User Defined Module
Oracle FLEXCUBE Universal Banking
Release 11.83.03.0.0
[April] [2014]
Oracle Part Number E80246-01

ORACLE®
FINANCIAL SERVICES

Table of Contents

ORACLE®

1. About this Manual

1.1 Introduction

This manual is intended as a guide to help you define User Defined Module (UDM) in FLEXCUBE. You can further obtain information specific to a particular field by placing the cursor on the relevant field and striking <F1> on the keyboard.

1.2 Organization

This manual is organized as follows:

Chapter 1	*About this Manual* gives information on the intended audience. It also lists the various chapters covered in this User Manual.
Chapter 2	*Building a User Defined Module* explains how to build a module in Oracle FLEXCUBE either for your own convenience or to suit the requirements of your bank.

1.3 Conventions Used in this Manual

Important information is preceded with the STOP symbol.

1.4 Glossary of Icons

This User Manual may refer to all or some of the following icons.

Icons	Function
	New
	Copy
	Save
	Delete
	Unlock
	Print
	Close
	Re-open
	Reverse
	Template

ORACLE®

Icons	Function
	Roll-over
	Hold
	Authorize
	Liquidate
	Exit
	Sign-off
	Help
	Add row
	Delete row
	Option List
	Confirm
	Enter Query
	Execute Query

Refer the Procedures User Manual for further details about the icons.

1.5 <u>Related Documents</u>

For further information on procedures discussed in the manual, refer to the Oracle FLEXCUBE manuals on:

- Core Entities
- Core Services
- Common Procedures
- Products
- User Defined Fields

ORACLE®

2. Building a User Defined Module

2.1 Introduction

Often while handling large quantity of data in your bank you might want to capture and process information in a particular fashion. Consequently you might want to define your own module either for your own convenience or to suit the requirements of your bank.

The User Defined module of Oracle FLEXCUBE Corporate gives you the opportunity to define your own module whereby you can capture and process information based on your specifications to meet your needs.

You can do this by way of defining the parameters listed below:

- Fields, to capture information specific to individual transaction. You can do this by way of defining custom fields through the User Defined Fields screen of the Core Services module

- Events to be triggered during contract processing

- Amount Tags which are nothing but tags to be attached to amounts which are used to apply charges and taxes

- Accounting Roles for the purpose of passing accounting entries

Oracle FLEXCUBE also offers you the flexibility of selecting the functionality that should form a part of the module being defined. You can choose to have a combination of any of the following features at the Product Definition level:

- Accounts

- Branch

- Charges

- Customer

- Events

- Interest

- MIS

- Preference

- Tax

In addition you also need to specify whether the parameters pertaining to Interest Charges, Commission; and Tax are applicable or not.

ORACLE

2.2 Building an Events class

A class is a specific type of component that you can build with certain attributes. You can build a charge class, for instance, with the attributes of a specific type of charge, such as Charges for provision of services. Similarly, you can build an event class with the attributes of a specific type of events, such as a Booking a Transaction, Collecting Charges, Cancellation and so on.

You can identify an Events Class with a unique Code and Description. When you define an Events Class, you choose, first of all, the set of events that would belong to the class.

Events are, typically, unique to a module.

You can build the events that you would like to include in an Events Class in the 'Events Class Maintenance' screen. You can invoke this screen by typing 'CSDACTCL' in the field at the top right corner of the Application tool bar and clicking on the adjoining arrow button.

For every event constituting the class that you are building, you have to specify the accounting entries that should be passed (if any), and the advices that should be generated. You can do this through the Product Accounting Entries and Advices maintenance screen, which is explained in detail later in the manual.

All events need to be triggered manually. The procedure of triggering events manually is explained subsequently in the chapter on Manual Triggering of events.

ORACLE

Since events will have to be triggered manually you will not be allowed to maintain an event Sequence Number as of now.

2.3 Creating a Module Using Oracle FLEXCUBE Corporate

The 'Module Details' screen allows you to define a module based on your requirements.

