SOAP Attachments

SOAP attachments are used to transport data outside of the SOAP envelope. This saves some space because Base64 encoding can be avoided. Additionally, XML parsers and composers don't have to serialize large chunks of data. Over the years several approaches have been designed to use MIME parts to achieve this goal. The Bridge supports two scenarios:

- SOAP with Attachments (SwA): This was the first approach. It it is very flexible because linking the actual SOAP payload with the MIME attachments is basically left to the user. Thus, the SOAP attachments must be explicitly modeled as <<SOAPAdapter>> input and output objects.
- MTOM with XOP: in order to avoid the problems of SwA, XML-binary Optimized Packaging (XOP) has been invented. This mechanism is basically transparent for the modeler. That is, the SOAP adapter can map XOP encoded messages directly to the output objects when parsing the SOAP response. Composing XOP message is not yet supported.

On this Page:

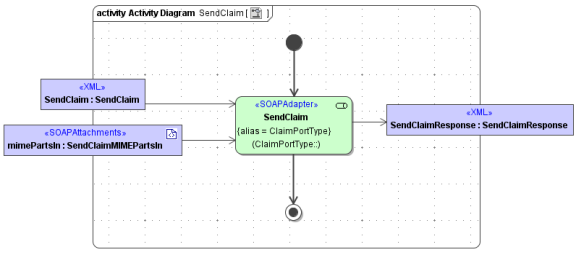
Related Pages:

- Accessing the SOAP Headers

Modeling SOAP Attachments

If the SOAP adapter shall send SOAP attachments, you have first to design a <<SOAPAttachments>> class containing all MIME parts that you want to attach to the SOAP message. For example, the **SendClaimMIMPartsIn** class contains two attributes **ClaimFiles** and **ClaimPhoto**. The first one stands for an array of MIME parts the latter one for a single MIME part. Attributes of <<SOAPAttachments>> classes must have either MIME parts or arrays of MIME parts. Typically, you get theses classes when importing the WSDL files (details see below).

This class can then be used to send the attachments as shown in the following activity diagram. Of course, receiving attachments works completely analogous. All MIME relevant information such as Content, Content-Type, or Content-ID must be set explicitly for **ClaimFiles** and **ClaimPhoto**.



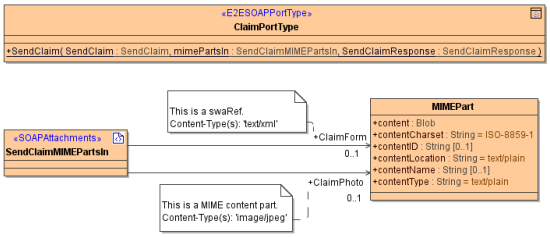
When receiving SOAP with Attachments, the modeler has to link the non-binary output data with the MIME parts. Typically fields such as Content-ID or Content-Location are used.

