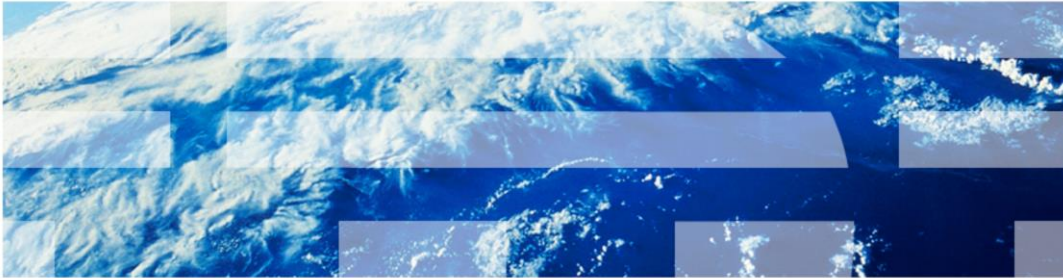


InfoSphere Master Data Management Collaboration Server V10

Leveraging web services features bundled with
Master Data Management Collaboration Server



© 2013 IBM Corporation

The topic of this presentation is how to use web services features built into IBM InfoSphere® Master Data Management Collaboration Server version 10 to integrate it with other products and form a complete solution. IBM InfoSphere Master Data Management Collaboration Server is referred to as MDMCS through out this presentation.

Terminology

- Aim
 - Web services features bundled with MDMCS
- Terminology
 - MDMCS
 - TOP
 - WSDL
 - WSDD
 - SOAP
 - Apache Axis
 - RSA
- What is not covered
 - Comprehensive details on programming or implementation



This presentation discusses web services features bundled with the product to make it more powerful and robust. There is some terminology in this presentation you need to be aware of.

TOP is a variable which points to the installation directory of the product.

WSDL stands for Web Services Description Language. It is used to describe (in XML) exactly what a web service does.

WSDD stands for Web Service Deployment Descriptor. It is an Axis specific web service deployment configuration file and can be used to specify resources that should be exposed as web services.

SOAP is an acronym for Simple Object Access Protocol. It is a transport protocol that sends XML messages using HTTP (which runs on top of TCP, typically on port 80).

Axis is an XML based, open source framework for developing web services. It is a global standard and currently MDMCS supports Axis 1.4.

Finally, RSA stands for Rational® Software architect and is the IBM implementation of Eclipse platform to develop applications and web services.

This presentation is not meant to serve as a comprehensive programming or implementation guide to help you implement web services. Rather, it is meant to familiarize you with the features of the product to make using web services more convenient and efficient.

Reference

Concept	Available information	Link
Web services	Wikipedia entry	http://en.wikipedia.org/wiki/Web_services
Web services	Latest news from W3C	http://www.w3.org/2002/ws/Activity
Apache Axis	Various beginner guides and reference documents	http://ws.apache.org/axis/java/index.html
Apache Axis	User Guide	http://ws.apache.org/axis/java/user-guide.html
Apache Axis	FAQ	http://wiki.apache.org/ws/FrontPage/Axis
Axis (De)Serializers	Reference class list	http://ws.apache.org/axis/java/apiDocs/org/apache/axis/encoding/ser/package-summary.html
WSDL2Java/Java2WSDL	Tool references	http://ws.apache.org/axis/java/reference.html
WSDO	Explanation and details	http://ws.apache.org/axis/java/user-guide.html
WSDL	Official W3C specification and guide	http://www.w3.org/TR/wsdl
SOAP	Official W3C specification and guide	http://www.w3.org/TR/soap/
java.util.Date	Information about using dates in Web services	http://www.ibm.com/developerworks/xml/library/ws-tip-roundtrip1.html

Detailed discussion of web services is beyond the scope of this presentation. See the links displayed on this slide for details.

