Installing OSCARS 0.6

Introduction 1

Preparing your environment 2

Installing the Java Development Kit (JDK) 2

Java version 2

Download and Installation 2

Setting the JAVA\_HOME Environment Variable 2

Installing Maven 2

Download and Installation 3

Installing MYSQL 3

Initializing MYSQL 3

Downloading the IDC 3

Installing the IDC 4

Set environment variables 4

Installing from the binary distribution 4

Installing from the Source Distribution 4

Create Databases 5

Configuration and customization 5

Editing configuration files from source distribution 6

Configuration files 6

Configuration Scripts 6

Configuring your deployment 7

Using the WBUI 10

Running 11

Testing 11

Multi-domain testing 12

Customizing logging for your deployment 13

# Introduction

This document intends to guide a user in installing OSCARS 0.6. This document

explains details to install, configure and customize the installation for your site.

# Preparing your environment

## Installing the Java Development Kit (JDK)

Java is the programming language in which the OSCARS IDC software was created and provides the environment in which it runs. In addition to running the software, the JDK also contains utilities required for compiling the source code and generating user certificates. This section details installation and configuration related to this package.

### Java version

Many systems come pre-installed with Java. To install the IDC, your system must not only have Java Runtime Environment (JRE) version 6 but also the various compilers and utilities. To verify that you have the necessary Java environment, issue the following command:

% javac –version

If the first line of output reads javac 1.6.0\_X, you should not need to install the Java Development Kit and may skip to section **0** Setting the JAVA\_HOME Environment Variable. If you get “command not found” or the version number is less than 1.6, you may need to install JDK 5.0 and should proceed to Section Download and Installation.

**NOTE: If you are not running the SUN distribution of Java you may encounter issues. The GNU and IBM versions of Java are not fully tested and some users have reported problems. It is recommended you run the SUN distribution of Java.**

### Download and Installation

You may download JDK 6.0 from Sun’s web site at:

<http://www.oracle.com/technetwork/java/javase/downloads/index.html>

It is recommended that you download the latest update of **JDK Version 6**. Choose the package most suitable for your operating system.

Once downloaded, unpack the file; this should create a new folder named something similar to “jdk1.6.0\_X”. The final step of installation is to move this folder to an easily accessible place. We recommend renaming the folder to java6 in /usr/local with the following command:

% sudo mv jdk1.5.0\_X /usr/local/java6

The location may be anywhere you choose – just make sure you note the location as it is required for setting the JAVA\_HOME environment variable in the next section.

### Setting the JAVA\_HOME Environment Variable

Once Java is installed, you need to set the JAVA\_HOME environment variable with its location. This variable is required by Maven (see section Error! Reference source not found.Error! Reference source not found.**)** to run. To set this environment variable, issue these commands:

% JAVA\_HOME=/usr/local/java5

% export JAVA\_HOME

You may permanently set this variable (recommended) by adding the above commands to the profile file in your home directory (i.e. .bash\_profile or .profile).

# Installing Maven

Maven is a tool that can be used for building and managing any Java-based project, and is used to used to build the IDC code and deploy various configuration files This section details how to install and configure Maven.

### Download and Installation

Download Maven from the project’s website:

<http://maven.apache.org/download.html>

Most IDC testing has been done with **Version 2.X**. Unpack and install Maven with the instructions provided in the “Installation instructions” section of the above page.

The first time maven is run, it will download all the third party dependencies.

# Installing MYSQL

OSCARS uses mysql version 5. If mysql v5 or later is not installed, you can use the following commands to install mysql. You need to be a “root” user to execute these commands.

You may use Package Managers like up2date, yum, or apt-get to install MySql.

For example, on a Centos machine, mysql can be installed using the following commands

yum install mysql-server

yum install mysql

yum install mysql-dlevel

On Ubuntu machine, mysql can be installed using the following commands

apt-get install mysql-server

On Redhat Enterprise Linux machine, mysql can be installed using the following command

up2date install mysql-server

Refer to the installation instructions for you specific package on the Oracle MySql page if you plan to manually download/install MySql.

## Initializing MYSQL

If mysql has just been installed on your machine you must run **mysql\_install\_db --user mysql**, where mysql is the user id under which mysql will run to initialize it for your host.

To have mysqld start up at boot check if there is a mysqld start file in init.d. If not, you should copy support-files/mysql.server to the right place for your system.