2.4 Selecting the icons for the product definition screen

In the Product Definition screen of any module, a horizontal array of buttons is displayed. You need to click on each of the buttons to define the specific attributes of the product. E.g. the 'Interest' button is used to define Interest details, 'tax' button is used to define tax details etc. While creating a new module, you need to indicate the icons that have to be included in the Product Definition screen.

Click 'Buttons' to specify the icons that have to be included in the 'Product Definition' screen of the new module that is being defined.

ORACLE
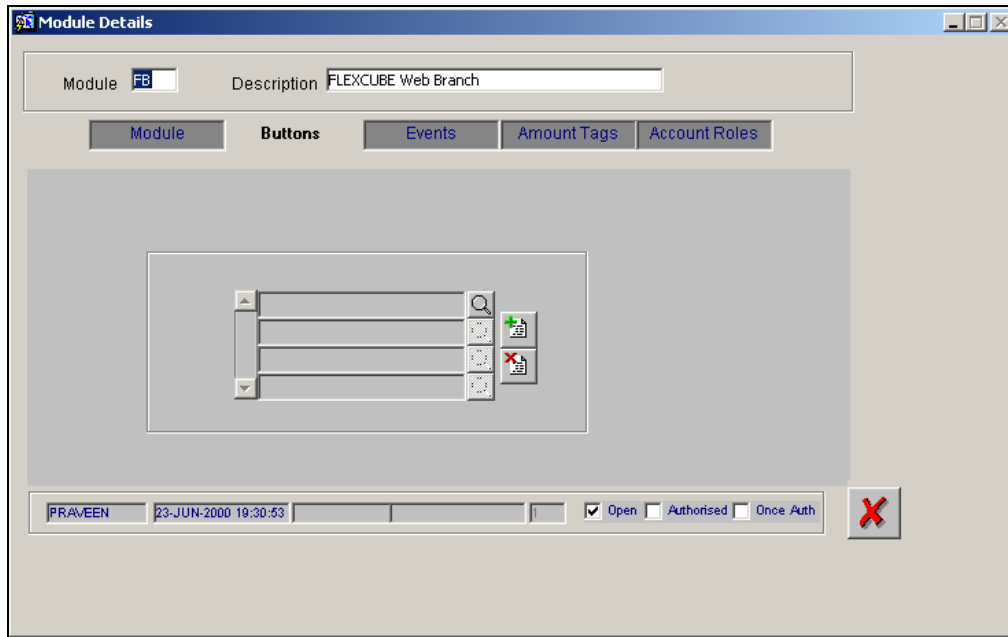
Depending on your selections in this screen, the Product Definition screen of the new module will have the attributes (represented by the respective icons as in any other Product Definition screen).

In this screen, you can specify the icons for:

- Defining the interest and charges that you would like to levy on transactions involving the product.

- Maintaining tax details that will be applicable on the transactions involving the product.

- Indicating the type of accounts and the GL/SLs to which the accounting entries have to be posted.

- The preferences specific to a product.

- Maintaining the events that will be generated at different points in the life cycle of contracts involving the product.

- Maintaining a list of allowed or disallowed branches, currencies and customers that can use a product.

- Maintaining the Management Information System (MIS) details.

2.5 <u>Indicating the Event Details</u>

A contract that you process in Oracle FLEXCUBE goes through different stages during its life cycle. These stages are defined as Events. Every new module that you maintain has to be associated with a set of events, which will be triggered at appropriate stages during the lifecycle of the contract.

Apart from the factory shipped events, you can create your own events as per the requirements of the bank.

