DP600 Web Services API

Version 1.0.0.6
May 21, 2009

Tables and Figures

SECTION 1
API DESCRIPTION

0.1 Overview

To facilitate the requirement for 3rd party developers to interact with the DP600 the following set of web service interfaces are currently available.

- Submit Workorders
- Retrieve Workorders
- Retrieve Workflow Profile Names
- Check Workorder Status
- Retrieve Workorder Reports
- Delete Workorders

0.2 Dependent Documents

http://www.w3.org/TR/ws-arch/

http://www.w3.org/TR/soap/

http://www.w3.org/XML/Schema

https://jax-ws.dev.java.net/

http://msdn2.microsoft.com/en-us/netframework/aa663324.aspx

http://www.codegear.com/products/delphi/win32

0.3 Architecture

The DP600 web service binding style is document/literal over SOAP whereby the gist of the messaging is XML-based (Documents) to facilitate our requirement to evolve DP600 document schemas independently from the underlying interface as described in the DP600 web service WSDL. Albeit the developer has to do additional XML work in submitting and responding to a Workorder request because the document schemas need to be negotiated out of band, i.e., both the DP600 and the client need prior knowledge of the “xsd:any” type payload contents because the WSDL file does not describe the schema of the expected XML documents; this separation of interface from the application/business XML documents lends itself well to future improvements and enhancements with minimal or no impact on existing clients in the field.

Due to the computational complexity of processing performed by the DP600 the web services are structured such that they are non-blocking by nature. Therefore, a web service client may interact with a DP600 web service in a non-blocking, asynchronous approach. To achieve this behavior the DP600 provides for an asynchronous request/reply polling model. In the future the possibility of a callback may be added in addition to the poll interface but at the time of this writing only the poll model is currently available. In addtion, RESTful web services are also being considered. **Figure 1.1** below depicts a request/reply asynchronous Workorder submission sequence with polling. **Figure 1.2** below depicts a request/reply Workorder Report query sequence.

0.4 Workorders

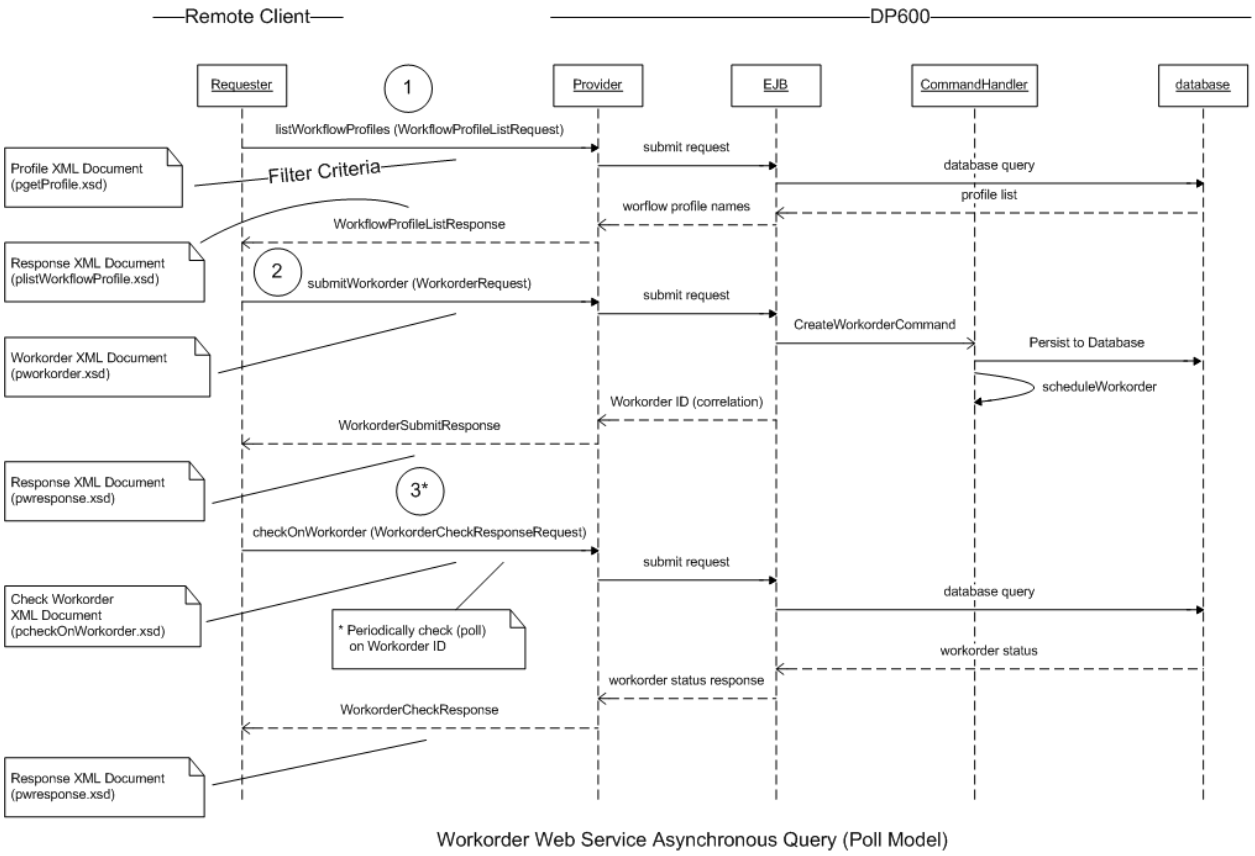

Figure 0-1 Workorder Submit Sequence

The general procedure to submit a Workorder is as follows (refer to Figure 0-1 Workorder Submit Sequence):

1. Retrieve a list of named workflow profiles (see 0.6.1).
2. Submit a Workorder XML document and populate the document with the desired profile name returned from step 1 in addition to all the other required elements within the XML document. The Response to this request is an XML document with a Workorder ID (correlation ID). The client typically will cache the Workorder ID for subsequent Workorder status queries or to retrieve Workorder reports. When populating the Workorder document the Source and Destination must be a valid URI, e.g., smb://[*username*[: *password*]@] *hostname*[: *port*][*absolute-path*]. Likewise, ftp://[*username*[: *password*]@] *hostname*[: *port*][*absolute-path*] or sftp://[*username*[: *password*]@] *hostname*[: *port*][*absolute-path*]. And for local shares file://username:password@/mount/someshare/somedir. The *username* and *password* can be optional if remote shares are configured anonymously (see 0.6.2).
3. Periodically check on the status of the Workorder given the Workorder ID returned by step 2. The Response to this request is an XML document with status in addition to the poll/refresh period (see 0.6.3).

