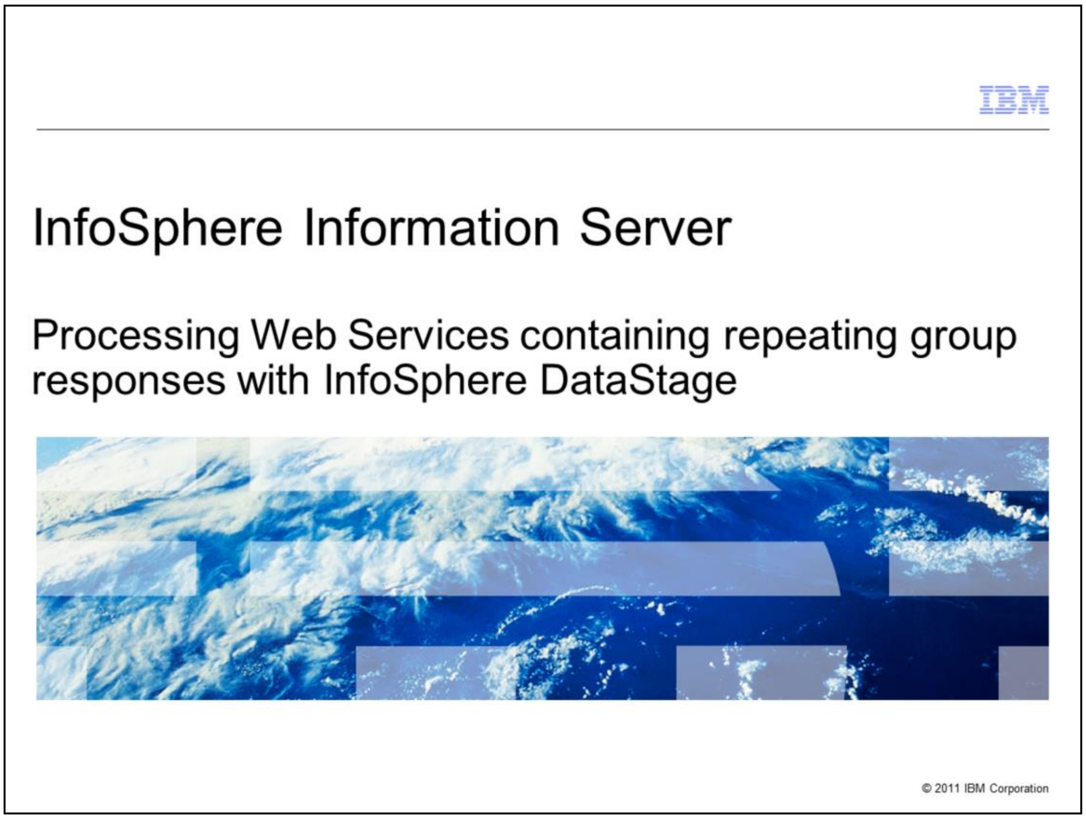
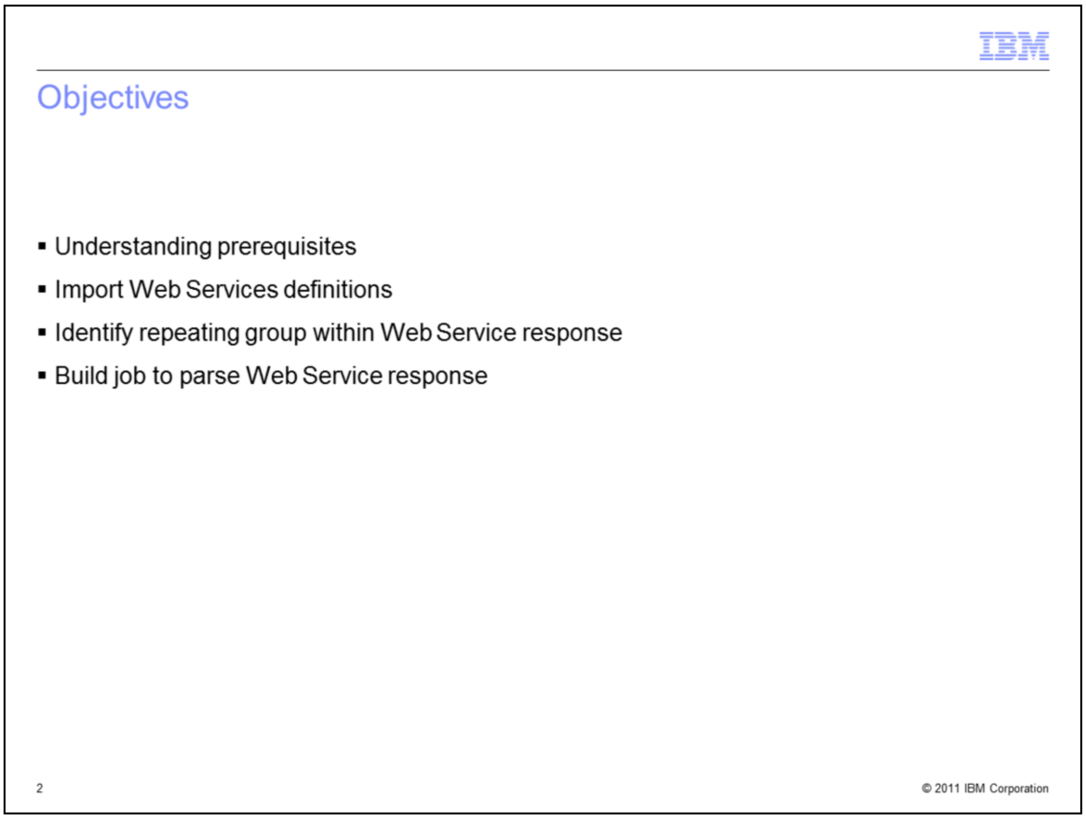


InfoSphere Information Server

Processing Web Services containing repeating group responses with InfoSphere DataStage

© 2011 IBM Corporation

This presentation describes how to create an InfoSphere™ DataStage® job to process Web Services Responses with repeating groups. The example in this module is based on DataStage 8.5 but the principles are the same for all versions of DataStage.



Objectives

- Understanding prerequisites
- Import Web Services definitions
- Identify repeating group within Web Service response
- Build job to parse Web Service response

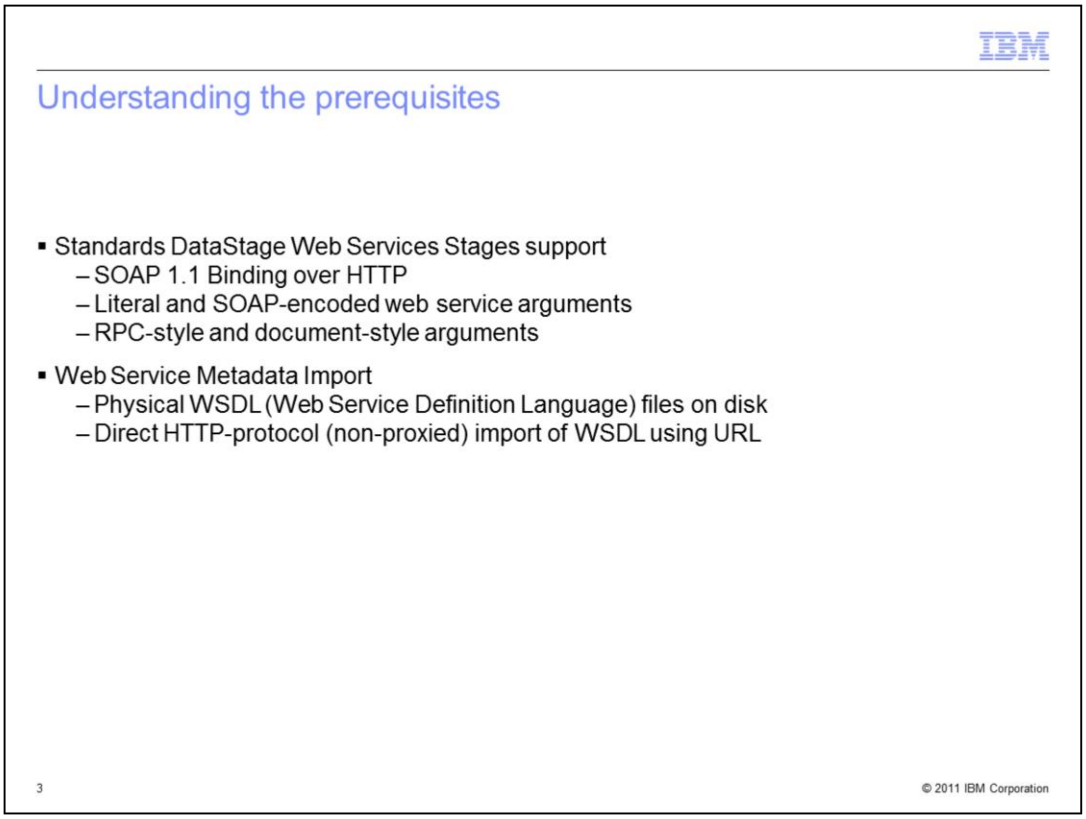
2

© 2011 IBM Corporation

The objectives of this presentation are to demonstrate the more advanced concepts and techniques required for accessing published Web Services which use repeating group structures. Note: the terminology 'Array' is also used to describe the group structure which repeats.

The Web Service client stage does not have the capability to process repeating element groups when used stand-alone. It must be paired with a separate XML processing stage to parse the repeating element groups into multiple rows and columns.

This module will show how to identify web services that contain repeating group structures, and how to create DataStage jobs that use both the Web Service Client and XML Input stage to handle these structures.


IBM

Understanding the prerequisites

- Standards DataStage Web Services Stages support
  - SOAP 1.1 Binding over HTTP
  - Literal and SOAP-encoded web service arguments
  - RPC-style and document-style arguments
- Web Service Metadata Import
  - Physical WSDL (Web Service Definition Language) files on disk
  - Direct HTTP-protocol (non-proxied) import of WSDL using URL

3

© 2011 IBM Corporation

This slide displays the web standards that are supported by the DataStage Web Services stages, and the methods available for importing Web Service definitions. The steps in this module will use Web Services published on the public internet. To complete these steps, your local DataStage client AND your DataStage server will both need to have access to the public internet.

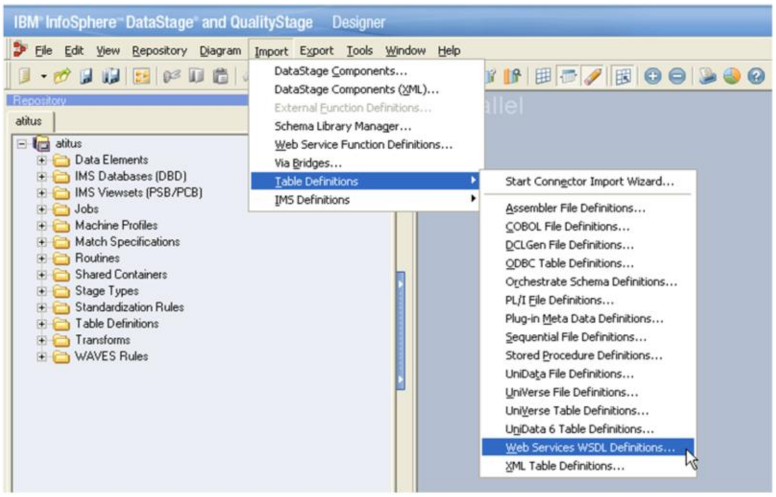
This module covers more advanced topics. While it is not absolutely necessary, you should first complete the “Processing Web Services as a data source with InfoSphere DataStage” IBM Education Assistant module before you begin with this module.

The examples in this presentation are based on DataStage version 8.5, using Parallel canvas jobs. You should be able to complete the module using other versions of DataStage, or by using Server canvas jobs, however the screen captures and exact steps may vary slightly.

