

4 Messaging Framework

4.1 Purpose

The Messaging Framework describes the way in which messages must be prepared from the transmission perspective in order for all connectors to successfully create and process messages. This includes SOAP and custom SOAP headers, Attachments, the WSDL, the XSD's and the Error Codes.

4.2 SOAP 1.1

SOAP is a lightweight protocol for exchange of information in a decentralized, distributed environment. It is an XML based protocol that consists of three parts: an envelope that defines a framework for describing what is in a message and how to process it, a set of encoding rules for expressing instances of application-defined data types, and a convention for representing remote procedure calls and responses.

SOAP messages are fundamentally one-way transmissions from a sender to a receiver but SOAP messages are often combined to implement patterns such as the Request/Response Pattern, where it provides for SOAP response messages to be delivered as HTTP responses, using the same connection as the inbound request. This is the pattern UI SIDES will use to accomplish its file transfer/acknowledgement scheme.

4.2.1 SOAPAction

From the W3C, the SOAPAction component in SOAP 1.1 is defined as follows.

“The SOAPAction HTTP request header field can be used to indicate the intent of the SOAP HTTP request. The value is a URI identifying the intent. SOAP places no restrictions on the format or specificity of the URI or that it is resolvable. An HTTP client **MUST** use this header field when issuing a SOAP HTTP Request.”

For UI SIDES, the SOAPAction must be specified in the request. The particular SOAP Action required for each message is specified in the Web Services Description Language (WSDL) section below.

4.3 Message Transmission Optimization Mechanism (MTOM)

Message Transmission Optimization Mechanism, or MTOM, is a mechanism for transmitting large binary attachments with SOAP messages as raw bytes, allowing for smaller messages. MTOM provides an elegant mechanism of efficiently transmitting binary data, such as images, PDF files, MS Word documents, between systems.

This is defined in the XSD as part of the attachment sequence as:

```
<xs:element name="AttachmentData" type="xs:base64Binary" />
```

An example of an attachment can be seen in the “Putting It All Together” section later in the document.

4.4 Web Services Description Language (WSDL)

The Web Services Description Language (WSDL, pronounced 'wiz-dol') is an XML-based language that provides a model for describing Web services. WSDL defines an XML grammar for describing network services as collections of communication endpoints capable of exchanging messages. WSDL service definitions provide documentation for distributed systems and serve as a recipe for automating the details involved in applications communication.

The WSDL is broken into two files, a StateBroker.wsdl that describes the interfaces exposed to the States from the Broker and EmployerTPABroker.wsdl that describes the interfaces exposed to the Employer/TPAs from the Broker. Along with each WSDL, there are a StateTransmissionQuery.xsd and EmployerTPATransmissionQuery.xsd that help define the types used in the WSDL.

A snippet of the description of each of the functions described in the WSDL is as follows in the next 10 sections. Note the Push to the Employer/TPA is not described in these WSDLs because that function belongs in the WSDL for the connector Web Service. Please see the full WSDL for a complete description of the following sections.

4.4.1 State Post

The State Post operation is defined as

```
<wsdl:operation name="postStateSeparationRequestCollection">
  <soap:operation soapAction="postStateSeparationRequestCollection" />
  <wsdl:input name="StateSeparationRequestCollection">
    <soap:body use="literal" />
  </wsdl:input>
  <wsdl:output name="StateSeparationRequestCollectionAcknowledgement">
    <soap:body use="literal" />
  </wsdl:output>
</wsdl:operation>
```

The input (http request) to this operation is a StateSeparationRequestCollection as defined in SeparationRequest.xsd. The output (http response) is a StateSeparationRequestCollectionAcknowledgement as defined in the SeparationRequest.xsd.

4.4.2 State Pull

The State Pull operation is defined as

```
<wsdl:operation name="pullStateSeparationResponseCollection">
  <soap:operation soapAction="pullStateSeparationResponseCollection" />
  <wsdl:input name="StateSeparationResponseCollectionQuery">
```

```

        <soap:body use="literal" />
    </wsdl:input>
    <wsdl:output name="StateSeparationResponseCollection">
        <soap:body use="literal" />
    </wsdl:output>
</wsdl:operation>

```

The input (http request) to this operation is a StateSeparationResponseCollectionQuery as defined in StateTransmissionQuery.xsd. The output (http response) is a StateSeparationResponseCollection as defined in the SeparationResponse.xsd.

The StateSeparationResponseCollectionQuery is a complex type that allows the caller to specify one of three operations, a Pull, a Re-Pull by StateSOAPTransactionNumber and by a Date Range.

```

<!-- Query element for states to collect claim responses they are expecting -->
<xs:element name="StateSeparationResponseCollectionQuery"
    type="StateSeparationResponseCollectionQueryType"/>

<!-- Types for query element for states to collect claim responses they are expecting -->
<xs:complexType name="StateSeparationResponseCollectionQueryType">
    <xs:sequence>
        <xs:element name="StatePostalCode" type="StateAbrCodes" />
        <xs:element name="StateSeparationResponseCollectionQueryCriteria"
            type="StateSeparationResponseCollectionQueryCriteriaType"
            minOccurs="0" />
    </xs:sequence>
</xs:complexType>

<xs:complexType name="StateSeparationResponseCollectionQueryCriteriaType">
    <xs:sequence>
        <xs:element name="StateSOAPTransactionNumber" type="GUID" minOccurs="0"/>
        <xs:group ref="StateSeparationResponseCollectionQueryCriteriaGroup"
minOccurs="0"/>
    </xs:sequence>
</xs:complexType>

<xs:group name="StateSeparationResponseCollectionQueryCriteriaGroup">
    <xs:sequence>
        <xs:element name="BrokerRecordEffectiveDateFrom" type="CustomDateTime" />
        <xs:element name="BrokerRecordEffectiveDateTo" type="CustomDateTime" />
    </xs:sequence>
</xs:group>

```

For the straight Pull, the caller needs to supply only the State Abbreviation. Although there are different ways to verify the calling State besides this element, there is a requirement in WSDL 1.1 that a WSDL definition have at least one input attribute.

