PROYECTO DE GRADO

Presentado ante la ilustre UNIVERSIDAD DE LOS ANDES como requisito parcial para obtener el Título de MAGISTER SCIENTIAE EN COMPUTACIÓN

COMPARACIÓN DE WSBPEL VS. EBXML BPSS, MEDIANTE EL DESARROLLO DE DOS PROCEDIMIENTOS PARA LA DESCRIPCIÓN DE PROCESOS DE NEGOCIOS; ENFOQUE EN PYME.

Por

Ing. Jackson Johan Guillen Mora
Tutor: Dr. Edgar Chacón R.

Noviembre 2007

A mi abuelo…

Agradecimientos

A esa fuerza divina, como quiera que la llamemos, por darme la oportunidad de ser y por los momentos que me ha permitido vivir.

A mi madre, por su amor incondicional y estar conmigo en los momentos más difíciles.

A mi hermana, por ser mi amiga y aliada.

A mi abuela, tía Ana, Reinaldo y el resto de mi familia por siempre creer en mí.

A Luisita, Milagros, Taniana y Vladimir por su aprecio y ayuda en el postgrado.

A Judith, Edgar y Jonas por su confianza y apoyo en la culminación de este trabajo de grado.

A todas las personas que de una u otra forma me apoyaron en mi postgrado y en este trabajo.

Resumen

El uso de Internet y la globalización de la información son hoy en día tendencias claras de nuestra sociedad. En este sentido la disminución de los costos de comunicación, los avances en las tecnologías de información y la presencia de una red pública de cobertura mundial han tenido una gran influencia.

Fundamentadas en estas tendencias han surgido, desde hace ya un tiempo, ideas claras de hacia dónde se dirigen los negocios y la manera de cómo estos se realizan. Los negocios electrónicos son una realidad que cada día se extiende más superando los últimos obstáculos o limitaciones para una acogida más amplia y a todos los niveles de negocios.

Una de estas limitaciones de los negocios es la interoperabilidad consistente y uniforme entre empresas. En este contexto se han planteado, adoptado y se siguen generando varias soluciones cada vez más cercanas al que se presume será el estándar definitivo.

ebXML BPSS y *WSBPEL*(BPEL4WS) son dos de las propuestas de envergadura más recientes que plantean la interoperabilidad a través de esquemas propios para la descripción de los procesos de negocios. *ebXML BPSS* es el estándar propuesto por el centro de la ONU para la facilitación del intercambio y los negocios electrónicos (*UN/CEFACT*) y por la organización para el avance de los estándares estructurados de información (*OASIS*) (1). *WSBPEL* es el estándar propuesto por Microsoft, *IBM* y *BEA* (2). La importancia de ambos estándares da pie al estudio y comparación de ellos.

En el presente trabajo se ha ofrecido una perspectiva diferente a la de comparar sintáctica o semánticamente estos estándares, pues se ha planteado un comparación pragmática mediante el desarrollo de dos procedimientos para la descripción del dominio público de los procesos de negocios, que usen los estándares en el escenario de las pequeñas y medianas empresas. Como resultado se ha encontrado que *WSBPEL* es más conveniente y altamente recomendado para PyME.

Índice

Índice de Figuras

Capítulo 1

Introducción

En el presente trabajo se hace necesario presentar conceptos esenciales que permiten entender mejor el contexto en el cual se ubica. Partiendo de la importancia del comercio electrónico, se pasa luego a mencionar la necesidad de la interoperabilidad, el uso de estándares y la utilizando de servicios Web. Acá se introduce conceptos como *EDI*[1], *SOAP*[2] y *WSDL*[3]. Luego se define la diferencia entre el dominio público (*B2B*)[4] y el dominio privado (*EAI*)[5] de los procesos de negocios para presentar entonces algunas de las familias de tecnologías utilizadas para describirlos. Presentando conceptos como *XLANG*, *BPML*[6], *WSFL* y la evolución que han seguido, para luego ubicar las tecnologías que son tema de estudio. Finalmente se introduce el objetivo y justificación del trabajo.

1.1 Interoperabilidad y Conceptos Esenciales

1.1.1 Comercio Electrónico

Entendido como la noción más genérica de hacer negocios a través de medios electrónicos sin vender necesariamente, brinda a las empresas grandes ventajas, entre ellas:

- Mayor competitividad para las pequeñas empresas pues al no poseer grandes recursos económicos e infraestructurales para detectar clientes potenciales y ofrecer servicio, utilizan Internet como medio de difusión aprovechando que allí la confección de la oferta y la manera de transmitirla toman un peso relativamente mayor.

---

[1] Electronic Data Interchange
[2] Simple Object Access Protocol
[3] Web Services Description Language
[4] Business to Business
[5] Enterprise Application Integration
[6] Business Process Modeling Languge

- Se reducen los costos operativos y se tiene una mayor eficiencia sobre todo en las relaciones entre empresas al automatizar los procesos y evitar errores humanos en las transacciones. Igual con el consumidor pues actualizar un catálogo electrónico es mucho más económico y que uno impreso.
- Se ofrece información actualizada a los clientes y mejora de los servicios relacionados con tener la última información de una manera cómoda y rápida.
- Se extienden las posibilidades del negocio y el servicio al cliente al poder trabajar las 24 horas (3).
- Se tiene de un canal directo (Internet) que permite una mayor cercanía entre proveedores y clientes.
- Por ser Internet un canal interactivo es factible tener información acerca del comportamiento de los clientes que permite afinar estrategias de la empresa (3).

De lo anterior resulta evidente el potencial del comercio electrónico y el porqué del gran interés en explotarlo de manera amplia y a todos los niveles. Es así como tratar de solventar el problema de la complejidad y variedad de la información a lo largo de los diferentes sectores industriales e incluso entre diferentes empresas de un mismo sector es una prioridad.

La interoperabilidad no es un problema nuevo y se han definido y adoptado algunos estándares para tal fin.

1.1.2 Interoperabilidad

Interoperar, es la capacidad de equipos y componentes de software de interactuar, realizando operaciones entre ellos, independientemente de la plataforma de cada uno. Es un medio, como cualquier otra tecnología, para alcanzar un objetivo en común de integración entre diferentes entes de una manera coherente y predecible.

1.1.3 Un Estándar para Interoperabilidad

Es un esfuerzo por crear un uso a gran escala de protocolos y formatos para permitir que software de diferentes vendedores interoperen. Un estándar para interoperabilidad provee el centro de gravedad para jugadores de la industria que los orienta a ellos mismos y coordina decisiones de inversión. Un estándar es necesario para promover comunicaciones transparentes que permiten la interacción entre los diferentes productos de tecnología de la información.

El primer estándar de comercio electrónico ampliamente adoptado fue *EDI* (Intercambio de datos electrónicos). Fue creado en 1970 y estaba orientado a facilitar el intercambio de información entre empresas sin intervención humana, siempre que los participantes se hayan puesto de acuerdo sobre el formato de los datos (3). Ha evolucionado pero algunas de las desventajas que lo hacen no viable, comparado a las nuevas iniciativas, son:

- *EDI* y sus extensiones son ambiguos pues fueron diseñados tan genéricamente que cada compañía debe crear una guía de implementación generando grandes costos y complejidad (4).
- No se publicó con el estándar información de uso y mejora del proceso de negocios *EDI* por lo que los usuarios, administradores y desarrolladores frecuentemente se confunden en como *EDI* se aplica a su proceso de negocios (4).
- Debido a la ambigüedad y carencia en los procesos de información las negociaciones inter-empresas consumen 70 % del tiempo de implementación (4).
- Los estándares *EDI* fueron basados en tecnología de 30 años que utilizaban sistemas mainframe, etc. que no explotan la habilidad de procesos interactivos y la gran capacidad de procesamiento existente hoy.
- *EDI* es "pesado", requiere de una personalización para cada empresa y de la existencia de *VPN*[7]. Estas últimas, las redes privada virtuales, hacen aun más costoso el uso de esta tecnología que hasta ahora ha sido un lujo solo al alcance de las grandes compañías (3).

Otras herramientas también han surgido para facilitar la integración y automatización de los procesos de negocios. Entre ellas se incluyen herramientas para modelado grafico de procesos, tecnologías *middleware* como *CORBA* y *JMS*[8], agentes de integración, Sistemas de gestión de procesos de negocios (*BPMS*) y servidores *B2B*. Desafortunadamente, hasta hace poco, los gastos requeridos por las empresas para integrar los sistemas informáticos dentro de la organización y más allá del firewall han sido muy altos. La principal razón es la variedad de interfaces propietarias y formatos de datos usados para cada aplicación, haciendo necesario grandes recursos tanto para obtener costosas herramientas de integración como de tiempo y capacitación para desarrollar la integración.

[7] Virtual Private Network
[8] Java Message Service

El uso de tecnologías de servicios Web está cambiando esto, al reemplazar las interfaces propietarias y formatos de datos con estándares de bajo costo y ampliamente difundidos para interfaces y datos, que trabajen tanto dentro de la organización como mas allá del firewall.

La primera generación de tecnología de servicios Web se ha enfocado en los fundamentos sobre mensajes soportados por *SOAP* y *WSDL*. Mientras *SOAP* se centra en la codificación de mensajes y la definición de llamadas a funciones remotas en *XML*[9], *WSDL* provee un mecanismo para describir un conjunto de llamadas a funciones remotas (operaciones) como un puerto coherente que puede ser ubicado por un mensaje *SOAP*. *SOAP* y *WSDL* permiten a los diseñadores crear envoltorios de servicio Web alrededor de sus aplicaciones que exponen la funcionalidad a través de un lenguaje estandarizado de definición de interfaces.

Mientras estos fundamentos son suficientes para algunas necesidades internas de integración de aplicaciones y bases para otras soluciones, no permiten una interoperabilidad coherente y predecible. Surgen entonces otras propuestas más amplias fundamentadas en la descripción de los procesos de negocios.

1.1.4 Procesos de Negocios

Pueden ser divididos en dos dominios (5):

- Público: Los procesos públicos son aquellos que una empresa comparte con sus clientes, proveedores y otros asociados. Este es el dominio de integración empresa a empresa (*B2B*).
- Privado: Los procesos privados son aquellos internos de la empresa. Es el dominio de integración de aplicaciones empresariales (*EAI*).

[9] eXtensible Markup Language

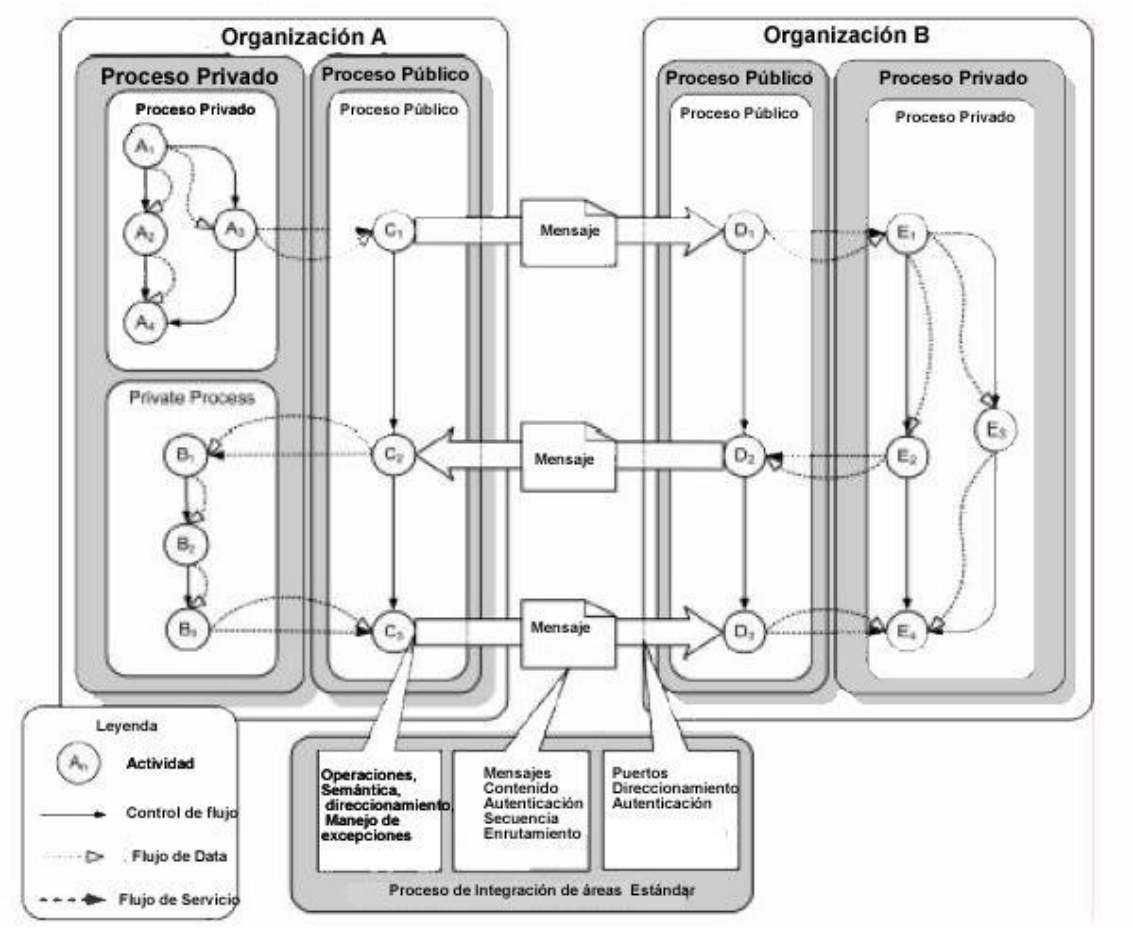

Figura 1 : Ejemplo de procesos públicos y privados. [Tomada de (6)]

La figura 1 permite apreciar la diferencia e interacción entre ambos dominios así como entre dos empresas. Los participantes tienen operaciones internas en forma de procesos privados (A, B, C) los cuales no son expuestos a la otra empresa. Los procesos públicos son creados y sus actividades (C1 a C3 y D1 a D3) sirven como puntos de contacto para el otro participante.