IBM

Import Web Services definitions (1 of 4)

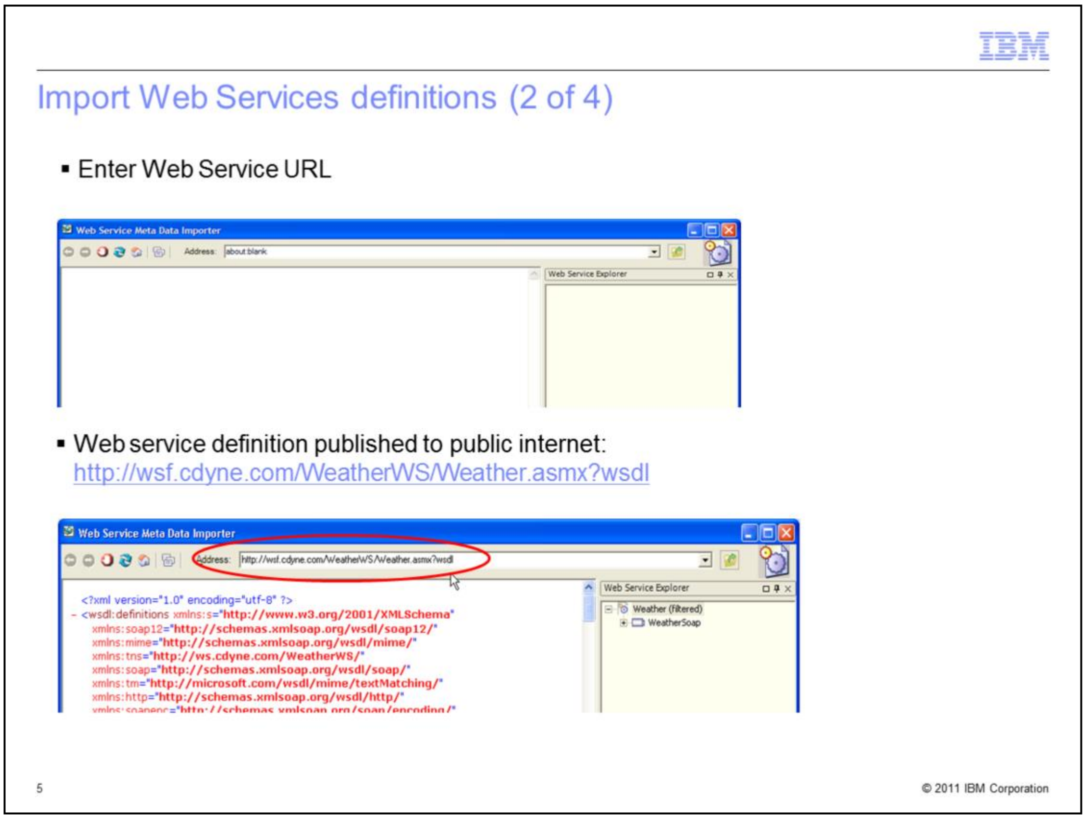
- Start Web Services Metadata Importer


IBM InfoSphere DataStage and QualityStage Designer
File Edit View Repository Diagram Import Export Tools Window Help
DataStage Components...
DataStage Components (XML)...
External Function Definitions...
Schema Library Manager...
Web Service Function Definitions...
Via Bridges...
Table Definitions
IMS Definitions
Start Connector Import Wizard...
Assembler File Definitions...
COBOL File Definitions...
DCLGen File Definitions...
ODBC Table Definitions...
Orchestrate Schema Definitions...
PL/I File Definitions...
Plug-in Meta Data Definitions...
Sequential File Definitions...
Stored Procedure Definitions...
UniData File Definitions...
UniVerse File Definitions...
UniVerse Table Definitions...
UniData 6 Table Definitions...
Web Services WSDL Definitions...
XML Table Definitions...
Repository
atitus
Data Elements
IMS Databases (DBD)
IMS Viewsets (PSB/PCB)
Jobs
Machine Profiles
Match Specifications
Routines
Shared Containers
Stage Types
Standardization Rules
Table Definitions
Transforms
WAVES Rules

4

© 2011 IBM Corporation

The first step is to import the Web Service Metadata. Start the Web Service Metadata Importer by opening the DataStage Designer (or DataStage Manager in version 7). Next, click Import from the main menu, then click Table Definitions, then click Web Services WSDL Definitions.



Import Web Services definitions (2 of 4)

- Enter Web Service URL

Web Service Meta Data Importer
Address: about:blank
Web Service Explorer

- Web service definition published to public internet:
  http://wsf.cdyne.com/WeatherWS/Weather.asmx?wsdl

Web Service Meta Data Importer
Address: http://wsf.cdyne.com/WeatherWS/Weather.asmx?wsdl
Web Service Explorer
Weather (filtered)
WeatherSoap

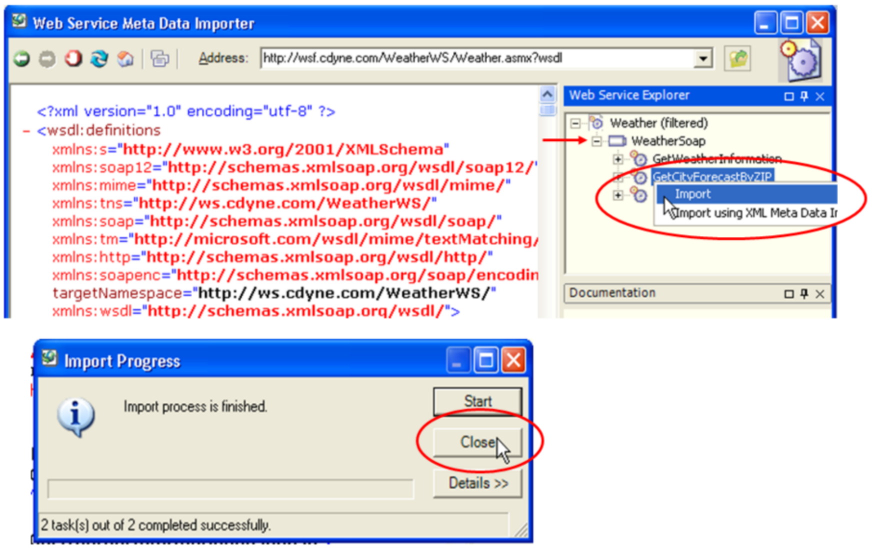
```
  <?xml version="1.0" encoding="utf-8" ?>
- <wsdl:definitions xmlns:s="http://www.w3.org/2001/XMLSchema"
    xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/"
    xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
    xmlns:tns="http://ws.cdyne.com/WeatherWS/"
    xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
    xmlns:tm="http://microsoft.com/wsdl/mime/textMatching/"
    xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
    xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
```

5 © 2011 IBM Corporation

The Web Service Metadata Importer should open as a new window. In this module, you will use a web service that returns weather forecast information. The URL for the web service is provided in the slide. To browse the Web Service, enter the URL for the Web Service WSDL document into the Address field in the Web Service Metadata Importer and press the Enter key.

Import Web Services definitions (3 of 4)

- Import GetCityForecastByZip



```
  <?xml version="1.0" encoding="utf-8" ?>
- <wsdl:definitions
    xmlns:s="http://www.w3.org/2001/XMLSchema"
    xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/"
    xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
    xmlns:tns="http://ws.cdyne.com/WeatherWS/"
    xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
    xmlns:tm="http://microsoft.com/wsdl/mime/textMatching/
    xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
    xmlns:soapenc="http://schemas.xmlsoap.org/soap/encodin
    targetNamespace="http://ws.cdyne.com/WeatherWS/"
    xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
```

Import process is finished.

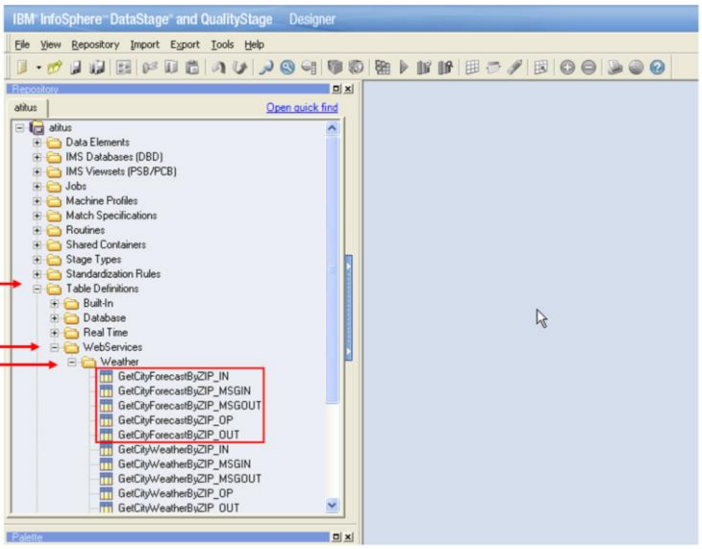
2 task(s) out of 2 completed successfully.

© 2011 IBM Corporation

Next, import the GetCityForecastByZIP operation. Click the [+] next to the entry for "WeatherSoap" in the Web Services Explorer pane to expand. Locate the GetCityForecastByZIP operation with the 'gears' icon next to it. Right-click this and select 'Import'. When the import finishes, press the Close button on the dialog box, and then the Close button on the Web Services Metadata Importer window.

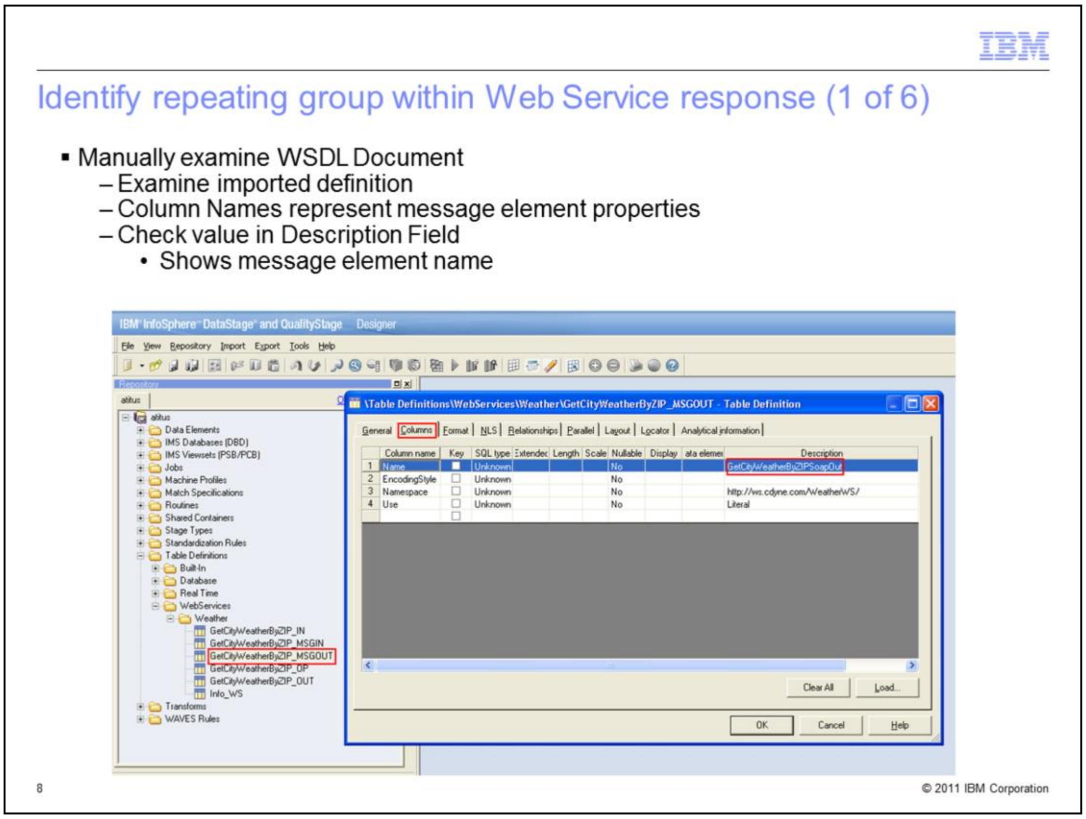
Import Web Services definitions (4 of 4)

- Verify import successful



7 © 2011 IBM Corporation

To verify that the import was successful, go to the repository view in the DataStage Designer (or the DataStage Manager in version 7) and locate the Table Definitions folder. Click to expand it. Then, locate the WebServices folder and expand it. You should have a Weather folder there for the operation that was just imported. Expanding it should show all of the entries labeled with GetCityForecastByZIP.



