OpenMRS Radiology Module

OpenMRS Radiology Module is a module adding capabilities of a Radiology Information System (RIS) onto OpenMRS. This module connects the open source enterprise electronic medical record system OpenMRS (http://www.openmrs.org) with the open source clinical image and object management system dcm4chee (http://www.dcm4che.org).

Project status

OpenMRS Radiology Module is not yet officially released to the OpenMRS modules (https://modules.openmrs.org). The API and UI are not yet stable and subject to frequent changes.

Table of Contents

1. Overview
2. Installation
   2.1. "Manual"
   2.2. Devops
3. Deployment
   3.1. Module Dependencies
   3.2. Module Package
4. Configuration
   4.1. OpenMRS Radiology Module
   4.2. Dcm4chee

1. Overview

The most important URLs are

- Source code: https://github.com/openmrs/openmrs-module-radiology
- Issues: https://issues.openmrs.org/browse/RAD
- Wiki: https://wiki.openmrs.org/display/docs/Radiology+Module

- Documentation: http://github.com/teleivo/openmrs-module-radiology-docs

Host and Port configuration

This guide assumes that your

- OpenMRS installation can be reached at http://localhost:8080/openmrs/
- dcm4chee listens to http://localhost:8081/dcm4chee-web3

Please adjust the URLs accordingly.

2. Installation

To setup OpenMRS with the OpenMRS Radiology Module and dcm4chee you have two main options

2.1. "Manual"

Follow installation instructions of both projects

- https://dcm4che.atlassian.net/wiki/display/ee2/Installation
- https://wiki.openmrs.org/display/docs/Installing+OpenMRS

and execute all the necessary steps yourself.

2.2. Devops

Follow

- https://github.com/teleivo/puppet-openmrs-radiologydcm4chee

which will do most of the steps for you.

You will need a stable internet connection. The setup is downloading all necessary archives (a few hundred MBs) from different sites that may or may not be responding well in your area. People have reported timeout issues. If this does not work out for you I suggest you go the "manual" route.

3. Deployment

Deployment of the Radiology Module consists of deploying all necessary dependencies and then the module itself into OpenMRS.

You need to login to your OpenMRS instance as administrative user and go to the Administration, Manage Modules page and add the modules shown in the following sections.

3.1. Module Dependencies

The Radiology Module currently depends on the OpenMRS EMRAPI Module (https://github.com/openmrs/openmrs-module-emrapi) version 1.13 which itself has a bunch of dependencies.

You can get the dependencies at OpenMRS modules (https://modules.openmrs.org).

The following list is a suggested order in which we have managed to successfully deploy the necessary dependencies:

1. providermanagement - version 2.3
2. uiframework - version 3.4
3. uilibrary - version 1.5
4. emrapi - version 1.13
5. event - version 2.2.1
6. reporting - version 0.9.8.1
7. htmlwidgets - version 1.7.0
8. calculation - version 1.1
9. serialization.xstream - version 0.2.9
10. metadatasharing - version 1.1.9
11. metadatamapping - version 1.0.1

Do not worry about alerts you might get due to a modules dependencies not being met. Just go ahead and deploy its dependencies and hit the start button of the module which created the alert. You might also be

> able to add newer versions of the modules listed here.

3.2. Module Package

Since there has not been an official release yet you need to compile the Radiology Module yourself.

Please follow the instructions at https://github.com/openmrs/openmrs-module-radiology.

4. Configuration

At this point I expect you to have OpenMRS 1.11.4 installed, OpenMRS Radiology Module deployed and dcm4chee up and running.

Whats missing is the connection (DICOM, HL7) between OpenMRS Radiology Module and the dcm4chee PACS and the setup of the DICOM web viewer in the PACS.