Teniendo presente los conceptos de procesos de negocios de dominio público y privado se puede introducir algunas de las familias de tecnologías que proponen solventar el problema de interoperabilidad.

1.1.5 Tecnologías para la Interoperabilidad

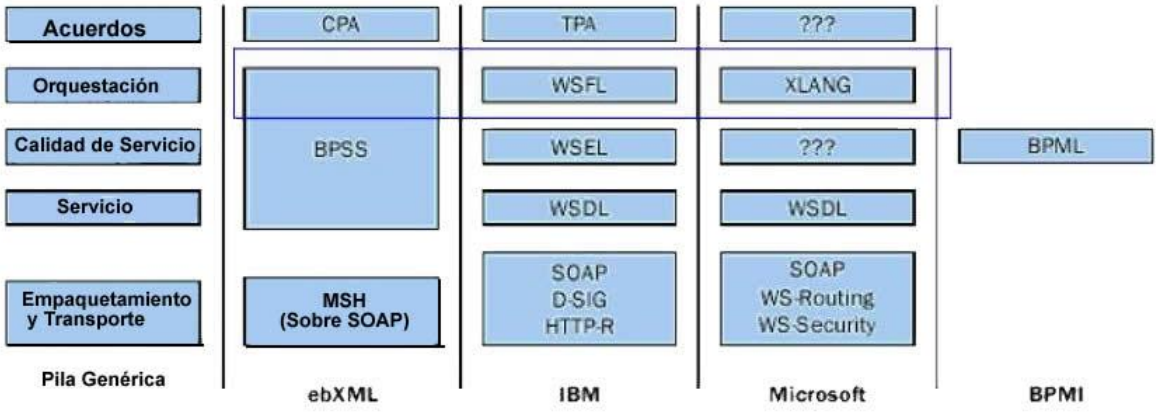

Figura 2 : Algunas tecnologías para la interoperabilidad.[Tomada de (5)]

Al omitir algunos elementos importantes que no son objetivo del presente trabajo se puede mencionar con respecto a la figura 2, lo siguiente:

- La capa de orquestación describe como los servicios interactúan correspondientemente utilizando descripciones de procesos de negocios. También referida como la capa de coreografía.
- *XLANG* es una propuesta de Microsoft que se enfoca completamente en los procesos públicos.
- *WSFL*[10] es una propuesta de *IBM* que cubre tanto los procesos públicos como privados.
- *BPML* es una especificación de BPMI.ORG. Su objetivo es proveer una manera coherente de describir los procesos de una empresa. Se centra en el dominio privado y utiliza otras propuestas, como *ebXML BPSS*[11], complementariamente para describir el dominio público.

Algunas de estas familias han cambiado dando origen a uno de los estándares que son tema de estudio. Es el caso de *WSBPEL*[12] que se deriva de *XLANG* y *WSFL* siendo producto de un acuerdo entre Microsoft, IBM y BEA como muestra la figura 3:

[10] Web Services Flow Language
[11] E-business XML Business Process Specification Schema
[12] Web Services Business Process Execution Language

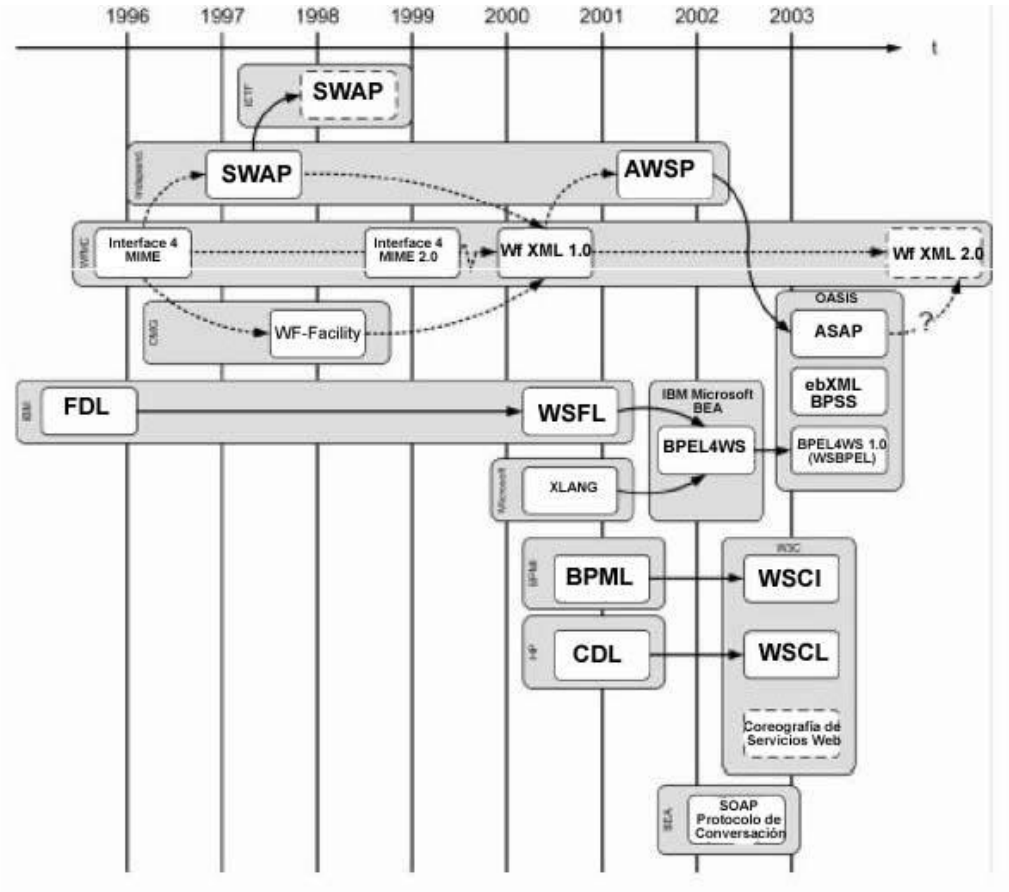

Figura 3 : Evolución de algunas tecnologías para interoperabilidad.[Tomada de (6)]

Finalmente se llega a las dos propuestas que por su relevancia actual e importancia son objeto de este estudio. Estas son *ebXML BPSS* y *WSBPEL*. Ambas pretenden ser la solución, o parte de ella, para una interoperabilidad consistente y uniforme. Aunque *WSBPEL* también puede describir procesos privados, el presente estudio se centra en la descripción del dominio público que es común para ambas propuestas. Para ambos estándares se han tomado las versiones más recientes, para la fecha de culminación del presente trabajo, *ebXML BPSS* v.2.0.4 de Diciembre 2006, y *WSBPEL* v.2.0 de Abril del 2007.

1.1.6 El Contexto de PyME

La Pequeña y Mediana Empresa(PyME) es un concepto muy difundido mundialmente. Las definiciones dadas son muy divergentes ya que los factores que dan definición a una PYME son considerados de diferente manera en cada país. Es casi un hecho el poder afirmar que existe una definición de PYME para cada país, a lo cual se suma las de los organismos internacionales, instituciones varias, congresos y convenciones, etc. No ha sido posible aún unificar criterios globales, esto es en parte lógico dado los diferentes escenarios en cada país, región, economías, significación y dimensiones de empresas a confrontar. Una definición general, aunque poco precisa de PYME es: Un tipo de empresa con un número reducido de trabajadores, y cuya facturación es moderada (7).

Sobre las características que la mayoría de ellas posee si existe un acuerdo, pues independientemente del criterio que las define, las razones que las mueven, los problemas que afrontan y el menor tamaño con respecto a las grandes empresas con sus implicaciones, es un común denominador.

Las características que establecen el contexto de PyME son:

- El presupuesto para inversiones tecnológicas es reducido.
- Sus sistemas no son tan heterogéneos como los de grandes empresas.
- No están constituidas por muchas sedes e incluso pueden tener una sola en muchos casos.
- La necesidad de vincularse con grandes empresas es importante pues en muchos casos dependen de estas relaciones.
- La escalabilidad y sencillez en las herramientas que emplean, son factores importantes pues no poseen, en muchos casos, recurso humano para dedicar por completo a actividades de asimilación y adopción de tecnologías.
- Tienen gran interés en aprovechar ventajas competitivas provenientes de la innovación siempre y cuando impliquen inversiones económicas relativamente pequeñas, y se pueda tener un retorno de inversión a corto plazo.

Este contexto está presente a lo largo del trabajo, en la percepción del autor, permitiendo generar conclusiones al final del trabajo sobre la base de la observación y análisis continuo.

1.2 Objetivo y Justificación

1.2.1 Objetivo

Desarrollar y contrastar dos procedimientos, uno que emplee *ebXML BPSS* y otro que utilice WSBPEL, para la descripción de la interfaz visible o dominio público de los procesos de negocio, con enfoque en las pequeñas y medianas empresas (PyME).

1.2.2 Justificación

La velocidad actual a la que se mueve la tecnología motiva la búsqueda de oportunidades de aprendizaje e innovación que permitan el crecimiento profesional y las posibilidades de contribución. Es así como el comercio electrónico y sus tendencias actuales ofrecen todo esto, al considerar la presión e interés mundial por el estudio y adopción de un estándar para la interoperabilidad amplia y a todos los niveles. Algunas de las motivaciones para la creación de *ebXML BPSS* y *WSBPEL* fueron explotar las tecnologías más recientes para ofrecer mejores propuestas a los planteamientos adoptados hasta ahora como *EDI* (Intercambio de datos electrónicos).

Unas de las mejoras que se han realizado a las soluciones aún vigentes, pero que cada día se vuelven más obsoletas, son las posibilidades (menor costo, menor complejidad, mayor interoperabilidad) que ofrecen a pequeñas y medianas empresas en cualquier lugar. De allí que concebir procedimientos que exploten estas características se vuelve importante y muy útil.

Poder analizar los estándares desde una perspectiva particular, surgida de una comparación pragmática dada al desarrollar los dos procedimientos que serán propuestos, brinda una gran motivación al estar generándose un trabajo único e interesante.

El trabajo se estructura en los siguientes capítulos: En el capítulo actual se introducen los conceptos vinculados al planteamiento del tema de estudio. El capítulo 2 establece el contexto necesario de *WSBPEL* mientras el capítulo 3 hace lo mismo para *ebXML BPSS*. El capítulo 4 se dedica al empleo de *WSBPEL* y es allí donde se establece el procedimiento para descripción de procesos de negocios para dicho estándar. El capítulo 5 es similar al 4 pero aborda el estándar *ebXML BPSS*. El capítulo 6 se dedica a la comparación de ambos estándares y el capítulo 7 a las conclusiones.

Capítulo 2

WSBPEL

En el presente capítulo se establece el contexto necesario para WSBPEL.

2.1 Introducción a WSBPEL

El estándar *WSBPEL* (Web Services Business Process Execution Language) define un lenguaje para especificar el comportamiento de procesos de negocios sobre la base de servicios Web. *WSBPEL* utiliza exclusivamente interfaces Web para exportar e importar la funcionalidad de los procesos.

Para *WSBPEL* los procesos de negocios se pueden clasificar en:

- Procesos de negocios ejecutables: Modelan el comportamiento actual de un participante en una interacción de negocios.
- Procesos de negocios abstractos (públicos): Son procesos parcialmente especificados que no se pretende sean ejecutados. Un proceso abstracto oculta detalles operacionales concretos. Ellos tienen un rol descriptivo con enfoque principal en el comportamiento observable.

WSBPEL tiene como objetivo modelar el comportamiento tanto de los procesos abstractos como los ejecutables, mediante un lenguaje que permita especificarlos. *WSBPEL* al definir ambos procesos extiende el modelo de interacción de los servicios Web y permite el soporte de transacciones de negocios. *WSBPEL* define un modelo de integración que facilita la expansión de procesos de integración automatizados, tanto en la intranet como en la extranet.