4.4.3 State Re-Pull by StateSoapTransactionNumber

The State Re-Pull by StateSoapTransactionNumber operation is defined as

```
<wsdl:operation name="pullStateSeparationResponseCollection">
  <soap:operation soapAction="pullStateSeparationResponseCollection" />
  <wsdl:input name="StateSeparationResponseCollectionQuery">
    <soap:body use="literal" />
  </wsdl:input>
  <wsdl:output name="StateSeparationResponseCollection">
    <soap:body use="literal" />
  </wsdl:output>
</wsdl:operation>
```

The input (http request) to this operation is a StateSeparationResponseCollectionQuery as defined in StateTransmissionQuery.xsd. The output (http response) is a StateSeparationResponseCollection as defined in the SeparationResponse.xsd.

The StateSeparationResponseCollectionQuery is a complex type that allows the caller to specify one of three operations, a Pull, a Re-Pull by StateSOAPTransactionNumber and by a Date Range.

```
<!-- Query element for states to collect claim responses they are expecting -->
<xs:element name="StateSeparationResponseCollectionQuery"
  type="StateSeparationResponseCollectionQueryType"/>

<!-- Types for query element for states to collect claim responses they are expecting -->
<xs:complexType name="StateSeparationResponseCollectionQueryType">
  <xs:sequence>
    <xs:element name="StatePostalCode" type="StateAbrCodes" />
    <xs:element name="StateSeparationResponseCollectionQueryCriteria"
      type="StateSeparationResponseCollectionQueryCriteriaType"
      minOccurs="0" />
  </xs:sequence>
</xs:complexType>

<xs:complexType name="StateSeparationResponseCollectionQueryCriteriaType">
  <xs:sequence>
    <xs:element name="StateSOAPTransactionNumber" type="GUID" minOccurs="0"/>
    <xs:group ref="StateSeparationResponseCollectionQueryCriteriaGroup"
      minOccurs="0"/>
  </xs:sequence>
</xs:complexType>

<xs:group name="StateSeparationResponseCollectionQueryCriteriaGroup">
  <xs:sequence>
    <xs:element name="BrokerRecordEffectiveDateFrom" type="CustomDateTime" />
```

```

    <xs:element name="BrokerRecordEffectiveDateTo" type="CustomDateTime" />
  </xs:sequence>
</xs:group>

```

For the Re-Pull by StateSOAPTransactionNumber, the caller needs to supply the State Abbreviation and the StateSOAPTransactionNumber element out of the StateSeparationResponseCollectionQueryCriteriaType. This will allow the Broker to send the file defined by the StateSOAPTransactionNumber.

4.4.4 State Re-Pull by Date

The State Re-Pull by Date operation is defined as

```

<wsdl:operation name="pullStateSeparationResponseCollection">
  <soap:operation soapAction="pullStateSeparationResponseCollection" />
  <wsdl:input name="StateSeparationResponseCollectionQuery">
    <soap:body use="literal" />
  </wsdl:input>
  <wsdl:output name="StateSeparationResponseCollection">
    <soap:body use="literal" />
  </wsdl:output>
</wsdl:operation>

```

The input (http request) to this operation is a StateSeparationResponseCollectionQuery as defined in StateTransmissionQuery.xsd. The output (http response) is a StateSeparationResponseCollection as defined in the SeparationResponse.xsd.

The StateSeparationResponseCollectionQuery is a complex type that allows the caller to specify one of three operations, a Pull, a Re-Pull by StateSOAPTransactionNumber and by a Date Range.

```

<!-- Query element for states to collect claim responses they are expecting -->
<xs:element name="StateSeparationResponseCollectionQuery"
  type="StateSeparationResponseCollectionQueryType"/>

<!-- Types for query element for states to collect claim responses they are expecting -->
<xs:complexType name="StateSeparationResponseCollectionQueryType">
  <xs:sequence>
    <xs:element name="StatePostalCode" type="StateAbrCodes" />
    <xs:element name="StateSeparationResponseCollectionQueryCriteria"
      type="StateSeparationResponseCollectionQueryCriteriaType"
      minOccurs="0" />
  </xs:sequence>
</xs:complexType>

<xs:complexType name="StateSeparationResponseCollectionQueryCriteriaType">
  <xs:sequence>
    <xs:element name="StateSOAPTransactionNumber" type="GUID" minOccurs="0"/>

```

```

<xs:group ref="StateSeparationResponseCollectionQueryCriteriaGroup"
minOccurs="0"/>
</xs:sequence>
</xs:complexType>

<xs:group name="StateSeparationResponseCollectionQueryCriteriaGroup">
<xs:sequence>
<xs:element name="BrokerRecordEffectiveDateFrom" type="CustomDateTime" />
<xs:element name="BrokerRecordEffectiveDateTo" type="CustomDateTime" />
</xs:sequence>
</xs:group>

```

For the Re-Pull by date range, the caller needs to supply the State Abbreviation and the StateSeparationResponseCollectionQueryCriteriaGroup element out of the StateSeparationResponseCollectionQueryCriteriaType. The StateSeparationResponseCollectionQueryCriteriaGroup is a complex type that is defined as a begin date (BrokerRecordEffectiveDateFrom), an end date (BrokerRecordEffectiveDateTo) and a StateSOAPTransactionNumber. The first time this operation is called, the StateSOAPTransactionNumber is null and the dates that the Connector wants to Re-Pull from are included. When the Broker sends back the first file, the Broker will include the next StateSOAPTransactionNumber during that date range in a SOAP header attribute. In the next call to this operation, the caller includes this StateSOAPTransactionNumber with the date range. This differentiates to the Broker the next call in the series from a brand new Re-Pull by Date request. The last file sent back to the Connector is indicated by a null value for the next StateSOAPTransactionNumber.