Sample web services

- \$TOP/samples/webservices/
- Search web services
 - String search(String wql, Context wsContext)
- Product web services
 - Item createItem(Item item, Context wsContext)
 - Item updateItem(Item item, Context wsContext)
 - Item getItem(Item item, Context wsContext)
 - Void deleteItem(Item item, Context wsContext)
- Scheduler web services
 - Report createReport(Report report, Context wsContext)
 - Schedule createSchedule(Schedule schedule, Context wsContext)
 - String getScheduleStatus(Schedule schedule, Context wsContext)
 - Void stopSchedule(Schedule schedule, Context wsContext)
 - Job getJob(Job job, Context wsContext)
 - Collection<job> getAllJobs(Context wsContext)
 - Collection<Schedule> getAllSchedules(Job job, Context wsContext)

The \$TOP/samples/webservices folder contains a list of web services which are bundled with the product. You can use them as out of box services or you may modify them as per your business need. In addition, the product also includes an Eclipse Dynamic Web Project that may be used as a template for developing new web services and modifying the ones that are bundled with the product.

This slide contains some of the out of box web services that the MDMCS administrator can use. The search service returns a string of XML representation for the search result set by the search query in WQL language. There are no limitations on this search and you may define any search criteria in the WQL.

The product web services can be used to create, update, delete, or retrieve information from the referenced item. The Item object included as an argument must include container name, primary key value and display name.

The scheduler web services implement a complete range of scheduler function and have a broad range of utilization. They may be used to create a report job, create a schedule to run a job, and to retrieve the status of a schedule for a job. They may also be used to stop a scheduled run, fetch details of a particular schedule for a job or all the schedules for the job, or all the jobs currently in the system.

You can also modify any of these web services or use them as a template to develop new ones for specialized functions. This is discussed in greater detail in the subsequent slides.

Deploying sample web services

- Difference between Weblogic and WebSphere® Application Server
- Steps for WebSphere Application Server
 - Ensure administrative server is running
 - \$WAS_HOME/profiles/AppServer_Name/bin/serverStatus.sh
 - \$WAS_HOME/profiles/AppServer_Name/bin/startServer.sh
 - \$TOP/samples/webservices/deploy/websphere/install_ws.sh
 - wsadmin command prompt
 - \$WAS_HOME/profiles/AppSvr01/bin/wsadmin.sh
 - \$AdminApp install EARfile "-usedefaultbindings -deployws"
 - WebSphere Application Server administrator console

Sample services need to be deployed before you can use them. This presentation reviews the steps required for deploying in WebSphere Application Server. Follow equivalent steps for other application servers like Weblogic. Note that web services need to be re-deployed in Weblogic every time you restart the application server. With WebSphere Application Server, web services need to be deployed once and can be used through multiple restarts.

In order to deploy these sample web services, ensure that the administration server, that is server1, is running. You can check the status using serverStatus.sh and if it is not running, start it using startServer.sh. After starting it, you can start the install_ws.sh to deploy it. Note that currently this script does not support horizontal and vertical MDMCS clusters but it will not work for Weblogic or a WebSphere Application Server node cluster.

If you are using a set-up which is not supported by the script, you need to wrap the web services in a .ear file and deploy them manually. You have two options for doing so; wsadmin command prompt and WebSphere Application Server administration console. For WebSphere Application Server command prompt, customize and use the command displayed on this slide. For the administration console, deploy the .ear file as a "New Enterprise Application. Before deploying manually, you need to ensure the JVM arguments like classpath and custom properties are properly set. See the IBM Education Assistant module "Developing Web Services" for details on how to configure them. This configuration is not needed if you use install_ws because the script configures these variables itself.

Accessing sample web service

- `http://hostname:Port/ws/Name_of_WebService`
 - Example: `http://hostname:7537/ws/SchedulerService`



{`http://samples.webservices.pim.ibm.com/`}SchedulerService

Hello! This is an Axis2 Web Service!

After deploying the web service, you can access it using the web address as displayed on this slide. Here hostname is the name of the server, port is the port number where the MDMCS application server process listens and ws is the default context root for all web services bundled with the product. Web services and MDMCS application server use the same port for communicating.

Since these have not been implemented as per your custom need, you will get a message as displayed in this screen capture. Once you customize it, it will perform the computation that you need.