4.1. OpenMRS Radiology Module

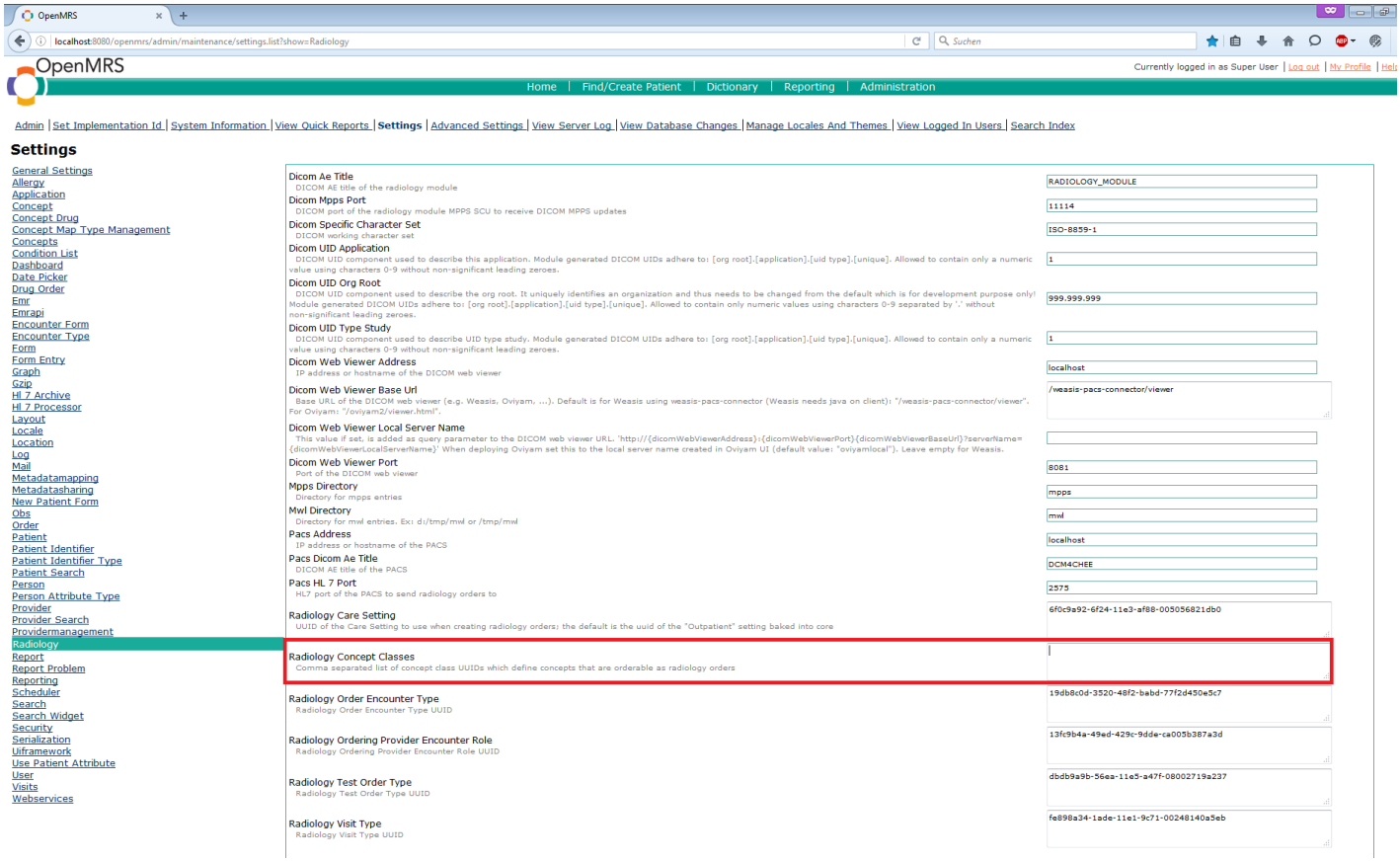
Follow the steps as shown to finish configuration of the OpenMRS Radiology Module.

4.1.1. Radiology Concept Classes

Before you can start to enter radiology orders you need to define the orderable procedures within your facility.