Identify repeating group within Web Service response (1 of 6)

- Manually examine WSDL Document
  - Examine imported definition
  - Column Names represent message element properties
  - Check value in Description Field
    - Shows message element name

\Table Definitions\WebServices\Weather\GetCityWeatherByZIP_MSGOUT - Table Definition

	Column name	Key	SQL type	Extended	Length	Scale	Nullable	Display	Data element	Description
1	Name	☐	Unknown				No			GetCityWeatherByZIPSoapOut
2	EncodingStyle	☐	Unknown				No			
3	Namespace	☐	Unknown				No			http://ws.cdyne.com/WeatherWS/
4	Use	☐	Unknown				No			Literal

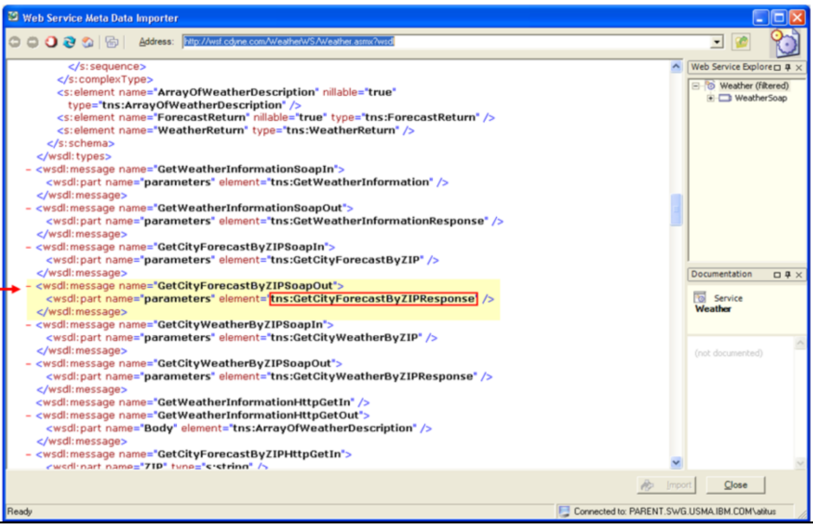
8 © 2011 IBM Corporation

DataStage does not have a graphical tool for identifying repeating element groups within a Web Service response. To do this, use a manual process of examining the WSDL (Web Services Definition Language) document, assisted by the Web Service Metadata importer.

Any given WSDL document may contain multiple operations and many different complex type definitions. To be sure you can identify what is needed, start by examining the imported definition stored in the DataStage repository. Using the DataStage Designer (or Manager with version 7) browse the repository under table definitions for the WebServices folder. Expand Weather and locate the GetCityForecastByZIP_MSGOUT definition. Double-click to open it and then select the Columns tab. The columns listed represent the message element properties. The 'name' property identifies the message element name that is returned from the web service in the Description field. As displayed on this slide, the message element name is GetCityForecastByZIPSoapOut.

Identify repeating group within Web Service response (2 of 6)

- Load the WSDL document
  - Locate wdsl:message
  - Locate element GetCityForecastByZIPSoapOut
    - GetCityForecastByZipResponse wsdl:part element encapsulated in GetCityForecastByZIPSoapOut

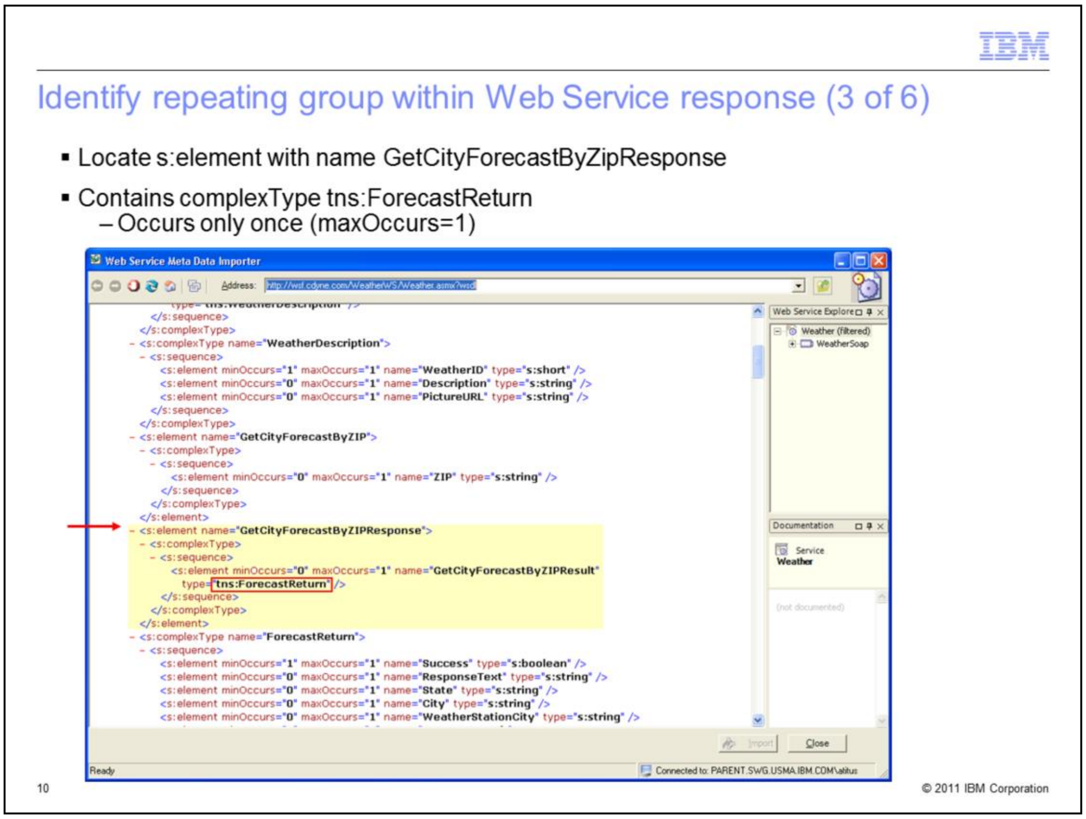

Web Service Meta Data Importer
Address: http://wsf.cdyne.com/WeatherWS/Weather.asmx?wsdl
Web Service Explorer
Weather (Filtered)
WeatherSoap
Documentation
Service
Weather
(not documented)
Import
Close
Ready
Connected to: PARENT.SWG.USMA.IBM.COM\altus

```
      </s:sequence>
    </s:complexType>
    <s:element name="ArrayOfWeatherDescription" nillable="true"
      type="tns:ArrayOfWeatherDescription" />
    <s:element name="ForecastReturn" nillable="true" type="tns:ForecastReturn" />
    <s:element name="WeatherReturn" type="tns:WeatherReturn" />
  </s:schema>
</wsdl:types>
- <wsdl:message name="GetWeatherInformationSoapIn">
    <wsdl:part name="parameters" element="tns:GetWeatherInformation" />
  </wsdl:message>
- <wsdl:message name="GetWeatherInformationSoapOut">
    <wsdl:part name="parameters" element="tns:GetWeatherInformationResponse" />
  </wsdl:message>
- <wsdl:message name="GetCityForecastByZIPSoapIn">
    <wsdl:part name="parameters" element="tns:GetCityForecastByZIP" />
  </wsdl:message>
- <wsdl:message name="GetCityForecastByZIPSoapOut">
    <wsdl:part name="parameters" element="tns:GetCityForecastByZIPResponse" />
  </wsdl:message>
- <wsdl:message name="GetCityWeatherByZIPSoapIn">
    <wsdl:part name="parameters" element="tns:GetCityWeatherByZIP" />
  </wsdl:message>
- <wsdl:message name="GetCityWeatherByZIPSoapOut">
    <wsdl:part name="parameters" element="tns:GetCityWeatherByZIPResponse" />
  </wsdl:message>
  <wsdl:message name="GetWeatherInformationHttpGetIn" />
- <wsdl:message name="GetWeatherInformationHttpGetOut">
    <wsdl:part name="Body" element="tns:ArrayOfWeatherDescription" />
  </wsdl:message>
- <wsdl:message name="GetCityForecastByZIPHttpGetIn">
    <wsdl:part name="ZIP" type="s:string" />
```

© 2011 IBM Corporation

From the DataStage Designer (or Manager in version 7), select Import from the main menu, expand Table Definitions, and click Web Services WSDL Definitions. This should open up the Web Service Metadata Importer as a new window. Enter the WSDL URL into the Address field, the same as you did in the previous slide, and press the 'Enter' key to load the document.

In the large main panel, scroll down to where the wsdl:message elements are being defined and locate the element with name="GetCityForecastByZIPSoapOut" which matches the value you obtained on the previous slide. This element is highlighted in yellow on this slide. Note that this message element encapsulates another wsdl:part element which is circled in red, GetCityForecastByZipResponse.



Identify repeating group within Web Service response (3 of 6)

- Locate s:element with name GetCityForecastByZipResponse
- Contains complexType tns:ForecastReturn
  - Occurs only once (maxOccurs=1)

Web Service Meta Data Importer
Address: http://wsf.cdyne.com/WeatherWS/Weather.asmx?wsdl
Web Service Explorer
Weather (filtered)
WeatherSoap
Documentation
Service
Weather
(not documented)
Import
Close
Ready
Connected to: PARENT.SWG.USMA.IBM.COM\altus

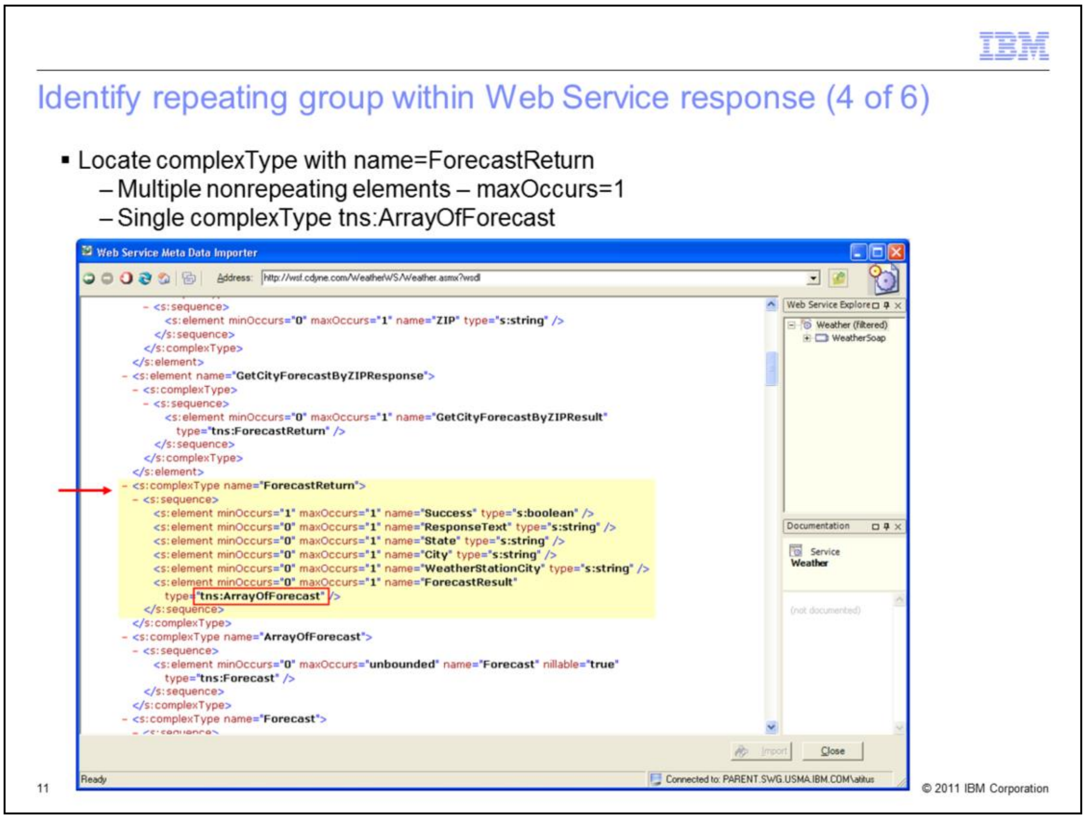
```
      </s:sequence>
    </s:complexType>
  - <s:complexType name="WeatherDescription">
    - <s:sequence>
        <s:element minOccurs="1" maxOccurs="1" name="WeatherID" type="s:short" />
        <s:element minOccurs="0" maxOccurs="1" name="Description" type="s:string" />
        <s:element minOccurs="0" maxOccurs="1" name="PictureURL" type="s:string" />
      </s:sequence>
    </s:complexType>
  - <s:element name="GetCityForecastByZIP">
    - <s:complexType>
      - <s:sequence>
          <s:element minOccurs="0" maxOccurs="1" name="ZIP" type="s:string" />
        </s:sequence>
      </s:complexType>
    </s:element>
  - <s:element name="GetCityForecastByZIPResponse">
    - <s:complexType>
      - <s:sequence>
          <s:element minOccurs="0" maxOccurs="1" name="GetCityForecastByZIPResult"
            type="tns:ForecastReturn" />
        </s:sequence>
      </s:complexType>
    </s:element>
  - <s:complexType name="ForecastReturn">
    - <s:sequence>
        <s:element minOccurs="1" maxOccurs="1" name="Success" type="s:boolean" />
        <s:element minOccurs="0" maxOccurs="1" name="ResponseText" type="s:string" />
        <s:element minOccurs="0" maxOccurs="1" name="State" type="s:string" />
        <s:element minOccurs="0" maxOccurs="1" name="City" type="s:string" />
        <s:element minOccurs="0" maxOccurs="1" name="WeatherStationCity" type="s:string" />
```