2.1.1 Servicios Web y WSBPEL como Modelo de Integración

Uno de los objetivos de la tecnología de servicios Web es alcanzar interoperabilidad entre aplicaciones, mediante el uso de estándares Web. Los servicios Web usan un modelo de pares para permitir una integración flexible de sistemas heterogéneos en una variedad de dominios, incluyendo negocio a

consumidor, negocio a negocio e integración de aplicaciones empresariales. Las siguientes especificaciones básicas definen el espacio de servicios Web:

- *SOAP* (8).
- *WSDL* (9).
- *UDDI* [13] (10).

SOAP define un protocolo de mensajes *XML* para servicios básicos de interoperabilidad. *WSDL* introduce una gramática común para describir servicios. *UDDI* provee la infraestructura requerida para publicar y descubrir servicios de una manera sistemática. Estas especificaciones, juntas, permiten que aplicaciones se encuentren unas a las otras e interactúen siguiendo un modelo de plataforma independiente (de acoplado impreciso) (2).

La integración de sistemas requiere más que la habilidad para conducir simples interacciones mediante el uso de protocolos estándares. El máximo potencial de servicios *Web* sólo será alcanzado cuando aplicaciones y procesos de negocios sean capaces de integrar sus complejas interacciones mediante el uso de un modelo estándar de integración de procesos. El modelo de interacción que es directamente soportado por *WSDL* es en esencia un modelo sin estados de solicitud-respuesta o de interacciones de una vía no-correlacionadas.

Los modelos para interacciones de negocios típicamente asumen secuencias de intercambio de mensajes uno-a-uno, tanto de solicitud como de respuesta, dentro de interacciones de largo-plazo con estado. Para definir tales interacciones, una descripción formal de los protocolos, para intercambio de mensajes, es necesaria. Un proceso abstracto puede ser usado para describir el comportamiento observable de intercambio de mensajes de cada participante, sin revelar sus implementaciones internas. Hay dos razones para separar los aspectos públicos de los aspectos internos o privados. Uno es que obviamente los participantes no quieren revelar todas sus decisiones internas tomadas y la gestión de data a sus compañeros de negocios. La otra razón es que, aún donde no sea el caso separar procesos públicos de privados, provee la libertad para cambiar aspectos privados de la implementación del proceso sin afectar el comportamiento observable. El Comportamiento observable debe ser descrito de tal manera que sea independiente de la plataforma y capturar aspectos del comportamiento que pueden tener significado económico inter-empresarial.

Los siguientes conceptos por describir procesos de negocios deben ser considerados:

---

[13] Universal Description, Discovery, and Integration

- Los procesos de negocios incluyen datos que dependen del comportamiento. Por ejemplo, una cadena de suministro depende de datos como el número de elementos ordenados, el valor total de la orden, o la fecha de entrega. El intento de definir procesos en estos casos requiere el uso de condicionales y de constructores para el control de tiempo.
- La habilidad para especificar condiciones de excepción y sus condiciones, incluyendo secuencias para recobrarse, es al menos tan importante para los procesos de negocios como la habilidad para definir el comportamiento cuando todo debe salir bien.
- Interacción de largo plazo incluyen múltiples unidades de trabajo, en muchos casos anidadas, cada una con sus propios requerimientos de datos. Procesos de negocios frecuentemente requieren coordinación entre participantes del resultado (éxito o falla) de unidades de trabajo a varios niveles de granularidad.

Los conceptos básicos de *WSBPEL* pueden aplicar en una de dos vías abstracta o ejecutable. Un proceso abstracto *WSBPEL* es un proceso parcialmente especificado que no se intenta sea ejecutable y que debe ser declarado como abstracto. Mientras los procesos ejecutables deben estar completamente especificados para poder ser ejecutados, un proceso abstracto puede ocultar algunos de los requerimientos operacionales concretos.

Todos los constructores de procesos ejecutables están disponibles para procesos abstractos, compartiendo así el mismo poder de expresión. Adicional a las características disponibles en procesos ejecutables, procesos abstractos proveen dos mecanismos para ocultar detalles operacionales:

- Uso de elementos opacos. Los elementos sólo se nombran y se les da el atributo de opaco no debiéndose definir completamente.
- Omisión. Los elementos sencillamente no se incluyen.

Aún cuando una definición de proceso abstracto podría contener información que lo haría ejecutable, su condición de abstracto establece que ninguna realización de esto sea permitida. Los procesos abstractos tienen un rol descriptivo, y se pueden aplicar en múltiples casos, incluso algunos sin definir. Uno de estos casos podría ser describir el comportamiento observable de algunos de los servicios ofrecidos por un proceso ejecutable. Otro podría ser la creación de una especie de plantilla que excluye detalles de ejecución que serán completados cuando se mapee a un proceso de ejecución. El primer caso, descripción del comportamiento público, es en donde se enfoca el presente trabajo.

Independientemente de su uso, todos los procesos abstractos comporten una base sintáctica que especifica las características que definen el universo sintáctico de los procesos abstractos. Con esta base,

luego se usan perfiles para definir la especialización necesaria sobre la base ejecutable de un caso particular.

Es posible usar *WSBPEL* para definir un proceso de negocios ejecutable y aunque un proceso abstracto no requiere ser especificado completamente, el lenguaje efectivamente define un formato de ejecución para procesos de negocios cuya base es exclusivamente servicios *Web* y datos *XML*. Incluso, tales procesos se ejecutan e interactúan con los participantes de una manera consistente sin importar la plataforma o modelo programático del ambiente en que se implantan.

WSBPEL define un modelo y una gramática para describir el comportamiento de un proceso de negocios en base a las interacciones entre este proceso y sus participantes. La interacción con cada compañero ocurre a través de interfaces de servicios *Web*. El proceso *WSBPEL* define como múltiples interacciones de servicios con sus parejas son coordinadas para alcanzar una meta común, así como también, el estado y lógica necesaria para esta coordinación.

2.2 Sobre los Conceptos Técnicos de WSBPEL

Para entender los conceptos técnicos específicos de *WSBPEL*, su sintaxis y una visión detallada de las relaciones entre ellos, así como su aplicación, se hace necesaria la consulta del esquema *WSBPEL* para procesos de negocios abstractos (anexo A.1) y de la especificación técnica *WSBPEL* (2)[14]. A continuación se sintetizan sólo algunos de estos conceptos:

- *PartnerLinks*: Ellos identifican relaciones entre los procesos de negocios y el resto de los participantes (servicios *Web*). Básicamente proveen definiciones *WSDL* de tipos de puertos para las interacciones de procesos.
- Variables: Ellas pueden transportar datos en mensajes y definir el estado cada instancia del proceso. Pueden contener *PartnerLinks*. Por lo tanto pueden ser usadas para conectar dinámicamente estructuras unas a las otras.
- *Correlation Sets*: Ellos identifican interacciones relevantes para una instancia de un proceso, siendo usados para manejar mensajes correctamente entre diferentes sesiones.
- Actividades: Describen el comportamiento del proceso de negocios. Pueden ser básicas o estructuradas.

[14] No incluida por tener más de 250 páginas

- *Receive*: Acepta un mensaje a través de la invocación de una operación especifica de una pareja participante.
- *Reply*: Envía un mensaje como una respuesta a una solicitud previamente aceptada a través de una actividad Receive.
- *Trow*: Usada cuando el proceso necesita una falla explícitamente.
- *Invoke*: Usada para la invocación de una operación de servicios Web ofrecida por un participante.
- *Link*: Define un enlace de un flujo; Una actividad dentro del flujo puede actuar como la fuente de un Link ó el destino de un Link.
- *Flow*: Provee concurrencia y sincronización.
- *While*: Soporta la ejecución repetida de una actividad específica; la ejecución continua hasta que condición booleana especificada no sea verdad.
- *Sequence*: Incluye una ó más actividades a ser ejecutadas secuencialmente, en el orden en el cual aparecen dentro de esta actividad.
- *Pick*: Espera la aparición de uno o más eventos y ejecuta la actividad asociada con dicho evento o eventos. Mensajes siendo recibidos ó valores de tiempo trascurrido forman los posibles eventos.
- *Switch*: Soporta comportamiento condicionado al especificar una ó más ramas que se ejecutan dependiendo de una condición específica.

Capítulo 3

ebXML BPSS

En el presente capítulo se establece el contexto necesario para ebXML BPSS.

3.1 Introducción a ebXML BPSS

El objetivo principal de *ebXML* es facilitar la integración de negocios electrónicos y para esto el enfoque principal del trabajo ha sido en la noción de procesos públicos, los procesos a través de los cuales los participantes interactúan unos a otros (11).

El estándar *ebXML BPSS* define un lenguaje mediante el cual sistemas de negocios pueden ser configurados para soportar colaboraciones de negocios, consistentes de transacciones de negocios (1). El termino *ebBP* es, y lo será en el presente trabajo, utilizado para abreviar *ebXML BPSS*. Una definición, de proceso de negocios, creada con la especificación *ebBP* es referida como una definición *ebBP*.

La especificación técnica *ebBP* soporta la definición de transacciones de negocios y la coreografía de transacciones de negocios dentro de colaboraciones de negocios. Todas las transacciones son implementadas usando uno de muchos patrones disponibles. Un patrón no es ejecutable, el detalla el tipo de intercambio de mensajes (solicitud, respuesta y señales) que aplica para una especificación dada de una transacción de negocios. Estos patrones pueden, en potencia, ser relacionados a diferentes clases de transacciones de comercio electrónico.

Las definiciones *ebBP* describen procesos de negocios interoperables que permiten a compañeros de negocios colaborar para alcanzar una meta de negocios dada. Estas definiciones deben ser ejecutadas por componentes de software que colaboran en nombre de los compañeros de negocios. El objetivo de la especificación *ebBP* es proveer el puente entre el modelado de procesos de comercio electrónico y la ejecución de componentes de software de comercio electrónico. ebBP provee el conjunto de elementos necesarios, para especificar una colaboración de negocios entre compañeros, y

para suministrar parámetros de configuración de los sistemas de los compañeros, con el objetivo de ejecutar colaboraciones de negocio entre un conjunto de componentes de software (1).

3.1.1 ebXML BPSS y Otras Tecnologías

ebBP forma parte de un familia *ebXML* que ha sido compuesta de varios componentes, independientes, pero relacionados o alineados. Especificaciones para cada componente pueden ser usadas individualmente, compuestas según se desee, o integradas con otras tecnologías emergentes. De allí que *ebBP* pueda ser usada efectivamente con otras tecnologías.

Desde el inicio, los creadores de las especificaciones han buscado que estén alineadas tanto como que sean prácticas y capaces de ser compuestas juntas y usadas con otras tecnologías. Esa flexibilidad y capacidad de ser compuestas son aspectos importantes, no sólo para la adopción de esos estándares pero para su efectivo uso y exitoso despliegue dentro de ambientes heterogéneos y entre dominios. En el contexto de *ebBP*, colaboraciones de negocios pueden ser ejecutadas usando la definición de procesos *ebXML* y/ó usadas con otras tecnologías. En cuanto a su relación con las otras tecnologías en la familia *ebXML*, una definición de procesos *ebBP* soporta el acoplamiento impreciso y el alineamiento necesarios para ejecutar colaboraciones de negocios. Esta especificación también puede ser empleada cuando muchos otros componentes de software son usados para permitir la ejecución de colaboraciones de negocio. La especificación *ebBP* es usada para especificar los parámetros de configuración, relacionados con los procesos de negocios, que emplea una BSI(Business Service Interface) para ejecutar y monitorear estas colaboraciones. La semántica de negocios y sintaxis de ebBP también son bien adecuadas para permitir la definición de bloques modulares de procesos, que son combinados en actividades complejas para suplir las necesidades de la comunidad de usuarios (1).

3.1.2 Uso de ebXML BPSS con Otras Especificaciones

La especificación técnica *ebBP* provee la estructura y semántica de las definiciones de colaboraciones de negocios. Una colaboración consiste de un conjunto de roles que colaboran al intercambiar documentos de negocios mediante un conjunto de transacciones coreografiadas. Como se muestra en la figura 4, documentos de negocios son definidos en la intersección entre *ebBP* y la especificación de componentes base *ebXML* (12). Una definición *ebBP* referenciará, pero no definirá, un conjunto de documentos de negocios lógicos. Dentro de una definición ebBP, documentos de negocios son ó definidos por alguna especificación externa de documentos, ó ensamblados de estructuras de bajo nivel de información, llamadas componentes base. La combinación de la especificación del proceso de negocios y la

especificación del documento, se vuelven la base contra la cual compañeros de negocios pueden hacer acuerdos, sobre la conducción de negocios electrónicos con sus correspondientes (1).

