Comment: wd-2-0-1-01

Section 2.2

Change from: The ebBP technical specification recognizes and concretely expresses the six defined, Business Transaction patterns-Commercial Transaction, Notification (of Failure), Information Distribution, Request-Response, Request-Confirm, and Query Response.

Change to:

The ebBP technical specification recognizes and concretely expresses the six defined, Business Transaction patterns-Commercial Transaction, Notification, Information Distribution, Request-Response, Request-Confirm, and Query Response.

Section 3.4.2

Change from:

The Business Transaction is defined as an abstract super class. It is associated with the six concrete Business Transaction patterns defined in the UMM:

- Commercial Transaction
- Information Distribution
- Notification (or Notification of Failure [NOF])

Change to:

The Business Transaction is defined as an abstract super class. It is associated with the six concrete Business Transaction patterns defined in the UMM:

- Commercial Transaction
- Information Distribution
- Notification: The Notification of Failure business transaction is based on the Notification pattern.

Section 3.4.9.1

Change from:

A Business Transaction definition specifies whether a response document is required. A Business Transaction with a request only is typically used for notifications (not the Notification Business Transaction pattern described later in this section), while a Business Transaction involving a response may be associated with the formation of contracts and agreements.

Change to:

A Business Transaction definition specifies whether a response document is required. A Business Transaction with a request or response could be used for notifications. A common example of a Response that is a notification is an Advance Ship Notice or Despatch (Dispatch) Advice. Business Transaction involving a response (to a request) may be associated with the formation of contracts and agreements.

Change from:

[Bulleted list]

- Notification (of Failure) : Used for notification of failure in line with a Commercial Transaction pattern. Represents a formal exchange between parties. Typically used to render a Business Transaction as null and void.

Change to:

- Notification: Used for business notifications such as a Notification of Failure

Business Transaction in line with a Commercial Transaction pattern. Represents a formal exchange between parties. Typically, in the case of NOF, used to render a Business Transaction as null and void. An Advance Ship Notice or Status Order are business notifications.

Add (after bulleted list of patterns):

The patterns are applied to Business Transactions.   In a Business Transaction, a Request may be manual, implicit or not apply, whereby the intent of the involved parties may be important. For example:

- If the Request is manual, a Commercial Transaction pattern could be used where the trigger is known when the Request occurs.
- If the Request may or may not be used, the Data Exchange pattern could be used as the parties may bound the scope of how the pattern is used.
- If the Request is implicit (i.e. the Response is based on previous Commercial Transaction), the Notification pattern could be used. The Request may be implicit and the Response indicative of the intent of the parties.

Section 3.6.2.3

Change from:

The Notification pattern is a formal exchange and requires non-repudiation. When the Notification of Failure is used a Business Transaction is set aside.

Change to:

The Notification pattern is a formal exchange and requires non-repudiation. When the Notification of Failure is used (for the Notification pattern), a Business Transaction is set aside.

Appendix A:

Change from:

```
        <!--  Here are just a few OperationMapping(s) to illustrate, one which integrates the business messages
and signals.  It is assumed that not all our ebXML BPSS V2 BTA/ComplexBTAs have been implemented in our
Web Service.  This is only for the brevity of this instance document.  -->
        <OperationMapping businessTransactionActivityRef="BTA100" roleRef="MeSeller100" nameID="OM100"
name="Receive Sales Order">
          <MessageMap operationName="processSalesOrder" documentEnvelopeRef="O100"
operationStep="input" interfaceName="Sales"/>
          <MessageMap operationName="processSalesOrder" documentEnvelopeRef="SOA110"
operationStep="output" interfaceName="Sales"/>
          <MessageMap operationName="processSalesOrder" documentEnvelopeRef="SOR190"
operationStep="fault" interfaceName="Sales"/>
          <MessageMap operationName="receiptNotification" documentEnvelopeRef="RA100"
operationStep="output" interfaceName="Sales"/>
          <MessageMap operationName="receiptNotification" documentEnvelopeRef="RA101"
operationStep="fault" interfaceName="Sales"/>
          <MessageMap operationName="acceptanceNotification" documentEnvelopeRef="AA102"
operationStep="output" interfaceName="Sales"/>
          <MessageMap operationName="acceptanceNotification" documentEnvelopeRef="AAE102"
operationStep="fault" interfaceName="Sales"/>
          <MessageMap operationName="receiptNotification" documentEnvelopeRef="RA110"
operationStep="output" interfaceName="Sales"/>
          <MessageMap operationName="receiptNotification" documentEnvelopeRef="RA111"
operationStep="fault" interfaceName="Sales"/>
          <MessageMap operationName="acceptanceNotification" documentEnvelopeRef="AA112"
operationStep="output" interfaceName="Sales"/>
          <MessageMap operationName="acceptanceNotification" documentEnvelopeRef="AAE112"
operationStep="fault" interfaceName="Sales"/>
        </OperationMapping>
        <OperationMapping businessTransactionActivityRef="Purchase1" roleRef="MeBuyer1000"
```

```
nameID="OM1000" name="Generate Purchase Order">
            <MessageMap operationName="CreatePurchaseOrder" documentEnvelopeRef="O1000"
operationStep="input" interfaceName="PurchaseOrder"/>
            <MessageMap operationName="CreatePurchaseOrder" documentEnvelopeRef="POA1100"
operationStep="output" interfaceName="PurchaseOrder"/>
            <MessageMap operationName="CreatePurchaseOrder" documentEnvelopeRef="POR1900"
operationStep="fault" interfaceName="PurchaseOrder"/>
                        </OperationMapping>
```

