ORACLE®

Oracle® Fusion Middleware

Configuring and Using the DICOM Component

12*c* (12.2.1.1)

E69274-01

June 2016

Describes installation and deployment of Oracle's Multimedia DICOM support product which enables storage and retrieval of DICOM images in Oracle WebCenter Content.

ORACLE®

Oracle Fusion Middleware Configuring and Using the DICOM Component, 12c (12.2.1.1)

E69274-01

Primary Author: Divya Ramabhadran

Contributors: Melliyal Annamalai, Marcus Diaz, Susan Mavris, David Noblet, Jie Zhang

Contents

4   Performance Tuning Tips

A   Oracle DICOM Protocol Adapter XML Schemas

Preface

This document details how to integrate Oracle Multimedia DICOM features with Oracle WebCenter Content.

Audience

This guide is intended for those people who will install and deploy the product. The primary users of this product are *system integrators*, who will install, configure and set up DICOM support for initial use within WebCenter Content. *End users* will use the product to check in and view images and *adminstrators* will set up appropriate WebCenter Content administrative tasks as needed.

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 customers that have purchased support have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Related Documents

For more information, the following documentation is available for Oracle WebCenter Content:

- *Oracle Multimedia DICOM Developer's Guide*

- *Managing Oracle WebCenter Content*

- *Administering Oracle WebCenter Content*

- *Developing with Oracle WebCenter Content*

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1

Overview

Digital Imaging and Communications in Medicine (DICOM) is a standard specifying an imaging file format and a networking protocol to be used in the medical imaging domain. Picture Archiving and Communications Systems (PACS) are designed to store and transfer medical images in the DICOM format using DICOM network protocol.

Medical imaging devices (such as CT scanners and X-ray machines) communicate with one or more PACS to store images. Health care professionals then review images using a DICOM viewer application to query PACS to retrieve relevant patient information.

DICOM support has been integrated into Oracle WebCenter Content to give Content Server users access to DICOM images. This integration allows images originally stored in a PACS or on the local file system to be managed using Content Server. The DICOM support that has been integrated is based on the functionality of Oracle Multimedia DICOM, a feature of Oracle Database.

This section discusses the following topics:

- Oracle Multimedia DICOM and WebCenter Content Integration

- Usage Scenarios

- Scalability

1.1 Oracle Multimedia DICOM and WebCenter Content Integration

The DICOM support that is integrated with Oracle WebCenter Content is based on Oracle Multimedia DICOM, which is a feature of Oracle Database that enables the efficient management and retrieval of image, audio, video, and medical imaging data in Oracle Database. Oracle Multimedia's comprehensive support for DICOM content includes a rich set of security, conformance, metadata, and processing features to enable open, scalable archives and applications in healthcare and life sciences.

DICOM functionality with Oracle WebCenter Content allows Content Server users to import images from PACS or a local file system, extract metadata associated with images, and generate thumbnail views to store in WebCenter Content. Users can also access DICOM images stored in a PACS via a WADO (Web Access to DICOM Persistent Objects)-compliant URL.

The DICOM metadata extracted from the images is mapped to custom metadata fields in Content Server. From there, it can be managed as needed by users. This provides a unified view of a set of PACS with a common interface for searching and retrieval of image information and metadata.

DICOM support in WebCenter Content has two main elements:

- The Oracle DICOM Protocol Adapter

- The Oracle DICOM Component for WebCenter

When the Oracle DICOM protocol adapter is configured to manage a set of DICOM clients (such as PACS), the adapter periodically queries the set of managed clients to check for new DICOM images. As new images are discovered, the adapter retrieves the images and checks them into Content Server using the Oracle DICOM component for WebCenter.

Depending on the configuration, the following types of content may be checked in:

- The original DICOM image

- A non-DICOM preview image (such as a .jpeg file) of the original image

- An XML representation of the metadata stored in the original DICOM image

After images are stored in Content Server, they can be accessed through standard services or from the DICOM adapter via the DICOM network protocol.
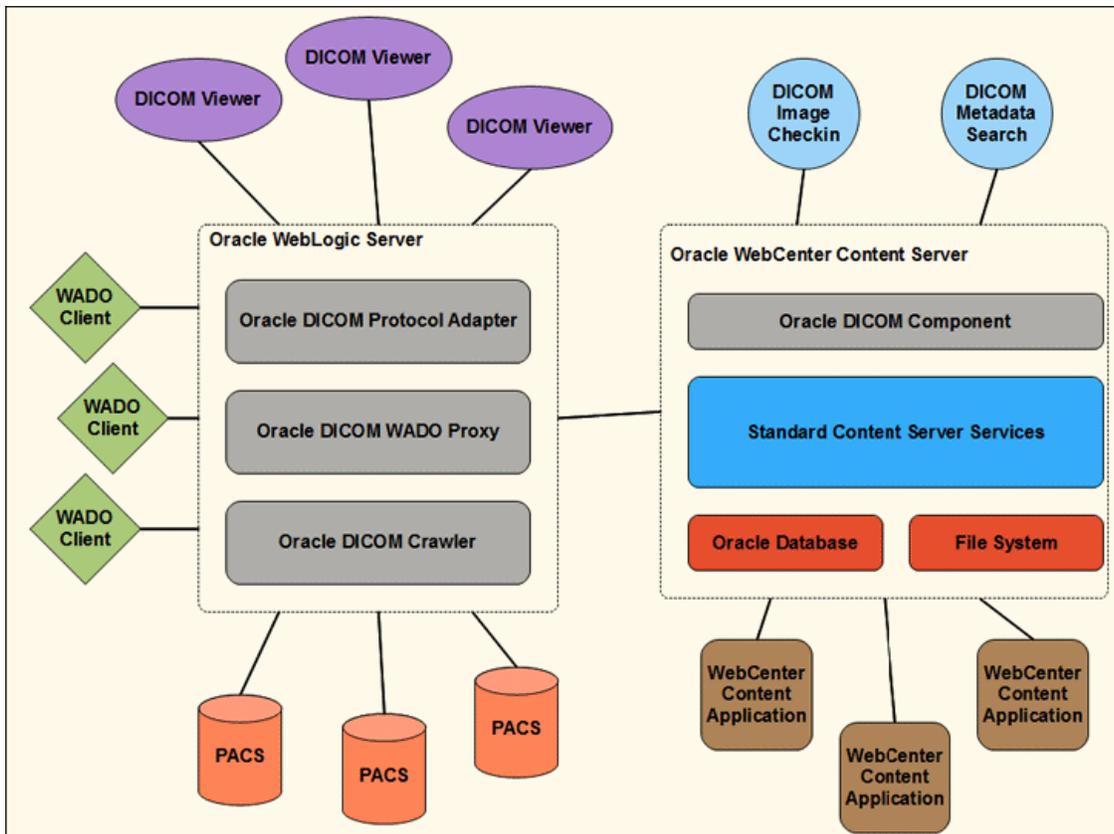
1.1.1 The Oracle DICOM Protocol Adapter

The Oracle DICOM protocol adapter suite communicates with DICOM clients, such as viewers or PACS. It interacts with WebCenter Content to perform operations based on these communications to store, query, and retrieve images from the content server repository.

The suite is comprised of three WebLogic applications that run in the mid-tier on the WebLogic Application Server. These applications communicate with PACS, viewers, and other DICOM clients:

- The Oracle DICOM protocol adapter, which is the core application of the suite. It serves as a bridge between standard DICOM clients (such as PACS and DICOM viewers) and WebCenter Content.

- The Oracle DICOM crawler, which is a scheduled service of the adapter. It periodically queries a set of PACS using the DICOM protocol to import new DICOM images and/or DICOM metadata into WebCenter Content.

- The Oracle DICOM WADO proxy, which is a service used to retrieve DICOM images from remote DICOM clients (such as PACS) via a WADO-formatted URL.

A Content Server entry is always created for DICOM images on checkin. The suite can be configured to manage only the metadata of DICOM images that are checked in or can be configured to manage the actual image as well. It can also be configured to manage a WADO link to point to the location of the original image.

For more details, see Installing and Configuring the Oracle DICOM Protocol Adapter.

1.1.2 The Oracle DICOM Component for WebCenter

When the Oracle DICOM component for WebCenter is installed and enabled on the Content Server, it is used to manage the storage and processing of images in the repository by implementing a checkin filter. The DICOM component is a collection of Content Server services that allow users to check in and process DICOM content and to configure the behavior of the DICOM check-in service.

For more details, see Installing and Configuring the Oracle DICOM Component for WebCenter.

1.2 Usage Scenarios

The Content Server serves as a centralized repository for DICOM images or for images plus metadata. This differs from traditional methods of DICOM usage where images may be scattered among a number of PACS.

There are two common usage scenarios:

- The user consolidates all DICOM images and image metadata from PACS into Content Server.

- The user consolidates only DICOM image metadata into Content Server.

In both scenarios, the Oracle DICOM crawler can be configured with the connection details of all the PACS used for importing. Associated with each PACS in the configuration will be a DICOM query that serves as a filter restricting which images will be returned from the PACS. For example, the Crawler might be configured to query a PACS on a daily basis and to retrieve all images added to the PACS since the day before.