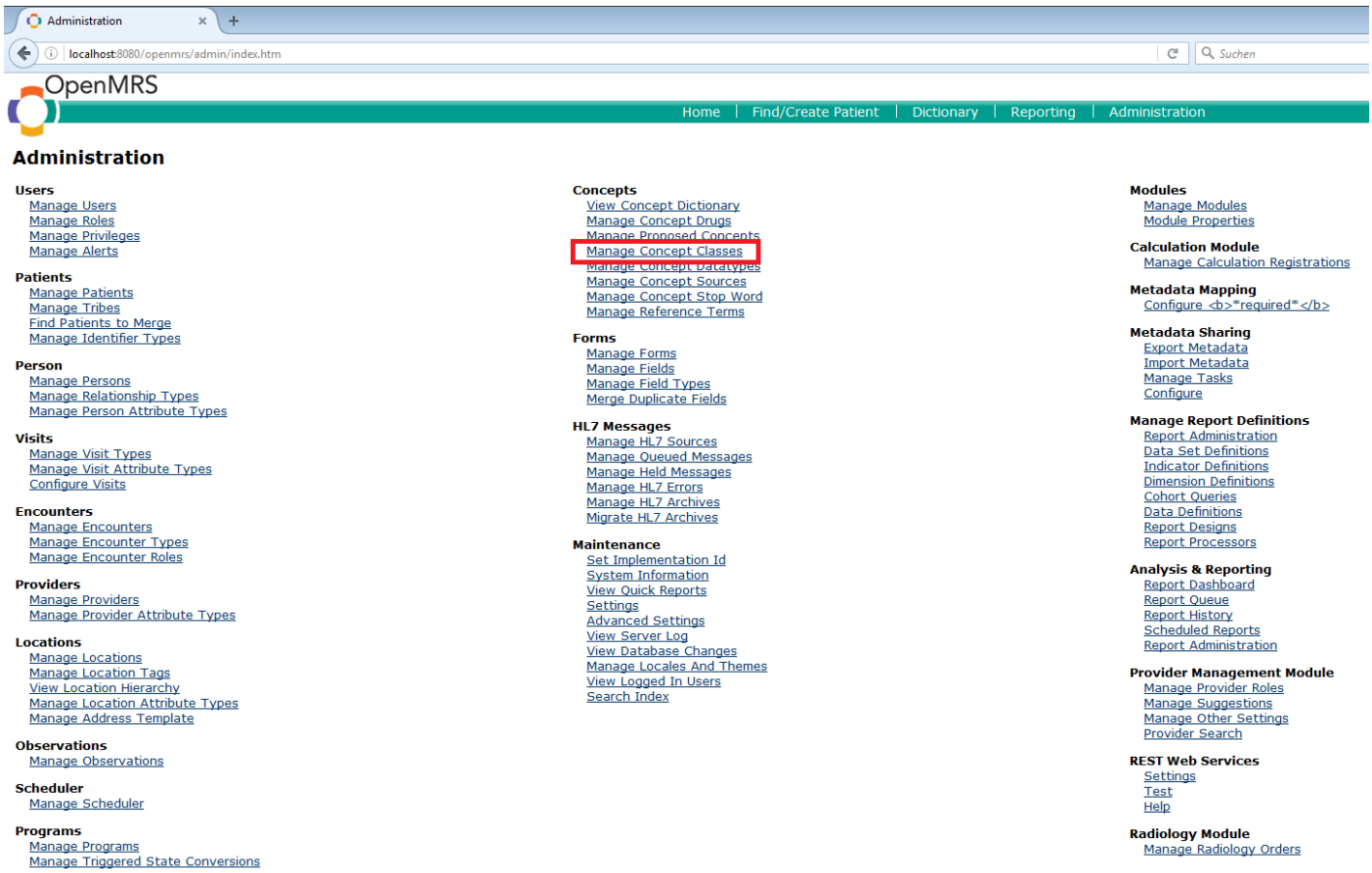
To do that you need to go to the Administration, Settings page and select "Radiology" in the menu on the left side.

