OGSA Hosting Environment Logging Service

Mike Williams (mdw@us.ibm.com)
John Wiley (wiley@us.ibm.com)

May 20, 2003

Draft 1.0

Preface

A core (built-in) grid service is needed within every OGSA hosting environment instance for the purpose of performing logging administration and management for the hosting environment and services running in that hosting environment. This service will be used by the IBM OGSA Management UI, however as a service, this could also be exploited by other management applications that are aware of and interested in the OGSA runtime.

Note this service will be built using Java objects such that it is capable of running in all of the supported Globus runtime environments. In the future, an EJB version may be considered.

Requirements

The current requirements for this service include:

1. The ability to administer and manage the runtime logging implementation. This includes:
    a. The ability to dynamically change the runtime logging characteristics of the hosting environment and the services running in that hosting environment.
    b. The ability to change the persistent logging configuration.
    c. The ability to view a window (subset) of the logs as service data.
2. A logging implementation abstraction[1] that would allow for plug-able implementations to be supported.

In the future, additional logging functions and capabilities may be added to this service.

[1] Note, Apache Commons provides an appropriate abstraction for a plug-able logging implementation for code that wishes to instrument itself for logging. However, Commons does not provide an abstraction for the controls (knobs) for an underlying implementation which is required here.

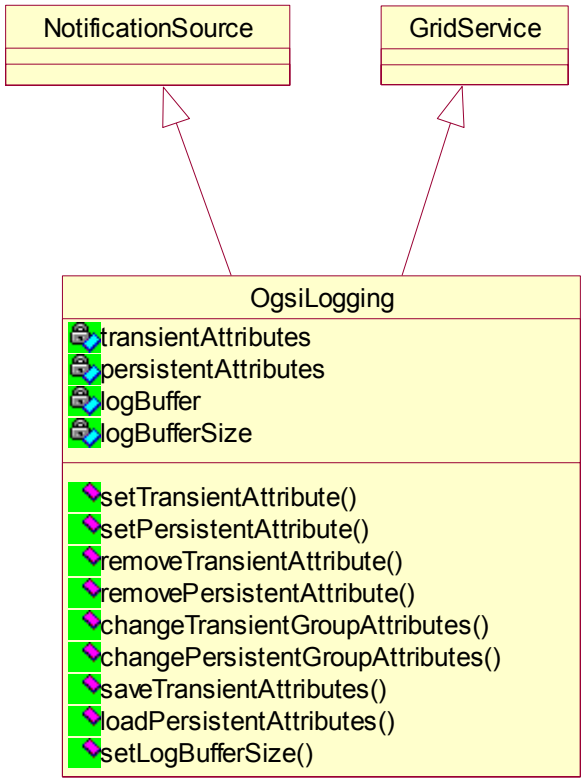
Design Details

The OGSA hosting environment logging service is a persistent service that implements the OgsiLoggingPortType. The OgsiLoggingPortType exposes the controls for the underlying logging implementation as a collection of runtime (transient) and configuration (persistent) attributes. This portType is described in more detail below.

