

GT 4.2.0 WS MDS WebMDS: System Administrator's Guide

DRAFT

GT 4.2.0 WS MDS WebMDS: System Administrator's Guide

Introduction

WebMDS enables end users to view monitoring information via a standard web browser interface, without installing any additional software on their PC. WebMDS is implemented as a servlet that uses a plugin interface to gather monitoring information (or any other information in XML format) and XSLT transforms, and present the data to the user in a readable form. Web site administrators can customize their own WebMDS deployments by using HTML form options, configuring different plugins to collect data and XSLT transforms, and creating their own plugins and XSLT transforms.

This guide contains advanced configuration information for system administrators working with WS MDS WebMDS. It provides references to information on procedures typically performed by system administrators, including installation, configuring, deploying, and testing the installation.

WebMDS is built and installed as part of a default GT installation. Read [Installing GT 4.2.0](#) and [WS MDS System Administrator's Guide](#) for more information.

Table of Contents

WebMDS Howtos	5
1. Configuring	1
1. Configuration overview	1
2. Syntax of the interface	1
3. XML Sources included with WebMDS	2
2. Deploying	4
1. Standard deployment into Tomcat 5.0.28	4
2. Deploying WebMDS and Globus in the same Tomcat Server	4
3. Custom deployment	5
3. Testing	7
4. Security Considerations	10
1. WebMDS Security Considerations	10
5. Debugging	11
1. Logging	11
6. Troubleshooting	12
1. Error Messages	12
Glossary	13
Index	14

List of Tables

1.1. Pre-configured information sources	1
1.2. Configuration parameters used with FileXMLSource	2
1.3. Configuration parameters used with NodeXMLSource	2
1.4. Configuration parameters used with ResourcePropertyQueryNodeSource	3
1.5. Configuration parameters used with ResourcePropertyNodeSource	3
6.1. WS MDS Trigger Service Error Messages	12

DRAFT

WebMDS Howtos

Symbols

\$GLOBUS_LOCATION/lib/webmds/conf,
\$GLOBUS_LOCATION/lib/webmds/conf/indexinfo,

C

configuration interface

- default,
- overview,

configuring

- default,
- monitor different Index Service,
- overview,

D

deploying,

- custom,
- standard, in Tomcat 5.0.28,
- WebMDS and GT in the same Tomcat Server,

E

errors,

I

information sources,

L

logging,

S

security considerations,

T

testing,
troubleshooting,

X

XML sources,

Chapter 1. Configuring

1. Configuration overview

WebMDS can be configured to get information from any of various sources and to filter it through any XSL transform. WebMDS uses configuration files to specify the location of (and to name) sources of information and xsl and web form arguments to select among these configured information sources and xsl transforms.

By default, WebMDS comes configured to report information about an index server using transaction-level security on the default port (8443) on the local system. If you are running the Globus Toolkit in this default configuration, then you can use WebMDS to query your local *Index Service* without any configuration changes.

If you wish to monitor a different Index Service, you will need to edit the file `$GLOBUS_LOCATION/lib/webm-
ds/conf/indexinfo` to change the URL in the line:

```
<value>https://127.0.0.1:8443/wsrf/services/DefaultIndexService</value>
```

to match the URL of your default index service. Changes to WebMDS configuration files take effect the next time that Tomcat is restarted.

For other configuration changes (e.g., monitoring different kinds of services), see the detailed configuration information below.

2. Syntax of the interface

Each configuration file in `$GLOBUS_LOCATION/lib/webm-
ds/conf` defines a source of XML, which can be used in an HTML form to specify sources of information and XSL transforms. The distribution contains some standard configuration files in this directory, including:

Table 1.1. Pre-configured information sources

<code>indexinfo</code>	all resource properties from an index server running with transaction-level security on port 8443 on the local host
<code>indexinfo_nosec</code>	all resource properties from an index server running with no security on port 8080 on the local host
<code>openEndedQuery</code>	all resource properties from a user-specified grid service
<code>openEndedRP</code>	a user-specified resource property from a user-specified grid service
<code>servicegroupxsl</code>	an xsl transform that presents summary information about a service group
<code>sgedetail</code>	an XSL transform that presents detailed information about a service group entry

Each configuration file defines a `Webm-
dsConfig` object. A `Webm-
dsConfig` object consists of:

- A `description`: a textual description of the XML source being defined.
- A `className`: the name of the Java class that will be used to acquire the XML data.
- Zero or more `parameter` objects, each of which consists of the name of some parameter recognized by the Java class specified by `className`, and the string value of that parameter.

For example, this is `$GLOBUS_LOCATION/lib/webmds/conf/servicegroupxml`, which defines the `servicegroupxml` XML source:

```
<WebmdsConfig>
  <description>
    XSL file to show service group summary information
  </description>
  <className>org.globus.mds.webmds.xmlSources.file.FileXmlSource</className>
  <parameter>
    <name>file</name>
    <value>xslfiles/servicegrouptable.xml</value>
  </parameter>
</WebmdsConfig>
```

This file tells WebMDS to use the `org.globus.mds.webmds.xmlSources.file.FileXmlSource` Java class (a class which reads XML from a local file) to collect XML data and to pass a `file` parameter (which that Java class interprets as the name of the file to open, relative to the WebMDS base directory).

Tomcat must be restarted (or one of the more advanced Tomcat administrative mechanisms must be used) for changes to these configuration files to take effect.

3. XML Sources included with WebMDS

3.1. FileXMLSource

The class `org.globus.mds.webmds.xmlSources.file.FileXmlSource` reads XML from a file, and recognizes a single parameter:

Table 1.2. Configuration parameters used with FileXMLSource

<code>file</code>	The name of the file to read. Relative path names are interpreted relative to the WebMDS base directory (<code>\$GLOBUS_LOCATION/lib/webmds</code>).
-------------------	--

3.2. NodeXMLSource

This XML source class uses a `WebmdsNodeSource` object to fetch an XML document and return it in a form that is usable by WebMDS. It recognizes the following options:

Table 1.3. Configuration parameters used with NodeXMLSource

<code>class</code>	The name of a class that implements the <code>WebmdsNodeSource</code> interface. An instance of this class will be used to get an XML document.
<code>parameters</code>	Additional parameters are passed to an instance of the class specified by the <code>class</code> argument.

3.3. Classes That Implement WebmdsNodeSource

The following classes implement the `NodeXMLSource` interfaces and can be used in conjunction with `NodeXMLSource`

3.4. ResourcePropertyQueryNodeSource

This class performs a resource property query to get all the resource properties for some web service. It recognizes the following configuration parameters:

Table 1.4. Configuration parameters used with ResourcePropertyQueryNodeSource

endpoint	The endpoint name to be used in a resource property query.
endpointKeyName and endpointKeyValue	An optional key/value pair to use as reference properties for the endpoint specified with the endpoint parameter.
allowUserEndpoints	If true, values for <code>xmlSource.sourceName.param.endpoint</code> , <code>xmlSource.sourceName.param.endpointKeyName</code> , and <code>xmlSource.sourceName.param.endpointKeyValue</code> specified in the request will override the configured endpoint value.
endpointFile	The name of a file from which the endpoint information (in XML) will be read. This configuration parameter can never be overridden by request arguments.

3.5. ResourcePropertyNodeSource

This class queries a web service for a single resource property. It recognizes the following parameters:

Table 1.5. Configuration parameters used with ResourcePropertyNodeSource

endpoint	The endpoint name to be used in a resource property query.
endpointKeyName and endpointKeyValue	An optional key/value pair to use as reference properties for the endpoint specified with the endpoint parameter.
allowUserEndpoints	If true, values for <code>xmlSource.sourceName.param.endpoint</code> , <code>xmlSource.sourceName.param.endpointKeyName</code> , and <code>xmlSource.sourceName.param.endpointKeyValue</code> specified in the request will override the configured endpoint value.
endpointFile	The name of a file from which the endpoint information (in XML) will be read. This configuration parameter can never be overridden by request arguments.
rpNamespace	The namespace part of the QName of the resource property to be queried for.
rpName	The local name part of the QName of the resource property to be queried for.
allowUserResourceProperties	If true, values of <code>xmlSource.sourceName.param.rpNamespace</code> and <code>xmlSource.sourceName.param.rpName</code> specified in the request will override the configured resource property namespace and name.

