

## SOAP Example (1): a general message

```
<?xml version="1.0" ?>
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope">
  <env:Header>
    <m:reservation xmlns:m="http://travelcompany.example.org/reservation"
      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="http://mycompany.example.com/employees"
      env:role="http://www.w3.org/2003/05/soap-envelope/role/next"
      env:mustUnderstand="true">
      <n:name>Åke Jógvind Øyvind</n:name>
    </n:passenger>
  </env:Header>
  <env:Body>
    <p:itinerary xmlns:p="http://travelcompany.example.org/reservation/travel">
      <p:departure>
        <p:departing>New York</p:departing>
        <p:arriving>Los Angeles</p:arriving>
        <p:departureDate>2001-12-14</p:departureDate>
        <p:departureTime>late afternoon</p:departureTime>
        <p:seatPreference>aisle</p:seatPreference>
      </p:departure>
      <p:return>
        <p:departing>Los Angeles</p:departing>
        <p:arriving>New York</p:arriving>
        <p:departureDate>2001-12-20</p:departureDate>
        <p:departureTime>mid-morning</p:departureTime>
        <p:seatPreference/>
      </p:return>
    </p:itinerary>
    <q:lodging xmlns:q="http://travelcompany.example.org/reservation/hotels">
      <q:preference>none</q:preference>
    </q:lodging>
  </env:Body>
</env:Envelope>
```

## SOAP Example (2): an RPC request

```
<?xml version="1.0" ?>
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope" >
<env:Header>
  <t:transaction
    xmlns:t="http://thirdparty.example.org/transaction"
    env:encodingStyle="http://example.com/encoding"
    env:mustUnderstand="true" >5</t:transaction>
</env:Header>
<env:Body>
  <m:chargeReservation
    env:encodingStyle="http://www.w3.org/2003/05/soap-encoding"
    xmlns:m="http://travelcompany.example.org/">
    <m:reservation xmlns:m="http://travelcompany.example.org/reservation">
      <m:code>FT35ZBQ</m:code>
    </m:reservation>
    <o:creditCard xmlns:o="http://mycompany.example.com/financial">
      <n:name xmlns:n="http://mycompany.example.com/employees">
        Åke Jógvan Øyvind
      </n:name>
      <o:number>123456789099999</o:number>
      <o:expiration>2005-02</o:expiration>
    </o:creditCard>
  </m:chargeReservation>
</env:Body>
</env:Envelope>
```

## SOAP Example (3-1): an RPC response (with a return value)

```
<?xml version="1.0" ?>
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope" >
  <env:Header>
    <t:transaction
      xmlns:t="http://thirdparty.example.org/transaction"
      env:encodingStyle="http://example.com/encoding"
      env:mustUnderstand="true">5</t:transaction>
  </env:Header>
  <env:Body>
    <m:chargeReservationResponse
      env:encodingStyle="http://www.w3.org/2003/05/soap-encoding"
      xmlns:rpc="http://www.w3.org/2003/05/soap-rpc"
      xmlns:m="http://travelcompany.example.org/">
      <rpc:result>m:status</rpc:result>
      <m:status>confirmed</m:status>
      <m:code>FT35ZBQ</m:code>
      <m:viewAt>
        http://travelcompany.example.org/reservations?code=FT35ZBQ
      </m:viewAt>
    </m:chargeReservationResponse>
  </env:Body>
</env:Envelope>
```

## SOAP Example (3-2): an RPC response (without a return value)

```
<?xml version="1.0" ?>
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope" >
  <env:Header>
    <t:transaction
      xmlns:t="http://thirdparty.example.org/transaction"
      env:encodingStyle="http://example.com/encoding"
      env:mustUnderstand="true">5</t:transaction>
  </env:Header>
  <env:Body>
    <m:chargeReservationResponse
      env:encodingStyle="http://www.w3.org/2003/05/soap-encoding"
      xmlns:m="http://travelcompany.example.org/">
      <m:code>FT35ZBQ</m:code>
      <m:viewAt>
        http://travelcompany.example.org/reservations?code=FT35ZBQ
      </m:viewAt>
    </m:chargeReservationResponse>
  </env:Body>
</env:Envelope>
```