Confirming deployment

- "WebSphere Application Server administrative console" > "Services" > "Service Provider"

Select	Name	Type	Deployed Asset	Status
You can administer the following resources:				
<input type="checkbox"/>	CatalogService	JAX-WS	ccd_sup3_7537_ws	➔
<input type="checkbox"/>	CategoryService	JAX-WS	ccd_sup3_7537_ws	➔
<input type="checkbox"/>	CollaborationService	JAX-WS	ccd_sup3_7537_ws	➔
<input type="checkbox"/>	DocumentService	JAX-WS	ccd_sup3_7537_ws	➔
<input type="checkbox"/>	HierarchyService	JAX-WS	ccd_sup3_7537_ws	➔
<input type="checkbox"/>	LookupTableService	JAX-WS	ccd_sup3_7537_ws	➔
<input type="checkbox"/>	ProductService	JAX-WS	ccd_sup3_7537_ws	➔
<input type="checkbox"/>	SampleService	JAX-WS	ccd_sup3_7537_ws	➔
<input type="checkbox"/>	SchedulerService	JAX-WS	ccd_sup3_7537_ws	➔
<input type="checkbox"/>	SearchService	JAX-WS	ccd_sup3_7537_ws	➔
<input type="checkbox"/>	SecurityService	JAX-WS	ccd_sup3_7537_ws	➔
<input type="checkbox"/>	SpecService	JAX-WS	ccd_sup3_7537_ws	➔
<input type="checkbox"/>	WorkflowService	JAX-WS	ccd_sup3_7537_ws	➔
Total 13				

7

Leveraging web services features bundled with MDMCS

© 2013 IBM Corporation

You can confirm and review the status of deployed web services through the "Service Provider" window of the WebSphere Application Server administrative console. The deployed web services opens as displayed in this screen capture. Remember, there is no implementation in these deployed services. The implementation aspect is discussed in the next slides.

Setting up workspace

- Download RSA using Passport Advantage® site
 - <https://www-112.ibm.com/software/howtobuy/softwareandservices/passportadvantage>
- For licensing question
 - <http://www-01.ibm.com/software/rational/support/licensing/index.html>
- Install RSA
- Set up workspace in RSA dedicated to web services

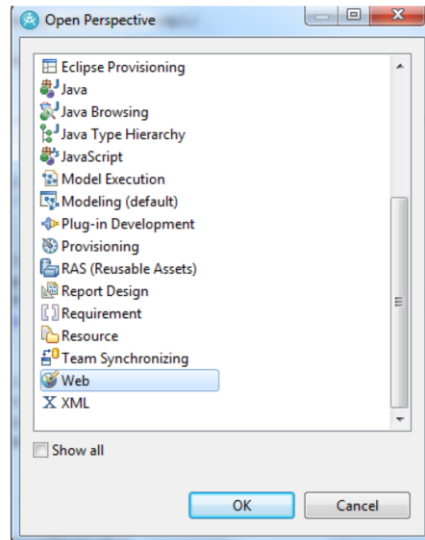
To implement and test web services, you will need an integrated Eclipse like development environment. RSA is IBM's version of an Eclipse like platform and currently it is the only platform which has been tested with MDMCS and hence, supported by MDMCS Support.

To download RSA, use the Passport Advantage site. The link is displayed on this slide. MDMCS comes with an RSA license and you can use that to validate it. The number of licenses are limited and you may have to buy additional licenses. For further licensing questions, go to the link displayed on this slide.

After downloading, install it and contact the IBM Rational help desk if you have questions. After installation, set up a workspace for your web services issues.

Configuring workspace (1 of 3)

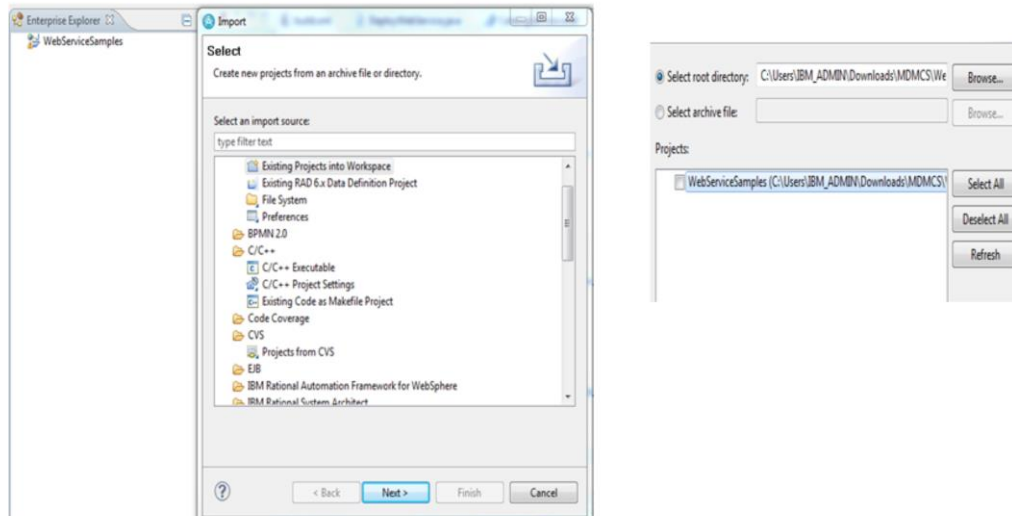
“Window” > “Open Perspective” > Select Web