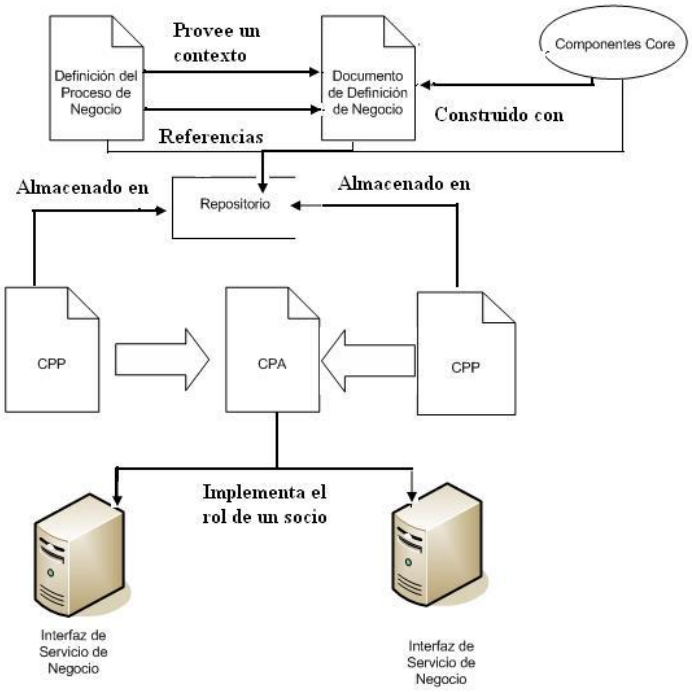

Figura 4 : Definición ebBP y otros miembros de la familia ebXML.[Tomada de (13)]

El usuario extraerá y transformará la información necesaria, creando con ella una representación *XML* de una *ebBP* definición. Esta representación *XML* se almacena y registra en el repositorio *ebXML* y el registro *ebXML*, correspondientemente, para su futura obtención.

Cuando implementadores quieren establecer un pareja de negociación *CPA*[15] (14), el documento de definición *ebBP*, o partes relevantes de este, son simplemente referenciadas por el *CPP*[16] (14) y usadas en los documentos XML CPA. *ebXML CPP* and documentos *XML CPA* pueden referenciar especificaciones de procesos de negocio en *XML* tal como una definición *ebBP* (1).

Una definición ebXML es, junto con especificaciones de protocolo, el objeto de acuerdo entre dos o más participantes. La definición *ebBP* puede, por lo tanto, ser incorporada con ó referenciada por

[15] Collaboration Protocol Agreement
[16] Collaboration Protocol Profile

parejas de negociación *CPP* ó *CPA*. El *CPA* articula el mecanismo técnico que configura un sistema de ejecución y estimula interoperabilidad entre dos participantes que pueden usar diferentes aplicaciones o software de diferentes vendedores. Cada *CPP* puede declarar su soporte para uno o más roles dentro de cada ebBP definición.

Guiado por las especificaciones *CPP* y *CPA*, el documento *XML* resultante luego puede convertirse en el archivo de configuración para una o más *BSI*[17] (13), es decir el componente de software que puede manejar cualquier participación de un compañero de negocios en una colaboración de negocios (1). La especificación técnica ebBP no especifica como una BSI es implementada (13).

3.2 Sobre los Conceptos Técnicos de ebXML BPSS

Para entender los conceptos técnicos específicos de *ebXML BPSS*, su sintaxis y una visión detallada de las relaciones entre ellos, así como de su aplicación, se hace necesaria la consulta del esquema *ebXML BPSS* para procesos de negocios(anexo A.2) y de la especificación técnica *ebXML BPSS* (2)[18]. A continuación se sintetizan sólo las ideas esenciales más importantes:

- *Business Transaction*: Flujo de documentos de negocios entre participantes que consultan y responden mediante roles. Es un protocolo especializado empleado para obtener y soportar semántica de transacciones y establecer el alineamiento entre participantes que colaboran.
- *Business Signal*: Una ó más señales pueden ser intercambiadas como parte de una transacción de negocios para asegurar el alineamiento entre ambos participantes.
- *Business Collaboration*: Es un conjunto de roles que interactúan a través de un grupo de transacciones de negocios, según una coreografía y mediante el intercambio de documentos de negocios.

[17] Business Service Interface
[18] No incluida por tener más de 100 páginas

Capítulo 4

Empleando WSBPEL

El presente capítulo se enfoca en el empleo de *WSBPEL*, por lo que se establecen las consideraciones que se necesitan para esto, se plasma un procedimiento desarrollado para asistir en la especificación de procesos mediante el estándar, se usa lo planteado en un caso dado y se concluye con la presentación de una herramienta de software, existente en el mercado.

4.1 Consideraciones para Emplear WSBPEL

La orientación principal hacia los procesos ejecutables que tiene *WSBPEL*, hace necesario un buen entendimiento de cómo se describen estos con *WSBPEL* antes de tratar de describir procesos abstractos. Describir los procesos abstractos es tratar de emplear todo el potencial para ejecutables de manera reducida y omitiendo u ocultando buena parte de ellos. Es así como la gestión de datos y el manejo de los detalles operativos no es importante al emplear *WSBPEL* para la descripción de procesos abstractos. Por naturaleza las descripciones *WSBPEL* de procesos abstractos son informales y la razón es que no pretenden ser ejecutadas sino sólo descriptivas.

Del enfoque hacia la interoperabilidad que tiene el presente trabajo se puede establecer fácilmente el tipo de procesos abstractos que se desean describir. Sin embargo, se debe destacar que al emplear *WSBPEL* para procesos abstractos, es obligatorio establecer este contexto mediante la inclusión del *Profile* que será empleado. Un *Profile* puede acotar la base común de procesos públicos y da una semántica basada en la misma empleado para los procesos ejecutables. En el presente trabajo se ha empleado el *Profile* de comportamiento observable incluido con la especificación, que sin lugar duda se ajusta a los procesos abstractos que se dan entre participantes que interoperan.

Aún cuando la especificación establece que no es necesario un modelaje de negocios previo, sólo lo sugiere, se considera que aún de manera informal es necesario esquematizar el entorno en el que

se desea emplear *WSBPEL*. En cualquier caso un buen conocimiento del proceso que se desea analizar es indispensable.

Para especificar el proceso *WSBPEL* se tendrá inicialmente un conjunto de descripciones *WSDL* que han sido construidas previamente también en base al conocimiento del proceso en estudio. Con referencia a estás descripciones se identifica buena parte de los datos necesarios para el procedimiento presentado a continuación.

4.2 Un Procedimiento para Emplear WSBPEL

La siguiente ilustración representa el procedimiento establecido para la definición de un proceso abstracto (público) de negocios en *WSBPEL*. Se ha tomado de algunas trabajos en otras áreas (15) (16), ideas sobre lo que es un método, la necesidad de un producto final, en este caso la representación *XML* de la definición *WSBPEL* del proceso abstracto de negocios, la definición de etapas y sus salidas, pero la formalidad, nivel de detalle y generalización han sido aspectos secundarios, poco considerados. Esto último debido a que lo plasmado es un procedimiento y no un método, y que adicional, este resultado es un medio para el objetivo principal que es comparar *WSBPEL* y *ebXML BPSS*, siendo la investigación necesaria para crearlo el aspecto más importante, por la información que ha aportado. De igual forma su uso y su funcionalidad se ha enfocado en captar los conceptos fundamentales de cada estándar. La idea del procedimiento es ser una guía complementaría para alguien que maneja los conceptos técnicos de *WSBPEL* referenciados en 2.2. Luego de la ilustración se agregan algunas claves importantes por cada etapa.

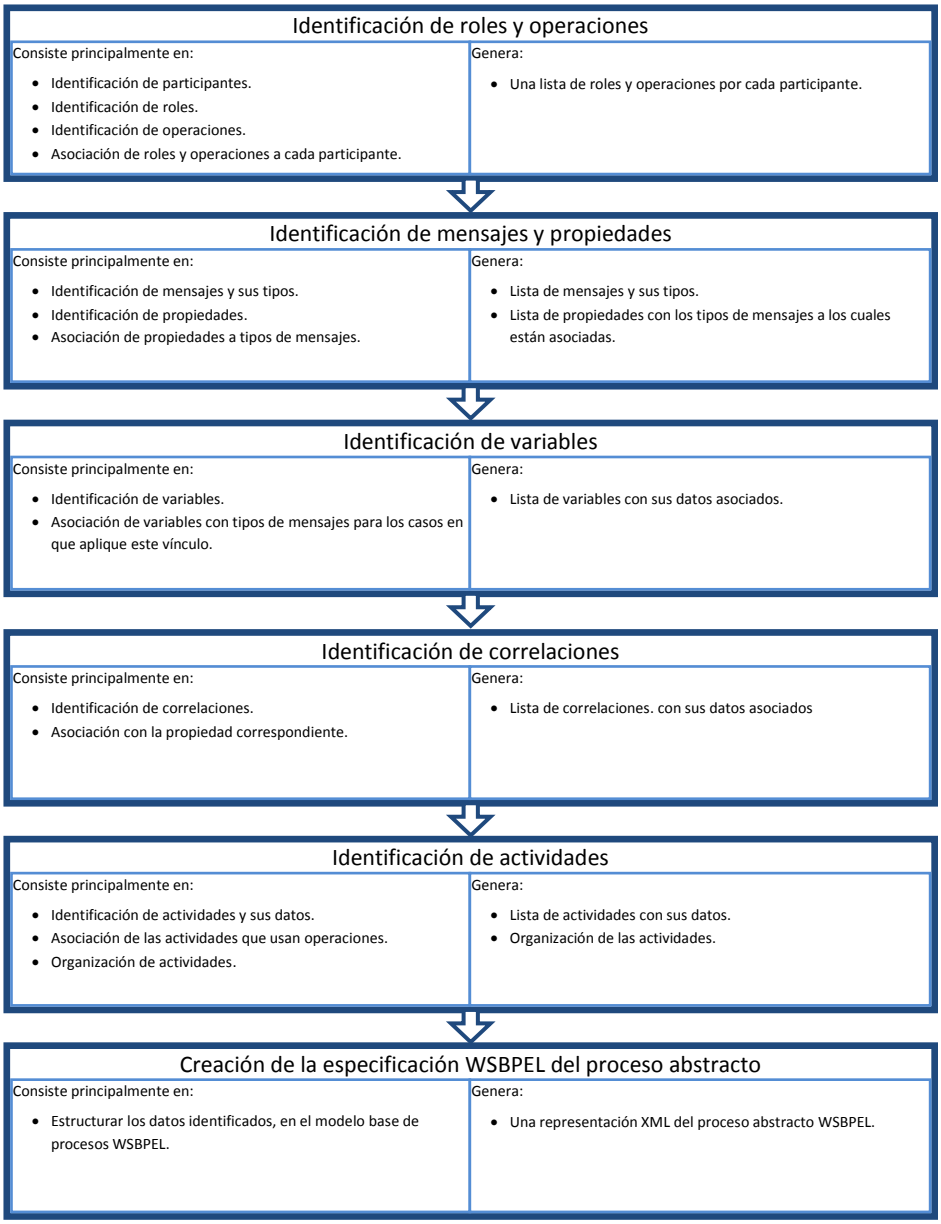

Figura 5 Un procedimiento para emplear WSBPEL.

4.2.1 Identificación de Roles y Operaciones

En esta etapa en esencia identificamos quienes participan y bajo que rol lo hacen. De igual forma se identifican las operaciones asociadas a cada rol. La información debe ser inicialmente plasmada en las definiciones *WSDL* necesarias. Tendremos entonces de la definición de *partnerLinkTypes* algo como:

```
<plnk:partnerLinkType name="CompradoVendedorEnlace">
<plnk:role name="Comprador" portType="comprar:CompradorPortType" />
<plnk:role name="Vendedor" portType="vender:VendedorPortType" />
</plnk:partnerLinkType>
```

Nuestros roles serán entonces "comprador" y "vendedor".

En la definición de *PortTypes* tendremos algo como:

```
<wsdl:portType name="CompradorPortType">
- <wsdl:operation name="enviarPeticion">
<wsdl:input message=" peticionMsg" />
</wsdl:operation>
```

Asociaciando con la definición anterior tendremos que el rol "vendedor" tiene la operación "enviarPeticion" pues están relacionado por el *portType*.

4.2.2 Identificación de Mensajes y Propiedades

En esta etapa identificamos mensajes y propiedades que luego referenciaremos al construir la especificación del proceso. En las definiciones *WSDL* tendremos algo como:

```
<wsdl:message name="PeticionMsg">
<wsdl:part name="partePeticion" type="comprar:peticionTipo" />
</wsdl:message>
```

De acá el nombre dado al mensaje "PeticiónMSg" conformara nuestra lista de mensajes que luego referenciaremos. En las definiciones también encontraremos:

```
<vprop:property name="peticionId" type=" xsd:int" />
<vprop:propertyAlias propertyName="peticionId" messageType="PeticionMsg" part="partePeticion">
<vprop:query>comprar:PeticionMsgHeader/comprar:peticionID</vprop:query>
</vprop:propertyAlias>
```

De acá tendremos la propiedad "peticionId"asociada al mensaje "PeticionMsg".

