



| z/TPF V1.1

TPF Users Group Spring 2009

Accessing Web Services From Your z/TPF System

Name: Edwin van de Grift
Venue: Ongoing TPF Education

AIM Enterprise Platform Software
IBM z/Transaction Processing Facility Enterprise Edition 1.1.0

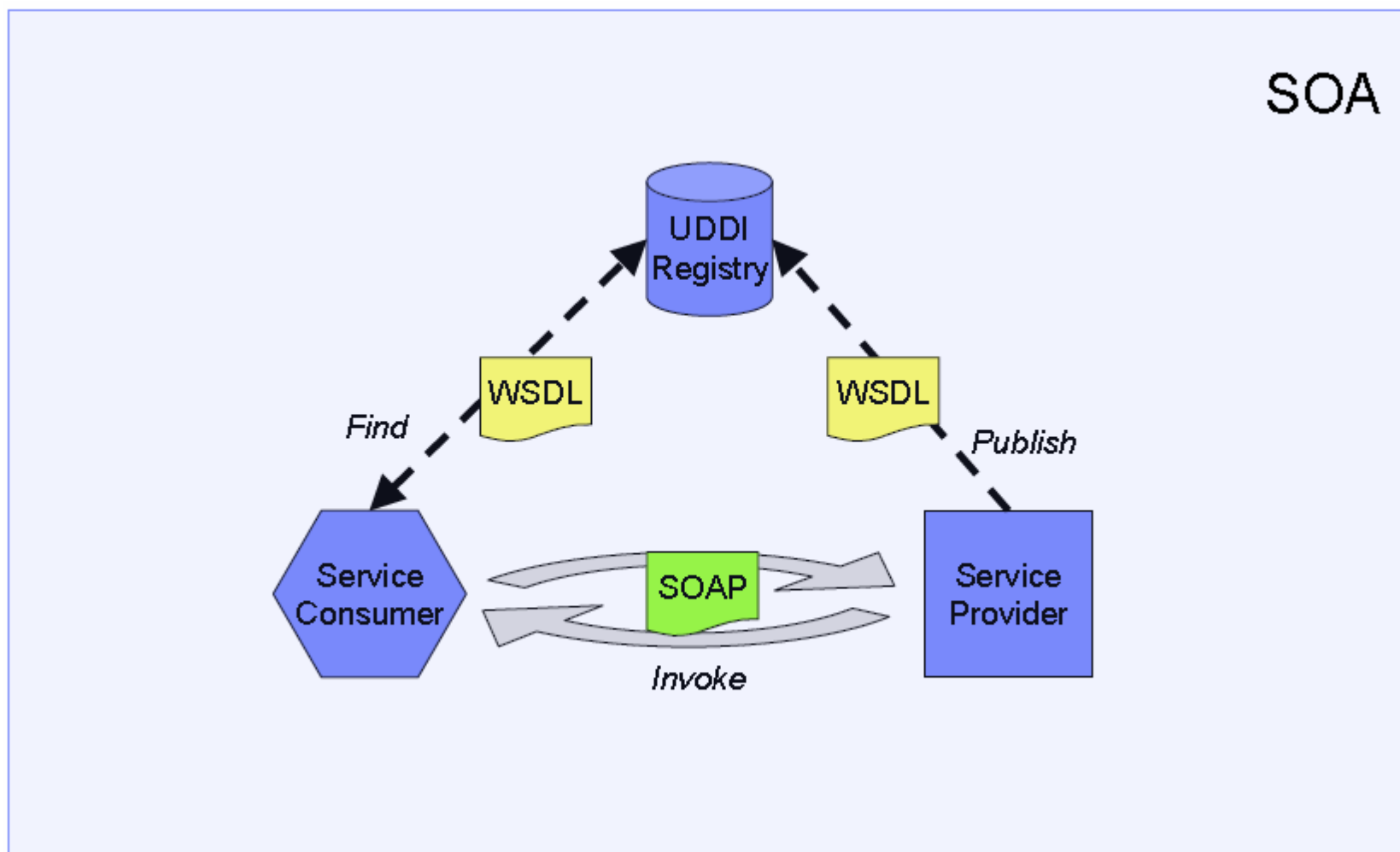
Any reference to future plans are for planning purposes only. IBM reserves the right to change those plans at its discretion. Any reliance on such a disclosure is solely at your own risk. IBM makes no commitment to provide additional information in the future.

Contents

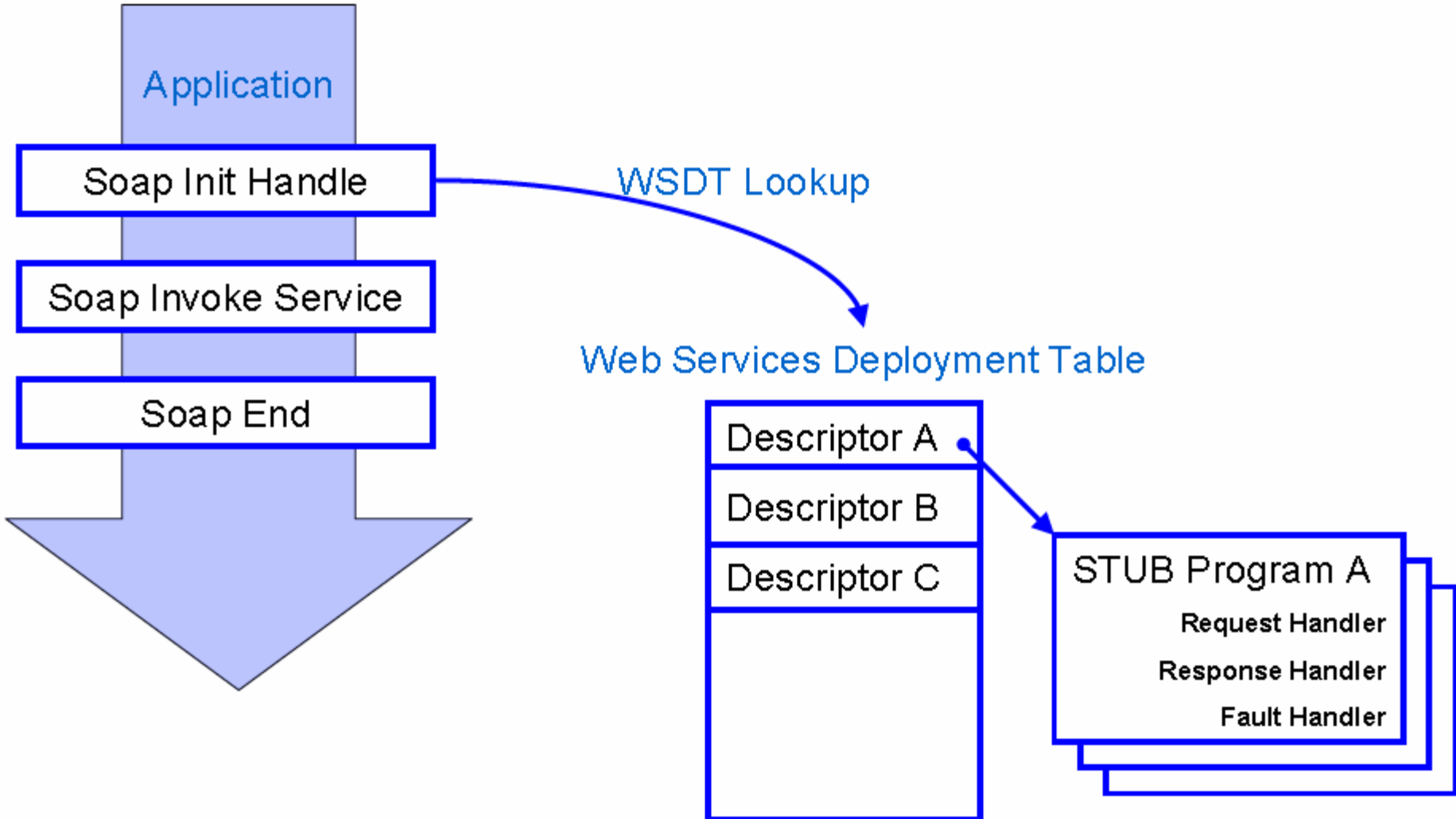
- **Web Services**
- **Soap Consumer Flow**
- **Accessing a Web Service from your z/TPF application**