10

© 2011 IBM Corporation

Scroll back up to where the wsdl:types are being defined and locate the s:element with name="GetCityForecastByZIPResponse". This matches the wsdl:part from the previous step.

This element contains a different complexType called "tns:forecastReturn". maxOccurs=1 shows you that it occurs only once (its not repeating). You need to proceed further and look at this type to see if it has any repeating groups.



Identify repeating group within Web Service response (4 of 6)

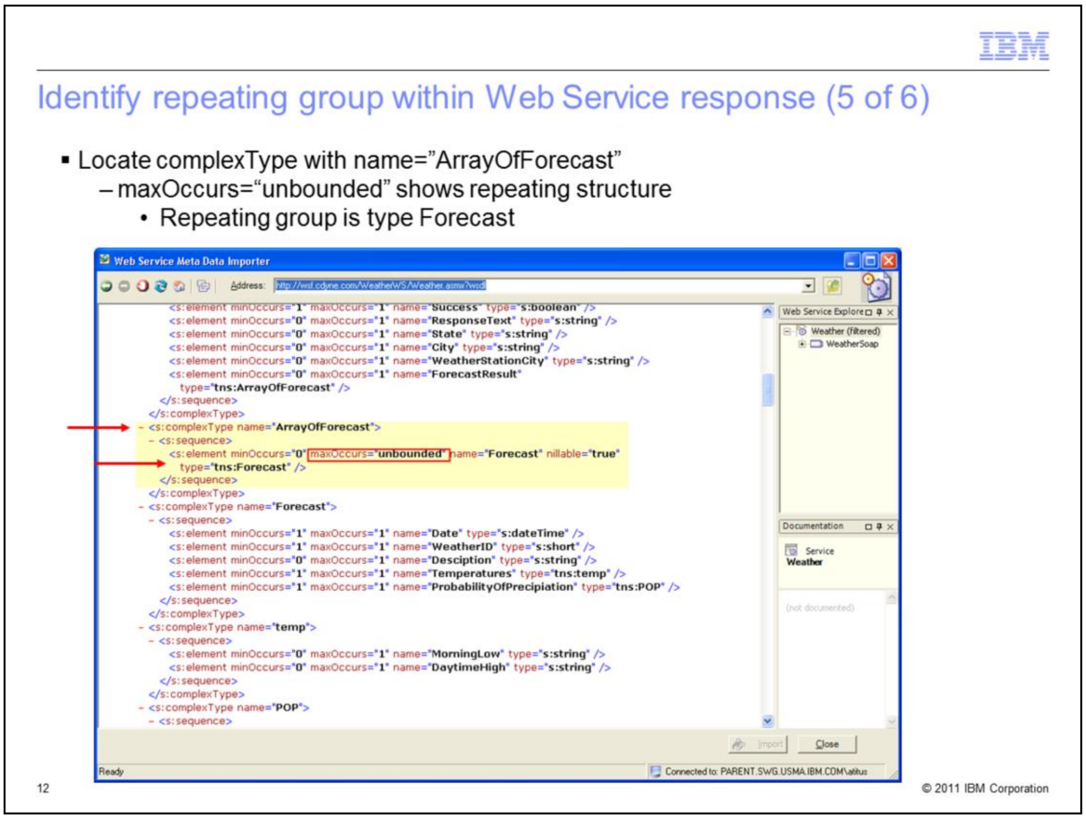
- Locate complexType with name=ForecastReturn
  - Multiple nonrepeating elements – maxOccurs=1
  - Single complexType tns:ArrayOfForecast

```
  - <s:sequence>
      <s:element minOccurs="0" maxOccurs="1" name="ZIP" type="s:string" />
    </s:sequence>
  </s:complexType>
</s:element>
- <s:element name="GetCityForecastByZIPResponse">
  - <s:complexType>
    - <s:sequence>
        <s:element minOccurs="0" maxOccurs="1" name="GetCityForecastByZIPResult"
          type="tns:ForecastReturn" />
      </s:sequence>
    </s:complexType>
  </s:element>
- <s:complexType name="ForecastReturn">
  - <s:sequence>
      <s:element minOccurs="1" maxOccurs="1" name="Success" type="s:boolean" />
      <s:element minOccurs="0" maxOccurs="1" name="ResponseText" type="s:string" />
      <s:element minOccurs="0" maxOccurs="1" name="State" type="s:string" />
      <s:element minOccurs="0" maxOccurs="1" name="City" type="s:string" />
      <s:element minOccurs="0" maxOccurs="1" name="WeatherStationCity" type="s:string" />
      <s:element minOccurs="0" maxOccurs="1" name="ForecastResult"
        type="tns:ArrayOfForecast" />
    </s:sequence>
  </s:complexType>
- <s:complexType name="ArrayOfForecast">
  - <s:sequence>
      <s:element minOccurs="0" maxOccurs="unbounded" name="Forecast" nillable="true"
        type="tns:Forecast" />
    </s:sequence>
  </s:complexType>
- <s:complexType name="Forecast">
```

Web Service Meta Data Importer
Address: http://wsf.cdyne.com/WeatherWS/Weather.asmx?wsdl
Web Service Explorer
Weather (filtered)
WeatherSoap
Documentation
Service
Weather
(not documented)
Import
Close
Ready
Connected to: PARENT.SWG.USMA.IBM.COM\altus

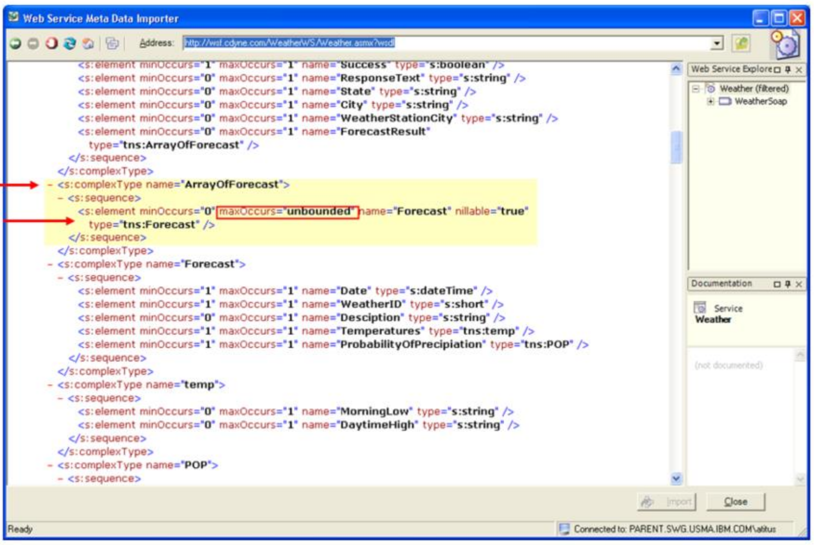
11 © 2011 IBM Corporation

Continue scrolling through the types until you locate the complexType element with name=”ForecastReturn”. This is highlighted in yellow on this slide. Once again, this defines several elements which are not repeating as seen by maxOccurs=”1”. There is also a single complexType “tns:ArrayOfForecast” which must be investigated as well.



Identify repeating group within Web Service response (5 of 6)

- Locate complexType with name="ArrayOfForecast"
  - maxOccurs="unbounded" shows repeating structure
    - Repeating group is type Forecast

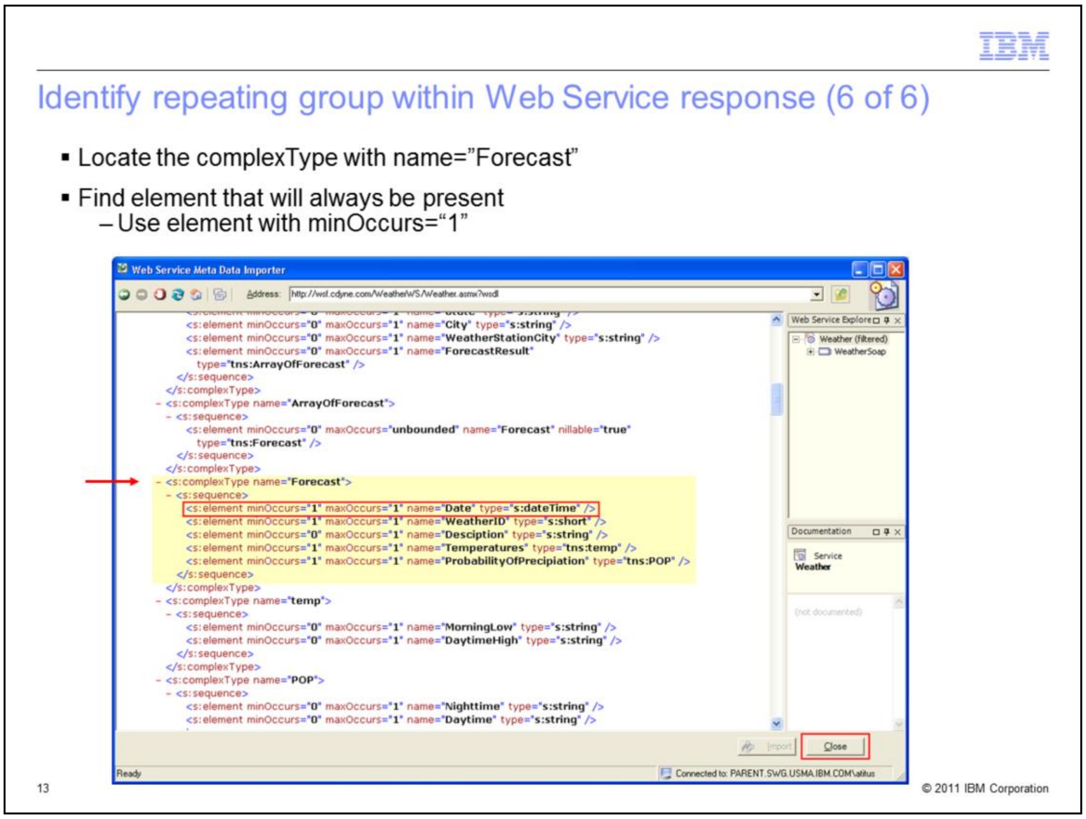

Web Service Meta Data Importer
Address: http://wsf.cdyne.com/WeatherWS/Weather.asmx?wsdl
Web Service Explorer
Weather (filtered)
WeatherSoap
Documentation
Service
Weather
(not documented)
Import
Close
Ready
Connected to: PARENT.SWG.USMA.IBM.COM\atitus

```
        <s:element minOccurs="1" maxOccurs="1" name="Success" type="s:boolean" />
        <s:element minOccurs="0" maxOccurs="1" name="ResponseText" type="s:string" />
        <s:element minOccurs="0" maxOccurs="1" name="State" type="s:string" />
        <s:element minOccurs="0" maxOccurs="1" name="City" type="s:string" />
        <s:element minOccurs="0" maxOccurs="1" name="WeatherStationCity" type="s:string" />
        <s:element minOccurs="0" maxOccurs="1" name="ForecastResult"
          type="tns:ArrayOfForecast" />
      </s:sequence>
    </s:complexType>
  - <s:complexType name="ArrayOfForecast">
    - <s:sequence>
        <s:element minOccurs="0" maxOccurs="unbounded" name="Forecast" nillable="true"
          type="tns:Forecast" />
      </s:sequence>
    </s:complexType>
  - <s:complexType name="Forecast">
    - <s:sequence>
        <s:element minOccurs="1" maxOccurs="1" name="Date" type="s:dateTime" />
        <s:element minOccurs="1" maxOccurs="1" name="WeatherID" type="s:short" />
        <s:element minOccurs="0" maxOccurs="1" name="Desciption" type="s:string" />
        <s:element minOccurs="1" maxOccurs="1" name="Temperatures" type="tns:temp" />
        <s:element minOccurs="1" maxOccurs="1" name="ProbabilityOfPrecipiation" type="tns:POP" />
      </s:sequence>
    </s:complexType>
  - <s:complexType name="temp">
    - <s:sequence>
        <s:element minOccurs="0" maxOccurs="1" name="MorningLow" type="s:string" />
        <s:element minOccurs="0" maxOccurs="1" name="DaytimeHigh" type="s:string" />
      </s:sequence>
    </s:complexType>
  - <s:complexType name="POP">
    - <s:sequence>
```