Chapter 2. Deploying

Because WebMDS is implemented as a servlet, it must be deployed into a servlet container, such as [Tomcat](#)¹. The following instructions assume that you've installed Tomcat version 5.0.28. and set the `$CATALINA_HOME` environment variable to the directory into which you've installed Tomcat.

1. Standard deployment into Tomcat 5.0.28

The standard deployment consists of two steps: creating a configuration file that tells Tomcat where to find the WebMDS servlet and related files, and restarting Tomcat so that it will read this new configuration file. These steps require write permission on files and directories in `$CATALINA_HOME`; they do not require write permission on anything in `$GLOBUS_LOCATION`.

To create the configuration file, run this command:

```
$GLOBUS_LOCATION/lib/webmds/bin/webmds-create-context-file \  
$CATALINA_HOME/conf/Catalina/localhost
```

This will create `$CATALINA_HOME/conf/Catalina/localhost/webmds.xml`. Note: if this file already exists (e.g., if you've previously installed another version of WebMDS), you'll need to use the `-f` option to `webmds-create-context-file`.

Next, make sure that Tomcat has a version of the Xalan library (used by WebMDS to do XSL transforms) that is compatible with the one used by Globus:

```
cp $GLOBUS_LOCATION/endorsed/xalan-2.6.jar $CATALINA_HOME/common/endorsed/.
```

Next, restart Tomcat. If Tomcat is already running, stop it:

```
$CATALINA_HOME/bin/shutdown.sh
```

Then, start Tomcat:

```
$CATALINA_HOME/bin/startup.sh
```

2. Deploying WebMDS and Globus in the same Tomcat Server

If you wish to run Globus and WebMDS in the same Tomcat instance (instead of, for example, running Globus in the Globus standalone container and WebMDS in Tomcat), then do the following:

1. Install Globus and deploy it into Tomcat, as described in [Installing GT 4.2.0](#).
2. Run `webmds-create-context-file`:

¹ <http://jakarta.apache.org/tomcat/>

```
$GLOBUS_LOCATION/lib/webmds/bin/webmds-create-context-file \  
$CATALINA_HOME/conf/Catalina/localhost
```

(see the previous section for more details about **webmds-create-context-file**).

3. The Globus and WebMDS deployments install identical copies of certain files in different places. The presence of these duplicates causes WebMDS to fail when sending requests to secure servers. To prevent this problem, remove the duplicates:

```
rm $GLOBUS_LOCATION/lib/webmds/WEB-INF/lib/puretls.jar  
rm $GLOBUS_LOCATION/lib/webmds/WEB-INF/lib/cryptix*.jar  
rm $GLOBUS_LOCATION/lib/webmds/WEB-INF/lib/jce-jdk*.jar
```

4. Finally, restart Tomcat. If Tomcat is already running, stop it:

```
$CATALINA_HOME/bin/shutdown.sh
```

Then, start Tomcat:

```
$CATALINA_HOME/bin/startup.sh
```

3. Custom deployment

If you are already running a Tomcat server (or other server that supports servlets) and your preferred mechanism for installing servlets is something other than creating a configuration file and restarting your web server, feel free to use that mechanism. The servlet root for WebMDS is `$GLOBUS_LOCATION/lib/webmds`.

For the rest of these instructions, the term *Globus user* will be used to refer to the owner of the `$GLOBUS_LOCATION` directory, and *Tomcat user* will be used to refer to the owner of the `$CATALINA_HOME` directory. If the Globus and Tomcat installations were performed from the same user account, the Globus user and Tomcat user will be the same.

Any time you change the servlet configuration (or any jar files used by the servlet), you'll need to let tomcat know there was a change. If you have a preferred way of configuring tomcat, feel free to use it, with `$GLOBUS_LOCATION/lib/webmds` as the servlet directory. These steps need to be performed by the Tomcat user.

If you're using tomcat 5.0.28 and haven't done any custom configuration (such as defining additional hosts) other than changing the tomcat port, you can configure tomcat by doing the following:

1. Create a context descriptor file called `webmds.xml` in the location where tomcat will look for it:

```
$GLOBUS_LOCATION/lib/webmds/bin/webmds-create-context-file \  
$CATALINA_HOME/conf/Catalina/localhost
```

Note: if the file `$CATALINA_HOME/conf/Catalina/localhost/webmds.xml` already exists, you can use the `-f` flag to `create-context-file` to overwrite it. to the tomcat configuration directory.

2. If tomcat is running, shut it down.

```
$CATALINA_HOME/bin/shutdown.sh
```

3. Start tomcat up.

```
$CATALINA_HOME/bin/startup.sh
```

DRAFT

Chapter 3. Testing

The easiest way to test your installation is to use it to view your *Index Service*, by pointing your web browser at `http://your-tomcat-host:your-tomcat-port/webmds` and clicking on the link labelled "A list of resources registered to the local default index service".

For more in-depth tests, you can run the WebMDS unit tests, by doing the following:

1. Install [httpunit](http://httpunit.sourceforge.net)¹, version 1.6 or later. Set the environment variable `GLOBUS_HTTPUNIT_DIR` to the directory into which httpunit has been installed.
2. Install the WebMDS test package; from the GT4 distribution directory, run

```
make gt4-webmds-test
```

3. Run the core WebMDS test suite. This tests the WebMDS servlet itself, the File XML Source, and the more commonly-used xslt transforms. There are two modes in which this test suite can be run.
 - The core WebMDS tests can be run in a servlet container simulator. This tests the WebMDS code but does not test whether or not WebMDS has been deployed correctly into Tomcat:

```
ant -f $GLOBUS_LOCATION/etc/globus_wsrf_mds_webmds_test/build.xml test-installed
```

The output should look something like this:

```
Buildfile: GLOBUS_LOCATION/etc/globus_wsrf_mds_webmds_test/build.xml
```

```
test-installed:
```

```
[junit] Running org.globus.mds.webmds.test.PackageTests
[junit] Running org.globus.mds.webmds.test.SimpleServletTest tests with servlet
[junit] No webmds.test.servletURL property specified; skipping org.globus.mds.we
[junit] Running org.globus.mds.webmds.test.ServletXsltTests tests with servlet si
[junit] No webmds.test.servletURL property specified; skipping org.globus.mds.we
[junit] Tests run: 8, Failures: 0, Errors: 0, Time elapsed: 4.516 sec
```

```
BUILD SUCCESSFUL
```

```
Total time: 8 seconds
```

- The core WebMDS tests can be run against a running WebMDS server, to test the local WebMDS deployment:

```
ant \
  -f $GLOBUS_LOCATION/etc/globus_wsrf_mds_webmds_test/build.xml \
  "-Dwebmds.test.servletURL=http://webmds_host:webmds_port/webmds/webmds" \
  test-installed
```

The output should look something like this:

¹ <http://httpunit.sourceforge.net>

```
Buildfile: GLOBUS_LOCATION/etc/globus_wsrf_mds_webmds_test/build.xml
```

```
test-installed:
```

```
[junit] Running org.globus.mds.webmds.test.PackageTests
[junit] Running org.globus.mds.webmds.test.SimpleServletTest tests with servlet
[junit] Running org.globus.mds.webmds.test.SimpleServletTest tests against server
[junit] Running org.globus.mds.webmds.test.ServletXslTests tests with servlet si
[junit] Running org.globus.mds.webmds.test.ServletXslTests tests against server
[junit] Tests run: 8, Failures: 0, Errors: 0, Time elapsed: 5.229 sec
```

```
BUILD SUCCESSFUL
```

```
Total time: 8 seconds
```

The tests have passed if the number of failures and number of errors are both 0. Detailed test output can be found in the file `$GLOBUS_LOCATION/etc/globus_wsrf_mds_webmds_test/test-reports/TEST-org.globus.mds.webmds.test.PackageTests.xml`.