ORACLE®

Click 'Events' button in the 'Module Details' screen. You can define events for a new module and also for existing modules in this screen.
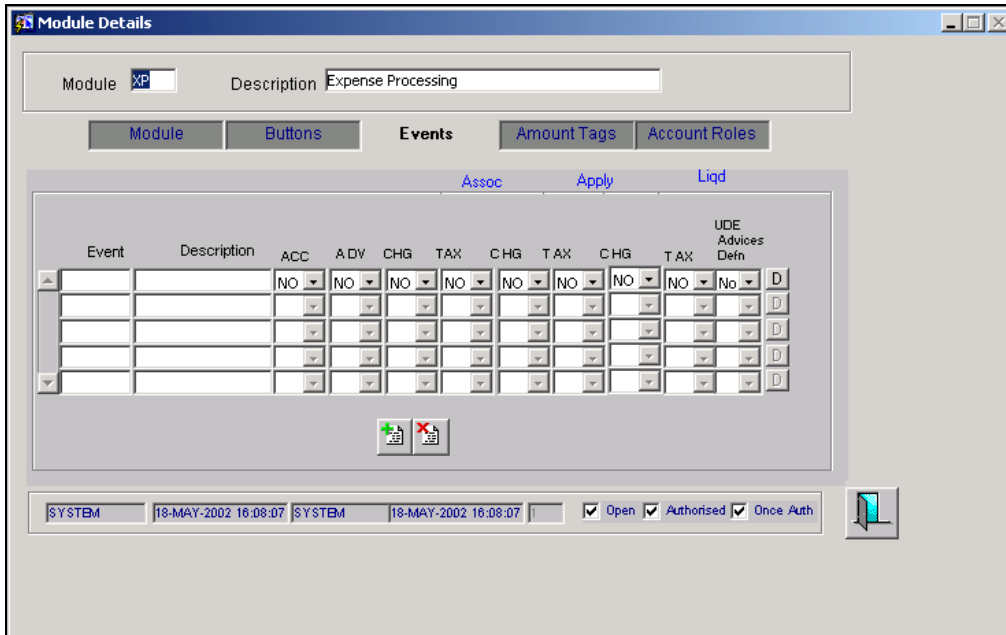
User defined events will be linked to a product and is triggered in the life cycle of a contract, which is processed under that product.

2.5.1 <u>Specifying the Event Code</u>

The event will be defined by a code. The event code should be unique for a module. Indicate the code through which the event will be identified and also give a brief description of the new event.

You can also specify the following parameters for the event being defined:

- Whether accounting entries and advices are allowed for this event.

- Whether interest, charge and tax must be computed, but not accrued or levied during this event. This is represented in the respective options under the section 'Assoc'.

- Whether the accounting *entries have to be passed* for interest, tax and charges. This can be indicated in the respective options under the section 'Apply'.

- Whether the interest, charge and tax components must be liquidated when the new event being defined is triggered.

- Whether contract UDE Advices are allowed for this event.

The two charge classes are linked to the LD product definition while accounting entries for the charges is defined in the HDBC and LIQD events respectively.

Click 'D' button to specify the event triggering details.

2.5.2 Specifying the method for event triggering

The event, which you are defining, can be triggered either:

- Manually, or
- Automatically

UDE for commission on highest outstanding balance of a contract

For instance, you can set up a user defined event, HDBC, for calculation of commission on the highest outstanding balance of a contract, with the following parameters:

- Event Trigger = Automatic
- Apply Charge = Yes
- Execution Query = Select all contracts which satisfy the following criteria:
  - Product type is L (Transfer)
  - Contract Status is Active
  - The system date is the last working day of the month so that the System processes HDBC events for the contracts only at the end of the month.
- An amount tag UDE_HOB_AMT, defined for the Highest Outstanding Balance
- The amount tag derivation rule for UDE_HOB_AMT returns the highest outstanding balance for the contract for the month
- Another pre-shipped amount tag MATU_HOB_AMT is provided which returns the highest outstanding balance of contract as of maturity, since the previous month end. The charge can be applied based on this amount tag on the LIQD event.
- The event is available at product level

The applicable charges are tracked through two charge classes, Charge on Highest Outstanding Balance and Charge on Maturity Balance. The waiver applicable to specific customers is defined at charge class maintenance level. The event of liquidation for charge for the Charge on Highest Outstanding Balance is HDBC and for the Charge on Maturity Balance, LIQD. These events must be linked to an LD product in the product definition.