Así conformamos nuestra lista de mensajes y propiedades asociadas con sus tipos.

4.2.3 Identificación de Variables

Del proceso identificamos las variables cuya valor debemos mantener durante el proceso para toma de decisiones posteriores. El caso más común es asociar una variable a un mensaje que necesitamos retener y luego referenciar.

4.2.4 Identificación de Correlaciones

Las correlaciones vienen dadas del proceso de negocios al identificar la necesidad de asociar más que las parejas que están negociando, las instancias particulares que están manejando un determinado proceso. Si se recibe una petición, quien la envía podría estar manejando varias peticiones a la vez por lo que al responder referenciamos exactamente la petición con un dato como su peticionId.

4.2.5 Identificación de Actividades

Las actividades se identifican por observación del proceso de negocios, siendo por ejemplo actividades de solicitar o responder. Adicionalmente se deben organizar de acuerdo al proceso, es decir, describir el orden o relación en que se dan.

4.2.6 Creación de la Especificación del Proceso Abstracto

En este punto la idea es plasmar en la base *XML* mostrada a continuación, los datos identificados anteriormente, que se complementarán, como ya se estableció, con el manejo de los conceptos técnicos que se deben tener de *WSBPEL*. La base *XML* es:

```
<process name="" targetNamespace="" abstractProcessProfile="http://docs.oasis-open.org/wsbpel/2.0/process/abstract/ap11/2006/08" xmlns="http://docs.oasis-open.org/wsbpel/2.0/process/abstract">
<import namespace="" location="" importType="" />
- <partnerLinks>
<partnerLink name="" partnerLinkType="" myRole="" partnerRole="" />
</partnerLinks>
- <variables>
<variable name="" messageType="" type="" />
</variables>
- <correlationSets>
<correlationSet name="" properties="" />
</correlationSets>
Activity
</process>
```

4.3 Uso de WSBPEL Mediante el Procedimiento Planteado

El escenario al que se aplica el procedimiento es el de una relación entre un compañía cliente y un servicio de envío de productos, donde la compañía cliente puede escoger entre que los productos sean

enviados todos juntos ó por grupos hasta completar el total. Las descripciones wsdl que son parte del escenario se muestran a continuación y luego de ellas la descripción WSBPEL del proceso.

```
<wsdl:definitions targetNamespace="http://fuente.com/envios/partnerLinkTypes/" xmlns:plnk="http://docs.oasis-
    open.org/wsbpel/2.0/plnktype" xmlns:sif="http://fuente.com/envios/interfaces/"
    xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
 <wsdl:import location="enviosPT.wsdl" namespace="http://fuente.com/envios/interfaces/" />
- <plnk:partnerLinkType name="enviosLT">
 <plnk:role name="enviosServicio" portType="sif:enviosServicioPT" />
 <plnk:role name="enviosServicioCliente" portType="sif:enviosServicioClientePT" />
    </plnk:partnerLinkType>
    </wsdl:definitions>

- <wsdl:definitions targetNamespace="http://fuente.com/envios/interfaces/"
    xmlns:envio="http://fuente.com/envios/envio.xsd" xmlns:tns="http://fuente.com/envios/interfaces/"
    xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
- <wsdl:types>
- <xsd:schema>
 <xsd:import namespace="http://fuente.com/envios/envio" schemaLocation="http://fuente.com/envios/envio.xsd" />
    </xsd:schema>
    </wsdl:types>
- <wsdl:message name="enviosSolicitudMsg">
 <wsdl:part name="envioOrden" type="envio:envioOrden" />
    </wsdl:message>
- <wsdl:message name="enviosNotaMsg">
 <wsdl:part name="envioNota" type="envio:envioNota" />
    </wsdl:message>
- <wsdl:portType name="enviosServicioPT">
- <wsdl:operation name="enviosSolicitud">
 <wsdl:input message="tns:enviosSolicitudMsg" />
    </wsdl:operation>
    </wsdl:portType>
- <wsdl:portType name="enviosServicioClientePT">
- <wsdl:operation name="enviosNota">
 <wsdl:input message="tns:enviosNotaMsg" />
    </wsdl:operation>
    </wsdl:portType>
    </wsdl:definitions>

<wsdl:definitions targetNamespace="http://fuente.com/envios/properties/" xmlns:bpel="http://docs.oasis-
    open.org/wsbpel/2.0/process/executable" xmlns:vprop="http://docs.oasis-open.org/wsbpel/2.0/varprop"
```

```
        xmlns:envio="http://fuente.com/envios/envio.xsd"          xmlns:sif="http://fuente.com/envios/interfaces/"
        xmlns:tns="http://fuente.com/envios/properties/"          xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
        xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <wsdl:import location="enviosPT.wsdl" namespace="http://fuente.com/envios/interfaces/" />
- <wsdl:types>
- <xsd:schema targetNamespace="http://fuente.com/envios/envio.xsd">
- <xsd:simpleType name="contadorType">
- <xsd:restriction base="xsd:int">
  <xsd:minInclusive value="1" />
  <xsd:maxInclusive value="40" />
     </xsd:restriction>
     </xsd:simpleType>
     </xsd:schema>
     </wsdl:types>
  <vprop:property name="envioOrdenID" type="xsd:int" />
  <vprop:property name="envioCompleto" type="xsd:boolean" />
  <vprop:property name="total" type="envio:contadorType" />
  <vprop:property name="contador" type="envio:contadorType" />
- <vprop:propertyAlias propertyName="tns:envioOrdenID" messageType="sif:enviosSolicitudMsg" part="envioOrden">
  <vprop:query>envio:envioOrdenSolicitudHeader/envio:envioOrdenID</vprop:query>
     </vprop:propertyAlias>
- <vprop:propertyAlias propertyName="tns:envioOrdenID" messageType="sif:enviosNotaMsg" part="envioNota">
  <vprop:query>envio:envioNotaHeader/envio:envioOrdenID</vprop:query>
     </vprop:propertyAlias>
- <vprop:propertyAlias propertyName="tns:envioCompleto" messageType="sif:enviosSolicitudMsg" part="envioOrden">
  <vprop:query>envio:envioOrdenSolicitudHeader/envio:envioCompleto</vprop:query>
     </vprop:propertyAlias>
- <vprop:propertyAlias propertyName="tns:total" messageType="sif:enviosSolicitudMsg" part="envioOrden">
  <vprop:query>envio:envioOrdenSolicitudHeader/envio:total</vprop:query>
     </vprop:propertyAlias>
- <vprop:propertyAlias propertyName="tns:contador" messageType="sif:enviosSolicitudMsg" part="envioOrden">
  <vprop:query>envio:envioOrdenSolicitudHeader/envio:contador</vprop:query>
     </vprop:propertyAlias>
- <vprop:propertyAlias propertyName="tns:contador" messageType="sif:enviosNotaMsg" part="envioNota">
  <vprop:query>envio:envioNotaHeader/envio:contador</vprop:query>
     </vprop:propertyAlias>
     </wsdl:definitions>

<process name="ServicioEnviosProductos" targetNamespace="http://fuente.com/envios/" xmlns="http://docs.oasis-
        open.org/wsbpel/2.0/process/abstract"          xmlns:plt="http://fuente.com/envios/partnerLinkTypes/"
        xmlns:props="http://fuente.com/envios/properties/"      xmlns:envio="http://fuente.com/envios/envio.xsd"
```

```
        xmlns:sif="http://fuente.com/envios/interfaces/"                abstractProcessProfile="http://docs.oasis-
        open.org/wsbpel/2.0/process/abstract/ap11/2006/08">
              <import              importType="http://schemas.xmlsoap.org/wsdl/"              location="enviosLT.wsdl"
        namespace="http://fuente.com/envios/partnerLinkTypes/" />
              <import              importType="http://schemas.xmlsoap.org/wsdl/"              location="enviosPT.wsdl"
        namespace="http://fuente.com/envios/interfaces/" />
            <import            importType="http://schemas.xmlsoap.org/wsdl/"            location="enviosProperties.wsdl"
        namespace="http://fuente.com/envios/properties/" />
- <partnerLinks>
          <partnerLink        name="cliente"        partnerLinkType="plt:enviosLT"        partnerRole="enviosServicioCliente"
        myRole="enviosServicio" />
      </partnerLinks>
- <variables>
  <variable name="envioSolicitud" messageType="sif:enviosSolicitudMsg" />
  <variable name="envioNota" messageType="sif:enviosNotaMsg" />
  <variable name="enviados" type="envio:contadorType" />
      </variables>
- <correlationSets>
  <correlationSet name="envioOrden" properties="props:envioOrdenID" />
      </correlationSets>
- <sequence>
- <receive partnerLink="cliente" operation="enviosSolicitud" variable="envioSolicitud">
- <correlations>
  <correlation set="envioOrden" initiate="yes" />
      </correlations>
      </receive>
- <if>
  <condition>bpel:getVariableProperty('envioSolicitud', 'props:envioCompleto')</condition>
- <sequence>
- <assign>
- <copy>
  <from variable="envioSolicitud" property="props:envioOrdenID" />
  <to variable="envioNota" property="props:envioOrdenID" />
      </copy>
- <copy>
  <from variable="envioSolicitud" property="props:contador" />
  <to variable="envioNota" property="props:contador" />
      </copy>
      </assign>
- <invoke partnerLink="cliente" operation="enviosNota" inputVariable="envioNota">
- <correlations>
  <correlation set="envioOrden" pattern="Solicitud" />
```

```
      </correlations>
      </invoke>
      </sequence>
- <else>
- <sequence>
- <assign>
- <copy>
  <from>0</from>
  <to>$enviados</to>
      </copy>
      </assign>
- <while>
  <condition>$enviados < bpel:getVariableProperty('envioSolicitud', 'props:total')</condition>
- <sequence>
- <assign>
- <copy>
  <opaqueFrom />
  <to variable="envioNota" property="props:envioOrdenID" />
      </copy>
- <copy>
  <opaqueFrom />
  <to variable="envioNota" property="props:contador" />
      </copy>
      </assign>
- <invoke partnerLink="cliente" operation="enviosNota" inputVariable="envioNota">
- <correlations>
  <correlation set="envioOrden" pattern="Solicitud" />
      </correlations>
      </invoke>
- <assign>
- <copy>
  <from>$enviados + bpel:getVariableProperty('envioNota', 'props:contador')</from>
  <to>$enviados</to>
      </copy>
      </assign>
      </sequence>
      </while>
      </sequence>
      </else>
      </if>
      </sequence>
      </process>
```

4.4 Una Herramienta de Software para WSBPEL

En la figura 6 se muestra la pantalla principal de ActiveBPEL, mostrando un ejemplo para ilustrar las funcionalidades que se presentan a simple vista. ActiveBPEL es una familia de productos construidos para soportar la creación, prueba, despliegue y ejecución de procesos *WSBPEL*. ActiveBPEL ofrece un entorno visual que oculta los detalles de código ó notación, permitiendo entonces enfocarse en aspectos de más alto nivel, de manera que aunque se aprecia una representación visual, se tiene por debajo una definición formal y 100% *WSBPEL*. Es un entorno amigable, bien desarrollado, estable, con una usabilidad muy buena y con una funcionalidad para asistir el trabajo con WSBPEL en cualquier etapa.

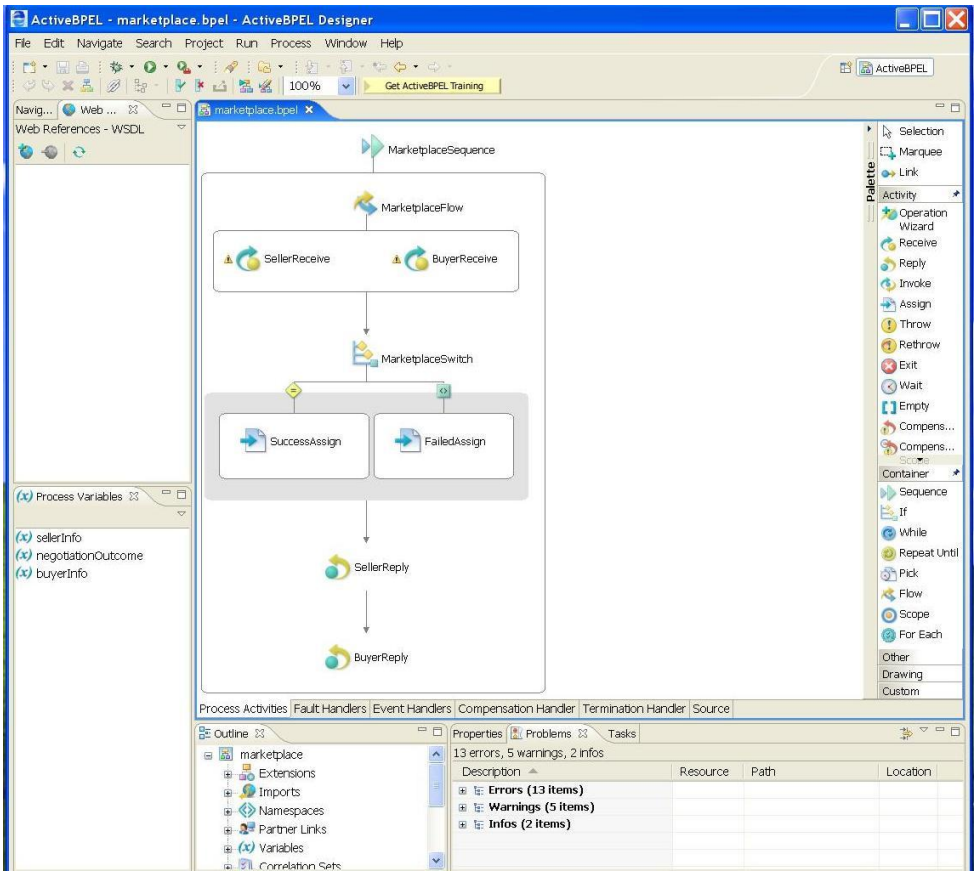

Figura 6 Pantalla principal de una herramienta para WSBPEL.