4.4.5 State Pull Acknowledgement

Since the Pull operation uses up a full http request/response pattern, the Broker WSDL needs to supply the State with an acknowledgement operation that must be called when a Pull is completed. It is defined as:

```

<wsdl:operation name="pullStateSeparationResponseCollectionAcknowledgement">
  <soap:operation
    soapAction="pullStateSeparationResponseCollectionAcknowledgement" />
  <wsdl:input name="StateSeparationResponseCollectionAcknowledgement">
    <soap:body use="literal" />
  </wsdl:input>
</wsdl:operation>

```

The input (http request) to this operation is a StateSeparationResponseCollectionAcknowledgement as defined in SeparationResponse.xsd. There is no output for this operation.

4.4.6 EmployerTPA Post

The EmployerTPA Post operation is defined as

```
<wsdl:operation name="postEmployerTPASeparationResponseCollection">
  <soap:operation
    soapAction="postEmployerTPASeparationResponseCollection" />
  <wsdl:input name="EmployerTPASeparationResponseCollection">
    <soap:body use="literal" />
  </wsdl:input>
  <wsdl:output
    name="EmployerTPASeparationResponseCollectionAcknowledgement">
    <soap:body use="literal" />
  </wsdl:output>
</wsdl:operation>
```

The input (http request) to this operation is an EmployerTPASeparationResponseCollection as defined in SeparationResponse.xsd. The output (http response) is an EmployerTPASeparationResponseCollectionAcknowledgement as defined in the SeparationResponse.xsd.

4.4.7 EmployerTPA Pull

The EmployerTPA Pull operation is defined as

```
<wsdl:operation name="pullEmployerTPASeparationRequestCollection">
  <soap:operation soapAction="pullEmployerTPASeparationRequestCollection" />
  <wsdl:input name="EmployerTPASeparationRequestCollectionQuery">
    <soap:body use="literal" />
  </wsdl:input>
  <wsdl:output name="EmployerTPASeparationRequestCollection">
    <soap:body use="literal" />
  </wsdl:output>
</wsdl:operation>
```

The input (http request) to this operation is an EmployerTPASeparationRequestCollectionQuery as defined in EmployerTPATransmissionQuery.xsd. The output (http response) is an EmployerTPASeparationRequestCollection as defined in the SeparationRequest.xsd.

The EmployerTPASeparationRequestCollectionQuery is a complex type that allows the caller to specify one of three operations, a Pull, a Re-Pull by EmployerTPASOAPTransactionNumber and by a Date Range.

```
<!-- Query element for employer to collect claim responses they are expecting -->
<xs:element name="EmployerTPASeparationRequestCollectionQuery"
  type="EmployerTPASeparationRequestCollectionQueryType"/>

<!-- Types for query element for Employers/TPAs to collect claim requests they are expecting
-->
```

```

<xs:complexType name="EmployerTPASeparationRequestCollectionQueryType">
  <xs:sequence>
    <xs:element name="UniqueID" type="UniqueIDType" />
    <xs:element name="EmployerTPASeparationRequestCollectionQueryCriteria"
      type="EmployerTPASeparationRequestCollectionQueryCriteriaType"
      minOccurs="0" />
  </xs:sequence>
</xs:complexType>

<xs:complexType name="EmployerTPASeparationRequestCollectionQueryCriteriaType">
  <xs:sequence>
    <xs:element name="EmployerTPASOAPTransactionNumber" type="GUID"
      minOccurs="0" />
    <xs:group ref="EmployerTPASeparationRequestCollectionQueryCriteriaGroup"
      minOccurs="0"/>
  </xs:sequence>
</xs:complexType>

<xs:group name="EmployerTPASeparationRequestCollectionQueryCriteriaGroup">
  <xs:sequence>
    <xs:element name="BrokerRecordEffectiveDateFrom" type="CustomDateTime" />
    <xs:element name="BrokerRecordEffectiveDateTo" type="CustomDateTime" />
  </xs:sequence>
</xs:group>

```

For the straight Pull, the caller needs to supply only the EmployerTPA UniqueID. Although there are different ways to verify the calling EmployerTPA besides this element, there is a requirement in WSDL 1.1 that a WSDL definition have at least one input attribute.

4.4.8 EmployerTPA Re-Pull by EmployerTPASoapTransactionNumber

The EmployerTPA Re-Pull by EmployerTPASoapTransactionNumber operation is defined as

```

<wsdl:operation name="pullEmployerTPASeparationRequestCollection">
  <soap:operation soapAction="pullEmployerTPASeparationRequestCollection" />
  <wsdl:input name="EmployerTPASeparationRequestCollectionQuery">
    <soap:body use="literal" />
  </wsdl:input>
  <wsdl:output name="EmployerTPASeparationRequestCollection">
    <soap:body use="literal" />
  </wsdl:output>
</wsdl:operation>

```

The input (http request) to this operation is an EmployerTPASeparationRequestCollectionQuery as defined in EmployerTPATransmissionQuery.xsd. The output (http response) is an EmployerTPASeparationRequestCollection as defined in the SeparationRequest.xsd.

