive Mode

read *InputFile*
Specifies to read the specified input file into internal memory.

write *OutputFile*
Specifies the output file to which the Pipeline Manager data is extracted. By default, the utility writes the output to a file named **default.out** in the current directory.

retrieve_all [-t *Modifidate*]
Extracts all objects from the Pipeline Manager database.
Using **-t** *Modifidate* retrieves only pricing objects that were modified after the specified time. Enter the time in the ISO-8601 format: *YYYY*-*MM*-*DD*T*hh*:*mm*:*ss* or *YYYY*-*MM*-*DD* with the server time zone as the default.

fetch [-t *Modifidate*]
Extracts from the database the objects listed in internal memory. You use this parameter after you use the **read** parameter.
Using **-t** *Modifidate* retrieves only pricing objects that were modified after the specified time. Enter the time in the ISO-8601 format: *YYYY*-*MM*-*DD*T*hh*:*mm*:*ss* or *YYYY*-*MM*-*DD* with the server time zone as the default.

list
Lists the current pipeline data stored in internal memory.

delete
Deletes from the database the objects listed in *InputFile*.

commit
Commits the data to the database. You use this command in conjunction with the **update** and **Insert** parameters.

update
Updates the Pipeline Manager database. Data is not actually updated in the database until it is committed with the **commit** parameter.

insert
Inserts data into the Pipeline Manager database. Data is not actually inserted into the database until it is committed with the **commit** parameter.

help
Displays help about using the utility.

nodep
Suppresses any object dependency relationships that you configured in the *Pipeline_home***/tools/XmlLoader/CustomConfig.xml** file. This allows the utility to extract only those objects that meet your criteria and to ignore any dependent objects. For more information about object dependencies, see "About Specifying to Extract Child and Dependent Objects" in *BRM Setting Up Pipeline Rating and Discounting*.

verbose [on | off]
Sets verbose information:

- **verbose on** displays the status of the command most recently run.

  Use the **ProcessLog** section of the registry file to specify the name and location of the file where debug messages are written.
- **verbose off** displays the status only if there is an error.

quit
Quits from the utility.

Results

If the **LoadIfwConfig** utility is successful, it displays a confirmation message. If unsuccessful, it displays errors.

Memory Monitor

Use the Memory Monitor module to warn you when available system memory is low and to shut down Pipeline Manager when memory reaches a specified threshold.

Registry Entries

Table 10-3 lists the Memory Monitor registry entries.

Table 10-3 Memory Monitor Registry Entries

Entry	Description	Mandatory
ScaleUnit	Specifies the unit for monitoring memory. • **P** specifies percentage. • **K** specifies Kilobytes. • **M** specifies MegaBytes.	Yes
ShutdownFreeMemLimit	Specifies the amount or percentage of remaining system memory that triggers Pipeline Manager to gracefully shut down. **Note:** For percentage, you must enter a value from 1 to 99 inclusive.	Yes
WarningFreeMemLimit	Specifies the amount or percentage of remaining system memory that triggers Pipeline Manager to issue a warning to the user. **Note:** For percentage, you must enter a value from 1 to 99 inclusive.	Yes

Sample Registry

```
ifw
{
    MemoryMonitor
    {
        ScaleUnit = P
        WarningFreeMemLimit = 10
        ShutdownFreeMemLimit = 5
    }
}
```

pin_container_to_stream_format

Use this utility to create EDR stream, input and output mapping, and input and output grammar files from an EDR container description file. FCT_CallAssembling then uses these files in the process of converting partially assembled call records to a new container description.

Location

BRM_home/**bin**

where *BRM_home* is the directory in which you installed BRM components.

Syntax

pin_container_to_stream_format -c *container_description_filename* **-g** *grammar_file_prefix* **-m** *mapping_file_prefix* **-s** *stream_file_prefix* **| -h**

Parameters

-c *container_description_filename*
Specifies the container description file to use to generate a stream file and the mapping and grammar files. Replace *container_description_filename* with the container description file to use.

-g *grammar_file_prefix*
Creates the input and output grammar description files based on the container description file. Replace *grammar_file_prefix* with a prefix to add to the grammar filenames.