The daemon can be started with **/usr/bin/mysqld\_safe &**

To set the password for the mysql root user

/usr/bin/mysqladmin -u root password 'new-password'

/usr/bin/mysqladmin -u root -h <localhost> password 'new-password'

The log file is in /var/log/mysqld.log and is readable by user and group mysql only. For mysql ver 5.5 the log is in /usr/local/mysql/data.

# Downloading the IDC

You can use subversion to download the most recent set of OSCARS sources or you can download a tar file of the last checkpointed SDK set of sources.

The tar file including all the necessary jar files can be downloaded from the oscars-idc wiki at

<http://code.google.com/p/oscars-idc/wiki/SoftwareDownload>

The above page, and this link below provides you access to the source code:

<http://code.google.com/p/oscars-idc/source/checkout>

# Installing the IDC

## Set environment variables

Set the environment variables $OSCARS\_DIST and $OSCARS\_HOME. $OSCARS\_DIST is the source directory for OSCARS. $OSCARS\_HOME is the deployment directory.

## Installing from the binary distribution

1. Run $OSCARS\_DIST/deployOscarsBin.sh to initialize the mysql tables.

Just before the script completes executing, you will see this prompt:

“do you want to edit the mysql oscars password or change any of the service ports? [y|n]”.

Answering “y” will print a information message that you have to run $OSCARS\_DIST/bin/exportconfig after you change the Oscars password/ports. Currently, the install process does not currently offer options to change ports. However, you may run configuration scripts, as described in section TBD to do so.

1. Run $OSCARS\_DIST/bin/exportconfig, if necessary, as mentioned in step above.
2. Build the system using: mvn install.

The following mvn switches may be useful:

-DskipTests :skips all the unit tests

-Dmaven.test.failure.ignore=true :executes the unit tests, but continues after

failures

## Installing from the Source Distribution

1. Run $OSCARS\_DIST/bin/deployOscarsSrc.sh script to configure the OSCARS package. This script may be run more than once. If it has already completed an action, it will not change the result. This script executes the following:

* Creates the $OSCARS\_HOME directory if it does not exist.
* Prompts for the mysql root user password. If this password is empty, set it with the following command:

/usr/bin/mysqladmin -u root password 'new-password'

* Executes several scripts to create and initialize the OSCARS Authentication (authN), Authorization(authZ) and ResourceManager (rm) databases and tables. If you receive the following errors you have incorrectly entered the mysql root password

ERROR 1045 (28000): Access denied for user 'root'@'localhost' (using password: YES) ERROR 1045 (28000): Access denied for user 'root'@'localhost' (using password: NO)

* Copies all the config.yaml.template files to their non-template versions. These files are needed for configuring the OSCARS package.

1. Build the system using: mvn install.

The following mvn switches may be useful:

-DskipTests : skips all the unit tests

-Dmaven.test.failure.ignore=true : executes the unit tests, but continues after

failures

NOTE: Use the deployOscarsSrc.sh script the first time you build the system. Subsequent builds can be done with just

svn update

mvn install

# Create Databases

This section is a guide that helps to install the OSCARS databases separately if there is a need to. You may skip this section if you ran either the “deployOscarsBin.sh” or “deployOscarsSrc.sh”, and move to setion TBD.

Before starting to create the databases, ensure that mysql is installed prior to setting up OSCARS on a fresh machine. Also ensure that mysql is running. If not, see section above for starting the mysql daemon.

Before the servers can be used for real requests, the Authentication, Authorization and ResourceManager databases need to be created and initialized. The script **$OSCARS\_DIST/bin/deployOscarsSrc.sh** will do this among other things. It calls the script **$OSCARS\_DIST/bin/oscarsdb** to do the following:

**oscarsdb init**

creates the mysql user 'oscars'@'localhost" user with a password of 'mypass'. This matches the value for the default user in {authN, authNPolicy, authZ, authZPolicy, resourceManager} /config/\*.yaml.template files.

See note below on how to change the default password.

**oscarsdb ct**

creates and populates the default tables.

**oscarsdb rt**

removes all the tables.

If you have not set a password for mysql root edit the script to use -u $SQLROOT rather than

–u $SQLROOT -p

If you wish to have the resourceManager tables populated with a set of anonymized reservation data use:

mysql -u root < resourceManager/sql/createAnonTables.sql

If you are running on a machine that already has OSCARS 0.5 databases, you can use the updgradeTables0.5-0.6.sql scripts to create the new tables and copy over the entries from the old tables. These scripts can be found in the {authN,authZ,resourceManager}/sql directories.