4. Run the WebMDS resource property node source test suite, to test the ability of WebMDS to query a running WS MDS Index Server. This test suite requires that both a secure Index server and an insecure Index server be running. As with the core tests, the resource property tests may be run in two modes.
 - The tests can be run in a servlet container simulator. This tests the WebMDS code, and the interaction between the WebMDS code and running Index servers, but does not test whether or not WebMDS has been deployed correctly into tomcat:

```
ant -f \
  $GLOBUS_LOCATION/etc/globus_wsrf_mds_webmds_resource_property_source_test/build.xml
  "-Dwebmds.rpTest.insecureServicePrefix=http://index_server_host:index_server_port/"
  "-Dwebmds.rpTest.secureServicePrefix=https://index_server_host:index_server_port/w
  test-installed
```

The output should look something like this:

```
Buildfile: GLOBUS_LOCATION/etc/globus_wsrf_mds_webmds_resource_property_source_test/
```

```
test-installed:
```

```
[junit] Running org.globus.mds.webmds.xmlSources.resourceProperties.test.Package
[junit] querying resource properties at 'http://insecure_index_server_host:insec
[junit] querying resource properties at 'https://secure_index_server_host:secure
[junit] Tests will use Globus servers at https://secure_index_server_host:secure
[junit] Running org.globus.mds.webmds.xmlSources.resourceProperties.test.Resourc
[junit] Tests will use Globus servers at https://secure_index_server_host:secure
[junit] No webmds.test.servletURL property specified; skipping org.globus.mds.we
[junit] Tests will use Globus servers at https://secure_index_server_host:secure
[junit] Tests run: 5, Failures: 0, Errors: 0, Time elapsed: 6.626 sec
```

```
BUILD SUCCESSFUL
```

```
Total time: 10 seconds
```

- To run an end-to-end test that tests the communication between a deployed WebMDS server and running index servers, do the following:

```
ant -f \  
  $GLOBUS_LOCATION/etc/globus_wsrf_mds_webmds_resource_property_source_test/build.xml  
  "-Dwebmds.rpTest.insecureServicePrefix=http://insecure_index_server_host:index_ser  
  "-Dwebmds.rpTest.secureServicePrefix=https://secure_index_server_host:index_server  
  "-Dwebmds.test.servletURL=http://webmds_host:webmds_port/webmds/webmds" \  
  test-installed
```

The output should look something like this:

```
Buildfile: GLOBUS_LOCATION/etc/globus_wsrf_mds_webmds_resource_property_source_test/
```

```
test-installed:  
 [junit] Running org.globus.mds.webmds.xmlSources.resourceProperties.test.Package  
 [junit] querying resource properties at 'http://insecure_index_server_host:insec  
 [junit] querying resource properties at 'https://secure_index_server_host:secure  
 [junit] Tests will use Globus servers at https://secure_index_server_host:secure  
 [junit] Running org.globus.mds.webmds.xmlSources.resourceProperties.test.Resourc  
 [junit] Tests will use Globus servers at https://secure_index_server_host:secure  
 [junit] Running org.globus.mds.webmds.xmlSources.resourceProperties.test.Resourc  
 [junit] Tests will use Globus servers at https://secure_index_server_host:secure  
 [junit] Tests run: 5, Failures: 0, Errors: 0, Time elapsed: 7.041 sec
```

```
BUILD SUCCESSFUL  
Total time: 10 seconds
```

The tests have passed if the number of failures and number of errors are both 0. Detailed test output can be found in the file *\$GLOBUS_LOCATION*/etc/globus_wsrf_mds_webmds_resource_property_source_test/test-reports/TEST-org.globus.mds.webmds.xmlSources.resourceProperties.test.PackageTests.xml.

Chapter 4. Security Considerations

1. WebMDS Security Considerations

By default, the WebMDS plugins distributed as part of the Toolkit do not use authentication credentials -- they retrieve information using anonymous SSL authentication or no authentication at all, and thus retrieve only publicly-available information.

The `ResourcePropertyNodeSource` and `ResourcePropertyQueryNodeSource` plugins can be configured either to allow users to specify what resources they want to query or to only allow users to query resources pre-configured by the web administrator. The standard WebMDS deployment allows users to specify the resources they want to query; to disallow this (for example, to ensure that people don't use your site's bandwidth to view information about some other site's services), remove the files `$GLOBUS_LOCATION/lib/webmds/conf/openEndedRP` and `$GLOBUS_LOCATION/lib/webmds/conf/openEndedQuery`.

DRAFT

Chapter 5. Debugging

1. Logging

As of 4.2.0, the Globus Toolkit provides system administration logs that are [CEDPs best practices](http://cedps.net/index.php/LoggingBestPractices)¹ compliant.

Configuration for this logger can be changed by editing `$GLOBUS_LOCATION/FIXME/path/to/cedpslogfile`.

For more details on the CEDPS Logging format, including descriptions of reserved name-value pairs, see <http://cedps.net/index.php/LoggingBestPractices>:

1.1. Configuring system administration logs

[FIXME the following is java core's info - tailor to this component] The specific logger to edit will be `log4j.logger.sysadmin` in `container-log4j.properties`. There you can configure the following properties:

```
log4j.appender.infoCategory=org.apache.log4j.RollingFileAppender
log4j.appender.infoCategory.Threshold=INFO
log4j.appender.infoCategory.File=var/containerLog
log4j.appender.infoCategory.MaxFileSize=10MB
log4j.appender.infoCategory.MaxBackupIndex=2
```

Above implies the logging file is rolling with each file size limited to 10MB and the logging information is stored in `$GLOBUS_LOCATION/var/containerLog`.

1.2. Sample log file

The [sample log file](#)² contains many log entries for various scenarios in the Java WS container [FIXME does this apply for your component? if not, can you provide a sample log file?].

¹ <http://cedps.net/index.php/LoggingBestPractices>

² <http://www.globus.org/toolkit/docs/4.2/4.2.0/common/javawscore/sample-container-log.txt>

Chapter 6. Troubleshooting

For a list of common errors in GT, see [Error Codes](#).

1. Error Messages

Error handling in WebMDS is currently done by throwing exceptions, which are displayed by Tomcat as stack traces.

Table 6.1. WS MDS Trigger Service Error Messages

Error Code	Definition
java.net.ConnectException: Connection refused	If you attempt to use WebMDS to collect information from a service that is not running, you will see a stack trace that includes the following exception: org.globus.mds.webmds.xmlSources.resourceProperties.ResourcePropertySourceException java.net.ConnectException: Connection refused
faultString: org.globus.common.ChainedIOException: Authentication failed [Caused by: Failure unspecified at GSS-API level [Caused by: Unknown CA]]	When WebMDS sends resource property queries to a secure WSRF service instance (such as an WS MDS Index Server), the service instance must have a certificate authority that issued the certificate used by the WSRF service instance. If the WebMDS server does not trust the certificate authority, the queries will produce a stack trace that includes this message.
WebMDS connections to secure Index Servers (or other secure WSRF servers) just hang	If the JVM used by Tomcat is configured to use a blocking random-number source, WebMDS connections to secure services can hang. This is the default configuration for many installations.

Glossary

I

Index Service

An aggregator service in WS MDS that serves as a registry similar to UDDI, but much more flexible. Indexes collect information and publish that information as WSRF resource properties.

DRAFT

Index

Symbols

\$GLOBUS_LOCATION/lib/webmds/conf, 1

\$GLOBUS_LOCATION/lib/webmds/conf/indexinfo, 1

C

configuration interface

 default, 1

 overview, 1

configuring

 default, 1

 monitor different Index Service, 1

 overview, 1

D

deploying, 4

 custom, 5

 standard, in Tomcat 5.0.28, 4

 WebMDS and GT in the same Tomcat Server, 4

E

errors, 12

I

information sources, 1

L

logging, 11

S

security considerations, 10

T

testing, 7

troubleshooting, 12

X

XML sources, 2

GT 4.2.0 WS MDS WebMDS: User's Guide

DRAFT

GT 4.2.0 WS MDS WebMDS: User's Guide

Introduction

WebMDS is a web-based interface for viewing formatted information about Grid resources. In the simplest instance, a web server administrator creates an HTML link that causes the WebMDS server to collect and format information that is presented to the user. Users can also use web forms to specify parameters that control what information is collected and how it's presented.

DRAFT

Table of Contents

- WebMDS Howtos 5
- 1. Visualizing the Index Service with WebMDS 1
- 2. Graphical User Interface 2
 - 1. Overview of the purpose and functionality of the GUI 2
 - 2. Command and options 2
 - 3. Customizing the web forms used to access WebMDS 2
 - 4. Limitations 2
- 3. Troubleshooting 3
 - 1. Error Messages 3
- Index 4

DRAFT

List of Tables

2.1. Form arguments used by WebMDS	2
3.1. WS MDS Trigger Service Error Messages	3

DRAFT

WebMDS Howtos

E

errors,

G

GUI interface,
 commands,
 customizing,
 limitations,
 options,
 overview,

T

troubleshooting,

U

using WebMDS,

DRAFT

Chapter 1. Visualizing the Index Service with WebMDS

Once you've deployed the WebMDS servlet, simply point your web browser at `http://your-tomcat-host:your-tomcat-port/webmds` and click on the link labelled "A list of resources registered to the local default index service". For more information, see Chapter 2, Graphical User Interface.