To work with web services, you need to change the perspective to web. Follow the prompts displayed on the screen to change it.

Configuring workspace (2 of 3)

- Download \$TOP/samples/webservices/webservices.samples.src.zip
- “Enterprise Explorer” > Right click > “Import” > “Existing Project into Workspace”



10

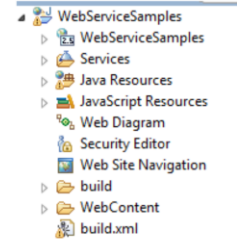
Leveraging web services features bundled with MDMCS

© 2013 IBM Corporation

After creating a workspace, you need to import the web service sample build. First, download the web services archive file from the location displayed on this slide to your hard disk. Then, use the Import an existing project into the workspace option to import this build.

Configuring workspace (3 of 3)

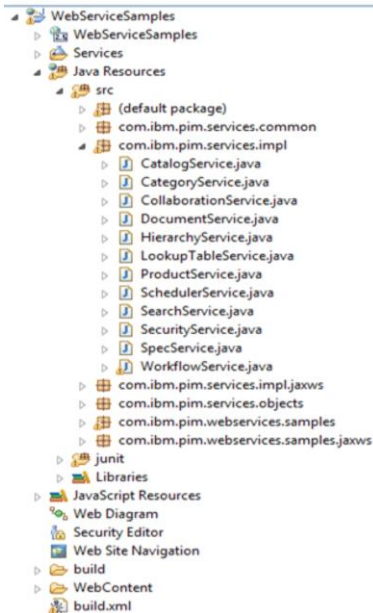
- Left navigation pane structure
- Right click "Build.xml" > "Run As" > "External Tools Configuration"
- In Main tab, "Arguments" section
 - Dmdmpim.home=<MDMCS install directory>
- Click "Apply" then "Run"



After you import the build, the directory structure is as displayed on this slide. Initially, you may see some errors because it has not been built properly. Right click Build.xml, select "Run As" and then "External Tools Configuration". The resulting screen is for run time parameters. In the Arguments section of the main tab, define a home parameter as displayed on this slide which points to the MDMCS download directory. This should contain the contents of \$TOP from the instance that you want to connect to. Specify the location of the \$TOP directory. Click Apply and click Run to complete the build. If the variable was defined correctly, there should not be any errors in the build.

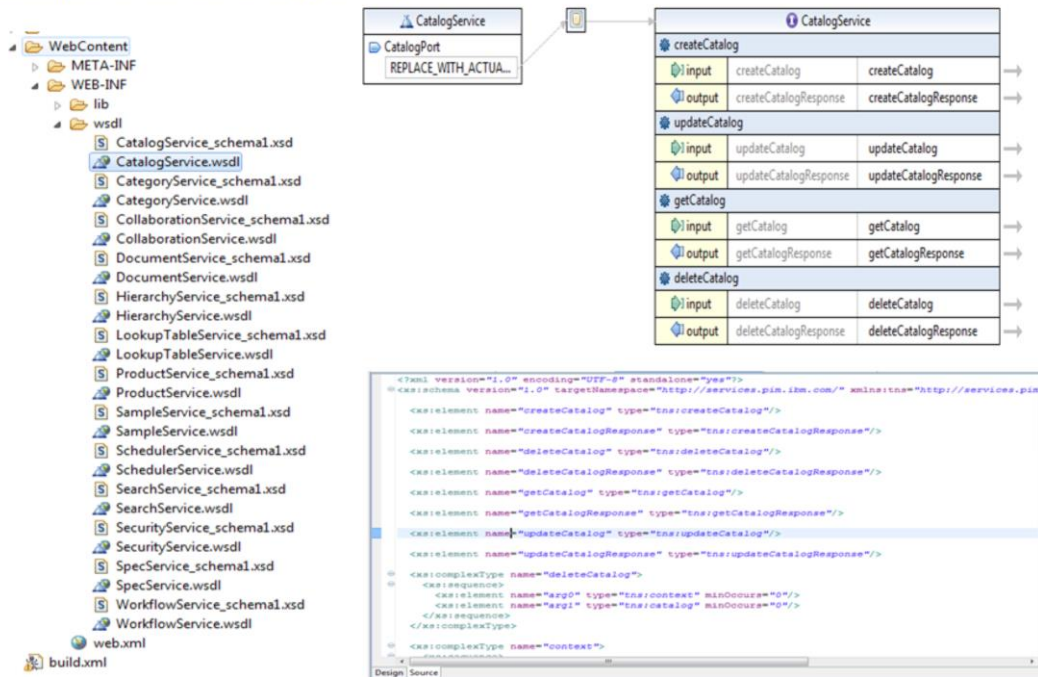
Note that if you did not add any external jars to this workspace, then all the necessary tasks have been completed by the script.