Models and Diagrams

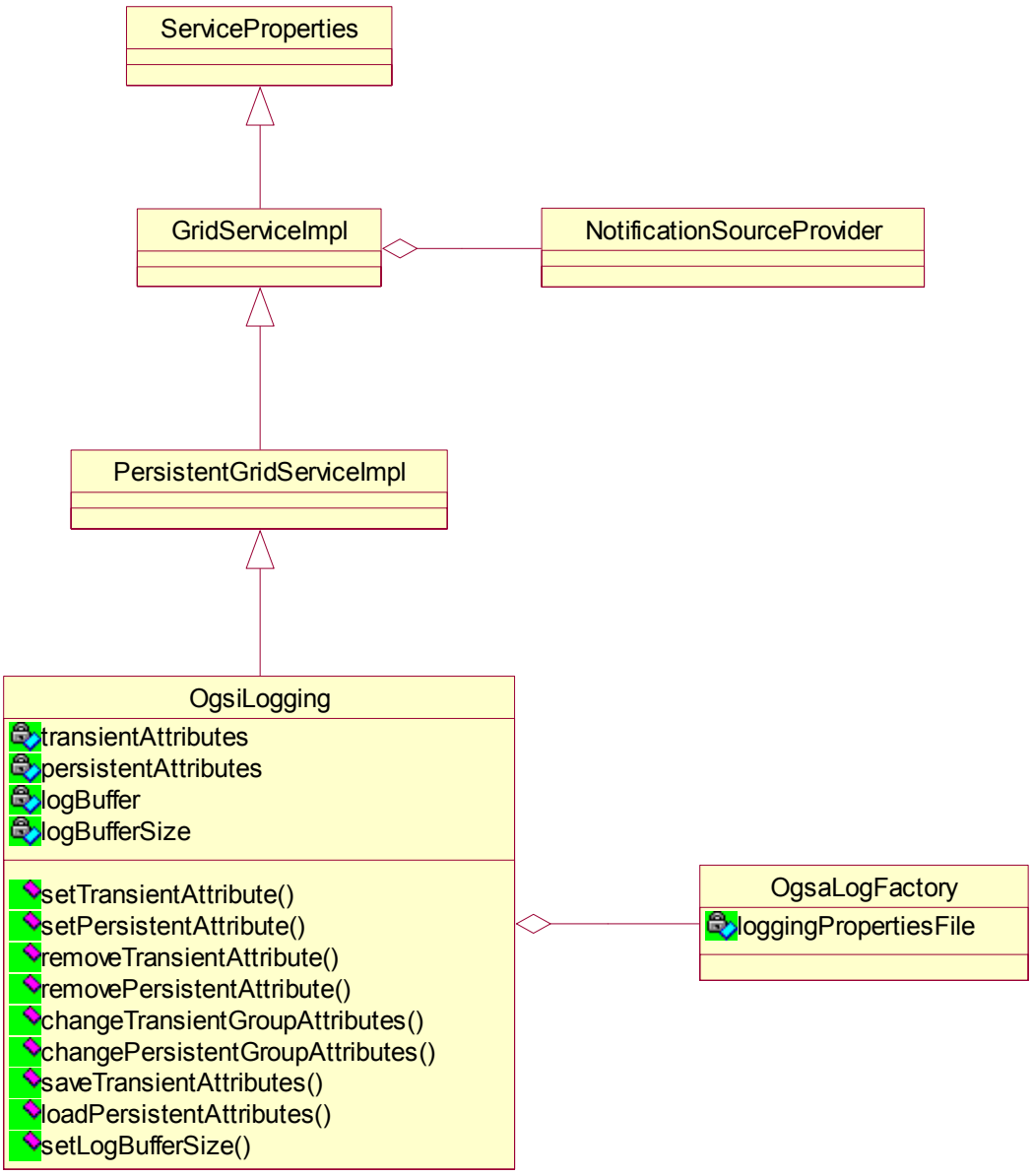
Logical Object Model

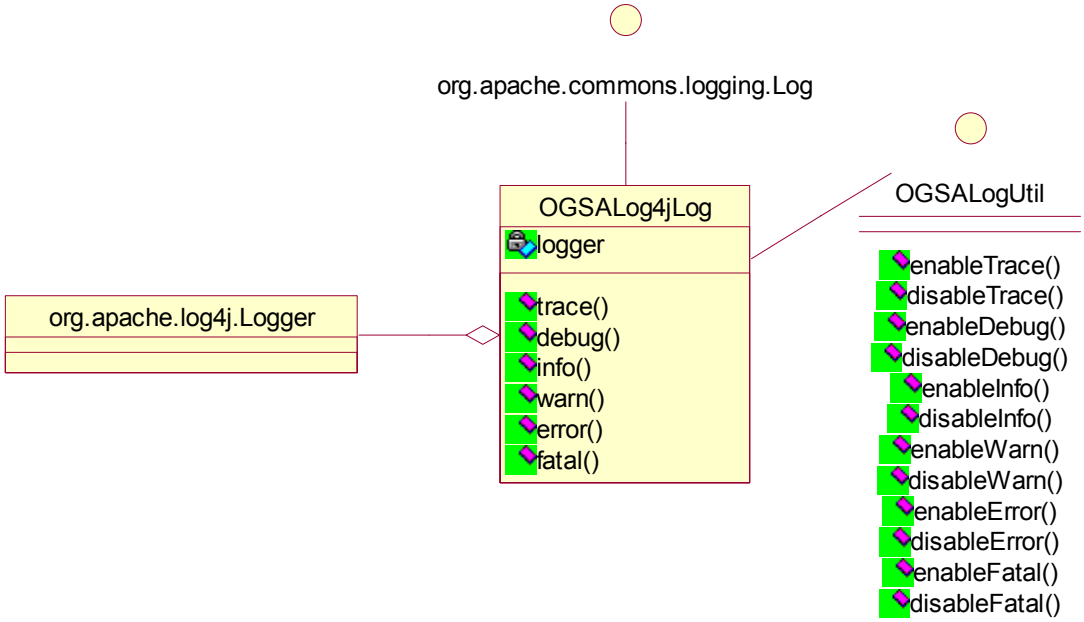
The following diagram is the logical object model as described via the gwsdl:portType for the logging service. The logging service portType extends both NotificationSource and GridService. Note, the entire gswdl portType for this service is included later in the document.



Class Diagram

In support of the logical diagram above, the following represents the actual implementation class diagram. In this case, the OgsiLogging implementation is an extension of the PersistentGridServiceImpl.


ServiceProperties
GridServiceImpl
NotificationSourceProvider
PersistentGridServiceImpl
OgsiLogging
transientAttributes
persistentAttributes
logBuffer
logBufferSize
setTransientAttribute()
setPersistentAttribute()
removeTransientAttribute()
removePersistentAttribute()
changeTransientGroupAttributes()
changePersistentGroupAttributes()
saveTransientAttributes()
loadPersistentAttributes()
setLogBufferSize()
OgsaLogFactory
loggingPropertiesFile



OgsiLoggingPortType

The OgsiLoggingPortType allows the logging for the runtime (hosting environment) to be administered. Administration is via named loggers or attributes. Attributes can be either persistent or transient. As the name implies, persistent attributes are persisted to a file and used at the hosting environment startup time. Transient attributes include the current in-memory set of attributes that will initially be populated from the set of persistent attributes and can subsequently be changed via the operations in this portType.

What are Attributes, Transient or Otherwise

Basically these are the set of data needed to manage logging. There are four key elements:

1. The logger name
2. The logging level
3. The log destination
4. The containing group

Logger name
The first item, "name" is known by the log generating application. It is typically the class name of the application generating the log information, or any name that binds a set of contributors to an identified log stream.