Capítulo 5

Empleando ebXML BPSS

El presente capítulo se enfoca en el empleo de *ebXML BPSS*, por lo que se establecen las consideraciones que se necesitan para esto, se plasma un procedimiento desarrollado para asistir en la especificación de procesos mediante el estándar, se usa lo planteado en un caso dado y se concluye con la presentación de una herramienta de software, existente en el mercado.

5.1 Consideraciones para Emplear ebXML BPSS

El escenario ideal para emplear *ebXML* es aquel donde se conoce el resto de los elementos de la familia *ebXML* (*CPP, CPA, BSI*, etc.) y se emplean de manera conjunta. Como mínimo se debe tener una idea de estos conceptos y manejar el contexto en el que se encuentra *ebXML BPSS*.

Aún cuando la especificación establece que no es necesario un modelaje de negocios previo, sólo lo sugiere, se considera que aún de manera informal es necesario esquematizar el entorno en el que se desea emplear ebXML BPSS. En cualquier caso un buen conocimiento del proceso que se desea analizar es indispensable.

5.2 Un Procedimiento para Emplear ebXML BPSS

La siguiente ilustración representa el procedimiento establecido para la definición de un proceso de negocios en *ebXML BPSS*. Aplican las mismas consideraciones sobre procedimientos y métodos, plasmadas en 4.2. La idea del procedimiento es ser una guía complementaría para alguien que maneja los conceptos técnicos de *ebXML BPSS* referenciados en 3.2. Luego de la ilustración se agregan algunas claves importantes por cada etapa.

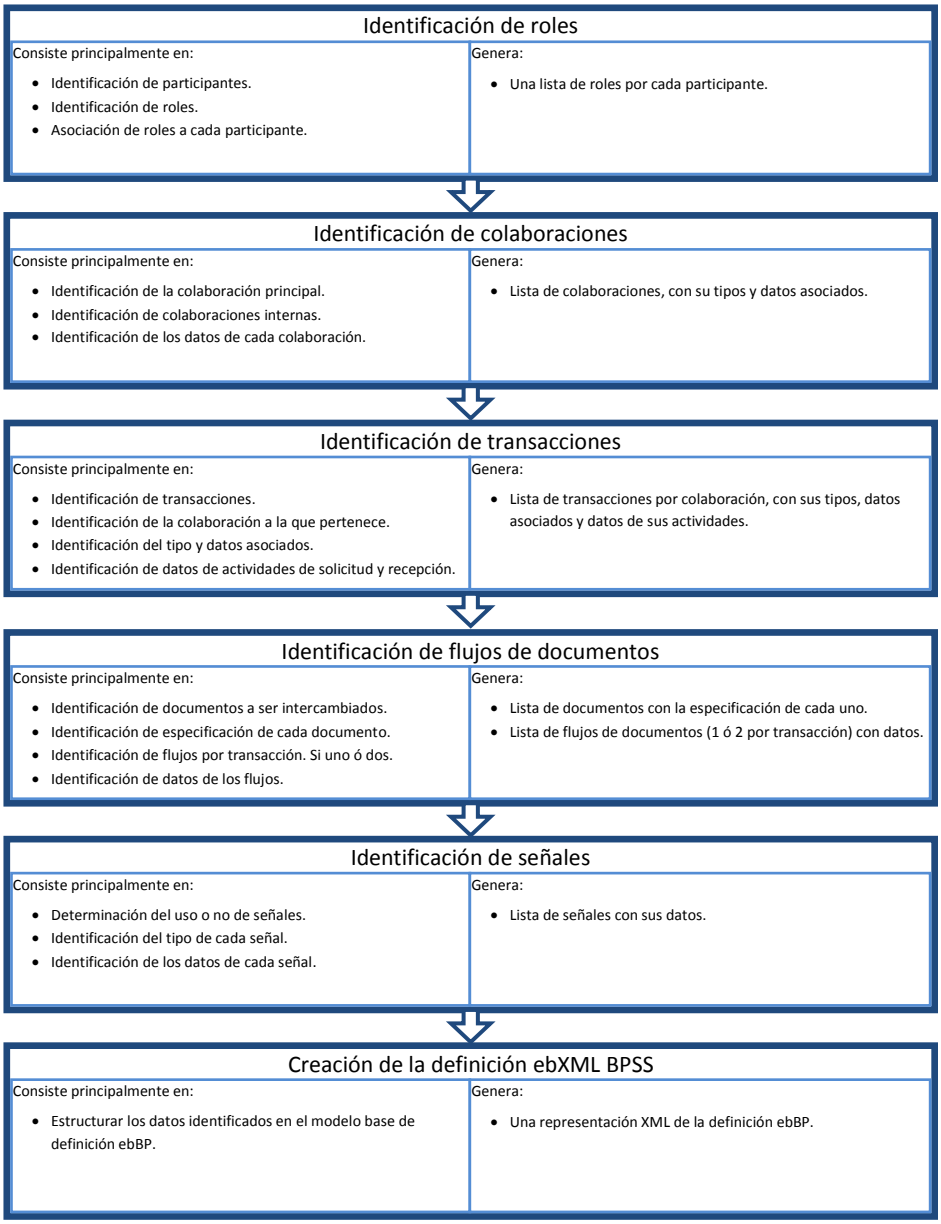

Figura 7 Un procedimiento para emplear ebXML BPSS

5.2.1 Identificación de Roles

En esta etapa en esencia identificamos quienes participan y bajo que rol lo hacen. Un punto importante a tener en cuenta es que estos roles deben estar siempre presenten al establecer los roles internos que

tiene una transacción a una colaboración anidada dentro de la principal. Es decir que siempre habrá un mapeo de estos roles principales a cuales quieran que pueda existir internamente.

5.2.2 Identificación de Colaboraciones

La colaboración principal es la línea de referencia, dentro de la cual se mueven los demás elementos utilizados para la descripción. Es la interacción, de más alto nivel que se realiza entre los participantes. Pueden existir otras colaboraciones pero siempre dentro de una colaboración principal. Las colaboraciones están compuestas a su vez por transacciones u otras colaboraciones. Una clave a este nivel es entender que los elementos denominados actividades, no son sino una especie de envoltorios de las transacciones o colaboraciones.

5.2.3 Identificación de Transacciones

Las transacciones obedecen a patrones preestablecidos que permiten definirlas con mayor facilidad. El tipo de transacción o patrón dependerá de la interacción que sea desea describir. Un punto importante acá es que una transacción siempre contempla una o dos actividades que son envoltorios de intercambios de documentos. El uso de una o dos de estas actividades dependerá del patrón de la transacción.

5.2.4 Identificación de Flujos de Documentos

El intercambio de documentos es el fundamento de *ebXML BPSS*. Debe considerarse como u hecho que en cualquier interacción siempre estará envuelto al menos un flujo de documentos. De cada uno de estos flujos debe surgir la lista de documentos que serán referenciados. Cada documento se específica de acuerdo a un esquema externo que tan sólo debe ser previamente acordado.

5.2.5 Identificación De Señales

Las señales son opcionales en las interacciones y se emplean cuando se necesita un nivel mayor de monitoreo de los intercambios realizados. Los tipos de señales son definidos de acuerdo a un esquema específico para ellos. Esté esquema es incluido en con el estándar y puede ser modificado o extendido.

5.2.6 Creación de la Definición ebXML BPSS

En este punto la idea es plasmar en la base *XML* mostrada a continuación, los datos identificados anteriormente, que se complementarán, como ya se estableció, con el manejo de los conceptos técnicos que se deben tener de *ebXML BPSS*. La base *XML* es:

```
<?xml version="1.0" encoding="UTF-8" ?>
- <ProcessSpecification xmlns="http://docs.oasis-open.org/ebxml-bp/ebbp-2.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xi="http://www.w3.org/2001/XInclude"
xsi:schemaLocation="http://docs.oasis-open.org/ebxml-bp/ebbp-2.0 ebbp-2.0.4.xsd" name="" nameID="" uuid=""
specificationVersion="2.0">
- <Signal name="" nameID="">
<Specification name="" nameID="" location="" />
</Signal>
- <BusinessDocument name="" nameID="">
<Specification type="" location="" name="" nameID="" />
</BusinessDocument>
- <CommercialTransaction_o_RequestResponse_o_Notification_o_otra name="" nameID="">
<RequestingRole name="" nameID="" />
<RespondingRole name="" nameID="" />
- <RequestingBusinessActivity name="" nameID="" isAuthorizationRequired="" isIntelligibleCheckRequired=""
isNonRepudiationRequired="" isNonRepudiationReceiptRequired="">
<DocumentEnvelope name="" nameID="" businessDocumentRef="" />
<ReceiptAcknowledgement name="" nameID="" signalDefinitionRef="" />
<ReceiptAcknowledgementException name="" nameID="" signalDefinitionRef="" />
</RequestingBusinessActivity>
- <RespondingBusinessActivity name="" nameID="" isAuthorizationRequired="" isIntelligibleCheckRequired=""
isNonRepudiationRequired="" isNonRepudiationReceiptRequired="">
<DocumentEnvelope name="" nameID="" businessDocumentRef="" />
<ReceiptAcknowledgement name="" nameID="" signalDefinitionRef="" />
<ReceiptAcknowledgementException name="" nameID="" signalDefinitionRef="" />
</RespondingBusinessActivity>
</CommercialTransaction_o_RequestResponse_o_Notification_o_otra>
- <BusinessCollaboration name="" nameID="" isInnerCollaboration="false">
<Role name="" nameID="" />
<Role name="" nameID="" />
<TimeToPerform type="" />
- <Start>
<ToLink toBusinessStateRef="" />
</Start>
- <BusinessTransactionActivity_o_collaborationActivity name="" nameID="" businessTransactionRef="" hasLegalIntent="">
<TimeToPerform type="" />
```

```
<Performs currentRoleRef="" performsRoleRef="" />
<Performs currentRoleRef="" performsRoleRef="" />
<EndsWhen expressionLanguage="ConditionGuardValue" expression="" />
</BusinessTransactionActivity_o_collaborationActivity>
<Transition_o_Fork_o_Decision_o_Join_o_otro />
<Success name="" nameID="" />
<Failure name="" nameID="" />
</BusinessCollaboration>
</ProcessSpecification>
```