## SOAP Example (4-1): an RPC response (fault)

```
<?xml version="1.0" ?>
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope"
               xmlns:rpc="http://www.w3.org/2003/05/soap-rpc">
  <env:Body>
    <env:Fault>
      <env:Code>
        <env:Value>env:Sender</env:Value>
        <env:Subcode>
          <env:Value>rpc:BadArguments</env:Value>
        </env:Subcode>
      </env:Code>
      <env:Reason>
        <env:Text xml:lang="en-US">Processing error</env:Text>
        <env:Text xml:lang="cs">Chyba zpracování</env:Text>
      </env:Reason>
      <env:Detail>
        <e:myFaultDetails
          xmlns:e="http://travelcompany.example.org/faults">
          <e:message>Name does not match card number</e:message>
          <e:errorCode>999</e:errorCode>
        </e:myFaultDetails>
      </env:Detail>
    </env:Fault>
  </env:Body>
</env:Envelope>
```

## SOAP Example (4-2): an RPC response (fault)

```
<?xml version="1.0" ?>
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope">
  <env:Header>
    <env:NotUnderstood qname="t:transaction"
      xmlns:t="http://thirdparty.example.org/transaction"/>
  </env:Header>
  <env:Body>
    <env:Fault>
      <env:Code>
        <env:Value>env:MustUnderstand</env:Value>
      </env:Code>
      <env:Reason>
        <env:Text xml:lang="en-US">Header not understood</env:Text>
        <env:Text xml:lang="fr">En-tête non compris</env:Text>
      </env:Reason>
    </env:Fault>
  </env:Body>
</env:Envelope>
```

## WSDL Example

```
<?xml version="1.0" encoding="utf-8" ?>
<description
    xmlns="http://www.w3.org/ns/wsdl"
    targetNamespace= "http://greath.example.com/2004/wsdl/resSvc"
    xmlns:tns= "http://greath.example.com/2004/wsdl/resSvc"
    xmlns:ghns = "http://greath.example.com/2004/schemas/resSvc"
    xmlns:wsoap= "http://www.w3.org/ns/wsdl/soap"
    xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
    xmlns:wsdlx= "http://www.w3.org/ns/wsdl-extensions">

<documentation>
    This document describes the GreatH Web service. Additional
    application-level requirements for use of this service --
    beyond what WSDL 2.0 is able to describe -- are available
    at http://greath.example.com/2004/reservation-documentation.html
</documentation>

<types>
    <xsschema
        xmlns:xs="http://www.w3.org/2001/XMLSchema"
        targetNamespace="http://greath.example.com/2004/schemas/resSvc"
        xmlns="http://greath.example.com/2004/schemas/resSvc">

        <xselement name="checkAvailability" type="tCheckAvailability"/>
        <xsccomplexType name="tCheckAvailability">
            <xsssequence>
                <xselement name="checkInDate" type="xs:date"/>
                <xselement name="checkOutDate" type="xs:date"/>
                <xselement name="roomType" type="xs:string"/>
            </xsssequence>
        </xsccomplexType>

        <xselement name="checkAvailabilityResponse" type="xs:double"/>

        <xselement name="invalidDataError" type="xs:string"/>
    </xsschema>
</types>
```

```

<interface name = "reservationInterface" >

    <fault name = "invalidDataFault"
        element = "ghns:invalidDataError"/>

    <operation name="opCheckAvailability"
        pattern="http://www.w3.org/ns/wsdl/in-out"
        style="http://www.w3.org/ns/wsdl/style/iri"
        wsdlx:safe = "true">
        <input messageLabel="In"
            element="ghns:checkAvailability" />
        <output messageLabel="Out"
            element="ghns:checkAvailabilityResponse" />
        <outfault ref="tns:invalidDataFault" messageLabel="Out"/>
    </operation>

</interface>

<binding name="reservationSOAPBinding"
    interface="tns:reservationInterface"
    type="http://www.w3.org/ns/wsdl/soap"
    wsoap:protocol="http://www.w3.org/2003/05/soap/bindings/HTTP/">

    <fault ref="tns:invalidDataFault"
        wsoap:code="soap:Sender"/>

    <operation ref="tns:opCheckAvailability"
        wsoap:mep="http://www.w3.org/2003/05/soap/mep/soap-response"/>

</binding>

<service name="reservationService"
    interface="tns:reservationInterface">

    <endpoint name="reservationEndpoint"
        binding="tns:reservationSOAPBinding"
        address ="http://greath.example.com/2004/reservation"/>

</service>

</description>

