ebXML BPSS Summary

Note: This summary is thanks to the work, words and efforts of Karsten Reimer, formerly of Sun, and Jean-Jacques Dubray, original BPSS editor currently from Attachmate. The main drawings and modeling diagrams were developed by JJ Dubray and published earlier this year to the public W3C WS-Choreography list and other details have been added.

Introduction
The ebXML BPSS (Electronic Business extensible Markup Language Business Process Specification Schema) specification is part of the ebXML set of specifications and deals with both design-time and run-time aspects of “business-to-business processes” (or collaborations) between two or more business partners. The specification includes:

- A business process specification schema which is used
  - At design time to specify the terms of the collaboration
  - At run-time to effectively but optionally validate the occurrences of these terms
- The specification of a run-time protocol that is used to guarantee state alignment between the collaboration

A business collaboration is expressed as a choreography of “Business Transactions” between two parties. (BinaryCollaboration). As the core of the BPSS technical specification, a Business Transaction is an atomic message exchange between 2 abstract roles (initiator and responder). A business transaction follows very specific message exchange patterns based on the purpose of the business transaction (notification, commitment, etc.). A business transaction represents a state of the choreography definition (i.e. the choreography is in the state of executing this business transaction). Transitions between states specify the overall choreography. Transitions can be guarded on business transaction status (success, business failure, technical failure, etc.), or on the content of business document exchanged as part of the business transaction.
BPSS is a collaboration specification which primary use is for interaction between businesses. BPSS has a full complete set of commercial transaction semantics. It is used with the ebXML CPA (Collaboration Protocol Agreement), which can be based CPP (Collaboration Protocol Profiles) [Technical system capabilities of a partner].

BPSS relevant artifacts are shown in the redlined boxes. Graphic provided by Sun Microsystems.

---

1 BPSS relevant artifacts are shown in the redlined boxes. Graphic provided by Sun Microsystems.

ebXML Business Process Technical Committee, OASIS, 12 November 2003
Non-normative summary
Page 2 of 7
Status
The BPSS version 1.01 was released in May 2001. A candidate draft version 1.05 was issued in July 2002, and version 1.1 subsequently released by the Techniques and Methodologies Group (TMG), UN/CEFACT (United Nations/Centre for Trade Facilitation and Electronic Business). The ebXML BPSS has been based on the UN/CEFACT Modeling Methodology (UMM) and meta-model, which took many of their roots from UML™, (Unified Modeling Language)\(^2\), constructs. The ebXML BPSS XML representation and framework continues its work under the OASIS (Organization for the Advancement of Structured Information Standards) ebXML Business Process TC (ebBP).

Details

Business Transaction
The core of the BPSS is the Business Transaction, a message exchange between 2 abstract roles (initiator and responder). The initiator sends a request to the responder. The responder may optionally respond with one of n possible message. Each message can be tagged to be a “positive” or “negative” response. A message is composed of a primary business document (XML) and any number of attachments (binary or XML).

The Business Transaction is implemented using standards patterns, the six defined in Chapter 8 of the UMM version 10 (for example, commercial transaction, request/confirm, request/response, information distribution, etc). The BPSS provides the capability to specify whether or not the Business Transaction is intended to be legally binding or not.

Each Business Transaction is expressed as an exchange of electronic business documents, where the sequence of exchange is determined by the business process and security and messaging considerations. The business documents are composed of reusable Business Information Objects.\(^3\)

Here is an example of document definition:

```xml
<BusinessDocument name="BD:Shipment insurance contract"
specificationLocation= "http://www.YeOldInsurer.com/Schemas/v1.1/shp_ins_ctc.dtd"
specificationElement="ShipmentInsuranceContract">
  <ConditionExpression expressionLanguage="XPath"
 expression='//Status="Accept"'/>
</BusinessDocument>
```

(Documents are logical documents and can be composed of a physical document definition and an XPath condition). This can be used to define a NAKPO document that is actually an AckPO document typed with a specific tag content.

There is a business transaction protocol associated with this message exchange. BPSS specifies two kinds of signals: receipt acknowledgement and acceptance acknowledgement. Both signals are optional and can be returned to both the request and the response. The schema of a signal is fixed. They are not designed to convey other meanings (for interoperability reasons), but can convey specific error messages.