For more detailed information about changing the look of WebMDS and more advanced configuration, see the WebMDS Admin Guide.

DRAFT

Chapter 2. Graphical User Interface

1. Overview of the purpose and functionality of the GUI

The WebMDS GUI is a web-based interface for browsing formatted XML data, such as the results of resource property queries on a grid service.

2. Command and options

WebMDS can be accessed using any web browser. In a default WebMDS installation, the URL `http://host-name:port/webmds` corresponds to the top-level WebMDS web page. This page includes a link to a WebMDS invocation that provides summary information (with links to detailed information) about a locally-running MDS Index server. It also contains a link to a page of sample web forms demonstrating other uses of WebMDS.

3. Customizing the web forms used to access WebMDS

The WebMDS servlet is located at `http://your-tomcat-host:your-tomcat-port/webmds/webmds`. It takes the following arguments:

Table 2.1. Form arguments used by WebMDS

<code>info</code>	The name of the XML source that will be used to collect the raw XML data. XML sources are defined by files in <code>\$GLOBUS_LOCATION/lib/webmds/conf</code> . This argument must be specified.
<code>xsl</code>	The name of the XML source that will provide the XSL transform. XML sources are defined by files in <code>\$GLOBUS_LOCATION/lib/webmds/conf</code> . If this argument is not specified, the WebMDS servlet will display raw, untransformed XML.
<code>xml-Source.info_name.param.source_specific_options</code>	Any additional options recognized by the <code>info_name</code> XML source (<code>info_name</code> must be the value of the <code>info</code> argument for this request). Source-specific options are discussed in the next section.
<code>xml-Source.xsl_name.param.source_specific_options</code>	Any additional options recognized by the <code>xsl_name</code> XML source (<code>xsl_name</code> must be the value of the <code>xsl</code> argument for this request). Source-specific options are discussed in the next section.

4. Limitations

Error conditions (such as typographical errors in resource property names) are presented as stack traces, rather than user-friendly error messages.

Chapter 3. Troubleshooting

The commonly-used WebMDS plugins do resource property queries; the Globus Toolkit `wsrf-query` can be used to determine whether the desired information is available directly from the resource.

For a list of common errors in GT, see [Error Codes](#).

For debugging information, see [Chapter 5, Debugging](#)

1. Error Messages

Error handling in WebMDS is currently done by throwing exceptions, which are displayed by Tomcat as stack traces.

Table 3.1. WS MDS Trigger Service Error Messages

Error Code	Definition
<code>java.net.ConnectException: Connection refused</code>	If you attempt to use WebMDS to collect information from a service that is not running, you will see a stack trace that includes the following: <pre>org.globus.mds.webmds.xmlSources.resourceProperties.ResourcePropertySourceException: java.net.ConnectException: Connection refused</pre>
<code>faultString: org.globus.common.ChainedIOException: Authentication failed [Caused by: Failure unspecified at GSS-API level [Caused by: Unknown CA]]</code>	When WebMDS sends resource property queries to a secure WSRF service instance (such as an WS MDS Index Server), the service instance must have a certificate authority that issued the certificate used by the WSRF service instance. If the WebMDS server does not trust the certificate authority, the queries will produce a stack trace that includes this message.
WebMDS connections to secure Index Servers (or other secure WSRF servers) just hang	If the JVM used by Tomcat is configured to use a blocking random-number source, WebMDS connections to secure WSRF servers can hang. This is the default configuration for many installations.

Index

E

errors, 3

G

GUI interface, 2
 commands, 2
 customizing, 2
 limitations, 2
 options, 2
 overview, 2

T

troubleshooting, 3

U

using WebMDS, 1

DRAFT

GT 4.2.0 WS MDS WebMDS: Developer's Guide

DRAFT

GT 4.2.0 WS MDS WebMDS: Developer's Guide

Introduction

WebMDS is a web-based interface for viewing formatted information about Grid resources. Information is collected via a plugin interface and then formatted using an XSLT transform.

Figure 1. WebMDS Information Flow

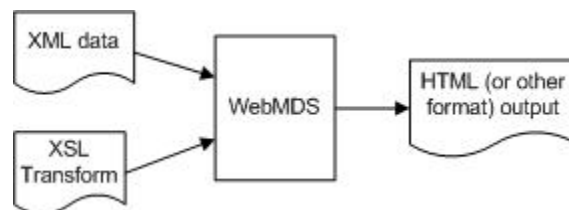


Table of Contents

WebMDS Howtos	6
1. Before you begin	1
1. Feature summary	1
2. Tested platforms	1
3. Backward compatibility summary	2
4. Technology dependencies	2
5. WebMDS Security Considerations	3
2. Usage scenarios	4
1. Creating a new plugin	4
2. Changing format of output	4
3. Architecture and design overview	5
4. APIs	6
1. Programming Model Overview	6
2. Component API	6
5. WebMDS Admin Commands	7
1. Tool description	7
2. Command syntax	7
3. Example	7
4. Limitations	7
6. Graphical User Interface	8
1. Overview of the purpose and functionality of the GUI	8
2. Command and options	8
3. Customizing the web forms used to access WebMDS	8
4. Limitations	8
7. Configuring	9
1. Configuration overview	9
2. Syntax of the interface	9
3. XML Sources included with WebMDS	10
8. Debugging	12
9. Troubleshooting	13
1. Error Messages	13
10. Related Documentation	14
Glossary	15
Index	16

List of Figures

1. WebMDS Information Flow	2
3.1. WebMDS Request Flow	5

DRAFT

List of Tables

6.1. Form arguments used by WebMDS	8
7.1. Pre-configured information sources	9
7.2. Configuration parameters used with FileXMLSource	10
7.3. Configuration parameters used with NodeXMLSource	10
7.4. Configuration parameters used with ResourcePropertyQueryNodeSource	11
7.5. Configuration parameters used with ResourcePropertyNodeSource	11
9.1. WS MDS Trigger Service Error Messages	13

DRAFT

WebMDS Howtos

Symbols

`$GLOBUS_LOCATION/lib/webmds/conf`,
`$GLOBUS_LOCATION/lib/webmds/conf/indexinfo`,

A

apis,
architecture,

C

changing format of the output,
compatibility,
configuration interface
 default,
 overview,
configuring
 default,
 monitor different Index Service,
 overview,
creating a new plugin,
creating Tomcat config files to deploy WebMDS,

D

dependencies,

E

errors,

F

features,

G

GUI interface,
 commands,
 customizing,
 limitations,
 options,
 overview,

I

information flow, 2
information sources,
installing
 on Windows,

P

platforms,

S

security considerations,

T

troubleshooting,

U

usage scenarios,
 changing format of output,
 creating a new plugin,

W

webmds-create-context-file,

X

XML sources,

Chapter 1. Before you begin

1. Feature summary

Features new in release 4.2.0:

- None

Other Supported Features

- Extensible plugin interface to support various mechanisms to gather monitoring information and XSLT transforms.
- Plugins to acquire monitoring information via resource property mechanisms.
- Plugin to acquire XSLT transforms by reading from local files.

Deprecated Features

- None

2. Tested platforms

Tested Platforms for WebMDS:

- The WebMDS server has only been tested with Tomcat version 5.0.28; it has been tested on RedHat Linux (i386) and, to a lesser extent, on Windows XP.
- On the client side, WebMDS should be accessible from any web browser on any platform.

2.1. Installing WebMDS on Windows

Although the WebMDS server is not officially supported on non-Unix platforms, and no Windows installer exists for WebMDS, it is possible to run WebMDS on Windows. The following instructions describe how to install WebMDS on a Windows platform.