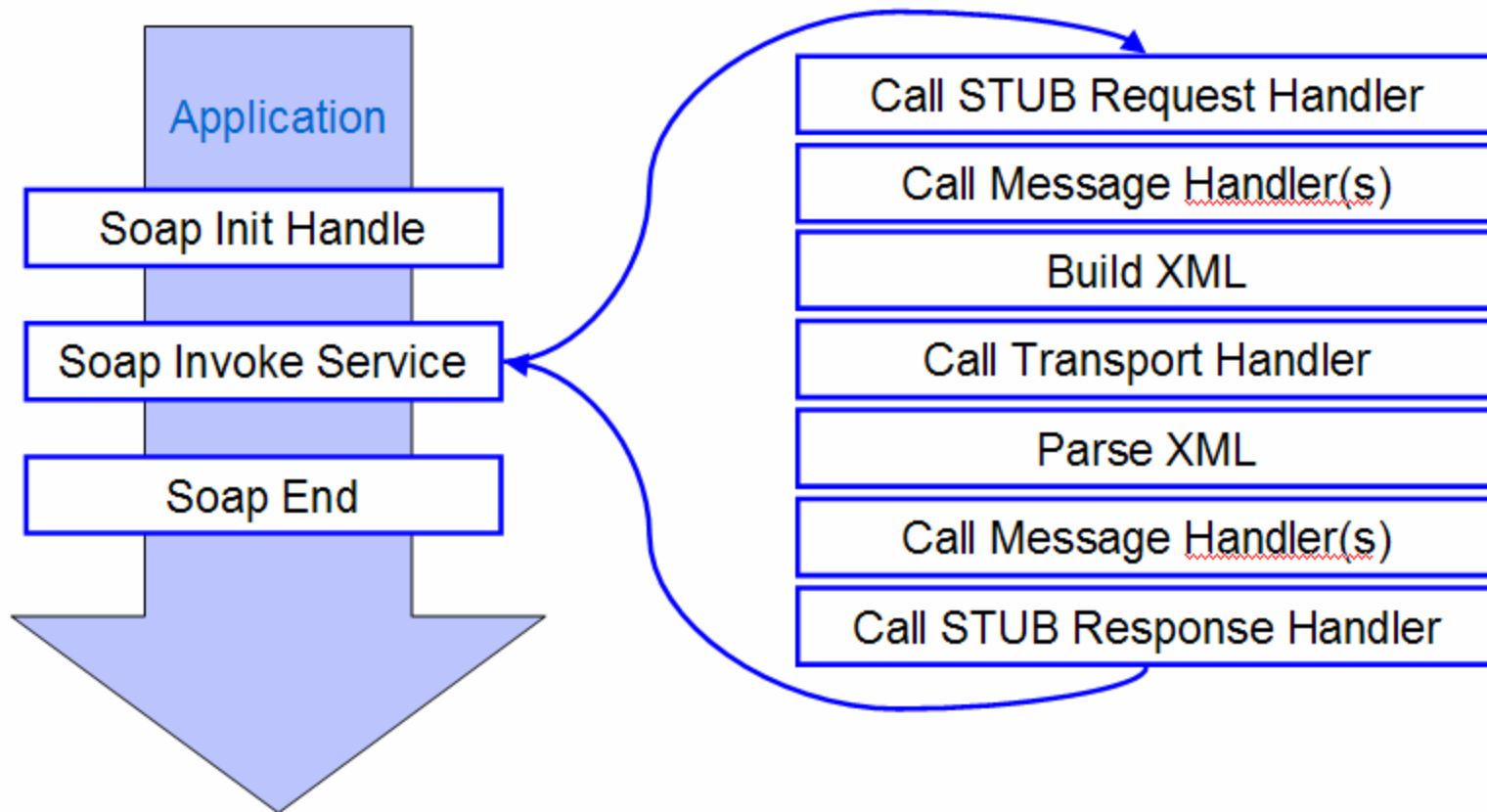
Web Services



Soap Consumer Flow



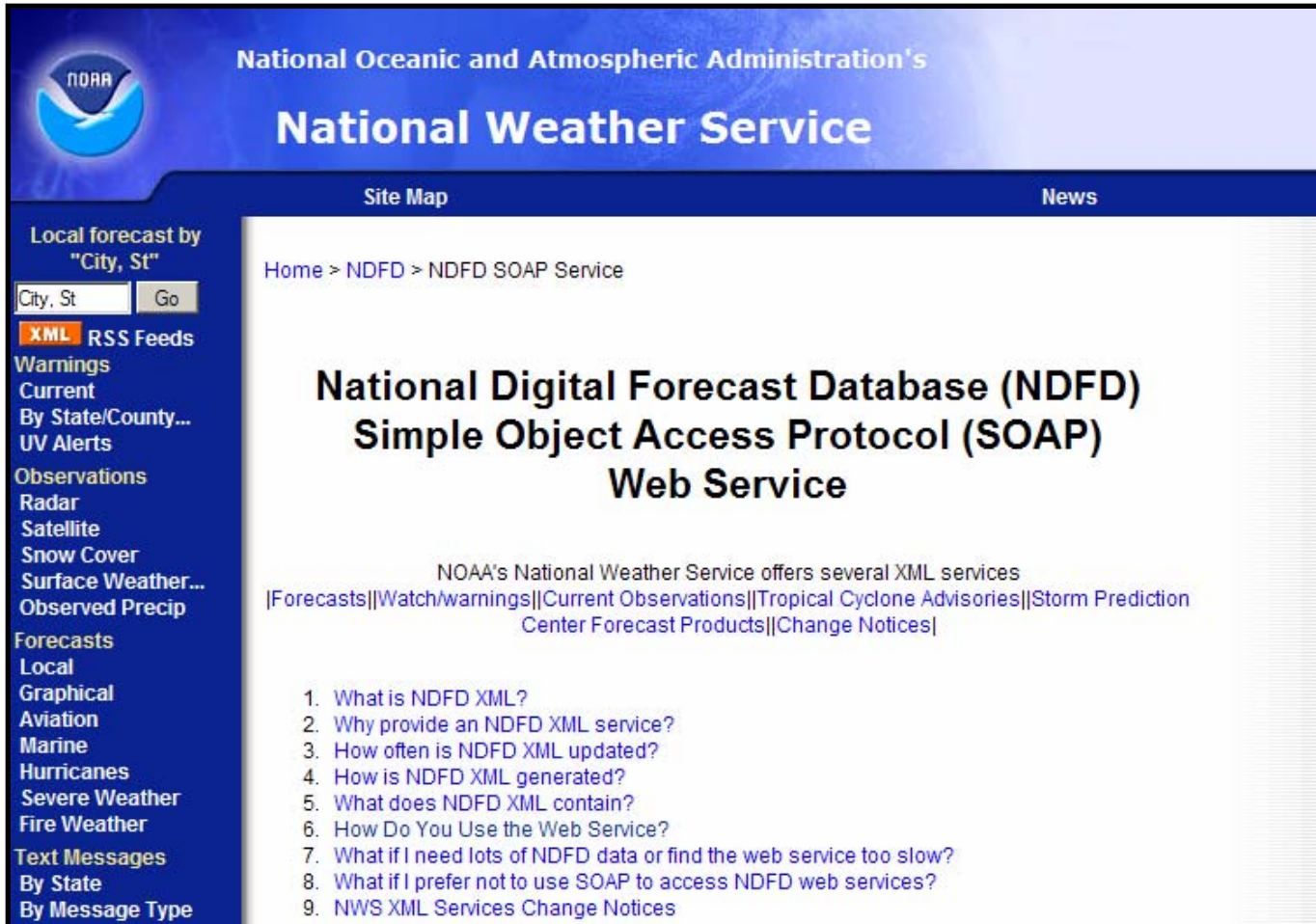
Soap Consumer Flow



Summary

- **Get the Service's WSDL**
- **Create a Deployment Descriptor**
- **Create STUB code**
- **Deploy the Descriptor**
- **Invoke the Web Service**

What We Will Consume



The screenshot shows the National Weather Service website. The header includes the NOAA logo and the text "National Oceanic and Atmospheric Administration's National Weather Service". Below the header, there are links for "Site Map" and "News". The main content area is titled "National Digital Forecast Database (NDFD) Simple Object Access Protocol (SOAP) Web Service". It includes a breadcrumb trail "Home > NDFD > NDFD SOAP Service" and a list of XML services: "Forecasts", "Watch/warnings", "Current Observations", "Tropical Cyclone Advisories", "Storm Prediction Center Forecast Products", and "Change Notices". A list of nine questions is provided, such as "What is NDFD XML?" and "Why provide an NDFD XML service?". On the left side, there is a navigation menu with options like "Local forecast by 'City, St'", "XML RSS Feeds", "Warnings", "Observations", "Forecasts", and "Text Messages".

National Oceanic and Atmospheric Administration's
National Weather Service

Site Map News

Local forecast by "City, St"
City, St Go

XML RSS Feeds
Warnings
Current
By State/County...
UV Alerts
Observations
Radar
Satellite
Snow Cover
Surface Weather...
Observed Precip
Forecasts
Local
Graphical
Aviation
Marine
Hurricanes
Severe Weather
Fire Weather
Text Messages
By State
By Message Type

Home > NDFD > NDFD SOAP Service

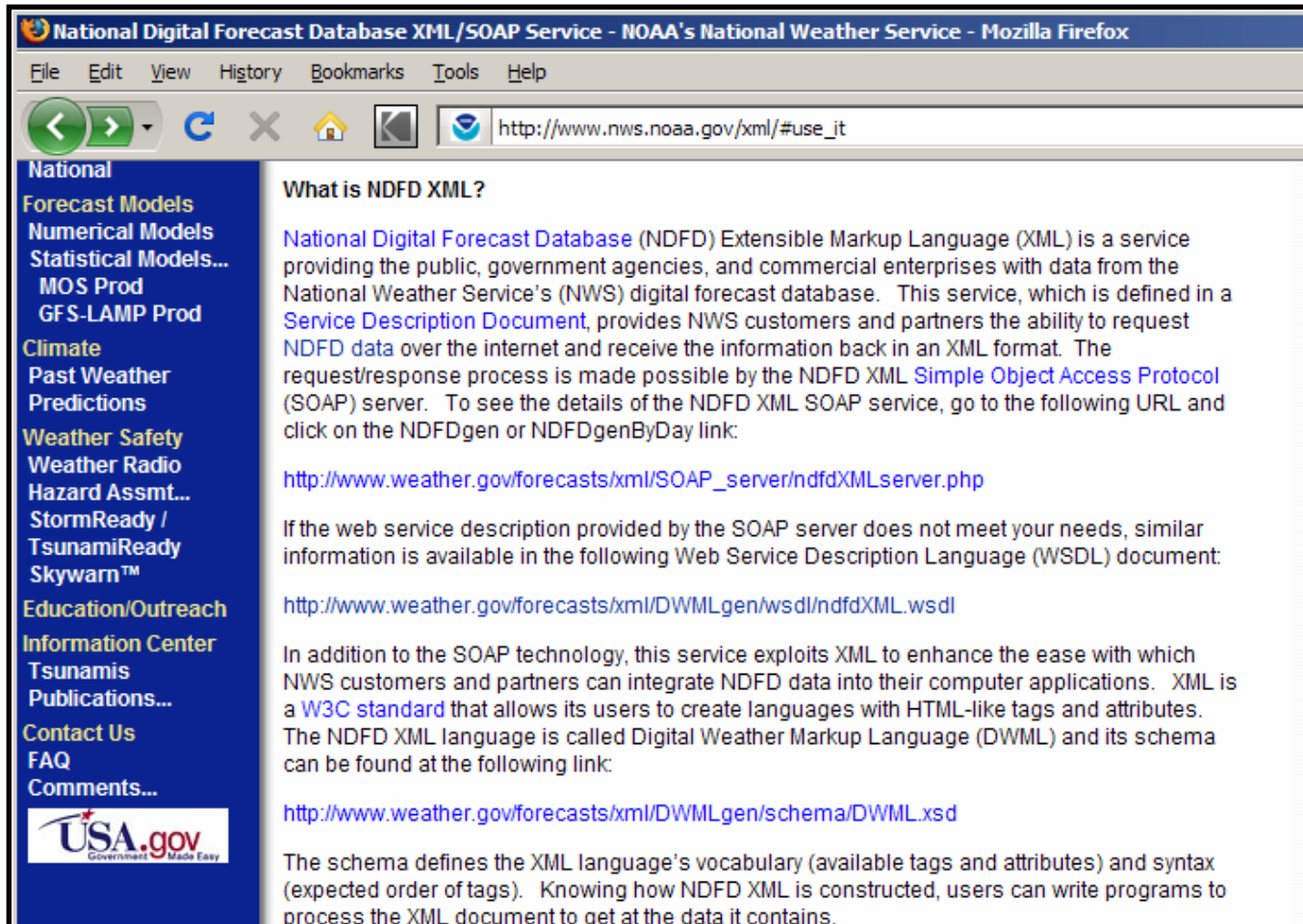
National Digital Forecast Database (NDFD) Simple Object Access Protocol (SOAP) Web Service

NOAA's National Weather Service offers several XML services
|Forecasts||Watch/warnings||Current Observations||Tropical Cyclone Advisories||Storm Prediction
Center Forecast Products||Change Notices|

1. What is NDFD XML?
2. Why provide an NDFD XML service?
3. How often is NDFD XML updated?
4. How is NDFD XML generated?
5. What does NDFD XML contain?
6. How Do You Use the Web Service?
7. What if I need lots of NDFD data or find the web service too slow?
8. What if I prefer not to use SOAP to access NDFD web services?
9. NWS XML Services Change Notices

The NWS material incorporated is not subject to copyright protection.

Download the WSDL




National Digital Forecast Database XML/SOAP Service - NOAA's National Weather Service - Mozilla Firefox

File Edit View History Bookmarks Tools Help

http://www.nws.noaa.gov/xml/#use_it

National

- Forecast Models
 - Numerical Models
 - Statistical Models...
 - MOS Prod
 - GFS-LAMP Prod
- Climate
 - Past Weather
 - Predictions
- Weather Safety
 - Weather Radio
 - Hazard Assmt...
 - StormReady /
 - TsunamiReady
 - Skywarn™
- Education/Outreach
- Information Center
 - Tsunamis
 - Publications...
- Contact Us
 - FAQ
 - Comments...



What is NDFD XML?

[National Digital Forecast Database](#) (NDFD) Extensible Markup Language (XML) is a service providing the public, government agencies, and commercial enterprises with data from the National Weather Service's (NWS) digital forecast database. This service, which is defined in a [Service Description Document](#), provides NWS customers and partners the ability to request NDFD data over the internet and receive the information back in an XML format. The request/response process is made possible by the NDFD XML [Simple Object Access Protocol](#) (SOAP) server. To see the details of the NDFD XML SOAP service, go to the following URL and click on the NDFDgen or NDFDgenByDay link:

http://www.weather.gov/forecasts/xml/SOAP_server/ndfdXMLserver.php

If the web service description provided by the SOAP server does not meet your needs, similar information is available in the following Web Service Description Language (WSDL) document:

<http://www.weather.gov/forecasts/xml/DWMLgen/wsd/ndfdXML.wsdl>

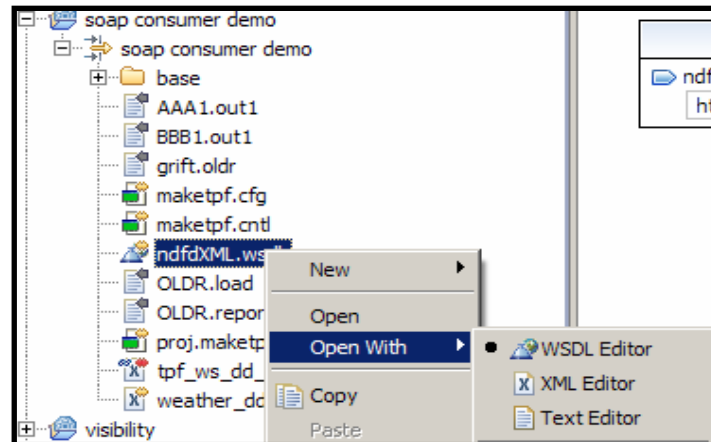
In addition to the SOAP technology, this service exploits XML to enhance the ease with which NWS customers and partners can integrate NDFD data into their computer applications. XML is a [W3C standard](#) that allows its users to create languages with HTML-like tags and attributes. The NDFD XML language is called Digital Weather Markup Language (DWML) and its schema can be found at the following link:

<http://www.weather.gov/forecasts/xml/DWMLgen/schema/DWML.xsd>

The schema defines the XML language's vocabulary (available tags and attributes) and syntax (expected order of tags). Knowing how NDFD XML is constructed, users can write programs to process the XML document to get at the data it contains.

The NWS material incorporated is not subject to copyright protection.

WSDL Management



WSDL Editor

The screenshot shows the WSDL Editor interface for a file named 'nfdXML.wsdl'. On the left, a tree view shows the 'nfdXML' namespace containing an 'nfdXMLPort' binding with the URL 'http://www.weather.gov...'. An arrow points from this binding to the main editor area, which displays the details for the 'nfdXMLPortType' port type.

