Host Information Service

Purpose

In order to be able to stage jobs to resources and launch them there on behalf of a user, host-dependent information must be obtained dynamically by the Workflow Broker after scheduling the node to run. While it would be optimal if every Grid resource exposed this information consistently and through a uniform set of APIs, that is unfortunately not (yet?) the case. Hence we have developed a small information service with an administrative interface to store this data and make it readily available both for viewing and for workflow configuration. This is once again a GSI web service requiring authentication and authorization via certificate.

Mode of Usage

In most cases, the user, if administrator, will interact with this service through a specially designed Siege view (refer to the tutorial for specifics). The programmatic interactions, such as those accomplished by the broker, take place via the web service ports. The sections which follow thus are only of interest to potential developers wishing to gain understanding of what lies "under the hood". There is an example of the XML data representation at the end of this page.

API

Service WSDL

Repository

Info be obtained through its `retrieve` ports, updated or modified through its `add`, `update` and `remove` ports. In addition, the `find` calls can be used to do simple listings.

```
public interface HostInfoRepository
{
   /*
    * Add a new root element.
    */
   public void addHost( String hostInfoXml );

   /*
    *  Diff'd update of the root.
    */
   public void updateHost( String hostId, String hostInfoXml );

   /*
    *  Diff'd update of the root.
    */
   public void updateNode( String hostId, String nodeId, String nodeInfoXml );

   /*
    *  Diff'd update of the root.
    */
   public void updateUser( String hostId, String userName, String userInfoXml );

   /*
    * Remove can be called anywhere on the HostInfo tree.
    */
   public void removeHost( String hostId );
   public void removeUserFromHost( String hostId, String user );
   public void removeUserFromAllHosts( String user );
   public void removeHostEnvironmentVariable( String hostId, String name );
   public void removeUserEnvironmentVariable( String hostId, String user, String name );
   public void removeNodeFromHost( String hostId, String nodeId );
   public void removeProtocol( String nodeId, String protocolType, String protocolName );
   public void removeOptimizationProperty( String nodeId, String protocolType, String protocolName, String
propertyName );
   public void removeSoftEnv( String nodeId );

   /*
    *  List queries.
    */
   public String[] findHosts();
   public String[] findUsers( String hostId );
   public String[] findNodes( String hostId );

   /*
    *  These return the serialized object.
    */
   public String retrieveHostInfo( String hostId );
   public String retrieveNodeInfo( String nodeId );
   public String retrieveUserInfoWithHostEnv( String nodeId, String user );
   public String retrieveUserInfoEntry( String hostId, String user );
}
```

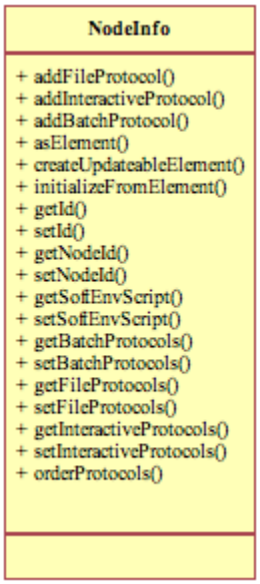
HostInfo

Defines an entire resource. Aside from a generic name for the entire resource tree, architecture and OS, a host can have a set of environment properties applicable to all users, a set of user accounts, and a set of node names which can define their own protocols for job submission and file movement.

```
public class HostInfo implements Updateable
{
   /*
    *  Updateable
    */
   public Element asElement();
   public Element createUpdateableElement();
   public void initializeFromElement( Element element );

   /*
    *  Hibernate
    */
   public Integer getId();
   public void setId( Integer id );

   public void addEnvVariable( String name, String value );
   public void addUserInfo( UserOnHostInfo info );
   public void addNodeInfo( NodeInfo info );

   public String getHostId();
   public void setHostId( String hostId );
   public String getArchitecture();
   public void setArchitecture( String architecture );
   public String getOsName();
   public void setOsName( String osName );
   public String getOsVersion();
   public void setOsVersion( String osVersion );
   public Map getEnvironment();
   public void setEnvironment( Map environment );
   public Map getNodeInfo();
   public void setNodeInfo( Map nodeInfo );
   public Map getUserInfo();
   public void setUserInfo( Map userInfo );
}
```