The Oracle DICOM component for WebCenter would also be configured to include metadata mapping for those elements the user wants to retain in Content Server when the image is imported.

See Sample Configuration Scenarios for details about how to configure these two scenarios.

1.3 Scalability

The DICOM crawler can be configured to manage a given set of DICOM clients (for example, PACS). For each DICOM client in the set, the DICOM crawler must periodically query the client to find new images that may have been added since the last query. This type of polling is necessary because the DICOM protocol does not specify a mechanism to register for asynchronous notifications of events, such as the addition of new images.

Due to this architectural constraint, the DICOM crawler is expected to scale to manage sets of DICOM clients of size less than 100 clients. At higher loads, there is risk of unnecessary traffic clogging up the network as a result of querying all the DICOM clients, even when there are no real updates.

2

Installing and Configuring the Oracle DICOM Protocol Adapter

The Oracle DICOM protocol adapter is the core component of Oracle DICOM protocol adapter suite, acting as a bridge between standard DICOM clients (such as PACS and DICOM viewers) and WebCenter Content. This chapter discusses how to install and configure the adapter, the crawler and the proxy.

This documentation assumes that you are familiar with the process for deploying applications on WebLogic Server. For complete documentation about application deployment, see *Oracle Fusion Middleware Deploying Applications to Oracle WebLogic Server*.

This chapter contains the following sections:

- About the Oracle DICOM Protocol Adapter

- Deploying the Oracle DICOM Protocol Adapter

- Configuring the Oracle DICOM Protocol Adapter

2.1 About the Oracle DICOM Protocol Adapter

The Oracle DICOM protocol adapter is composed of three components. You can deploy all or just selected components on WebLogic Server.

The following sections provide a detailed overview of each component:

- The Oracle DICOM Protocol Adapter

- The Oracle DICOM Crawler

- The Oracle DICOM WADO Proxy

For details about deploying the components, see the following sections:

- Deploying the Archive Files

- Standard Deployment

- Advanced Deployment

For details about configuring each component, see the following sections:

- Configuring the DICOM Adapter

- Configuring the DICOM Crawler

- Configuring the DICOM WADO Proxy

2.1.1 The Oracle DICOM Protocol Adapter

The DICOM adapter is deployed as a ResourceAdapter application in Oracle WebLogic Server. The adapter supports the following DICOM Protocol roles:

- C-ECHO SCP

- C-FIND SCP

- C-MOVE SCP

- C-STORE SCP

- C-STORE SCU

See Protocol Message Handling for more details about those protocol roles.

The following iillustration shows how the protocol adapter interacts with other DICOM clients, such as DICOM viewers and PACS, as well as with WebCenter Content. This example shows how a PACS might push an image to the DICOM adapter for checkin to Content Server. It also illustrates how a DICOM viewer might attempt to query and retrieve DICOM images from Content Server via the DICOM adapter.
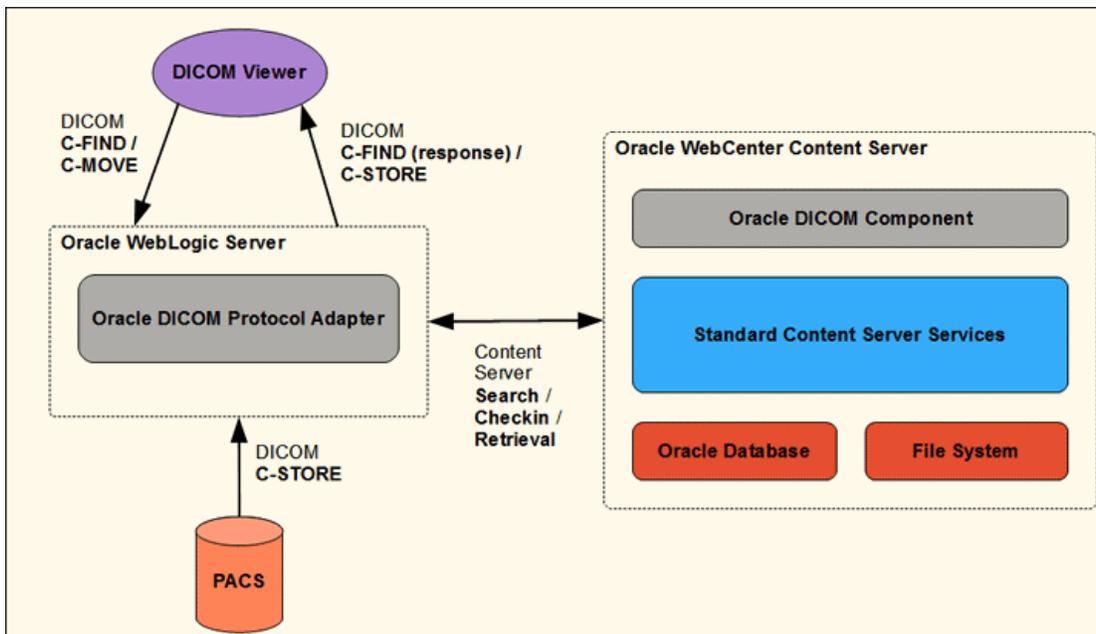
2.1.1.1 Protocol Message Handling

The DICOM adapter supports the following operations of the DICOM network protocol (as specified in the DICOM standard). DICOM clients may issue these DICOM commands to the DICOM adapter to query and retrieve DICOM image content stored in Content Server:

- C-ECHO: used to determine connectivity between the DICOM adapter and a DICOM client. Upon receipt of a C-ECHO request, the DICOM adapter issues a C-ECHO response indicating that the message was received and that the connection is good.

- C-MOVE: used to query the DICOM adapter to retrieve images from the content server repository that match the given metadata attributes (wildcards are supported) defined in the C-MOVE request.

- C-STORE: used to transfer an image to the DICOM adapter. When the DICOM adapter receives a C-STORE request, it takes the associated DICOM image and checks the image into Content Server.

- C-FIND: used to query the DICOM adapter to retrieve image metadata in the content server repository that matches the given query. Wildcards are supported in the queries.

2.1.2 The Oracle DICOM Crawler

The Oracle DICOM crawler is a scheduled service of the DICOM adapter that periodically queries a set of PACS using the DICOM protocol to import new DICOM images and/or DICOM metadata into Content Server. It is deployed as a ResourceAdapter application in Oracle WebLogic Server.

The user can configure the DICOM crawler to periodically issue DICOM C-FIND requests to a PACS in order to identify DICOM images of interest according to user-specified search criteria. When the DICOM crawler identifies such images, it issues a C-MOVE request to the PACS in order to transfer these images to the DICOM adapter which will check in the images or the images' metadata to the content server repository. The following graphic depicts how these messages are exchanged between the DICOM crawler, PACS, and the DICOM adapter.
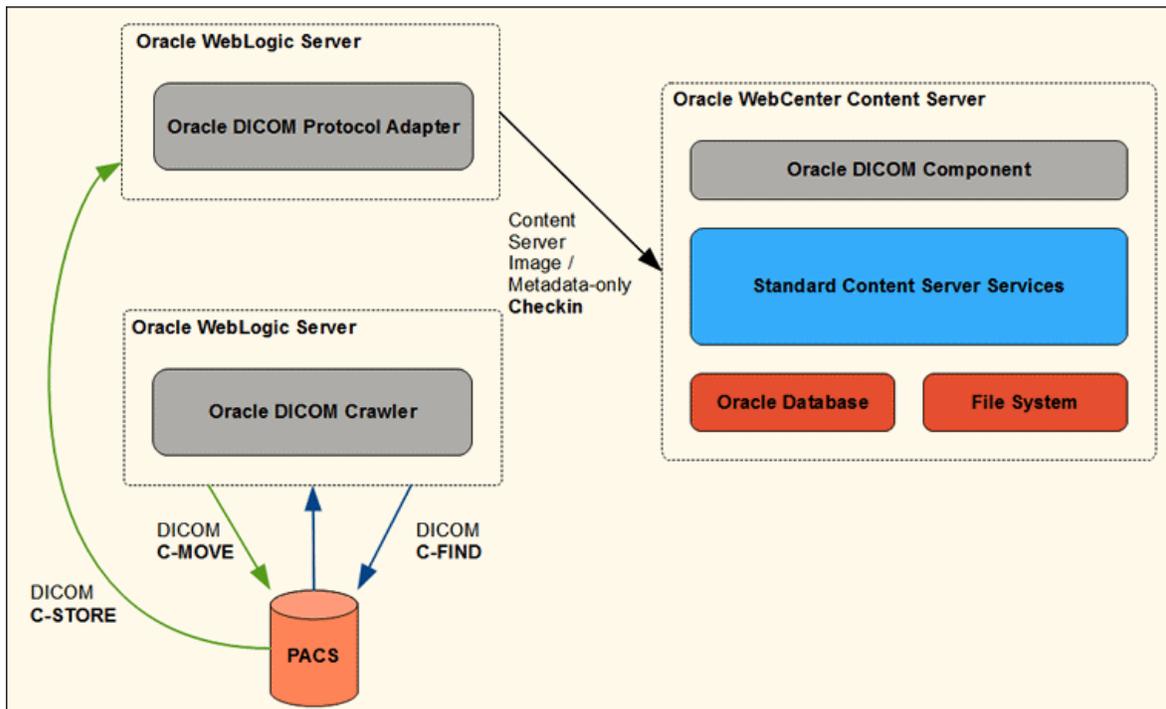
2.1.3 The Oracle DICOM WADO Proxy

The Oracle DICOM WADO proxy is a service that retrieves DICOM images via a WADO-formatted URL when the image metadata is in the Content Server and the images are in the PACS.