# Configuration and customization

Set the following environment variables:

$OSCARS\_DIST - source directory

$OSCARS\_HOME - deployment directory

After this step has been completed, you can proceed to configuring the other areas.

This section has been divided into two parts. The [first one](#_Editing_configuration_files), “Editing configuration files from source distribution” is a brief section intended for advanced users, developers and others who would like to change the configuration files (for example, the ones mentioned in the section above) manually. The [second part](#_Configuring_your_deployment), “Configuring your deployment”, details scripts and configuration details that mainly intend to make changes to the deployment environment.

### Editing configuration files from source distribution

You may manually make changes to various configuration files used by OSCARS. A brief list of the configuration files and steps involved in manually configuring OSCARS is presented in this section.

## Configuration files

Each service has the following configuration files in <service>/config.

**manifest.yaml**

contains the path names of the configuration files needed by that service. Variations of the configuration files are selected for UNITTEST, SDK, DEVELOPMENT and PRODUCTION contexts. The context value is held in the ContexConfig class and should be defined by the program that starts the service (often named Invoker, e.g. es.net.oscars.coord.common.Invoker.java). The startServer.sh scripts take a command line argument and pass it to the invoker. For standalone clients, the context needs to be defined in whatever script or program is being used to call the client interface.

**{service>|config}.yaml.[template**]

contains the publishTo address and optionally hibernate user values. If a yaml.template exists, it contains dummy hibernate oscars-user and password values. It will be copied to a .yaml file if one does not exist. The .yaml file should be edited to match the values in the local database.

**log4j.properties** – variations for DEBUG, MESSAGES, INFO

contains log4j logging properties

**server-cxf.xml** - variations for SSL or HTTP

contains https configuration, enables cxf message logging

**client-cxf.xml** - variations for SSL or HTTP

contains https configuration, enables cxf message logging

**<service>.cxf.xml**

contains the hibernate database configuration, if the service uses a database

### Configuration Scripts

The following scripts are used to edit and install configuration files into the $OSCARS\_HOME area.

**bin/exportconfig**

copies the config files to $OSCARS\_HOME/<ServiceName>/conf and edits some file names to match $OSCARS\_HOME. **Does not replace any files that already exist in $OSCARS\_HOME**. If you want to get new versions of the configuration files deployed, you must delete any existing ones in $OSCARS\_HOME before running bin/exportconfig.

**sampleDomain/bin**

contains scripts to create sample keystores that are needed by https and message signing. It also edits all the servers' deployed client/server-cxf files to use the correct pathnames for the keystores. Any time you do an exportconfig for a server, it must be followed by sampleDomain/bin/exportconfig. The exportconfig scripts are not run automatically, but can all be run from the top by $OSCARS\_DIST/bin/exportconfig which will run sampleDomain last.

#### Changing the default oscars@localhost mysql password

Anyone who has login access to the host on which the mysql server is running and knows the oscars mysql password can modify the OSCARS database files with mysql commands. Thus, before you put any critical user or reservation information in those files, you should change the default password and protect the files that contain it to the user that the services will run as. The distribution includes the following files:

authN/config/authN.yaml.template

authNPolicy/config/authNPolicy.yaml.template

authZconfig/authZ.yaml.template

authZPolicy/config/authZPolicy.yaml.template

resourceManager/config/config.yaml.template

The deployOscarsSrc.sh script will copy the yaml.template files to .yaml files if such files do not already exist. To change the default password edit the \*.yaml files and change the protection of those files to be readable only by the userId that the service runs under. Each of the 5 yaml files must have the same password. Once you have changed the password that the OSCARS software will use, you must change the password in the mysql database. To do this:

mysql –u Oscars –p mypass

SET PASSWORD = password(’newpassword’);

quit;

### Configuring your deployment

This section describes the tools and methods necessary to configure an OSCARS 0.6 deployment.

#### Changing your DB password

It is recommended that the OSCARS MySql password for the “OSCARS” user be modified before setting in critical user/reservation information. To do so, follow these steps

cd $OSCARS\_DIST/tools/bin

./ idc-dbpassmod

This script offers 3 options:

1.Change Database password in ALL locations (Default option)

2. Change OSCARS references to Database password

3. Change MySQL Database password

Selecting option 1 will to change both the MySQL and OSCARS configuration file references to your intended values. This option is the default option. You will be prompted for your old password, which will be used to authenticate and then change over to the new value you provide.

Option 2 is used to change the OSCARS password in configuration files referenced by OSCARS. For example, if you intend to manually change the MySql password, OSCARS can then be made to use the same by running the script with this option.