12

© 2011 IBM Corporation

Next, locate the complexType with name="ArrayOfForecast". This type identifies a repeating structure because the attribute maxOccurs="unbounded". This means it has no upper limit. The repeating group is the type "Forecast" which can repeat any number of times with no upper limit.



Identify repeating group within Web Service response (6 of 6)

- Locate the complexType with name=”Forecast”
- Find element that will always be present
  - Use element with minOccurs=“1”

Web Service Meta Data Importer
Address: http://wsf.cdyne.com/WeatherWS/Weather.asmx?wsdl
Web Service Explorer
Weather (filtered)
WeatherSoap
Documentation
Service
Weather
(not documented)
Import
Close
Ready
Connected to: PARENT.SWG.USMA.IBM.COM\altus

```
      <s:element minOccurs="0" maxOccurs="1" name="City" type="s:string" />
      <s:element minOccurs="0" maxOccurs="1" name="WeatherStationCity" type="s:string" />
      <s:element minOccurs="0" maxOccurs="1" name="ForecastResult"
        type="tns:ArrayOfForecast" />
    </s:sequence>
  </s:complexType>
- <s:complexType name="ArrayOfForecast">
  - <s:sequence>
      <s:element minOccurs="0" maxOccurs="unbounded" name="Forecast" nillable="true"
        type="tns:Forecast" />
    </s:sequence>
  </s:complexType>
- <s:complexType name="Forecast">
  - <s:sequence>
      <s:element minOccurs="1" maxOccurs="1" name="Date" type="s:dateTime" />
      <s:element minOccurs="1" maxOccurs="1" name="WeatherID" type="s:short" />
      <s:element minOccurs="0" maxOccurs="1" name="Desciption" type="s:string" />
      <s:element minOccurs="1" maxOccurs="1" name="Temperatures" type="tns:temp" />
      <s:element minOccurs="1" maxOccurs="1" name="ProbabilityOfPrecipiation" type="tns:POP" />
    </s:sequence>
  </s:complexType>
- <s:complexType name="temp">
  - <s:sequence>
      <s:element minOccurs="0" maxOccurs="1" name="MorningLow" type="s:string" />
      <s:element minOccurs="0" maxOccurs="1" name="DaytimeHigh" type="s:string" />
    </s:sequence>
  </s:complexType>
- <s:complexType name="POP">
  - <s:sequence>
      <s:element minOccurs="0" maxOccurs="1" name="Nighttime" type="s:string" />
      <s:element minOccurs="0" maxOccurs="1" name="Daytime" type="s:string" />
```

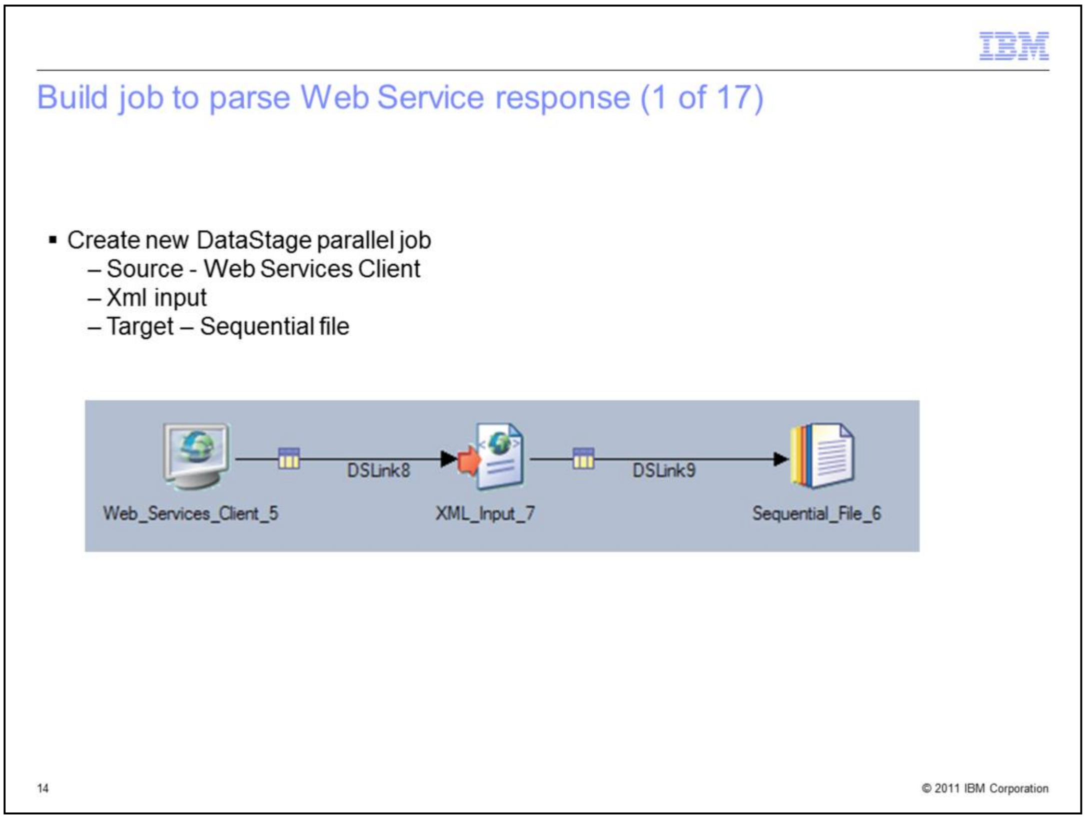
13

© 2011 IBM Corporation

Next, scroll one more time through the types and locate the complexType with name=”Forecast”. This is the repeating structure found on the previous slide. The type is highlighted in yellow on this slide.

When a DataStage job is built, a column in the XML Input Stage needs to be defined that is mapped to an element within the XML document. The XML document will always be present to trigger a new row to be created. Look for an element with minOccurs=“1”. For this structure, the 'Date' element meets this requirement. It has minOccurs=1, which means that it will always be present for each instance of this repeating group of elements. You will see in the upcoming slides how this trigger is applied.

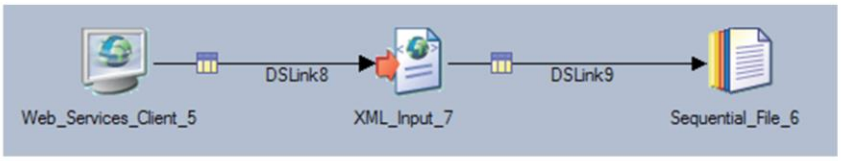
Before proceeding, click the “close” button in the lower right corner of the Web Service Metadata Importer.



IBM

Build job to parse Web Service response (1 of 17)

- Create new DataStage parallel job
  - Source - Web Services Client
  - Xml input
  - Target – Sequential file



14

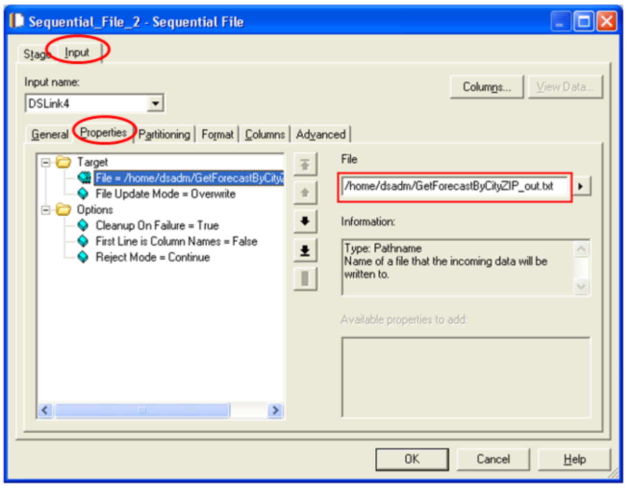
© 2011 IBM Corporation

In the DataStage Designer, create a new parallel job. On the job canvas, add a web services client as the source, followed by a XML Input stage and then a sequential file as the target. The Web Services Client stage (WSClientPX) and the XML Input stage (XMLInputPX) are located within the Real Time group in the Designer palette.

IBM

Build job to parse Web Service response (2 of 17)

- Define target file for sequential file stage



Input name: DSLink4

Target
- File = /home/dsadm/GetForecastByCity
- File Update Mode = Overwrite

Options
- Cleanup On Failure = True
- First Line is Column Names = False
- Reject Mode = Continue

File: /home/dsadm/GetForecastByCityZIP_out.txt

Information:
Type: Pathname
Name of a file that the incoming data will be written to.

Available properties to add:

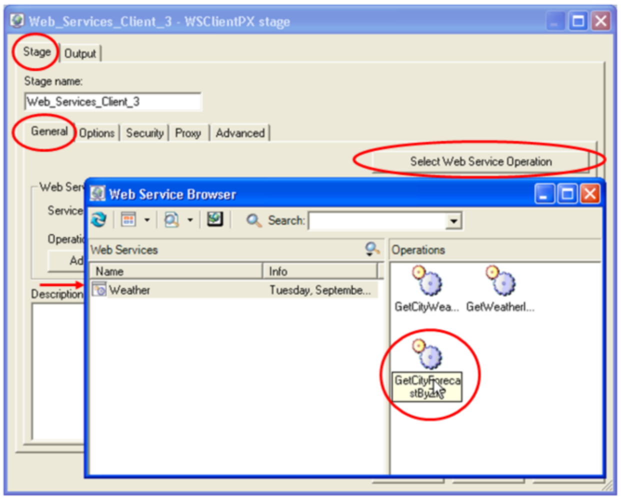
15

© 2011 IBM Corporation

Next, double click the sequential file stage. Click the Input tab at the top and then on the Properties tab. Edit the path for the File property and provide the full path to the location on your DataStage server where you want to write the file. This is a small file. After typing the path, press the Enter key on the keyboard to submit. It is not necessary to change any other options; the default values should be used. Press OK at the bottom of the dialog box to save the changes.