-m *mapping_file_prefix*
Creates the input and output mapping description files based on the container description file. Replace *mapping_file_prefix* with a prefix to add to the mapping filenames.

-s *stream_file_prefix*
Creates the stream description file based on the container description file. Replace *stream_file_prefix* with a prefix to add to the stream filename.

> **Note:**
>
> If you do not specify one or more of the **-g**, **-m**, or **-s** parameters, this utility generates the files using the container description filename as a prefix. However, if you specify these options, you must also specify their arguments. Otherwise this utility returns an error.

-h
Displays help for this utility.

Example

This example:

```
pin_container_to_stream_format -c containerDesc.dsc -g OLD_ -m OLD_ -s OLD_
```

Creates these files using the information in **containerDesc.dsc**:

- **OLD_Stream.dsc**
- **OLD_InGrammar.dsc**
- **OLD_OutGrammar.dsc**
- **OLD_InMap.dsc**
- **OLD_OutMap.dsc**

Results

The **pin_containter_to_stream_format** utility notifies you only if it encounters errors.

pin_recycle

Use this utility to search for failed EDRs in the BRM database and queue the EDRs for recycling or test recycling, or delete them. This utility can:

- Recycle calls from the same CDR file as part of the BRM standard recycling feature. For details, see "About the Standard Recycling Mechanism" in *BRM Suspending and Recycling Pipeline EDRs*.
- Recycle all EDRs that contain the same recycle key as part of either Suspense Manager or standard recycling. For details, see "About Recycling Suspended EDRs After Rating Interruptions" in *BRM Suspending and Recycling Pipeline EDRs*.

- Recycle all EDRs that have the same suspense reason code.

This utility calls the suspense manger opcodes to actually perform the recycling.

Location

BRM_home/**bin**

Syntax

```
pin_recycle [ -f CDR_file] [ -k recycle_key ] [ -d | -D| -r reason_code| -t ]
```

Parameters

-f *CDR_file*
Queues all the failed EDRs that arrived in a single CDR file. Pipeline Manager rates these calls as soon as it can.

-k *recycle_key*
Searches for and queues EDRs for rating that contain:

- The *recycle_key*, an application-specific string that is added to each EDR as it is suspended by Pipeline Manager. See "About Standard Recycling" in *BRM Suspending and Recycling Pipeline EDRs* for details.
- A status of **suspended**.

These EDRs are queued for rating by Pipeline Manager as soon as possible.

-d
Searches for and deletes all EDRs with a status of **succeeded** or **written off**.

-D
Searches for and deletes all EDRs with a status of **succeeded**, **written off**, or **suspended**.

-r *reason_code*
Searches for and recycles all EDRs that have the specified reason code.

-t
Specifies a test recycle. In test mode, **pin_recycle** creates a report about the processing, but does not make any changes to the database. Test results written to the directory and file you specified using the FCT_Suspense module **RecycleLog** registry entries. You must also set the FCT_Suspense **LogTestResults** registry entry for standard recycling implementations.

Results

This utility logs messages to **stdout**.

The following message is returned after you use **pin_recycle** to recycle EDRs:

```
pin_recycle tool, number_of_EDRs EDRs Submitted for Recycling
```

The following message is returned after you use **pin_recycle** to test recycle EDRs:

```
pin_recycle tool, number_of_EDRs EDRs submitted for test recycling
```

The following message is returned after you use **pin_recycle** to delete EDRs:

```
pin_recycle tool, number_of_EDRs suspended EDRs deleted
```

uninstaller

Use this utility to uninstall the BRM server software, client applications, and optional components from a single machine. If your BRM system is distributed among multiple machines, you must run the **uninstaller** utility on each machine.

This utility does not remove all BRM files and directories from your system or reverse changes made to your configuration files and database.

Location

BRM_home/**uninstaller**

Syntax

```
uninstaller  -log BRM_home/uninstaller/uninst
             [ + | - | = ]product product_name
             [ + | - | = ]component component_name product_name
             read text_file_name
```

Parameters

-log *BRM_home*/uninstaller/uninst
Logs status and error messages to the **uninst** log file.