NodeInfo

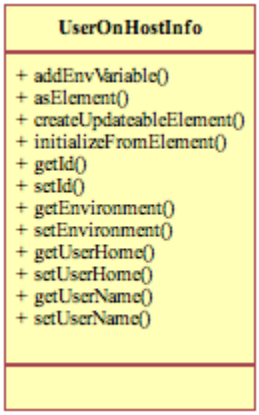
Defines a node mapping for a resource. This is usually the name of a physical "head node" with a specific IP address, but a logical name is also possible, provided the protocol contacts actually point to physical nodes. The node is thus a way for specifying independent configurations of interactive, batch and file movement protocols available on the resource.




```
public class NodeInfo implements Updateable
{
   /*
    *  Updateable
    */
   public Element asElement();
   public Element createUpdateableElement();
   public void initializeFromElement( Element element );

   /*
    *  Hibernate
    */
   public Integer getId();
   public void setId( Integer id );

   public void addFileProtocol( ProtocolInfo info );
   public void addInteractiveProtocol( ProtocolInfo info );
   public void addBatchProtocol( ProtocolInfo info );

   public String getNodeId();
   public void setNodeId( String nodeId );
   public String getSoftEnvScript();
   public void setSoftEnvScript( String softEnvScript );
   public Map getBatchProtocols();
   public void setBatchProtocols( Map batchProtocols );
   public Map getFileProtocols();
   public void setFileProtocols( Map fileProtocols );
   public Map getInteractiveProtocols();
   public void setInteractiveProtocols( Map interactiveProtocols );
   public ProtocolInfo[] orderProtocols( String type );
}
```

UserInfo

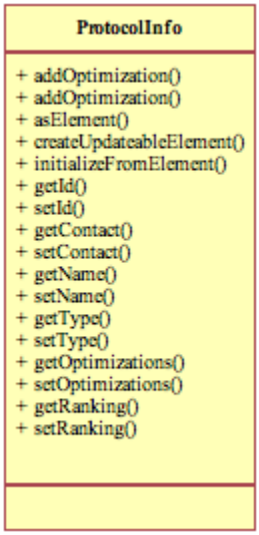
Maps user to home directory on the resource, and provides any specific environment variables needed to run the user's jobs there.




```
public class UserOnHostInfo implements Updateable
{
   /*
    *  Updateable
    */
   public Element asElement();
   public Element createUpdateableElement();
   public void initializeFromElement( Element element );

   /*
    *  Hibernate
    */
   public Integer getId();
   public void setId( Integer id );

   public void addEnvVariable( String name, String value );

   public Map getEnvironment();
   public void setEnvironment( Map environment );
   public String getUserHome();
   public void setUserHome( String userHome );
   public String getUserName();
   public void setUserName( String userName );
}
```

ProtocolInfo

Defines protocols for job submission and file movement that are available on the given resource. Associated properties for optimal performance can also be stored for configuration purposes. The ranking is an indexed position used in conjunction with the `orderProtocols()` method on the `NodeInfo` object; for instance, a ranking of '0' puts the protocol at the head of the list in front of any alternates with a higher ranking value.




```
public class ProtocolInfo implements Updateable
{
   /*
    *  Updateable
    */
   public Element asElement();
   public Element createUpdateableElement();
   public void initializeFromElement( Element element );

   /*
    *  Hibernate
    */
   public Integer getId();
   public void setId( Integer id );

   public void addOptimization( String name, String value );
   public void addOptimization( Property property );

   public String getContact();
   public void setContact( String contact );
   public String getName();
   public void setName( String name );
   public String getType();
   public void setType( String type );
   public Map getOptimizations();
   public void setOptimizations( Map optimizations );
   public Integer getRanking();
   public void setRanking( Integer ranking );
}
```