Option 3 is a utility which helps set only the DB password. Please note that this is for the “Oscars” MySQL user alone.

#### Defining your local domain

You must define your local domain so OSCARS knows what elements to control. Every domain has a unique identifier. For example Internet2’s domain ID is ion.internet2.edu and ESnet is identified by es.net. You are free to choose your own domain identifier.

The local domain used by your IDC installation can be viewed set using the steps below:

./ idc-dbpassmod

idc-localdomainview

You can change the local domain used by your IDC installation using the steps below:

cd $OSCARS\_DIST/tools/bin

./idc-dbpassmod

idc-localdomainmod

#### Defining your Topology

##### Generating an XML Topology Description

OSCARS currently requires you to manually generate an XML file that describes your network’s topology in the Open Grid Forum (OGF) Network Measurement Working Group (NMWG) control plane topology. The topology description you generate describes what is possible on your network. For example, it is not concerned with what VLANs are currently provisioned on a network, rather the possible VLANs that could be provisioned on the network.

###### Example XML files

The easiest way to generate this file is to start from the two examples provided with the OSCARS Software Suite. These files describe the same topology of a “testdomain-1” domain with 2 nodes, and a “testdomain-2” domain with 4 nodes. You can find the example XML files in the following locations:

$OSCARS\_HOME/TopoBridgeService/conf/testdomain-1.xml

$OSCARS\_HOME/TopoBridgeService/conf/testdomain-2.xml

*Setting location of topology file*

Once you have the topology file ready, you can place it in any location you find convenient, and instruct OSCARS to find it using the following commands:

cd $OSCARS\_DIST/tools/bin

./idc-domaininfoadd <CONTEXT>

You will be prompted to enter your domain first. If the domain is not already present, you will then be prompted to specify whether it is a “file” or “topology server”. Choose “file”, since you have created a “topology file” for your local domain. You will then be prompted to enter the topology file name. The location can be either in the form of an absolute path, or a relative path. A relative path indicates that the Topology service will try to locate the file relative to $OSCARS\_HOME/TopoBridgeService/conf.

*Adding topology server information*

OSCARS provides a Topology Bridge Service whose function is also to store topology information. This topology information can be stored or located in two ways, currently. The first method is that of storing the topology information in a static file, and instructing OSCARS to look for the topology for a particular domain in a static file. The second method is to indicate OSCARS to look for a particular domain via a bridge to a PerfSONAR topology Server.

You can add information about such a topology server using the following commands.

cd $OSCARS\_DIST/tools/bin

./idc-toposerveradd <CONTEXT>

You will be now be prompted for the domain. If you would not want to specify a particular domain, and would like to use a PerfSonar Topology Server for all domains you did not specify individually, then choose “\*” when you are prompted for a domain.

If the domain is not already present, the Topology Server details will be added afresh. On the other hand, if the domain is already present, the topology Server location you specify will be added on to this. In other words, the same script can be used to add and modify your topology server information. Also, having more than one Topology Server for a specific domain simply means that OSCARS will try to locate the topology associated with this domain using these Topology Servers in succession.

*Modifying Domain Information*

The previous two sections provided details about adding topology information to OSCARS.

However, you may also want to modify your current set up. The commands below will help you achieve this.

cd $OSCARS\_DIST/tools/bin

./idc-domaininfoadd <CONTEXT>

You will be prompted to enter a domain. Once you have entered a domain name, you will be prompted to specify whether it is a File/Topology Server you would like to newly associate to this domain. Note that you may even change your original setup to anything among File or TopologyServer .In other words, if you had originally associated domain X with a “file”, you can switch it to use a Topology Server instead)

*Viewing/Setting local domain*

The following script helps to view your local domain:

cd $OSCARS\_DIST/tools/bin

./idc-localdomainview

You can change your local domain to point to another domain using the below commands:

cd $OSCARS\_DIST/tools/bin

./idc-localdomainview <CONTEXT>

You will be prompted for a domain. If the domain you specify is not found, you will see a message to run idc-domainadd to add in this domain newly. However, your choice of local domain will be stored.

#### Changing ports used

You can customize the ports on which OSCARS services are being run currently.

To view the ports currently being used, run the following set of commands:

cd $OSCARS\_DIST/tools/bin

./idc-portview <CONTEXT> <server>

You can view the port number used by any specific service by using a specific option for “server”, or specifying “ALL”.

To change any of the ports being used by the different services, use the following set of commands:

cd $OSCARS\_DIST/tools/bin

./idc-portmod <CONTEXT> <server>

Listing the services here may help, but this section is TBD based on whether we would move to a “single JVM” option.