Change to:

```
          <!--  Here are just a few OperationMapping(s) to illustrate, one which integrates the business messages
and signals.  It is assumed that not all our ebXML BPSS V2 BTA/ComplexBTAs have been implemented in our
Web Service.  This is only for the brevity of this instance document.  -->
          <OperationMapping businessTransactionActivityRef="BTA100" roleRef="MeSeller100" nameID="OM100"
name="Receive Sales Order">
            <MessageMap operationName="processSalesOrder" documentEnvelopeRef="O100"
operationStep="input" interfaceName="Sales"/>
            <MessageMap operationName="processSalesOrder" documentEnvelopeRef="SOA110"
operationStep="output" interfaceName="Sales"/>
            <MessageMap operationName="processSalesOrder" documentEnvelopeRef="SOR190"
operationStep="fault" interfaceName="Sales"/>
            <MessageMap operationName="receiptSignal" documentEnvelopeRef="RA100" operationStep="output"
interfaceName="Sales"/>
            <MessageMap operationName="receiptException" documentEnvelopeRef="RA101"
operationStep="fault" interfaceName="Sales"/>
            <MessageMap operationName="acceptanceSignal" documentEnvelopeRef="AA102"
operationStep="output" interfaceName="Sales"/>
            <MessageMap operationName="acceptanceException" documentEnvelopeRef="AAE102"
operationStep="fault" interfaceName="Sales"/>
            <MessageMap operationName="receiptSignal" documentEnvelopeRef="RA110" operationStep="output"
interfaceName="Sales"/>
            <MessageMap operationName="receiptException" documentEnvelopeRef="RA111"
operationStep="fault" interfaceName="Sales"/>
            <MessageMap operationName="acceptanceSignal" documentEnvelopeRef="AA112"
operationStep="output" interfaceName="Sales"/>
            <MessageMap operationName="acceptanceException" documentEnvelopeRef="AAE112"
operationStep="fault" interfaceName="Sales"/>
          </OperationMapping>
          <OperationMapping businessTransactionActivityRef="Purchase1" roleRef="MeBuyer1000"
nameID="OM1000" name="Generate Purchase Order">
            <MessageMap operationName="CreatePurchaseOrder" documentEnvelopeRef="O1000"
operationStep="input" interfaceName="PurchaseOrder"/>
            <MessageMap operationName="CreatePurchaseOrder" documentEnvelopeRef="POA1100"
operationStep="output" interfaceName="PurchaseOrder"/>
            <MessageMap operationName="CreatePurchaseOrder" documentEnvelopeRef="POR1900"
operationStep="fault" interfaceName="PurchaseOrder"/>
                        </OperationMapping>
```

Appendix F:

Change from:

Notification of Failure	A defined business message that is sent when an unwanted event that could reasonably be anticipated occurs, such as a party cannot determine if a contract has been formed

Change to:

Notification of Failure	A Business Transaction based on the Notification pattern that involves a defined business message that is sent when an unwanted event that could reasonably be anticipated occurs, such as a party cannot determine if a contract has been formed

Add:

Notification	A Business Transaction based on the Notification pattern that involves a defined business message that is sent when an unwanted event that could reasonably be anticipated occurs, such as a party cannot determine if a contract has been formed

Pattern	A pattern specifies the type of the message exchange (request, response and signals) that applies for a given Business Transaction definition. It is a way to define classes of Business Transaction definitions

Schema:
(annotation only)

Change from:

```
....
- <xsd:element name="Notification" substitutionGroup="BusinessTransactionHead">
- <xsd:annotation>
  <xsd:documentation>A concrete Business Transaction Pattern where there is a formal
information exchange between parties that affects the previous completion of a
Commercial or Business Transaction (of the BusinessTransactionType). The Notification
pattern for Notification of Failure involves a business message. Note: This concrete
pattern is added in v2.0.</xsd:documentation>
  </xsd:annotation>
...
```

Change to:

```
...
- <xsd:element name="Notification" substitutionGroup="BusinessTransactionHead">
- <xsd:annotation>
  <xsd:documentation>A concrete Business Transaction Pattern where there is a formal
information exchange between parties that may affect the previous completion of a
Commercial or Business Transaction (of the BusinessTransactionType). For example,
when the Notification pattern is used for the Notification of Failure Business Transaction,
this involves a business message. Another example of a business notification is an
Advance Ship Notice. Note: This concrete pattern is added in v2.0.</xsd:documentation>
  </xsd:annotation>
...
```