Logging Level
The other 3 elements are control points introduced by an administrator of logging activity. Historically some of these have been found in various properties files of logging

implementations. The level is the key log-filtering item. There are two views of level, the application view, and the administrative view. From the standpoint of the log generating application, it selects one of the following levels indicating the overall log priority:

1. DEBUG
2. TRACE
3. INFO
4. WARN
5. ERROR
6. FATAL

The level attribute contained herein is of the administrative variety, and is used in a hierarchical filtering of log message flows. Legal level attributes, in hierarchical order are:

1. ALL or DEBUG
2. TRACE
3. INFO
4. WARN
5. ERROR
6. FATAL
7. OFF

The administrator will set a single level associated with a logger name, from one of the above. The selection of x (being a number) enables all log filters below it, with "OFF" being a special case disabling all log messages for this named logger. So, with ALL or DEBUG you get all priorities (log generator perspective). With TRACE, you also enable INFO, WARN, ERROR, FATAL. With INFO, WARN, ERROR, FATAL, and so on

Transient attributes refer to the current cache of log attribute control points. Manipulation of this class of attributes, immediately impacts the allowed priorities of message flows for a given logger. Setting persistent attributes only affects the saved configuration data. The existence of "persistent" attributes is an administrative option, and not required for operation. The logging components that comprise this service will, if found, load and initially cache (make transient), all persistent attributes, at logFactory initialization time. Assuming a commons implementation, this would occur dynamically at log use, first reference.

Note: an initial config file could be created with the saveTransientAttributes() operation. Additionally, a newly updated persistent attribute can be put in effect, by the loadPersistentAttributes() operation (see Special Attributes and Parameters discussion for additional details).

Log destination

The destination field is either "CONSOLE" or filename. The filename will be pre-pended by the contents of the `logDestinationBasePath` service data property .
(see Global Configuration discussion below).

The Group
This is an unmanaged tag, which can be used in group wide operations. An attribute can only belong to one group.

Special Attributes and Parameters

There are three locations for logging service directed parameters. One is in a logging parameters properties file, called ogsilogging_parm.properties, another is service properties contained in the deployment descriptor: (core-config.wsdd). And the third is a special attribute, called the "default" attribute, optionally settable via a service interface call, primarily contained in the ogsilogging.properties file, which contains the actual logger controls.

Logging Parameters Properties File : ogsilogging_parm.properties

The contents of this file are:

1. `persistentAttributeLocation=ogsilogging.properties`
   This parameter identifies the explicit location of the ogsilogging.properties file, which contains the saved configuration data for logger controls. It can be fully qualified or relative to the current directory. In this example, the file will be fetched and modified in the current directory.
2. `logDestinationBasePath=/`
   This parameter is the vehicle by which administrators control the placement of log file output. It is pre-pended to the contents of the *destination* field in the LogAttributeElement talked about earlier. The default setting nets out to "root" which is the most unrestrictive. This value should be modified appropriately for deployed situations.

The file as shipped with the component looks like this:

```
#this file can be located any where within reach of a classpath based search
#the logdestinationbasepath will be pre-pended to all non-console destinations
#the persistentAttributeLocation absolute or relative to current directory
#    a file or URI - default as specified assumes in the current directory.
logDestinationBasePath=/
persistentAttributeLocation=ogsilogging.properties
```

Service Properties

The logging service introduces two additional parameters in the service definition area of the deployment descriptors:

1. **`<parameter name="logBufferSize" value="10"/>`**
   This is a persistent property containing the size of the log messages buffer. It will be used to set the LogBufferSize service data, which in turn will control the expansion and contraction of the log messages view.
2. **`<parameter name="logBufferSizeMax" value="1000"/>`**
   This parameter caps the size of the change in log buffer size.

Default Attribute

A LogAttributeElement whose name is “default” can be used to establish the default attributes of all currently unknown loggers. For example, if the default destination is CONSOLE and level is WARN, any log generating application that uses a logger name, not previously configured in a persistent attribute file, will inherit the default settings.

OgsiLoggingPortType: Service Data Declarations

The OgisLoggingPortType includes the following serviceData elements.

Note the following logAttributeElement definition is used within several of the service data declarations.

```
<types>
     <xsd:schema targetNamespace=”http://ogsa.globus.org/core/logging”
                    xmlns:tns=”http://ogsa.globus.org/core/logging”
                    xmlns:xsd=”http://www.w3.org/2001/XMLSchema”>
          <xsd:complexType name=”logAttributeElement”>
               <xsd:all>
                    <xsd:element name=”attributeName” type=”xsd:String”/>
                    <xsd:element name=”destination” type=”xsd:String”/>
                    <xsd:element name=”level” type=”xsd:String”/>
                    <xsd:element name=”groupName” type=”xsd:String”
                                 minOccurs=’0’/>
               </xsd:all>
          </xsd:complexType>
     </xsd:schema>
</types>
```

- TransientAttributes

  The currently active collection of transient (runtime) attributes.

```
<sd: serviceData name=”TransientAttributes”
     type=”tns:logAttributeElement”
     minOccurs=”1” maxOccurs=”unbounded”
     mutability=”mutable”
     modifiable=”false”
     nillable=”false” />
```

- PersistentAttributes

  The currently persisted collection of logging attributes.

```
<sd: serviceData name=”PersistentAttributes”
     type=”logAttributeElement”
     minOccurs=”1” maxOccurs=”unbounded”
     mutability=”mutable”
     modifiable=”false”
     nillable=”false” />
```