The EmployerTPASeparationRequestCollectionQuery is a complex type that allows the caller to specify one of three operations, a Pull, a Re-Pull by EmployerTPASOAPTransactionNumber and by a Date Range.

```

<!-- Query element for employer to collect claim responses they are expecting -->
<xs:element name="EmployerTPASeparationRequestCollectionQuery"
  type="EmployerTPASeparationRequestCollectionQueryType"/>

<!-- Types for query element for Employers/TPAs to collect claim requests they are expecting
-->
<xs:complexType name="EmployerTPASeparationRequestCollectionQueryType">
  <xs:sequence>
    <xs:element name="UniqueID" type="UniqueIDType" />
    <xs:element name="EmployerTPASeparationRequestCollectionQueryCriteria"
      type="EmployerTPASeparationRequestCollectionQueryCriteriaType"
      minOccurs="0" />
  </xs:sequence>
</xs:complexType>

<xs:complexType name="EmployerTPASeparationRequestCollectionQueryCriteriaType">
  <xs:sequence>
    <xs:element name="EmployerTPASOAPTransactionNumber" type="GUID"
      minOccurs="0" />
    <xs:group ref="EmployerTPASeparationRequestCollectionQueryCriteriaGroup"
      minOccurs="0"/>
  </xs:sequence>
</xs:complexType>

<xs:group name="EmployerTPASeparationRequestCollectionQueryCriteriaGroup">
  <xs:sequence>
    <xs:element name="BrokerRecordEffectiveDateFrom" type="CustomDateTime" />
    <xs:element name="BrokerRecordEffectiveDateTo" type="CustomDateTime" />
  </xs:sequence>
</xs:group>

```

For the Re-Pull by EmployerTPASOAPTransactionNumber, the caller needs to supply the EmployerTPA UniqueID and the EmployerTPASOAPTransactionNumber element out of the EmployerTPASeparationResponseCollectionQueryCriteriaType. This will allow the Broker to send the file defined with the EmployerTPASOAPTransactionNumber.

4.4.9 EmployerTPA Re-Pull by Date

The EmployerTPA Re-Pull by EmployerTPASoapTransactionNumber operation is defined as

```

<wsdl:operation name="pullEmployerTPASeparationRequestCollection">
  <soap:operation soapAction="pullEmployerTPASeparationRequestCollection" />
  <wsdl:input name="EmployerTPASeparationRequestCollectionQuery">

```

```

        <soap:body use="literal" />
    </wsdl:input>
    <wsdl:output name="EmployerTPASeparationRequestCollection">
        <soap:body use="literal" />
    </wsdl:output>
</wsdl:operation>

```

The input (http request) to this operation is an EmployerTPASeparationRequestCollectionQuery as defined in EmployerTPATransmissionQuery.xsd. The output (http response) is an EmployerTPASeparationRequestCollection as defined in the SeparationRequest.xsd.

The EmployerTPASeparationRequestCollectionQuery is a complex type that allows the caller to specify one of three operations, a Pull, a Re-Pull by EmployerTPASOAPTransactionNumber and by a Date Range.

```

<!-- Query element for employer to collect claim responses they are expecting -->
<xs:element name="EmployerTPASeparationRequestCollectionQuery"
    type="EmployerTPASeparationRequestCollectionQueryType"/>

<!-- Types for query element for Employers/TPAs to collect claim requests they are expecting
-->
<xs:complexType name="EmployerTPASeparationRequestCollectionQueryType">
    <xs:sequence>
        <xs:element name="UniqueID" type="UniqueIDType" />
        <xs:element name="EmployerTPASeparationRequestCollectionQueryCriteria"
            type="EmployerTPASeparationRequestCollectionQueryCriteriaType"
            minOccurs="0" />
    </xs:sequence>
</xs:complexType>

<xs:complexType name="EmployerTPASeparationRequestCollectionQueryCriteriaType">
    <xs:sequence>
        <xs:element name="EmployerTPASOAPTransactionNumber" type="GUID"
minOccurs="0" />
        <xs:group ref="EmployerTPASeparationRequestCollectionQueryCriteriaGroup"
minOccurs="0"/>
    </xs:sequence>
</xs:complexType>

<xs:group name="EmployerTPASeparationRequestCollectionQueryCriteriaGroup">
    <xs:sequence>
        <xs:element name="BrokerRecordEffectiveDateFrom" type="CustomDateTime" />
        <xs:element name="BrokerRecordEffectiveDateTo" type="CustomDateTime" />
    </xs:sequence>
</xs:group>

```

For the Re-Pull by date range, the caller needs to supply the UniqueID and the EmployerTPASeparationRequestCollectionQueryCriteriaGroup element out of the

EmployerTPASeparationRequestCollectionQueryCriteriaType. The EmployerTPASeparationRequestCollectionQueryCriteriaGroup is a complex type that is defined as a begin date (BrokerRecordEffectiveDateFrom), an end date (BrokerRecordEffectiveDateTo) and an EmployerTPASOAPTransactionNumber. The first time this operation is called, the EmployerTPASOAPTransactionNumber is null and the dates that the Connector wants to Re-Pull from are included. When the Broker sends back the first file, the Broker will include the next EmployerTPASOAPTransactionNumber during that date range in a SOAP header attribute. In the next call to this operation, the caller includes this EmployerTPASOAPTransactionNumber with the date range. This differentiates to the Broker the next call in the series from a brand new Re-Pull by Date request. The last file sent back to the Connector is indicated by a null value for the next EmployerTPASOAPTransactionNumber.

4.4.10 EmployerTPA Pull Acknowledgement

Since the Pull operation uses up a full http request/response pattern, the Broker WSDL needs to supply the EmployerTPA with an acknowledgement operation that must be called when a Pull is completed. It is defined as:

```
<wsdl:operation
  name="pullEmployerTPASeparationRequestCollectionAcknowledgement">
  <soap:operation
    soapAction =
      "pullEmployerTPASeparationRequestCollectionAcknowledgement" />
  <wsdl:input
    name="EmployerTPASeparationRequestCollectionAcknowledgement">
    <soap:body use="literal" />
  </wsdl:input>
</wsdl:operation>
```

The input (http request) to this operation is an EmployerTPASeparationRequestCollectionAcknowledgement as defined in SeparationRequest.xsd. There is no output for this operation.