Left pane structure (1 of 2)



You will notice that the structure of the left pane is similar to what is displayed on this slide. A key thing to note is that the source code for the implementation of the web service is stored under “Java Resources” > “src” directory. You can click any of the web services to view the source code.

Left pane structure (2 of 2)



13

Leveraging web services features bundled with MDMS

© 2013 IBM Corporation

You will notice that all the WSDL files are under the “WebContent” then “WEB-INF” then “wsdl” directory as displayed on this slide in the figure to the left. The right part of the figure shows the simplified view of the WSDL in RSA. It shows the various options with the web service and the expected input and output parameters. For example, in the web service displayed on this slide, the WSDL illustrates that you can use CatalogService which has four web services to create, update, delete and get the details of a catalog.

The schematic diagram is from the design tab; you can switch to the “Source” tab to see the actual WSDL file. You will observe that some source WSDLs specify the SOAP address location as "REPLACE_WITH_ACTUAL_URL". Replace this with the actual URL of your instance in the WSDL file.

Testing web services (1 of 3)

Right click WSDL file > "Web Services" > "Test with Web Services Explorer"

The screenshot displays the Web Services Explorer application. The left pane shows a tree view of the WSDL file structure, with the 'CatalogService' node selected. The right pane shows the 'WSDL Binding Details' for the selected service. It includes a table of operations and a list of endpoints.

WSDL Binding Details

Shown below are the details for this SOAP <binding> element. Click on an operation to fill in its parameters and invoke it or specify additional endpoints.

Operations

Name	Documentation
getCatalog	--
createCatalog	--
deleteCatalog	--
updateCatalog	--

Endpoints [Add](#) [Remove](#)

Endpoints
<input type="checkbox"/> REPLACE_WITH_ACTUAL_URL
<input type="checkbox"/> http://supapp12.svl.ibm.com:7537/ws/CatalogService

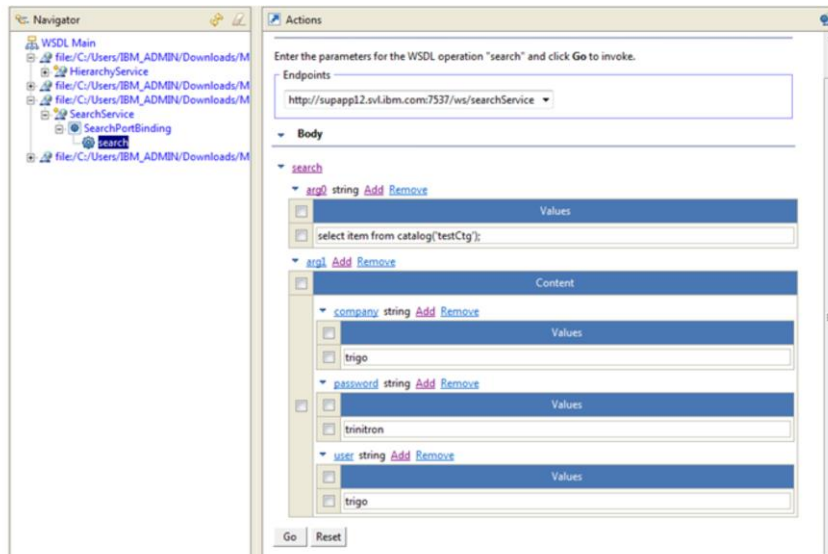
Status

IWAB03881 Endpoints were successfully updated.