The 'nfdXMLPortType' section is expanded to show two message types: 'NDFDgen' and 'NDFDgenByDay'. The 'NDFDgen' message is further divided into 'input' and 'output' sections. The 'input' section lists the following parameters:

latitude	decimal
longitude	decimal
product	productType
startTime	dateTime
endTime	dateTime
weatherParameters	weatherParametersType

The 'output' section lists:

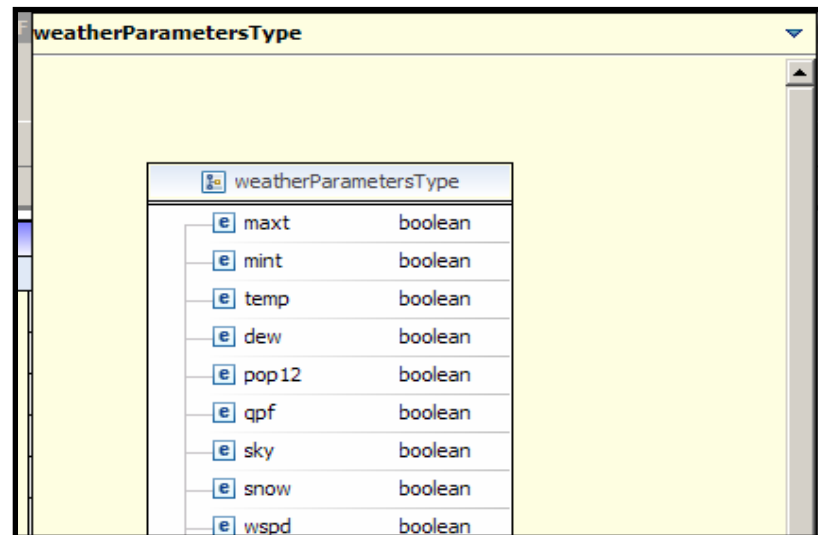
dwmlOut	string
---------	--------

The 'NDFDgenByDay' message lists:

latitude	decimal
longitude	decimal

Arrows on the right side of the parameter table indicate the direction of data flow. A red circle highlights the arrow pointing to the right from the 'weatherParameters' parameter.

WSDL Editor



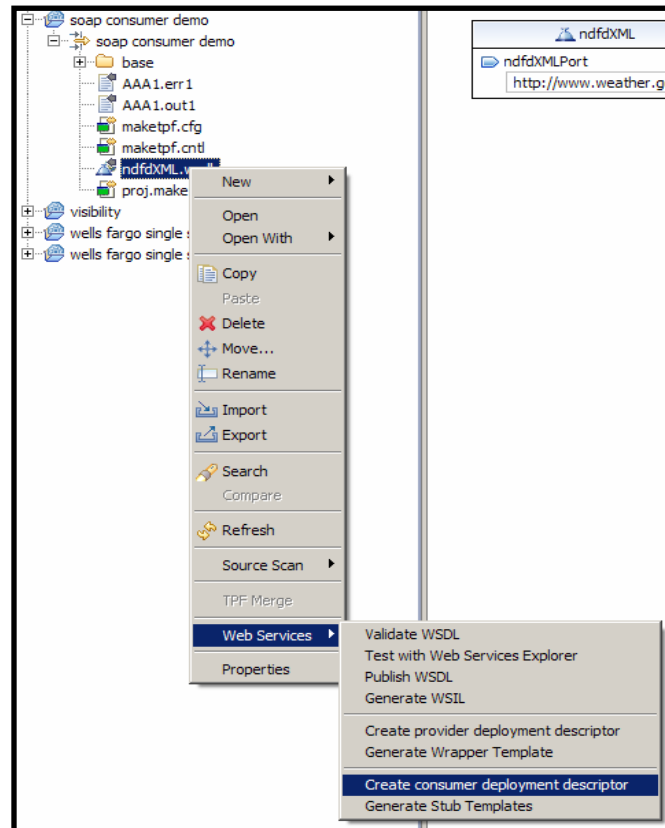
The screenshot shows a WSDL Editor window titled "weatherParametersType". Inside the editor, a schema definition is displayed for the type "weatherParametersType". The schema is a complex type containing several boolean elements, each with a blue "e" icon in a square to its left. The elements and their types are listed in a table below.

weatherParametersType	
<input type="checkbox"/> e maxt	boolean
<input type="checkbox"/> e mint	boolean
<input type="checkbox"/> e temp	boolean
<input type="checkbox"/> e dew	boolean
<input type="checkbox"/> e pop12	boolean
<input type="checkbox"/> e qpf	boolean
<input type="checkbox"/> e sky	boolean
<input type="checkbox"/> e snow	boolean
<input type="checkbox"/> e wspd	boolean

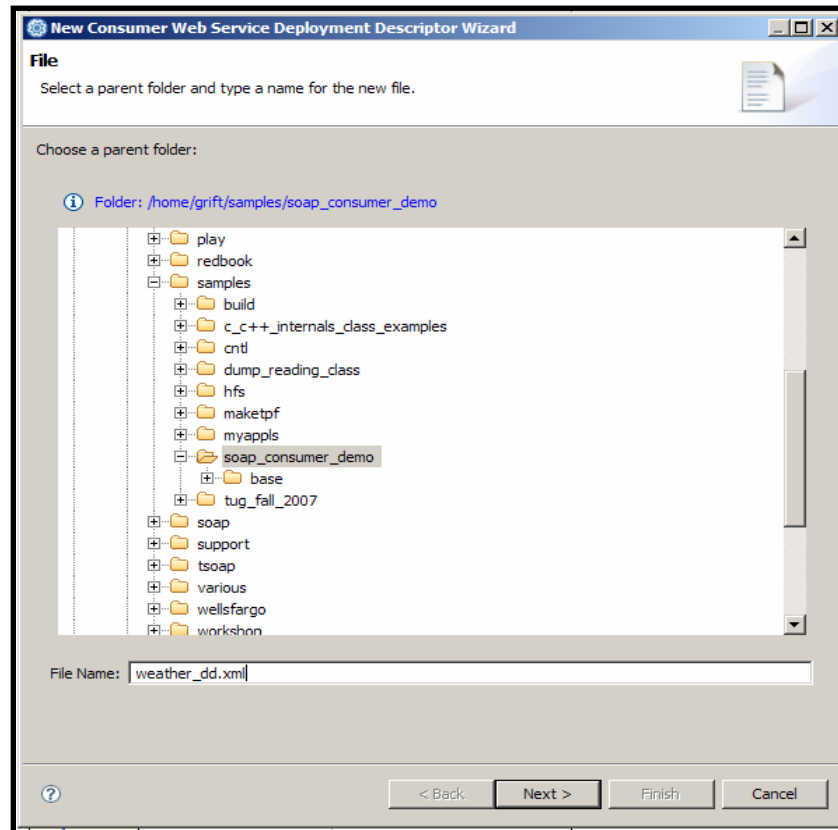
Summary

- **Get the Service's WSDL**
- **Create a Deployment Descriptor**
- **Create STUB code**
- **Deploy the Descriptor**
- **Invoke the Web Service**

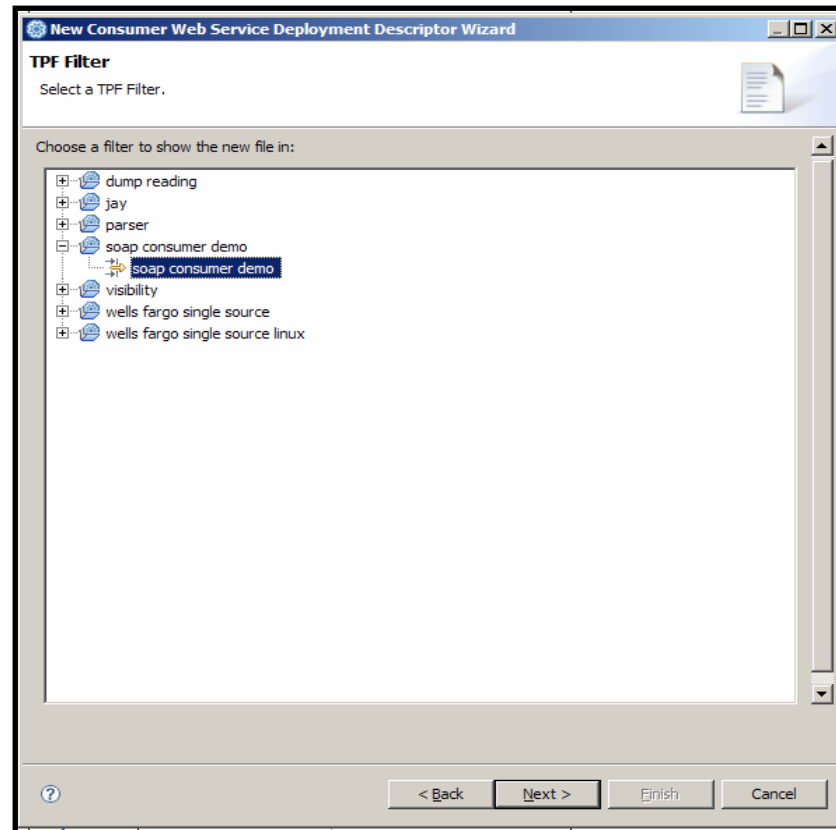
Create Consumer Deployment Descriptor



Name the Deployment Descriptor



Select a TPF Filter



Specify the STUB Name

New Consumer Web Service Deployment Descriptor Wizard

Deployment Descriptor

Specify at least one transport.

URI:

Stub:

Description:

Provider Code Page

Consumer Code Page

▶ **SOAP Message Handlers**

▶ **SOAP Versions**

▼ **Operations**

Specify the name of service operations exposed in this web service:

NDFDgenByDay
NDFDgenLatLonList
NDFDgenByDayLatLonList
GmlLatLonList
GmlTimeSeries

▼ **Transports**

Specify the applicable transports for this consumer Web service:

Transport	Value

Add a Transport

Add Consumer Web Service Transport

Add

Invalid URL. Specify a URL between 1 and 1024 characters. For example, http://site.domain.com/MyService.

Transport type: HTTP

URL:

Headers

Specify HTTP headers and values using the format header:value.

Content-Type: text/xml

Buttons: Add, Edit..., Remove, Up, Down

Buttons: OK, Cancel

Add a Transport

Add Consumer Web Service Transport

Add

Transport information

Transport type: HTTP

URL: `http://www.weather.gov/forecasts/xml/SOAP_server/ndfdXMLserver.php`

Headers

Specify HTTP headers and values using the format header:value.

	Add
Content-Type: text/xml	Edit...
	Remove
	Up
	Down

OK Cancel

Finish the Descriptor Wizard

New Consumer Web Service Deployment Descriptor Wizard

Deployment Descriptor
Specify consumer web service details

URI:

Stub:

Description:

Provider Code Page

Consumer Code Page

▶ **SOAP Message Handlers**

▶ **SOAP Versions**

▼ **Operations**
Specify the name of service operations exposed in this web service:

- NDFDgen
- NDFDgenByDay
- NDFDgenLatLonList
- NDFDgenByDayLatLonList
- GmlLatLonList
- GmlTimeSeries

Add Edit... Remove

▼ **Transports**
Specify the applicable transports for this consumer Web service:

Transport	Value
HTTP	http://www.weather.gov/forecasts/xml/SOAP_server/ndfdXMLse
Headers	Content-Type: text/xml
Header	
Header	
Header	