The WADO proxy is deployed as a JAX-RS Servlet application in Oracle WebLogic Server. The proxy listens for HTTP GET requests that correspond to the retrieval of a specific DICOM image from a specific PACS. Both the AE title of the PACS and the unique SOP instance UID of the image are encoded as part of the WADO URL. The format of the WADO URL is as follows:

http://*hostname*:*port*/dicom-net/resource/*AE_TITLE*/wado? requestType=WADO&studyUID=*study_uid*&seriesUID=*series_uid* &objectUID=*instance_uid*

When the proxy receives a WADO request like this, it issues a C-MOVE request to the corresponding PACS in order to retrieve the image. After the proxy receives the C-STORE request from the PACS with the DICOM image data corresponding to the C-MOVE it issued, the proxy passes the DICOM image data to the HTTP client that issued the WADO request.

The following figure depicts how the set of messages required to process a WADO request from a WADO client is exchanged between the various components. The figure also shows how this type of WADO URL is generated at checkin time for a DICOM image originating from a PACS.
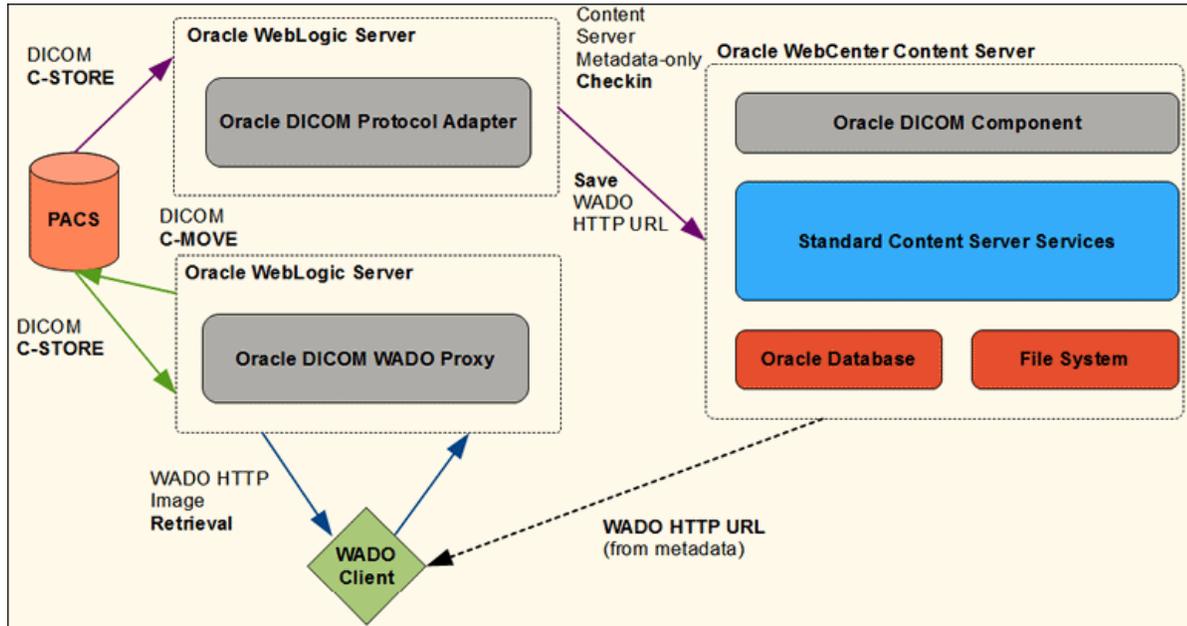
2.2 Deploying the Oracle DICOM Protocol Adapter

Several steps must be followed in order to deploy the Oracle DICOM protocol adapter:

- Install and deploy the OrdDicomLibrary.ear file, which contains the Java classes and Java libraries required to run the Oracle DICOM protocol adapter.

- Also install and deploy the OrdDicomAdapter.ear archive file, which contains the files for each component in the Oracle DICOM protocol adapter. See Deploying the Archive Files for details.

- Perform a standard deployment (as described in Standard Deployment) or an advanced deployment to choose individual components for installation (as described in Advanced Deployment).

This documentation assumes that you are familiar with the process for deploying applications on WebLogic Server. For complete documentation about application deployment, see *Oracle Fusion Middleware Deploying Applications to Oracle WebLogic Server*.

2.2.1 Deploying the Archive Files

The Oracle DICOM protocol adapter is distributed as two WebLogic enterprise archive files.

The OrdDicomLibrary.ear archive contains all the Java classes and other Java libraries required to run the adapter.

The OrdDicomAdapter.ear archive references the OrdDicomLibrary.ear archive file and contains the XML descriptors and configuration files that are necessary to provide a bare-bones deployment configuration for the three components of the Oracle DICOM protocol adapter.

To deploy the OrdDicomLibrary.ear archive to WebLogic, use the weblogic.Deployer tool by issuing a command similar to the following:

```
java weblogic.Deployer -upload -name OrdDICOM -source OrdDicomLibrary.ear
-targets [target_name] -adminurl [weblogic_admin_url] -user [weblogic_user]
-deploy -library
```

To deploy the individual components of the Oracle DICOM protocol adapter, extract the contents of the OrdDicomAdapter.ear archive using the Java jar tool. Then modify the XML configuration files before deploying, as described in Configuring the Oracle DICOM Protocol Adapter.

If the archive is extracted to a directory named oracle-dicom, the file structure of the directory appears as follows:

```
oracle-dicom/META-INF/MANIFEST.MF
oracle-dicom/META-INF/application.xml
oracle-dicom/META-INF/weblogic-application.xml
oracle-dicom/APP-INF/lib/dicom-net_adapter.jar
oracle-dicom/APP-INF/lib/dicom-net_adapter.rar
oracle-dicom/APP-INF/lib/dicom-net_crawler.jar
oracle-dicom/APP-INF/lib/dicom-net_crawler.rar
oracle-dicom/APP-INF/lib/dicom-net_wado.war
oracle-dicom/APP-INF/classes/ordim_adapter_config.xml
oracle-dicom/APP-INF/classes/ordim_crawler_config.xml
oracle-dicom/APP-INF/classes/ordim_wado_config.xml
```

The XML files in the oracle-dicom/APP-INF/classes directory are the files used to configure each of the Oracle DICOM protocol adapter's components. See Configuring the DICOM Adapter, Configuring the DICOM Crawler, and Configuring the DICOM WADO Proxy.

The archive files in the oracle-dicom/APP-INF/lib/ directory correspond to the modules comprising the Oracle DICOM protocol adapter.

The XML files in the oracle-dicom/META-INF/ directory correspond to the WebLogic deployment descriptors describing all the modules and resources used in the application.

2.2.2 Standard Deployment

Note:

The OrdDicomLibrary.ear library must be deployed to WebLogic and the Oracle DICOM component for WebCenter installed to Content Server before deploying the Oracle DICOM protocol adapter. See Deploying the Archive Files and Installing and Using the Oracle DICOM Component for WebCenter for details.

To deploy a single instance of each of the three components of the Oracle DICOM protocol adapter, edit the XML files in the oracle-dicom/APP-INF/classes/ directory.

After the configuration files are modified, deploy the application to WebLogic by issuing a command similar to the following command. In this example, it is assumed that the user extracted the OrdDicomLibrary.ear archive to a directory named oracle-dicom.

```
java weblogic.Deployer -upload -name OracleDicomAdapterSuite -source 'oracle-dicom' -
targets myserver -adminurl [weblogic_admin_url] -user [weblogic_user] -deploy
```

This command deploys the three components of the Oracle DICOM protocol adapter to WebLogic and starts them.

2.2.3 Advanced Deployment

Although the components of the Oracle DICOM protocol adapter are bundled together in a single enterprise archive, it is not necessary for all of the components to be deployed together at once. Multiple instances can be deployed of a particular component, or no instance at all can be deployed of a particular component.