4.5 Document Error Codes

The following sections define the Error Codes that are passed back to the State and Employer/TPA in the ErrorOccurance element for those errors that occur within the record structure of the Request or Response.

4.5.1 Claim Request Acknowledgement Error Codes

These are the error codes that will be passed back to the States in the ErrorOccurance element.

```
<ErrorOccurance>
  <ErrorCode>101</ErrorCode>
  <ErrorMessage> XSD validation violation</ErrorMessage>
```

</ErrorOccurrence>

Error Code	Error Message
101.	XSD validation violation
102.	Business Rule violation - Two or more UniqueAttachmentIDs assigned to a specific Separation Information Request are the same - they must be unique.
110.	Business Rule violation - There is no Employer URI lookup
111.	Business Rule violation - There must be a value (Date) for WagesNeededBeginDate if WagesWeeksNeededCode = WO/WW
112.	Business Rule violation - There must be a value (Date) for WagesNeededEndDate if WagesWeeksBeginDate is completed.

4.5.2 Claim Response Acknowledgement Error Codes

These are the error codes that will be passed back to the Employer/TPAs in the ErrorOccurance element.

<ErrorOccurrence>
 <ErrorCode>201</ErrorCode>
 <ErrorMessage>XSD validation violation</ErrorMessage>
 </ErrorOccurrence>

Error Code	Error Message
201.	XSD validation violation
202.	Business Rule violation - Two or more UniqueAttachmentIDs assigned to a specific Separation Information Response are the same - they must be unique.
210.	Business Rule violation - There is no matching Claim Request record with fields matching A1 to B1, A2 to B2, A3 to B3, A4 to B4, the StateRequestRecordGUID, and the BrokerRecordTransactionNumber.
211.	Business Rule violation - There must be a value (Character – Size 1) for TotalEarnedWagesNeededInd if WagesWeeksNeededCode = WO/WW and EmployerSeparationReasonCode does not equal 20 or 21

Version 9 – Design Baseline – 12-08-2009

Copyright © 2008 - 2009 National Association of State Workforce Agencies. All Rights Reserved.

212.	Business Rule violation - There must be a value (Numeric – Size 15.2) for TotalEarnedWages if TotalEarnedWagesNeededInd = 1 for <i>Wages are Available</i>
213.	Business Rule violation - There must be a value (Character – Size 1) for TotalWeeksWorkedNeededInd if WagesWeeksNeededCode = WW and EmployerSeparationReasonCode does not equal 20 or 21
214.	Business Rule violation - There must be a value (Numeric – Size 2) for TotalWeeksWorked if TotalWeeksWorkedNeededInd = 1 for <i>Weeks are Available</i>
215.	Business Rule violation - There must be a value (Numeric – Size 2) for NumberOfHoursWorkedAfterClaimEffectiveDate if WagesEarnedAfterClaimEffectiveDate > 0
216.	Business Rule violation - There must be a value (Numeric – Size 15.2) for AverageWeeklyWage if remuneration included
217.	Business Rule violation - There must be a value (Character – Size 1 – Values Y N) for ReturnToWorkInd if EmployerSepReasonCode is 4 for <i>Vacation/Holiday Shutdown</i> or is 7 for <i>School Employee Between Semesters or Terms, Likely to Return</i> or is 17 for <i>Professional Athlete Between Sports Seasons</i> or is 1 for <i>Temporary Layoff</i> or is 15 for <i>Disciplinary Suspension</i>
218.	Business Rule violation - There must be a value (Date) for ReturnToWorkDate if ReturnToWorkInd = ‘Y’.
219.	Business Rule violation - There must be a value (Character – Size 2 – Values Y N 99) for WorkingAllAvailableHoursInd if EmployerSepReasonCode is 11 for <i>Still Employed, Hours Reduced by Employer</i>
220.	Business Rule violation - There must be a value (Character –Size 500) for NotWorkingAvailableHoursReason if WorkingAllAvailableHoursInd = N
221.	Business Rule violation - There must be a value (Character – Size 1 – Values S L) for LaborDisputeTypeInd if EmployerSepReasonCode is 16 for <i>Labor Dispute</i>
222.	Business Rule violation - There must be a value (Date) for AllocationBeginDate if EmployerAllocationCode is Y
223.	Business Rule violation - There must be a value (Date) for

Version 9 – Design Baseline – 12-08-2009

Copyright © 2008 - 2009 National Association of State Workforce Agencies. All Rights Reserved.

	AllocationEndDate if EmployerAllocationCode is <i>Y</i>
224.	Business Rule violation - There must be a value (Numeric – Size 2) for AverageNumberHoursWorkedPerWeek if remuneration included
225.	Business Rule violation - There must be a value (Character – Size 1 – Values Y N) for MandatoryRetirementInd if EmployerSepReasonCode = 14 for <i>Retirement</i>
226.	Business Rule violation - There must be a value (Character – Size 1 – Values Y N) for MandatoryPension if any entry for RemunerationTypeCode = 5 for <i>Pension</i>
227.	Business Rule violation - There must be a value (Character – Size 1 – Values Y N) for ContributoryOrNotContributoryClaimantInd if any entry for RemunerationTypeCode = 5 for <i>Pension</i>
228.	Business Rule violation - There must be a value (Numeric – Size 3) for ClaimantPensionContributionPercent if ContributoryOrNotContributoryClaimantInd is <i>Y</i>
229.	Business Rule violation - There must be a value (Character – Size 2000) for EmployerSepReasonComments if EmployerSepReasonCode is other than 3 for <i>Discharged</i> or 6 for <i>Voluntary Quit</i>
230.	Business Rule violation - There must be a value (Character – Size 2 – Value 1-8 or 99) for DischargeReasonCode if EmployerSepReasonCode is 3 for <i>Discharged</i> or 5 for <i>Asked to Resign</i>
231.	Business Rule violation - There must be a value (Character – Size 1000) for FinalIncidentReason if DischargedReasonCode is 2-8
232.	Business Rule violation - There must be a value (Date) for FinalIncidentDate if FinalIncidentReason is completed
233.	Business Rule violation - There must be a value (Character – Size 1 – Values Y N) for ViolateCompanyPolicyInd if DischargeReasonCode is 2-8
234.	Business Rule violation - There must be a value (Character – Size 1 – Values Y N) for DischargePolicyAwareInd if ViolateCompanyPolicyInd is <i>Y</i>
235.	Business Rule violation - There must be a value (Character – Size 1 – Values W V) for DischargePolicyAwareExplanationCode if DischargePolicyAwareInd is <i>Y</i>