Add... Edit... Remove Move Up Move Down

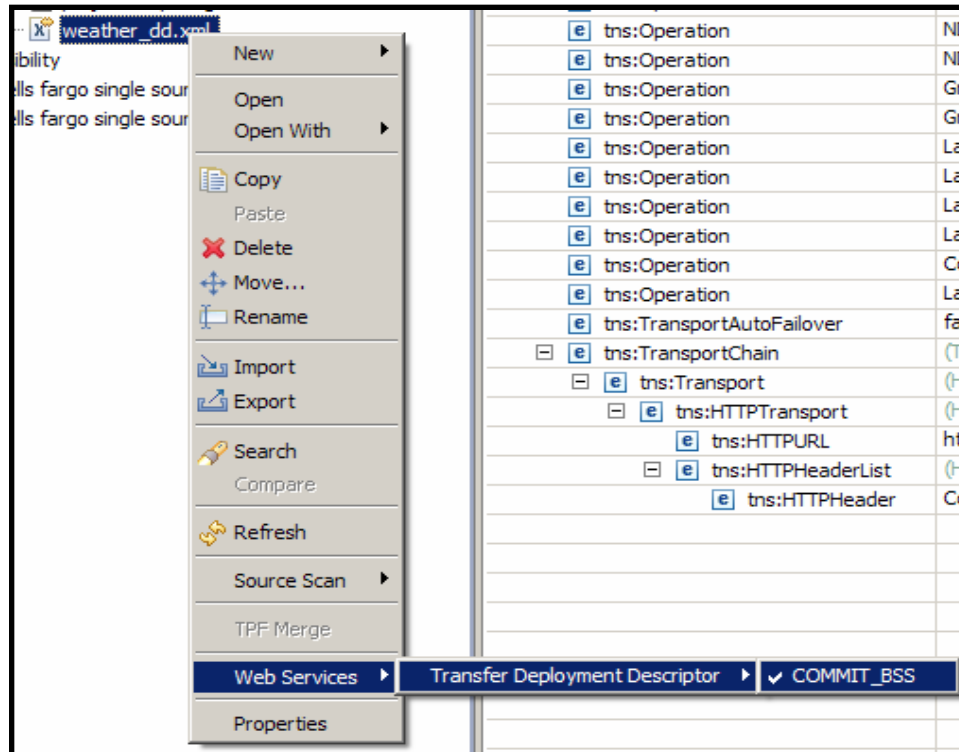
Auto Fail Over

? < Back Next > Finish Cancel

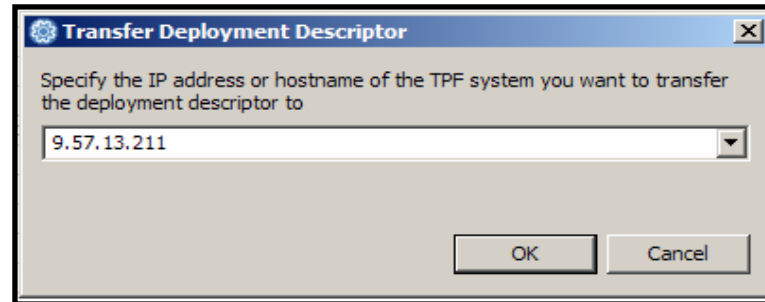
The Deployment Descriptor

File Name	Content
weather_dd.xml	version="1.0" encoding="UTF-8"
tns:ConsumerDeploymentDescriptor	(ServiceURI, Stub, SOAPProcessing, Operation*, Description?, TransportAutoFailover, Co
xmlns:tns	http://www.ibm.com/xmlns/prod/ztpf/soap
xmlns:xsi	http://www.w3.org/2001/XMLSchema-instance
xsi:schemaLocation	http://www.ibm.com/xmlns/prod/ztpf/soap file:/etc/tpf-ws/schema/tpf_ws_dd_v2.xsd
tns:ServiceURI	/forecasts/xml/SOAP_server/ndfdXMLserver.php
tns:Stub	BBB1
tns:SOAPProcessing	(SOAPMessageHandlerChain*, VerifySOAPHeaders, SOAPVersion*)
tns:VerifySOAPHeaders	false
tns:SOAPVersion	SOAP1.1
tns:Operation	NDFDgen
tns:Operation	NDFDgenByDay
tns:Operation	NDFDgenLatLonList
tns:Operation	NDFDgenByDayLatLonList
tns:Operation	GmlLatLonList
tns:Operation	GmlTimeSeries
tns:Operation	LatLonListSubgrid
tns:Operation	LatLonListLine
tns:Operation	LatLonListZipCode
tns:Operation	LatLonListSquare
tns:Operation	CornerPoints
tns:Operation	LatLonListCityNames
tns:TransportAutoFailover	false
tns:TransportChain	(Transport+)
tns:Transport	(HTTPTransport MQTransport TPFTTransport UserTransport)
tns:HTTPTransport	(HTTPURL, HTTPHeaderList?)
tns:HTTPURL	http://www.weather.gov/forecasts/xml/SOAP_server/ndfdXMLserver.php
tns:HTTPHeaderList	(HTTPHeader+)
tns:HTTPHeader	Content-Type: text/xml

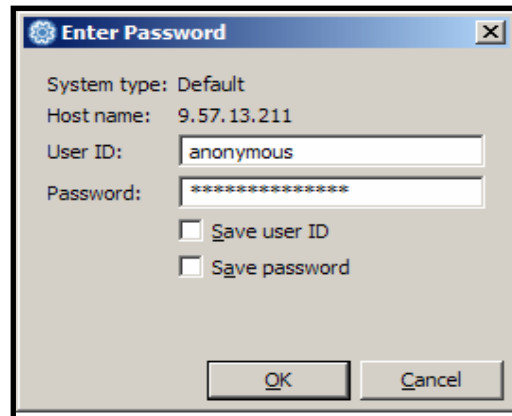
Transfer the Deployment Descriptor



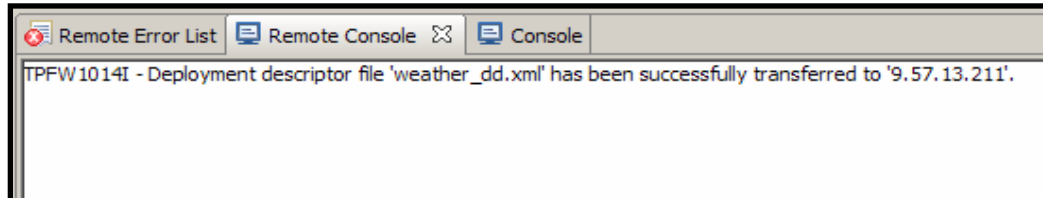
Specify the TPF System's Address



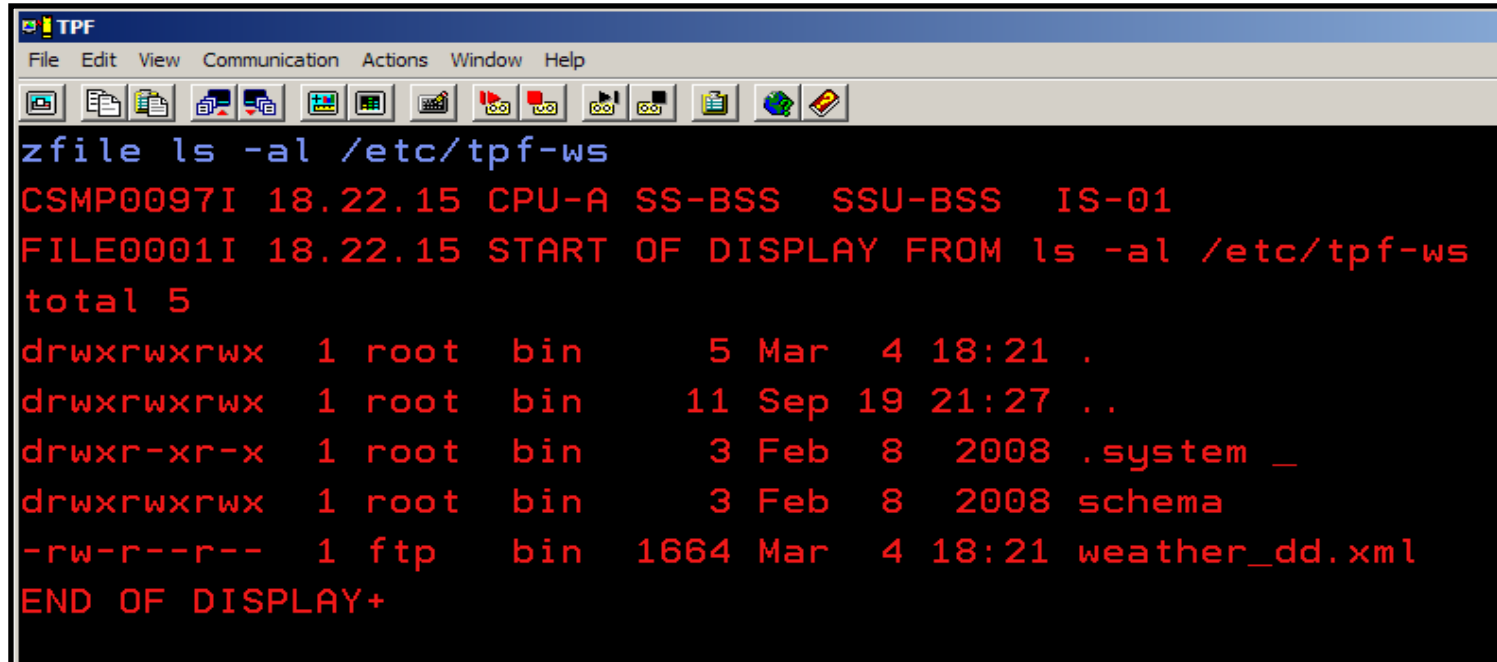
Authenticate with FTP



FTP Results

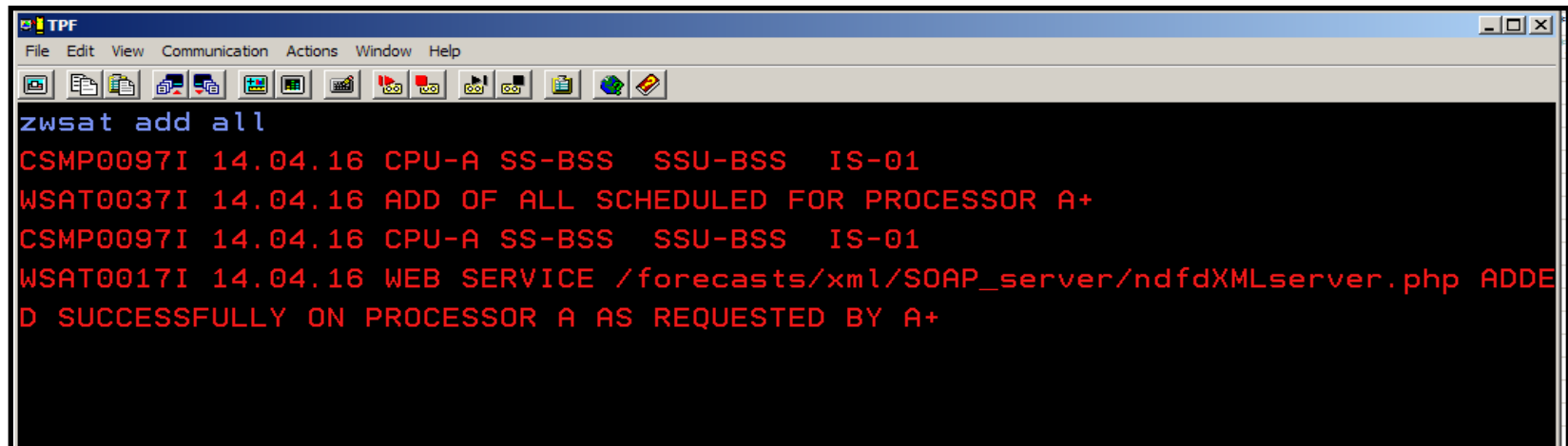


The Descriptor on TPF



```
zfile ls -al /etc/tpf-ws
CSMP0097I 18.22.15 CPU-A SS-BSS  SSU-BSS  IS-01
FILE0001I 18.22.15 START OF DISPLAY FROM ls -al /etc/tpf-ws
total 5
drwxrwxrwx  1 root  bin    5 Mar  4 18:21 .
drwxrwxrwx  1 root  bin   11 Sep 19 21:27 ..
drwxr-xr-x  1 root  bin    3 Feb  8 2008 .system _
drwxrwxrwx  1 root  bin    3 Feb  8 2008 schema
-rw-r--r--  1 ftp   bin  1664 Mar  4 18:21 weather_dd.xml
END OF DISPLAY+
```

Add the Descriptor



```
zwsat add all
CSMP0097I 14.04.16 CPU-A SS-BSS  SSU-BSS  IS-01
WSAT0037I 14.04.16 ADD OF ALL SCHEDULED FOR PROCESSOR A+
CSMP0097I 14.04.16 CPU-A SS-BSS  SSU-BSS  IS-01
WSAT0017I 14.04.16 WEB SERVICE /forecasts/xml/SOAP_server/ndfdXMLserver.php ADDED SUCCESSFULLY ON PROCESSOR A AS REQUESTED BY A+
```

Summary

- **Get the Service's WSDL**
- **Create a Deployment Descriptor**
- **Create STUB code**
- **Deploy the Descriptor**
- **Invoke the Web Service**