Note on Implementation

Updates to the service are achieved by diff'ing the current representation of the object against the new one. In order for this to work with a given bean, a special interface must be implemented:

```
public interface Updateable extends UserFacing
{
   public static final String ID_NAMESPACE = "ncsa.updateable.id";
   public static final String NEW_NAMESPACE = "ncsa.updateable.new";
   public static final String DELETED_NAMESPACE = "ncsa.updateable.deleted";

   /**
    * The contract is that this method should be called inside the
    * asElement() method to return the top-level element
    * with an ID_NAMESPACE whose value is equal to whatever
    * distinguishes this object from any siblings of the same type.
    */
   public Element createUpdateableElement();
}
```

As can be seen, special XML namespaces for adding and deleting are used in the diff'ing process. The code responsible for merging the XML is in the `UserFacingUtils` class of the `ncsa.tools.common` plugin. Siege's **`HostInfoAdmin`** view handles the namespace semantics required by those methods.

`HostInfo` XML Example

```
<?xml version="1.0" encoding="UTF-8"?>
<host xmlns:ncsa.updateable.id="TUNGSTEN" hostId="TUNGSTEN" architecture="i686 SMP" osName="Linux" osVersion="
2.4.21-32.0.1.ELsmp-perfctr-lustre">

  <!-- common environment -->
  <property xmlns:ncsa.updateable.id="JAVA_HOME" name="JAVA_HOME" category="environment">
    <value xmlns:ncsa.updateable.id="value">/u/ncsa/jalameda/j2sdk1.4.2_04</value>
  </property>
  <property xmlns:ncsa.updateable.id="SCR" name="SCR" type="int" category="workdirEnvironment">
    <value xmlns:ncsa.updateable.id="value">0</value>
  </property>
  <property xmlns:ncsa.updateable.id="TG_CLUSTER_SCRATCH" name="TG_CLUSTER_SCRATCH" type="int" category="
workdirEnvironment">
    <value xmlns:ncsa.updateable.id="value">1</value>
  </property>
  <property xmlns:ncsa.updateable.id="TG_CLUSTER_PFS" name="TG_CLUSTER_PFS" type="int" category="
workdirEnvironment">
    <value xmlns:ncsa.updateable.id="value">2</value>
  </property>
  <property xmlns:ncsa.updateable.id="scr" name="scr" type="int" category="workdirEnvironment">
    <value xmlns:ncsa.updateable.id="value">3</value>
  </property>
  <property xmlns:ncsa.updateable.id="PWD" name="PWD" type="int" category="workdirEnvironment">
    <value xmlns:ncsa.updateable.id="value">4</value>
  </property>

  <!-- users -->
  <user xmlns:ncsa.updateable.id="arossi" userName="arossi" userHome="/u/ncsa/arossi"/>
  <user xmlns:ncsa.updateable.id="bbaker" userName="bbaker" userHome="/u/ncsa/bbaker"/>
  <user xmlns:ncsa.updateable.id="daues" userName="daues" userHome="/u/ncsa/daues"/>
  <user xmlns:ncsa.updateable.id="shawn" userName="shawn" userHome="/u/ncsa/shawn"/>
  <user xmlns:ncsa.updateable.id="jalameda" userName="jalameda" userHome="/u/ncsa/jalameda"/>
  <user xmlns:ncsa.updateable.id="bjewett" userName="bjewett" userHome="/u/ncsa/bjewett"/>
  <user xmlns:ncsa.updateable.id="scottp" userName="scottp" userHome="/u/ncsa/scottp"/>
  <user xmlns:ncsa.updateable.id="tbaltzer" userName="tbaltzer" userHome="/u/ac/tbaltzer"/>
  <user xmlns:ncsa.updateable.id="kathla" userName="kathla" userHome="/u/ac/kathla"/>
  <user xmlns:ncsa.updateable.id="danielw" userName="danielw" userHome="/u/ac/danielw"/>
  <user xmlns:ncsa.updateable.id="govett" userName="govett" userHome="/u/ac/govett"/>