The META-INF/application.xml file describes the modules that will be enabled at deployment time. The following is an example of this file:

```
<?xml version="1.0" encoding="UTF-8"?>
<application xmlns="http://java.sun.com/xml/ns/javaee"
             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
             xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://
java.sun.com/xml/ns/javaee/application_5.xsd"
             version="5">
    <!-- Oracle DICOM Protocol Adapter -->
    <module>
        <connector>/APP-INF/lib/dicom-net_adapter.rar</connector>
    </module>
    <module>
        <ejb>/APP-INF/lib/dicom-net_adapter.jar</ejb>
    </module>

    <!-- Oracle DICOM Crawler -->
    <module>
        <connector>/APP-INF/lib/dicom-net_crawler.rar</connector>
    </module>
    <module>
        <ejb>/APP-INF/lib/dicom-net_crawler.jar</ejb>
    </module>

    <!-- Oracle DICOM WADO Proxy -->
```

```
    <module>
        <web>
            <web-uri>/APP-INF/lib/dicom-net_wado.war</web-uri>
            <context-root>dicom-net</context-root>
        </web>
    </module>
</application>
```

In this example, all three components of the Oracle DICOM protocol adapter are enabled. To disable any of these components, delete (or comment out) the module references corresponding to the component to remove.

Both the DICOM adapter and the DICOM crawler components require at least two module definitions.

The *.rar archive is the ResourceAdapter archive for the component (there must be one of these for the component).

The *.jar is the MessageDrivenBean (MDB) archive for the component (there must be **at least** one of these for the component).

There is only a single module definition required per instance of the DICOM WADO proxy component (the *.war archive).

By default, there is an Oracle DICOM protocol adapter configuration file for each of the MDB archives and one for the WADO proxy archive, as shown in the following list:

- dicom-net_adapter.jar and ordim_adapter_config.xml

- dicom-net_crawler.jar and ordim_crawler_config.xml

- dicom-new_wado.war and ordim_wado_config.xml

Follow these steps to deploy multiple instances of any of the three components of the Oracle DICOM protocol adapter:

1. Create a copy of the appropriate MDB (*.jar) archive or WADO (*.war) archive.

2. Modify the copy of the archive as described in Modifying the Individual Component Archives.

3. Place the modified archive alongside the original in the APP-INF/lib/ directory.

4. Place the new XML configuration file (modified in step 2) in the APP-INF/classes/ directory.

5. Add a `<module>` entry in the META-INF/application.xml file for the new archive.

6. Deploy the components using the weblogic.Deployer utility (see Standard Deployment).

2.2.3.1 Modifying the Individual Component Archives

This section describes the necessary modifications for the different Oracle DICOM protocol adapter components.

2.2.3.1.1 DICOM adapter

To modify the adapter .jar archive, rename the MDB and update the name of the XML configuration file to use. To do this, edit the META-INF/ejb-jar.xml file from the archive and change the following parameter values (marked in ***bold italics***):

```
<message-driven>
    <display-name>DICOM_ADAPTER</display-name>
    <ejb-name>DICOM_ ADAPTER</ejb-name>
    […]
</message-driven>

<container-transaction>
    <method>
        <ejb-name>DICOM_ ADAPTER</ejb-name>
        […]
    </method>
    […]
</container-transaction>

<activation-config-property>
  <activation-config-property-name>adapterConfig</activation-config-property-name>
  <activation-config-property-value>ordim_adapter_config.xml</activation-config-property-value>
 </activation-config-property>
```

The META-INF/weblogic-ejb-jar.xml file from the archive must also be edited with changes to these additional parameter values (marked in ***bold italics***):

```
<weblogic-enterprise-bean>
    <ejb-name>DICOM_ ADAPTER</ejb-name>
    […]
</ weblogic-enterprise-bean>
```

2.2.3.1.2 DICOM crawler

In order to modify the *.jar archive to create a new instance of the Oracle DICOM crawler, the user must rename the MDB and update the name of the XML configuration file to use.

To do this, edit the META-INF/ejb-jar.xml file from the archive and change the following parameter values (marked in ***bold italics***):

```
<message-driven>
    <display-name>DICOM_CRAWLER</display-name>
    <ejb-name>DICOM_CRAWLER</ejb-name>
    […]
</message-driven>

<container-transaction>
    <method>
        <ejb-name>DICOM_CRAWLER</ejb-name>
        […]
    </method>
    […]
</container-transaction>

<activation-config-property>
  <activation-config-property-name>crawlerConfig</activation-config-property-name>
  <activation-config-property-value>ordim_crawler_config.xml</activation-config-property-value>
 </activation-config-property>
```

You must also edit the META-INF/weblogic-ejb-jar.xml file from the archive and change these additional parameter values (marked in ***bold italics***):

```
<weblogic-enterprise-bean>
    <ejb-name>DICOM_CRAWLER</ejb-name>
    […]
</ weblogic-enterprise-bean>
```

2.2.3.1.3 Oracle DICOM WADO proxy

To create a new instance of the WADO proxy, first modify the WADO .war archive. To do this, rename the servlet and update the name of the XML configuration file to use by editing the WEB-INF/web.xml file from the archive and change the following values (marked in ***bold italics***):

```
<context-param>
    <param-name>wadoConfig</param-name>
    <param-value>ordim_wado_config.xml</param-value>
 </context-param>


<servlet>
    <display-name>JAX-RS Servlet</display-name>
    <servlet-name>jersey</servlet-name>
    […]
</servlet>


<servlet-mapping>
    <servlet-name>jersey</servlet-name>
    <url-pattern>/*</url-pattern>
</servlet-mapping>
```

2.3 Configuring the Oracle DICOM Protocol Adapter

This section describes how to configure the three components of the Oracle DICOM protocol adapter:

- Configuring the DICOM Adapter

- Configuring the DICOM Crawler

- Configuring the DICOM WADO Proxy

2.3.1 Configuring the DICOM Adapter

The DICOM adapter can be configured by creating an XML file that adheres to the `ordim_adapter_config.xsd` schema as detailed in ordim_adapter_config.xsd.

This section describes the elements which are configurable in that schema.

2.3.1.1 adapter

Used to identify the DICOM host information for the DICOM adapter.

- **title**: specifies the DICOM AE title for the host.

- **bind**: specifies the port number the host will listen on for incoming DICOM assocations.

The following is an example of the **adapter** element.

```
<adapter useSSL="false">
    <title>UCM</title>
    <bind port="11112"/>
</adapter>
```

2.3.1.2 UCM

Used to identify how to connect to the WebCenter Content server with Content Server checkin properties.

- **connection**: specifies the parameters required for the DICOM adapter to connect to WebCenter Content.

  - **url**: the URL used to connect to the WebCenter Content server.

  - **login**: the login credentials (user name and password) for the connection to the WebCenter Content server. If the password is not supplied, it will be retrieved via the credential map for the default WebLogic security realm associated with the resource adapter for the DICOM adapter.

- **properties**: used to identify the Content Server checkin properties to use when checking in DICOM images using the DICOM adapter.

  - **property**: specifies the name and value of a Content Server checkin property. Valid properties include the following:

    - SECURITY_GROUP: a string value representing the name of the WebCenter Content security group to use for checkins.

    - DICOM_DOC_CHECKIN: a Boolean value representing whether or not to store an image representation of a DICOM image on checkin.

    - DICOM_DOC_TYPE: a string value representing the Content Server document type to use for DICOM image checkins.

    - DICOM_DOC_FOLDER: a string value representing the name of the Content Server folder where DICOM images should be stored.

    - DICOM_DOC_ORIGINAL: a Boolean value representing whether to retain the original DICOM image on checkin.

    - DICOM_DOC_PREVIEW: a Boolean value representing whether to generate an image preview of a DICOM image on checkin.

    - DICOM_XML_CHECKIN: a Boolean value representing whether to generate an XML-based representation of the metadata of a DICOM image on checkin.

    - DICOM_XML_TYPE: a string value representing the Content Server document type to use for DICOM XML metadata on checkin.

    - DICOM_XML_FOLDER: a string value representing the name of the Content Server folder where DICOM XML metadata will be stored.

    - DICOM_WADO_LINK_ENABLE: a Boolean value representing whether to generate a WADO URL on DICOM image checkin.

    - DICOM_WADO_LINK_PREFIX: a string value representing the base URL (the prefix string) for the WADO link generation.

The following is an example of the **UCM** element:

```
<UCM>
    <connection>
        <url location="http://hostname:16200/cs/idcplg"/>
```

```
        <login username="user" password="pass"/>
    </connection>
    <properties>
      <property name="SECURITY_GROUP"     value="DICOMGroup"/>
      <property name="DICOM_DOC_CHECKIN"  value="true"/>
      <property name="DICOM_DOC_TYPE"     value="DICOM"/>
      <property name="DICOM_DOC_FOLDER"   value="/Contribution Folders/DICOM"/>
      <property name="DICOM_DOC_ORIGINAL" value="true"/>
      <property name="DICOM_DOC_PREVIEW"  value="false"/>
      <property name="DICOM_XML_CHECKIN"  value="false"/>
      <property name="DICOM_XML_TYPE"     value="DICOMXML"/>
      <property name="DICOM_XML_FOLDER"   value="/Contribution Folders/DICOM"/>
      <property name="DICOM_WADO_LINK_ENABLE" value="true"/>
      <property name="DICOM_WADO_LINK_PREFIX" value="http://hostname:7001/dicom-net/resource"/>
    </properties>
</UCM>
```

2.3.1.3 DICOM

Specifies DICOM-specific configurations such as AE title to IP address/port mapping.