The setting that needs to be specified is called "Radiology Concept Classes" as shown below

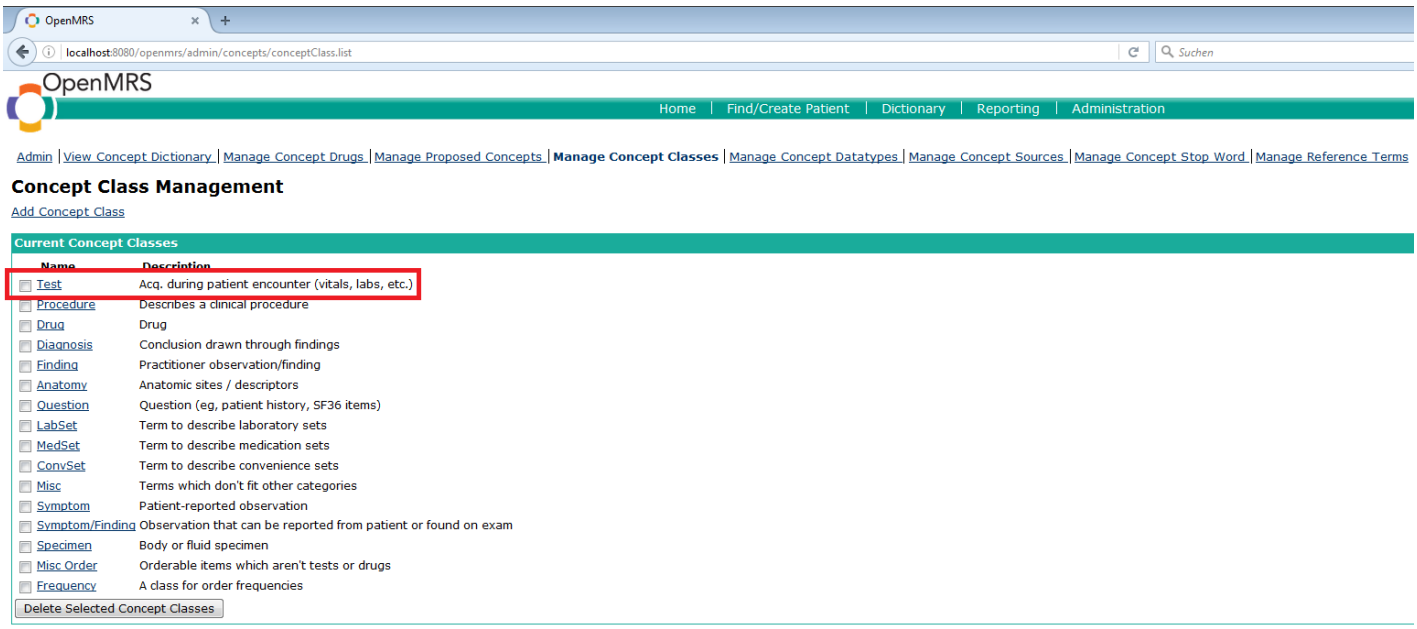


You can enter any list of OpenMRS concept class UUIDs.

Its easy to find these UUIDs via "Manage Concept Classes" link in the admin page



Click on any concept class you whish to configure, for example "Test"



And copy the UUID you see greyed out at the bottom

Paste this concept class UUIDs into the "Radiology Concept Classes" settings field and hit "Save".

4.2. Dcm4chee

Follow all the steps exactly as shown in the images to configure the connection between the OpenMRS Radiology Module and dcm4chee and to setup the DICOM web viewer weasis in dcm4chee.