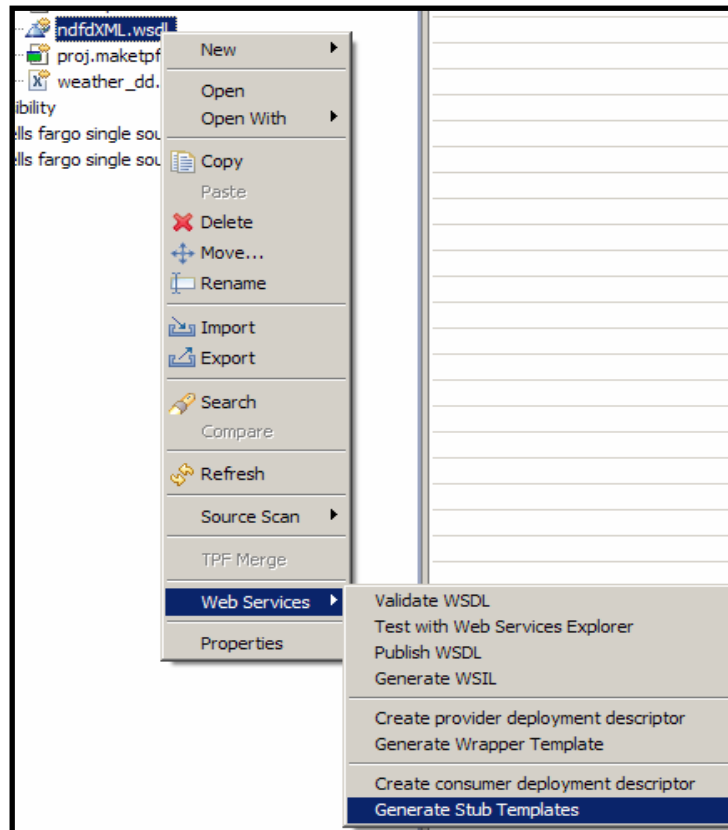
The STUB

- **An isolation point for the portion of code that:**
 - Serializes the operation and parameter data into the body portion of the SOAP request.
 - Deserializes the return from the service provider from the parsed SOAP response.
- **This hides the following details from the application:**
 - Message formats
 - Service locations
 - Communication protocol information

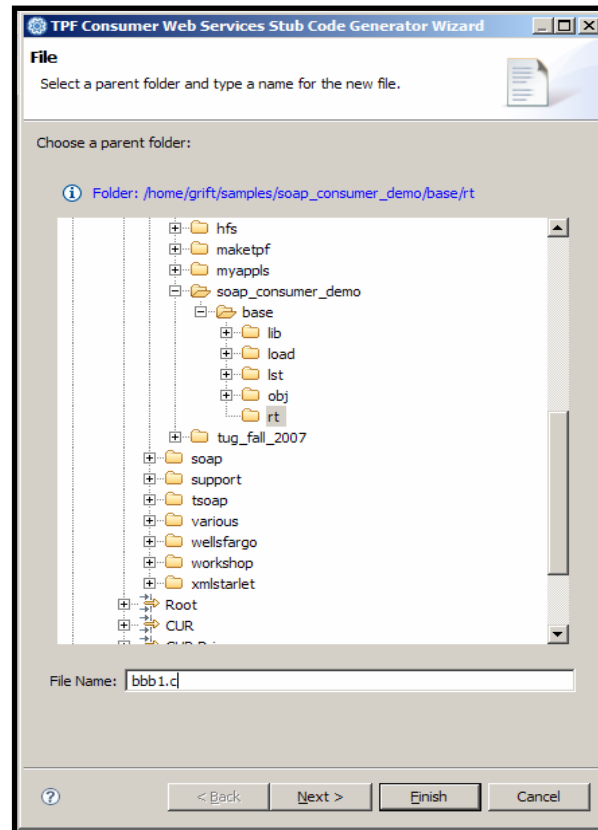
The STUB

- **Uses TPF_CALL_BY_NAME to call the Web service stub program name**
- **Processing is implemented in C shared objects**

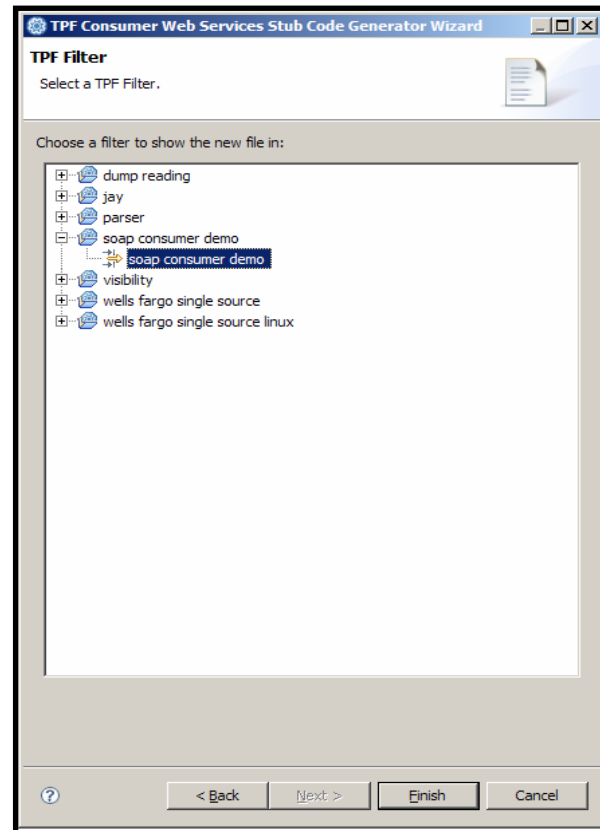
Generate STUB Templates



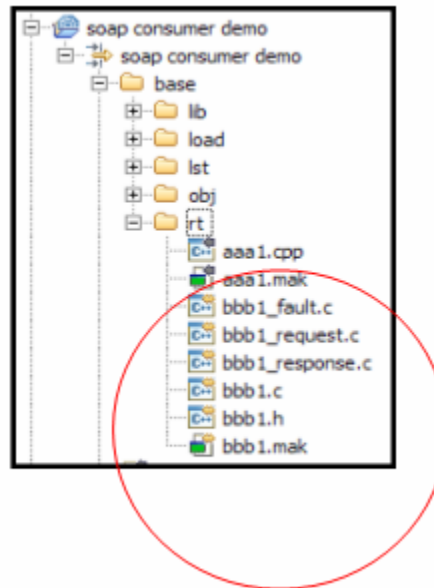
Specify STUB Location & Name



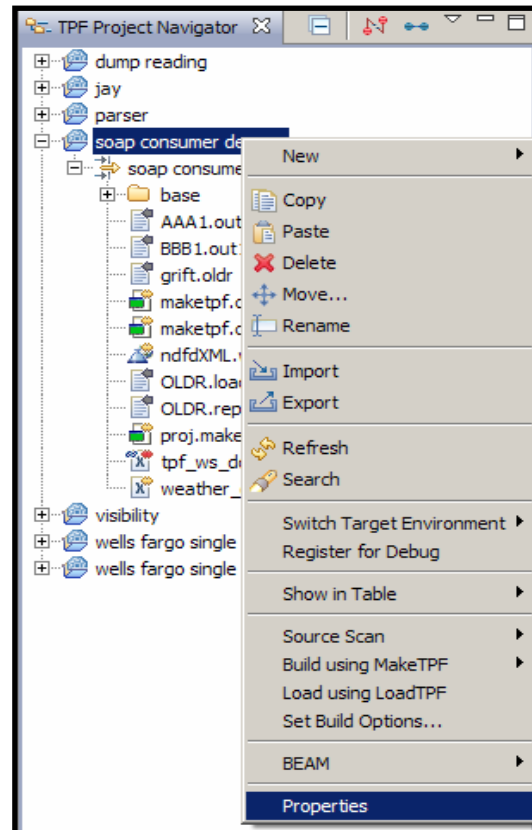
Specify a Filter



Generated Template Segments



Add the STUB to the Project



Add the STUB to the Build List

Properties for soap consumer demo

type filter text

- Build Order
- Info
- Remote Working Directory
- Target Environments
- TPF Make Build List**
- TPF Make Configuration
- TPF Make Load File
- User Variables

TPF Make Build List

Use an external control file

External control file: Browse...

For example, /home/path/external.cnt

Use this project's control file

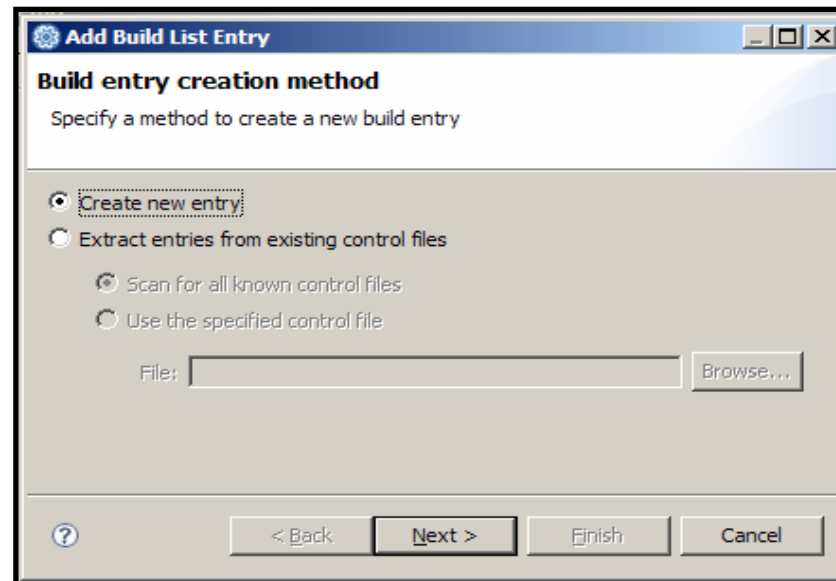
Control file version: 2

Build List

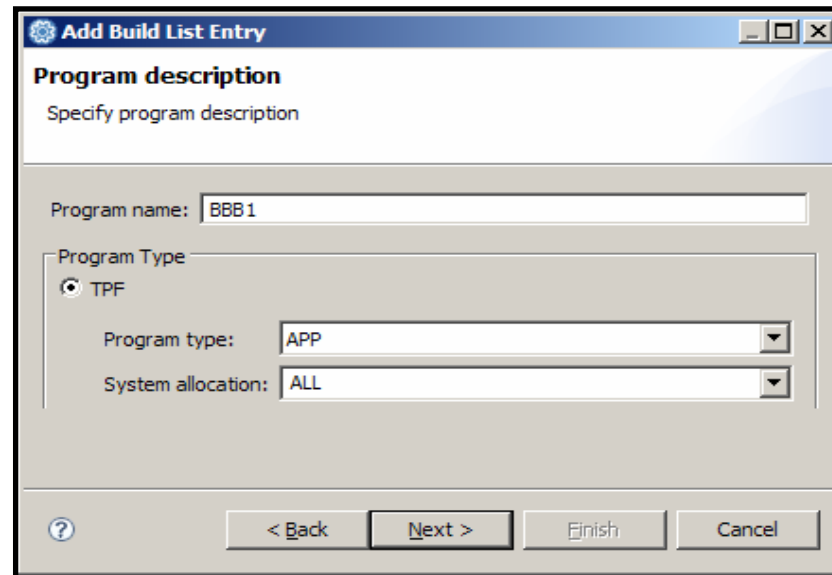
Program Name	Program Type	Makefile	Build Passes	System Allocation	Object
AAA1	APP	/hom...	1	ALL	OBJ

Buttons: Add..., Edit..., Remove, Move Up, Move Down, Reload from project

Create a New Build List Entry



Specify the STUB Program Name



Add Build List Entry

Program description
Specify program description

Program name: BBB1

Program Type

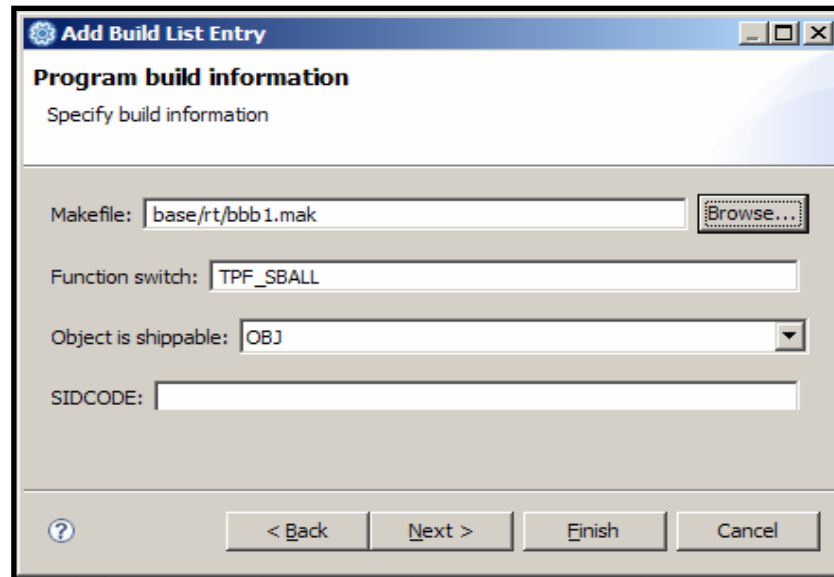
TPF

Program type: APP

System allocation: ALL

? < Back Next > Finish Cancel

Specify the STUB Make File



Add Build List Entry

Program build information
Specify build information

Makefile:

Function switch:

Object is shippable:

SIDCODE:

Add the STUB to the Control File

Properties for soap consumer demo

type filter text

- Build Order
- Info
- Remote Working Directory
- Target Environments
- TPF Make Build List**
- TPF Make Configuration
- TPF Make Load File
- User Variables

TPF Make Build List

Use an external control file

External control file:

For example, /home/path/external.cntl

Use this project's control file

Control file version: 2

Build List

Program Name	Program Type	Makefile	Build Passes	System Allocation	Object
AAA1	APP	/hom...	1	ALL	OBJ
BBB1	APP	base/...	1	ALL	OBJ

Request Message

```
<Body>
  <NDFDGen>
    <latitude>41.90</latitude>
    <longitude>-87.65</longitude>
    <product>glance</product>
    <startTime>2009-03-10T12:00</startTime>
    <endTime>2009-03-10T12:00</endTime>
    <weatherParameters>maxt=TRUE</weatherParameters>
  </NDFDGen>
</Body>
```


Response Message

```
<Body>
  <NDFDGen>
    <dwmlOut>

      ...

      <parameters applicable-location="point1">
        <temperature type="maximum" units="Fahrenheit" time-layout="k-p24h-n1-1">
          <name>Daily Maximum Temperature</name>
          <value>51</value>
        </temperature>
      </parameters>

      ...

    </dwmlOut>
  </NDFDGen>
</Body>
```

Generated Request Handler

```
int bbb1_NDFDgen_request (t_soapHandle soapHandle, XMLHandle requestMsg,
                          t_soapParms *parmsArray, void **returnValue)
{
    int rc = WSSTUB_RET_CONT;
    /* Function return code */

    /******
    /* ADD YOUR CODE HERE TO APPEND THE NECESSARY ELEMENTS UNDER
    /* THE "BODY" ELEMENT TO BUILD A VALID SOAP REQUEST MESSAGE
    /* FOR THIS OPERATION.
    /******

    /******
    /* Return to tpf_soapInvokeService processing.
    /******
    return (rc);
} /* end of function bbb1_NDFDgen_request ( ) */
```

Code the Request Handler 1/2

```
int bbb1_NDFDgen_request (t_soapHandle soapHandle, XMLHandle requestMsg,
                        t_soapParms *parmsArray, void **returnValue)
{
    int rc = WSSTUB_RET_CONT;
    int xmlRC = 0;
    xmlRC = tpf_xml_appendElement(requestMsg, "Body", "NDFDgen",
                                NULL, BBB1_NAMESPACE_PREFIX, BBB1_NAMESPACE, 0);
    if(xmlRC == TPF_SOAP_ERROR) {
        // Error handling
    }
    xmlRC = tpf_xml_appendElement(requestMsg, "NDFDgen", "latitude",
                                parmsArray->parmArray[0].parmPtr,
                                BBB1_NAMESPACE_PREFIX, BBB1_NAMESPACE, 0);
    if(xmlRC == TPF_SOAP_ERROR) {
        // Error handling
    }
    xmlRC = tpf_xml_appendElement(requestMsg, "NDFDgen", "longitude",
                                parmsArray->parmArray[1].parmPtr,
                                BBB1_NAMESPACE_PREFIX, BBB1_NAMESPACE, 0);
    if(xmlRC == TPF_SOAP_ERROR) {
        // Error handling
    }
}
```

Code the Request Handler 2/2

```
xmlRC = tpf_xml_appendElement(requestMsg, "NDFDgen", "product",
                              parmsArray->paramArray[2].parmPtr,
                              BBB1_NAMESPACE_PREFIX, BBB1_NAMESPACE, 0);

if(xmlRC == TPF_SOAP_ERROR) {
    // Error handling
}

xmlRC = tpf_xml_appendElement(requestMsg, "NDFDgen", "startTime",
                              parmsArray->paramArray[3].parmPtr,
                              BBB1_NAMESPACE_PREFIX, BBB1_NAMESPACE, 0);

if(xmlRC == TPF_SOAP_ERROR) {
    // Error handling
}

xmlRC = tpf_xml_appendElement(requestMsg, "NDFDgen", "endTime",
                              parmsArray->paramArray[4].parmPtr,
                              BBB1_NAMESPACE_PREFIX, BBB1_NAMESPACE, 0);

if(xmlRC == TPF_SOAP_ERROR) {
    // Error handling
}

xmlRC = tpf_xml_appendElement(requestMsg, "NDFDgen", "weatherParameters",
                              parmsArray->paramArray[5].parmPtr,
                              BBB1_NAMESPACE_PREFIX, BBB1_NAMESPACE, 0);

if(xmlRC == TPF_SOAP_ERROR) {
    // Error handling
}

return (rc);
}
```

Generated Response Handler

```
int bbb1_NDFDgen_response (t_soapHandle soapHandle, XMLHandle responseMsg,
                          void **returnValue)
{
    int          rc = WSSTUB_RET_SUCCESS;
    /* Function return code          */

    /******
    /* ADD YOUR CODE HERE TO EXTRACT THE NECESSARY DATA FROM THE      */
    /* SOAP RESPONSE MESSAGE AND CREATE THE REQUIRED INFORMATION          */
    /* TO BE PASSED BACK TO THE ORIGINAL CALLING APPLICATION.          */
    /******

    /******
    /* Return to tpf_soapInvokeService processing.                      */
    /******

    return (rc);
} /* end of function bbb1_NDFDgen_response ( )          */
```

Code the Response Handler 1/2

```
int bbb1_NDFDgen_response (t_soapHandle soapHandle, XMLHandle responseMsg,
                          void **returnValue)
{
    int rc = WSSTUB_RET_SUCCESS;
    xmlNodesArray* xptr = NULL;
    xmlNodesArray* wptr = NULL;
    int wRc;
    int parserRc;
    XMLHandle w;

    xptr = tpf_xml_getElementsByTagName (responseMsg, TYPE_TEXT, "dwmlOut");
    if(xptr == NULL) || (xptr->nodesArraySize == 0) {
        // Error handling
    }
}
```

Code the Response Handler 2/2

```
wRc = tpf_xml_initialize_handle(&w, B2B_XML_SCANNER, NULL);
if((wRc != 0) || (w == 0)) {
    // Error handling
}

wRc = tpf_xml_parseDocument(w, xptr->nodesArray[0].nodeValueStr,
                           TPF_CCSID_IBM1047,
                           strlen(xptr->nodesArray[0].nodeValueStr),
                           &parserRc, 0);

if(wRc != 0) {
    // Error Handling
}

wptr = tpf_xml_getElementsByTagName (w, TYPE_INT, "value");
if((wptr == NULL) || (wptr->nodesArraySize == 0)){
    // Error handling
}

*returnValue = malloc(sizeof(int));
*(int*)*returnValue = wptr->nodesArray[0].nodeValueDataType.nodeValueInt;
return (rc);
}
```

Code the Fault Handler

```
#include <tpf/tpfapi.h> /* edwin */

int bbb1_NDFDgen_fault (t_soapHandle soapHandle, XMLHandle faultMsg, void **returnValue)
{
    int rc = WSSTUB_RET_CONT; /* Function return code */

    /* *****
    /* ADD YOUR CODE HERE TO EXTRACT THE NECESSARY DATA FROM THE
    /* SOAP FAULT MESSAGE AND CREATE THE REQUIRED INFORMATION TO BE
    /* PASSED BACK TO THE ORIGINAL CALLING APPLICATION.
    /* *****

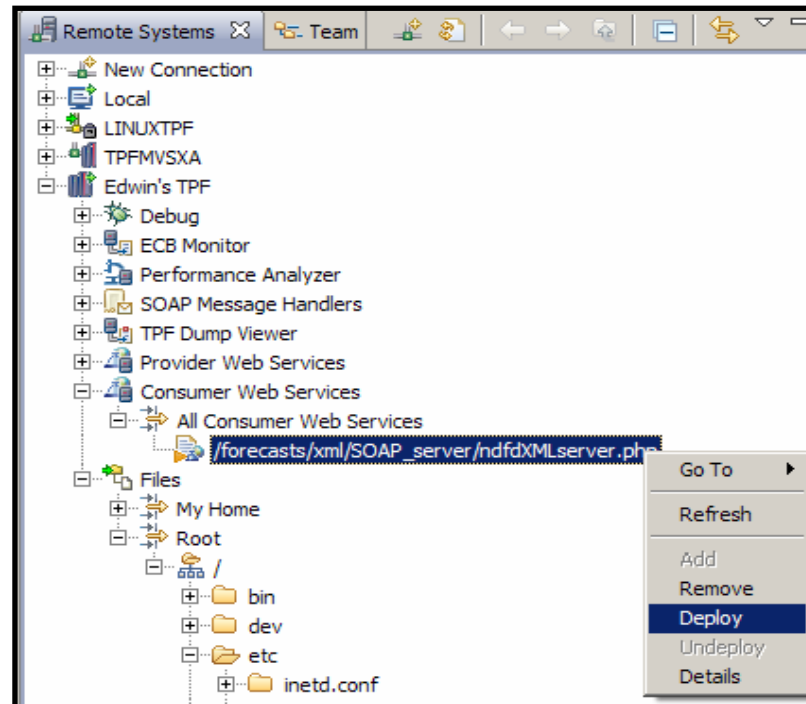
wtopc("bbb1 NFDgen fault", WTOPC_EBROUT + WTOPC_PRC, NULL, NULL); /* edwin */

    /* *****
    /* Return to tpf_soapInvokeService processing.
    /* *****
    return (rc);
}/* end of function bbb1_NDFDgen_fault ( ) */
```

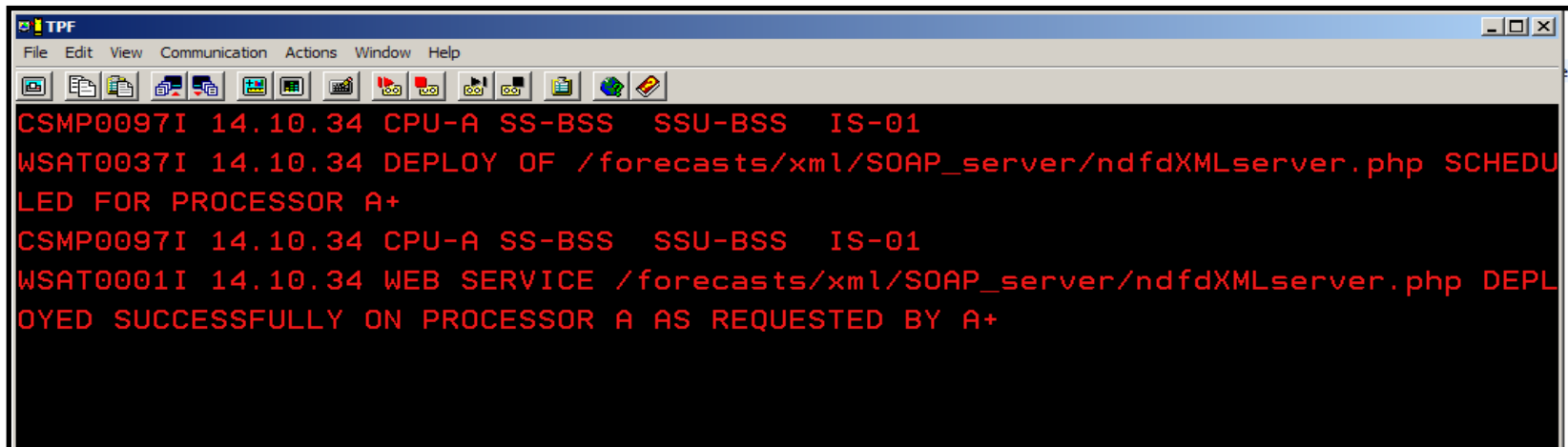

Summary

- **Get the Service's WSDL**
- **Create a Deployment Descriptor**
- **Create STUB code**
- **Deploy the Descriptor**
- **Invoke the Web Service**