```

## WS-BPEL Example

```
<process name="purchaseOrderProcess"
  targetNamespace="http://example.com/ws-bp/purchase"
  xmlns="http://docs.oasis-open.org/wsbpel/2.0/process/executable"
  xmlns:lns="http://manufacturing.org/wsdl/purchase">

<partnerLinks>
  <partnerLink name="purchasing"
    partnerLinkType="lns:purchasingLT" myRole="purchaseService" />
  <partnerLink name="invoicing" partnerLinkType="lns:invoicingLT"
    myRole="invoiceRequester" partnerRole="invoiceService" />
  <partnerLink name="shipping" partnerLinkType="lns:shippingLT"
    myRole="shippingRequester" partnerRole="shippingService" />
  <partnerLink name="scheduling"
    partnerLinkType="lns:schedulingLT"
    partnerRole="schedulingService" />
</partnerLinks>

<variables>
  <variable name="PO" messageType="lns:POMessage" />
  <variable name="Invoice" messageType="lns:InvMessage" />
  <variable name="shippingRequest"
    messageType="lns:shippingRequestMessage" />
  <variable name="shippingInfo"
    messageType="lns:shippingInfoMessage" />
  <variable name="shippingSchedule"
    messageType="lns:scheduleMessage" />
</variables>

<faultHandlers>
  <catch faultName="lns:cannotCompleteOrder"
    faultVariable="POFault"
    faultMessageType="lns:orderFaultType">
    <reply partnerLink="purchasing"
      portType="lns:purchaseOrderPT"
      operation="sendPurchaseOrder" variable="POFault"
      faultName="cannotCompleteOrder" />
  </catch>
</faultHandlers>

<sequence>
  <receive partnerLink="purchasing" portType="lns:purchaseOrderPT"
    operation="sendPurchaseOrder" variable="PO"
    createInstance="yes">
    <documentation>Receive Purchase Order</documentation>
  </receive>

  <flow>
    <documentation>
```

A parallel flow to handle shipping, invoicing and scheduling

```

</documentation>
<links>
    <link name="ship-to-invoice" />
    <link name="ship-to-scheduling" />
</links>
<sequence>
    <assign>
        <copy>
            <from>$PO.customerInfo</from>
            <to>$shippingRequest.customerInfo</to>
        </copy>
    </assign>
    <invoke partnerLink="shipping" portType="lns:shippingPT"
        operation="requestShipping"
        inputVariable="shippingRequest"
        outputVariable="shippingInfo">
        <documentation>Decide On Shipper</documentation>
        <sources>
            <source linkName="ship-to-invoice" />
        </sources>
    </invoke>
    <receive partnerLink="shipping"
        portType="lns:shippingCallbackPT"
        operation="sendSchedule" variable="shippingSchedule">
        <documentation>Arrange Logistics</documentation>
        <sources>
            <source linkName="ship-to-scheduling" />
        </sources>
    </receive>
</sequence>
<sequence>
    <invoke partnerLink="invoicing"
        portType="lns:computePricePT"
        operation="initiatePriceCalculation"
        inputVariable="PO">
        <documentation>
            Initial Price Calculation
        </documentation>
    </invoke>
    <invoke partnerLink="invoicing"
        portType="lns:computePricePT"
        operation="sendShippingPrice"
        inputVariable="shippingInfo">
        <documentation>
            Complete Price Calculation
        </documentation>
        <targets>
            <target linkName="ship-to-invoice" />
        </targets>
    </invoke>
</sequence>
```

```

</invoke>
<receive partnerLink="invoicing"
    portType="lns:invoiceCallbackPT"
    operation="sendInvoice" variable="Invoice" />
</sequence>
<sequence>
    <invoke partnerLink="scheduling"
        portType="lns:schedulingPT"
        operation="requestProductionScheduling"
        inputVariable="PO">
        <documentation>
            Initiate Production Scheduling
        </documentation>
    </invoke>
    <invoke partnerLink="scheduling"
        portType="lns:schedulingPT"
        operation="sendShippingSchedule"
        inputVariable="shippingSchedule">
        <documentation>
            Complete Production Scheduling
        </documentation>
        <targets>
            <target linkName="ship-to-scheduling" />
        </targets>
    </invoke>
</sequence>
</flow>
<reply partnerLink="purchasing" portType="lns:purchaseOrderPT"
    operation="sendPurchaseOrder" variable="Invoice">
    <documentation>Invoice Processing</documentation>
</reply>
</sequence>

</process>

```