Note: *Both submitWorkorder and checkOnWorkorder return a Response object. If the property status is not FINISHED, then a subsequent checkResponse shall be no sooner than the value specified by the refresh time (minutes) property returned in the body of the Response. If the property status is FINISHED, it indicates the service is complete. The submitWorkorder Response contains a Workorder ID (correlation ID) which must be used as the input to checkResponse. Otherwise, the client will not be able to correlate its initial Workorder submittal request with a correct response. At any time a Workorder report request can be made for a given Workorder ID to get the latest reports as they are generated by the profile executed by the Workorder.*

0.5 Reports

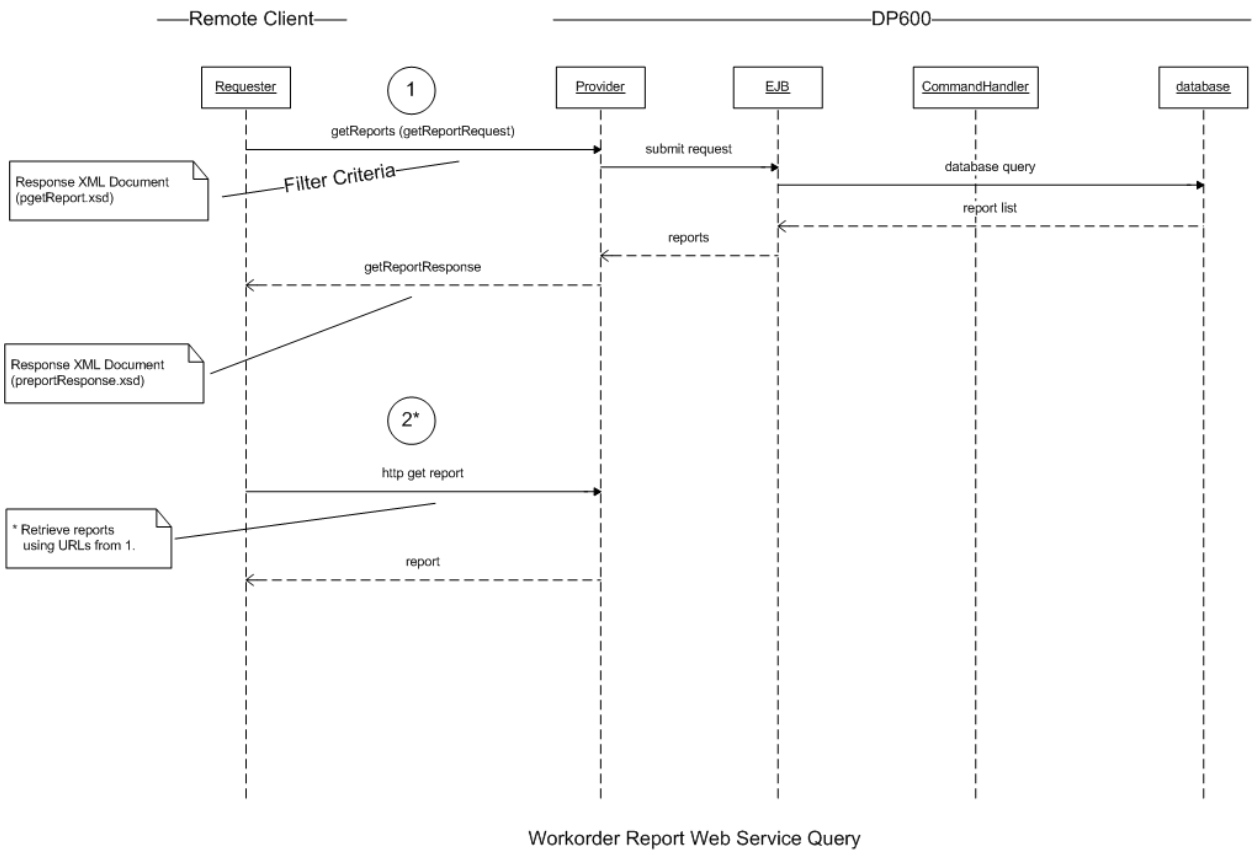

Figure 0-2 Report Query Sequence

The general procedure to retrieve a Workorder report is as follows (refer to Figure 0-2 Report Query Sequence):

1. Retrieve a list of report XML documents associated to a Workorder using its unique Workorder ID cached previously when submitting a Workorder or by retrieving a list of existing Workorders based on report severity.
2. Actual reports can be retrieved thru the URL element contained in a report XML document (1). And the report formatting is determined by the mimeType element contained in a report XML document.

0.6 Web Service Messaging

0.6.1 Method listWorkflowProfiles ()

Method listWorkflowProfiles () takes as its argument a XML (pgetProfile.xsd) filter/criteria document and returns a XML **(**plistWorkflowProfile.xsd) document as follows:

For pgetProfile.xsd schema the sample XML is

```
<pgetProfile:pgetProfile xmlns:pgetProfile="http://getprofile.nlps.dolby.com">
   <pgetProfile:profileGroup>default</pgetProfile:profileGroup>
</pgetProfile:pgetProfile>
```

For plistWorkflowProfile.xsd schema the sample XML is

```
<profiles:profiles xmlns:profiles="http://profile.nlps.dolby.com"
xmlns:common="common.nlps.dolby.com">
  <profiles:status>53 profiles were retrieved.</profiles:status>
  <profiles:profile>
    <profiles:profilename>WAV/E_WAV/AC3-2</profiles:profilename>
  </profiles:profile>
  <profiles:profile>
    <profiles:profilename>GXF_GXF-2</profiles:profilename>
  </profiles:profile>
  <profiles:profile>
    <profiles:profilename>GXF_GXF-7</profiles:profilename>
  </profiles:profile>
  ....
</profiles:profiles>
```

0.6.2 Method submitWorkorder ()

Method submitWorkorder () takes as its argument a XML (pworkorder.xsd) Workorder document and returns a XML (pwresponse.xsd) document as follows:

For pworkorder.xsd schema the sample XML is

```
<workorder xmlns="workorder.nlps.dolby.com" xmlns:common="common.nlps.dolby.com">
<name>wstest</name>
<scheduledStartTime>2007-09-19T05:08:26</scheduledStartTime>
```

<type>SWO</type>
<source>
<common:filename>smb://mediastore1.eng/dp600/content/default_profile_assets/profile_29/10347422s.dvl.mpg</common:filename>
</source>
<destination>
<common:filename>smb://mediastore1.eng/dp600/content/devel_test_assets/temp_out</common:filename>
</destination>
<wfprofile>TS_TS-1</wfprofile>
<wopriority>1</wopriority>
</workorder>