Deploy the Web Service

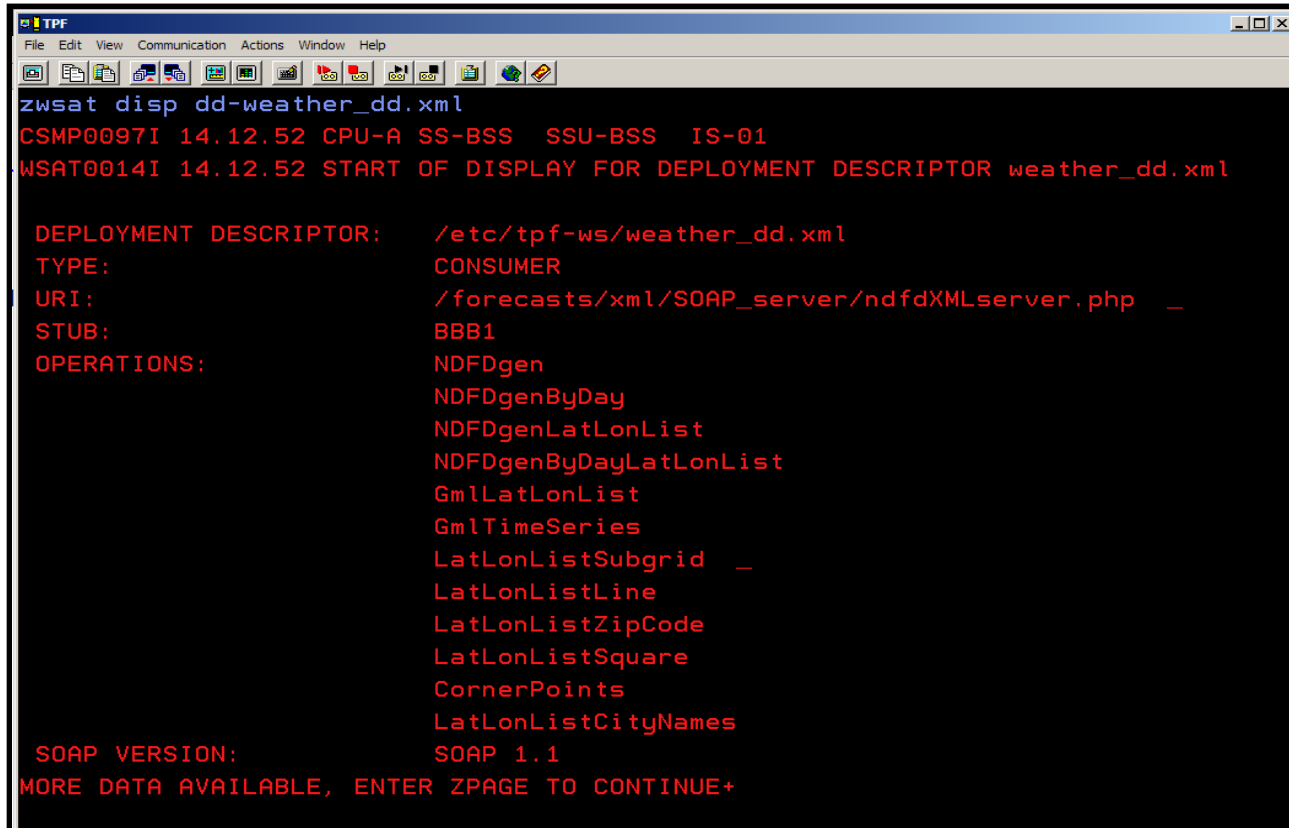


Deploy the Web Service



```
TPF
File Edit View Communication Actions Window Help
CSMP0097I 14.10.34 CPU-A SS-BSS  SSU-BSS  IS-01
WSAT0037I 14.10.34 DEPLOY OF /forecasts/xml/SOAP_server/ndfdXMLserver.php SCHEDU
LED FOR PROCESSOR A+
CSMP0097I 14.10.34 CPU-A SS-BSS  SSU-BSS  IS-01
WSAT0001I 14.10.34 WEB SERVICE /forecasts/xml/SOAP_server/ndfdXMLserver.php DEPL
OYED SUCCESSFULLY ON PROCESSOR A AS REQUESTED BY A+
```

Display the Web Service

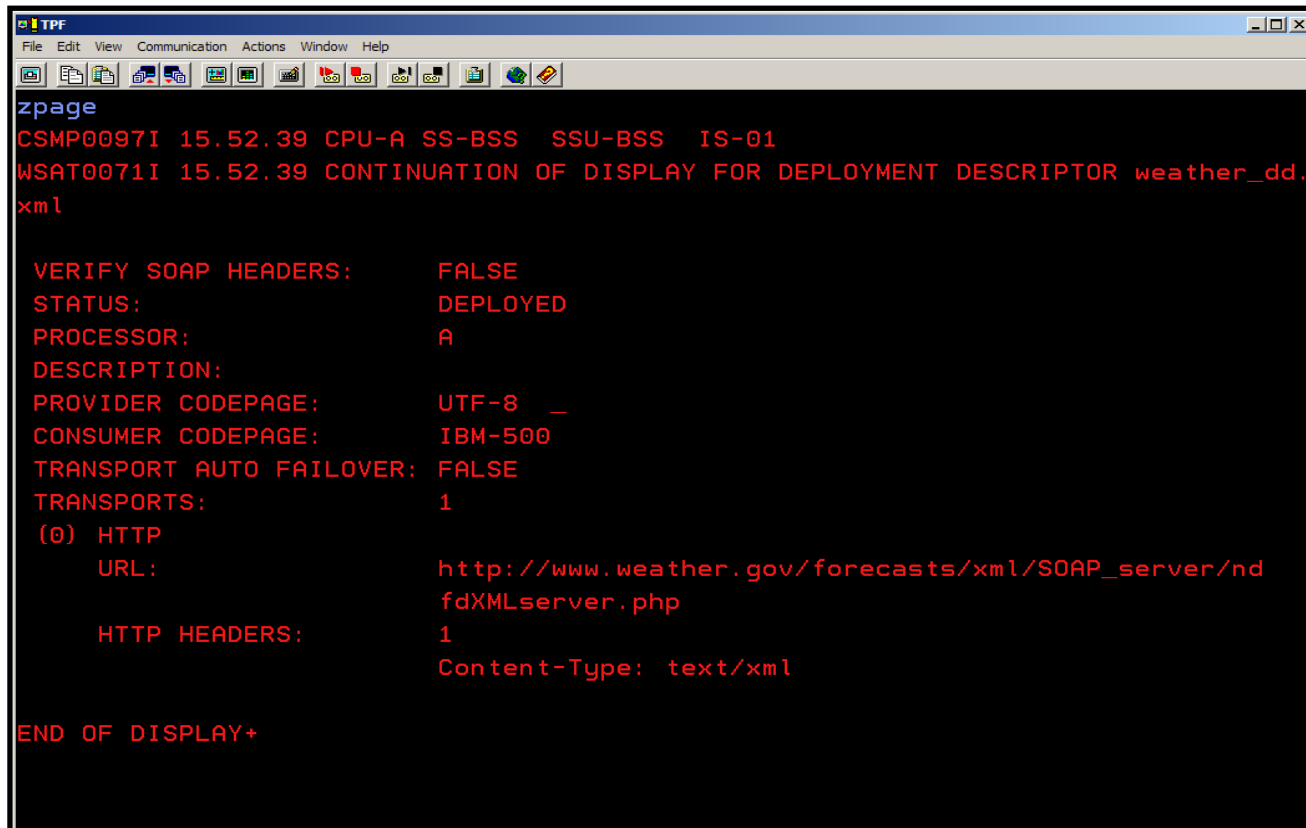


```
zwsat disp dd-weather_dd.xml
CSMP0097I 14.12.52 CPU-A SS-BSS  SSU-BSS  IS-01
WSAT0014I 14.12.52 START OF DISPLAY FOR DEPLOYMENT DESCRIPTOR weather_dd.xml

DEPLOYMENT_DESCRIPTOR:  /etc/tpf-ws/weather_dd.xml
TYPE:                   CONSUMER
URI:                   /forecasts/xml/SOAP_server/ndfdXMLserver.php  _
STUB:                  BBB1
OPERATIONS:            NDFDgen
                       NDFDgenByDay
                       NDFDgenLatLonList
                       NDFDgenByDayLatLonList
                       GmlLatLonList
                       GmlTimeSeries
                       LatLonListSubgrid  _
                       LatLonListLine
                       LatLonListZipCode
                       LatLonListSquare
                       CornerPoints
                       LatLonListCityNames

SOAP VERSION:          SOAP 1.1
MORE DATA AVAILABLE, ENTER ZPAGE TO CONTINUE+
```

Display the Web Service



```
zpage
CSMP0097I 15.52.39 CPU-A SS-BSS  SSU-BSS  IS-01
WSAT0071I 15.52.39 CONTINUATION OF DISPLAY FOR DEPLOYMENT DESCRIPTOR weather_dd.
xml

  VERIFY SOAP HEADERS:      FALSE
  STATUS:                   DEPLOYED
  PROCESSOR:                A
  DESCRIPTION:
  PROVIDER CODEPAGE:       UTF-8  _
  CONSUMER CODEPAGE:      IBM-500
  TRANSPORT AUTO FAILOVER: FALSE
  TRANSPORTS:              1
  (0) HTTP
    URL:                    http://www.weather.gov/forecasts/xml/SOAP_server/nd
                           fdXMLserver.php
  HTTP HEADERS:            1
                           Content-Type: text/xml

END OF DISPLAY+
```

Summary

- **Get the Service's WSDL**
- **Create a Deployment Descriptor**
- **Create STUB code**
- **Deploy the Descriptor**
- **Invoke the Web Service**

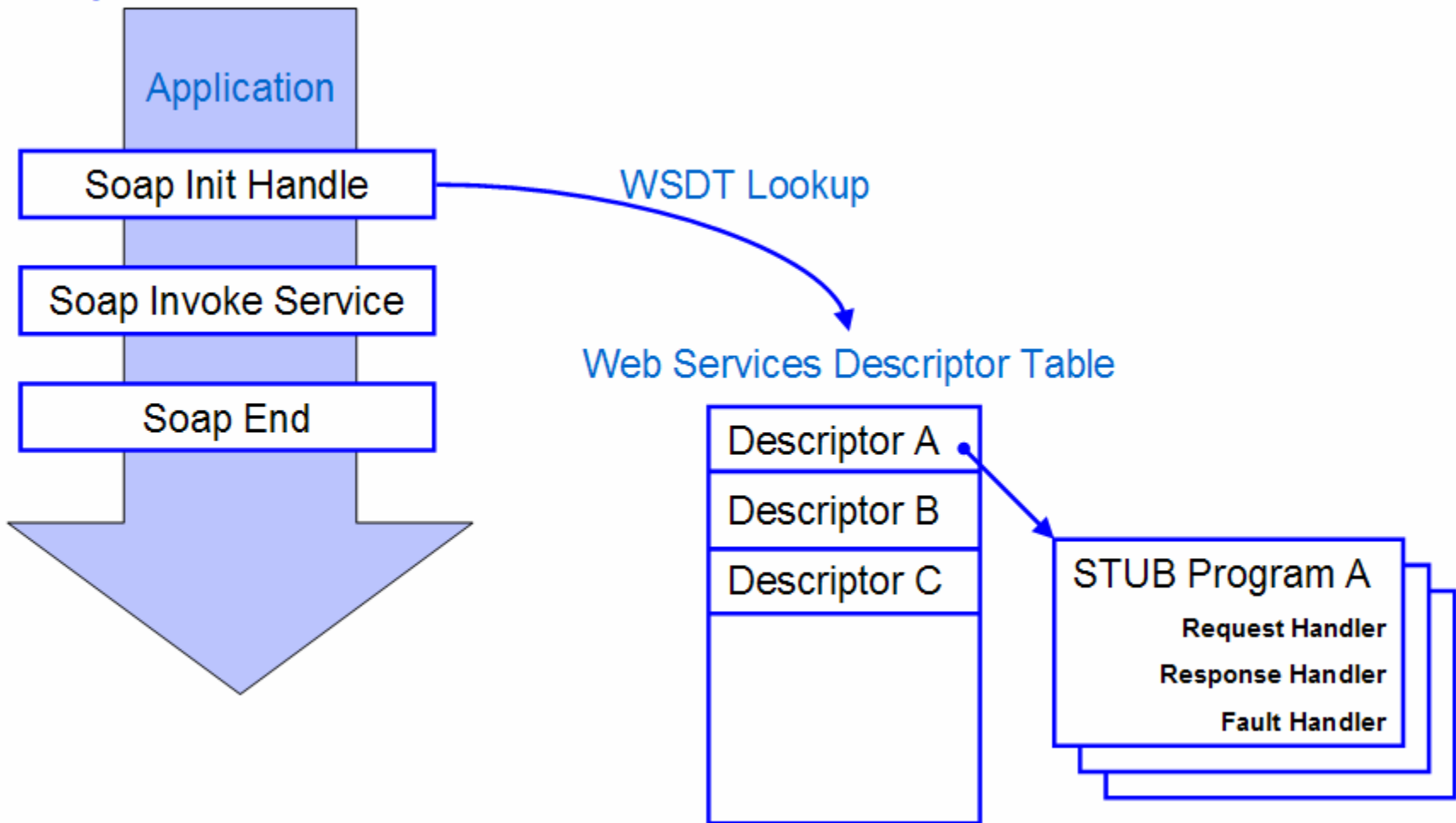
Start Using the Web Service

- **Shared object AAA1**
- **One C++ program aaa1.cpp**
- **To be invoked using ZFMSG**

aaa1.mak

```
#####  
# Define shared object name  
#####  
APP := AAA1  
APP_EXPORT := ENTRY  
APP_ENTRY := AAA1  
  
#####  
# Environments needed for build  
#####  
maketpf_env := base_rt  
maketpf_env += system  
  
LIBS += CWSA  
  
#####  
# CPP segments  
#####  
CXX_SRC := aaa1.cpp  
  
#####  
# Include maketpf build rules  
#####  
include maketpf.rules
```


Soap Consumer Flow



Initialize a Soap Consumer Handle

```
LIBS := CWSA
#include <tpf/c_soap.h>
t_soapHandle tpf_soapInitHandle (char *serviceName);
```

serviceName

A null-terminated string that represents the name of the Web service.