Version 9 – Design Baseline – 12-08-2009

Copyright © 2008 - 2009 National Association of State Workforce Agencies. All Rights Reserved.

236.	Business Rule violation - There must be a value (Date) for PriorIncidentWarningDate if PriorIncidentWarningInd is <i>Y</i>
237.	Business Rule violation - There must be a value (Character – Size 1000) for PriorIncidentWarningDescription if PriorIncidentWarningInd is <i>Y</i>
238.	Business Rule violation - There must be a value (Character – Size 60) for WhoDischargedName if DischargeReasonCode is included
239.	Business Rule violation - There must be a value (Character – Size 60) for WhoDischargedTitle if DischargeReasonCode is included
240.	Business Rule violation - There must be a value (Character – Size 1000) for DischargeReasonComments if DischargeReasonCode is 8 for <i>Other</i>
241.	Business Rule violation - There must be a value (Character – Size 2 – Values 1-10 or 99) for VoluntarySepReasonCode if EmployerSepReasonCode is 6 for <i>Voluntary Quit</i>
242.	Business Rule violation - There must be a value (Character – Size 1 – Values 1-6)for HiringAgreementChangeCode if VoluntarySepReasonCode is 9 for <i>Working Conditions</i>
243.	Business Rule violation - There must be a value (Character – Size 1000) for HiringAgreementChangeComments if HiringAgreementChangeCode is 2-6
244.	Business Rule violation - There must be a value (Character – Size 1 – Values Y N) for ClaimantActionsAvoidQuitInd if VoluntarySepReasonCode is included
245.	Business Rule violation - There must be a value (Character – Size 1000) for ActionTakenComments if VoluntarySepReasonCode is included and if ClaimantActionsToAvoidQuitInd = <i>Y</i>
246.	Business Rule violation - There must be a value (Character – Size 1 – Values Y N) for ContinuingWorkAvailableInd if VoluntarySepReasonCode is included
247.	Business Rule violation - There must be a value (Character – Size 2000) for VoluntarySepReasonComments if VoluntarySepReasonCode is 10 for <i>Other</i>
248.	Business Rule violation - There must be a value (Character – Size 60) for PreparerCompanyName if PreparerTypeCode = T for <i>Third Party</i>

Version 9 – Design Baseline – 12-08-2009

Copyright © 2008 - 2009 National Association of State Workforce Agencies. All Rights Reserved.

	<i>Administrator</i>
249.	Business Rule violation - If PreparerTypeCode is "E" for employer, then valid EmployerSepReasonCode (B-20) codes are 1-20 or 99. If PreparerTypeCode is "T" for Third Party Administrator then valid EmployerSepReasonCode (B-20) codes are 1-21 or 99.
250.	Business Rule violation - If the value for EmployerSepReasonCode is 21 <i>for TPA does not represent this employer</i> then the value for PreparerTypeCode must be T for <i>Third Party Administrator</i>
251.	Business Rule violation – EmployerReportedClaimantFirstDayofWork (Date) must be in the past.
252.	Business Rule violation – EmployerReportedClaimantFirstDayofWork (Date) must be before or equal to EmployerReportedClaimantLastDayofWork.
253.	Business Rule violation – EmployerReportedClaimantLastDayofWork (Date) must be in the past.
254.	Business Rule violation – EmployerReportedClaimantLastDayofWork (Date) must be after or equal to EmployerReportedClaimantFirstDayofWork.
255.	Business Rule violation – EffectiveSeparationDate (Date) must be in the past.
256.	Business Rule violation – ReturnToWorkDate (Date) must be after ClaimEffectiveDate
257.	Business Rule violation – FinalIncidentDate (Date) must be in the past
258.	Business Rule violation – PriorIncidentDate (Date) #<number> must be in the past
259.	Business Rule violation – PriorIncidentWarningDate (Date) #<number> must be in the past.
260.	Business Rule violation – AmendedResponseDescription (Character – Size 2000) cannot be empty when AmendedResponse is not null.
261.	Business Rule violation - EffectiveSeparationDate (Date) must be after or equal to EmployerReportedClaimantLastDayofWork.
262.	Responding Employer/TPA does not match the Employer/TPA that the request was sent to
263.	State that the Employer/TPA is responding to does not match the State

	that the request was sent from
264.	Business Rule violation – OtherSSN (Character – Size 9) cannot equal the SSN used for the claim.

4.6 State WSDL

Please see <https://uidataexchange.org/ws/stateBroker.wsdl> for the latest copy of the State WSDL. If the site is unavailable for any reason, please see the ITSC SIDES Website (<http://www.itsc.org/sides.asp>). In case of a discrepancy between the two, the uidataexchange.org is considered to be the latest copy and should be treated as such.

4.7 EmployerTPA WSDL

Please see <https://uidataexchange.org/ws/employerTPABroker.wsdl> for the latest copy of the Employer/TPA WSDL. If the site is unavailable for any reason, please see the ITSC SIDES Website (<http://www.itsc.org/sides.asp>). In case of a discrepancy between the two, the uidataexchange.org is considered to be the latest copy and should be treated as such.