\(^2\) Trademarked by Object Management Group (OMG)
\(^3\) In the most current version 1.90 ebXML Core Components specification, these are called Business Information Entities (BIE). There are also higher-level entities called Business Entities that may be integrated into BPSS in the future.
- Receipt Acknowledgement: Message was received and passed structural validity checks by the receiving party.
- Acceptance acknowledgement: The system passed the business rules of the receiving party and is now in a system of record (whatever the rules and the system are – this is not exposed in BPSS).

These signals guarantee “state alignment” between parties. These signals are part of a layer on top of reliable messaging, and in particular they assume that messages and signals are sent via reliable communication channels.

The BPSS also provides parameters in business terms that can be used to specify certain levels of security and reliability. These parameters can be used to instruct the CPP and CPA to require the BSI (Business Service Interface) or delivery channel to achieve these service levels.

Generally, the BSI is the runtime software that can isolate the internal communications of a given legacy or other application from the collaboration model, and once built represent the party in a collaboration model.

Here is an example of BT definitions:

```xml
<BusinessTransaction name="BT:Insure shipment">
  <RequestingBusinessActivity name="ReqBA:Shipment insurance request">
    <DocumentEnvelope businessDocument="BD:Shipment insurance request">
      <Attachment businessDocument="BD:Equipment description"/>
    </DocumentEnvelope>
  </RequestingBusinessActivity>
  <RespondingBusinessActivity name="RspBA:Shipment insurance response">
    <DocumentEnvelope businessDocument="BD:Shipment insurance contract" isPositiveResponse="true">
      <Attachment businessDocument="BD:Equipment description"/>
    </DocumentEnvelope>
    <DocumentEnvelope businessDocument="BD:Reject shipment insurance" isPositiveResponse="false">
    </DocumentEnvelope>
  </RespondingBusinessActivity>
</BusinessTransaction>
```
Here is a possible notation for the business transaction definition:

Initiator

Requesting Business Activity

Responding Business Activity

SUCCESS

ANYFAILURE

Shipment insurance request

Shipment insurance contract

Reject shipment insurance

Success

Failure
This is what happens during a business transaction activity:

A business transaction can have exceptions (technical and business exceptions, including timeouts). The choreography can use these exceptions.

Here is a possible scenario:

**Business Collaboration**

A BPSS collaboration (binary or multi-party) is a series of "usage" of business transaction definition. This usage is called a "business transaction activity". Collaboration can also be composed via a collaboration activity. A binary collaboration is defined between two roles (buyer and seller, for example). A business transaction activity will then bind the abstract roles of the business transaction to the concrete roles of the collaboration (initiator-> buyer, etc., for example). In particular the same business transaction could potentially be used in either direction (this is useful for instance in a "cancel PO" transaction when each party can potentially decide to cancel the order).
Here is a possible notation using UML activity diagrams:

![UML Activity Diagram]

The control flow is based on a state/transition model and is very close to the UML™ activity diagram (transition[source,target], fork, join, start, end). Transitions can be guarded either by: exception types, message types (got an AckPO response or a NAKPO [Negative acknowledgment] response), or XPath statements on the content of documents. All these concepts are explicitly defined in a BPSS definition (i.e., the BPSS is declarative in nature). There is no notion of transaction scopes and associated compensating transactions. Compensation schemes must be explicitly specified in a BPSS instance since there is no construct allowing an implicit behavior.

Here is the corresponding document:

```xml
<BinaryCollaboration name="BC:Insure shipment" timeToPerform="P7D">
  <InitiatingRole name="AR:Seller"/>
  <RespondingRole name="AR:Insurer"/>
  <Start toBusinessState="BTA:Insure shipment request"/>
  <Transition fromBusinessState="BTA:Insure shipment request" toBusinessState="BTA:Send invoice">
    <ConditionExpression expressionLanguage="XPath" expression="//Value[@insuredValue<10000]"/>
  </Transition>
  <Transition fromBusinessState="BTA:Insure shipment request" toBusinessState="BTA:Process down payment">
    <ConditionExpression expressionLanguage="XPath" expression="//Value[@insuredValue>=10000]"/>
  </Transition>
  <Transition fromBusinessState="BTA:Process down payment" toBusinessState="BTA:Send invoice" conditionGuard="Success"/>
  <Failure fromBusinessState="BTA:Insure shipment request" conditionGuard="AnyFailure"/> …
</BinaryCollaboration>
```