5.3 Uso de ebXML BPSS Mediante el Procedimiento Planteado

El escenario al que se aplica el procedimiento es el de una sub-contratista y la compañía que le asigna ciertas ordenes de productos. Se realiza la orden, luego el despacho de insumos, posteriormente se genera el estado de la orden por petición de la compañía ó se generan un reporte periódico del estado de la orden, finalmente se genera un recibo. A continuación la descripción *ebXML BPSS* del proceso:

```
<?xml version="1.0" encoding="UTF-8" ?>
- <ProcessSpecification xmlns="http://docs.oasis-open.org/ebxml-bp/ebbp-2.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xi="http://www.w3.org/2001/XInclude"
  xsi:schemaLocation="http://docs.oasis-open.org/ebxml-bp/ebbp-2.0 ebbp-2.0.4.xsd" name="Ejemplo Sub-
  Contratista" nameID="SCE-2007-1" uuid="subcontratistaejemplo-2007-1" specificationVersion="2.0">
  - <!-- Señales -->
- <Signal name="ra" nameID="ra1">
  <Specification name="ReceiptAcknowledgement" nameID="rabpss1" location="ebbp-signals-2.0.4.xsd" />
  </Signal>
- <Signal name="rae" nameID="rae1">
  <Specification name="ReceiptAcknowledgementException" nameID="raebpss1" location="ebbp-signals-2.0.4.xsd" />
  </Signal>
  - <!-- Documentos de negocios -->
- <BusinessDocument name="Orden" nameID="bd-orden-1">
  <Specification type="schema" location="http://www.fuente.org/schema/2005-1/orden.xsd" name="Schema Orden"
  nameID="schema-orden-2" />
  </BusinessDocument>
- <BusinessDocument name="Orden Respuesta" nameID="bd-ordenrespuesta-3">
  <Specification type="schema" location="http://www.fuente.org/schema/2005-1/ordenrespuesta.xsd" name="Schema
  Orden Respuesta" nameID="schema-ordenrespuesta-4" />
```

```
    </BusinessDocument>
- <BusinessDocument name="Despacho Insumos" nameID="bd-despachoinsumos-21">
  <Specification type="schema" location="http://www.fuente.org/schema/2005-1/despachoinsumos.xsd" name="Schema
      Despacho Insumos" nameID="schema-despachoinsumos-22" />
    </BusinessDocument>
- <BusinessDocument name="Recibo Insumos" nameID="bd-reciboinsumos-23">
  <Specification type="schema" location="http://www.fuente.org/schema/2005-1/reciboinsumos.xsd" name="Schema
      Recibo Insumos" nameID="schema-reciboinsumos-24" />
    </BusinessDocument>
- <BusinessDocument name="Orden Status Solicitud" nameID="bd-ordenstatussolicitud-50">
    <Specification type="schema" location="http://www.fuente.org/schema/2005-1/ordenstatussolicitud.xsd"
      name="Schema Orden Status Solicitud" nameID="schema-ordenstatussolicitud-51" />
    </BusinessDocument>
- <BusinessDocument name="Orden Status Reporte" nameID="bd-ordenstatusreporte-52">
    <Specification type="schema" location="http://www.fuente.org/schema/2005-1/ordenstatusreporte.xsd"
      name="Schema Orden Status Reporte" nameID="schema-ordenstatusreporte-53" />
    </BusinessDocument>
- <BusinessDocument name="Recibo" nameID="bd-recibo-604">
  <Specification type="schema" location="http://www.fuente.org/schema/2005-1/recibo.xsd" name="Schema Recibo"
      nameID="schema-recibo-605" />
    </BusinessDocument>
- <BusinessDocument name="ContraRecibo" nameID="bd-contrarecibo-606">
  <Specification type="schema" location="http://www.fuente.org/schema/2005-1/contrarecibo.xsd" name="Schema
      ContraRecibo" nameID="schema-contrarecibo-607" />
    </BusinessDocument>
  - <!-- Transacciones de negocios -->
  - <!-- Orden -->
- <CommercialTransaction name="BT Orden" nameID="bt-orden-5">
  <RequestingRole name="Requester" nameID="requester-bt-orden-5" />
  <RespondingRole name="responder" nameID="responder-bt-orden-5" />
-   <RequestingBusinessActivity name="Req Orden" nameID="req-bt-order-6" isAuthorizationRequired="false"
      isIntelligibleCheckRequired="false" isNonRepudiationRequired="false" isNonRepudiationReceiptRequired="false">
  <DocumentEnvelope name="DE Req Orden" nameID="de-req-bt-orden-7" businessDocumentRef="bd-orden-1" />
  <ReceiptAcknowledgement name="Rcp Ack Req Orden" nameID="rcpack-req-bt-orden-8" signalDefinitionRef="ra1" />
    <ReceiptAcknowledgementException name="Rcp Ack Exc Req Orden" nameID="rcpackexc-req-bt-orden-9"
      signalDefinitionRef="rae1" />
    </RequestingBusinessActivity>
-   <RespondingBusinessActivity name="Res Orden" nameID="res-bt-orden-10" isAuthorizationRequired="false"
      isIntelligibleCheckRequired="false" isNonRepudiationRequired="false" isNonRepudiationReceiptRequired="false">
  <DocumentEnvelope name="DE Res Orden" nameID="de-res-bt-orden-11" businessDocumentRef="bd-ordenrespuesta-3"
      />
  <ReceiptAcknowledgement name="Rcp Ack Res Orden" nameID="rcpack-req-bt-orden-12" signalDefinitionRef="ra1" />
```

```
    <ReceiptAcknowledgementException name="Rcp Ack Exc Res Orden" nameID="rcpackexc-req-bt-orden-13"
      signalDefinitionRef="rae1" />
    </RespondingBusinessActivity>
    </CommercialTransaction>
    - <!-- Despacho -->
- <CommercialTransaction name="BT Despacho Insumos" nameID="bt-despachoinsumos-27">
  <RequestingRole name="Requester" nameID="requester-bt-despachoinsumos-27" />
  <RespondingRole name="responder" nameID="responder-bt-despachoinsumos-27" />
-   <RequestingBusinessActivity name="Req Despacho Insumos" nameID="req-bt-despachoinsumos-28"
      isAuthorizationRequired="false" isIntelligibleCheckRequired="false" isNonRepudiationRequired="false"
      isNonRepudiationReceiptRequired="false">
    <DocumentEnvelope name="DE Req Despacho Insumos" nameID="de-req-bt-despachoinsumos-29"
      businessDocumentRef="bd-despachoinsumos-21" />
  <ReceiptAcknowledgement name="Rcp Ack Req Despacho Insumos" nameID="rcpack-req-bt-despachoinsumos-30"
      signalDefinitionRef="ra1" />
   <ReceiptAcknowledgementException name="Rcp Ack Exc Req Despacho Insumos" nameID="rcpackexc-req-bt-
      despachoinsumos-31" signalDefinitionRef="rae1" />
    </RequestingBusinessActivity>
-   <RespondingBusinessActivity name="Res Despacho Insumos" nameID="res-bt-despachoinsumos-32"
      isAuthorizationRequired="false" isIntelligibleCheckRequired="false" isNonRepudiationRequired="false"
      isNonRepudiationReceiptRequired="false">
    <DocumentEnvelope name="DE Res Despacho Insumos" nameID="de-res-bt-despachoinsumos-33"
      businessDocumentRef="bd-reciboinsumos-23" />
  <ReceiptAcknowledgement name="Rcp Ack Res Despacho Insumos" nameID="rcpack-res-bt-despachoinsumos-34"
      signalDefinitionRef="ra1" />
   <ReceiptAcknowledgementException name="Rcp Ack Exc Res Despacho Insumos" nameID="rcpackexc-res-bt-
      despachoinsumos-35" signalDefinitionRef="rae1" />
    </RespondingBusinessActivity>
    </CommercialTransaction>
    - <!-- Orden Status (al solicitarse) -->
- <RequestResponse name="BT Orden Status Solicitud" nameID="bt-ordenstatussolicitud-60">
  <RequestingRole name="Requester" nameID="requester-bt-ordenstatussolicitud-60" />
  <RespondingRole name="responder" nameID="responder-bt-ordenstatussolicitud-60" />
-   <RequestingBusinessActivity name="Req Orden Status Solicitud" nameID="req-bt-ordenstatussolicitud-61"
      isAuthorizationRequired="false" isIntelligibleCheckRequired="false" isNonRepudiationRequired="false"
      isNonRepudiationReceiptRequired="false">
    <DocumentEnvelope name="DE Req Orden Status Solicitud" nameID="de-req-bt-ordenstatussolicitud-62"
      businessDocumentRef="bd-ordenstatussolicitud-50" />
    </RequestingBusinessActivity>
-   <RespondingBusinessActivity name="Res Orden Status Solicitud" nameID="res-bt-ordenstatussolicitud-63"
      isAuthorizationRequired="false" isIntelligibleCheckRequired="false" isNonRepudiationRequired="false"
      isNonRepudiationReceiptRequired="false">
```

```
    <DocumentEnvelope name="DE Res Orden Status Solicitud" nameID="de-res-bt-ordenstatussolicitud-64"
      businessDocumentRef="bd-ordenstatusreporte-52" />
    </RespondingBusinessActivity>
    </RequestResponse>
   - <!-- Orden Status (periódica) -->
- <Notification name="BT Orden Status Reporte" nameID="bt-ordenstatusreporte-70">
  <RequestingRole name="Requester" nameID="requester-bt-ordenstatusreporte-70" />
  <RespondingRole name="responder" nameID="responder-bt-ordenstatusreporte-70" />
- <RequestingBusinessActivity name="Req Orden Status Reporte" nameID="req-bt-ordenstatusreporte-71"
    isAuthorizationRequired="false" isIntelligibleCheckRequired="false" isNonRepudiationRequired="false"
    isNonRepudiationReceiptRequired="false" timeToAcknowledgeReceipt="PT10S" retryCount="1">
    <DocumentEnvelope name="DE Req Orden Status Reporte" nameID="de-req-bt-ordenstatusreporte-72"
      businessDocumentRef="bd-ordenstatusreporte-52" />
  <ReceiptAcknowledgement name="Rcp Ack Req Orden Status Reporte" nameID="rcpack-req-bt-ordenstatusreporte-53"
      signalDefinitionRef="ra1" />
   <ReceiptAcknowledgementException name="Rcp Ack Exc Req Orden Status Reporte" nameID="rcpackexc-req-bt-
      ordenstatusreporte-54" signalDefinitionRef="rae1" />
    </RequestingBusinessActivity>
  <RespondingBusinessActivity name="Res Orden Status Reporte" nameID="res-bt-ordenstatusreporte-74" />
    </Notification>
   - <!-- Recibo y Respuesta -->
- <CommercialTransaction name="BT Recibo" nameID="bt-recibo-632">
  <RequestingRole name="Requester" nameID="requester-bt-recibo-632" />
  <RespondingRole name="responder" nameID="responder-bt-recibo-632" />
- <RequestingBusinessActivity name="Req Recibo" nameID="req-bt-recibo-633" isAuthorizationRequired="false"
    isIntelligibleCheckRequired="false" isNonRepudiationRequired="false" isNonRepudiationReceiptRequired="false">
  <DocumentEnvelope name="DE Req Recibo" nameID="de-req-bt-recibo-634" businessDocumentRef="bd-recibo-604" />
  <ReceiptAcknowledgement name="Rcp Ack Req Recibo" nameID="rcpack-req-bt-recibo-635" signalDefinitionRef="ra1" />
   <ReceiptAcknowledgementException name="Rcp Ack Exc Req Recibo" nameID="rcpackexc-req-bt-recibo-636"
      signalDefinitionRef="rae1" />
    </RequestingBusinessActivity>
- <RespondingBusinessActivity name="Res Recibo" nameID="res-bt-recibo-637" isAuthorizationRequired="false"
    isIntelligibleCheckRequired="false" isNonRepudiationRequired="false" isNonRepudiationReceiptRequired="false">
  <DocumentEnvelope name="DE Res Recibo" nameID="de-res-bt-recibo-638" businessDocumentRef="bd-contrarecibo-606"
    />
  <ReceiptAcknowledgement name="Rcp Ack Res Recibo" nameID="rcpack-res-bt-recibo-639" signalDefinitionRef="ra1" />
   <ReceiptAcknowledgementException name="Rcp Ack Exc Res Recibo" nameID="rcpackexc-res-bt-recibo-640"
      signalDefinitionRef="rae1" />
    </RespondingBusinessActivity>
    </CommercialTransaction>
   - <!-- Colaboración de negocios principal -->
```

```
- <BusinessCollaboration name="Colaboracion del Ejemplo Sub-Contratista" nameID="bc-main-14"
    isInnerCollaboration="false">
  <Role name="Productor" nameID="productor-15" />
  <Role name="Sub-Contratista" nameID="subcontratista-16" />
  <TimeToPerform type="runtime" />
- <Start>
  <ToLink toBusinessStateRef="bs-main-2" />
    </Start>
- <BusinessTransactionActivity name="BTA Orden" nameID="bs-main-2" businessTransactionRef="bt-orden-5"
    hasLegalIntent="true">
  <TimeToPerform type="runtime" />
  <Performs currentRoleRef="productor-15" performsRoleRef="requester-bt-orden-5" />
  <Performs currentRoleRef="subcontratista-16" performsRoleRef="responder-bt-orden-5" />
  <EndsWhen expressionLanguage="ConditionGuardValue" expression="fase-ordenar-terminada" />
    </BusinessTransactionActivity>
- <Transition>
  <FromLink fromBusinessStateRef="bs-main-2" />
  <ToLink toBusinessStateRef="bs-main-3" />
    </Transition>
- <BusinessTransactionActivity name="BTA Despacho Insumos" nameID="bs-main-3" businessTransactionRef="bt-
    despachoinsumos-27" hasLegalIntent="true">
  <TimeToPerform type="runtime" />
  <Performs currentRoleRef="productor-15" performsRoleRef="requester-bt-despachoinsumos-27" />
  <Performs currentRoleRef="subcontratista-16" performsRoleRef="responder-bt-despachoinsumos-27" />
    </BusinessTransactionActivity>
- <Fork type="OR">
- <FromLink fromBusinessStateRef="bs-main-3">
  <ConditionExpression expressionLanguage="ConditionGuardValue" expression="despacho-insumos recibido" />
    </FromLink>
  <ToLink toBusinessStateRef="bs-main-4" />
  <ToLink toBusinessStateRef="bs-main-5" />
    </Fork>
- <BusinessTransactionActivity name="BTA Orden Status Solicitud" nameID="bs-main-4" businessTransactionRef="bt-
    ordenstatussolicitud-60" hasLegalIntent="true">
  <TimeToPerform type="runtime" />
  <Performs currentRoleRef="productor-15" performsRoleRef="requester-bt-ordenstatussolicitud-60" />
  <Performs currentRoleRef="subcontratista-16" performsRoleRef="responder-bt-ordenstatussolicitud-60" />
  <EndsWhen expressionLanguage="ConditionGuardValue" expression="fase-manufactura-terminada" />
    </BusinessTransactionActivity>
- <BusinessTransactionActivity name="BTA Orden Status Reporte" nameID="bs-main-5" businessTransactionRef="bt-
    ordenstatusreporte-70" hasLegalIntent="true">
  <TimeToPerform type="runtime" />
```

```
<Performs currentRoleRef="subcontratista-16" performsRoleRef="requester-bt-ordenstatusreporte-70" />
<Performs currentRoleRef="productor-15" performsRoleRef="responder-bt-ordenstatusreporte-70" />
<EndsWhen expressionLanguage="ConditionGuardValue" expression="fase-manufactura-terminada" />
</BusinessTransactionActivity>
- <Join waitForAll="false">
<FromLink fromBusinessStateRef="bs-main-4" />
<FromLink fromBusinessStateRef="bs-main-5" />
- <ToLink toBusinessStateRef="bs-main-11">
<ConditionExpression expressionLanguage="ConditionGuardValue" expression="fase-manufactura-terminada" />
</ToLink>
</Join>
- <BusinessTransactionActivity name="BTA Recibo" nameID="bs-main-11" businessTransactionRef="bt-recibo-632"
hasLegalIntent="true">
<TimeToPerform type="runtime" />
<Performs currentRoleRef="subcontratista-16" performsRoleRef="requester-bt-recibo-632" />
<Performs currentRoleRef="productor-15" performsRoleRef="responder-bt-recibo-632" />
<EndsWhen expressionLanguage="ConditionGuardValue" expression="recibo-finaliza" />
</BusinessTransactionActivity>
- <Decision>
<FromLink fromBusinessStateRef="bs-main-11" />
- <ToLink toBusinessStateRef="bs-main-12">
<ConditionExpression expressionLanguage="ConditionGuardValue" expression="Success" />
</ToLink>
- <ToLink toBusinessStateRef="bs-main-13">
<ConditionExpression expressionLanguage="ConditionGuardValue" expression="Failure" />
</ToLink>
</Decision>
<Success name="Colaboracion del Ejemplo Sub-Contratista finaliza exitosamente" nameID="bs-main-12" />
<Failure name="Colaboracion del Ejemplo Sub-Contratista finaliza con fallas" nameID="bs-main-13" />
</BusinessCollaboration>
</ProcessSpecification>
```