+
Registers the product or component to uninstall.

–
Points to the product or component to uninstall.

=
Verifies that the product or component is registered for uninstallation.

Commands

- **product** *product_name*

  Uninstalls the specified product. You can only uninstall one product at a time.

  The **Infranet.prod** file, located in the directory where you downloaded and extracted your BRM software, stores the names of all products installed on your system. *product_name* must match one of the names in this file.

  For example, to uninstall BRM, type:

  ```
  % uninstaller -log BRM_home/uninstaller/uninst -product Portal_Base
  ```

- **component** *component_name product_name*

  Uninstalls the specified component. You must specify the component name and the parent product.

The **comps** directory, located in the directory where you downloaded and extracted your BRM software, lists the names of all components installed on your system. *component_name* must match one of the file names, minus the extension, in this directory.

The **Infranet.prod** file, located in the directory where you downloaded and extracted your BRM software, stores the names of all products installed on your system. *product_name* must match one of the names in this file.

For example, to remove the Connection Manager (CM) only, type:

```
% uninstaller -log BRM_home/uninstaller/uninst -component CM Portal_Base
```

- **read** *text_file_name*

  Reads the text file and performs any batch operations specified in the text file.

Results

The **uninstaller** utility doesn't notify you whether it was successful or unsuccessful. You must look in your directory structure to see if your files were removed.

purge_np_data.p

Use this utility to purge existing records from the number portability data file that are older than a specified date and time. See "Purging and Reloading the Memory Records" in *BRM Setting Up Pipeline Rating and Discounting*.

Location

*Pipeline_home/***bin**

Syntax

```
purge_np_data.pl NP_FileName TimeStamp [-b backup_filename] [-n] [-help]
```

Parameters

NP_FileName
Specifies the name of the number portability data file that will be purged.

TimeStamp
Specifies the date prior to which all the number portability records are purged. After the data is purged, the number portability data file is updated with the purged data.
Format: *YYYYMMDDhhmmss*.

-b *backup_filename*
Specifies the name of the backup file that will contain the unpurged number portability records.

-n
Sorts in the ascending order of the CLI. Default sorting is in the ascending order of the time stamp.

-help
Displays the syntax and parameters for this utility.

Results

The **purge_np_data.pl** utility notifies you when it successfully purges the number portability data file. Otherwise, it displays an error message.

RoamingConfigGen64

Use this utility to retrieve the roaming partner data from the Pipeline Manager database and create the roaming configuration data file. The data file is used by the Instances module to configure multiple instances of sequencers, output streams, or system brands based on the template sections or entries in the roaming registry file.

Location

*Pipeline_home***/bin**

Syntax

```
RoamingConfigGen64 -l database_access_library -s server_name [-d
database_name] -c operator_code [-o output_path] [-b base_path] [-h]
```

Parameters

-l database_access_library
The database access library.

-s *server_name*
Specifies the name of the host machine running the Pipeline Manager database.

-d *database_name*
Specifies the database name of the Pipeline Manager database. The default is an empty string (**' '**).

-c *operator_code*
Specifies the home network operator code. The default is **PORTL**.

-o *output_path*
Specifies the output path for the data file generated by the **RoamingConfigGen64** utility. By default, the data file is saved in the *Pipeline_home***/conf/** directory.

-b *base_path*
Specifies the base path to the directory for Transferred Account Procedure (TAP) and Near Real Time Roaming Data Exchange (NRTRDE) output files. The default path is *Pipeline_home***/data/outcollect/**.
For example, if the base path is *Pipeline_home***/data/outcollect/**, the following new subdirectories are created in the *Pipeline_home***/data/outcollect/** directory:

- **tapout/** for TAP output files
- **nrtrdeout/** for NRTRDE output files

-h
Displays the syntax and parameters for this utility.

> **Note:**
>
> When prompted, enter the database user name and password.

Example

```
RoamingConfigGen64 -l liboci10g6312d.so -s $ORACLE_SID -c EUR01
```

where:

- **liboci10g6312d.so** is the database access library.
- **$ORACLE_SID** is the database alias.
- **EUR01** is the home network operator code.