For pwresponse.xsd schema the sample XML is

<pwresponse:pwresponse xmlns:pwresponse="http://response.nlps.dolby.com" xmlns:common="common.nlps.dolby.com">
<pwresponse:status/>
<pwresponse:refreshtime>1</pwresponse:refreshtime>
<pwresponse:correlationID>644289ed-7621-4479-ae94-463c375c65b7</pwresponse:correlationID>
</pwresponse:pwresponse>

0.6.3 Method checkOnWorkorder ()

Method checkOnWorkorder () takes as its argument a XML (pcheckOnWorkorder.xsd) filter/criteria document and returns a XML (pwresponse.xsd) document as follows:

For pcheckOnWorkorder.xsd schema the sample XML is

<pwcheckresponse xmlns="http://checkresponse.nlps.dolby.com" xmlns:common="common.nlps.dolby.com">
<correlationID>644289ed-7621-4479-ae94-463c375c65b7 </correlationID>
</pwcheckresponse>

For pwresponse.xsd schema the sample XML is

<pwresponse:pwresponse xmlns:pwresponse="http://response.nlps.dolby.com" xmlns:common="common.nlps.dolby.com">
<pwresponse:status>FINISHED</pwresponse:status>
<pwresponse:refreshtime>0</pwresponse:refreshtime>
<pwresponse:correlationID>644289ed-7621-4479-ae94-463c375c65b7</pwresponse:correlationID>
</pwresponse:pwresponse>

0.6.4 Method getReports ()

Method getReports () takes as its argument a XML (pgetReport.xsd) filter/criteria document and returns a XML (preportResponse.xsd) document as follows:

* Due to performance reasons, the getReports() API call will only return up to 1000 reports. The optional “index” field in the schema provides a way for clients to request for reports beyond it. E.g. if the client would like to get the next 1000 reports from 5000, then the index field will be 5000.

For pgetReport.xsd schema the sample XML is

```
<pgetReport:pgetReport xmlns:pgetReport="http://getreport.nlps.dolby.com"
xmlns:common="common.nlps.dolby.com">
<pgetReport:index>0</pgetReport:index>
<pgetReport:workorderID>
      9aa3902a-692f-4a02-beeb-1b2426581ff3
</pgetReport:workorderID>
<pgetReport:reportSeverity>
     INFO
</pgetReport:reportSeverity>
</pgetReport:pgetReport>
```

For preportResponse.xsd schema the sample XML is

```
<reports:reports xmlns:reports="http://report.nlps.dolby.com"
xmlns:common="common.nlps.dolby.com">
  <reports:status>
    6 reports were retrieved for workorder ID 9aa3902a-692f-4a02-beeb-
1b2426581ff3.
  </reports:status>
  <reports:report>
    <reports:date>Thu Dec 20 16:25:17 PST 2007</reports:date>
    <reports:severity>INFO</reports:severity>
    <reports:description>MPEG2TS Remux Log (ingestFile)</reports:description>
    <reports:url>
      http://orochi:80/dp600-reportms-server/getReport?reportId=48fa7b4d-e697-
45de-90e0-4c198492b872
    </reports:url>
    <reports:mimeType>application/xml</reports:mimeType>
  </reports:report>
  <reports:report>
    <reports:date>Thu Dec 20 16:25:16 PST 2007</reports:date>
    <reports:severity>INFO</reports:severity>
    <reports:description>
      Loudness Correction Log (ingestFile/estream[1E2])</reports:description>
      <reports:url>
        http://orochi:80/dp600-reportms-server/getReport?reportId=ee70c703-
7e9d-4a38-9c14-fc4df2656515
      </reports:url>
      <reports:mimeType>application/xml</reports:mimeType>
    </reports:report>
  <reports:report>
```

```
    <reports:date>Thu Dec 20 16:25:15 PST 2007</reports:date>
    <reports:severity>INFO</reports:severity>
    <reports:description>
      Estimated Levels Log (ingestFile/estream[1E2])</reports:description>
      <reports:url>
        http://orochi:80/dp600-reportms-server/getReport?reportId=d701ba4b-
beaa-4914-9feb-784d148b7794
      </reports:url>
      <reports:mimeType>text/csv</reports:mimeType>
    </reports:report>
  <reports:report>
    <reports:date>Thu Dec 20 16:25:15 PST 2007</reports:date>
    <reports:severity>INFO</reports:severity>
    <reports:description>
      Loudness Estimation Log (ingestFile/estream[1E2])</reports:description>
      <reports:url>
        http://orochi:80/dp600-reportms-server/getReport?reportId=0d6ca2d3-
d126-41b4-a3a6-600f56cbe989
      </reports:url>
      <reports:mimeType>application/xml</reports:mimeType>
    </reports:report>
```

0.6.5 Method getWorkorders ()

Method getWorkorders () takes as its argument a XML (pgetWorkorder.xsd) filter/criteria document and returns a XML (pworkorderResponse.xsd) document as follows:

* Due to performance reasons, the getWorkorders() API call will only return up to 1000 workorders. The optional “index” field in the schema provides a way for clients to request for workorders beyond it. E.g. if the client would like to get the next 1000 workorders from 5000, then the index field will be 5000.

For pgetWorkorder.xsd schema the sample XML is

For a specific Workorder

```
<pgetWorkorder:pgetWorkorder xmlns:pgetWorkorder="http://getworkorder.nlps.dolby.com">
  <pgetWorkorder:workorderID>
    9aa3902a-692f-4a02-beeb-1b2426581ff3
  </pgetWorkorder:workorderID>
</pgetWorkorder:pgetWorkorder>
```

For a list of Workorders

```
<pgetWorkorder:pgetWorkorder xmlns:pgetWorkorder="http://getworkorder.nlps.dolby.com">
<pgetWorkorder:index>0</pgetWorkorder:index>
</pgetWorkorder:pgetWorkorder>
```

For pworkorderResponse.xsd schema the sample XML is

<workorders:workorders xmlns:workorders="http://workorder.nlps.dolby.com" xmlns:workorder="workorder.nlps.dolby.com" xmlns:common="common.nlps.dolby.com">
  <workorders:status>1 workorders were retrieved for workorder ID 9aa3902a-692f-4a02-beeb-1b2426581ff3.</workorders:status>
  <workorders:workorder>
    <workorder:name>wstest</workorder:name>
    <workorder:scheduledStartTime>2007-09-19T06:08:26</workorder:scheduledStartTime>
    <workorder:type>SWO</workorder:type>
    <workorder:source>

<common:filename>smb://mediastore1.eng/dp600/content/devel_test_assets/temp_out/test_in/10347422s.dvl.mpg</common:filename>
    </workorder:source>
    <workorder:destination>

<common:filename>smb://mediastore1.eng/dp600/content/devel_test_assets/temp_out/test_out</common:filename>
    </workorder:destination>
    <workorder:wfprofile>TS_TS-1</workorder:wfprofile>
  </workorders:workorder>
</workorders:workorders>

