1.0 Overview

The Metadata Registry Search web service provides the capability to search the Metadata Registry via the Simple Object Access Protocol (SOAP).

A capability to search the Metadata Registry currently exists on the Department of Defense (DoD) Metadata Registry website. The search is performed via Representational State Transfer (REST), and the capability is available only through the website. The Metadata Registry Search web service makes the search capability available to external client applications. SOAP was chosen as the communication protocol because it is based on Extensible Markup Language (XML), platform and language independent, and provides a way to communicate between disparate applications over Hypertext Transport Protocol (HTTP).

1.1 Who should use the web service?

The Metadata Registry Search web service is available to any client application with a need to search the DoD Metadata Registry. A client application is required to provide a valid username and password in order to perform a search. A username and password may be obtained by registering with the DoD Metadata Registry website at http://xml.dod.mil/mdreg/user/registration/register.cfm.

2.0 Requirements

The Metadata Registry Search web service satisfies the following requirements:

- The Metadata Registry Search web service shall be SOAP-based.
- The Metadata Registry Search web service shall comply with Web Services Interoperability Organization (WS-I) standards.
- The Metadata Registry Search web service shall provide the same functionality and results as the existing REST-based search.

In order to satisfy the requirement to provide the same functionality as the existing REST-based search, the following document was used as a guide: "REST-Based Search & Retrieval Web Service 1.0." A copy may be obtained from the DoD Metadata Registry website at http://xml.dod.mil/xmlreg/user/briefings.cfm. Login is required to access the document.

3.0 Detailed Description

The Metadata Registry Search web service provides several classes and methods. Each is described in the following sections.

3.1 SearchCriteria class

The `SearchCriteria` class must be used to specify the criteria for a search. The following table describes the criteria values. A client application must fill the required fields and may fill any or all of

the optional fields. Obviously, filling in many of the fields will narrow the search while filling in only a few will result in a broad search.

Name	Type	Required?	Description	Default Value
userId	String	yes	The user id that you would use to login onto the Metadata Registry - ***http://xml.dod.mil/xmlreq/user/index.cfm*** This is normally your email address.	N/A
password	String	yes	The password for the supplied user id.	N/A
phrase	String	no	Search phrase	All
caseSensitive	boolean	no	Use 'true' if the supplied phrase should be a case-sensitive search, 'false' otherwise.	false
searchFields	String	no	Limit the search to this field	All
namespace	String	no	The namespace to search within. Must contain the id of the namespace, not the name or abbreviation.	All
type	String	no	Limit the search to this type of item. Must contain the id of the type, not the type name.	All
status	String	no	Limit the search to items with a particular status. Must contain the id of the status, not the status name.	All
versionNumber	String	no	Version number of the item to be retrieved.	All
submitter	String	no	Limit the search to items submitted by a particular person. Must contain the id of the person, not the person's name.	All

Name	Type	Required?	Description	Default Value
startDate	String	no	Limit the search to items with an effective date greater than or equal to the supplied date. The date format should be "DD-MON-YYYY"; for example July 17, 2002 should be sent as "17-JUL-2002". The correct abbreviations for 'MON' are: • JAN • FEB • MAR • APR • MAY • JUN • JUL • AUG • SEP • OCT • NOV • DEC	All
endDate	String	no	Limit the search to IR's with an effective date less than or equal to the supplied date. Same format as startDate should be used.	All

The `SearchCriteria` class provides the following getter/setter methods.

- `public void setUserId(String s)`
- `public String getUserId()`

- `public void setPassword(String p)`
- `public String getPassword()`

- `public void setPhrase(String p)`
- `public String getPhrase()`

- `public void setCaseSensitive(boolean cs)`
- `public boolean getCaseSensitive()`

- `public void setSearchFields(String s)`
- `public String getSearchFields()`

- `public void setNamespace(String ns)`
- `public String getNamespace()`

- `public void setType(String t)`

- `public String getType()`

- `public void setStatus(String s)`
- `public String getStatus()`

- `public void setVersionNumber(String vn)`
- `public String getVersionNumber()`

- `public void setSubmitter(String s)`
- `public String getSubmitter()`

- `public void setStartDate(String sd)`
- `public String getStartDate()`

- `public void setEndDate(String ed)`
- `public String getEndDate()`

3.2 SearchResults class

The results of a search operation are provided to the client application in an array of `SearchResults`. Each result item contains the following values.