4.8 XSD Files

The following sections contain the XML Schema Definitions (XSDs) for the XML used in the SIDES system. Encoding will use US ASCII. SIDES will support only the printable US ASCII character set (Codes 32 – 126). Records containing ASCII codes outside of this range will fail validation and will be rejected by the Broker with an XSD validation error code.

Null values have been described in the supporting documentation called Appendix_E_v<#>.xls. An example of this is ‘*Value is either a digit or null.*’ There are two different ways in which that can be interpreted in the construction of the XML. For the purposes of SIDES, the interpretation is that if the value of a given field is null, then this field has been completely left out of the XML.

The SeparationRequest.xsd, SeparationResponse.xsd and RequestResponseTypeElements.xsd are the main functional XSDs for the system. The StateTransmissionQuery.xsd, EmployerTPATransmissionQuery.xsd and the TransmissionQueryCommonElements.xsd are design level support XSD’s that bridge the gap between the SeparationRequest.xsd, SeparationResponse.xsd and RequestResponseTypeElements.xsd and the WSDL operations.

4.8.1 StateTransmissionQuery.xsd

Please see <https://uidataexchange.org/schemas/StateTransmissionQuery.xsd> for the latest copy of the StateTransmissionQuery.xsd. If the site is unavailable for any reason, please see the ITSC SIDES Website (<http://www.itsc.org/sides.asp>). In case of a discrepancy between the two, the uidataexchange.org is considered to be the latest copy and should be treated as such.

4.8.2 EmployerTPATransmissionQuery.xsd

Please see <https://uidataexchange.org/schemas/EmployerTPATransmissionQuery.xsd> for the latest copy of the EmployerTPATransmissionQuery.xsd. If the site is unavailable for any reason, please see the ITSC SIDES Website (<http://www.itsc.org/sides.asp>). In case of a discrepancy between the two, the uidataexchange.org is considered to be the latest copy and should be treated as such.

4.8.3 TransmissionQueryCommonElements.xsd

Please see <https://uidataexchange.org/schemas/StateTransmissionQuery.xsd> for the latest copy of the TransmissionQueryCommonElements.xsd. If the site is unavailable for any reason, please see the ITSC SIDES Website (<http://www.itsc.org/sides.asp>). In case of a discrepancy between the two, the uidataexchange.org is considered to be the latest copy and should be treated as such.

4.8.4 SeparationRequest.xsd

Please see <https://uidataexchange.org/schemas/SeparationRequest.xsd> for the latest copy of the SeparationRequest.xsd. If the site is unavailable for any reason, please see the ITSC SIDES Website (<http://www.itsc.org/sides.asp>). In case of a discrepancy between the two, the uidataexchange.org is considered to be the latest copy and should be treated as such.

4.8.5 SeparationResponse.xsd

Please see <https://uidataexchange.org/schemas/SeparationResponse.xsd> for the latest copy of the SeparationResponse.xsd. If the site is unavailable for any reason, please see the ITSC SIDES Website (<http://www.itsc.org/sides.asp>). In case of a discrepancy between the two, the uidataexchange.org is considered to be the latest copy and should be treated as such.

4.8.6 RequestResponseTypeElements.xsd

Please see <https://uidataexchange.org/schemas/RequestResponseTypeElements.xsd> for the latest copy of the RequestResponseTypeElements.xsd. If the site is unavailable for any reason, please see the ITSC SIDES Website (<http://www.itsc.org/sides.asp>). In case of a discrepancy between the two, the uidataexchange.org is considered to be the latest copy and should be treated as such.

4.9 SOAP Headers

Now that we have the all the building blocks of the operations, we still have to define the custom SOAP header in addition to other aspects of the SOAP header for each SOAP message that needs to be passed.

SOAP Header Element	Definition
To	The Unique ID of the Connector to which the message is intended.

From	The Unique ID of the Connector of which the message originates.
StateRequestFileGUID	The State generated GUID applied to this message that can uniquely identify this file.
EmployerTPAResponseFileGUID	The Employer/TPA generated GUID applied to this message that can uniquely identify this file.
MessageCode	The acknowledgement code applied to the message that indicates success or failure of the entire transmission. See <i>Requirements and Rules.doc</i> for more information.
PullCollection	Signifies one of three PULLSs desired by the Connector. 1 indicates a regular PULL 2 indicates a re-PULL by SOAP Transaction Number 3 indicates a re-PULL by date
StateSOAPTransactionNumber	The number given to the message on a PULL by the Broker that allows a re-PULL by SOAP Transaction Number
NextStateSOAPTransactionNumber	The next SOAP Transaction Number for a re-PULL by date.
EmployerTPASOAPTransactionNumber	The number given to the message on a PULL by the Broker that allows a re-PULL by SOAP Transaction Number
NextEmployerTPASOAPTransactionNumber	The next SOAP Transaction Number for a re-PULL by date.