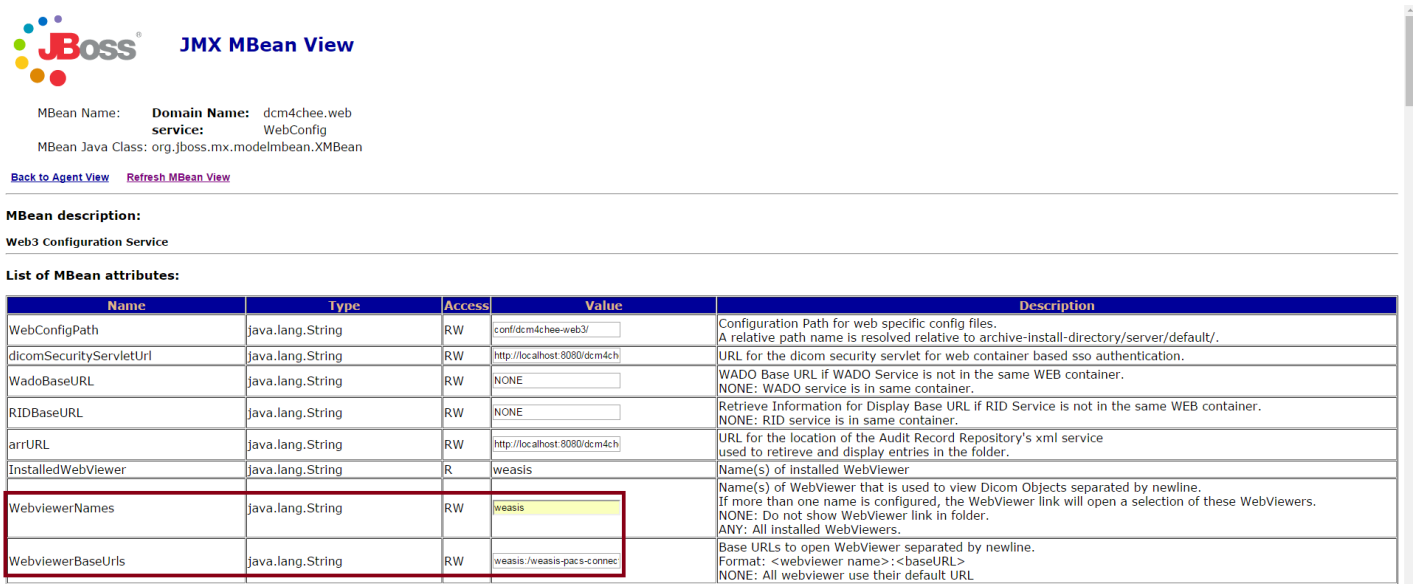
Go to http://localhost:8081/jmx-console/ (http://localhost:8081/jmx-console/) and login as default admin user

- username: `admin`
- password: `admin`

4.2.1. Setup Weasis

To configure weasis as the DICOM web viewer used by dcm4chee go to section **dcm4chee.web** service=WebConfig
(http://localhost:8081/jmx-console/HtmlAdaptor?
action=inspectMBean&name=dcm4chee.web%3Aservice%3DWebConfig)
and set

- WebviewerNames: `weasis`
- WebviewerBaseUrls: `weasis:/weasis-pacs-connector/viewer`



JMX MBean View

MBean Name: **Domain Name:** dcm4chee.web
service: WebConfig
MBean Java Class: org.jboss.mx.modelmbean.XMBean

Back to Agent View Refresh MBean View

MBean description:

Web3 Configuration Service

List of MBean attributes:

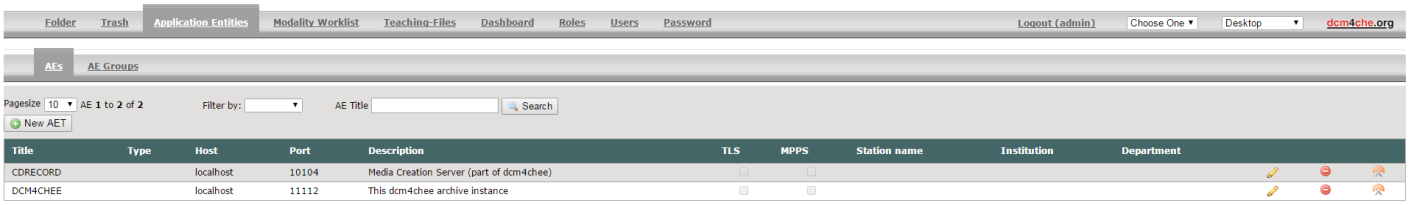
Name	Type	Access	Value	Description
WebConfigPath	java.lang.String	RW	conf/dcm4chee-web3/	Configuration Path for web specific config files. A relative path name is resolved relative to archive-install-directory/server/default/.
dicomSecurityServletUrl	java.lang.String	RW	http://localhost:8080/dcm4ch	URL for the dicom security servlet for web container based sso authentication.
WadoBaseURL	java.lang.String	RW	NONE	WADO Base URL if WADO Service is not in the same WEB container. NONE: WADO service is in same container.
RIDBaseURL	java.lang.String	RW	NONE	Retrieve Information for Display Base URL if RID Service is not in the same WEB container. NONE: RID service is in same container.
arrURL	java.lang.String	RW	http://localhost:8080/dcm4ch	URL for the location of the Audit Record Repository's xml service used to retireve and display entries in the folder.
InstalledWebViewer	java.lang.String	R	weasis	Name(s) of installed WebViewer
WebviewerNames	java.lang.String	RW	weasis	Name(s) of WebViewer that is used to view Dicom Objects separated by newline. If more than one name is configured, the WebViewer link will open a selection of these WebViewers. NONE: Do not show WebViewer link in folder. ANY: All installed WebViewers.
WebviewerBaseUrls	java.lang.String	RW	weasis:/weasis-pacs-connec	Base URLs to open WebViewer separated by newline. Format: <webviewer name>:<baseURL> NONE: All webviewer use their default URL