Name	Type	Description
name	String	Item name
namespaceName	String	Namespace the item belongs to
namespaceCode	String	??
versionNumber	String	Version number of the item
context	String	??
type	String	Item type
description	String	Item description
comment	String	Comment associated with the item
effectiveDate	String	Effective date of the item. Format will be "DD-MON-YYYY"; for example 17-JUL-2002.
status	String	Item status
securityClassification	String	Item security classification
submitter	String	Name of person who submitted the item.
url	String	A URL to retrieve this item.

The `SearchResults` class provides the following getter/setter methods.

- `public void setName(String n)`
- `public String getName()`

- `public void setNamespaceName(String nn)`
- `public String getNamespaceName()`

- `public void setNamespaceCode(String nc)`
- `public String getNamespaceCode()`

- `public void setVersionNumber(String vn)`
- `public String getVersionNumber()`

- `public void setContext(String c)`
- `public String getContext()`

- `public void setType(String t)`
- `public String getType()`

- `public void setDescription(String d)`
- `public String getDescription()`

- `public void setComment(String c)`
- `public String getComment()`

- `public void setEffectiveDate(String ed)`
- `public String getEffectiveDate()`

- `public void setStatus(String s)`
- `public String getStatus()`

- `public void setSecurityClassification(String sc)`
- `public String getSecurityClassification()`

- `public void setSubmitter(String s)`
- `public String getSubmitter()`

- `public void setUrl(String u)`
- `public String getUrl()`

3.3 SearchService_Impl and SearchServicePort classes

The `SearchService_Impl` and `SearchServicePort` classes together allow a client application to invoke the Metadata Registry Search web service. The `SearchService_Impl` class is a stub factory that implements the Java API for XML-based Remote Procedure Calls (JAX-RPC) `Service` interface. The `SearchServicePort` class is a stub, created via a call to the stub factory.

The constructor of `SearchService_Impl` creates a stub based on the parameter, the location of the Search service WSDL file. The `getSearchServicePort()` method is used to return an instance of the `SearchServicePort` stub implementation.

The following code shows how to create the `SearchServicePort` stub:

```
SearchService_Impl ws = new SearchService_Impl(wsdlURL);
SearchServicePort port = ws.getSearchServicePort();
```

Once the stub is created, the client application may invoke the Metadata Registry Search web service methods. The following code shows a call to the `search()` method:

```
SearchResults[] results = port.search(criteria);
```

See the sample code in Appendix A for a complete example client application. The following section describes the `search()` method in detail.

3.3.1 search() method

The `search()` method is called by a client application to perform a search of the Metadata Registry. The only parameter of the method is `SearchCriteria`, a set of values to bound or narrow the search. The method returns an array of `SearchResults`. The array will be empty if no results are found. The array will be `null` if the search could not be performed for some reason. The method signature is provided below.

```
public SearchResults[] search(SearchCriteria criteria)
```

4.0 System requirements

- BEA WebLogic Server 8.1 is required for deployment of the web service.
- Java Runtime Environment, version 1.4, is required on the WebLogic server machine, and on client application machines.
- Network connectivity between the WebLogic server machine and the Metadata Registry database server is required.
- Network connectivity between the WebLogic server and client application machines is required.

5.0 Installation / Configuration

The Metadata Registry Search web service delivery package contains two sets of installation/configuration instructions.

To deploy and configure the service on the WebLogic server, reference:
    README-ServerPackage.txt

To install the client library and run the test client, reference:
    README-ClientPackage.txt

6.0 Additional Information

For additional information about the Metadata Registry Search web service, please reference the javadoc files or the installation instructions included with the delivery package.