  <user xmlns:ncsa.updateable.id="akulkar3" userName="akulkar3" userHome="/u/ac/akulkar3"/>

  <!-- head nodes -->
  <node xmlns:ncsa.updateable.id="tunc.ncsa.uiuc.edu" nodeId="tunc.ncsa.uiuc.edu">
    <file-protocol xmlns:ncsa.updateable.id="gridftp" name="gridftp" ranking="0">
      <contact xmlns:ncsa.updateable.id="contact">gridftp://gridftp-w.ncsa.teragrid.org:2811</contact>
    </file-protocol>
```

```
    <file-protocol xmlns:ncsa.updateable.id="gsiscp" name="gsiscp" ranking="1">
      <contact xmlns:ncsa.updateable.id="contact">gsiscp://grid-w.ncsa.teragrid.org</contact>
    </file-protocol>
    <interactive-protocol xmlns:ncsa.updateable.id="GRAM" name="GRAM" ranking="0">
      <contact xmlns:ncsa.updateable.id="contact">jobmanager://grid-w.ncsa.teragrid.org:2119</contact>
    </interactive-protocol>
    <interactive-protocol xmlns:ncsa.updateable.id="GSISSH" name="GSISSH" ranking="1">
      <contact xmlns:ncsa.updateable.id="contact">gsissh://grid-w.ncsa.teragrid.org</contact>
    </interactive-protocol>
    <batch-protocol xmlns:ncsa.updateable.id="MOAB" name="MOAB" ranking="0">
      <contact xmlns:ncsa.updateable.id="contact">msub://localhost</contact>
    </batch-protocol>
  </node>
  <node xmlns:ncsa.updateable.id="tund.ncsa.uiuc.edu" nodeId="tund.ncsa.uiuc.edu">
    <file-protocol xmlns:ncsa.updateable.id="gridftp" name="gridftp" ranking="0">
      <contact xmlns:ncsa.updateable.id="contact">gridftp://gridftp-w.ncsa.teragrid.org:2811</contact>
    </file-protocol>
    <file-protocol xmlns:ncsa.updateable.id="gsiscp" name="gsiscp" ranking="1">
      <contact xmlns:ncsa.updateable.id="contact">gsiscp://grid-w.ncsa.teragrid.org</contact>
    </file-protocol>
    <interactive-protocol xmlns:ncsa.updateable.id="GRAM" name="GRAM" ranking="0">
      <contact xmlns:ncsa.updateable.id="contact">jobmanager://grid-w.ncsa.teragrid.org:2119</contact>
    </interactive-protocol>
    <interactive-protocol xmlns:ncsa.updateable.id="GSISSH" name="GSISSH" ranking="1">
      <contact xmlns:ncsa.updateable.id="contact">gsissh://grid-w.ncsa.teragrid.org</contact>
    </interactive-protocol>
    <batch-protocol xmlns:ncsa.updateable.id="GRAM" name="GRAM" ranking="0">
      <contact xmlns:ncsa.updateable.id="contact">jobmanager-lsf://grid-w.ncsa.teragrid.org:2119</contact>
    </batch-protocol>
  </node>
  <node xmlns:ncsa.updateable.id="grid-w.ncsa.teragrid.org" nodeId="grid-w.ncsa.teragrid.org">
    <file-protocol xmlns:ncsa.updateable.id="gridftp" name="gridftp" ranking="0">
      <contact xmlns:ncsa.updateable.id="contact">gridftp://gridftp-w.ncsa.teragrid.org:2811</contact>
    </file-protocol>
    <file-protocol xmlns:ncsa.updateable.id="gsiscp" name="gsiscp" ranking="1">
      <contact xmlns:ncsa.updateable.id="contact">gsiscp://grid-w.ncsa.teragrid.org</contact>
    </file-protocol>
    <interactive-protocol xmlns:ncsa.updateable.id="GRAM" name="GRAM" ranking="0">
      <contact xmlns:ncsa.updateable.id="contact">jobmanager://grid-w.ncsa.teragrid.org:2119</contact>
    </interactive-protocol>
    <interactive-protocol xmlns:ncsa.updateable.id="GSISSH" name="GSISSH" ranking="1">
      <contact xmlns:ncsa.updateable.id="contact">gsissh://grid-w.ncsa.teragrid.org</contact>
    </interactive-protocol>
    <batch-protocol xmlns:ncsa.updateable.id="GRAM" name="GRAM" ranking="0">
      <contact xmlns:ncsa.updateable.id="contact">jobmanager-lsf://grid-w.ncsa.teragrid.org:2119</contact>
    </batch-protocol>
   </node>
</host>
```