4.2.2. Setup Connection With OpenMRS Radiology Module

To connect the OpenMRS radiology module with dcm4chee you need to

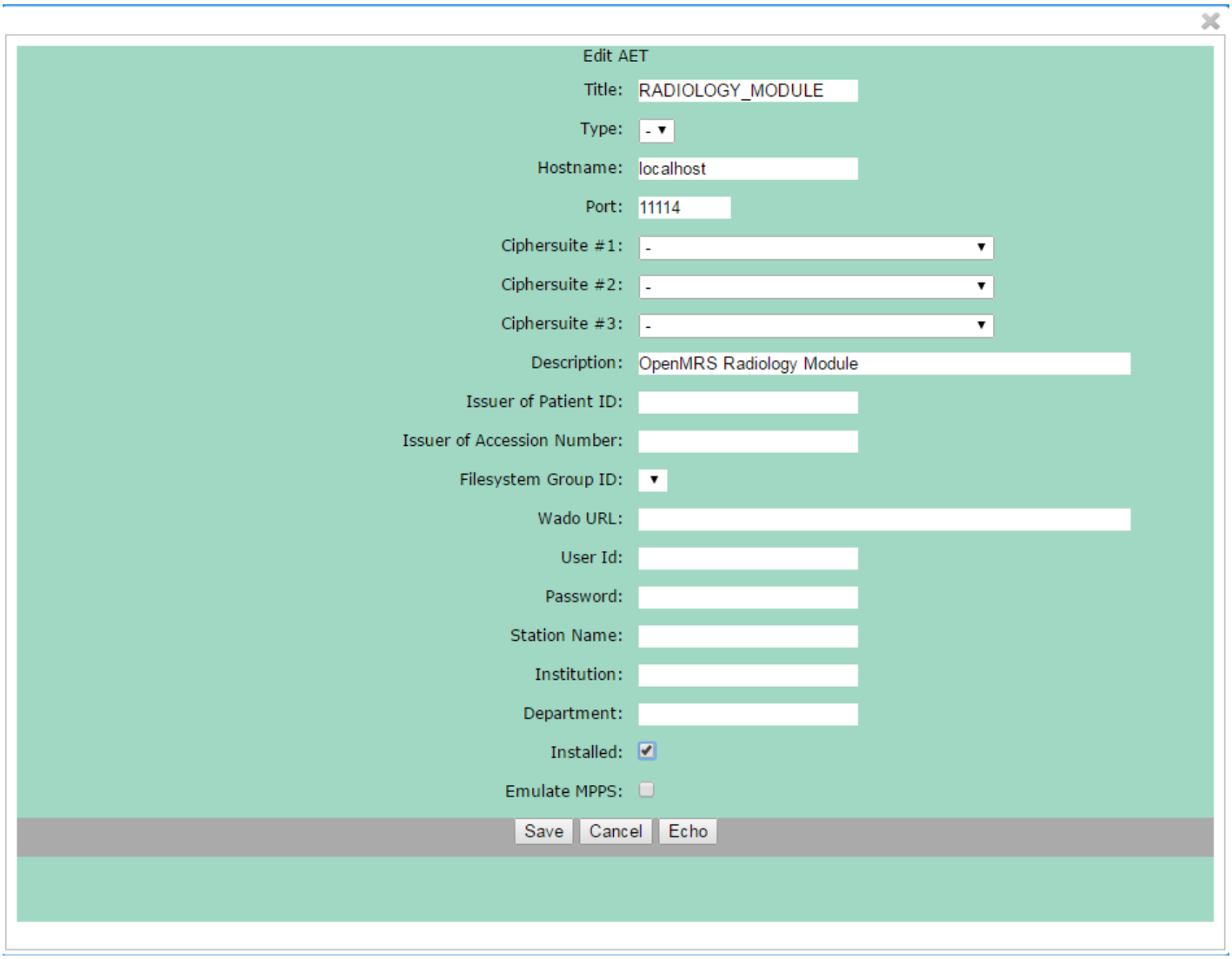
4.2.3. Add New AET

Add the radiology module as DICOM Application Entity at http://localhost:8081/dcm4chee-web3/ (http://localhost:8081/dcm4chee-web3/) under the tab **Application Entities**



Select **New AET** and enter:

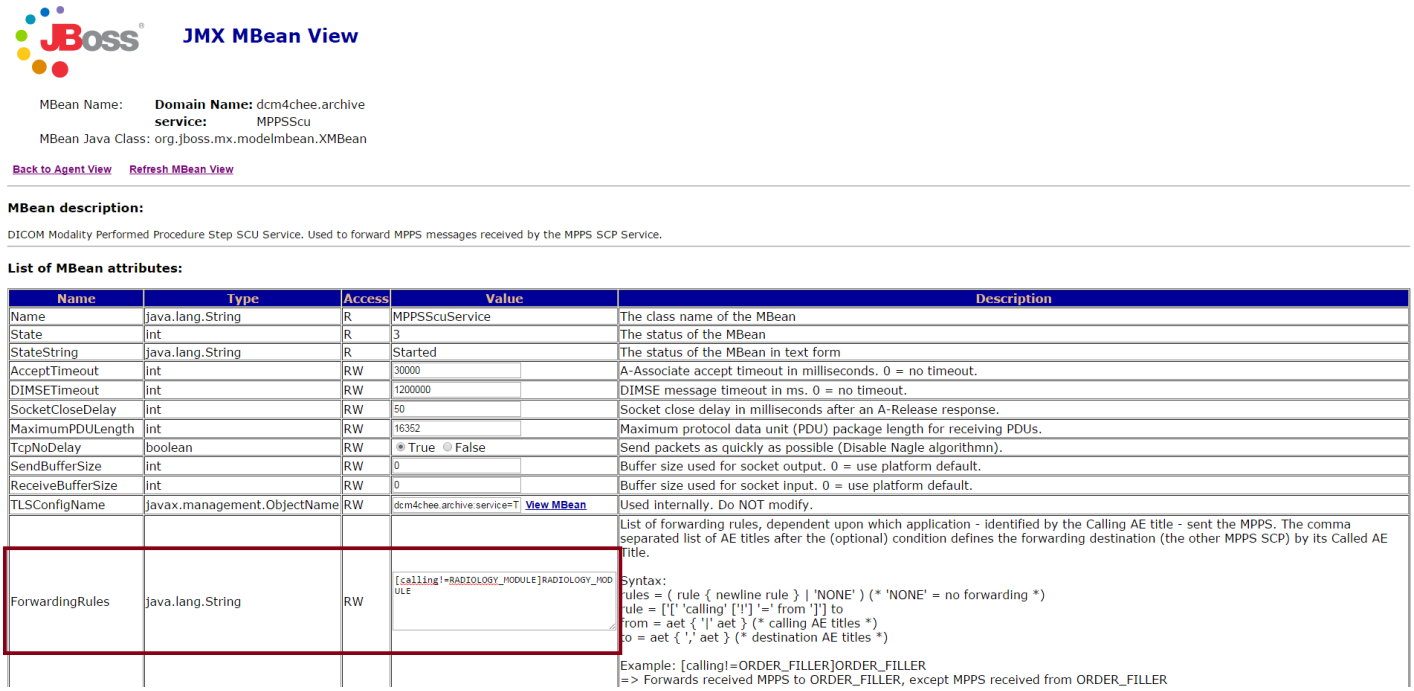
- Title: `RADIOLOGY_MODULE`
- Type: `-`
- Hostname: `localhost`
- Port: `11114`
- Description (optional): `OpenMRS Radiology Module`
- Installed (optional): enable