- **hosts**: specifies the set of DICOM clients which are permitted to connect to the adapter.

  - host: specifies the AE title, IP address, and port of a remote DICOM client to which the adapter can connect.

The following is an example of the DICOM element.

```
<DICOM>
    <hosts>
      <host title="TESTPACS" address="testpacs.hostname" port="11112"/>
      <host title="WADO" address="127.0.0.1" port="12347"/>
    </hosts>
</DICOM>
```

2.3.2 Configuring the DICOM Crawler

The Oracle DICOM crawler can be configured by creating an XML file that adheres to the ordim_crawler_config.xsd schema as detailed in ordim_crawler_config.xsd. This section describes those elements in the schema that are configurable.

2.3.2.1 crawler

Specifies the DICOM host configuration for the DICOM crawler.

- **title**: the DICOM AE title for the crawler.

The following is an example of the **crawler** element.

```
<crawler>
    <title>UCM_CRAWLER</title>
</crawler>
```

2.3.2.2 jobs

Represents a single crawler job.

- **target**: the AE title of the DICOM client to search.

- **dest**: the AE title of the DICOM host where DICOM images should be sent.

- **period**: the delay between job runs, including time units.

- **search**: the list of queries to perform, in order, during the job invocation. Queries are chained such that the output of one query serves as the base dataset for the next query.

  - **query**: represents a single query, specifying the name of the dataset and the DICOM search root to use.

The following is an example of the **jobs** element.

```
<jobs>
    <job>
      <target title=" PACS "/>
      <dest title="UCM"/>
      <period duration="15" timeUnit="s"/>
      <search>
        <query searchRoot="study" dataset="Q1"/>
        <query searchRoot="study" dataset="Q2"/>
        <query searchRoot="study" dataset="Q3"/>
      </search>
    </job>
</jobs>
```

2.3.2.3 hosts

Specifies the set of DICOM clients to which the crawler is permitted to connect.

- **host**: specifies the title, IP address, and port of a remote DICOM client to which the crawler can connect.

The following is an example of the **hosts** element.

```
<hosts>
    <host title="UCM" address="localhost" port="12345"/>
    <host title="PACS " address="pacs.hostname" port="11112"/>
</hosts>
```

2.3.2.4 datasets

Specifies the set of DICOM datasets to use as part of the queries for the crawler jobs. Each dataset is assigned a name that can be referenced in the **query** tag for a job.

- dataset: specifies a single dataset.

  - value: a DICOM tag and its corresponding value, as a tag/value pair.

The following is an example of the **datasets** element.

```
<datasets>
    <dataset name="Q1">
      <value tag="00080052">STUDY</value>
      <value tag="0020000D">*</value>
    </dataset>
    <dataset name="Q2">
      <value tag="00080052">SERIES</value>
      <value tag="0020000E">*</value>
    </dataset>
    <dataset name="Q3">
      <value tag="00080052">IMAGE</value>
      <value tag="00080018">*</value>
```

```
        </dataset>
    </datasets>
```

2.3.3 Configuring the DICOM WADO Proxy

The Oracle DICOM WADO proxy can be configured by creating an XML file that adheres to the ordim_wado_config.xsd schema as detailed in ordim_wado_config.xsd. This section describes those elements in the schema that are configurable.

2.3.3.1 wado

Used to identify the DICOM host information for the Oracle DICOM WADO proxy.

- **wado**: identifies the DICOM host information for the proxy.

  - **title**: specifies the DICOM AE title for the host.

  - **bind**: specifies the port number the host will listen on for incoming DICOM assocations.

The following is an example of the **wado** element.

```
<wado useSSL="false">
    <title>WADO</title>
    <bind port="12347"/>
</wado>
```

2.3.3.2 hosts

Specifies the set of DICOM clients to which the proxy is permitted to connect.

- **host**: specifies the AE title, IP address, and port of a remote DICOM client to which the proxy can connect.

The following is an example of the **hosts** element.

```
<hosts>
    <host title="UCM" address="localhost" port="12345"/>
    <host title="PACS " address="pacs.hostname" port="11112"/>
</hosts>
```

2.4 Sample Configuration Scenarios

Two typical scenarios were discussed in Usage Scenarios. This section provides the details for configuring those scenarios.

In each scenario, the following steps must be taken:

- Configure the Oracle DICOM crawler with the connection details of all PACS from which the user will import images. See Configuring the DICOM Crawler for details.

- Set up the DICOM query that serves as a filter restricting which images will be returned.

- Configure the DICOM component by adding metadata mapping in the oracle_dicom_config.hda file UCM_DICOM_MAP ResultSet. Add a metadata mapping for each of the DICOM metadta elements the user wants to retain on image import. See Editing the oracle_dicom_config.hda File for details.

2.4.1 Scenario 1: Importing Images and Image Metadata

This scenario assumes the user will consolidate all DICOM images and image metadata from PACS into Content Server.

To accomplish this, the following properties must be set in the DICOM adapter configuration:

```
<property name="DICOM_DOC_CHECKIN" value="true"/>
<property name="DICOM_DOC_ORIGINAL" value="true"/>
```

These settings ensure that when an image is imported from a PACS, the metadata is extracted from the image and populated into Content Server metadata fields and the original image is retained in Content Server. After an image is imported, a user can search for the image based on its metadata and retrieve it directly from Content Server or via the DICOM adapter.

2.4.2 Scenario 2: Importing Only Image Metadata

This scenario assumes a user will consolidate only DICOM image metadata from PACS into Content Server.

The user must configure the WADO proxy so it has the connection details for all the PACS (that is, the same set as the DICOM crawler). Set the properties in the DICOM adapter configuration as follows, setting *hostname:port* to the hostname and port set for the WADO proxy:

```
<property name="DICOM_DOC_CHECKIN" value="true"/>
<property name="DICOM_DOC_ORIGINAL" value="false"/>
<property name="DICOM_WADO_LINK_ENABLE" value="true"/>
<property name="DICOM_WADO_LINK_PREFIX" value="http://hostname:port/dicom-net/resource"/>
```

The first two properties ensure that only the DICOM image metadata is retained by Content Server in the populated metadata fields even though the image is checked into Content Server on import.

The last two properties ensure that a WADO link is generated and added as one of the Content Server metadata fields. This link is a URL that points to the WADO proxy and includes all the information required by the proxy to retrieve the original image from the PACS where the image is stored. Thus the user can always access the original image by visiting the WADO link URL.

In this scenario, the user may still search for imported images based on metadata (the same as scenario 1), but the original images themselves are no longer available directly from Content Server. Instead, the user can search for an image and look up its corresponding WADO link in order to retrieve it (when searching for an image via Content Server directly).

Or the user can issue a DICOM query via the DICOM adapter to retrieve an image. The adapter will transparently retrieve the original image from the PACS via the WADO link and pass it on to the DICOM client that issued the request.

3

Installing and Configuring the Oracle DICOM Component for WebCenter

The Oracle DICOM component for WebCenter is a collection of services that allows users to check in and process DICOM content in Content Server.

This chapter discusses how to install and configure the DICOM component. It covers the following topics:

- About the Oracle DICOM Component for WebCenter
- Installing and Using the Oracle DICOM Component for WebCenter
- Customizing the Component

3.1 About the Oracle DICOM Component for WebCenter

The DICOM component contains a filter that allows users to check in DICOM images into the Content Server repository. It also allows users to create DICOM images from a non-DICOM image source and corresponding DICOM XML metadata.

The following figure illustrates the case when a user checks in a DICOM source image to Content Server.
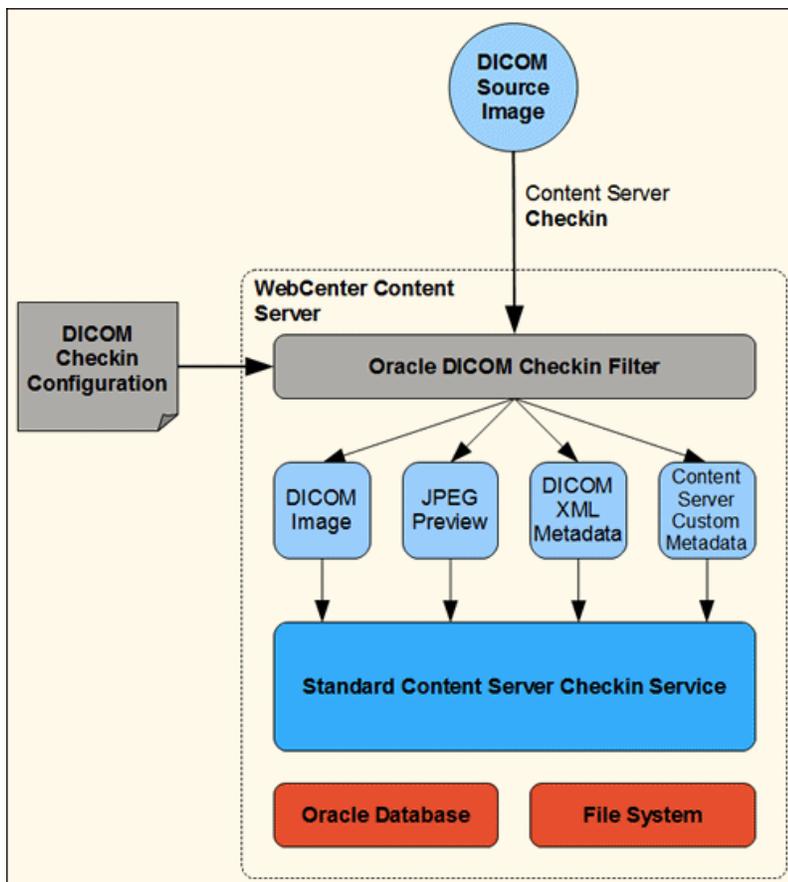
Depending on the checkin configuration options, the filter will perform one or more of the following processing steps:

- Populate Content Server custom metadata fields

- Retain the original DICOM source image

- Generate an image preview, such as a .jpeg file

- Generate DICOM XML metadata

After the filter completes its processing, it passes on the result data to the standard Content Server checkin service.