Build job to parse Web Service response (3 of 17)

- Select GetCityForcastByZIP web service operation



16

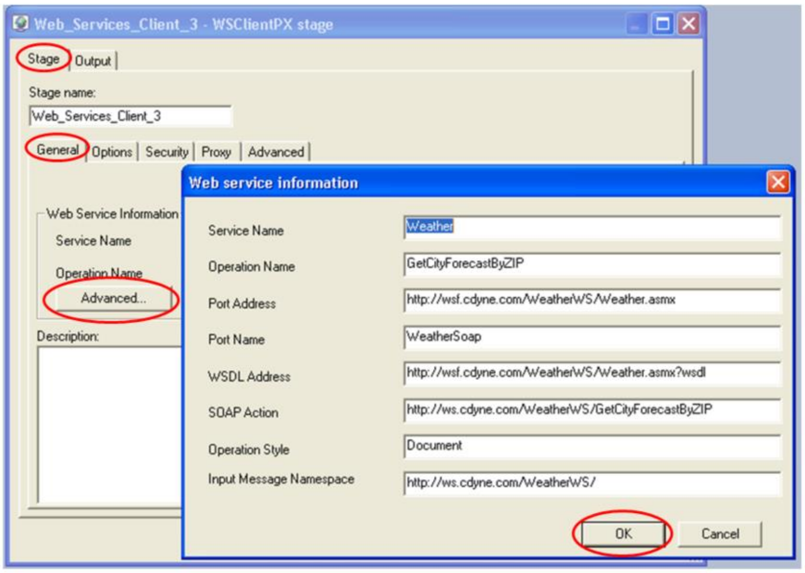
© 2011 IBM Corporation

Open the Web Services Client stage. Select General tab. Click the 'Select Web Service Operation' button. In the Web Service Browser window, select the 'Weather' web service in the left pane, and the operations are listed in the right pane. Double click the 'GetCityForecastByZIP' operation in the right pane.

IBM

Build job to parse Web Service response (4 of 17)

- Verify Web services information populated



Stage name: Web_Services_Client_3

Web Service Information
Service Name
Operation Name

Web service information

Service Name	Weather
Operation Name	GetCityForecastByZIP
Port Address	http://wsf.cdyne.com/WeatherWS/Weather.asmx
Port Name	WeatherSoap
WSDL Address	http://wsf.cdyne.com/WeatherWS/Weather.asmx?wsdl
SOAP Action	http://ws.cdyne.com/WeatherWS/GetCityForecastByZIP
Operation Style	Document
Input Message Namespace	http://ws.cdyne.com/WeatherWS/

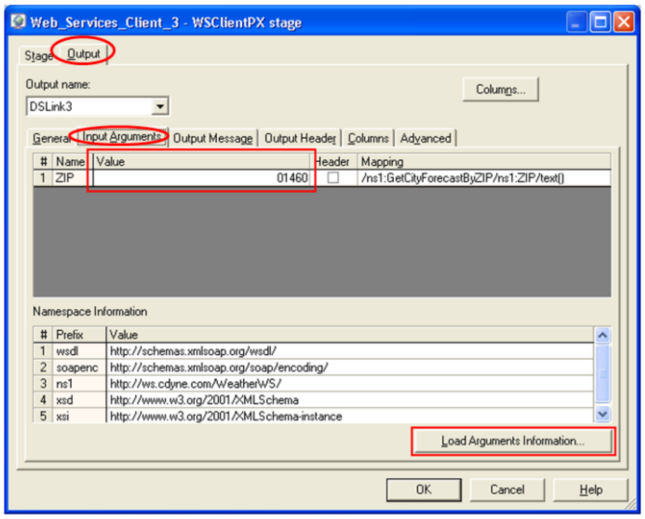
17

© 2011 IBM Corporation

Next, verify the Web Service information is populated. Click the Advanced button to view the Web Service Information. If the information is not populated properly, go back and verify the previous steps were completed correctly. Click OK when finished.

Build job to parse Web Service response (5 of 17)

- Load namespace and define input arguments
- Enter valid US Zip Code



#	Name	Value	Header	Mapping
1	ZIP	01460	☐	/ns1:GetCityForecastByZIP/ns1:ZIP/text()

#	Prefix	Value
1	wsdl	http://schemas.xmlsoap.org/wsdl/
2	soapenc	http://schemas.xmlsoap.org/soap/encoding/
3	ns1	http://ws.cdyne.com/WeatherWS/
4	xsd	http://www.w3.org/2001/XMLSchema
5	xsi	http://www.w3.org/2001/XMLSchema-instance

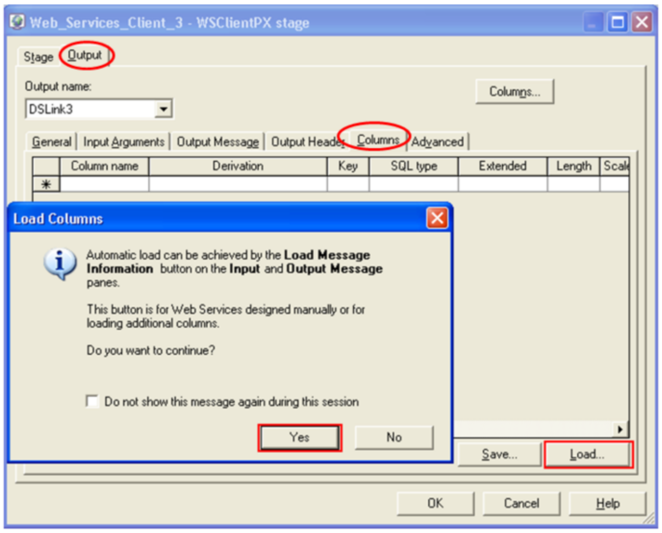
18 © 2011 IBM Corporation

Next, click the Output tab at the top and then on the Input Arguments tab. Press the Load Arguments Information button and the grid for Namespace information should be automatically populated. In the grid row for ZIP argument, enter a valid US zip code into the Value column, such as 01460 for Littleton, Massachusetts.

IBM

Build job to parse Web Service response (6 of 17)

▪ Manually load column definition to hold entire SOAP message body



Automatic load can be achieved by the **Load Message Information** button on the **Input** and **Output Message** panes.

This button is for Web Services designed manually or for loading additional columns.

Do you want to continue?

☐ Do not show this message again during this session

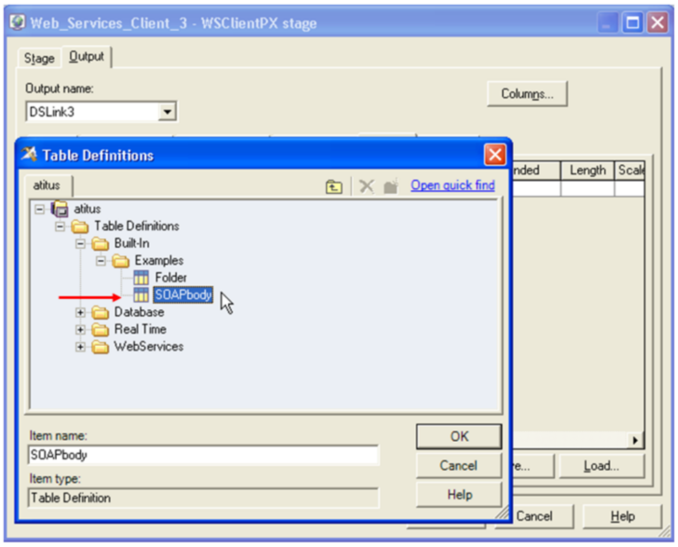
19

© 2011 IBM Corporation

With the Output tab still selected at the top, select the Columns tab. Press the Load button and a Load Columns message box will appear, confirming if you want to proceed with manual load. Click Yes.

Build job to parse Web Service response (7 of 17)

- Locate SOAPbody definition

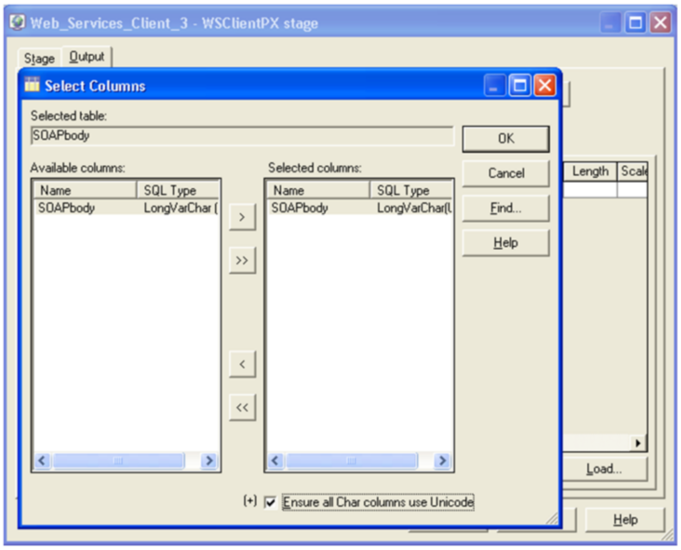


20 © 2011 IBM Corporation

Browse the Table definitions and expand the Built-in folder and then the Examples folder. Double click the SOAPbody definition to select it.

Build job to parse Web Service response (8 of 17)

- Manually load column definition to hold entire SOAP message body



Name	SQL Type
SOAPbody	LongVarChar (

Name	SQL Type
SOAPbody	LongVarChar(L

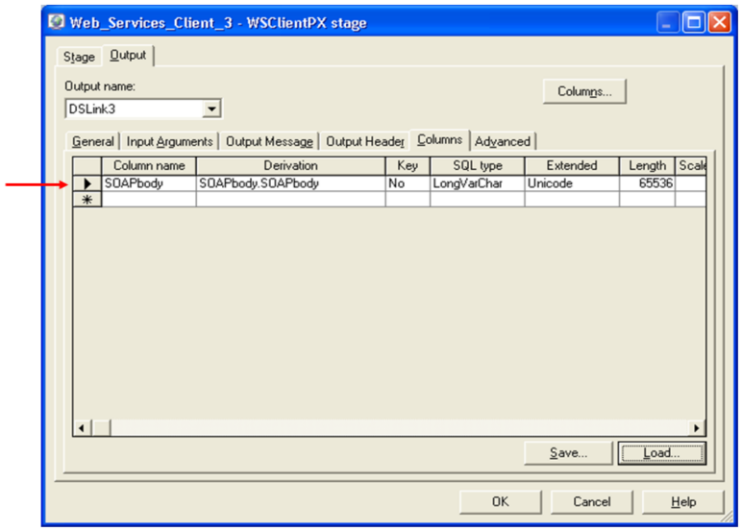
(+) [✓] Ensure all Char columns use Unicode

21

© 2011 IBM Corporation

On the Select Columns dialog, SOAPbody should appear in both panels. Press the Ok button to add the column.

Build job to parse Web Service response (9 of 17)

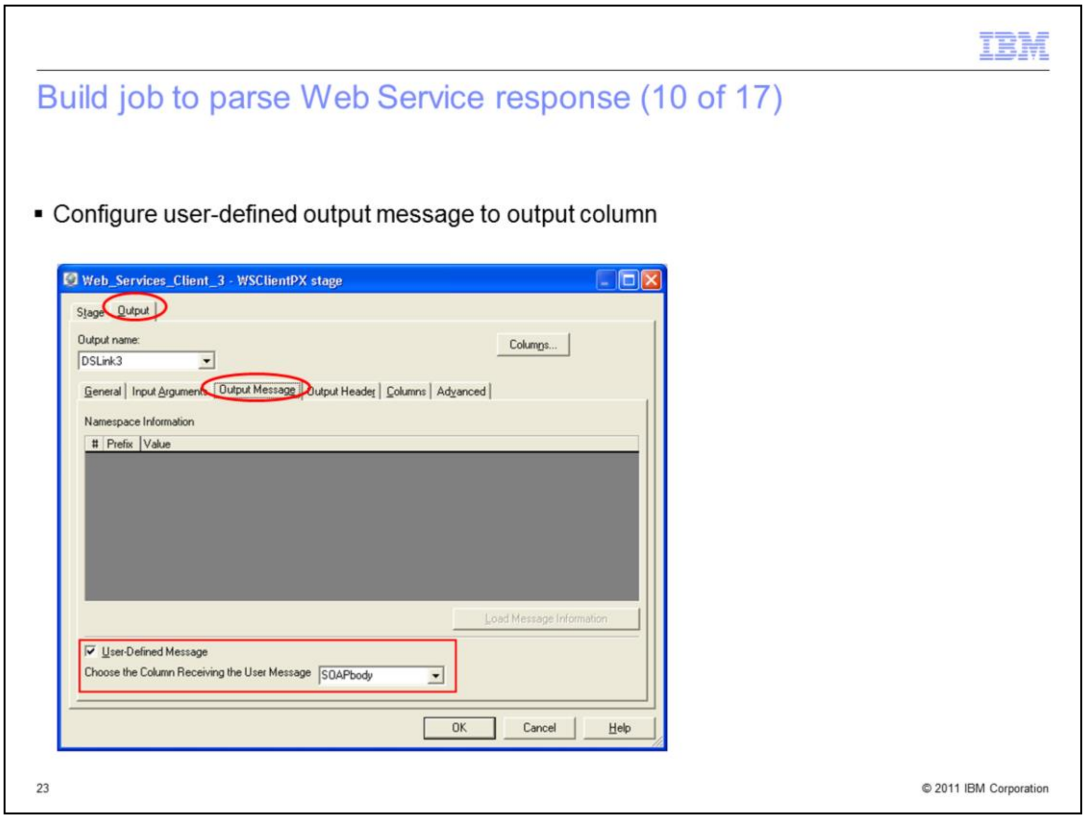


	Column name	Derivation	Key	SQL type	Extended	Length	Scale
▶	SOAPbody	SOAPbody.SOAPbody	No	LongVarChar	Unicode	65536	
*							