0.6.6 Method deleteWorkorder ()

For pgetWorkorder.xsd schema the sample XML is

<pgetWorkorder:pgetWorkorder xmlns:pgetWorkorder="http://getworkorder.nlps.dolby.com">
  <pgetWorkorder:workorderID>
     9aa3902a-692f-4a02-beeb-1b2426581ff3
  </pgetWorkorder:workorderID>
</pgetWorkorder:pgetWorkorder>

0.6.7 Document Schemas & WSDL

For convenience all XML document schemas and WSDL are located on commissioned DP600s at the following URL's where *dp600hostname* is the name given to the DP600 at the time of commissioning.

http://*dp600hostname*/pcheckOnWorkorder.xsd

http://*dp600hostname*/pcommon.xsd

http://*dp600hostname*/pgetReport.xsd

http://*dp600hostname*/pgetProfile.xsd

http://*dp600hostname*/pgetWorkorder.xsd

http://*dp600hostname*/pworkorderResponse.xsd

http://*dp600hostname*/plistWorkflowProfile.xsd

http://*dp600hostname*/preportResponse.xsd

http://*dp600hostname*/pworkorder.xsd

http://*dp600hostname*/pwresponse.xsd

http://*dp600hostname*:80/WorkorderService/WorkorderWsImpl?wsdl/

Note: For reference launching a web browser and entering the above URL with a valid dp600hostname will display the schema.

SECTION 2
CLIENT EXAMPLES

The following examples illustrate how to submit and check on Workorder status. Additional services described in the WSDL behave in a similar fashion.

1.1 Java

This java example assumes the reader is somewhat familiar with the JAX-WS 2.0 web service stack. If not, refer to https://jax-ws.dev.java.net/ . In addition, refer to the service messaging discussed above for XML document formatting.

1. **For example use the wsimport command-line tool or IDE (Netbeans, Eclipse…) to generate the interface from the DP600 WSDL.**
   http://*dp600hostname*:80/WorkorderService/WorkorderWsImpl?wsdl

Note: dp600hostname is the name given to the DP600 at the time of commissioning and the wsimport command line options may differ from above based on development environment. Furthermore, client implementations are varied so the brief example below only illustrates the necessary web service calls to list Workflow profiles, submit Workorders, and check on Workorder status.

2. **For example to list Workflow Profiles.**

```
public void listWorkflowProfileExample () throws Exception
{
  try
  {
      WorkorderService service = new WorkorderService();
      WorkorderPortType port = service.getWorkorderPort();

      // call listWorkflowProfile
      ProfileListRequest profileListRequest = new ProfileListRequest();
      WorkflowProfileList profileList = port.
        listWorkflowProfiles(profileListRequest);
      Element responseData = (Element) profileList.getAny();

      ...
  }
  catch (Exception e)
  {
      System.out.println(
        "exception in listWorkflowProfileExample method " + e);
      throw new Exception(e);
  }
}
```

3. For example to submit a Workorder.

```
public void submitWorkorderWS (String srcFilePath, String destFilePath,
   String testProfileName) throws Exception
{
   Element elem = null;
   try
   {
       WorkorderService service = new WorkorderService();
       WorkorderPortType port = service.getWorkorderPort();
       String s = "<workorder xmlns='workorder.nlps.dolby.com'
   xmlns:common='common.nlps.dolby.com'><name>wstest</name><type>SWO</type>
   <source><common:filename>"+srcFilePath+"</common:filename></source><dest
   ination><common:filename>"+destFilePath+"</common:filename></destination
   ><wfprofile>"+testProfileName+"</wfprofile></workorder>";
       Element requestData = XMLUtil.getDocument(s).getDocumentElement();
       Request request_1 = new Request();
       request_1.setAny(requestData);
       Response result = port.submitWorkorder(request_1);
       elem = (Element) result.getAny();
       System.out.println("\nsubmitWorkorderWS runs successfully, received
       the following xml document:" + XMLUtil.getXML(elem)+"\n");
       woId = XMLUtil.getValue(elem.getOwnerDocument(), "correlationID");
       System.out.println ("Workorder ID: " + woId);
   }
   catch (Exception e)
   {
       System.out.println("exception in submitWorkorderWS method"+e);
       throw new Exception(e);
   }
 }
```

4. For example to check on Workorder status.

```
public void checkOnWorkorderExample () throws Exception
{
   try
   {
       WorkorderService service = new WorkorderService();
       WorkorderPortType port = service.getWorkorderPort();
       Element requestData = null;
       String s = "<?xml version='1.0' encoding='UTF-8'?>" +
         "<pwcheckresponse:pwcheckresponse
         xmlns:pwcheckresponse='http://checkresponse.nlps.dolby.com'
         xmlns:common='common.nlps.dolby.com'>" +
         "<pwcheckresponse:correlationID>" + woId +
         "</pwcheckresponse:correlationID>" +
         "</pwcheckresponse:pwcheckresponse>";
       requestData =  XMLUtil.getDocument(s).getDocumentElement();

       ResponseCheck request_1 = new ResponseCheck();
       request_1.setAny(requestData);
       Response result = port.checkOnWorkorder(request_1);
       Element responseData = (Element) result.getAny();
```

```
        String status = XMLUtil.getValue
          (responseData.getOwnerDocument(), "status");
        System.out.println("\ncheckOnWorkorderWS for UUID "+
          woId+" is called, workorder status is: "+status+"\n");

    }
    catch (Exception e)
    {
        System.out.println("exception in checkOnWorkorderWS method"+e);
        throw new Exception(e);
    }
 }
```

5. For example to get Workorders.

```
public void getWorkordersWS () throws Exception
 {
   try
   {
       WorkorderService service = new WorkorderService();
       WorkorderPortType port = service.getWorkorderPort();
       Element requestData = null;
       String s = "<?xml version='1.0' encoding='UTF-8'?>" +
         "<pgetWorkorder:pgetWorkorder
         xmlns:pgetWorkorder='http://getworkorder.nlps.dolby.com'>" +
         "<pgetWorkorder:workorderID>" + woId +
         "</pgetWorkorder:workorderID>" +
         "</pgetWorkorder:pgetWorkorder>";
       requestData =  XMLUtil.getDocument(s).getDocumentElement();

       WorkorderRequest request_1 = new WorkorderRequest();
       request_1.setAny(requestData);
       WorkorderList result = port.getWorkorders(request_1);
       Element responseData = (Element) result.getAny();

       System.out.println("\ngetWorkordersWS for UUID "+woId+" is called,
         response is: "+ XMLUtil.getXML(responseData) +"\n");

   }
   catch (Exception e)
   {
       System.out.println("exception in getWorkordersWS method"+e);
       throw new Exception(e);
   }
 }
```