Normal return

A SOAP handle (`t_soapHandle`) is returned; this represents the SOAP consumer session handle. You must provide this handle for any following SOAP consumer function calls that will use the same session.

Call a Web Service

```
LIBS := CWSA
#include <tpf/c_soap.h>
int tpf_soapInvokeService (t_soapHandle soapHandle, t_soapInv *invocationParms,
                          void **soapResponse);
```

soapHandle

The SOAP consumer session handle to perform the request for.

invocationParms

A pointer to a fully initialized t_soapInv structure.

soapResponse

A pointer to the SOAP response that will point to different values depending on the dataFormat field.

Normal return

When the SOAP_CONSUMER_SYNC MEP is used, a return of TPF_SOAP_SUCCESS indicates that the tpf_soapInvokeService call was successful.

End Web Service Session

```
LIBS := CWSA
#include <tpf/c_soap.h>
int tpf_soapEnd (t_soapHandle soapHandle);
```

`soapHandle`

The SOAP consumer session handle to end.

Normal return

TPF_SOAP_SUCCESS indicates that the specified consumer Web service session ended.

aaa1.cpp 1/4

```
#include <tpf/c_soapc.h>
#include <stdio.h>
#include <stdlib.h>

extern "C" int AAA1() {

    char* date = "2009-03-10T12:00";

    t_soapHandle h = tpf_soapInitHandle("/forecasts/xml/SOAP_server/ndfdXMLserver.php");

    if(h == TPF_SOAP_ERROR) {

        // Error handling
        printf("Soap init handle error\n");
        exit(0);

    } else {
```

aaa1.cpp 2/4

```
t_soapInv parameters;

strcpy(parameters.operation, "NDFDgen");

parameters.timeout = 5;

parameters.messageExchangePattern = SOAP_CONSUMER_SYNC;

parameters.invocParmsVersion = SOAPC_VERSION_1;

parameters.dataFormat = SOAP_PARMS;

t_soapParms* parameterData = (t_soapParms*)calloc(1,
                                                    (sizeof(t_soapParms) + (sizeof(t_soapParmValues)*6)));

if(parameterData == 0) {
    // Error handling
    printf("Memory allocation failed\n");
    exit(0);
}
```

aaa1.cpp 3/4

```
parameterData->numParms = 6;

parameterData->parmArray[0].parmLength = 5;
parameterData->parmArray[0].parmPtr = (void*)"41.90";           // latitude of Chicago, IL
parameterData->parmArray[1].parmLength = 6;
parameterData->parmArray[1].parmPtr = (void*)"-87.65";         // longitude of Chicago, IL
parameterData->parmArray[2].parmLength = 6;
parameterData->parmArray[2].parmPtr = (void*)"glance";         // product
parameterData->parmArray[3].parmLength = 16;
parameterData->parmArray[3].parmPtr = (void*)date;             // startTime
parameterData->parmArray[4].parmLength = 16;
parameterData->parmArray[4].parmPtr = (void*)date;             // endTime
parameterData->parmArray[5].parmLength = 9;
parameterData->parmArray[5].parmPtr = (void*)"maxt=TRUE";     // weatherParameters

parameters.data = parameterData;
```

aaa1.cpp 4/4

```
void* response;
if(tpf_soapInvokeService(h, &parameters, &response) != TPF_SOAP_SUCCESS) {

    // Error handling
    tpf_soapEnd(h);
    printf("Soap invoke service error - %s\n", strerror(errno));
    exit(0);

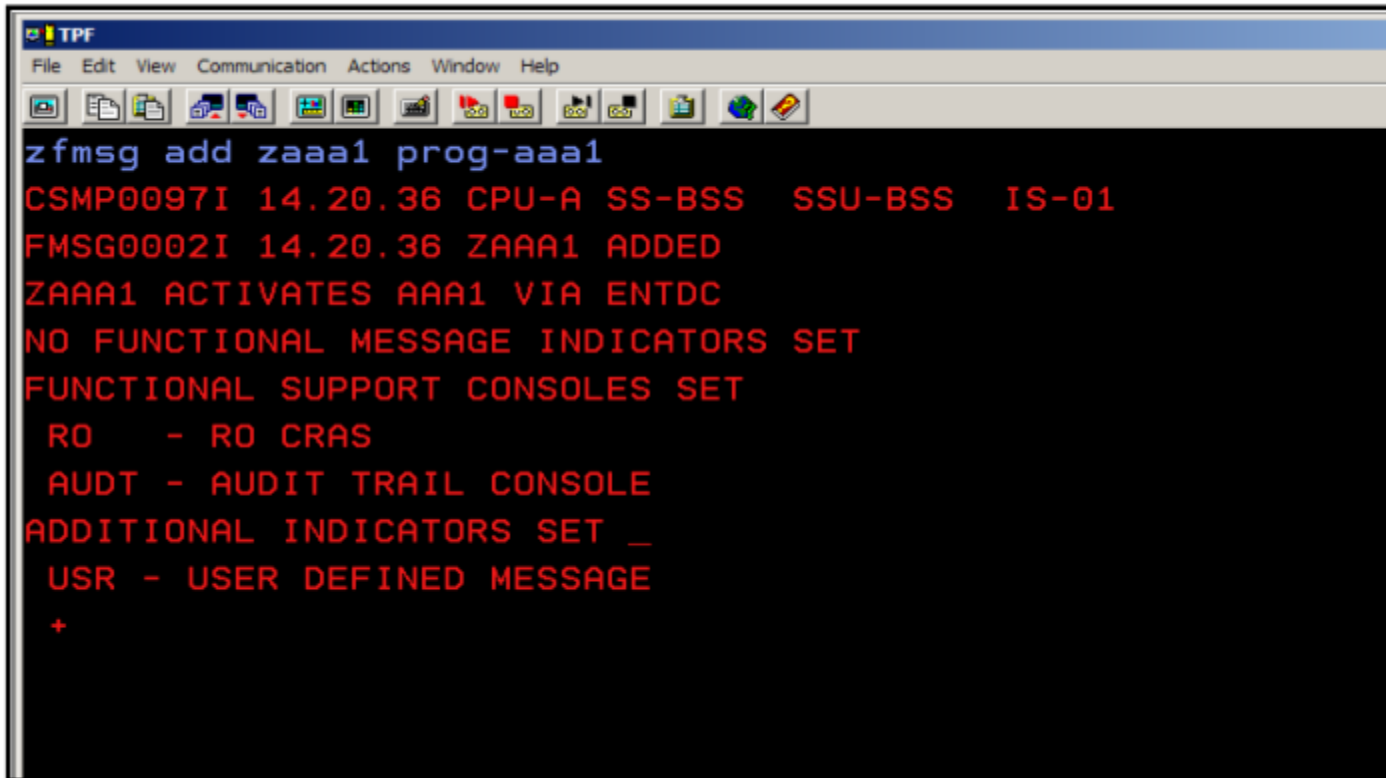
}

free(parameterData);
printf("High Temperature Forecast for %s is %d\n", date, *(int*)response);
free(response);
tpf_soapEnd(h);

}

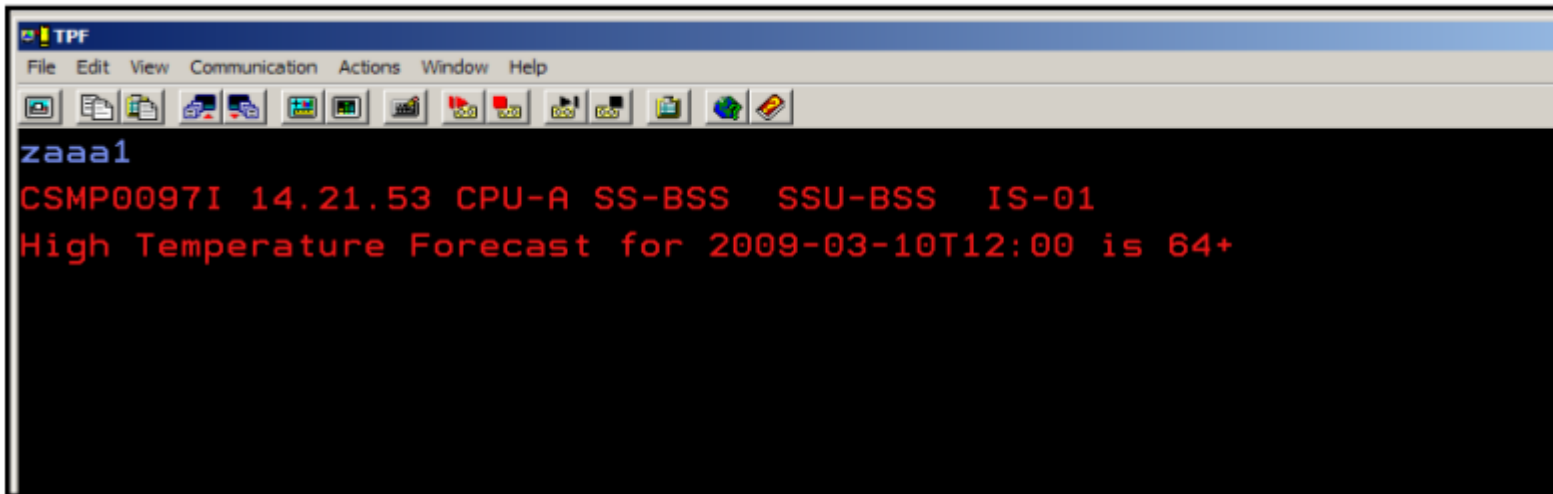
return 0;
}
```


ZAAA1



```
zfmsg add zaaa1 prog-aaa1
CSMP0097I 14.20.36 CPU-A SS-BSS SSU-BSS IS-01
FMSG0002I 14.20.36 ZAAA1 ADDED
ZAAA1 ACTIVATES AAA1 VIA ENTDC
NO FUNCTIONAL MESSAGE INDICATORS SET
FUNCTIONAL SUPPORT CONSOLES SET
  RO   - RO CRAS
  AUDT - AUDIT TRAIL CONSOLE
ADDITIONAL INDICATORS SET _
  USR  - USER DEFINED MESSAGE
+
```

ZAAA1



The screenshot shows a terminal window titled "TPF" with a menu bar (File, Edit, View, Communication, Actions, Window, Help) and a toolbar. The terminal content is as follows:

```
zaaa1  
CSMP0097I 14.21.53 CPU-A SS-BSS SSU-BSS IS-01  
High Temperature Forecast for 2009-03-10T12:00 is 64+
```

Summary

- **Get a Web service's WSDL**
- **Create a Deployment Descriptor**
- **Create STUB code**
- **Deploy the Descriptor**

Travel Home Safely!

Edwin W. van de Grift

edwinvandegrift@us.ibm.com

IBM Software Group

Application & Integration Middleware



Trademarks

- **IBM and WebSphere are trademarks of International Business Machines Corporation in the United States, other countries, or both.**
- **Notes**
- **This publication was produced in the United States. IBM may not offer the products, services or features discussed in this document in other countries, and the information may be subject to change without notice. Consult your local IBM business contact for information on the product or services available in your area.**
- **All statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.**
- **Information about non-IBM products is obtained from the manufacturers of those products or their published announcements. IBM has not tested those products and cannot confirm the performance, compatibility, or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.**
- **This presentation and the claims outlined in it were reviewed for compliance with US law. Adaptations of these claims for use in other geographies must be reviewed by the local country counsel for compliance with local laws.**