Results

The **RoamingConfigGen64** utility creates the roaming configuration data file. Otherwise, it displays an error message.

settlement_extract

Use this utility to retrieve roaming settlement information from the IC-Daily tables in the Pipeline Manager database. When Pipeline Manager rates roaming usage, it stores the amounts owed each roaming partner in the IC-Daily tables.

> **Note:**
>
> To ensure only unbilled events are extracted, before running this utility, you must close the bill run for each roaming partner account. You close the bill run by using Pricing Center or Pipeline Configuration Center (PCC).

This utility creates one file containing all settlement information stored in the Pipeline Manager database that has not already been extracted. The settlement information includes the amounts owed to each network that was used for roaming calls.

> **Note:**
>
> - This utility requires Perl version 5.004_00.
> - This utility uses DBI and DBD drivers which are not part of the Pipeline Manager installation. You download these drivers from https://www.cpan.org and compile and install them separately.
>
> For example:
>
> ```
> # setenv LD_PRELOAD /u01/app/oracle/product/817/JRE/lib/PA_RISC/
> native_threads/libjava.so
> ```

Location

BRM_home/**apps/uel**

Syntax

```
settlement_extract.pl [-u] dbi:dcs username password [filepath]
```

Parameters

-u
Creates a unique file name for the new file using the current time. The format of the file name is:
"settlement_YYYY-MM-DD_hh-mm-ss.txt"

dcs
The database connection string. This required parameter enables the script to access the database. The string is different for each database type. Example dcs for Oracle:

```
Oracle:orcl
```

> **Note:**
>
> The database connection string is the standard database access module for Perl scripts. It defines a set of methods, variables, and conventions that provide a consistent database interface, independent of the actual database being used.

username
The database username.

password
The database password.

filepath
The location where the file should be written to. If you don't include this parameter, the file is written to the current directory.

Results

Creates a roaming settlement data file and reports success or displays an error.

stateconfigtool

Use this utility to load state configuration (**state.config**) files for use with the Pipeline Manager data migration feature.

> **Note:**
>
> Before you run **stateconfigtool**, make sure that the following files are listed in your system CLASSPATH environment variable:
>
> - **msbase.jar**
> - **msutil.jar**

Location

*Pipeline_home***/tools/StateConfigTool**

Syntax

```
stateconfigtool -f file_name -d database_type -h host -n port -i database_id
```

Parameters

-f

The path and file name of the of the **state.config** file to be loaded. This file contains descriptions about changeset state transitions, such as currentState, nextState, and Action.

The default directory is *Pipeline_home***/tools/StateConfigTool**.

-d

The database type. The supported database is **oracle**.

-h

The host name of the computer running the Pipeline Manager database.

-n

The port number used by the Pipeline Manager database.

-i

The database ID of the Pipeline Manager database.

Results

The utility loads the contents of the **state.config** file into the Pipeline Manager database. The states defined in the file become available in the Change Set Manager when it is restarted.

StopRapGen

The **StopRapGen** utility searches the database to collect information required by the Stop RAP Generator pipeline to create Stop Return Returned Account Procedure (RAP) files.

It retrieves information on the following:

- Transferred Account Procedure (TAP) files that were received by BRM and stored in the database more than seven days ago.
- Stop Return RAP files that were generated by BRM and sent more than seven days ago to the Visited Public Mobile Network (VPMN) operator.

> **Note:**
>
> The output from the **StopRapGen** utility is used by the Stop RAP Generator pipeline to generate the Stop Return RAP file.
>
> Use the **StopRapGen** utility along with the Stop RAP Generator pipeline.

Location

Pipeline_home/**bin**

where *Pipeline_home* is the directory in which you installed Pipeline Manager.

Syntax

```
StopRapGen64 database_access_library server_name database_name path [prefix] [days]
```

Parameters

database_access_library
The database access library.

server_name
Specifies the name of the host machine running the Pipeline Manager database.

database_name
Specifies the database ID of the Pipeline Manager database.

path
Specifies the output directory of the flat file generated by the **StopRapGen** utility. This file is used by the Stop RAP Generator pipeline.

