

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 rpNames</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 be configured to trust the 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 Index Servers 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