22

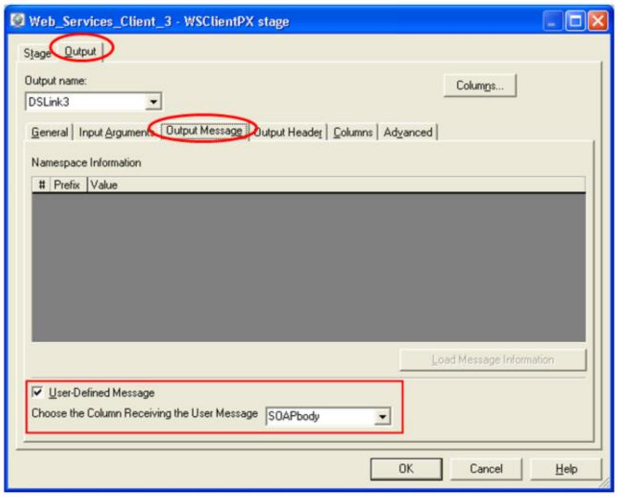
© 2011 IBM Corporation

The columns grid should now be populated with a single column, SOAPBody. This column will hold the entire XML contents of the SOAP Body response from the web service.



Build job to parse Web Service response (10 of 17)

- Configure user-defined output message to output column



Output name: DSLink3

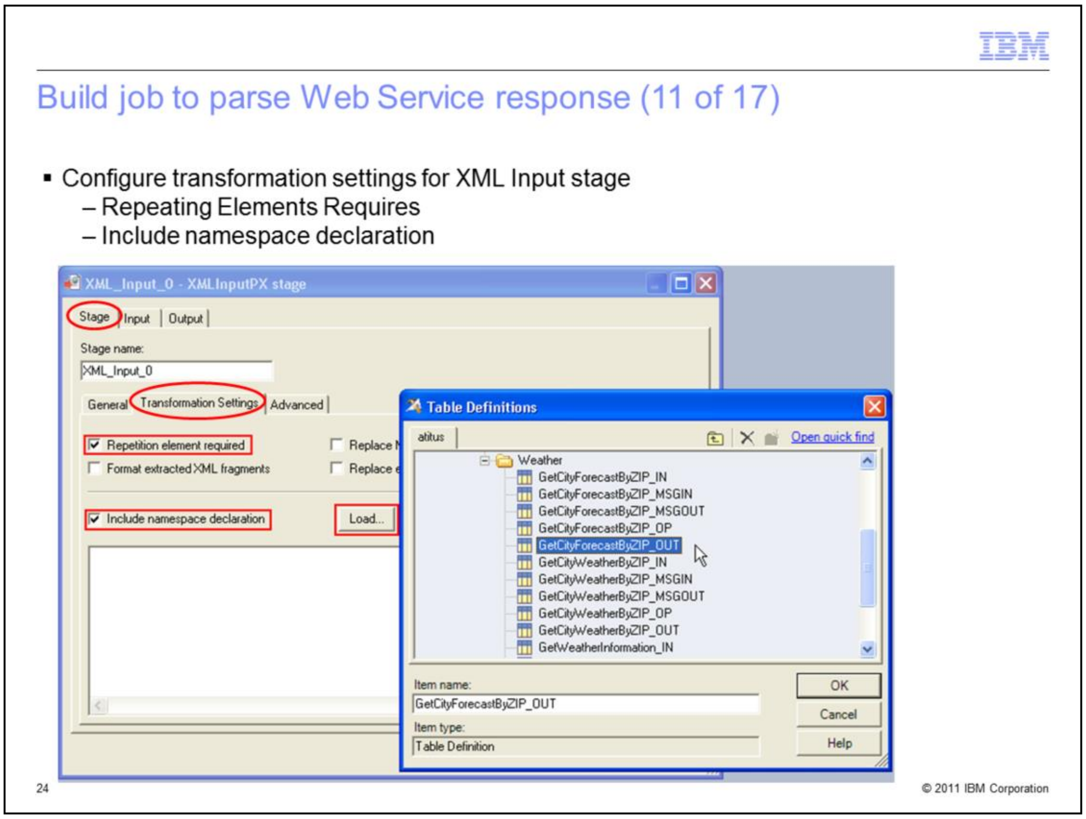
☑ User-Defined Message

Choose the Column Receiving the User Message SOAPbody

23

© 2011 IBM Corporation

With the Output tab still selected at the top, select the Output Message tab. DO NOT load the namespace information. Instead, click the box for User-Defined Message. In the drop down menu, select the SOAPBody column you configured in the previous slide. Close the Web Service Stage by pressing the OK button.



Build job to parse Web Service response (11 of 17)

- Configure transformation settings for XML Input stage
  - Repeating Elements Requires
  - Include namespace declaration

XML_Input_0 - XMLInputPX stage

Stage | Input | Output

Stage name: XML_Input_0

General | Transformation Settings | Advanced

- [✓] Repetition element required
- [ ] Format extracted XML fragments
- [✓] Include namespace declaration
- [ ] Replace N
- [ ] Replace e

Load...

Table Definitions

atitus

Open quick find

- Weather
  - GetCityForecastByZIP_IN
  - GetCityForecastByZIP_MSGIN
  - GetCityForecastByZIP_MSGOUT
  - GetCityForecastByZIP_OP
  - GetCityForecastByZIP_OUT
  - GetCityWeatherByZIP_IN
  - GetCityWeatherByZIP_MSGIN
  - GetCityWeatherByZIP_MSGOUT
  - GetCityWeatherByZIP_OP
  - GetCityWeatherByZIP_OUT
  - GetWeatherInformation_IN

Item name: GetCityForecastByZIP_OUT

Item type: Table Definition

OK | Cancel | Help

24

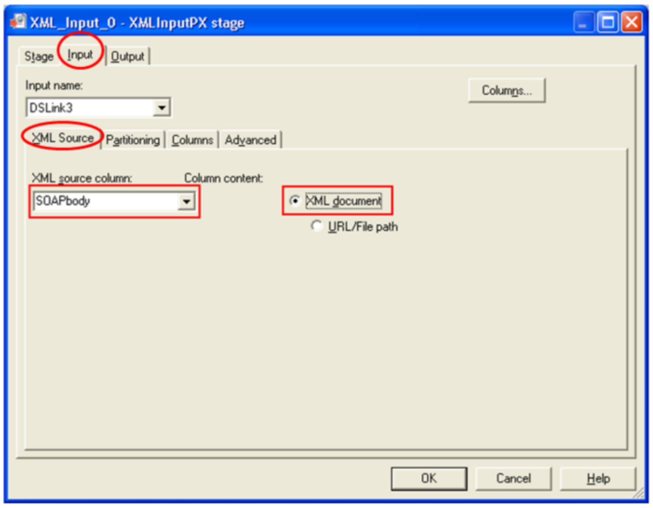
© 2011 IBM Corporation

Next, open the XML Input stage. With the Stage tab selected at the top, select the Transformation Settings tab. As displayed on this slide, check the boxes for Repetition Element Required and Include Namespace declaration. Ensure the rest of the boxes are unchecked.

Click the Load button to import the namespace information. Browse the repository tree under Table definitions and then the WebServices folder. Expand Weather, and locate the GetCityForecastByZIP_OUT definition. Double click to select it.

Build job to parse Web Service response (12 of 17)

- Configure XML Input stage source
  - Select XML Source column – SOAPbody
  - Select XML Document


XML_Input_0 - XMLInputPX stage

Input name: DSLink3

XML source column: SOAPbody

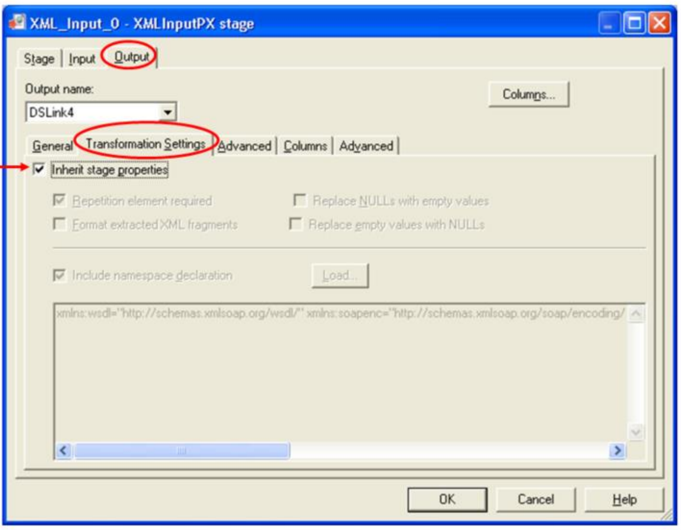
Column content: [✓ XML document] URL/File path

© 2011 IBM Corporation

Select the Input tab at the top, then select the XML Source tab. In the drop down menu for XML Source column, select SOAPBody. For the Column content option, choose XML document.

Build job to parse Web Service response (13 of 17)

- Configure output transformation settings
  - Inherit stage properties



XML_Input_0 - XMLInputPX stage

Stage | Input | Output

Output name: DSLink4

General | Transformation Settings | Advanced | Columns | Advanced

- [✓] Inherit stage properties
- [✓] Repetition element required
- [ ] Replace NULLs with empty values
- [ ] Format extracted XML fragments
- [ ] Replace empty values with NULLs
- [✓] Include namespace declaration

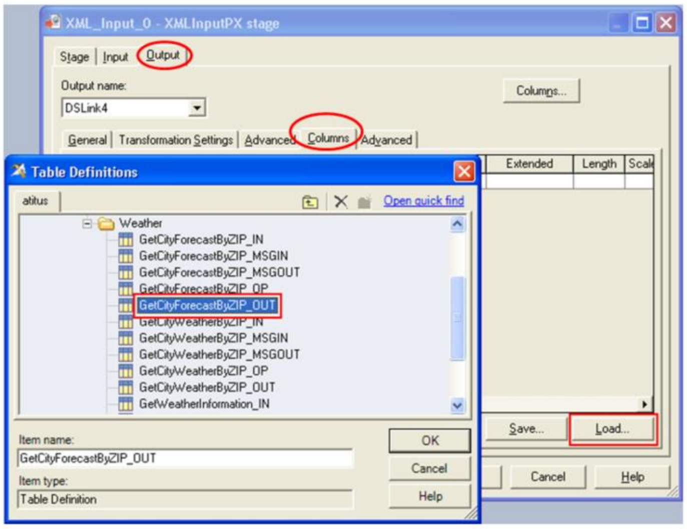
```
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/
```

© 2011 IBM Corporation

Select the Output tab at the top, then select the Transformation Settings tab. Check the box to Inherit stage properties. The remaining options should now appear disabled as displayed in the screen capture on this slide.