The following figure shows a similar checkin scenario, but in this case, the filter first creates the source DICOM image by combining a non-DICOM source image and DICOM metadata in XML format.
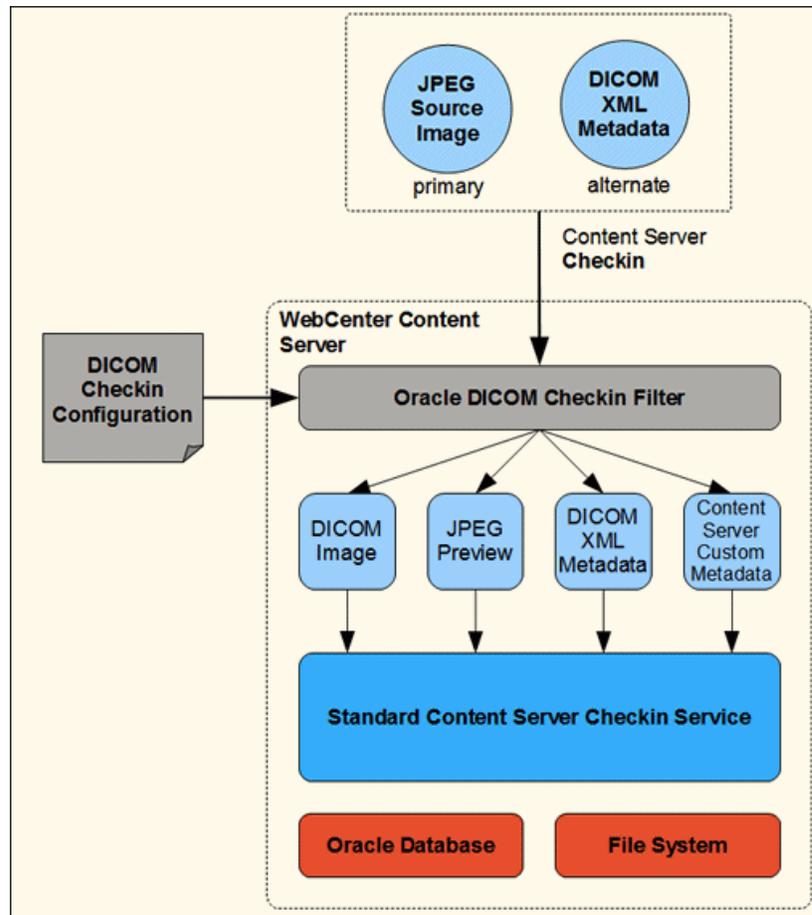
3.2 Installing and Using the Oracle DICOM Component for WebCenter

This section describes how to install and enable the component on the Content Server as well as describing prerequisites for using the component after it is enabled. The following topics are discussed:

- Installing and Enabling the Component

- Prerequisites for Use

3.2.1 Installing and Enabling the Component

The Oracle DICOM component for WebCenter is distributed as a single zip archive file named `OracleDicom.zip`.

Follow these instructions to install and enable the component on the Content Server instance:

1. Click **Administration** then **Admin Server** on the Main Menu.

2. Click the **Advanced Component Manager** link in the text at the top of the screen.

3. Click **Browse** next to the Install New Component prompt. Browse to the location where the files are stored and select the OracleDicom.zip file.

4. Click **Install**. A screen displays what elements will be installed. Click **Continue**.

5. After the files are installed, an information page is displayed. Click the link to enable the component and restart the Content Server.

After enabling the component, perform the following actions to use the Oracle DICOM component for WebCenter:

- Prerequisites for Use

- Customizing the Component

After initializing the component, you can optionally configure it for use, as described in Editing the oracle_dicom_config.hda File.

3.2.2 Prerequisites for Use

Several steps should be taken before using the Oracle DICOM component for WebCenter for the first time:

1. Enable the FrameworkFolders component.

2. Modify the following parameters in the *domain*/ucm/cs/config/config.cfg file.

   - Set `AllowPrimaryMetaFile` to TRUE.

   - `SocketHostAddressSecurityFilter`: add the IP address of the WebLogic server where the DICOM adapter is installed.

   - `IntradocServerPort`: set the value to a port number the IDC protocol uses (normally 4444).

   - Set `IsAutoNumber` to TRUE.

3. Verify that the oracle_dicom_config.hda file and oracle_dicom_meta_init.hda file exist in *domain*/ucm/cs/data/OracleDicom. Note that this directory does not exist prior to the first installation. You must create this directory and copy the .hda files to it.

4. Verify that the following .jar files exist in the Java CLASSPATH. These files are shipped in the /lib directory of the OracleDicom installation directory. Link or copy jai*./jar to the *domain*/lib/ directory then restart both the WebLogic and the WebCenter Content servers.

   - aurora.jar

   - orddcmmt.jar

   - clibwrapper_jiio.jar

   - ordimdcm.jar

   - jai_codec.jar

   - ordimimg.jar

   - jai_core.jar

   - ordim.jar

   - jai_imageio.jar

- xmlparserv2.jar

3.3 Customizing the Component

The oracle_dicom_config.hda and oracle_dicom_meta_init.hda files exist in the /data directory of the OracleDicom component installation directory. These files can be edited to customize the component:

- The oracle_dicom_config.hda file is the OracleDicom component configuration file. See Editing the oracle_dicom_config.hda File for details.

- The oracle_dicom_meta_init.hda file is the initialization file that lists the WebCenter Content custom metadata fields, document types, security groups, tables and folders which are created when the component is installed and enabled. These custom metadata fields are used by the component for mapping DICOM metadata tags to custom Content Server fields. Document types and security groups are used to check in DICOM images. See The oracle_dicom_meta_init.hda File for more details.

The user can change these files as needed and link or copy them to the *domain*/ucm/cs/data/OracleDicom/ directory before enabling the component.

3.3.1 Editing the oracle_dicom_config.hda File

The configuration file is used to configure the appearance and behavior of the Oracle DICOM component for WebCenter. The file is formatted as a Oracle WebCenter Content Application Data File and is stored on the instance at *domain*/ucm/cs/data/OracleDicom.The file is named oracle_dicom_config.hda.

The following items can be created beforehand in WebCenter Content then referenced and used in the configuration file as needed:

- Security group for DICOM images or metadata

- Document types for the following:

  - DICOM images

  - DICOM XML metadata

  - Newly-created DICOM images

- Custom metadata fields mapping to DICOM metadata tags

- Folders where the DICOM images or metadata are stored

- Custom metadata fields mapping to the generated WADO link

This section describes the ResultSet entries in the oracle_dicom_config.hda file which can be configured.

- UCM_DICOM_MAP

- ORACLE_DICOM_FILTER_ALLOWED_GROUPS

- ORACLE_DICOM_FILTER_ALLOWED_TYPES

- ORACLE_DICOM_FILTER_ALLOWED_CREATEDDICOM_TYPES

- ORACLE_DICOM_FILTER_ALLOWED_XML_TYPES

- ORACLE_DICOM_IMAGE_CONFIG

- ORACLE_DICOM_XML_CONFIG

- ORACLE_DICOM_EXTRA_CONFIG

3.3.1.1 UCM_DICOM_MAP

This ResultSet specifies how to map Content Server custom metadata fields to DICOM metadata attributes. It includes three columns: the name of the Content Server custom metadata field, the DICOM metadata tag, and a flag to indicate whether to extract this tag value from the DICOM image during checkin.