4.9.1 State Post (http request)

```
<To xmlns="https://uidataexchange.org/schemas">0000000001</To>
<From xmlns="https://uidataexchange.org/schemas">CO</From>
<StateRequestFileGUID
xmlns="https://uidataexchange.org/schemas">00000000000000000000000000000001</StateR
equestFileGUID>
```

4.9.2 State Post Acknowledgement (http response)

```
<To xmlns="https://uidataexchange.org/schemas">CO</To>
<From xmlns="https://uidataexchange.org/schemas">Broker</From>
<StateRequestFileGUID
xmlns="https://uidataexchange.org/schemas">00000000000000000000000000000001</StateR
equestFileGUID>
<MessageCode xmlns="https://uidataexchange.org/schemas">2</MessageCode>
```

4.9.3 State Pull (http request)

```
<To xmlns="https://uidataexchange.org/schemas">Broker</To>  
<From xmlns="https://uidataexchange.org/schemas">CO</From>  
<PullCollection xmlns="https://uidataexchange.org/schemas">1</PullCollection>
```

4.9.4 State Re-Pull by StateSOAPTransactionNumber (http request)

```
<To xmlns="https://uidataexchange.org/schemas">Broker</To>  
<From xmlns="https://uidataexchange.org/schemas">CO</From>  
<PullCollection xmlns="https://uidataexchange.org/schemas">2</PullCollection>  
<StateSOAPTransactionNumber  
xmlns="https://uidataexchange.org/schemas">00000000000000000000000000000002</StateS  
OAPTransactionNumber>
```

4.9.5 State Re-Pull by Date (http request)

```
<To xmlns="https://uidataexchange.org/schemas">Broker</To>  
<From xmlns="https://uidataexchange.org/schemas">CO</From>  
<PullCollection xmlns="https://uidataexchange.org/schemas">3</PullCollection>  
<StateSOAPTransactionNumber  
xmlns="https://uidataexchange.org/schemas"></StateSOAPTransactionNumber>
```

4.9.6 State Pull (http response)

```
<To xmlns="https://uidataexchange.org/schemas">CO</To>  
<From xmlns="https://uidataexchange.org/schemas">BR000000001</From>  
<StateSOAPTransactionNumber  
xmlns="https://uidataexchange.org/schemas">00000000000000000000000000000002</StateS  
OAPTransactionNumber>  
<PullCollection xmlns="https://uidataexchange.org/schemas">3</PullCollection>  
<MessageCode xmlns="https://uidataexchange.org/schemas">2</MessageCode>  
<NextStateSOAPTransactionNumber  
xmlns="https://uidataexchange.org/schemas">00000000000000000000000000000003</NextSt  
ateSOAPTransactionNumber>
```

4.9.7 State Pull Acknowledgement (http request)

```
<To xmlns="https://uidataexchange.org/schemas">Broker</To>  
<From xmlns="https://uidataexchange.org/schemas">CO</From>  
<StateSOAPTransactionNumber  
xmlns="https://uidataexchange.org/schemas">00000000000000000000000000000002</StateS
```

OAPTransactionNumber>

<MessageCode xmlns="https://uidataexchange.org/schemas">2</MessageCode>

4.9.8 EmployerTPA Post (http request)

<To xmlns="https://uidataexchange.org/schemas">CO</To>

<From xmlns="https://uidataexchange.org/schemas">BR000000001</From>

<EmployerTPAResponseFileGUID

xmlns="https://uidataexchange.org/schemas">00000000000000000000000000000010</EmployerTPAResponseFileGUID >

4.9.9 EmployerTPA Post Acknowledgement (http response)

<To xmlns="https://uidataexchange.org/schemas">BR000000001</To>

<From xmlns="https://uidataexchange.org/schemas">Broker</From>

<EmployerTPAResponseFileGUID

xmlns="https://uidataexchange.org/schemas">00000000000000000000000000000010</EmployerTPAResponseFileGUID>

<MessageCode xmlns="https://uidataexchange.org/schemas">2</MessageCode>

4.9.10 EmployerTPA Pull (http request)

<To xmlns="https://uidataexchange.org/schemas">Broker</To>

<From xmlns="https://uidataexchange.org/schemas">BR000000001</From>

<PullCollection xmlns="https://uidataexchange.org/schemas">1</PullCollection>

4.9.11 EmployerTPA Re-Pull by EmployerTPASOAPTransactionNumber (http request)

<To xmlns="https://uidataexchange.org/schemas">Broker</To>

<From xmlns="https://uidataexchange.org/schemas">BR000000001</From>

<PullCollection xmlns="https://uidataexchange.org/schemas">2</PullCollection>

<EmployerTPASOAPTransactionNumber

xmlns="https://uidataexchange.org/schemas">00000000000000000000000000000020</EmployerTPASOAPTransactionNumber>

4.9.12 EmployerTPA Re-Pull by Date (http request)

<To xmlns="https://uidataexchange.org/schemas">Broker</To>

<From xmlns="https://uidataexchange.org/schemas">BR000000001</From>

<PullCollection xmlns="https://uidataexchange.org/schemas">3</PullCollection>

```
<EmployerTPASOAPTransactionNumber  
xmlns="https://uidataexchange.org/schemas"></EmployerTPASOAPTransactionNumber>
```

4.9.13 EmployerTPA Pull (http response)

```
<To xmlns="https://uidataexchange.org/schemas">BR000000001</To>  
<From xmlns="https://uidataexchange.org/schemas">CO</From>  
<EmployerTPASOAPTransactionNumber  
xmlns="https://uidataexchange.org/schemas">00000000000000000000000000000020</Emplo  
yerTPASOAPTransactionNumber>  
<PullCollection xmlns="https://uidataexchange.org/schemas">3</PullCollection>  
<MessageCode  
xmlns="https://uidataexchange.org/schemas">2</BrokerAckFileMessageCode>  
<NextEmployerTPASOAPTransactionNumber  
xmlns="https://uidataexchange.org/schemas">00000000000000000000000000000030</NextE  
mployerTPASOAPTransactionNumber>
```

4.9.14 EmployerTPA Pull Acknowledgement (http request)

```
<To xmlns="https://uidataexchange.org/schemas">Broker</To>  
<From xmlns="https://uidataexchange.org/schemas">BR000000001</From>  
<EmployerTPASOAPTransactionNumber  
xmlns="https://uidataexchange.org/schemas">00000000000000000000000000000020</Emplo  
yerTPASOAPTransactionNumber>  
<MessageCode xmlns="https://uidataexchange.org/schemas">2</MessageCode>
```