Build job to parse Web Service response (14 of 17)

- Load output columns



Item name: GetCityForecastByZIP_OUT

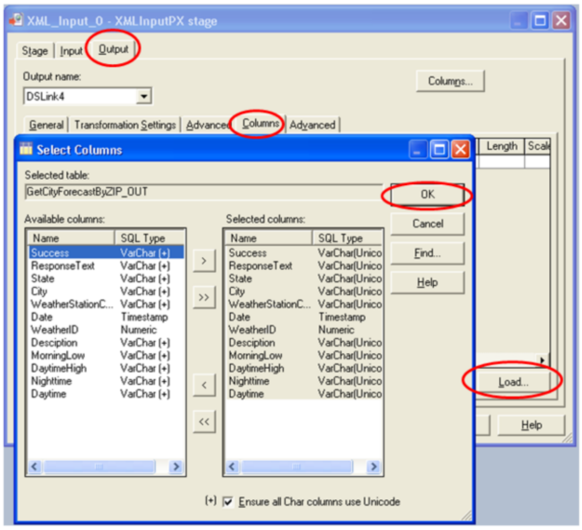
Item type: Table Definition

© 2011 IBM Corporation

With the Output tab selected at the top, select the Columns tab. Press the Load button to import columns. Browsing the repository, expand the Table definitions and WebServices folders, then the Weather folder. Double-click the GetCityForecastByZIP_OUT definition to import it.

Build job to parse Web Service response (15 of 17)

- Load output columns



XML_Input_0 - XMLInputPX stage

Output name: DSLink4

Select Columns

Selected table: GetCityForecastByZIP_OUT

Available columns:

Name	SQL Type
Success	VarChar (+)
ResponseText	VarChar (+)
State	VarChar (+)
City	VarChar (+)
WeatherStationC...	VarChar (+)
Date	Timestamp
WeatherID	Numeric
Desciption	VarChar (+)
MorningLow	VarChar (+)
DaytimeHigh	VarChar (+)
Nighttime	VarChar (+)
Daytime	VarChar (+)

Selected columns:

Name	SQL Type
Success	VarChar(Unico
ResponseText	VarChar(Unico
State	VarChar(Unico
City	VarChar(Unico
WeatherStationC...	VarChar(Unico
Date	Timestamp
WeatherID	Numeric
Desciption	VarChar(Unico
MorningLow	VarChar(Unico
DaytimeHigh	VarChar(Unico
Nighttime	VarChar(Unico
Daytime	VarChar(Unico

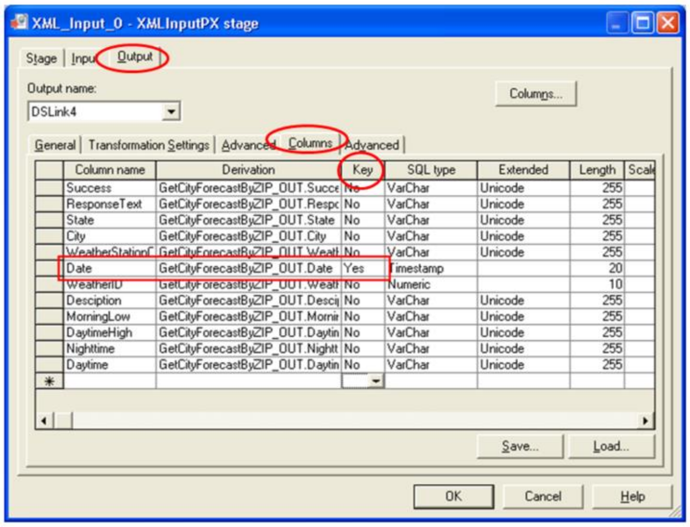
(+) ☑ Ensure all Char columns use Unicode

© 2011 IBM Corporation

In the Select Columns dialog all of the columns are listed on both the left and right side. Click the OK button to load all of the columns.

Build job to parse Web Service response (16 of 17)

- Configure “Date” as repeating element
  - Set Key = “Yes”



Output name: DSLink4

Column name	Derivation	Key	SQL type	Extended	Length	Scale
Success	GetCityForecastByZIP_OUT.Succe	No	VarChar	Unicode	255	
ResponseText	GetCityForecastByZIP_OUT.Respc	No	VarChar	Unicode	255	
State	GetCityForecastByZIP_OUT.State	No	VarChar	Unicode	255	
City	GetCityForecastByZIP_OUT.City	No	VarChar	Unicode	255	
WeatherStationC	GetCityForecastByZIP_OUT.Weatl	No	VarChar	Unicode	255	
Date	GetCityForecastByZIP_OUT.Date	Yes	Timestamp		20	
WeatherID	GetCityForecastByZIP_OUT.Weatl	No	Numeric		10	
Desciption	GetCityForecastByZIP_OUT.Descij	No	VarChar	Unicode	255	
MorningLow	GetCityForecastByZIP_OUT.Mornir	No	VarChar	Unicode	255	
DaytimeHigh	GetCityForecastByZIP_OUT.Daytin	No	VarChar	Unicode	255	
Nighttime	GetCityForecastByZIP_OUT.Nightt	No	VarChar	Unicode	255	
Daytime	GetCityForecastByZIP_OUT.Daytin	No	VarChar	Unicode	255	

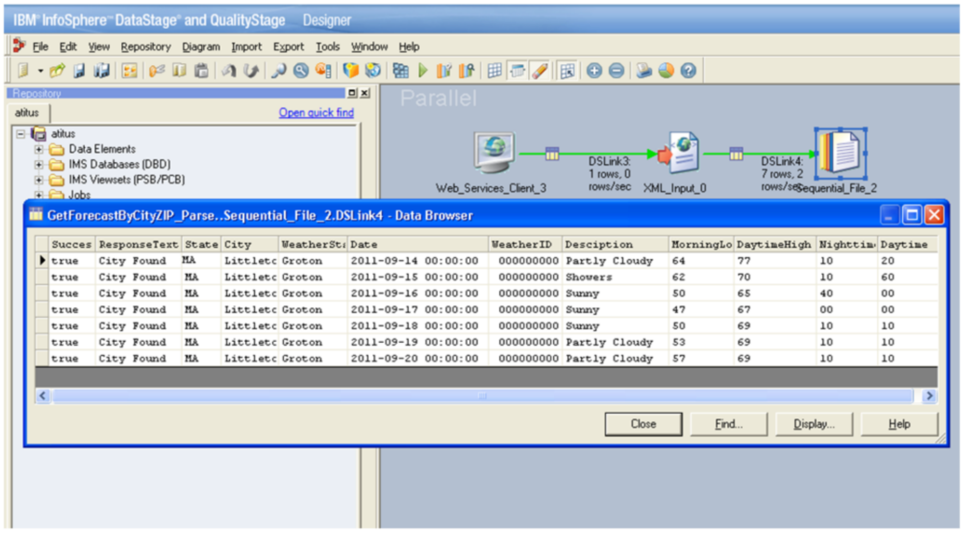
© 2011 IBM Corporation

The grid should now be populated with all the columns from the previous step.

On slide 13, you identified a repeating element, “date”, which will always be present. You determined this from examining the WSDL document. Locate the column named 'Date' within the grid. For the Key property, choose 'Yes' from the drop down menu. This step configures Date as the repeating element.

Build job to parse Web Service response (17 of 17)

- Save, compile, and run job

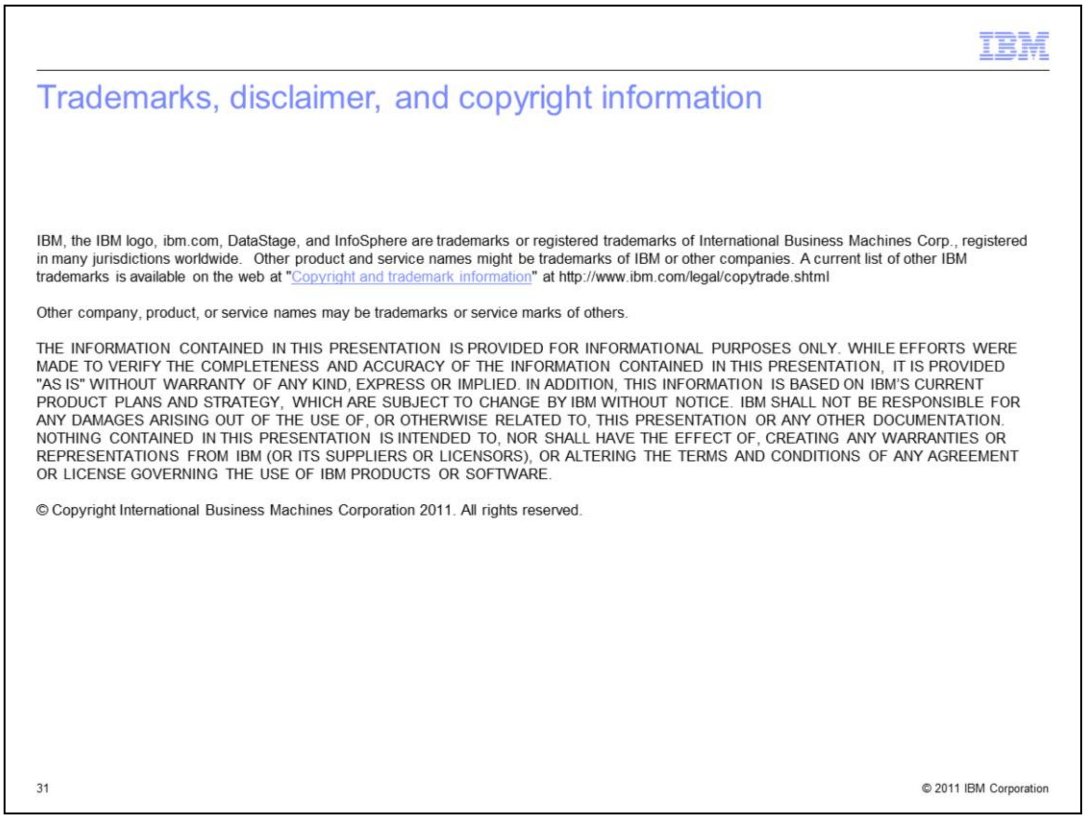

IBM InfoSphere DataStage and QualityStage Designer
File Edit View Repository Diagram Import Export Tools Window Help
Repository
atitus
Open quick find
atitus
Data Elements
IMS Databases (DBD)
IMS Viewsets (PSB/PCB)
Jobs
Parallel
Web_Services_Client_3
DSLink3: 1 rows, 0 rows/sec
XML_Input_0
DSLink4: 7 rows, 2 rows/sec
Sequential_File_2
GetForecastByCityZIP_Parse..Sequential_File_2.DSLink4 - Data Browser
Close
Find...
Display...
Help

Succes	ResponseText	State	City	WeatherSt:	Date	WeatherID	Desciption	MorningLo	DaytimeHigh	Nighttim	Daytime
true	City Found	MA	Littletc	Groton	2011-09-14 00:00:00	000000000	Partly Cloudy	64	77	10	20
true	City Found	MA	Littletc	Groton	2011-09-15 00:00:00	000000000	Showers	62	70	10	60
true	City Found	MA	Littletc	Groton	2011-09-16 00:00:00	000000000	Sunny	50	65	40	00
true	City Found	MA	Littletc	Groton	2011-09-17 00:00:00	000000000	Sunny	47	67	00	00
true	City Found	MA	Littletc	Groton	2011-09-18 00:00:00	000000000	Sunny	50	69	10	10
true	City Found	MA	Littletc	Groton	2011-09-19 00:00:00	000000000	Partly Cloudy	53	69	10	10
true	City Found	MA	Littletc	Groton	2011-09-20 00:00:00	000000000	Partly Cloudy	57	69	10	10

© 2011 IBM Corporation

Save and compile the job. To enable performance statistics, right click anywhere on the blank area of the job canvas and select Show Performance Statistics. Run the job from Designer. The link should turn green and report one row processed as input to the XML stage and multiple rows written to the sequential file stage. Right click the Sequential file stage and view data to see the individual column values returned.

Note: You may find that the view data browser displays only one column of data at a time. This is because the maximum lengths for the fields are set to 255. To see more than one column at a time, use the mouse to resize the column widths in the view data browser.



Trademarks, disclaimer, and copyright information

IBM, the IBM logo, ibm.com, DataStage, and InfoSphere 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" 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 2011. All rights reserved.

© 2011 IBM Corporation