SOAP Routing and Processing Concepts

Marlon Pierce, Bryan Carpenter, Geoffrey Fox
Community Grids Lab
Indiana University
mpierce@cs.indiana.edu
SOAP Processing Assumptions

- SOAP assumes messages have an originator, one or more ultimate receivers, and zero or more intermediaries.
- The reason is to support distributed message processing.
- Implementing this message routing is out of scope for SOAP.
  - Assume each node is a Tomcat server or JMS broker.
- That is, we can go beyond client-server messaging.
Processing and SOAP Structure

• SOAP processing rules are directly related to the SOAP message envelope:
  – The body is only for final recipients (“ultimateReceivers”)
  – Header sections may be processed by one or more intermediaries as well as final recipient nodes.
  – SOAP headers are the extensibility elements for defining other features.

• The Header therefore has three optional attributes:
  – Role (called actor in SOAP 1.0 and 1.1): Determines if a header should process a particular header.
  – mustUnderstand: If set to “true”, the node must know how to process the header.
  – Relay: Indicates whether or not an unprocessed header block should be forwarded.
Roles, Understanding, and Relays

- Role?
- Yes: must Understand
- No: Forward Header

- Understand?
- Yes: Process Header
- No: Relay?

- Relay?
- Yes: Remove Header
- No: Forward Header
SOAP Intermediaries

• Forwarding Intermediaries:
  – Are used to route messages to other SOAP nodes, based on header information.
  – May do additional processing as described in a SOAP header.

• Active Intermediaries
  – Act as forwarding intermediaries
  – Do additional processing to a message that is NOT described in any of the message headers.
    • For example, may insert additional headers needed for additional processing, or may encrypt parts of the message for security.
SOAP Forwarding Intermediaries

• A forwarding intermediary must do the following:
  – Process any headers as required by its role and mustUnderstand.
  – Relay any unprocessed headers.
• It is also required by the specification to
  – Remove all processed header blocks.
  – Remove all unprocessed and non-relayable header blocks.
• Forwarding Intermediaries may also insert new headers.
  – This may be a reinsertion of a processed header, for example.
  – Oddly, there seems to be no built-in way to label a header as “persistent”.
• Next we will see how these nodes relate to parts of the SOAP message.
Example Header from SOAP Primer

```xml
<?xml version='1.0' ?>
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope">
  <env:Header>
    <m:reservation xmlns:m="..."
      env:role="http://www.w3.org/2003/05/soap-envelope/role/next"
      env:mustUnderstand="true">
      <m:reference>uuid:093a2da1-q345-739r-ba5d-pqff98fe8j7d</m:reference>
      <m:dateAndTime>2001-11-29T13:20:00.000-05:00</m:dateAndTime>
    </m:reservation>
    <n:passenger xmlns:n="..."
      env:role="http://www.w3.org/2003/05/soap-envelope/role/next"
      env:mustUnderstand="true">
      <n:name>Åke Jógván Øyvind</n:name>
    </n:passenger>
  </env:Header>
</env:Envelope>
```
What This Header Means

• The actual content of the header is an example of transaction and session state information needed to carry out a set of multiple, linked interactions to book an airline flight.
  – Don’t worry about this.
• The role attributes are “next” for both header entries.
  – This means all intermediaries and the final recipient should process the header if they can.
• The “mustUnderstand” attribute is also true, so if a node does not know how to process this header, it must throw a fault back to the originator.
SOAP Nodes and Roles

- Originators, intermediaries, and receivers of SOAP messages are all called **SOAP Nodes**.
  - Each node is labeled with a URI
- For a particular message, the **Node** can act in one or more **SOAP Roles**.
  - Each role is labeled with a URI
  - The following table lists predefined roles.
- You can define your own roles
  - “Log message” role
  - “Check authorization” role
- When a node receives a message, it must examine the message for a role definition and process the headers as required.
- The SOAP specification itself does not specify how you assign a role to a node.
  - This depends upon the implementation.
# Standard SOAP 1.2 Roles