1. Install [Tomcat](http://jakarta.apache.org/tomcat/)¹ and set your CATALINA_HOME environment variable to the directory into which Tomcat was installed.
2. Install the Globus Java WS-Core distribution from the [Globus Toolkit download page](http://www.globus.org/toolkit/downloads/)². Set your GLOBUS_LOCATION environment variable to the directory into which you installed Globus Java WS-Core
3. Check the ws-mds distribution out of the [Globus CVS repository](http://www.globus.org/toolkit/docs/development/remote-cvs.html)³, using the globus_4_0_branch tag.
4. Install the servicegroup package:

```
cd c:\wherever\ws-mds\servicegroup\schema
ant deploy
```

¹ <http://jakarta.apache.org/tomcat/>

² <http://www.globus.org/toolkit/downloads/>

³ <http://www.globus.org/toolkit/docs/development/remote-cvs.html>

```
cd ..\source
ant deploy
```

where *wherever* is the directory into which you checked out the ws-mds sources.

5. Install WebMDS:

```
cd c:\wherever\ws-mds\webmds
ant deploy
```

6. Create the webmds context file (this tells Tomcat where to find WebMDS):

```
%GLOBUS_LOCATION%\lib\webmds\bin\webmds-create-context-file %CATALINA_HOME%\conf\Catali
```

7. Restart Tomcat.

WebMDS can then be configured and used as described in the rest of the documentation: [WebMDS](#).

3. Backward compatibility summary

Protocol changes since GT version 4.0.x:

- None

API changes since GT version 4.0.x:

- None

Exception changes since GT version 4.0.x:

- None

Schema changes since GT version 4.0.x:

- None

4. Technology dependencies

WebMDS depends on the following GT components:

- Java WS Core

WebMDS depends on the following 3rd party software:

- [Tomcat](#)⁴

⁴ <http://jakarta.apache.org/tomcat/>

5. WebMDS Security Considerations

By default, the WebMDS plugins distributed as part of the Toolkit do not use authentication credentials -- they retrieve information using anonymous SSL authentication or no authentication at all, and thus retrieve only publicly-available information.

The `ResourcePropertyNodeSource` and `ResourcePropertyQueryNodeSource` plugins can be configured either to allow users to specify what resources they want to query or to only allow users to query resources pre-configured by the web administrator. The standard WebMDS deployment allows users to specify the resources they want to query; to disallow this (for example, to ensure that people don't use your site's bandwidth to view information about some other site's services), remove the files `$GLOBUS_LOCATION/lib/webmds/conf/openEndedRP` and `$GLOBUS_LOCATION/lib/webmds/conf/openEndedQuery`.

DRAFT

Chapter 2. Usage scenarios

There is no "client" programmatic interface to WebMDS; clients communicate using HTTP requests. The web form arguments recognized by WebMDS are documented in [User's Guide](#).

1. Creating a new plugin

To create a new plugin to collect raw XML data, write a Java class that implements the `WebmdsXmlSource` or `WebmdsNodeSource` interface. These are documented in [APIs](#). The `FileXmlSource` and `NodeXmlSource` classes distributed with WebMDS are examples of classes that implement `WebmdsXmlSource`; the `ResourcePropertyNodeSource` and `ResourcePropertyQueryNodeSource` classes distributed with WebMDS are examples of classes that implement the `WebmdsNodeSource` interface.

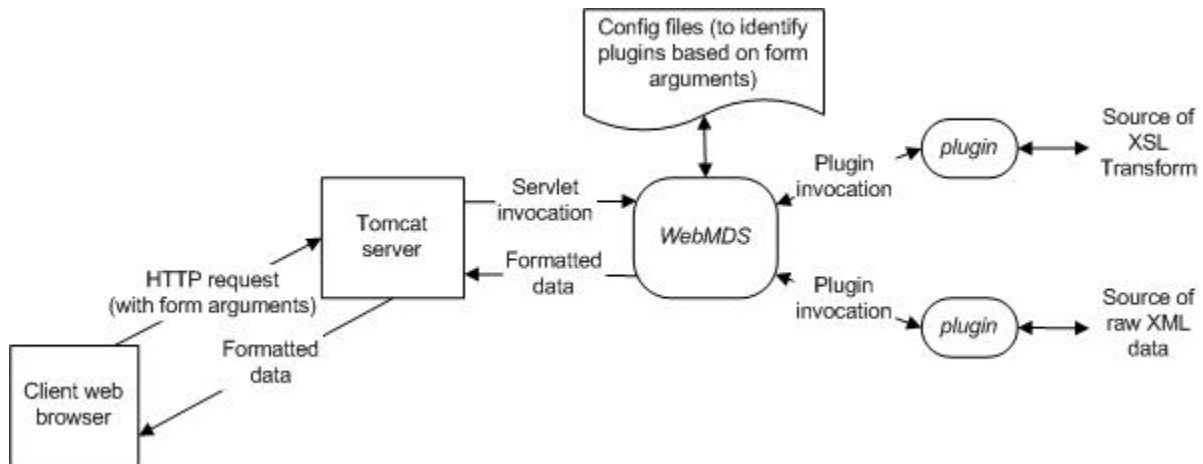
2. Changing format of output

To change the appearance of the output of WebMDS, create a new XSLT transform; see the [W3C XSLT Documentation](#)¹ for more information.

¹ <http://www.w3.org/TR/xslt>

Chapter 3. Architecture and design overview

Figure 3.1. WebMDS Request Flow



In a typical WebMDS transaction, a user uses a web browser to send an HTTP request, including some web form arguments, to a web server / servlet container. The web server invokes the WebMDS servlet, which uses the form arguments to determine what plugins to use to retrieve the requested XML data and the XSLT transform to apply to it. The WebMDS servlet passes arguments to the plugins, which then retrieve the appropriate data and XSLT transform. The WebMDS servlet applies the XSLT transformation to the XML data and returns the result to the web server, which sends it back to the client's web browser.

Chapter 4. APIs

1. Programming Model Overview

There is no "client" API for accessing WebMDS; WebMDS is a servlet that is accessed via web forms.

WebMDS uses a *WebMDS plugin* (a Java class that implements the `WebmdsXmlSource` interface) to acquire XML documents (which can be used either as raw information sources or as XSL transformations). WebMDS comes with two WebMDS plugins: `FileXmlSource`, which reads XML from a file (and is primarily used to acquire XSL transformations), and `NodeXmlSource`. `NodeXmlSource` in turn calls a *node source plugin* (a Java class that implements the `WebmdsNodeSource` interface) to acquire an XML DOM document. acquires XML information using a *WebMDS XML source*, a Java class that implements the `WebmdsXmlSource` interface. To summarize:

- WebMDS is a servlet that uses plugins to acquire XML documents containing raw data and XSL transformations, and then applies the acquired XSL transformation on the acquired data.
 - The plugins used by WebMDS implement the `org.globus.mds.webmds.WebmdsXmlSource` interface.
 - WebMDS plugins include:
 - `org.globus.mds.webmds.xmlSources.file.FileXmlSource`, which reads XML from a file, and
 - `org.globus.mds.webmds.xmlSources.xmlDomNode.NodeXmlSource`, which uses its own plugin interface to acquire XML DOM documents.
 - The plugins used by `NodeXmlSource` implement the `org.globus.mds.webmds.xmlSources.xmlDomNode.WebmdsNodeSource` interface
 - Node source plugins include `org.globus.mds.webmds.xmlSources.resourceProperties.ResourcePropertyNodeSource` and `org.globus.mds.webmds.xmlSources.resourceProperties.ResourcePropertyQueryNodeSource`, which acquire resource property information.
 - The raw XML data acquired by WebMDS is processed by XSL transformations; see the [W3C XSLT Documentation](#)¹ for more information on creating XSL transforms.

2. Component API

- [Core WebMDS documentation](#)² (includes the WebMDS servlet and the `WebmdsNodeSource` interface)
- [FileXMLSource documentation](#)³
- [NodeXmlSource documentation](#)⁴ (including the `WebmdsNodeSource` interface)
- [Resource property node source plugins](#)⁵.

¹ <http://www.w3.org/TR/xslt>

² http://www.globus.org/api/javadoc-4.0.0/globus_wsrf_mds_webmds/

³ http://www.globus.org/api/javadoc-4.0.0/globus_wsrf_mds_webmds_file_source/

⁴ http://www.globus.org/api/javadoc-4.0.0/globus_wsrf_mds_webmds_xml_dom_source/

⁵ http://www.globus.org/api/javadoc-4.0.0/globus_wsrf_mds_webmds_resource_property_source/

Chapter 5. WebMDS Admin Commands

There is no end-user command-line tool for WebMDS.

1. Tool description

The command-line tool `webmds-create-context-file` is used to create Tomcat configuration files needed to deploy WebMDS.