SOAP Attachments and WSDL

WSDL can define a binding to MIME parts. For maximal interoperability we follow the SOAP with Attachment Profile by Web Service Interoperability Organisation (WS-I). We recommend to look at the examples in this document. The following WSDL document is taken from the there. It is an example for a rpc/literal binding, but the other supported bindings work analogous.

```
 <?xml version="1.0"?>
<wsdl:definitions xmlns:types="http://example.com/mimetypes"
                  xmlns:ref="http://ws-i.org/profiles/basic/1.1/xsd"
                  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
                  xmlns:soapbind="http://schemas.xmlsoap.org/wsdl/soap/"
                  xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
                  xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
                  targetNamespace="http://example.com/mimewsdl"
                  xmlns:tns="http://example.com/mimewsdl">

    <wsdl:types>
        <xsd:schema targetNamespace="http://example.com/mimetypes"
                    xmlns:xsd="http://www.w3.org/2001/XMLSchema">

            <xsd:import namespace="http://ws-i.org/profiles/basic/1.1/xsd"
/>
            <xsd:complexType name="ClaimDetailType">
                <xsd:sequence>
                    <xsd:element name="Name" type="xsd:string"/>
                    <xsd:element name="ClaimForm" type="ref:swaRef"/>
                </xsd:sequence>
            </xsd:complexType>
        </xsd:schema>
    </wsdl:types>

    <wsdl:message name="ClaimIn">
        <wsdl:part name="ClaimDetail" type="types:ClaimDetailType"/>
        <wsdl:part name="ClaimPhoto" type="xsd:base64Binary"/>
    </wsdl:message>

    <wsdl:message name="ClaimOut">
        <wsdl:part name="ClaimRefNo" type="xsd:string"/>
    </wsdl:message>

    <wsdl:portType name="ClaimPortType">
        <wsdl:operation name="SendClaim">
            <wsdl:input message="tns:ClaimIn"/>
            <wsdl:output message="tns:ClaimOut"/>
        </wsdl:operation>
    </wsdl:portType>

    <wsdl:binding name="ClaimBinding" type="tns:ClaimPortType">
        <soapbind:binding style="rpc"
                          transport="http://schemas.xmlsoap.org/soap/http"
/>
        <wsdl:operation name="SendClaim">
            <soapbind:operation soapAction="http://example.com/soapaction"
/>
            <wsdl:input>
                <mime:multipartRelated>
                    <mime:part>
                        <soapbind:body use="literal"
                                       parts="ClaimDetail"
                                       namespace="http://example.com
/mimetypes"/>
                    </mime:part>
                    <mime:part>
                        <mime:content part="ClaimPhoto"
                                      type="image/jpeg"/>
                    </mime:part>
                </mime:multipartRelated>
            </wsdl:input>
            <wsdl:output>
                <soapbind:body use="literal"
                               namespace="http://example.com/mimetypes"/>
            </wsdl:output>
        </wsdl:operation>
 </wsdl:binding>
</wsdl:definitions>
```

When importing the above WSDL file, we get (explanation see below)



The WSDL port type **ClaimPortType** (line 34) has the operation **SendClaim** (line 35). Its input message is defined in line 25 containing two parts, **ClaimDetail** and **ClaimPhoto**. **ClaimDetail** is just a standard SOAP body but **ClaimPhoto** is a MIME content because this part is used in the MIME content binding in line 54. Thus, **ClaimPhoto** is put into the <<SOAPAttachmants>> class **SendCliamMIMEPartsIn**. Besides this explicit MIME binding, there is also an implicit reference to a MIME part using the SwA reference mechanism on line 19. This leads to an additional **MIMEPart** in the SOAP attachments class, namely **ClaimForm**.

The content types of the explicitly given MIME parts are found in the **mime:content** elements (e.g. line 55). However, more than one content type is possible. All allowed content types are written to the documentation (see figure above). For SwA references (swaRef) the content type is always **text/xml**. If the content type is **multipart/*** we use an array of MIME parts.

The resulting message may look like (also taken from the the WS-I profile):

```
 MIME-Version: 1.0
Content-Type: Multipart/Related; boundary=MIME_boundary; type=text/xml;
        start="<rootpart@example.com>"
Content-Description: This is the optional message description.

--MIME_boundary
Content-Type: text/xml; charset=UTF-8
Content-Transfer-Encoding: 8bit
Content-ID: <rootpart@example.com>

<?xml version='1.0' ?>
<SOAP-ENV:Envelope
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
    <SOAP-ENV:Body xmlns:types="http://example.com/mimetypes">
        <types:SendClaim>
            <ClaimDetail>
                <Name>...</Name>
                <ClaimForm>cid:claimform@example.com</ClaimForm>
            </ClaimDetail>
        </types:SendClaim>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

--MIME_boundary
Content-Type: text/xml
Content-Transfer-Encoding: 8bit
Content-ID: <claimform@example.com>

...claim form referenced by the swaRef...

--MIME_boundary
Content-Type: image/jpeg
Content-Transfer-Encoding: binary
Content-ID: <ClaimPhoto=4d7a5fa2-14af-451c-961b-5c3abf786796@example.com>

...MIME attachment of binary photograph...
--MIME_boundary--
```