6. For example to get reports.

```
public void getReportsWS () throws Exception
 {
   try
   {
       WorkorderService service = new WorkorderService();
       WorkorderPortType port = service.getWorkorderPort();
       Element requestData = null;
       String s = "<?xml version='1.0' encoding='UTF-8'?>" +
         "<pgetReport:pgetReport
         xmlns:pgetReport='http://getreport.nlps.dolby.com'>" +
```

```
          "<pgetReport:workorderID>" + woId + "</pgetReport:workorderID>" +
          "</pgetReport:pgetReport>";
        requestData =  XMLUtil.getDocument(s).getDocumentElement();

        ReportRequest request_1 = new ReportRequest();
        request_1.setAny(requestData);
        ReportList result = port.getReports(request_1);
        Element responseData = (Element) result.getAny();

        System.out.println("\ngetReportsWS for UUID "+woId+" is called,
          response is: "+ XMLUtil.getXML(responseData) +"\n");

    }
    catch (Exception e)
    {
        System.out.println("exception in getReportsWS method"+e);
        throw new Exception(e);
    }
 }
```

7. For example to delete Workorders.

```
public void deleteWorkorderWS () throws Exception
{
     try
     {
         WorkorderService service = new WorkorderService();
         WorkorderPortType port = service.getWorkorderPort();
         Element requestData = null;
         String s = "<?xml version='1.0' encoding='UTF-8'?>" +
               "<pgetWorkorder:pgetWorkorder
               xmlns:pgetWorkorde='http://getworkorder.nlps.dolby.com'>" +
               "<pgetWorkorder:workorderID>" + woId +
               "</pgetWorkorder:workorderID>" +
               "</pgetWorkorder:pgetWorkorder>";
         requestData =  XMLUtil.getDocument(s).getDocumentElement();

         DeleteWorkorderRequest request_1 = new DeleteWorkorderRequest();
         request_1.setAny(requestData);
         Response result = port.deleteWorkorder(request_1);
         Element responseData = (Element) result.getAny();

         System.out.println("\ndeleteWorkorderWS for UUID "+woId+
         " is called, response is: "+ XMLUtil.getXML(responseData) + "\n");
     }
     catch (Exception e)
     {
         System.out.println("exception in deleteWorkorderWS method"+e);
         throw new Exception(e);
     }
 }
```

1.2 Axis2 (Java)

This java example assumes the reader is somewhat familiar with the Apache Axis2 framework. If not, refer to http://ws.apache.org/axis2/index.html. In addition, refer to the service messaging discussed above for XML document formatting.

1. **For example use the wsdl2java command-line tool or IDE (Eclipse…) to generate the interface from the DP600 WSDL.**
   http://*dp600hostname*:80/WorkorderService/WorkorderWsImpl?wsdl

Note: dp600hostname is the name given to the DP600 at the time of commissioning and the wsimport command line options may differ from above based on development environment. Furthermore, client implementations are varied so the brief example below only illustrates the necessary web service calls to list Workflow profiles, submit Workorders, and check on Workorder status.

2. **For example to list Workflow Profiles.**

```
public static void listWorkflowProfiles() {
    try {
        WorkorderServiceStub stub = new WorkorderServiceStub();

        WorkorderServiceStub.ProfileListRequest0 req = new
              WorkorderServiceStub.ProfileListRequest0();
        WorkorderServiceStub.ProfileListRequest param = new
              WorkorderServiceStub.ProfileListRequest();

        OMFactory fac = OMAbstractFactory.getOMFactory();
        OMNamespace omNs =
              fac.createOMNamespace("http://getprofile.nlps.dolby.com",
              "ns1");
        OMElement elem = fac.createOMElement("pgetProfile", omNs);

        param.setExtraElement(elem);
        req.setProfileListRequest(param);

        WorkorderServiceStub.WorkflowProfileList9 res =
              stub.listWorkflowProfiles(req);

        System.out.println(
              res.getWorkflowProfileList().getExtraElement().toString());
    } catch (Exception e) {
        e.printStackTrace();
    }
}
```

3. **For example to submit a Workorder.**

```
public static void submitWorkorder() {
    try {
        WorkorderServiceStub stub = new WorkorderServiceStub();

        WorkorderServiceStub.Request2 req = new
              WorkorderServiceStub.Request2();
        WorkorderServiceStub.Request param = new
              WorkorderServiceStub.Request();

        File file = new File("workorder.xml");
        FileInputStream fis = new FileInputStream(file);
        XMLInputFactory xif = XMLInputFactory.newInstance();
        XMLStreamReader reader = xif.createXMLStreamReader(fis);

        StAXOMBuilder builder = new StAXOMBuilder(reader);
        OMElement elem = builder.getDocumentElement();

        param.setExtraElement(elem);
        req.setRequest(param);

        WorkorderServiceStub.Response4 res = stub.submitWorkorder(req);
        System.out.println(
              res.getResponse().getExtraElement().toString());
    } catch (Exception e) {
        e.printStackTrace();
    }
}
```

4. For example to check on Workorder status.

```
public void checkOnWorkorder () throws Exception
{
   try
   {
         WorkorderServiceStub stub = new WorkorderServiceStub();

         WorkorderServiceStub.ResponseCheck1 req = new
               WorkorderServiceStub.ResponseCheck1();
         WorkorderServiceStub.ResponseCheck param = new
               WorkorderServiceStub.ResponseCheck();

         OMFactory fac = OMAbstractFactory.getOMFactory();
         OMNamespace omNs =
               fac.createOMNamespace("http://checkresponse.nlps.dolby.com",
                 "pwcheckresponse");
         OMElement elem = fac.createOMElement("pwcheckresponse", omNs);
         OMElement elem2 = fac.createOMElement("correlationID", omNs);
         elem2.addChild(fac.createOMText(elem2,
                 "2feb1e3e-61e5-4aac-b1a8-eb77718177dc"));
         elem.addChild(elem2);

         param.setExtraElement(elem);
         req.setResponseCheck(param);

         WorkorderServiceStub.Response4 res = stub.checkOnWorkorder(req);

         System.out.println(res.getResponse().getExtraElement().
               toString());
   }
```

```
      catch (Exception e)
      {
          System.out.println("exception in checkOnWorkorder method"+e);
      }
    }
```