> **Tip:**
>
> The output directory for the **StopRapGen** utility should be the same as the input directory for the Stop RAP Generator pipeline.

prefix
Specifies the prefix to be added to the output flat file. The default prefix is **RC**.

days
Specifies the number of days to consider for generating a Stop Return RAP file. The default is 7, in accordance with the RAP standard.

Example

```
StopRapGen64 liboci10g6312d.so $ORACLE_SID '' ./data/stoprap/in
```

where:

- **liboci10g6312d.so** is the database access library.
- **$ORACLE_SID** is the database alias.
- ' ' is the empty string passed in as the database name.

- **.data/stoprap/in** is the output directory of the sample usage data for the **StopRapGen** utility (the flat file it generates). This is also the input directory of the Stop RAP Generator pipeline.

Results

The **StopRapGen** utility generates the input required by the Stop RAP Generator pipeline.

ZoneDBImport

The **ZoneDBImport** utility loads data in the IFW_STANDARD_ZONE table of the Pipeline Manager database.

This utility uses the following files:

- **Control File (zoneLoader.ctl**)

  The **zoneLoader.ctl** file controls how the data is loaded. It contains information about the table name, column datatypes, field delimiters, and so on.

  Initialize the infile variable with the path and file name of the file that contains the data to be imported.

- **Execution File (zoneLoader.pl)**

  Update the entries for the **DatabaseName** and **UserName** with the database name and user name of the current database.

Location

*Pipeline_home***/tools**

Syntax

```
./zoneLoader.pl
```

11

Pipeline Manager Opcode Reference

Learn about the Oracle Communications Billing and Revenue Management (BRM) opcodes related to Pipeline Manager.

Topics in this document:

> **Caution:**
>
> - Always use the BRM API to manipulate data. Changing data in the database without using the API can corrupt the data.
> - Do not use SQL commands to change data in the database. Always use the API.

Account Synchronization FM Opcodes

The opcodes in Table 11-1 synchronize customer and service data with pipeline rating data.

Header File

Include the **ops/ifw_sync.h** header file in all applications that call these opcodes.

Opcode Index

Table 11-1 Account Synchronization FM Opcodes

Opcode	Description	Use
PCM_OP_IFW_SYNC_PUBLISH_EVENT	Passes events associated with this opcode in your system's event notification list to the policy opcode.	Limited
PCM_OP_IFW_SYNC_POL_PUBLISH_EVENT	Policy for modifying the events passed to the standard opcode.	Recommended

PCM_OP_IFW_SYNC_PUBLISH_EVENT

Passes events associated with this opcode in your system's event notification list to the PCM_OP_IFW_SYNC_POL_PUBLISH_EVENT policy opcode for processing.

By default, PCM_OP_IFW_SYNC_PUBLISH_EVENT passes events without modifying them.

PCM_OP_IFW_SYNC_POL_PUBLISH_EVENT

Modifies events included in business events synchronized between BRM and Pipeline Manager.

Events that trigger event notification for synchronization make up the business events that the Oracle Data Manager (DM) sends to Pipeline Manager. This opcode modifies specified triggering events before they are published to Pipeline Manager.

This opcode can also be used to filter out certain events that are not included in the synchronization process (for example, an event that has only balance impacts for currency balance elements), thereby reducing the traffic in the listener queue.

If you pass an event that does not need any modification to this opcode, the opcode passes that event to the EAI framework for publishing.

Batch Suspense Manager FM Standard Opcodes

The opcodes listed in Table 11-2 manage batch files for suspended EDRs stored in the BRM database as **/suspended_batch** objects.

Header File

Include the **ops/batch_suspense.h** header file in all applications that call these opcodes.