14 Leveraging web services features bundled with MDMS © 2013 IBM Corporation

RSA provides some built in features to test web services. One of them is the web services explorer which you can get to by right clicking the WSDL file, selecting "Web services" and then "Test with Web Services Explorer". The slide illustrates this approach. You can specify the actual URL which is the one where this web service is deployed in the WSDL file to populate the end point. The URL in this case is http://hostname:port/ws/CatalogService. The referenced node contains four web services, each one of which may be invoked.

Testing web services (2 of 3)



15

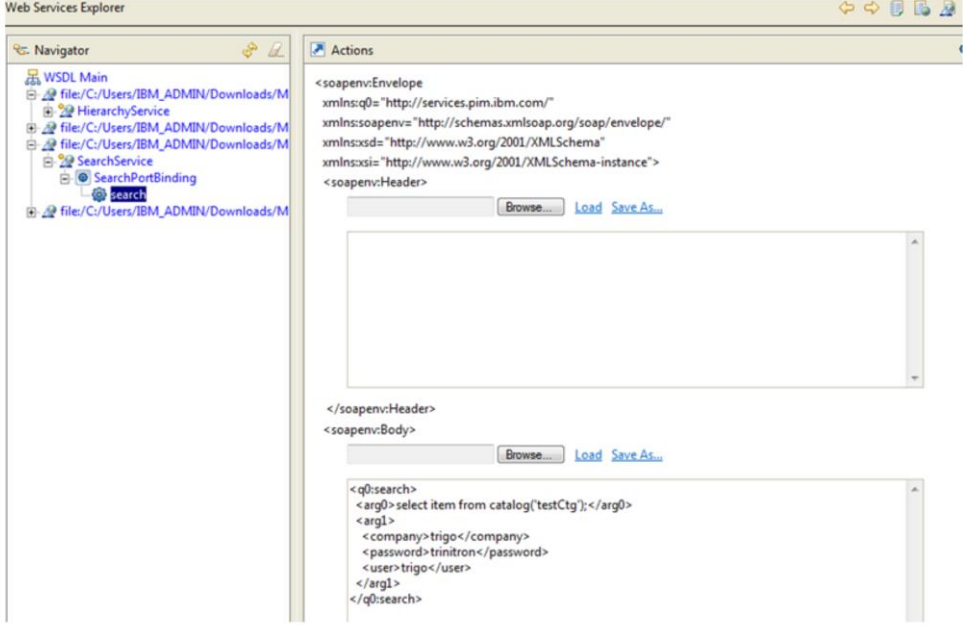
Leveraging web services features bundled with MDMCS

© 2013 IBM Corporation

Displayed on this slide is a sample apparatus to test a web service. Select the end point which was defined in the previous slide. The first parameter is the WQL that is started by this search web service. This WQL will return all items within the specified catalog. The context parameters like company, password, and user are the credentials which is used by this web service to connect to MDMCS. Click go to start the web service and test the returned results.

These parameters are defined in the web service implementation code and may be different for other web services.

Testing web services (3 of 3)



The screenshot shows the Web Services Explorer interface. On the left, the Navigator pane displays a tree view with 'Search' selected under 'SearchPortBinding'. The main area shows the SOAP envelope structure with the following XML content:

```
<soapenv:Envelope
xmlns:q0="http://services.pim.ibm.com/"
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns: xsi="http://www.w3.org/2001/XMLSchema-instance">
<soapenv:Header>
</soapenv:Header>
<soapenv:Body>
<q0:search>
<arg0>select item from catalog('testCtg');</arg0>
<arg1>
<company>trigo</company>
<password>trinitron</password>
<user>trigo</user>
</arg1>
</q0:search>
```

16

Leveraging web services features bundled with MDMCS

© 2013 IBM Corporation

On the same test screen, if you click the source link at the top right of the screen, it will show you the SOAP envelope which was generated and sent by this web service. This is helpful if you are familiar with SOAP programming and SOAP envelopes. You can also develop your web services using XML instead of SOAP.