<table>
<thead>
<tr>
<th>Short-name</th>
<th>Full Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>next</td>
<td>&quot;<a href="http://www.w3.org/2003/05/soap-envelope/role/next">http://www.w3.org/2003/05/soap-envelope/role/next</a>&quot;</td>
<td>Each SOAP intermediary and the ultimate SOAP receiver MUST act in this role.</td>
</tr>
<tr>
<td>none</td>
<td>&quot;<a href="http://www.w3.org/2003/05/soap-envelope/role/none">http://www.w3.org/2003/05/soap-envelope/role/none</a>&quot;</td>
<td>SOAP nodes MUST NOT act in this role. That is, the header block should not be directly processed. It may carry supplemental information.</td>
</tr>
<tr>
<td>ultimateReceiver</td>
<td>&quot;<a href="http://www.w3.org/2003/05/soap-envelope/role/ultimateReceiver">http://www.w3.org/2003/05/soap-envelope/role/ultimateReceiver</a>&quot;</td>
<td>The ultimate receiver MUST act in this role. If no role is specified in a header, it is treated as being in this role.</td>
</tr>
</tbody>
</table>
Understanding Headers

• SOAP role definitions may require SOAP nodes to process headers.
• In a distributed processing model, it is possible that certain nodes will not have the required capability to process the header.
• We must therefore identify a header as optional or required.
• We do this with the mustUnderstand attribute.
  – If true, the node must process the header or else stop processing and return a Fault message.
  – If false, the header is optionally processed, depending on the role of the node. This is the default value.
• The SOAP specification requires that a node identify all required headers and determine if they are understood before any processing takes place.
Relaying SOAP Messages

• As we have seen, SOAP headers may or may not be processed by an intermediate node.
  – mustUnderstand and role attributes determine this.
  – For example, if the role is “ultimateReceiver” than intermediaries don’t process this header.

• Processed headers must be removed from the SOAP message before forwarding.

• But there are times when a node role indicates processing, but processing is optional.
  – Role is “next” but mustUnderstand=“false”

• What happens to these headers?

• SOAP 1.2 defines an optional attribute called “relay” to resolve this.
  – Relay is a boolean attribute.
# Summary of Relay Forwarding

<table>
<thead>
<tr>
<th>Role</th>
<th>Assumed</th>
<th>Header block</th>
<th>Header Forwarded?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Short-name</td>
<td></td>
<td><strong>Understood &amp; Processed</strong></td>
<td></td>
</tr>
<tr>
<td>next</td>
<td>Yes</td>
<td>Yes</td>
<td>No, unless reinserted</td>
</tr>
<tr>
<td></td>
<td></td>
<td>No</td>
<td>No, unless relay = &quot;true&quot;</td>
</tr>
<tr>
<td>user-defined</td>
<td>Yes</td>
<td>Yes</td>
<td>No, unless reinserted</td>
</tr>
<tr>
<td></td>
<td></td>
<td>No</td>
<td>No, unless relay = &quot;true&quot;</td>
</tr>
<tr>
<td></td>
<td>No</td>
<td>n/a</td>
<td>Yes</td>
</tr>
<tr>
<td>ultimateReceiver</td>
<td>Yes</td>
<td>Yes</td>
<td>n/a</td>
</tr>
<tr>
<td></td>
<td></td>
<td>No</td>
<td>n/a</td>
</tr>
<tr>
<td>none</td>
<td>No</td>
<td>n/a</td>
<td>Yes</td>
</tr>
</tbody>
</table>
SOAP + HTTP
Putting SOAP into HTTP

• Assume that I know the port of a particular HTTP server that speaks SOAP.
  – Obtained from WSDL through UDDI
• Then I can easily construct an HTTP message with a SOAP payload.
• Then write the message to the remote socket.

POST /axis/service/echo
HTTP/1.0
Host: www.myservice.com
Content-Type: text/xml; charset=“utf-8”
Content-Length: nnn

<SOAP:Envelope>
  ...
</SOAP:Envelope>
What Does It Mean?

- The POST line specifies that we will use the POST method and assume HTTP 1.0 (not HTTP 1.1).
  - /axis/services/echo is the relative path part of the URL.
  - Host is in on a separate line.
- Host: specifies the name of the host.
- Content-Type: Type of content we are sending.
  - We must use text/xml for SOAP.
  - In general these are called mime-types.
- Content-Length: number of characters in the HTTP payload.