- LogBufferSize

  The size of the log message buffer exposed as service data.

```
<sd: serviceData name="LogBufferSize"
     type="xsd:int"
     minOccurs="1" maxOccurs="1"
     mutability="mutable"
     modifiable="false"
     nillable="false" />
```

- LogMessages

  The buffer of log messages of size LogBufferSize.

```
<types>
     <xsd:schema targetNamespace="http://ogsa.globus.org/base/logging"
                 xmlns:tns="http://ogsa.globus.org/base/logging"
                  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
          <xsd:complexType name="logMessageElement">
               <xsd:all>
                    <xsd:element name="level" type="xsd:String"/>
                    <xsd:element name="message" type="xsd:String"/>
               </xsd:all>
          </xsd:complexType>
     </xsd:schema>
</types>

<sd: serviceData name="LogMessages"
     type="tns:logMessageElement"
     minOccurs="1" maxOccurs="unbounded"
     mutability="mutable"
     modifiable="false"
     nillable="false" />
```

OGSALoggingPortType: Operations

OGSALoggingPortType::setTransientAttribute

Sets a transient log attribute. This also allows for an optional group name or tag to be specified to allow multiple loggers to be grouped together. If the attribute is a pre-existing transient attribute, it will be updated with the specified destination, level and group tag. This change takes effect immediately, however will be lost when the hosting environment is shutdown.

Input:

- *Attribute:* A log attribute Element.

Output:

- *None*

Fault(s):

- *InvalidDestinationFaultElement:* The attribute destination is not valid.
- *InvalidLevelFaultElement:* The level is not valid.
- *RemoteException:* The service has been disabled.

OGSALoggingPortType::setPersistentAttribute

Sets a persistent log attribute. This also allows for an optional group name or tag to be specified to allow multiple loggers to be grouped together. This change does not take effect until the next time the hosting environment is started, or explicitly loaded.

Input:

- *Attribute:* A log attribute.

Output:

- *None*

Fault(s):

- *InvalidDestinationFaultElement:* The attribute destination is not valid.
- *InvalidLevelFaultElement:* The level is not valid.
- *RemoteException:* The service has been disabled.
- *UnableToPersistFaultElement:* An error occurred attempting to save the attribute.

OGSALoggingPortType::removeTransientAttribute

Removes a transient log attribute. This change takes effect immediately, however will be lost when the hosting environment is shutdown. The effect of a removal is; the named logger will come under the "default attribute" umbrella (see **Default Attribute** discussion).

Input:

- *Attribute Name:* The name of a transient log attribute.

Output:

- *None*

Fault(s):

- *UnknownNameFault:* The attribute name is not a currently defined transient attribute.

OGSALoggingPortType::removePersistentAttribute

Removes a persistent log attribute from the attribute configuration file (specified by the "location" attribute discussed above).

Input:

- *Attribute Name:* The name of a persistent log attribute.

Output:

- *None*

Fault(s):

- *UnknownNameFaultElement:* The attribute name is not a currently defined persistent attribute.

- *UnableToPersistFaultElement:* An error occurred attempting to save the specified change.

Group Log Attribute

```
<types>
     <xsd:schema targetNamespace="http://ogsa.globus.org/core/logging"
                 xmlns:tns="http://ogsa.globus.org/core/logging"
                 xmlns:xsd="http://www.w3.org/2001/XMLSchema">
          <xsd:complexType name="GroupLogAttribute">
               <xsd:sequence>
                    <xsd:element name="group" type="xsd:String"/>
                    <xsd:element name="destination" type="xsd:String"/>
                    <xsd:element name="level" type="xsd:String"/>
               </xsd:sequence>
          </xsd:complexType>
     </xsd:schema>
</types>
```

The GroupLogAttribute is a complex type used on group operations. It has the following special semantics:

1. Group name can be a legitimate group tag, or an "*", which signifies an all groups scope.
2. The destination element is similar to the LogAttributeElement definition, or can be "*", which would indicate that this field is to be left as-is. A value other than "*" would change all destination elements for all attributes identified by the target group.
3. The level element, likewise is similar to the valid values of a LogAttributeElement type, with the optional "*" similarly indicating leave this field as is.

OGSALoggingPortType::changeTransientGroupAttributes

Changes a group of transient (in-memory) log attributes. This change takes effect immediately, however will be lost when the hosting environment is shutdown.

Input:

- *GroupLogAttribute:* see above

Output:

- *None*

Fault(s):

- *UnknownGroupNameFault:* The group name (tag) specified is not a known group.
- *InvalidDestinationFaultElement:* The attribute destination is not valid.
- *InvalidLevelFaultElement:* The attribute level is not valid.

OGSALoggingPortType::changePersistentGroupAttributes

Changes a group of persistent log attributes. This change does not take effect until the next time the hosting environment is started, or the configuration is loaded.

Input:
- *GroupLogAttribute:* see above

Output:
- *None*

Fault(s):
- *UnknownGroupNameFault:* The group name (tag) specified is not a known group.
- *InvalidDestinationFaultElement:* The attribute destination is not valid.
- *InvalidLevelFaultElement:* The attribute level is not valid.
- *UnableToPersistFaultElement:* An error occurred attempting to save the attribute.

OGSALoggingPortType::saveTransientAttributes

Save the current set of transient log attributes

Input:
- *None*

Output:
- *None*

Fault(s):
- *UnableToPersistFaultElement:* An error occurred attempting to save the attribute.