The current ports being used are:

api: 9001, 9002 (internal)

authN: 9090

authNPolicy: 9004

authNStub: 9011

authZ: 9190

authZPolicy: 9005

coordinator: 9003

coordinator:pceRuntime: 10000

lookup: 9014

notify: 9012

PCEs

stubPCE 9007 (only run if no other PCE’s are run)

connectivityPCE 9007

bandwidthPCE 9009

vlanPCE 9010

dijkstraPCE: 9008

resourceManager: 9006

PSS: 9050

topoBridge: 9019

wbui:http 8080

wbui:https 8443

#### Key stores

OSCARS creates a set of default “test” keys that are used by the various services and in inter-domain communication. However, to use you own set of keys, refer to

http://code.google.com/p/oscars-idc/wiki/KeystoreFiles

# Using the WBUI

If you wish to use the web interface to manage users and their privileges you must first create an administrative user. To do this:

cd $OSCARS\_DIST/tools

mvn install

bin/idc-useradd

The idc-useradd commad will prompt for all the necessary values. Be sure to grant your user the

OSCARS-administrator attribute (1). If you wish to create, query or list reservations from the WBUI, you should also give this user the OSCARS-engineer attribute.

Verify that the authN , authZ and wbui servers are running and use your browser to go to https:/localhost:8443/OSCARS. Once there you will need to login as user you just created.

Currently you can use the browser interface to manager users, create, query or list reservations.

#### WBUI access

The default configuration for the WEBUI is to allow access to only the “localhost”. To allow web access to all hosts other than the one hosting the WBUI, use the following script:

cd $OSCARS\_DIST/tools/bin

./idc-wbuiaccess <CONTEXT> <option>

<Option> is one of allow or deny.

Thus, the above script could be used to either allow web UI access to all hosts, or to limit it to the current host alone (i.e deny to every one but localhost).

# Running

**mvn install** creates the jar and one-jar files in the directories, $OSCARS\_DIST/<service>/target, and puts them in your maven repository. It then runs all the unit tests. You can use the –DskipTests switch to mvn to skip the unit tests, or the –Dmaven.test.failure.ignore=true to run the tests but ignore all failures. Note: the Coordinator unit tests will fail if there is already a Coordinator service running.

A server is run by the command **java -jar target/<service>-0.0.1-SNAPSHOT.one-jar.jar**.

There are **<service>/bin/startServer.sh** scripts for each service that contain this command and some useful options. The scripts **$OSCARS\_DIST/bin/{startServers.sh,stopServers.sh}** will start and stop servers. The testServers will check to see that all the servers have been started and are listening on their service ports.

startServers.sh <context> <server >

<context> is one of: PRODUCTION|pro UNITTEST|test DEVELOPMENT|dev SDK|sdk

<server> is either ALL or one or more of:

authN authZ api coord topoBridge rm stubPSS lookup wbui stubPCE

bwPCE connPCE dijPCE vlanPCE nullAGG stubPSS notifyBridge wsnbroker

stopServers <server>

<server> is either ALL or one or more of:

authN authZ api coord topoBridge rm stubPSS lookup wbui stubPCE

bwPCE connPCE dijPCE vlanPCE nullAGG stubPSS notifyBridge wsnbroker

testservers <CONTEXT>

<context> is one of: PRODUCTION|pro UNITTEST|test DEVELOPMENT|dev SDK|sdk

Note: it can take up to several minutes for all the services to be fully initialized. It is recommended to run on a host with at least 4G of memory. Occasionally the OSCARSService (api) will fail to register itself with the Lookup Service because the Lookup Service was not started in time. In this case, you can just stop the OSCARSService and restart it once the LookupService is responding

stopServers.sh api

startServers.sh <context> api

# Testing

Currently the api, authN, authZ, coordinator and resourceManager bin directories contain scripts to run client test programs. To test the whole system, only the commands in api/bin are needed. The other scripts can be used to test just their specific services. The scripts contain comments on how to run them and can be called with a –h option for more information. These scripts use the library jars in <service>/target/tmp/lib which is created from the latest one-jar.jar.