Opcode Index

Table 11-2 Batch Suspense Manager FM Standard Opcodes

Opcode	Description	Use
PCM_OP_BATCH_SUSPENSE_DELETE_BATCHES	Deletes suspended batches from the BRM database. Available with Suspense Manager.	Recommended
PCM_OP_BATCH_SUSPENSE_RESUBMIT_BATCHES	Resubmits the batches which have been suspended by the pipeline.	Recommended
PCM_OP_BATCH_SUSPENSE_WRITE_OFF_BATCHES	Writes off suspended batches.	Recommended

PCM_OP_BATCH_SUSPENSE_DELETE_BATCHES

Deletes suspended batches from the BRM database.

> **Note:**
>
> This opcode is available to Suspense Manager customers only.

PCM_OP_BATCH_SUSPENSE_RESUBMIT_BATCHES

Initiates batch resubmission. During the resubmission process, suspended batches are sent back through their original rating pipelines. The Suspense Management Center calls this opcode when the user chooses to resubmit suspended batches.

PCM_OP_BATCH_SUSPENSE_WRITE_OFF_BATCHES

Writes off the batches which are at the “Suspended" stage because of some business rule. The GUI calls this opcode to write off the batches.

> **Note:**
>
> This opcode is available to Suspense Manager customers only.

Filter Set FM Standard Opcodes

This document describes the filter set opcodes listed in Table 11-3. These opcodes support BRM Pricing Center in providing separate products and discounts to the different market segments of your customer base. These opcodes allow you to divide your customers into market segments by filtering them based on criteria that you set in Pricing Center.

Header File

Include the **ops/filterset.h** header file in all applications that call these opcodes. See the discussion on header files in *BRM Developer's Guide*.

Opcode Index

Table 11-3 Filter Set FM Standard Opcodes

Opcode	Description	Use
PCM_OP_FILTER_SET_CREATE	Creates a new **/filter_set/product** object. See the discussion on creating filter sets in *BRM Setting Up Pricing and Rating*.	Recommended
PCM_OP_FILTER_SET_DELETE	Deletes a **/filter_set/product** object. See the discussion on deleting filter sets in *BRM Setting Up Pricing and Rating*.	Recommended
PCM_OP_FILTER_SET_UPDATE	Modifies a **/filter_set/product** object. See the discussion on updating filter sets in *BRM Setting Up Pricing and Rating*.	Recommended

PCM_OP_FILTER_SET_CREATE

Creates **/filter_set/product** objects, which store the list of system products and discounts that belong to a particular filter set. This opcode is called directly by Pricing Center.

See the discussion on creating filter sets in *BRM Setting Up Pricing and Rating*.

PCM_OP_FILTER_SET_DELETE

Deletes **/filter_set/product** objects. This opcode is called directly by Pricing Center.

See the discussion on creating filter sets in *BRM Setting Up Pricing and Rating*.

PCM_OP_FILTER_SET_UPDATE

Modifies the following data in **/filter_set/product** objects:

- The filter criteria
- The list of applicable system products and discounts
- The validity period

This opcode is called directly by Pricing Center.

See the discussion on updating filter sets in *BRM Setting Up Pricing and Rating*.

Suspense Manager FM Standard Opcodes

The opcodes listed in Table 11-4 manage suspended EDRs stored in the BRM database as **/suspended_usage** objects.

Header File

Include the **ops/suspense.h** header file in all applications that call these opcodes.

Opcode Index

Table 11-4 Suspense Manager FM Standard Opcodes

Opcode	Description	Use
PCM_OP_SUSPENSE_DEFERRED_DELETE	Deletes records for suspended EDRs after Revenue Assurance has been completed. Available with Suspense Manager.	Recommended
PCM_OP_SUSPENSE_DELETE_USAGE	Deletes records for suspended EDRs. Available with Suspense Manager.	Recommended
PCM_OP_SUSPENSE_EDIT_USAGE	Changes the contents of fields in suspended EDRs. Available with Suspense Manager.	Recommended
PCM_OP_SUSPENSE_RECYCLE_USAGE	Initiates EDR recycling.	Recommended
PCM_OP_SUSPENSE_SEARCH_DELETE	Deletes call records with a specific recycle key and a status of **succeeded** or **written off**.	Recommended
PCM_OP_SUSPENSE_SEARCH_EDIT	Changes fields in a large number of suspended records in one database operation.	Recommended
PCM_OP_SUSPENSE_SEARCH_RECYCLE	Recycles suspended EDRs. Available with Suspense Manager.	Recommended
PCM_OP_SUSPENSE_SEARCH_WRITE_OFF	Writes off a large number of suspended records in one database operation.	Recommended
PCM_OP_SUSPENSE_UNDO_EDIT_USAGE	Undoes edits to suspended EDRs. Available with Suspense Manager.	Recommended
PCM_OP_SUSPENSE_WRITTEN_OFF_USAGE	Writes off suspended EDRs.	Recommended