OGSALoggingPortType::loadPersistentAttributes

Load the current set of persistent log attributes as the transient set. This change will affect loggers that have yet to be created, or update runtime loggers already allocated.

Input:
- *None*

Output:
- *None*

Fault(s):
- *None*

Log Message Buffer

A buffer is maintained of a designated size, of the most recently issued log messages. This buffer is exposed as service data and further is made notifyable. Basically what this means is, clients or users of the OgsiLoggingPortType, can subscribe to changes in this logging window, or drill down via xpath subscriptions to receive messages of interest.

OGSALoggingPortType::setLogBufferSize
Sets the size of the LogMessages buffer.

Input:
- *Size:* The size (number of messages) for the log message buffer exposed as service data.

Output:
- *None*

Fault(s):
- *None*

Logging Service, Elements of Implementation

Log generating applications will use Jakarta commons logging APIs:

Log Making Appl Code Snippet

```
import org.apache.commons.logging.Log;
import org.apache.commons.logging.LogFactory;

static Log logger = LogFactory.getLog(MyApplName.class.getName());
```

….
logger.debug("this is a debug message");
logger.info("this is an info message");
logger.fatal("this is a fatal message, plus an exception",Exception e);

Install OGSA Log Factory

The OGSALogFactory module will serve as the commons LogFactory by
Placing a **commons-logging.properties** file in a classpath reachable location, containing the following line:

org.apache.commons.logging.LogFactory=org.globus.ogsa.impl.core.logging.OGSALogFactory

Underlying log Engine

Log4j is required with the current level of service. The log4j-core.jar file should be in the classpath. Any application using log4j direct, instead of commons APIs, will NOT be supported from a management standpoint. Any discovered log4j.properties file will be in effect for direct users. Commons users will have OGSA management prevail. So for example, if a

category logger is defined in log4j as DEBUG, but is defaulted by an ogsilogging.properties as WARN, then WARN is the designated level of service. Enhanced integration in this area could be targeted as a future requirement.

Log Managing Appl Code Snippet

```
// GSH details
     private static final String HOST_NAME = "localhost";
     private static final String PORT = "8080";
     private static final String INSTANCE = "ogsa";
     private static final String CONTEXT = "services";
     private static final String LOGGING_SERVICE =
            "core/logging/OgsiLoggingService";


private OgsiLoggingPortType logFactory = null;

OgsiLoggingServiceLocator loggingService =
                  new OgsiLoggingServiceLocator();

try {       // Build up the GSH for the Admin Service
     String loggingServiceEndPt =
                          "http://"
                                + HOST_NAME
                                + ":"
                                + PORT
                                + "/"
                                + INSTANCE
                                + "/"
                                + CONTEXT
                                + "/"
                                + LOGGING_SERVICE;
     String loggingServiceGSHString = loggingServiceEndPt + "?wsdl";
      logFactory = loggingService.getOgsiLoggingPort(new
                   URL(loggingServiceGSHString));
} catch (Exception e) {
            System.out.println("Failed to get Service: " + e.getMessage());
            e.printStackTrace();
}



try {
     LogAttributeElement parm=logA("componentLoggerName",
                          "console",
                          "warn",
                          "Itsgroup");

     logFactory.setTransientAttribute(parm);
     parm.setAttributeName("componentNextName");
     logFactory.setTransientAttribute(parm);
     //save what we have as an example
     logFactory.setTransientAttribute(logA("Location",
                          "c:\\snapshot.txt",
                          "",
                          ""));
     System.out.println("will snapshot runtime to new Location ");
     logFactory.saveTransientAttributes();
     System.out.println("will load config ");
      logFactory.loadPersistentAttributes();
      logFactory.setLogBuffer(100);
```

```
} catch (Exception e) {
      System.out.println(" something went wrong:" + e.toString());
}
```

//helper method

```
private static LogAttributeElement logA(String n, String m, String l, String g) {
         LogAttributeElement loga= new LogAttributeElement();
         loga.setAttributeName(n);
         loga.setDestination(m);
         loga.setLevel(l);
         loga.setGroupName(g);
         return loga;
}
```