For example, to map the custom Content Server metadata field `DICOM_STUDY_INSTANCE_UID` to DICOM attribute `(0020, 000D)` and Content Server metadata field `DICOM_PATIENT_NAME` to DICOM attribute `(0010, 0010)`, the configuration file would contain the following lines.

```
@ResultSet UCM_DICOM_MAP
3
ucm_metadata_name
dicom_metadata_tag
extracted
DICOM_STUDY_INSTANCE_UID
0020000D
1
DICOM_PATIENT_NAME
00100010
1
@end
```

3.3.1.2 ORACLE_DICOM_FILTER_ALLOWED_GROUPS

This ResultSet specifies the set of security groups for which checked-in DICOM images will be processed by the DICOM component. The following example shows how to specify this ResultSet:

```
@ResultSet ORACLE_DICOM_FILTER_ALLOWED_GROUPS
1
security_group_name
DICOMGroup
@end
```

3.3.1.3 ORACLE_DICOM_FILTER_ALLOWED_TYPES

This ResultSet specifies the set of document types that will be processed by the DICOM component upon DICOM image checkin. The following example shows how to specify this ResultSet.

```
@ResultSet ORACLE_DICOM_FILTER_ALLOWED_TYPES
1
doc_type_name
DICOM
@end
```

3.3.1.4 ORACLE_DICOM_FILTER_ALLOWED_CREATEDDICOM_TYPES

This ResultSet specifies the set of document types where the non-DICOM source images checked in will be processed by the DICOM component for DICOM image creation.

These types are specified during checkin when the user wants to create a DICOM image from a non-DICOM image source (such as JPEG) with the corresponding DICOM XML metadata. The following is an example of this ResultSet.

```
@ResultSet ORACLE_DICOM_FILTER_ALLOWED_CREATEDDICOM_TYPES
1
doc_type_name
CREATEDDICOM
@end
```

3.3.1.5 ORACLE_DICOM_FILTER_ALLOWED_XML_TYPES

This ResultSet specifies the set of document types for which checked-in DICOM images will be processed by the DICOM component for DICOM XML metadata generation.

These types are specified during checkin when the user wants to create a DICOM XML metadata document based on the source DICOM image that the user checks in. The following is an example of this ResultSet.

```
@ResultSet ORACLE_DICOM_FILTER_ALLOWED_XML_TYPES
1
doc_type_name
DICOMXML
@end
```

3.3.1.6 ORACLE_DICOM_IMAGE_CONFIG

This ResultSet specifies a series of attribute-value pairs that configure the behavior of the DICOM component during DICOM image checkin. The user may set values for the following attributes:

- DICOM_PREFIX: a string representing the prefix of the document title given to a DICOM image on checkin.

- DICOM_SUFFIX: a string representing the suffix of the document title given to a DICOM image on checkin.

- IF_DICOM_CHECKIN_PACS: a Boolean value representing whether to keep the original DICOM image in the case that the image is checked in from the DICOM adapter.

- IF_DICOM_CHECKIN_FILE: a Boolean value representing whether to keep the original DICOM image in the case that the image is checked in from the Content Server web interface.

- IF_THUMBNAIL_CHECKIN: a Boolean value representing whether to generate a non-DICOM preview image of the original DICOM image checked in.

- IF_THUMBNAIL_PRIMARY: a Boolean value representing whether to make the preview image the primary file for the document in the case where the original DICOM image checked in is not retained.

- IF_FILL_CUSTOM_METADATA: a Boolean value representing whether to extract metadata from the DICOM image checked in for the purpose of populating the custom Content Server metadata fields corresponding to the DICOM tags.

- THUMBNAIL_SUFFIX: a string representing the suffix of the document title given to a preview image generated on checkin of a DICOM image.

- THUMBNAIL_VERB: a string representing how to generate the image preview from the DICOM image checked in. For example, to generate a JPEG image preview, the user could specify the value `fileFormat=jpeg` for this attribute.

The following is an example specifying values for these options.

```
@ResultSet ORACLE_DICOM_IMAGE_CONFIG
2
attribute_name
attribute_value
DICOM_PREFIX
dicom
DICOM_SUFFIX
.dcm
IF_DICOM_CHECKIN_PACS
true
IF_DICOM_CHECKIN_FILE
true
IF_THUMBNAIL_CHECKIN
true
IF_THUMBNAIL_PRIMARY
false
IF_FILL_CUSTOM_METADATA
true
THUMBNAIL_SUFFIX
.jpg
THUMBNAIL_VERB
fileFormat=jpeg
@end
```

3.3.1.7 ORACLE_DICOM_XML_CONFIG

This ResultSet specifies a series of attribute-value pairs that configure the behavior of the DICOM component during DICOM image check-in for DICOM XML metadata generation. The user may set values for the following attributes:

- SUFFIX: a string representing the suffix of the document title given to a DICOM XML metadata file generated on checkin.

- IF_DICOM_CHECKIN: a Boolean value representing whether to keep the original DICOM image in the case when a DICOM XML metadata file is generated.

- IF_XML_PRIMARY: a Boolean value representing whether to make the DICOM XML metadata file the primary file for the document in the case where the original DICOM image checked in is not retained.

- IF_FILL_CUSTOM_METADATA: a Boolean value representing whether to extract metadata from the DICOM image checked in for the purpose of populating the custom Content Server metadata fields corresponding to the DICOM tags.

The following is an example of this ResultSet:

```
@ResultSet ORACLE_DICOM_XML_CONFIG
2
attribute_name
attribute_value
SUFFIX
.xml
IF_DICOM_CHECKIN
false
IF_XML_PRIMARY
```

```
true
IF_FILL_CUSTOM_METADATA
false
@end
```

3.3.1.8 ORACLE_DICOM_EXTRA_CONFIG

This ResultSet specifies a set of attribute-value pairs used by the DICOM component which are not defined in the other configuration ResultSet sections. At this time, there is only one valid attribute: `OracleDicomWADOLink`. The value of this attribute is a string that specifies the name of the custom Content Server metadata field where the generated WADO link will be stored on DICOM image check-in.

Below is an example of this ResultSet:

```
@ResultSet ORACLE_DICOM_EXTRA_CONFIG
2
attribute_name
attribute_value
OracleDicomWADOLink
ORACLE_DICOM_WADO_LINK
@end
```

3.3.2 The oracle_dicom_meta_init.hda File

The oracle_dicom_meta_init.hda file includes the ResultSet UCM_DICOM_META_LIST, which specifies the custom metadata fields to be created when OracleDicom component is installed and enabled.

The ResultSet UCM_DICOM_META_LIST includes three columns:

- ucm_dicom_metadata_name

- ucm_dicom_metadata_caption

- ucom_dicom_metadata_type

These list the caption and the type of the custom metadata to be created.

For example, to create the custom metadata with the name DICOM_STUDY_DATE, the caption "DICOM Study Date" and the type Date, the ResultSet would contain the following lines:

```
@ResultSet UCM_DICOM_META_LIST
3
ucm_dicom_metadata_name
ucm_dicom_metadata_caption
ucm_dicom_metadata_type
DICOM_STUDY_DATE
DICOM Study Date
Date
@end
```

4

Performance Tuning Tips

This chapter discusses ways to obtain the optimum performance from this feature.

This chapter includes the following:

- Performance and Usage

- Performance Tuning

4.1 Performance and Usage

The performance of the feature can be directly affected by the different usage scenarios for implementation. This section discusses different elements that can affect performance.

4.1.1 Performance Metrics

The following factors can be used to measure performance:

- Number of images per second (for an image of fixed size) that the DICOM component can insert into Content Server for each checkin configuration option.

- Number of images per second (for an image of fixed size) that the DICOM adapter can serve to a DICOM client.

- Number of images per second (for an image of fixed size) that the DICOM adapter can retrieve from a DICOM client.

4.1.2 Factors Affecting Performance

The performance of this feature can be impacted by the following elements:

- The response time of Content Server

- The response time of the DICOM client

- The number of managed DICOM clients

- The polling interval of each managed client

- DICOM image size

4.1.3 Usage Scenarios and Performance

The following list describes several key usage scenarios, with a discussion following about the performance impacts of each scenario:

- The DICOM Crawler queries the set of managed DICOM clients (such as PACS), retrieves any new images, and the DICOM adapter uses the DICOM component to check the images into Content Server.

- The DICOM adapter responds to a query from another DICOM client (such as a DICOM viewer), searches for relevant images and metadata stored in Content Server, and sends the result set back to the DICOM client.

- A WebCenter Content application uses the DICOM component to directly check in a DICOM image to the Content Server.

- A WebCenter Content application uses the DICOM component to create a new DICOM image by checking in a non-DICOM source image (such as a .jpeg file) and the corresponding DICOM XML metadata to the Content Server.

- A WebCenter Content application queries the Content Server directly to search and retrieve DICOM images checked in using the DICOM component.

- An HTTP client queries the WADO proxy to retrieve a DICOM image where Content Server only manages image metadata (but where the original image is stored in the PACS) via a WADO-formatted URL.

The following list discusses the performance impacts of the different scenarios:

- Scenario 1: the performance is impacted by all of the performance elements listed above. In this case, the DICOM Crawler and DICOM adapter must communicate with both DICOM clients and WebCenter Content. Therefore, performance will be impacted by the responsiveness of the systems. The DICOM component must process each of the images added so the size of the images will impact processing such as thumbnail generation, metadata extraction, and check-in time. In addition, there may be many managed DICOM clients and the number of these clients will impact the load on the system (as well as how frequently these clients are polled).