ORACLE®

The event HDBC computes the charge for Charge on Highest Outstanding Balance, and is executed as part of the UDE batch which executes during the Post-EOTI batch programs; it is run after the execution of the LD batch. The event trigger logic for HDBC ensures that the event only triggers on month ends.

2.5.3 Specifying the validations (Execution Query)

You can write an Execution query by which you can instruct the system to retrieve certain contracts to associate with the event that is being defined. The validation rule contains the condition based on which the contracts will be retrieved and associated with the event that is being defined.

2.5.4 Event Processing

The system derives the values for certain UDF parameters based on the values you give here. During the processing of the event, the values that are mentioned in this place will supercede the ones maintained for the contract.

2.5.5 Deriving the Value Date

System derives the value date of the event based on the derivation rule. The accounting entries associated with the event being defined will be posted on the value date.

If the event triggering is set to 'Automatic', you need to write a code to derive the value date of the event. The accounting entries associated with the event being defined will be posted on the value date derived by the system.

When you select automatic mode of event triggering, 'Value Date Derivation' option is automatically checked, as it is mandatory for events triggered automatically. Click 'D' button to write a code to derive the value date for the event.

ORACLE®
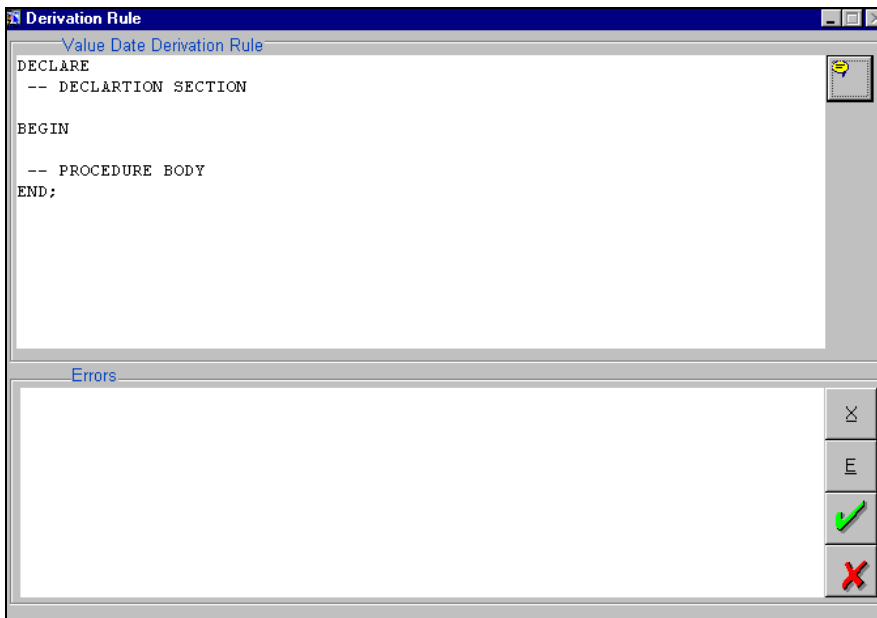
The value that you derive in these procedures should be assigned to specific tags. You can use the following condition keys:

L_VAL_DT	For assigning value date
P_CONTRACT_REF_NO	Contract reference number
P_VERSION_NO	Version number
P_EVENT_CODE	Event code
P_EVENT_SEQ_NO	Event sequence number

🛑 After entering the code, click on the X button to compile the code.

Example (A)

Scenario

All Deposit transactions having prepayments will incur a penalty on the outstanding amount.

The rate at which the penalty should be charged is maintained at the UDF defined for the contract. The UDF is called PREPAYRATE.

To meet this requirement we need to define a new event called PREP for the Deposits module. The event will be allowed for accounting entries only.

In the event triggering screen you can either select automatic or manual processing. For manual processing, there is no need for Value Date derivation or Execution query. In case of automatic processing, under value date derivation capture the following procedure to pass the application date as the value date for posting entries.

Begin

L_VAL_DT := global.application_date ;

End;

In the event processing procedure window, (which is invoked by selecting the button) there is no need to pass any values or parameters runtime. So we will write the following:

Begin

Null;

End;

In case of automatic triggering of event an execution/validation query is required. This will identify the contracts for which the event should be executed.