Sample **ogsilogging.properties** file, after a save operation:

```
#ogsi log properties
#Tue Apr 08 15:31:05 EDT 2003
default=console,error,unassigned
org.globus.ogsa.impl.security.authentication.SecureResponseHandler=console,warn,security
org.apache.axis.encoding.ser.JAFDataHandlerDeserializerFactory=console,warn,axis
org.globus.ogsa.handlers.RoutingSecRequestHandler=console,warn,security
org.globus.ogsa.impl.ogsi.ServiceGroupProvider=console,warn,serviceGroup
org.globus.ogsa.impl.security.authentication.X509SignHandler=console,warn,security
org.apache.axis.encoding.SerializationContextImpl=console,warn,axis
org.globus.ogsa.impl.ogsi.HandleResolverProvider=console,warn
org.globus.ogsa.impl.ogsi.FactoryProvider=console,warn
org.globus.ogsa.config.ContainerConfig=console,warn
org.apache.axis.providers.java.JavaProvider=console,warn,axis
org.globus.ogsa.impl.security.authentication.WSSecurityRequestHandler=console,warn,security
org.globus.ogsa.impl.security.authentication.SecContextDelegationSkeleton=console,warn,security
org.apache.axis.SimpleChain=console,warn,axis
org.apache.axis.deployment.wsdd.WSDDProvider=console,warn,axis
org.globus.ogsa.ServiceGroupEntryGenerator=console,warn,serviceGroup
org.globus.ogsa.proxy.ReferenceRewriter=console,warn
org.apache.axis.utils.SessionUtils=console,warn,axis
org.apache.axis.description.ServiceDesc=console,warn,axis
org.globus.ogsa.server.ServiceThread.performance.process=console,warn
org.apache.axis.encoding.DeserializerImpl=console,warn,axis
org.apache.axis.utils.BeanPropertyDescriptor=console,warn
org.apache.axis.configuration.FileProvider=console,warn,axis
org.apache.axis.message.BodyBuilder=console,warn,axis
org.apache.axis.utils.XMLUtils=console,warn,axis
org.apache.axis.encoding.ser.JAFDataHandlerDeserializer=console,warn,axis
org.apache.axis.configuration.EngineConfigurationFactoryFinder=console,warn,axis
org.apache.axis.message.SOAPEnvelope=console,debug,axis
org.globus.ogsa.impl.core.factory.ServiceSweeperTask=console,warn,serviceGroup
org.apache.axis.deployment.wsdd.WSDDDeployment=console,debug,axis
org.apache.axis.encoding.ser.ElementDeserializer=console,warn,axis
org.apache.axis.AxisProperties=console,warn,axis
org.globus.ogsa.impl.security.authentication.WSSecurityResponseHandler=console,warn
org.apache.axis.SimpleTargetedChain=console,warn,axis
org.apache.axis.message.MessageElement=console,warn,axis
org.apache.axis.encoding.ser.MapDeserializer=console,warn,axis
org.apache.axis.Message=console,warn,axis
org.globus.ogsa.impl.core.service.QueryEngineImpl=console,warn,gridService
org.apache.axis.message.SOAPBodyElement=console,warn,axis
org.apache.axis.encoding.MethodTarget=console,warn,axis
org.apache.axis.handlers.http.URLMapper=console,warn,axis
org.apache.axis.enterprise=console,warn,axis
org.apache.axis.encoding.ser.OctetStreamDataHandlerDeserializer=console,warn,axis
org.globus.ogsa.server.ServiceContainer=console,warn
org.apache.axis.encoding.ser.BeanDeserializer=console,warn,axis
org.globus.ogsa.impl.core.handle.HandleHelper=console,warn
```

```
org.apache.axis.encoding.ser.PlainTextDataHandlerDeserializer=console,warn
org.globus.ogsa.handlers.RoutingRequestHandler=console,warn
org.apache.axis.deployment.wsdd.WSDDDocument=console,warn,axis
org.apache.axis.encoding.ser.MimeMultipartDataHandlerSerializer=console,warn,axis
org.globus.ogsa.impl.ogsi.GridServiceImpl=console,warn,gridService
org.apache.axis.message.RPCHandler=console,warn,axis
org.apache.axis.transport.http.HTTPSender=console,warn,axis
org.apache.axis.utils.XMLUtils$ParserErrorHandler=console,warn,axis
org.apache.axis.message.SAXOutputter=console,warn,axis
org.globus.ogsa.impl.security.authentication.SecureRequestHandler=console,warn,security
org.apache.axis.AxisEngine=console,warn,axis
org.apache.axis.AxisFault=console,warn,axis
org.apache.axis.components.net.SocketFactoryFactory=console,warn,axis
org.apache.axis.utils.JavaUtils=console,warn,axis
org.globus.ogsa.server.ServiceDispatcher=console,warn,gridService
org.apache.axis.utils.NSStack=console,warn,axis
org.apache.axis.attachments.AttachmentsImpl=console,warn,axis
org.apache.axis.encoding.ser.JAFDataHandlerSerializer=console,warn
org.apache.axis.encoding.ser.EnumSerializer=console,warn,axis
org.apache.axis.handlers.soap.SOAPService=console,warn,axis
org.globus.ogsa.handlers.OnceInvocationHandler=console,warn
org.apache.axis.TIME=console,warn,axis
org.globus.ogsa.impl.security.authentication.SecureConversationProvider=console,warn,security
org.apache.axis.message.RPCParam=console,warn,axis
org.globus.ogsa.repository.ServiceNode=console,warn
org.globus.ogsa.deployment.ServiceDeployment.performance=console,warn
org.apache.axis.client.Call=console,warn,axis