You can use a similar approach to modify and test other sample web services or any of your custom web services.

Modifying source code

```

}
}
}
/**
 * Search the system.
 * <p>
 * @param wql the query.
 * @return the result.
 * @param wsContext the given context.
 */
@WebMethod
public String Search(String wql, Context wsContext)
{
    try
    {
        oom.ibm.pim.context.Context ctx = wsContext.toPIMObject();
        oom.ibm.pim.search.SearchQuery wqlQuery = ctx.createSearchQuery(wql);
        oom.ibm.pim.search.SearchResultSet rs = wqlQuery.execute();

        StringBuffer buf = new StringBuffer();
        buf.append("<results>\n");
        while (rs.next())
        {
            toXML(rs, wqlQuery.getColumnTypes(), buf);
        }
        buf.append("</results>\n");

        return buf.toString();
    }
    catch (Exception e)
    {
        e.printStackTrace();
    }
    return null;
}

```

17 Leveraging web services features bundled with MDMCS © 2013 IBM Corporation

As mentioned previously, the source code is located under “Java Resources” then “src”. Review the source code and edit it as needed. As displayed on this slide, the search method takes in a WQL and context to perform the search and these are the parameters you set while you were testing the search web service. You can modify the implementation in any way that is best for your business use case.

You can even add custom methods to any of the sample web services and use them as invocation points for the custom implementation. After such modifications, you can build the file again using build.xml to update the WSDL file as per the modification.

If you are not adding any new web services, then you can use the install_ws script to deploy them after modification. For custom web services, you have to deploy them manually.

Note that all web service methods begin with @WebMethod keyword. Without this keyword, the builder will not recognize them as a web service.

Adding web services

- com.ibm.pim.services.impl package
- build.xml

```
<!-- search services -->  
<ant antfile="{webservices.samples.dir}/build.xml" target="webservices.samples.gen.SEI">  
<property name="SEI" value="com.ibm.pim.services.impl.SearchService"/> </ant>
```

Another option is to use bundled web services as a template to create new ones. Create a new web services under com.ibm.pim.services.impl package and use the structure of existing web services for your custom build.

Once you add a new web service, the existing structure of build.xml is no longer accurate and you need to add an entry for the new web service in the build.xml file. Right click build.xml and select Open. Then add an entry for the new web service in accordance with the existing entries. An example of Search Service entry is displayed on this slide. After this, you can continue to leverage build.xml for building and compiling.

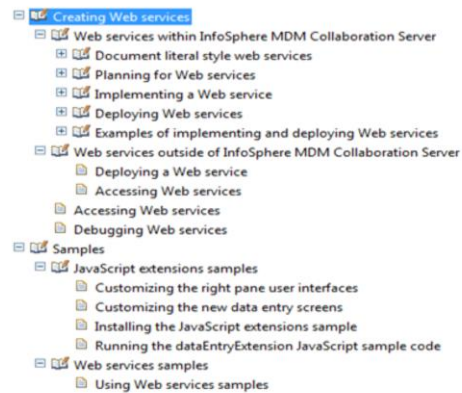
References

- Web services

http://publib.boulder.ibm.com/infocenter/mdm/v10r0m0/topic/com.ibm.pim.app.doc/webservices/pim_con_dev_creatingwebservices.html

- Sample web services

http://publib.boulder.ibm.com/infocenter/mdm/v10r0m0/topic/com.ibm.pim.app.doc/samples/pim_con_soldevsamples.html



The Information Center has comprehensive documentation on implementing general web services and these sample web services bundled with the product. Review the links displayed on this slide for further details.

Invoking custom hosted secure web services (1 of 2)

- Import license to application server
- For .cer certificates
 - \$WAS_HOME/java/jre/bin/keytool -importcert -file Location_of_Certificate -keystore keystore.jks -alias "Alias"
- Make \$TOP/samples/webservices/webservices.clients.jar part of your classpath
 - \$TOP/jars or \$TOP/bin/conf/classpath/jars-custom.txt
 - \$TOP/bin/configureEnv.sh
- Invoke web service

There is frequently a need to invoke a secure web service from MDMCS which is hosted outside of the product. To do so, first import the SSL license into your application server. Without it, the appserver JVM is unable to connect to the secure server hosting the web service. In WebSphere Application Server, to import a certificate with a .cer extension, you can use the command displayed on this slide to import the certificate.