- Scenario 2: the DICOM adapter must issue a query to the content server, retrieve any relevant DICOM image data or metadata, and send this result set back to the DICOM client. The performance is primarily impacted by the content server response time (both search response and data exchange) and the DICOM client response time. The DICOM image size will also play a role because the image data must be transferred between the Content Server and the DICOM client.

- Scenarios 3 and 4: the DICOM component must process an incoming DICOM image designated for check-in (where in case 4 the DICOM image is generated first). The service is responsible for processing the DICOM image. For example depending on configuration options and service parameters, the service must extract metadata from and generate a thumbnail image of the given DICOM image. The primary performance considerations in this scenario are Content Server response time and DICOM image size.

- Scenario 5: a WebCenter Content application queries Content Server directly to retrieve DICOM image content (or metadata) that has been checked in by the DICOM component (or indirectly from the DICOM adapter). The performance considerations are similar to retrieving any type of image from Content Server.

- Scenario 6: an HTTP client queries the WADO proxy to retrieve a DICOM image stored in a DICOM client (such as a PACS). In this case, the WADO proxy will first issue a DICOM request to retrieve the image from the DICOM client and then will return the image data to the HTTP client in response to its request. The

performance for this type of request is influenced by the responsiveness of the DICOM client and the size of the image.

4.2 Performance Tuning

One tunable performance attribute is the managed DICOM client polling interval (see Configuring the DICOM Crawler). For each managed DICOM client, this interval indicates how long the DICOM crawler must wait before querying the DICOM client for new images since the time the last query was initiated. As the value for this parameter is lowered, the load on the system, network, and DICOM client increases due to the increased frequency that the DICOM client is queried.

As this parameter is tuned, the indexing interval for Content Server may need to be modified in order to make the new images that have been checked in available for search.

Another tunable performance attribute are the options controlling what data is retained or generated by the DICOM component on checkin. Options include:

- Retain the original DICOM image

- Metadata-only checkin with no image checked into Content Server. Only Content Server metadata fields are populated.

- Image preview generation

- DICOM XML metadata generation

The metadata-only checkin option yields the best performance for checkin, but may yield sub-optimal performance for image retrieval where it would be better if Content Server stores the original image). In either case, image preview generation and DICOM XML metadata generation will only impact checkin performance.

4.2.1 Tuning Responsibilities

The Content Server administrator (or user) who manages the DICOM adapter suite should tune the managed DICOM client polling interval to balance responsiveness with the load of managed DICOM clients. Tuning cannot be performed automatically because the user may have a preference for how much load this component should impose on DICOM clients in the given setup.

The administrator or user should also tune the checkin options for DICOM images. This value cannot be tuned automatically because the user may require the usage of image preview and/or DICOM XML metadata generation features, or specific storage concerns may need to be addressed for the site where the product is installed.

Oracle DICOM Protocol Adapter XML Schemas

This appendix provides the XML schemas needed for the components in the Oracle DICOM protocol adapter.

This appendix includes the following:

- ordim_adapter_config.xsd
- ordim_crawler_config.xsd

A.1 ordim_adapter_config.xsd

The following schema is used when configuring the DICOM adapter.

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:include schemaLocation="ordim_common.xsd"/>

  <xs:complexType name="dicom_ucm_metadata_pairing_t">
    <xs:attribute name="tag" type="dicom_tag_t" use="required"/>
    <xs:attribute name="field" type="ucm_field_t" use="required"/>
  </xs:complexType>

  <xs:complexType name="ucm_property_t">
    <xs:attribute name="name" type="ucm_field_t" use="required"/>
    <xs:attribute name="value" type="ucm_field_value_t" use="required"/>
  </xs:complexType>

  <xs:complexType name="dicom_ucm_metadata_map_t">
    <xs:sequence>
      <xs:element name="entry" type="dicom_ucm_metadata_pairing_t" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="dicom_tag_set_t">
    <xs:sequence>
      <xs:element name="tag" type="dicom_tag_t" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="ucm_properties_map_t">
    <xs:sequence>
      <xs:element name="property" type="ucm_property_t" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="ucm_config_t">
    <xs:sequence>
      <xs:element name="connection" type="ucm_connection_info_t"/>
      <xs:element name="properties" type="ucm_properties_map_t"/>
    </xs:sequence>
  </xs:complexType>
```

```
<xs:complexType name="dicom_config_t">
  <xs:sequence>
    <xs:element name="hosts" type="dicom_host_config_t"/>
    <xs:element name="wado" minOccurs="0">
      <xs:complexType>
        <xs:attribute name="baseURL" type="url_t" use="required"/>
      </xs:complexType>
    </xs:element>
    <xs:element name="selectedTags" type="dicom_tag_set_t"/>
    <xs:element name="tagMap" type="dicom_ucm_metadata_map_t"/>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="ordim_adapter_config_t">
  <xs:sequence>
    <xs:element name="adapter" type="dicom_local_host_t"/>
    <xs:element name="UCM" type="ucm_config_t"/>
    <xs:element name="DICOM" type="dicom_config_t"/>
  </xs:sequence>
</xs:complexType>

<xs:element name="ordim_ucm_adapter_config" type="ordim_adapter_config_t"/>

</xs:schema>
```

A.2 ordim_crawler_config.xsd

The following schema is used when configuring the DICOM Crawler.

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:include schemaLocation="ordim_common.xsd"/>

  <xs:simpleType name="dicom_value_t">
    <xs:restriction base="string_t"/>
  </xs:simpleType>

  <xs:complexType name="dicom_tag_value_t">
    <xs:simpleContent>
      <xs:extension base="dicom_value_t">
        <xs:attribute name="tag" type="dicom_tag_t" use="required"/>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>

  <xs:complexType name="dicom_dataset_t">
    <xs:sequence>
      <xs:element name="value" type="dicom_tag_value_t" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="named_dicom_dataset_t">
    <xs:complexContent>
      <xs:extension base="dicom_dataset_t">
        <xs:attribute name="name" type="nonempty_string_t" use="required"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>

  <xs:simpleType name="dicom_search_root_t">
    <xs:restriction base="string_t">
      <xs:enumeration value="patient"/>
      <xs:enumeration value="study"/>
      <xs:enumeration value="no_root"/>
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="time_unit_t">
```

```
      <xs:restriction base="string_t">
        <xs:enumeration value="ms"/>
        <xs:enumeration value="s"/>
        <xs:enumeration value="h"/>
        <xs:enumeration value="m"/>
        <xs:enumeration value="d"/>
      </xs:restriction>
    </xs:simpleType>

    <xs:complexType name="dicom_query_t">
      <xs:attribute name="searchRoot" type="dicom_search_root_t"/>
      <xs:attribute name="dataset" type="nonempty_string_t"/>
    </xs:complexType>

    <xs:complexType name="dicom_query_list_t">
      <xs:sequence>
        <xs:element name="query" type="dicom_query_t" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>

    <xs:complexType name="crawler_period_t">
      <xs:attribute name="start" type="nonempty_string_t"/>
      <xs:attribute name="duration" type="integer_t" use="required"/>
      <xs:attribute name="timeUnit" type="time_unit_t"/>
    </xs:complexType>

    <xs:complexType name="crawler_job_t">
      <xs:sequence>
        <xs:element name="target" type="dicom_simple_host_t"/>
        <xs:element name="dest" type="dicom_simple_host_t"/>
        <xs:element name="period" type="crawler_period_t"/>
        <xs:element name="search" type="dicom_query_list_t"/>
      </xs:sequence>
    </xs:complexType>

    <xs:complexType name="crawler_job_list_t">
      <xs:sequence>
        <xs:element name="job" type="crawler_job_t" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>

    <xs:complexType name="dicom_dataset_set_t">
      <xs:sequence>
        <xs:element name="dataset" type="named_dicom_dataset_t" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>

    <xs:complexType name="ordim_crawler_config_t">
      <xs:sequence>
        <xs:element name="crawler" type="dicom_local_client_t"/>
        <xs:element name="jobs" type="crawler_job_list_t"/>
        <xs:element name="hosts" type="dicom_host_config_t"/>
        <xs:element name="datasets" type="dicom_dataset_set_t"/>
      </xs:sequence>
    </xs:complexType>

    <xs:element name="ordim_crawler_config" type="ordim_crawler_config_t"/>

</xs:schema>
```

A.3 ordim_wado_config.xsd

The following schema is used when configuring the DICOM WADO proxy.

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:include schemaLocation="ordim_common.xsd"/>
```

```
            <xs:complexType name="ordim_wado_config_t">
              <xs:sequence>
                <xs:element name="wado" type="dicom_local_host_t"/>
                <xs:element name="hosts" type="dicom_host_config_t"/>
              </xs:sequence>
            </xs:complexType>

            <xs:element name="ordim_wado_config" type="ordim_wado_config_t"/>

          </xs:schema>
```