2. Command syntax

```
webmds-create-context-file [-f] tomcat_context_file
```

The `tomcat_context_file` argument is the location of the Tomcat configuration file defining the WebMDS context; in a default Tomcat installation, the location of this file will be `$CATALINA_HOME/conf/Catalina/localhost`.

By default, `webmds-create-context-file` will not overwrite an existing context file; the `-f` option is used to force `webmds-create-context-file` to overwrite an existing file.

Note: `webmds-create-context-file` is found in `$GLOBUS_LOCATION/lib/webmds/bin`

3. Example

```
$GLOBUS_LOCATION/lib/webmds/bin/webmds-create-context-file -f \  
$CATALINA_HOME/conf/Catalina/localhost
```

4. Limitations

Changes to the Tomcat context do not take effect until Tomcat is restarted or reloaded.

Chapter 6. Graphical User Interface

1. Overview of the purpose and functionality of the GUI

The WebMDS GUI is a web-based interface for browsing formatted XML data, such as the results of resource property queries on a grid service.

2. Command and options

WebMDS can be accessed using any web browser. In a default WebMDS installation, the URL `http://host-name:port/webmds` corresponds to the top-level WebMDS web page. This page includes a link to a WebMDS invocation that provides summary information (with links to detailed information) about a locally-running MDS Index server. It also contains a link to a page of sample web forms demonstrating other uses of WebMDS.

3. Customizing the web forms used to access WebMDS

The WebMDS servlet is located at `http://your-tomcat-host:your-tomcat-port/webmds/webmds`. It takes the following arguments:

Table 6.1. Form arguments used by WebMDS

<code>info</code>	The name of the XML source that will be used to collect the raw XML data. XML sources are defined by files in <code>\$GLOBUS_LOCATION/lib/webmds/conf</code> . This argument must be specified.
<code>xsl</code>	The name of the XML source that will provide the XSL transform. XML sources are defined by files in <code>\$GLOBUS_LOCATION/lib/webmds/conf</code> . If this argument is not specified, the WebMDS servlet will display raw, untransformed XML.
<code>xml-Source.info_name.param.source_specific_options</code>	Any additional options recognized by the <code>info_name</code> XML source (<code>info_name</code> must be the value of the <code>info</code> argument for this request). Source-specific options are discussed in the next section.
<code>xml-Source.xsl_name.param.source_specific_options</code>	Any additional options recognized by the <code>xsl_name</code> XML source (<code>xsl_name</code> must be the value of the <code>xsl</code> argument for this request). Source-specific options are discussed in the next section.

4. Limitations

Error conditions (such as typographical errors in resource property names) are presented as stack traces, rather than user-friendly error messages.

Chapter 7. Configuring

1. Configuration overview

WebMDS can be configured to get information from any of various sources and to filter it through any XSL transform. WebMDS uses configuration files to specify the location of (and to name) sources of information and xsl and web form arguments to select among these configured information sources and xsl transforms.

By default, WebMDS comes configured to report information about an index server using transaction-level security on the default port (8443) on the local system. If you are running the Globus Toolkit in this default configuration, then you can use WebMDS to query your local *Index Service* without any configuration changes.

If you wish to monitor a different Index Service, you will need to edit the file `$GLOBUS_LOCATION/lib/webm-
ds/conf/indexinfo` to change the URL in the line:

```
<value>https://127.0.0.1:8443/wsrf/services/DefaultIndexService</value>
```

to match the URL of your default index service. Changes to WebMDS configuration files take effect the next time that Tomcat is restarted.

For other configuration changes (e.g., monitoring different kinds of services), see the detailed configuration information below.

2. Syntax of the interface

Each configuration file in `$GLOBUS_LOCATION/lib/webm-
ds/conf` defines a source of XML, which can be used in an HTML form to specify sources of information and XSL transforms. The distribution contains some standard configuration files in this directory, including:

Table 7.1. Pre-configured information sources

<code>indexinfo</code>	all resource properties from an index server running with transaction-level security on port 8443 on the local host
<code>indexinfo_nosec</code>	all resource properties from an index server running with no security on port 8080 on the local host
<code>openEndedQuery</code>	all resource properties from a user-specified grid service
<code>openEndedRP</code>	a user-specified resource property from a user-specified grid service
<code>servicegroupxsl</code>	an xsl transform that presents summary information about a service group
<code>sgedetail</code>	an XSL transform that presents detailed information about a service group entry

Each configuration file defines a `Webm-
dsConfig` object. A `Webm-
dsConfig` object consists of:

- A `description`: a textual description of the XML source being defined.
- A `className`: the name of the Java class that will be used to acquire the XML data.
- Zero or more `parameter` objects, each of which consists of the name of some parameter recognized by the Java class specified by `className`, and the string value of that parameter.

For example, this is `$GLOBUS_LOCATION/lib/webmds/conf/servicegroupxml`, which defines the `servicegroupxml` XML source:

```
<WebmdsConfig>
  <description>
    XSL file to show service group summary information
  </description>
  <className>org.globus.mds.webmds.xmlSources.file.FileXmlSource</className>
  <parameter>
    <name>file</name>
    <value>xslfiles/servicegrouptable.xml</value>
  </parameter>
</WebmdsConfig>
```

This file tells WebMDS to use the `org.globus.mds.webmds.xmlSources.file.FileXmlSource` Java class (a class which reads XML from a local file) to collect XML data and to pass a `file` parameter (which that Java class interprets as the name of the file to open, relative to the WebMDS base directory).

Tomcat must be restarted (or one of the more advanced Tomcat administrative mechanisms must be used) for changes to these configuration files to take effect.

3. XML Sources included with WebMDS

3.1. FileXMLSource

The class `org.globus.mds.webmds.xmlSources.file.FileXmlSource` reads XML from a file, and recognizes a single parameter:

Table 7.2. Configuration parameters used with FileXMLSource

<code>file</code>	The name of the file to read. Relative path names are interpreted relative to the WebMDS base directory (<code>\$GLOBUS_LOCATION/lib/webmds</code>).
-------------------	--

3.2. NodeXMLSource

This XML source class uses a `WebmdsNodeSource` object to fetch an XML document and return it in a form that is usable by WebMDS. It recognizes the following options:

Table 7.3. Configuration parameters used with NodeXMLSource

<code>class</code>	The name of a class that implements the <code>WebmdsNodeSource</code> interface. An instance of this class will be used to get an XML document.
<code>parameters</code>	Additional parameters are passed to an instance of the class specified by the <code>class</code> argument.

3.3. Classes That Implement WebmdsNodeSource

The following classes implement the `NodeXMLSource` interfaces and can be used in conjunction with `NodeXMLSource`

3.4. ResourcePropertyQueryNodeSource

This class performs a resource property query to get all the resource properties for some web service. It recognizes the following configuration parameters:

Table 7.4. Configuration parameters used with ResourcePropertyQueryNodeSource

endpoint	The endpoint name to be used in a resource property query.
endpointKeyName and endpointKeyValue	An optional key/value pair to use as reference properties for the endpoint specified with the endpoint parameter.
allowUserEndpoints	If true, values for <code>xmlSource.sourceName.param.endpoint</code> , <code>xmlSource.sourceName.param.endpointKeyName</code> , and <code>xmlSource.sourceName.param.endpointKeyValue</code> specified in the request will override the configured endpoint value.
endpointFile	The name of a file from which the endpoint information (in XML) will be read. This configuration parameter can never be overridden by request arguments.

3.5. ResourcePropertyNodeSource

This class queries a web service for a single resource property. It recognizes the following parameters:

Table 7.5. Configuration parameters used with ResourcePropertyNodeSource

endpoint	The endpoint name to be used in a resource property query.
endpointKeyName and endpointKeyValue	An optional key/value pair to use as reference properties for the endpoint specified with the endpoint parameter.
allowUserEndpoints	If true, values for <code>xmlSource.sourceName.param.endpoint</code> , <code>xmlSource.sourceName.param.endpointKeyName</code> , and <code>xmlSource.sourceName.param.endpointKeyValue</code> specified in the request will override the configured endpoint value.
endpointFile	The name of a file from which the endpoint information (in XML) will be read. This configuration parameter can never be overridden by request arguments.
rpNamespace	The namespace part of the QName of the resource property to be queried for.
rpName	The local name part of the QName of the resource property to be queried for.
allowUserResourceProperties	If true, values of <code>xmlSource.sourceName.param.rpNamespace</code> and <code>xmlSource.sourceName.param.rpNames</code> specified in the request will override the configured resource property namespace and name.

