Processing SOAP Messages with the XML Parser Transformation
Overview

You can parse a SOAP message with an XML Parser transformation to retrieve the data from the SOAP message body. The SOAP message can come from MQ, JMS, a web service, or any application that writes a SOAP message to a flat file. Create the XML Parser transformation with a schema that describes each element in the SOAP message. The schema must include a SOAP envelope schema, a SOAP encoding schema, and namespaces for these schemas. The following example describes how to format the schema and create an XML Parser transformation.

Soap Envelope Schema

A SOAP envelope schema defines the SOAP message structure. The schema defines the complex types that define the basic structure of a SOAP document. You can download a standard SOAP schema from http://www.w3.org. You do not have to modify the schema, but you must be able to reference its location when you create the XML Parser transformation.

The SOAP envelope schema contains the following elements:

- **Envelope.** Root element of a SOAP message. The envelope element defines the message as a SOAP message.
- **Header.** Contains application specific information about the SOAP message. The header element is optional. If the SOAP envelope contains a header element, the header element must be the first child element of the envelope element.
- **Body.** Contains the SOAP message. The message is the data to pass to a web service. The SOAP envelope body can include elements from other namespaces.
- **Fault.** Stores error and status information for a SOAP message. If the fault element is present, it is the first element in the body.

Soap Encoding Schema

The SOAP encoding schema defines the attributes and datatypes that you can include in a SOAP document. It defines a set of rules for mapping datatypes, compound data structures, array types, and reference datatypes to XML datatypes. You can download a SOAP encoding schema from http://www.w3.org. You do not have to modify this schema, but you must be able to reference its location when you create the XML Parser transformation.
Example Schema

The following example shows an XML schema that defines order information for a company. The schema includes the elements that define each order, such as Order ID, Date Entered, and Customer ID. The order information is included in the body of a SOAP message.

You can import this schema and create an XML Parser transformation. The schema contains the schema location for the SOAP envelope and SOAP encoding schemas. Each schema has a namespace.

The schema also includes the XML schema namespace. The XML schema namespace identifies elements and simple types defined by the XML Schema language.

The following schema defines the orders for the company:

```xml
<?xml version = "1.0" encoding = "UTF-8"?>
<xs:schema xmlns = "http://namespace.weston.com/Products/Westonv1"
          targetNamespace = "http://namespace.weston.com/Products/Westonv1"
          xmlns:xs = "http://www.w3.org/2001/XMLSchema"
          xmlns:soapenv = "http://schemas.xmlsoap.org/soap/envelope/"
          xmlns:soap-enc = "http://schemas.xmlsoap.org/soap/encoding/"
          elementFormDefault = "qualified">

  <xs:import namespace = "http://schemas.xmlsoap.org/soap/encoding/"
              schemaLocation = "soap_encoding.xsd"/>

  <xs:import namespace = "http://schemas.xmlsoap.org/soap/envelope/"
              schemaLocation = "soap_env.xsd"/>

  <xs:element name="ORDERS">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="ORDER_ID">
          <xs:simpleType>
            <xs:restriction base="xs:integer">
            </xs:restriction>
          </xs:simpleType>
        </xs:element>

        <xs:element name="DATE_ENTERED">
          <xs:simpleType>
            <xs:restriction base="xs:date">
            </xs:restriction>
          </xs:simpleType>
        </xs:element>

        <xs:element name="DATE_PROMISED">
          <xs:simpleType>
            <xs:restriction base="xs:date">
            </xs:restriction>
          </xs:simpleType>
        </xs:element>

        <xs:element name="DATE_SHIPPED">
          <xs:simpleType>
            <xs:restriction base="xs:date">
            </xs:restriction>
          </xs:simpleType>
        </xs:element>

        <xs:element name="CUSTOMER_ID">
          <xs:simpleType>
            <xs:restriction base="xs:integer">
            </xs:restriction>
          </xs:simpleType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

</xs:schema>
```
The following XML statements list the namespaces for XML schema, SOAP envelope, and SOAP encoding schemas:

```xml
xmlns:xsd = "http://www.w3.org/2001/XMLSchema"
xmns:soapenv = "http://schemas.xmlsoap.org/soap/envelope/
xmlns:soap-enc = "http://schemas.xmlsoap.org/soap/encoding/"
```

The following statements list the schema locations for the SOAP envelope and SOAP encoding schemas:

```xml
<xs:import namespace = "http://schemas.xmlsoap.org/soap/envelope/" schemaLocation = "soap_env.xsd" />
<xs:import namespace = "http://schemas.xmlsoap.org/soap/encoding/" schemaLocation = "soap_enc.xsd"/>
```

Note: The schema names might vary from the names in this example.

**Creating the XML Parser Transformation**

When you create the XML Parser transformation, the Designer shows the schemas that define the SOAP message:

You do not need all the elements from these schemas in the XML Parser definition. When you create the transformation, do not generate default XML views. Manually create XML views in the XML Editor and select the elements for each view.
The following figure shows the option to skip creating default views:

In this example, the XML Parser transformation has the DataInput view and an Orders view. Manually create the Orders view in the XML Editor. The Designer creates the DataInput view by default.

**Mapping Example**

The following figure shows a mapping that includes the XML Parser transformation:
The mapping contains the following objects:

- **Flat file source.** Represents each SOAP message as a string of 64,000 bytes.
- **Source qualifier.** Converts source datatypes to transformation datatypes.
- **XML Parser transformation.** Receives the SOAP message in the DataInput port, parses the SOAP message, and returns the order information.
- **Flat file target.** Receives the order information.

**Summary**

You can parse SOAP messages with an XML Parser transformation in a standard workflow. You must create the XML Parser transformation with a schema that describes all of the SOAP message elements. The schema that describes the SOAP message body must also include the SOAP encoding schema and the SOAP envelope schema.

**Author**

Ellen Chandler  
Principal Technical Writer