The scripts in api/bin are:

createRes.sh –pf <paramFile>

modifyRes.sh –pf <paramFile> (not tested completely yet)

cancelRes.sh –gri <gri>

query.sh –gri <gri>

list.sh –n <numReq> -o <offset> -st <status>

setupPath.sh –gri <gri>

teardownPath.sh –gri <gri>

All the commands also take an optional context parameter: –C sdk|dev|pro which defaults to dev (DEVELOPMENT). If you are running the servers in PRODUCTION context, you must also run the clients in PRODUCTION mode since it requires https connections. SDK and DEVELOPMENT contexts are compatible, but SDK will not print the messages sent to the server.

Some sample paramFiles can be found in api/src/test/resources/\*.yaml

autoTD1.yaml – creates a reservation of 4 minutes duration, starting immediately in testdomain-1. It should work with the default installation.

autoTD2.yaml – like TD1 except it creates the reservation in testdomain-2. To use this you must change the localDomain:id to testdomain-2 in topoBridge/config/config.yaml

signalTD1.yaml – creates a signal-xml reservation in testdomain-1 that will be exist for 5 days. You can use this reservation to test the setupPath and teardownPath operations.

autoTD3TD4.yaml – creates a mutli-domain reservation starting in testdomain-3 and ending in testdomain-4.

## Multi-domain testing

To test a multi-domain reservation you will need to have two OSCARS IDC services running: the first should have the localDomain:id in topoBridge/config/config.yaml set to testdomain-3, the second should have testdomain-4 as its localDomain:id. An OSCARSService finds a peer service for another domain using a Lookup Service. The lookup service that we distribute is a bridge server can contact an external Lookup service and cache the URLs of services for peer domains. Since we are not currently deploying and external service, we need to add the service/domain information directly to the Lookup Service cache. This is done for testdomain-3 (running on host3) and testdomain-4 (running on host3) by the following commands:

On the host3 that is serving testdomain-3:

cd $OSCARS\_DIST/lookup/

bin/oscars-idcadd -d testdomain-4 -p <http://oscars.es.net/OSCARS/06>

-l [http://host4:9001/OSCARS](http://odev-vm-12.es.net:9001/OSCARS)

On the host4 that is serving testdomain-4:

cd $OSCARS\_DIST/lookup/

bin/oscars-idcadd -d testdomain-3 -p <http://oscars.es.net/OSCARS/06>

-l [http://host3:9001/OSCARS](http://odev-vm-12.es.net:9001/OSCARS)

These commands will add the appropriate entries to $OSCARS\_HOME/LookupService/data and only need to be repeated if you delete that directory.

The two servers also need to share a common root CA and RA, so that the signed messages that are passed between them can be verified. To accomplish this:

On the primary IDC domain, e.g testdomain-3, when first running sampledomain/bin/gencerts use an empty $OSCARS\_HOME/sampleDomain/certs which will cause a new CA, RA, client, localhost and oscarsidc keystores to be created. On the second IDC, testdomain-4, copy the sharedCred.tar file that was created in the first domain into $OSCARS\_HOME/sampleDomain/certs on the second host. Then this command will use the shared RA to generate and sign a new oscarsidc certificate and keystore.

# Customizing logging for your deployment

Getting the right amount of logging is often a delicate balance, so we have provided ways to customize the logging in the OSCARS configuration files. In our default logging setup all the log4j messages (root logging) are put into per service .out files in $OSCARS\_DIST. In addition, only the OSCARS specific messages are put into per service log files in $OSCARS\_HOME/logs. We have provided three default logging profiles:

INFO: intended for production systems that only logs *info* level log messages and in addition puts the log messages from all the services into $OSCARS\_HOME/oscars.log

DEBUG: which logs the *debug* level log messages

MESSAGES: which logs *debug* level log messages and enables the cxf message logging which will put the all the inter-service messages in the .out files.

By default, the PRODUCTION context uses the INFO log files, SDK uses DEBUG level and DEVELOPMENT uses MESSAGE. Note that the message logging can be very verbose for the coordinator/pce messages as they contain the entire topography. One can change the log level used by any service, by editing that service’s manifest file.

Once a log4j.\*.properties file has been copied to $OSCARS\_HOME, it can be site-edited there. It will not be overwritten by the distributed version unless it is deleted first.

Two useful customizations are:

1. To add or delete message logging by editing the manifest.yaml file.
2. To decrease the number of log messages by editing the $OSCARS\_HOME/<ServiceName/conf/log4j.\*propertiesfile by changing the line “log4j.logger.net.es.oscars=DEBUG, <logName>”to “log4j.logger.net.es.oscars.<serviceName>=DEBUG, <logName>”

which will eliminate the log messages from the utils classes. <serviceName> matches the name of the source directory for the service: e.g. api,coordinator,authN.