Select contract_ref_no from ldtbs_amount_due

where component = 'PRINCIPAL'

and due_date > GLOBAL.APPLICATION_DATE

and contract_ref_no in (select trn_ref_no from actbs_daily_log where delete_stat <>'D' and auth_stat ='A')

ORACLE

having sum(amount_settled) > 0 group by contract_ref_no

🛑 The derivation rule for the value date can be written for events, which are triggered manually also. However, if the derivation rule is not written, the value date can be entered at the time of manually triggering an event.

2.6 <u>Defining the Amount Tags</u>

You can specify the amount tags that have to be included in a new module/existing module and the method by which the system has to derive the amount and currency for a particular amount tag. Click 'Amount Tags' button in the 'Module Details' screen to specify these details.

You can link two amount tags to a single accounting role at the time of creating a product.

At the time of processing transactions, the currencies of the two amount tags for the debit and credit legs can be different.

<u>*Example (B)*</u>

Let us assume that you have created a product LDML. The accounting entries for the event LIQD are as follows:

Accounting Role	Amount Tag	Debit/Credit
CUSTOMER	ASSGN-DISCOUNT	Credit
LDML-INTINC	ASSGN-DISCOUNT	Debit
CUSTOMER	ASSGN-PREMIUM	Debit
LDML-INTINC	ASSGN-PREMIUM	Credit

You will notice that the amount tags for the debit and credit legs are different for the event LIQD.

ORACLE®

Let us assume that at the time of liquidating a transaction under the product LDML:

- The currency of amount for the debit leg is GBP

- The currency of amount for the credit leg is INR.

However, the local currency of your bank is USD.

Therefore, the currencies of the amount tags linked though linked to the same accounting role are different for the debit and credit legs.

The different currencies of the amount tags for the debit and credit legs when converted to local currency may not match. In such cases, wherein the amounts in different currencies when converted to local currency result in different amounts, you can instruct the system to calculate the amount by taking the average of the two amounts.

With reference to the above example, when the amounts in GBP and INR are converted to USD, the amounts in local currency maybe different because of different exchange rates.

Suppose:

- The amount in GBP when converted to USD (local currency) results in 110 USD, and

- The amount in INR when converted to USD results in 120 USD.

Therefore, at the time of specifying the amount tag ASSGN-DISCOUNT, you can specify the method as 'Average' and amount tag as ASSGN-PREMIUM.

Consequently, system will calculate the average amount of the amount tags ASSGN-DISCOUNT and ASSGN-PREMIUM as 110 USD for the event LIQD.

This calculation can be done for both automatic and manually triggered events.
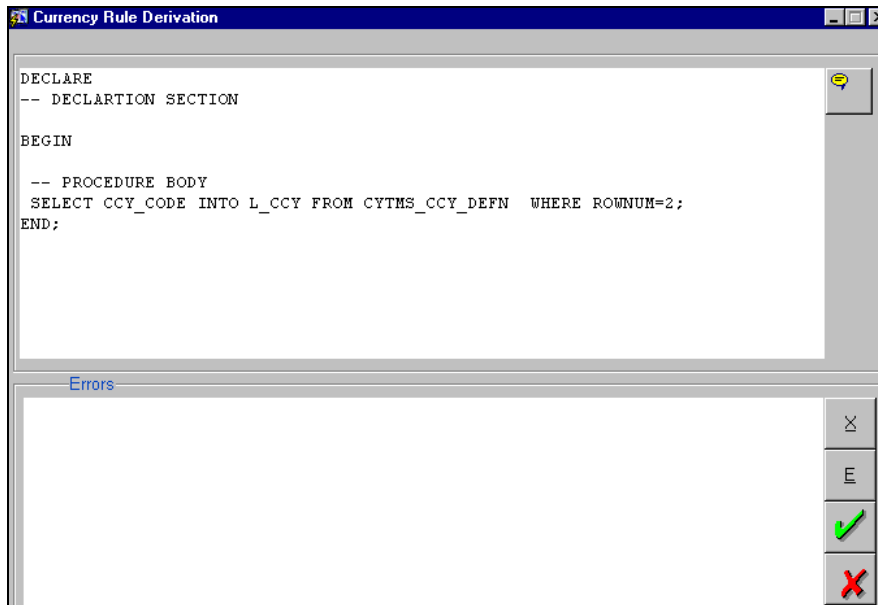
2.6.1 <u>Writing derivation rules for amount and currency</u>