5. **For example to get workorders.**

```
public static void getWorkorders(WorkorderServiceStub stub) {
        try {
            WorkorderServiceStub.WorkorderRequest5 req =
                  new WorkorderServiceStub.WorkorderRequest5();
            WorkorderServiceStub.WorkorderRequest param =
                  new WorkorderServiceStub.WorkorderRequest();

            OMFactory fac = OMAbstractFactory.getOMFactory();
            OMNamespace omNs = fac.createOMNamespace(
                  "http://getworkorder.nlps.dolby.com", "ns1");
            OMElement elem = fac.createOMElement("pgetWorkorder", omNs);
            OMElement elem2 = fac.createOMElement("workorderID", omNs);
            elem2.addChild(fac.createOMText(elem2,
                    "058e260d-1ee0-4113-9ac7-56ca52668076"));
            elem.addChild(elem2);

            param.setExtraElement(elem);
            req.setWorkorderRequest(param);

            WorkorderServiceStub.WorkorderList3 res = stub.getWorkorders(req);

            System.out.println(res.getWorkorderList().getExtraElement().
                  toString());
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
```

6. **For example to get reports.**

```
public static void getReports(WorkorderServiceStub stub) {
        try {
            WorkorderServiceStub.ReportRequest6 req =
                  new WorkorderServiceStub.ReportRequest6();
            WorkorderServiceStub.ReportRequest param =
                  new WorkorderServiceStub.ReportRequest();

            OMFactory fac = OMAbstractFactory.getOMFactory();
            OMNamespace omNs = fac.createOMNamespace(
                  "http://getreport.nlps.dolby.com", "ns1");
            OMElement elem = fac.createOMElement("pgetReport", omNs);
            OMElement elem2 = fac.createOMElement("workorderID", omNs);
            elem2.addChild(fac.createOMText(elem2,
                    "cae3b638-0472-447d-b690-d61df224f7ca"));
            elem.addChild(elem2);

            param.setExtraElement(elem);
            req.setReportRequest(param);
```

```
            WorkorderServiceStub.ReportList8 res = stub.getReports(req);

            System.out.println(res.getReportList().getExtraElement().
                  toString());
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
```

7. **For example to delete Workorders.**

```
public static void deleteWorkorder(WorkorderServiceStub stub) {
        try {
            WorkorderServiceStub.DeleteWorkorderRequest7 req =
                  new WorkorderServiceStub.DeleteWorkorderRequest7();
            WorkorderServiceStub.DeleteWorkorderRequest param =
                  new WorkorderServiceStub.DeleteWorkorderRequest();

            OMFactory fac = OMAbstractFactory.getOMFactory();
            OMNamespace omNs = fac.createOMNamespace(
                  "http://getworkorder.nlps.dolby.com", "ns1");
            OMElement elem = fac.createOMElement("pgetWorkorder", omNs);
            OMElement elem2 = fac.createOMElement("workorderID", omNs);
            elem2.addChild(fac.createOMText(elem2,
                    "cae3b638-0472-447d-b690-d61df224f7ca"));
            elem.addChild(elem2);

            param.setExtraElement(elem);
            req.setDeleteWorkorderRequest(param);

            WorkorderServiceStub.Response4 res = stub.deleteWorkorder(req);

            System.out.println(res.getResponse().getExtraElement().
            toString());
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
```

1.3 .Net (C#)

This .Net example assumes the reader is somewhat familiar with WCF web service stack. If not, refer to http://msdn2.microsoft.com/en-us/netframework/aa663324.aspx and/or http://wcf.netfx3.com/ . In addition, refer to the service messaging discussed above for XML document formatting.

1. **For example use the .Net command line tool or Visual Studio .Net to generate the interface from the DP600 WSDL**
   http://*dp600hostname*:80/WorkorderService/WorkorderWsImpl?wsdl

Note: dp600hostname is the name given to the DP600 at the time of commissioning. Furthermore, client implementations are varied so the brief example below only

illustrates the necessary web service calls to list Workflow profiles, submit Workorders, and check on Workorder status.

2. **For example to list Workflow profiles, submit Workorders, check on Workorder status, get Workorders, get reports and delete Workorders.**

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Xml;

using ConsoleApplication1.eng.dp600hostname;
namespace ConsoleApplication1
{
    class Program
    {
        static void Main(string[] args)

        {
            // Next part is to invoke the xsd:any based web service
            WorkorderService service = new WorkorderService();
            Console.WriteLine("Invoking JAX-WS Client on " + service.Url);
            Console.WriteLine("calling listWorkflowProfiles ....");
            XmlDocument doc = new XmlDocument();
            doc.Load("getProfile.xml");
            XmlElement request = doc.DocumentElement;

            // Send the request
            XmlElement status = service.listWorkflowProfiles(request);

            // Print the response
            XmlTextWriter writer = new XmlTextWriter(Console.Out);
            writer.Formatting = Formatting.Indented;
            status.WriteTo(writer);

            // submit workorder
            Console.WriteLine("\n\ncalling submitWorkorder ....");
            doc = new XmlDocument();
            doc.Load("workorder.xml");
            request = doc.DocumentElement;
            status = service.submitWorkorder(request);
            status.WriteTo(writer);

            // check on workorder status
            Console.WriteLine("\n\ncalling checkOnWorkorder ....");
            doc = new XmlDocument();
            doc.Load("checkWorkorder.xml");
            request = doc.DocumentElement;
            status = service.checkOnWorkorder(request);
            status.WriteTo(writer);

            // get workorder reports
            Console.WriteLine("\n\ncalling getReports ...");
            doc = new XmlDocument();
            doc.Load("getReports.xml");
            request = doc.DocumentElement;
            status = service.getReports(request);
            status.WriteTo(writer);
```

```
            Console.ReadLine();

            // get workorders
            Console.WriteLine("\n\ncalling getWorkorders ...");
            doc = new XmlDocument();
            doc.Load("getWorkorder.xml");
            request = doc.DocumentElement;
            status = service.getWorkorders(request);
            status.WriteTo(writer);

            // delete workorders
            Console.WriteLine("\n\ncalling deleteWorkorder ...");
            doc = new XmlDocument();
            doc.Load("getWorkorder.xml");
            request = doc.DocumentElement;
            status = service.deleteWorkorder(request);
            status.WriteTo(writer);

        }
    }
```

1.4 .Net (Visual basic)

Sample codes:

```
Imports ConsoleApplication1.eng.orochi
Imports System.Xml

Module Module1

    Sub Main()

        Dim service As New WorkorderService()
        Dim request, status As XmlElement
        Dim xmlDoc As New XmlDocument

        Console.WriteLine("calling listWorkflowProfiles ....")
        xmlDoc.Load("getProfile.xml")
        request = xmlDoc.DocumentElement
        status = service.listWorkflowProfiles(request)
        Console.WriteLine(status.OuterXml)

        Console.WriteLine("calling submitWorkorder ....")
        xmlDoc.Load("workorder.xml")
        request = xmlDoc.DocumentElement
        status = service.submitWorkorder(request)
        Console.WriteLine(status.OuterXml)