Chapter 8. Debugging

For information on sys admin logs, see [Chapter 5, Debugging](#).

Log information from WebMDS and any WebMDS plugins will be logged by the servlet container into which WebMDS has been deployed. In a vanilla Tomcat 5.0.28 distribution, this information will show up in the file `$CATALINA_HOME/logs/catalina.out`.

DRAFT

Chapter 9. Troubleshooting

For a list of common errors in GT, see [Error Codes](#).

1. Error Messages

Error handling in WebMDS is currently done by throwing exceptions, which are displayed by Tomcat as stack traces.

Table 9.1. WS MDS Trigger Service Error Messages

Error Code	Definition
java.net.ConnectException: Connection refused	If you attempt to use WebMDS to collect information from a service that is not running, you will see a stack trace that includes the following exception: org.globus.mds.webmds.xmlSources.resourceProperties.ResourcePropertySourceException java.net.ConnectException: Connection refused
faultString: org.globus.common.ChainedIOException: Authentication failed [Caused by: Failure unspecified at GSS-API level [Caused by: Unknown CA]]	When WebMDS sends resource property queries to a secure WSRF service instance (such as an WS MDS Index Server), the service instance must be configured with a certificate authority that issued the certificate used by the WSRF service instance. If the WebMDS server does not trust the certificate authority, the queries will produce a stack trace that includes this message.
WebMDS connections to secure Index Servers (or other secure WSRF servers) just hang	If the JVM used by Tomcat is configured to use a blocking random-number source, WebMDS connections to secure services can hang. This is the default configuration for many installations.

Chapter 10. Related Documentation

None available at this time.

DRAFT

Glossary

I

Index Service

An aggregator service in WS MDS that serves as a registry similar to UDDI, but much more flexible. Indexes collect information and publish that information as WSRF resource properties.

DRAFT

Index

Symbols

`$GLOBUS_LOCATION/lib/webmds/conf`, 9

`$GLOBUS_LOCATION/lib/webmds/conf/indexinfo`, 9

A

apis, 6

architecture, 5

C

changing format of the output, 4

compatibility, 2

configuration interface

 default, 9

 overview, 9

configuring

 default, 9

 monitor different Index Service, 9

 overview, 9

creating a new plugin, 4

creating Tomcat config files to deploy WebMDS, 7

D

dependencies, 2

E

errors, 13

F

features, 1

G

GUI interface, 8

 commands, 8

 customizing, 8

 limitations, 8

 options, 8

 overview, 8

I

information flow, 2

information sources, 9

installing

 on Windows, 1

P

platforms, 1

S

security considerations, 3

T

troubleshooting, 13

U

usage scenarios, 4

 changing format of output, 4

 creating a new plugin, 4

W

`webmds-create-context-file`, 7

X

XML sources, 10

GT 4.2.0 WS MDS Migration Guide

DRAFT

GT 4.2.0 WS MDS Migration Guide

Introduction

The following provides available information about migrating from previous versions of the Globus Toolkit.

DRAFT

Table of Contents

1. Migrating MDS from GT4	1
2. Migrating MDS from GT3	2
3. Migrating MDS from GT2	3
Glossary	4

DRAFT

List of Tables

1.1. Comparison of MDS in GT3 and GT4	1
2.1. Comparison of MDS in GT3 and GT4	2
3.1. Comparison of MDS in GT2 and GT4	3

DRAFT

Chapter 1. Migrating MDS from GT4

Although the basic functionality remains the same for MDS in GT4, the architecture has changed from OGSi in GT3 to WSRF in GT4. In OGSi, services advertise *service data*; in WSRF, services advertise *resource properties*. Resource Properties and service data are very similar -- both provide a mechanism for expressing arbitrary data about grid resources in XML format, as well as query and notification/subscription interfaces to that data.

The GT4 *Index Service* provides the same functionality as the GT3 Index Service; however, the GT4 Index Service supports WSRF service group registration and resource property query and subscription/notification mechanisms, while the GT3 Index Service supported OGSi service group registration and service data query and subscription/notification mechanisms.

The following table shows a mapping of some GT3 concepts/tools to GT4.

Table 1.1. Comparison of MDS in GT3 and GT4

Description	GT2 Version	GT4 Version
Query Operations	FindServiceData (to retrieve a single service data element by name or to perform an XPath query against a service's service data elements)	GetResourceProperty (to retrieve a single resource property by name), GetMultipleResourceProperties (to retrieve multiple resource properties by name), and QueryResourceProperties (to perform an XPath query against a service's resource properties).
APIs used for queries	OGSi (GT3) Core APIs	WS Core APIs
Command-line clients used for queries	<code>ogsi-find-service-data</code>	<code>wsrf-get-property</code> , <code>wsrf-get-properties</code> , <code>wsrf-query</code>
Available GUIs	globus-sdb (standalone client) and WebSDB (web interface)	WebMDS (web interface)
Operations for subscription/notification	OGSi NotificationSource / NotificationSink	WS-Notification
APIs used for subscription/notification	OGSi (GT3) Core APIs	WS Core APIs
Index registration mechanism	GT3 services can be configured to publish their service data to index services.	Index Servers maintain aggregating service groups that include registration information (timeout values, the mechanism to use to acquire information, and additional mechanism-specific parameters) The registration is accomplished by adding an entry to an aggregating service group via the <code>mds-servicegroup-add</code> command. In addition, services may be configured to register themselves to the default index server running in the same container.

A more detailed mapping of OGSi concepts to WSRF concepts can be found [here](http://www-106.ibm.com/developerworks/library/ws-resource/ogsi_to_wsrf_1.0.pdf)¹.

¹ http://www-106.ibm.com/developerworks/library/ws-resource/ogsi_to_wsrf_1.0.pdf

Chapter 2. Migrating MDS from GT3

Although the basic functionality remains the same for MDS in GT4, the architecture has changed from OGSi in GT3 to WSRF in GT4. In OGSi, services advertise *service data*; in WSRF, services advertise *resource properties*. Resource Properties and service data are very similar -- both provide a mechanism for expressing arbitrary data about grid resources in XML format, as well as query and notification/subscription interfaces to that data.

The GT4 *Index Service* provides the same functionality as the GT3 Index Service; however, the GT4 Index Service supports WSRF service group registration and resource property query and subscription/notification mechanisms, while the GT3 Index Service supported OGSi service group registration and service data query and subscription/notification mechanisms.

The following table shows a mapping of some GT3 concepts/tools to GT4.

Table 2.1. Comparison of MDS in GT3 and GT4

Description	GT2 Version	GT4 Version
Query Operations	FindServiceData (to retrieve a single service data element by name or to perform an XPath query against a service's service data elements)	GetResourceProperty (to retrieve a single resource property by name), GetMultipleResourceProperties (to retrieve multiple resource properties by name), and QueryResourceProperties (to perform an XPath query against a service's resource properties).
APIs used for queries	OGSi (GT3) Core APIs	WS Core APIs
Command-line clients used for queries	<code>ogsi-find-service-data</code>	<code>wsrf-get-property</code> , <code>wsrf-get-properties</code> , <code>wsrf-query</code>
Available GUIs	globus-sdb (standalone client) and WebSDB (web interface)	WebMDS (web interface)
Operations for subscription/notification	OGSi NotificationSource / NotificationSink	WS-Notification
APIs used for subscription/notification	OGSi (GT3) Core APIs	WS Core APIs
Index registration mechanism	GT3 services can be configured to publish their service data to index services.	Index Servers maintain aggregating service groups that include registration information (timeout values, the mechanism to use to acquire information, and additional mechanism-specific parameters) The registration is accomplished by adding an entry to an aggregating service group via the <code>mds-servicegroup-add</code> command. In addition, services may be configured to register themselves to the default index server running in the same container.

A more detailed mapping of OGSi concepts to WSRF concepts can be found [here](http://www-106.ibm.com/developerworks/library/ws-resource/ogsi_to_wsrf_1.0.pdf)¹.

¹ http://www-106.ibm.com/developerworks/library/ws-resource/ogsi_to_wsrf_1.0.pdf

Chapter 3. Migrating MDS from GT2

Although the basic functionality remains the same for MDS in GT4, the architecture, standards used, and implementation have changed significantly in GT2. The following table shows a mapping of some GT2 concepts to GT4 concepts.