org.globus.ogsa.impl.ogsi.NotificationSourceProvider=console,warn,notification
org.globus.ogsa.impl.core.service.ServiceDataNameSetEvaluator=console,warn,serviceData
org.apache.axis.providers.BasicProvider=console,warn,axis
org.apache.axis.encoding.ser.PlainTextDataHandlerSerializer=console,warn,axis
org.apache.axis.encoding.ser.BeanSerializer=console,warn,axis
org.apache.axis.encoding.ser.VectorSerializer=console,warn,axis
org.globus.ogsa.WSDLGenerator=console,warn
org.apache.axis.encoding.DeserializationContextImpl=console,warn,axis
org.apache.axis.client.AxisClient=console,warn,axis
org.apache.axis.description.OperationDesc=console,warn,axis
org.apache.axis.encoding.ser.SourceDataHandlerDeserializer=console,warn
org.apache.axis.deployment.wsdd.WSDDDeployableItem=console,warn
org.apache.axis.encoding.TypeMappingImpl=console,warn,axis
org.apache.axis.encoding.ser.OctetStreamDataHandlerSerializer=console,warn,axis
org.apache.axis.encoding.ser.ArrayDeserializer=console,warn,axis
org.apache.axis.SOAPPart=console,warn,axis
org.globus.ogsa.utils.XmlFactory=console,warn
org.globus.ogsa.proxy.RedirectProvider.performance=console,warn
org.globus.ogsa.utils.AnyHelper=console,warn
org.apache.axis.encoding.ser.MimeMultipartDataHandlerDeserializer=console,warn
org.apache.axis.handlers.BasicHandler=console,warn,axis
org.globus.ogsa.server.ServiceThreadPool=console,warn
org.globus.ogsa.impl.core.logging.OgsiLogging=console,debug
org.globus.ogsa.repository.DefaultServiceActivator=console,warn
org.apache.axis.encoding.ser.SourceDataHandlerSerializer=console,warn,axis
org.globus.ogsa.wsdl.CachingWSDLReader=console,warn
org.apache.axis.server.AxisServer=console,warn,axis
org.globus.ogsa.wsdl.GSR=console,warn,gridService
org.apache.axis.utils.bytecode.ParamNameExtractor=console,warn,axis
org.apache.axis.handlers.JWSHandler=console,warn,axis
org.globus.ogsa.server.ServiceRequestQueue=console,warn,gridService
org.globus.ogsa.deployment.ServiceDeployment=console,warn,gridService
org.globus.ogsa.proxy.RedirectProvider=console,warn
org.globus.ogsa.impl.security.authentication.SecContextHandler=console,warn,security
org.globus.ogsa.handlers.RPCURIProvider=console,warn
org.globus.ogsa.handlers.PersistentServiceHandler=console,warn
org.apache.axis.encoding.ser.ImageDataHandlerSerializer=console,warn
org.apache.axis.encoding.ser.BeanPropertyTarget=console,warn
org.globus.ogsa.impl.ogsi.PersistentGridServiceImpl=console,warn,gridService
org.apache.axis.utils.BeanUtils=console,warn,axis
org.globus.ogsa.server.ServiceThread.performance=console,warn
org.apache.axis.i18n.ProjectResourceBundle=console,warn
org.globus.ogsa.encoding.ObjectDeserializationContext=console,warn
org.globus.ogsa.impl.security.authentication.ContextManager=console,warn,security
```

```
org.globus.ogsa.impl.security.authentication.SignHandler=console,warn,security
org.globus.ogsa.impl.ogsi.HandleResolverImpl=console,warn
org.apache.axis.encoding.ser.VectorDeserializer=console,warn,axis
org.globus.ogsa.impl.core.service.ServiceDataNameEvaluator=console,warn,serviceData
org.apache.axis.MessageContext=console,warn,axis
org.globus.ogsa.impl.core.service.ServiceDataXPathEvaluator=console,warn,serviceData
org.apache.axis.message.SOAPBody=console,warn,axis
org.globus.ogsa.impl.core.service.ServiceLocator=console,warn,gridService
org.globus.ogsa.impl.ogsi.NotificationSubscriptionImpl=console,warn,notification
org.globus.ogsa.handlers.RoutingResponseHandler=console,warn
org.globus.ogsa.impl.security.authentication.WSSecurityUtil=console,warn,security
org.globus.ogsa.impl.core.registry.ContainerRegistryImpl=console,warn,serviceGroup
org.globus.ogsa.wsdl.GSRDescription=console,warn,gridService
org.apache.axis.enum.Enum=console,warn,axis
org.apache.axis.message.SOAPHeader=console,warn,axis
org.apache.axis.encoding.ser.ImageDataHandlerDeserializer=console,warn,axis
org.globus.ogsa.wsdl.SymbolTable.performance=console,warn
org.globus.ogsa.utils.SweeperPool=console,warn
org.globus.ogsa.impl.security.authentication.WSSecurityHandler=console,warn,security
org.apache.axis.encoding.ser.MapSerializer=console,warn,axis
org.globus.ogsa.handlers.HandleResolverHandler=console,warn
org.globus.ogsa.server.ServiceThread=console,warn,gridService
org.apache.axis.encoding.ser.ArraySerializer=console,warn
org.apache.axis.configuration.EngineConfigurationFactoryServlet=console,warn,axis
org.apache.axis.providers.java.RPCProvider=console,warn
org.globus.ogsa.wsdl.SymbolTable=console,warn
org.apache.axis.configuration.EngineConfigurationFactoryDefault=console,warn,axis
org.apache.axis.components.net.DefaultSocketFactory=console,warn,axis
org.globus.ogsa.server.ServiceContainerCollection=console,warn,serviceGroup
org.globus.ogsa.impl.ogsi.SubscriptionEntry=console,warn
```