Appendix A – Sample Client Code

The following code is a sample class called `SearchClient.java`. It provides an example of how the Metadata Registry Search web service might be used by a client application. The location of the Metadata Registry Search web service WSDL file is required to be the first command line argument. It should be something similar to:

http://localhost:7001/searchservice/SearchService?WSDL

```
public class SearchClient
{
   public static void main(String[] args)
   {
      // Setup the global JAXM message factory
      System.setProperty("javax.xml.soap.MessageFactory",
                "weblogic.webservice.core.soap.MessageFactoryImpl");

      // Setup the global JAX-RPC service factory
      System.setProperty( "javax.xml.rpc.ServiceFactory",
                "weblogic.webservice.core.rpc.ServiceFactoryImpl");

     /* Many of the criteria 'set' calls are commented out below.  They
        are provided to serve as examples of setting each field. */
      try
      {
         SearchService_Impl ws = new SearchService_Impl(args[0]);
         SearchServicePort port = ws.getSearchServicePort();
         SearchCriteria criteria = new SearchCriteria();
         criteria.setUserId("jill");
         criteria.setPassword("password");
//       criteria.setPhrase("search for metadata registry records");
//       criteria.setCaseSensitive(true);
         criteria.setSearchFields("All");
//       criteria.setNamespace("105"); //ACQ
         criteria.setType("12");  //XSL Stylesheet
//       criteria.setStatus("4"); //Operational
//       criteria.setVersionNumber("6.7");
//       criteria.setSubmitter("1000000443"); //Test User
//       criteria.setStartDate("10-MAY-1960");
//       criteria.setEndDate("10-MAY-1980");
         System.out.println(criteriaToString(criteria));

         SearchResults[] results = port.search(criteria);
         if (results != null)
         {
            System.out.println("SearchClient: printing results...");
            for (int i = 0; i < results.length; i++)
            {
               if (results[i] != null)
               {
                  System.out.println("\t" +
                     resultToString(results[i]));
               }

            }//end for
```

```
            System.out.println("SearchClient: done printing results.");

         }
         else
         {
            System.out.println("SearchClient: no results returned");
         }

      }
      catch (Exception ex)
      {
         ex.printStackTrace();
      }

   }


   /* The following methods are for debugging purposes only */

   private static String criteriaToString(SearchCriteria criteria)
   {
      String returnValue = "Criteria = {" +
                                  criteria.getUserId() + ", " +
                           criteria.getPassword() + "," +
                           criteria.getPhrase() + ", " +
                           criteria.getSearchFields() + ", " +
                           criteria.getNamespace() + ", " +
                           criteria.getType() + ", " +
                           criteria.getStatus() + ", " +
                           criteria.getVersionNumber() + ", " +
                           criteria.getSubmitter() + ", " +
                           criteria.getStartDate() + ", " +
                           criteria.getEndDate() + "}";
      return returnValue;

   }

   private static String resultToString(SearchResults result)
   {
      String returnValue = "SearchResult = {" +
                                  result.getName() + ", " +
                           result.getNamespaceName() + ", " +
                           result.getNamespaceCode() + ", " +
                           result.getVersionNumber() + ", " +
                           result.getContext() + ", " +
                           result.getType() + ", " +
                           result.getDescription() + ", " +
                           result.getComment() + ", " +
                           result.getEffectiveDate() + ", " +
                           result.getStatus() + ", " +
                           result.getSecurityClassification() + ", " +
                           result.getSubmitter() + ", " +
                           result.getUrl() + "}";
      return returnValue;

   }
```

```
}//End Class SearchClient
```

Appendix B – Known Limitations

Searching Multiple Namespaces

The search operation is currently limited to either a single namespace or all namespaces (if no value is provided in the search criteria). In the REST-based version, separating namespaces with commas could search multiple namespaces. This limitation will be corrected in a future version of the web service.

URL Population

The URL is currently not populated in the results returned from a search. This limitation will be corrected in a future version of the web service.

Secure Messaging

The SOAP messages passing from a client application to the Metadata Registry Search web service are not encrypted. The `SearchCriteria` class contains a `password` field and should be secured via encryption. This limitation will be corrected in a future version of the web service.