Click 'D' button if you want the system to populate the values of amount and currency based on certain conditions.
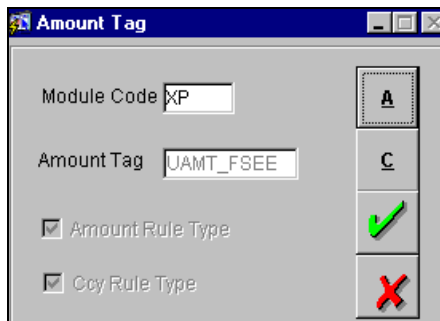
Currency can either be derived from the existing contract currency or can be maintained as a UDF field in the contract or it can be hard coded in the 'Currency Rule Definition' screen.

ORACLE

In the 'Amount Tag' screen, check against 'Ccy Rule Type' if you want to write a derivation code for currency. Then, click 'C' button to write the derivation code.

The variable should necessarily be assigned to L_CCY. The Selecting/Condition keys available are:

- P_CONTRACT_REF_NO
- P_VERSION_NO,
- P_EVENT_SEQ_NO
- P_EVENT_CODE.

With reference to **_Example (A)_**:

The **Amount tag** you can use UAMT_PREP_CHG

The currency derivation can be taken from cstb_contract as follows:

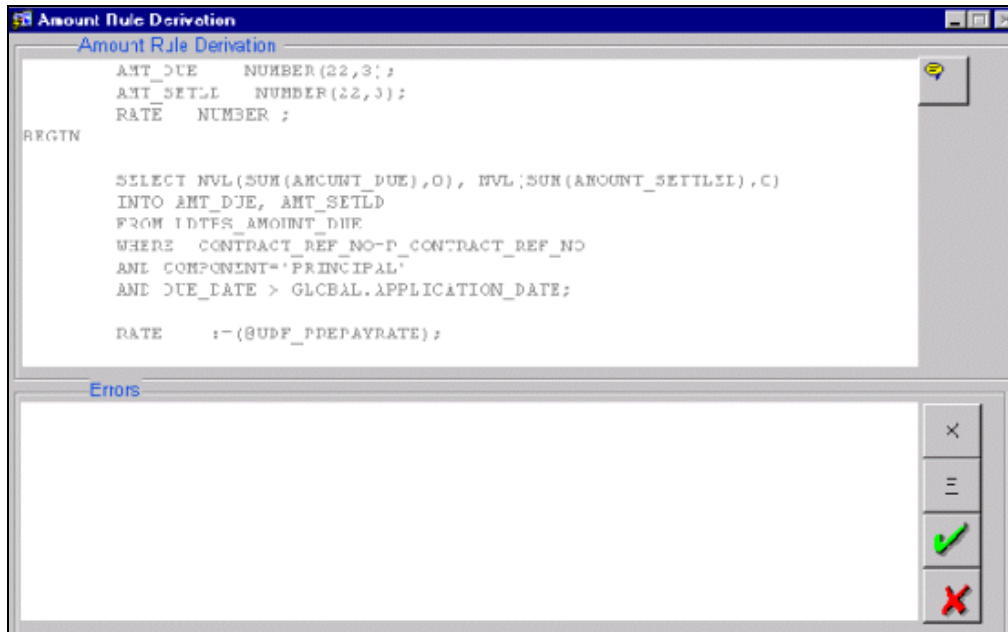Begin

Select contract_ccy into l_ccy from cstbs_contract

where contract_ref_no = p_contract_ref_no;

end;

To write a derivation code for amount, check against 'Amount Rule Type' and then click 'A' button.