        Console.WriteLine("calling checkOnWorkorder ....")
        xmlDoc.Load("checkWorkorder.xml")
        request = xmlDoc.DocumentElement
        status = service.checkOnWorkorder(request)
        Console.WriteLine(status.OuterXml)

        Console.WriteLine("calling getReports ....")
        xmlDoc.Load("getReport.xml")
        request = xmlDoc.DocumentElement
        status = service.getReports(request)
        Console.WriteLine(status.OuterXml)
```

```
        Console.WriteLine("calling getWorkorders ....")
        xmlDoc.Load("getWorkorder.xml")
        request = xmlDoc.DocumentElement
        status = service.getWorkorders(request)
        Console.WriteLine(status.OuterXml)

        Console.WriteLine("calling deleteWorkorder ....")
        xmlDoc.Load("getWorkorder.xml")
        request = xmlDoc.DocumentElement
        status = service.deleteWorkorder(request)
        Console.WriteLine(status.OuterXml)

        Console.ReadLine()

    End Sub

End Module
```

1.5 Delphi

This Delphi example assumes the reader is somewhat familiar with Delphi web service stack. If not, refer to http://www.codegear.com/ . In addition, refer to the service messaging discussed above for XML document formatting.

1. **For example use the Code Gear RAD Studio to generate the interface from the DP600 WSDL** http://*dp600hostname*:80/WorkorderService/WorkorderWsImpl?wsdl

   It's imperative that all of the auto-generated TRemotable data types be changed to TXMLData. The reason for this is that there is a known issue with the RAD studio wizard in dealing with the xsd:any type. Code Gear is aware of the issue.

   Note: dp600hostname is the name given to the DP600 at the time of commissioning. Furthermore, client implementations are varied so the brief example below only illustrates the necessary web service calls to list Workflow profiles, submit Workorders, and check on Workorder status.

2. **For example to list Workflow profiles, submit Workorders, and check on Workorder status.**

```
// ************************************************************************ //
// The types declared in this file were generated from data read from the
// WSDL File described below:
// WSDL     : http://dp600hostname/WorkorderService/WorkorderWsImpl?WSDL
//  >Import : http://dp600hostname/WorkorderService/WorkorderWsImpl?WSDL:0
// Encoding : UTF-8
// Version  : 1.0
// (11/5/2007 3:00:09 PM - - $Rev: 7300 $)
// ************************************************************************ //
```

```
unit WorkorderWsImpl1;
interface
uses InvokeRegistry, SOAPHTTPClient, Types, XSBuiltIns;
type

  ResponseCheck        = class;                 {
"http://dolby.nlps.com"[Lit][GblCplx] }

  Response             = class;                 {
"http://dolby.nlps.com"[Lit][GblCplx] }

  Request              = class;                 {
"http://dolby.nlps.com"[Lit][GblCplx] }

  profileListRequest   = class;                 {
"http://dolby.nlps.com"[Lit][GblCplx] }

  WorkflowProfileList  = class;                 {
"http://dolby.nlps.com"[Lit][GblCplx] }

  ResponseCheck2       = class;                 {
"http://dolby.nlps.com"[Lit][GblElm] }

  Response2            = class;                 {
"http://dolby.nlps.com"[Lit][GblElm] }

  Request2             = class;                 {
"http://dolby.nlps.com"[Lit][GblElm] }

  ProfileListRequest2  = class;                 {
"http://dolby.nlps.com"[Lit][GblElm] }

  WorkflowProfileList2 = class;                 {
"http://dolby.nlps.com"[Lit][GblElm] }


// ************************************************************************
//
// XML       : ResponseCheck, global, <complexType>
// Namespace : http://dolby.nlps.com
// Serializtn: [xoLiteralParam]
// Info      : Wrapper
// ************************************************************************
//

  ResponseCheck = class(TXMLData)
  private
  public
  constructor Create; override;
  published
  end;


// ************************************************************************
//
// XML       : Response, global, <complexType>
// Namespace : http://dolby.nlps.com
// Serializtn: [xoLiteralParam]
// Info      : Wrapper
// ************************************************************************
```

```
//

  Response = class(TXMLData)
  private
  public
  constructor Create; override;
  published
  end;


// ************************************************************************
//
// XML       : Request, global, <complexType>
// Namespace : http://dolby.nlps.com
// Serializtn: [xoLiteralParam]
// Info      : Wrapper
// ************************************************************************
//

  Request = class(TXMLData)
  private
  public
  constructor Create; override;
  published
  end;

// ************************************************************************
//
// XML       : profileListRequest, global, <complexType>
// Namespace : http://dolby.nlps.com
// Serializtn: [xoLiteralParam]
// Info      : Wrapper
// ************************************************************************
//

  profileListRequest = class(TXMLData)

  private
  public
  constructor Create; override;
  published
  end;


// ************************************************************************
//
// XML       : WorkflowProfileList, global, <complexType>
// Namespace : http://dolby.nlps.com
// Serializtn: [xoLiteralParam]
// Info      : Wrapper
// ************************************************************************
//

  WorkflowProfileList = class(TXMLData)
  private
  public
  constructor Create; override;
  published
  end;
```

```
// ************************************************************************
//
// XML       : ResponseCheck, global, <element>
// Namespace : http://dolby.nlps.com
// Info      : Wrapper
// ************************************************************************
//

  ResponseCheck2 = class(ResponseCheck)
  private
  published
  end;


// ************************************************************************
//
// XML       : Response, global, <element>
// Namespace : http://dolby.nlps.com
// Info      : Wrapper
// ************************************************************************
//

  Response2 = class(Response)
  private
  published
  end;

// ************************************************************************
//
// XML       : Request, global, <element>
// Namespace : http://dolby.nlps.com
// Info      : Wrapper
// ************************************************************************
//

  Request2 = class(Request)
  private
  published
  end;


// ************************************************************************
//
// XML       : ProfileListRequest, global, <element>
// Namespace : http://dolby.nlps.com
// Info      : Wrapper
// ************************************************************************
//

  ProfileListRequest2 = class(profileListRequest)
  private
  published
  end;


// ************************************************************************
//
// XML       : WorkflowProfileList, global, <element>
```

```
// Namespace : http://dolby.nlps.com
// Info      : Wrapper
// ************************************************************************
//

  WorkflowProfileList2 = class(WorkflowProfileList)
  private
  published
  end;

// ************************************************************************
//
// Namespace : http://dolby.nlps.com
// transport : http://schemas.xmlsoap.org/soap/http
// style     : document
// binding   : WorkorderBinding
// service   : WorkorderService
// port      : WorkorderPort
// URL       : http://dp600hostname:80/WorkorderService/WorkorderWsImpl
// ************************************************************************
//