note: a **saveTransientAttributes** operation will create a file of all currently obtained loggers, be they previously configured or newly discovered during runtime. Should a lesser subset be desired, the **setPersistentAttribute** api should be used for a tailored subset. Be sure to include a write of the Default and Location attributes.

Deployment Descriptor

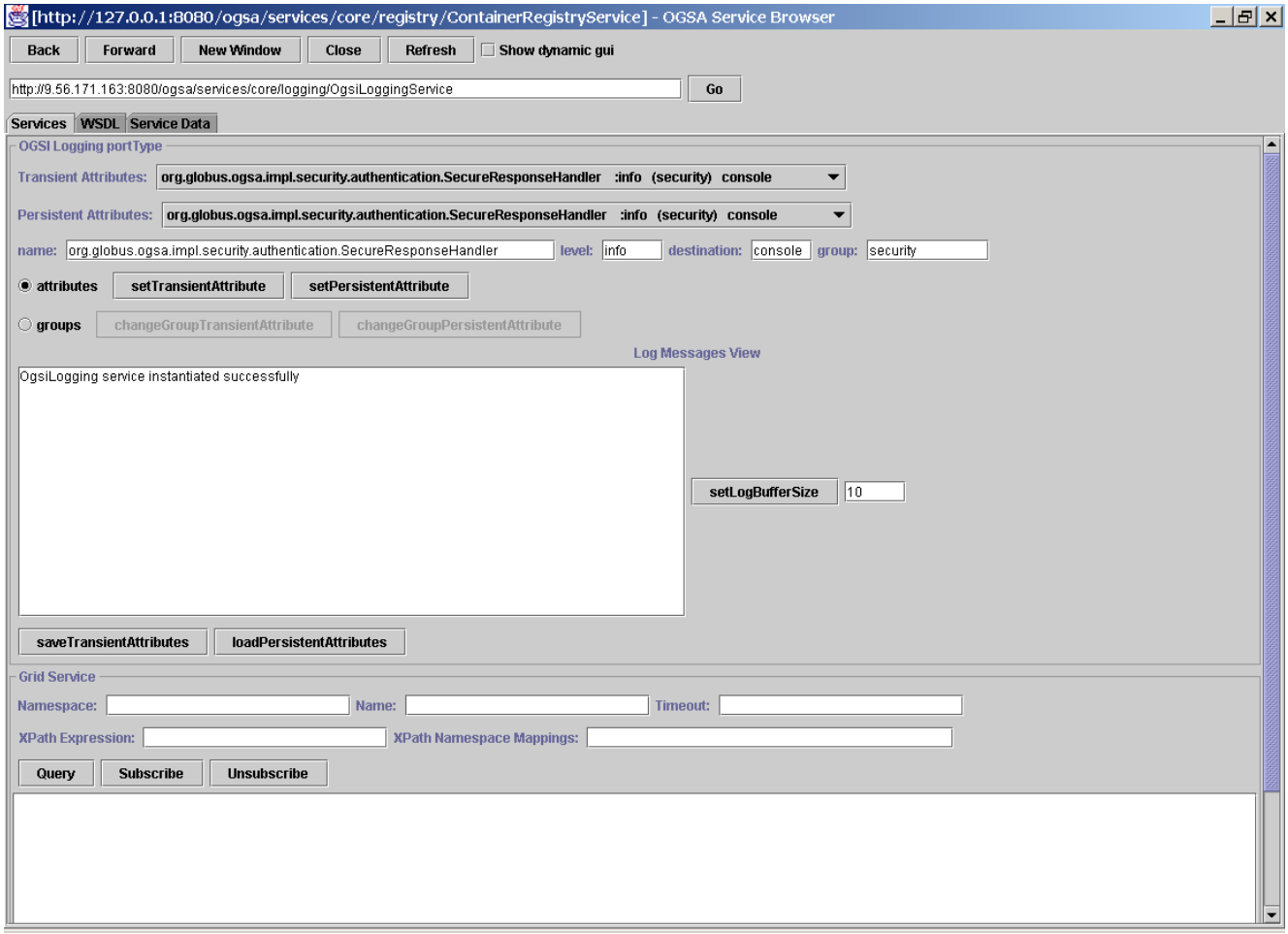
```
<service name="core/logging/OgsiLoggingService"
         provider="Handler" style="wrapped" use="literal">
  <parameter name="baseClassName"
value="org.globus.ogsa.impl.core.logging.OgsiLogging"/>
  <parameter name="allowedMethods" value="*"/>
  <parameter name="persistent" value="true"/>
  <parameter name="schemaPath"
value="schema/core/logging/logging_service.wsdl"/>
  <parameter name="handlerClass"
value="org.globus.ogsa.handlers.RPCURIProvider"/>
  <parameter name="className"
value="org.globus.ogsa.core.logging.OgsiLoggingPortType"/>
  <parameter name=”operationProviders”
value=”org.globus.ogsa.impl.ogsi.NotificationSourceProvider"/>
   <parameter name="logBufferSize" value="10"/>
  <parameter name="logBufferSizeMax" value="1000"/>
  <parameter name="lifecycle" value="persistent"/>
</service>
```

Per this deployment descriptor, this service’s GSH will be well-known and defined as follows:

```
http://fqdn:port/<OGSIInstanceName>/services/core/logging/OgsiLoggingService
```

Service Browser

For demonstration and debug purposes a panel has been added to the service browser specific to the OgsiLoggingPortType. This panel exposes the operations associated with this portType. An example of this is show below.



Log service hints and tips:

1. drop down box actions(Transient or Persistent) will update input text areas
2. you must toggle between “attribute” operations and “group” operations
3. the log view is a scrollable pane
4. the load and save buttons require proper specification of “persistentAttributeLocation” in the ogsilogging_parms.properties file. This property is to specify the location to write persistent log attributes.
5. the default ogsilogging.properties file, comes with some sample group assignments
6. a log buffer size of zero, effectively halts log message notifies, and eliminates any existing bucketed messages