PCM_OP_SUSPENSE_DEFERRED_DELETE

Deletes EDRs in a written off state or succeeded state. This opcode is scheduled to run at a later time to ensure Revenue Assurance.

> **Note:**
>
> This opcode is available to Suspense Manager customers only.

PCM_OP_SUSPENSE_DELETE_USAGE

Deletes EDRs in a written off state or succeeded state.

> **Note:**
>
> This opcode is available to Suspense Manager customers only.

PCM_OP_SUSPENSE_EDIT_USAGE

Changes the contents of EDR fields for a suspended call record. The Suspense Management Center calls this opcode to edit a suspended call record.

> **Note:**
>
> This opcode is available to Suspense Manager customers only.

PCM_OP_SUSPENSE_RECYCLE_USAGE

Initiates EDR recycling. During recycling, suspended EDRs are sent back through their original rating pipelines. The Suspense Management Center calls this opcode when the user chooses to recycle suspended EDRs.

PCM_OP_SUSPENSE_SEARCH_DELETE

Deletes call records with a status of **succeeded** or **written off** that match criteria specified in the input flist. You can specify the following criteria:

- A recycle key
- A CDR file
- A search template

This opcode can also delete a **suspended** call record if PIN_FLD_MODE is set correctly.

PCM_OP_SUSPENSE_SEARCH_EDIT

This opcode makes changes to a large number of suspended records that meet the criteria specified in the input template.

PCM_OP_SUSPENSE_SEARCH_RECYCLE

Searches for and queues suspended call records for recycling based on criteria specified in the input flist. You can specify the following criteria:

- A recycle key
- A CDR file
- A search template

PCM_OP_SUSPENSE_SEARCH_WRITE_OFF

This opcode writes off a large number of suspended records that match the search criteria in the input flist.

PCM_OP_SUSPENSE_UNDO_EDIT_USAGE

Undoes edits to suspended call records used by Suspense Manager. This opcode is called by Suspense Management Center to perform the undo edit action. It replaces the value of a field in a suspended call record with the value in that field before the last edit was made.

> **Note:**
>
> This opcode is available to Suspense Manager customers only.

PCM_OP_SUSPENSE_WRITTEN_OFF_USAGE

Writes off suspended EDRs. When a suspended EDR is written off, they cannot be edited or recycled.

> **Note:**
>
> This opcode is available to Suspense Manager customers only.

12

Revenue Assurance Manager Reports

Learn about the Oracle Communications Billing and Revenue Management (BRM) reports that support batch rating-related Revenue Assurance Manager features.

Topics in this document:

Revenue Assurance Rating Summary Report

> **Note:**
>
> You must upload the **RevAssuranceRating.source** file that is in the **StoredProcedures** folder to the database after the report is installed.

The Revenue Assurance Rating Summary report (**RevAssuranceRatingSummary.rtf**) shows revenue assurance data collected from pipeline rating. This report returns information for a specified time period.

There are different subreports created for different rating **/process_audit** objects.

The SQL query for this report is in the *OAP_home***xmlp\XMLP\Reports\BRM Reports\Revenue Assurance Reports\StoredProcedures\RevAssuranceRating.source** file, where *OAP_home* is the directory in which Oracle Analytics Publisher is installed.

No charts are available for this report.