  WorkorderPortType = interface(IInvokable)
  ['{8057BB02-647B-ECED-7531-6D442D593A85}']
  // Cannot unwrap:
  //     - Input element wrapper name does not match operation's name
  function  listWorkflowProfiles(const profileListRequest:
ProfileListRequest2): WorkflowProfileList2; stdcall;



    // Cannot unwrap:
    // - Input element wrapper name does not match operation's name
    function  submitWorkorder(const request: Request2): Response2; stdcall;
    // Cannot unwrap:
    //     - Input element wrapper name does not match operation's name
    function  checkOnWorkorder(const check: ResponseCheck2): Response2;
stdcall;
  end;

function GetWorkorderPortType(UseWSDL: Boolean=System.False; Addr: string='';
HTTPRIO: THTTPRIO = nil): WorkorderPortType;
implementation
  uses SysUtils;

function GetWorkorderPortType(UseWSDL: Boolean; Addr: string; HTTPRIO:
THTTPRIO): WorkorderPortType;

const
  defWSDL = 'http://dp600hostname/WorkorderService/WorkorderWsImpl?WSDL';
  defURL  = 'http://dp600hostname:80/WorkorderService/WorkorderWsImpl';
  defSvc  = 'WorkorderService';
  defPrt  = 'WorkorderPort';

var

  RIO: THTTPRIO;

begin
```

```
  Result := nil;
  if (Addr = '') then
  begin
    if UseWSDL then
      Addr := defWSDL
    else
      Addr := defURL;
  end;

  if HTTPRIO = nil then
    RIO := THTTPRIO.Create(nil)
  else
    RIO := HTTPRIO;

  try
    Result := (RIO as WorkorderPortType);
    if UseWSDL then
    begin
      RIO.WSDLLocation := Addr;
      RIO.Service := defSvc;
      RIO.Port := defPrt;
    end else

      RIO.URL := Addr;

  finally

    if (Result = nil) and (HTTPRIO = nil) then

      RIO.Free;
  end;

end;

constructor ResponseCheck.Create;

begin
  inherited Create;
  FSerializationOptions := [xoLiteralParam];
end;

constructor Response.Create;
begin
  inherited Create;
  FSerializationOptions := [xoLiteralParam];
end;

constructor Request.Create;
begin
  inherited Create;
  FSerializationOptions := [xoLiteralParam];
end;

constructor profileListRequest.Create;
begin
  inherited Create;
  FSerializationOptions := [xoLiteralParam];
end;
```

```
constructor WorkflowProfileList.Create;
begin
  inherited Create;
  FSerializationOptions := [xoLiteralParam];
end;

initialization

  InvRegistry.RegisterInterface(TypeInfo(WorkorderPortType),
'http://dolby.nlps.com', 'UTF-8');

  InvRegistry.RegisterDefaultSOAPAction(TypeInfo(WorkorderPortType), '');

  InvRegistry.RegisterInvokeOptions(TypeInfo(WorkorderPortType), ioDocument);

  InvRegistry.RegisterInvokeOptions(TypeInfo(WorkorderPortType), ioLiteral);

  RemClassRegistry.RegisterXSClass(ResponseCheck, 'http://dolby.nlps.com',
'ResponseCheck');

  RemClassRegistry.RegisterSerializeOptions(ResponseCheck, [xoLiteralParam]);

  RemClassRegistry.RegisterXSClass(Response, 'http://dolby.nlps.com',
'Response');

  RemClassRegistry.RegisterSerializeOptions(Response, [xoLiteralParam]);

  RemClassRegistry.RegisterXSClass(Request, 'http://dolby.nlps.com',
'Request');

  RemClassRegistry.RegisterSerializeOptions(Request, [xoLiteralParam]);

  RemClassRegistry.RegisterXSClass(profileListRequest,
'http://dolby.nlps.com', 'profileListRequest');

  RemClassRegistry.RegisterSerializeOptions(profileListRequest,
[xoLiteralParam]);

  RemClassRegistry.RegisterXSClass(WorkflowProfileList,
'http://dolby.nlps.com', 'WorkflowProfileList');

  RemClassRegistry.RegisterSerializeOptions(WorkflowProfileList,
[xoLiteralParam]);

  RemClassRegistry.RegisterXSClass(ResponseCheck2, 'http://dolby.nlps.com',
'ResponseCheck2', 'ResponseCheck');

  RemClassRegistry.RegisterXSClass(Response2, 'http://dolby.nlps.com',
'Response2', 'Response');

  RemClassRegistry.RegisterXSClass(Request2, 'http://dolby.nlps.com',
'Request2', 'Request');

  RemClassRegistry.RegisterXSClass(ProfileListRequest2,
'http://dolby.nlps.com', 'ProfileListRequest2', 'ProfileListRequest');

  RemClassRegistry.RegisterXSClass(WorkflowProfileList2,
'http://dolby.nlps.com', 'WorkflowProfileList2', 'WorkflowProfileList');

end.
```

```
program dp600ws;
{$APPTYPE CONSOLE}

uses
  SysUtils,
  Forms,
  XMLDoc,
  WorkorderWsImpl1 in 'WorkorderWsImpl1.pas';
var
   service:WorkorderPortType;
   request:profileListRequest2;
   response:WorkflowProfileList2;
   woRequest:Request2;
   woResponse:Response2;
   checkRequest:ResponseCheck2;
   xmlDoc:TXMLDocument;
begin

  try
    { TODO -oUser -cConsole Main : Insert code here }
    Application.Initialize;
    service := GetWorkorderPortType;
    request := profileListRequest2.Create;
    response := service.listWorkflowProfiles(request);
    writeln( 'after listWorkflowProfiles is called ...'+response.XMLNode.XML);
    woRequest := Request2.Create;
    xmlDoc := TXMLDocument.create(nil);
```

```
  xmlDoc.LoadFromFile('workorder.xml');
  woRequest.LoadFromXML(xmlDoc.XML.Text);
  woResponse := service.submitWorkorder(woRequest);
  writeln( 'after submitWorkorder is called ...'+woResponse.XMLNode.XML);
  checkRequest := ResponseCheck2.Create;
  xmlDoc := TXMLDocument.create(nil);
  xmlDoc.LoadFromFile('checkWorkorder.xml');
  checkRequest.LoadFromXML(xmlDoc.XML.Text);
  woResponse := service.checkOnWorkorder(checkRequest);
  writeln( 'after checkOnWorkorder is called ...'+woResponse.XMLNode.XML);
  Readln;
except
  on E:Exception do
    Writeln(E.Classname, ': ', E.Message);
end;
```