Table 3.1. Comparison of MDS in GT2 and GT4

Description	GT2 Version	GT4 Version
Format of data describing a resource	LDAP data hierarchy	XML data document
Query language	LDAP queries	XPath queries
Wire protocol for queries	LDAP	WS-ResourceProperties
APIs used for queries	LDAP APIs	WS Core APIs
Command-line clients used for queries	<code>grid-info-search</code>	<code>wsrf-get-property</code> , <code>wsrf-get-properties</code> , <code>wsrf-query</code>
Available GUIs	Various LDAP browsers	WebMDS
Wire protocol for subscription/notification	Not supported	WS-Notification
APIs used for subscription/notification	Not supported	WS Core APIs
Security support	SAML-based security using X.509 user, proxy and host certificates	HTTPS-based security using X.509 user, proxy and host certificates
Queryable index of aggregated information	GIIS, which publishes data using the LDAP-related standards listed above	WS MDS Index Server, which publishes data using the WSRF-related standards listed above
Queryable source of non-aggregated information	GRIS, which uses <i>information providers</i> to gather data from services and then publishes that data the LDAP-related standards listed above	Individual web services, which publish data about their own resources using WSRF-related standards listed above.
Index registration mechanism	MDS servers (GRIS's and, in some cases, GIIS's) register themselves with a GIIS. An MDS server is configured to register itself to a remote index by editing the local MDS server's <code>grid-info-resource-register.conf</code> file, providing information about the location of the remote index to register to and timeout values for the registration	WS MDS Index servers maintain aggregating service groups that include registration information (timeout values, the mechanism to use to acquire information, and additional mechanism-specific parameters) The registration is accomplished by adding an entry to an aggregating service group via the <code>mds-servicegroup-add</code> command. In addition, services may be configured to register themselves to the default index server running in the same container.
Mechanism used by an index to collect information	GIIS's send LDAP queries to remote serves.	WS MDS Index servers use a plugin-based architecture to support several mechanisms to collect information. The Globus Toolkit supplies plugins that support collecting information via polling (resource property queries), subscription/notification, and by program execution.

Glossary

I

Index Service

An aggregator service in WS MDS that serves as a registry similar to UDDI, but much more flexible. Indexes collect information and publish that information as WSRF resource properties.

information provider

A "helper" software component that collects or formats resource information, for use in WS MDS by an aggregator source or by a WSRF service when creating resource properties.

DRAFT

GT 4.2.0 WS MDS WebMDS: Quality Profile

Table of Contents

1. Test coverage reports	1
2. Code analysis reports	1
3. Outstanding bugs	1
4. Bug Fixes	1
5. Performance reports	1

<titleabbrev>Quality Profile</titleabbrev>

1. Test coverage reports

- None available at this time.

2. Code analysis reports

- None available at this time.

3. Outstanding bugs

- [3040: Webmds can break if started from the wrong directory](#)¹
- [3051: Handle huge indexes](#)²

4. Bug Fixes

- [3926: Misconfigured RFT has a messy entry in WebMDS](#)³
- [4534: Improve error messages in WebMDS](#)⁴

5. Performance reports

- None available at this time.

¹ http://bugzilla.globus.org/globus/show_bug.cgi?id=3040

² http://bugzilla.globus.org/globus/show_bug.cgi?id=3051

³ http://bugzilla.globus.org/globus/show_bug.cgi?id=3926

⁴ http://bugzilla.globus.org/globus/show_bug.cgi?id=4534

GT 4.2.0 Development Release Notes for WS MDS WebMDS

Table of Contents

1. Component Overview	1
2. Feature summary	1
3. Summary of Changes in WebMDS	1
4. Bug Fixes	2
5. Known Problems	2
6. Technology dependencies	2
7. Tested platforms	2
8. Backward compatibility summary	3
9. Associated standards for WS MDS WebMDS	4
10. For More Information	4

<titleabbrev>Release Notes</titleabbrev>

1. Component Overview

WebMDS enables end users to view monitoring information via a standard web browser interface, without installing any additional software on their PC. WebMDS is implemented as a servlet that uses a plugin interface to gather monitoring information (or any other information in XML format) and XSLT transforms, and present the data to the user in a readable form. Web site administrators can customize their own WebMDS deployments by using HTML form options, configuring different plugins to collect data and XSLT transforms, and creating their own plugins and XSLT transforms.

2. Feature summary

Features new in release 4.2.0:

- None

Other Supported Features

- Extensible plugin interface to support various mechanisms to gather monitoring information and XSLT transforms.
- Plugins to acquire monitoring information via resource property mechanisms.
- Plugin to acquire XSLT transforms by reading from local files.

Deprecated Features

- None

3. Summary of Changes in WebMDS

The following changes have occurred for WS MDS WebMDS since the last stable release, 4.0.x:

- Error handling has improved -- error pages now include a summary and not just a Java stack trace.
- The service group summary view now has an option to periodically refresh the display.
- WebMDS now supports optional user-specified Xpath queries.
- WebMDS now supports the use of user-specified namespace mappings with user-specified Xpath queries.
- Entries in the service group summary view are now sorted.
- The service group summary view is now easier to update to support local data types.

4. Bug Fixes

- [3926: Misconfigured RFT has a messy entry in WebMDS](#)¹
- [4534: Improve error messages in WebMDS](#)²

5. Known Problems

The following problems and limitations are known to exist for WS MDS WebMDS at the time of the 4.2.0 release:

5.1. Limitations

- No known limitations exist.

5.2. Outstanding bugs

- [3040: Webmds can break if started from the wrong directory](#)³
- [3051: Handle huge indexes](#)⁴

6. Technology dependencies

WebMDS depends on the following GT components:

- Java WS Core

WebMDS depends on the following 3rd party software:

- [Tomcat](#)⁵

7. Tested platforms

Tested Platforms for WebMDS:

¹ http://bugzilla.globus.org/globus/show_bug.cgi?id=3926

² http://bugzilla.globus.org/globus/show_bug.cgi?id=4534

³ http://bugzilla.globus.org/globus/show_bug.cgi?id=3040

⁴ http://bugzilla.globus.org/globus/show_bug.cgi?id=3051

⁵ <http://jakarta.apache.org/tomcat/>

- The WebMDS server has only been tested with Tomcat version 5.0.28; it has been tested on RedHat Linux (i386) and, to a lesser extent, on Windows XP.
- On the client side, WebMDS should be accessible from any web browser on any platform.

7.1. Installing WebMDS on Windows

Although the WebMDS server is not officially supported on non-Unix platforms, and no Windows installer exists for WebMDS, it is possible to run WebMDS on Windows. The following instructions describe how to install WebMDS on a Windows platform.

1. Install [Tomcat](#)⁶ and set your CATALINA_HOME environment variable to the directory into which Tomcat was installed.
2. Install the Globus Java WS-Core distribution from the [Globus Toolkit download page](#)⁷. Set your GLOBUS_LOCATION environment variable to the directory into which you installed Globus Java WS-Core
3. Check the ws-mds distribution out of the [Globus CVS repository](#)⁸, using the globus_4_0_branch tag.
4. Install the servicegroup package:

```
cd c:\wherever\ws-mds\servicegroup\schema
ant deploy
cd ..\source
ant deploy
```

where *wherever* is the directory into which you checked out the ws-mds sources.

5. Install WebMDS:

```
cd c:\wherever\ws-mds\webmds
ant deploy
```

6. Create the webmds context file (this tells Tomcat where to find WebMDS):

```
%GLOBUS_LOCATION%\lib\webmds\bin\webmds-create-context-file %CATALINA_HOME%\conf\Catali
```

7. Restart Tomcat.

WebMDS can then be configured and used as described in the rest of the documentation: [WebMDS](#).

8. Backward compatibility summary

Protocol changes since GT version 4.0.x:

- None

⁶ <http://jakarta.apache.org/tomcat/>

⁷ <http://www.globus.org/toolkit/downloads/>

⁸ <http://www.globus.org/toolkit/docs/development/remote-cvs.html>

API changes since GT version 4.0.x:

- None

Exception changes since GT version 4.0.x:

- None

Schema changes since GT version 4.0.x:

- None

9. Associated standards for WS MDS WebMDS

- HyperText Transfer Protocol (HTTP)
- HyperText Markup Language (HTML)
- XSL Transformations (XSLT)
- WebMDS is implemented as a Java Servlet

10. For More Information

See [WebMDS](#) for more information about this component.