Revenue Assurance Rating Summary Report Parameters

You can change the following parameters to modify the output of the Revenue Assurance Rating Summary report:

- Start Date (process start date or date-time)
- End Date (process end date or date-time)
- Trans Start Date (transaction start date or date-time)
- Trans End Date (transaction end date or date-time)
- Flow
- Control Point
- Service Type

> **Note:**
>
> If you want to generate a report for the service types that are not in the list of the default service types, add the service type in the list of values in Oracle Analytics Publisher. For more information on adding the list of values, see the Oracle Analytics Publisher documentation.

- Batch Type

Table 12-1 Revenue Assurance Rating Summary Parameters

Parameter	Description	Valid Values
Flow	A collection of linked control points in a pipeline.	Any valid flow name. Default: not specified
Control point	An instance of FCT_AggreGate that you configure in your pipeline to collect revenue assurance data. For example, CP_Afterrating.	Any valid control point names or **ALL**. Default: **ALL**
Service Type	A commodity sold by your company and that your customers can purchase and use. For example, SMS or TEL.	Any valid service type names. Default: not specified
Batch Type	The type of batch that the EDRs belong to. • **0** for normal • **1** for rerating • **2** for recycling • **3** for write-off	The batch type numbers. Default: not specified

Revenue Assurance Rating Detail Report

> **Note:**
>
> You must upload the **RevAssuranceRating.source** file that is in the **StoredProcedures** folder to the database after the report is installed.

The Revenue Assurance Rating Detail report (**RevAssuranceRatingDetail.rtf**) shows detailed statistics for the revenue assurance data collected from pipeline rating. By default, the statistics are organized by EDR status. For each status, this report displays the following:

- Volume sent
- Volume received

The SQL query for this report is in the *OAP_home***xmlp\XMLP\Reports\BRM Reports\Revenue Assurance Reports\StoredProcedures\RevAssuranceRating.source** file, where *OAP_home* is the directory in which Oracle Analytics Publisher is installed.

No charts are available for this report.

Revenue Assurance Rating Detail Parameters

You can change the following parameters to modify the output of the Revenue Assurance Rating Detail report:

- Start Date (process start date or date-time)
- End Date (process end date or date-time)
- Trans Start Date (transaction start date or date-time)
- Trans End Date (transaction end date or date-time)
- Flow
- Control Point
- Service Type
- Batch Type
- Batch ID
- Successful records
- Failed records
- Written Off records
- Dynamic Data
- Revenue Assurance Scope

Table 12-2 Revenue Assurance Rating Detail Parameters

Parameter	Description	Valid Values
Flow	A collection of linked control points in a pipeline. For example, Flow1. **Important**: This field is mandatory.	Any valid flow name. Default: not specified
Control point	An instance of FCT_AggreGate that you configure in your pipeline to collect revenue assurance data. For example, CP_Afterrating. **Important**: This field is mandatory.	Any valid control point names or **ALL**. Default: **ALL**
Service Type	A commodity sold by your company and that your customers can purchase and use. For example, SMS or TEL.	Any valid service type names. Default: not specified
Batch Type	The type of batch that the EDRs belong to. • **0** for normal • **1** for rerating • **2** for recycling • **3** for write-off	The batch type numbers. Default: not specified
Batch ID	Batch ID of the EDRs. For example, MED1 or MED2. **Important**: This field is mandatory.	Any valid batch ID. Default: not specified
Successful Records	The status of successful EDRs.	The name given to define the successful EDRs in pipeline rating. Default: **Successful**

Table 12-2 (Cont.) Revenue Assurance Rating Detail Parameters

Parameter	Description	Valid Values
Failed Records	The status of failed EDRs.	The name given to define the failed EDRs in pipeline rating. Default: **Suspense**
Written Off Records	The status of written-off EDRs.	The name given to define the written-off EDRs in pipeline rating. Default: **Written-off**
Dynamic Data	Determines whether the revenue assurance data is dynamic or static. Select **YES** to show dynamic data. Select **NO** to display static data. The static data displays data on the call data records (CDRs) rated from the mediation batches. The dynamic data displays data on the CDRs that are rejected, recycled, or rerated.	**YES** or **NO**. Default: **NO**
Revenue Assurance Scope	Refers to the Batch Type parameter.	Batch rating or rerating. Default: not specified