ORACLE®

According to your selections and the derivation code, the derivation rule will return a value (either a currency or an amount).

You can use a user defined field in the currency and amount tag derivation rule. While attaching these amount tags in the 'Product Event Accounting Entries Maintenance' screen at the product level, the user defined fields, which are used in the amount tag and derivation rule will be automatically attached to the product. Enter the values for the user defined fields. Consequently, system will derive the value of the amount/currency from the product.

🛑 The variable should necessarily be assigned to L_AMOUNT. The Selecting/Condition keys available are

- P_CONTRACT_REF_NO
- P_VERSION_NO
- P_EVENT_SEQ_NO
- P_EVENT_CODE.

With reference to Example A

Here we will derive the outstanding amount of the LD contract that has undergone a prepayment today before the prepayment was done.

Note: There are 2 possibilities by which we can arrive at the outstanding amount.

ORACLE®

In case of **Manual Triggering** of the event, we use the following and trigger the event before the payment.

```
DECLARE

RATE            NUMBER;

PRIN_OUT        NUMBER;

BEGIN

SELECT PRINCIPAL_OUTSTANDING_BAL

INTO PRIN_OUT

FROM LDTBS_CONTRACT_BALANCE

WHERE CONTRACT_REF_NO=P_CONTRACT_REF_NO;

RATE            :=(@UDF_UD#PREPAYRATE);

L_AMOUNT        := RATE * PRIN_OUT;

END;
```

In case of automatic triggering of the event , we can use the following:

```
DECLARE

AMT_DUE    NUMBER(22,3);

AMT_SETLD   NUMBER(22,3);

RATE   NUMBER ;

BEGIN

SELECT NVL(SUM(AMOUNT_DUE),0), NVL(SUM(AMOUNT_SETTLED),0)

INTO AMT_DUE, AMT_SETLD

FROM LDTBS_AMOUNT_DUE

WHERE CONTRACT_REF_NO=P_CONTRACT_REF_NO

AND COMPONENT='PRINCIPAL'

AND DUE_DATE > GLOBAL.APPLICATION_DATE;

RATE:=(@UDF_PREPAYRATE);

L_AMOUNT:=(AMT_DUE)*RATE/100;

END;
```

Use of UDF

You can use the UDF as a variable in the Derivation in two ways. The options are:

ORACLE

Let us assume that the UDF is called PREPAYRATE

Option I

RATE: =(@UDF_UD#PREPAYRATE);

This used to be the case in the beginning

Option II

RATE: =(@UDF_PREPAYRATE);

2.7 <u>Specifying the Role Type</u>

Click 'Account Roles' button to specify the details of the accounting roles.

In this screen, you can define user defined accounting roles. These accounting roles will be available at the product while mapping the accounting roles to the account heads in the 'Accounting Role to Head Mapping Definition' screen.

2.8   Automatic Triggering of an user defined event

You should run the batch program for automatic firing of a user defined event.

At the time of running the batch process, invoke the 'UD Batch Event' screen from the Application Browser.

Click 'Ok' button to run the Batch Event Triggering program. The batch program will check if there are any events that have to be fired for active contracts.

These are the steps involved in the automatic triggering of a user defined event:

1. The system will execute the validation code and retrieve the appropriate contracts associated with the event.

2. Derivation rule will be executed to get the value date of the new event. (Accounting entries are posted to the respective GL's on the value date).

3. If there are any derivation rules written for amount and currency, the system will execute the derivation rules to get the amount and currency for a particular amount tag for each entry that is passed.

2.9   Manual triggering of Events

Any user defined event, which is set for 'Manual' event triggering can be triggered through the 'User defined Event Triggering' screen. Invoke this screen from the Application Browser.

ORACLE

Navigate to the contract for which you want to trigger an event and click new icon. The user defined events linked to the product under which the contract has been processed will be displayed. Select the event, which has to be triggered. System displays the value date, amount and currency if any derivation logic is written. However, you can change the values (of value date, currency and amount) to suit your requirements.