Before invoking secure web services, you have to add webservices.client.jar to the classpath. Doing this is a two step process. First is to either add the jar to the \$TOP/jars directory or to add the location of the jar to the jars-custom.txt file. After that, run configureEnv.sh to update the classpath. You may also do this manually in your application server.

Finally, you need to invoke the web service using specific code. A sample has been provided on the next slide.

Invoking custom hosted secure web services (2 of 2)

```
▪ var endpointUrl = https://Hostname:Port/Location_of_URL";
▪ var requestXmlStr = "<?xml version='1.0' encoding='UTF-8'?>";
▪ requestXmlStr = requestXmlStr + "<soapenv:Envelope xmlns:soapenv='http://schemas.xmlsoap.org/soap/envelope/'> ";
▪ requestXmlStr = requestXmlStr + "<soapenv:Header> ";
▪ requestXmlStr = requestXmlStr + " <wsse:Security xmlns:wsse='http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd' soapenv:mustUnderstand='1'> ";
▪ requestXmlStr = requestXmlStr + " <wsse:UsernameToken> ";
▪ requestXmlStr = requestXmlStr + " <wsse:Username>cusadmin</wsse:Username> ";
▪ requestXmlStr = requestXmlStr + " <wsse:Password>cusadmin</wsse:Password> ";
▪ requestXmlStr = requestXmlStr + " </wsse:UsernameToken> ";
▪ requestXmlStr = requestXmlStr + " </wsse:Security> ";
▪ requestXmlStr = requestXmlStr + "</soapenv:Header> ";
▪ requestXmlStr = requestXmlStr + "<soapenv:Body> ";
▪ .....
▪ requestXmlStr = requestXmlStr + "</soapenv:Body> ";
▪ requestXmlStr = requestXmlStr + "</soapenv:Envelope> ";
▪ var soapClientConstructor = createJavaConstructor("com.ibm.pim.services.clients.S SoapClient");
▪ var client = runJavaConstructor(soapClientConstructor );
▪ var soapClientMethod = createJavaMethod("com.ibm.pim.services.clients.S SoapClient", "invokeService", "java.lang.String", "java.lang.String", "boolean");
▪ var resp = runJavaMethod(client, soapClientMethod, endpointUrl, requestXmlStr, false);
```

This slide displays a sample code to invoke the custom hosted secure web service.

Trademarks, disclaimer, and copyright information

IBM, the IBM logo, ibm.com, InfoSphere, Passport Advantage, Rational, and WebSphere are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of other IBM trademarks is available on the web at "[Copyright and trademark information](http://www.ibm.com/legal/copytrade.shtml)" at <http://www.ibm.com/legal/copytrade.shtml>

Other company, product, or service names may be trademarks or service marks of others.

THE INFORMATION CONTAINED IN THIS PRESENTATION IS PROVIDED FOR INFORMATIONAL PURPOSES ONLY. WHILE EFFORTS WERE MADE TO VERIFY THE COMPLETENESS AND ACCURACY OF THE INFORMATION CONTAINED IN THIS PRESENTATION, IT IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. IN ADDITION, THIS INFORMATION IS BASED ON IBM'S CURRENT PRODUCT PLANS AND STRATEGY, WHICH ARE SUBJECT TO CHANGE BY IBM WITHOUT NOTICE. IBM SHALL NOT BE RESPONSIBLE FOR ANY DAMAGES ARISING OUT OF THE USE OF, OR OTHERWISE RELATED TO, THIS PRESENTATION OR ANY OTHER DOCUMENTATION. NOTHING CONTAINED IN THIS PRESENTATION IS INTENDED TO, NOR SHALL HAVE THE EFFECT OF, CREATING ANY WARRANTIES OR REPRESENTATIONS FROM IBM (OR ITS SUPPLIERS OR LICENSORS), OR ALTERING THE TERMS AND CONDITIONS OF ANY AGREEMENT OR LICENSE GOVERNING THE USE OF IBM PRODUCTS OR SOFTWARE.

© Copyright International Business Machines Corporation 2013. All rights reserved.