4.2.4. Setup DICOM MPPS Forwarding

To configure DICOM MPPS message forwarding to the OpenMRS radiology module go to section **dcm4chee.archive** service=MPPSScu
(http://localhost:8081/jmx-console/HtmlAdaptor?
action=inspectMBean&name=dcm4chee.archive%3Aservice%3DMPPSScu)
and set

- ForwardingRules: `[calling!=RADIOLOGY_MODULE]RADIOLOGY_MODULE`



JMX MBean View

MBean Name: **Domain Name:** dcm4chee.archive
service: MPPSScu
MBean Java Class: org.jboss.mx.modelmbean.XMBean

Back to Agent View Refresh MBean View

MBean description:

DICOM Modality Performed Procedure Step SCU Service. Used to forward MPPS messages received by the MPPS SCP Service.

List of MBean attributes:

Name	Type	Access	Value	Description
Name	java.lang.String	R	MPPSScuService	The class name of the MBean
State	int	R	3	The status of the MBean
StateString	java.lang.String	R	Started	The status of the MBean in text form
AcceptTimeout	int	RW	30000	A-Associate accept timeout in milliseconds. 0 = no timeout.
DIMSETimeout	int	RW	1200000	DIMSE message timeout in ms. 0 = no timeout.
SocketCloseDelay	int	RW	50	Socket close delay in milliseconds after an A-Release response.
MaximumPDULength	int	RW	16352	Maximum protocol data unit (PDU) package length for receiving PDUs.
TcpNoDelay	boolean	RW	True / False	Send packets as quickly as possible (Disable Nagle algorithmn).
SendBufferSize	int	RW	0	Buffer size used for socket output. 0 = use platform default.
ReceiveBufferSize	int	RW	0	Buffer size used for socket input. 0 = use platform default.
TLSConfigName	javax.management.ObjectName	RW	dcm4chee.archive:service=T View MBean	Used internally. Do NOT modify.
ForwardingRules	java.lang.String	RW	[calling!=RADIOLOGY_MODULE]RADIOLOGY_MODULE	List of forwarding rules, dependent upon which application - identified by the Calling AE title - sent the MPPS. The comma separated list of AE titles after the (optional) condition defines the forwarding destination (the other MPPS SCP) by its Called AE title. Syntax: rules = (rule { newline rule } | 'NONE') (* 'NONE' = no forwarding *) rule = ['[' 'calling' ['!'] '=' from ']'] to from = aet { '|' aet } (* calling AE titles *) to = aet { ',' aet } (* destination AE titles *) Example: [calling!=ORDER_FILLER]ORDER_FILLER => Forwards received MPPS to ORDER_FILLER, except MPPS received from ORDER_FILLER

Last updated 2016-05-01 18:20:19 CEST