The associated settlement details, advices, charge and tax details are picked up and the event will be triggered when all the functions are successfully executed.

2.10 <u>Upload for Manual Event triggering</u>

Any user defined event, which is set for 'Manual' event triggering, can also be triggered through the 'User Defined Event Upload' screen. Invoke this screen from the Application Browser.
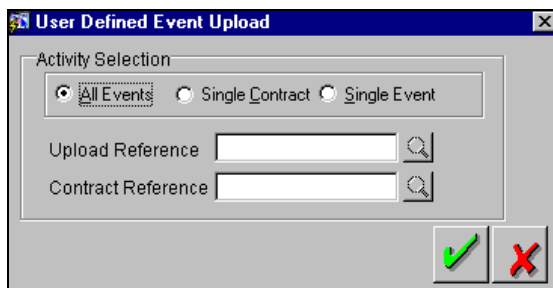
In this method of event triggering, events can be triggered from an upload table, which contains information like the amount, currency and value date of the events, which are set to 'Manual' type of triggering.

The options available for selecting the events for triggering are:

- Single Event – select the appropriate event for triggering from the option list for Upload Reference.

- Single Contract - If this option is selected, all events related to a contract will be triggered. Therefore select a Contract Reference Number. The system will trigger all related events.

- All events – the system will trigger all events of all active transactions.

System will trigger all events of all active transactions.

Select the appropriate option according to your requirement and click 'Ok' button. System will execute the following steps and trigger the event through the upload table:

1. The system will check the method of event triggering for the events that have been selected. Only if it is set to 'Manual', the event will be processed further. Otherwise, it will move on to the next record.

2. System will check whether the upload table has the values of value date, amount and currency. If the table doesn't have the values, the derivation rules will be executed to fetch the values.

ORACLE

3. Next, settlement details and advices are picked up. When all the functions are successfully executed, the event will be triggered from the upload table.

2.11 Defining advices for user defined events linked to Deposits products

The User Defined Event Maintenance screen in Oracle FLEXCUBE allows you to define if the event requires a contract advice to be generated or not. You can generate contract advices for a commitment or a deposit contract when it passes through these stages. You are allowed to generate messages for combination of Branch, Product and each user defined event.

After you specify a list of user defined events applicable for a product, the messages are generated and then handed off at the next stage of the contract upon authorization.

You can generate as many messages for each settlement account if the contract has split settlements or messages can be sent to the counter party of the contract.

Linking the Product to the UDE

You can select the product for which the user defined event details are being maintained. The user defined event details will be validated only for transactions involving the product selected in this field. The description for the product code will be displayed on selection of the product code.

In Oracle FLEXCUBE, when ever you generate messages with type codes 'LD_CONT_' + UDE for any of the UDE, the following message tags is displayed for a LD_CONT_ advice:

- A unique contract reference number

- The contract amount

- The currency of the contract

- A brief description of the customer including counter-party name, address etc.

- The value date of the contract

- The maturity date of the contract

- The interest rate

- The currency of the contract

- Details of the split settlement

ORACLE®

2.11.1.1   Maintaining the Receiver Mapping

You can maintain the receiver of the user defined advice as the counter party or the owner of the
Settlement account using the 'UDE – Advices - Receiver Mapping' screen invoked from the
Application Browser.

If receiver mapping is not maintained then the receiver will be defaulted to counterparty of the
contract.

Specifying the Branch

Capture the branch code for which the user defined advice is generated.

Specifying the Message Type

Select the message type from the option list available.

Indicating the Receiver

Specify whether the receiver of the user defined advice is a Counterparty or the owner of the
Settlement account. Based on this, the advices generated will be sent either to the counter party
or to the owner of the settlement account.

ORACLE®

User Defined Module

[April] [2014]
Version 11.3.83.02.0

Oracle Financial Services Software Limited

Oracle Park

Off Western Express Highway
Goregaon (East)
Mumbai, Maharashtra 400 063
India
Worldwide Inquiries:
Phone: +91 22 6718 3000
Fax:+91 22 6718 3001
www.oracle.com/financialservices/