5.4 Una Herramienta de Software para ebXML BPSS

En la figura 8 se muestran las pantallas principales de FreeBXMLBP, ilustrando las funcionalidades que se presentan a simple vista. FreeBXMLBP es una herramienta diseñada para ayudar a los usuarios a crear especificaciones de procesos de negocios en base a *ebXML BPSS*. Más que ser un producto, por sus características y funcionamiento, es más un prototipo. En esencia permite realizar una definición apoyándose en elementos visuales. Desde su instalación hasta el uso de cualquiera de sus

opciones se aprecia un software muy básico, no muy bien desarrollado, con pocas funcionalidades y con una usabilidad muy mala.

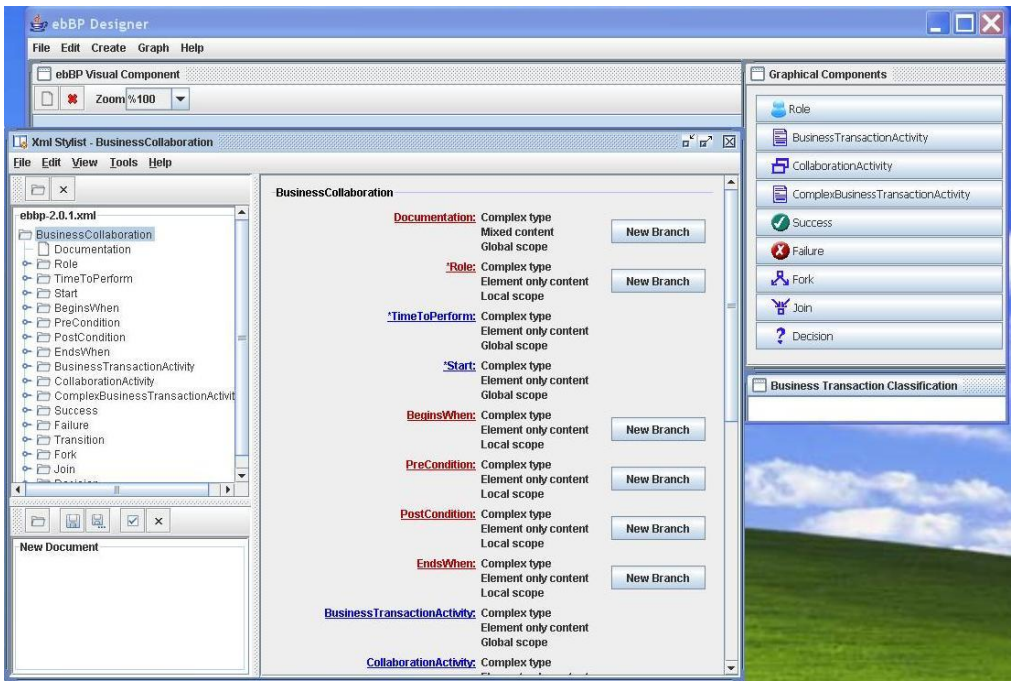

Figura 8 Pantallas principales de una herramienta para ebXML BPSS.

Capítulo 6

Comparación de WSBPEL y ebXML BPSS

El presente capítulo ofrece la comparación de *WSBPEL* y *ebXML BPSS*.

6.1 Sobre el Sustento de la Comparación

El presente trabajo permite dar un buen sustento al contraste de ambos estándares, ya que:

- Se estableció un contexto sobre la interoperabilidad y los conceptos fundamentales con los cuales se relaciona.
- Se profundizó en el estudio de cada estándar y sus conceptos técnicos.
- Se desarrolló una perspectiva nueva al platear procedimientos que captan los conceptos fundamentales de cada estándar, lo cual tiene gran valor pues las especificaciones técnicas de ambos estándares no incluyen ninguna metodología.
- Se tuvo un contexto de la PyME a lo largo de toda la investigación que delimito la amplitud de escenarios y permitió enfocarse en los aspectos importantes vinculados a ese contexto.
- Se ahondó en los detalles técnicos al estudiar ejemplos y realizar los casos de uso.
- Se mantuvo una actualización constante con las tendencias actuales más relevantes vinculadas al tema, surgiendo de esto, el hecho de migrar parte de la investigación a las versiones más recientes de los estándares.
- Se consideró informalmente la perspectiva real de herramientas de software existentes en el mercado, para asistir el uso de los estándares.

6.2 La Comparación

- *ebXML BPSS*, ofrece estrategias para adaptar el uso de algunos de sus elementos para el uso de servicios Web, mientras que *WSBPEL* tiene como base a los servicios Web.

- Aún cuando *WSBPEL* puede intercambiar datos estructurados (documentos), mediante tipos complejos, el intercambio de documentos estructurados es una base para *ebXML BPSS*, incluso pudiéndose definir el esquema particular que posee cada documento a ser utilizado.
- Una vez se asimilan los conceptos que envuelven a *ebXML BPSS*, se aprecia claridad en su objetivo de definir los procesos de interacción, cosa que no pasa para *WSBPEL* donde definir los procesos abstractos, no es transparente pues se presentan varios casos de uso para estos procesos, a través del empleo de *Profiles*, y es imposible asimilarlos sin antes entender la descripción de procesos ejecutables, ya que comparten la misma sintaxis.
- *ebXML BPSS* puede desempeñarse, sin implicar un esfuerzo adicional, en ambientes donde la robustez, seguridad y fiabilidad sean primordiales, mientras en el caso de *WSBPEL* esto no necesariamente es cierto. En la familia *ebXML*, donde existe incluso un componente completo para el intercambio de mensaje *ebMS*, se aprecia se han considerado prácticamente todos los aspectos de las grandes empresas, probablemente por tener orígenes vinculados a *EDI*. En el caso de *WSBPEL*, su naturaleza más universal, hace que no sean explícitas funcionalidades importantes para un buen número de grandes empresas como es el intercambio de documentos de negocios.
- Por su robustez y poder, *ebXML BPSS* o mejor dicho la familia *ebXML*, tiene ciclos de desarrollo e implementación considerablemente mayores a los de *WSBPEL*. Mencionando los conceptos que estarán implicados, para cada caso, se considera suficiente para aclarar este punto. Para *WSBPEL*, los conceptos de servicios *Web* y de motor *WSBPEL*, mientras para *ebXML*, *CPP*, *CPA*, registro/repositorio *ebXML*, *ebMS*, *core components ebXML*.
- Cuando se considera el objetivo de la interoperabilidad, en su forma más universal, multi-plataforma y a todos los niveles de empresas, se establece una relación mucho más trasparente con lo brindado por *WSBPEL*. *WSBPEL* es "relativamente sencillo" y ofrece un potencial nativo gracias a los servicios *Web*. En cambio *ebXML BPSS* y su familia, parecen cruzaran la línea de la complejidad deseada, situándose en un escenario particular y probablemente conveniente sólo para grandes empresas.
- El tiempo para asimilar los conceptos involucrados para *ebXML BPSS* es como mínimo 3 veces mayor que el necesario para *WSBPEL*. Para *WSBPEL* aún cuando el estándar puede ser largo e implicar una buena cantidad de términos y situaciones, se observa una información más coherente y un contexto más auto-contenido en el mismo estándar. La principal razón

de esto es que la comprensión de *ebXML BPSS* depende de otros elementos externos de la familia *ebXML (CPPA, CPPP, BSI,* etc.), considerablemente más, de lo que depende *WSBPEL* de la comprensión de elementos externos (*WSDL, UDDI,* etc.).

- La presencia de herramientas de software en el mercado que permitan asistir el empleo de cualquiera de los estándares, es un factor considerable entre *WSBPEL y ebXML*, y es otra razón a favor de *WSBPEL*. Como una consecuencia indirecta de la presente investigación se pudo observar que las herramientas para *WSBPEL* tienen una funcionalidad y madurez mucho mayor a las de *ebXML*. Las dos herramientas de software libre que fueron referenciadas, son la mejor muestra de esto. Y aunque no se reviso con el mismo detalle ninguna herramienta paga, se observo mucha mayor existencia de posibilidades para WSBPEL. Se considera que ActiveBPEL y FreeBXMLBP son las mejores herramientas que existen en el mercado para los estándares *WSBPEL* y *ebXML BPSS* respectivamente. En el contexto de PyME tener una herramienta completa como ActiveBPEL, que adicionalmente es libre, y que no sólo te asiste en el desarrollo sino en la implementación al traer funcionalidades para desplegar en motores *WSBPEL*, le da una gran ventaja a *WSBPEL* sobre *ebXML BPSS*.
- La necesidad de negociación y acuerdo entre compañeros de negocios es considerable mayor para *ebXML BPSS* que para *WSBPEL*. Para *ebXML BPSS* se debe acordar sobre mayor cantidad de elementos por implicar el uso de la familia *ebXML*. Puede que esto se realice automáticamente luego, después de establecer una base común y los esfuerzos necesarios para emplear elementos que no utilizan la misma base, pero en cualquier caso *WSBPEL* ofrece un contexto más universal que permite reducir considerablemente esas negociaciones en todas las etapas.
- Si se considera lo escalable de una implementación empleando alguno de los estándares, definitivamente *WSBPEL* sería más adecuado para *PyME*. En una solución ya implementa probablemente lo escalable pueda ser comparable, pero si consideramos lo escalable de principio a fin, el esfuerzo inicial necesario para *ebXML BPSS* es mucho mayor que el necesario para *WSBPEL*. Esto último por los componentes de la familia *ebXML* que se deberían implementar al principio, mientras para *WSBPEL* con una plataforma que soporte servicios Web y un motor *WSBPEL* se podría comenzar.

6.3 Comparación por características

Característica del estándar	*WSBPEL*	*ebXML BPSS*
Basado en servicios Web	Si	No(Puede utilizarlos mediante una adaptación)
Orientado a intercambio de documentos	No	Si(Permite incluso especificar un formato particular para documento)
Enfocado en procesos públicos	No(Su enfoque son los procesos privados ejecutables)	Si
Robustez, seguridad, fiabilidad más adecuada para grandes empresas	No	Si
Auto-contenido	Si	No(Es necesario conceptos de la familia ebXML)
Complejidad para asimilar sus conceptos	Baja	Alta
Negociación entre empresas antes de implementar	Baja	Alta
Tiempo para implementarse	De Corto a mediano plazo	De mediano a largo plazo

Capítulo 7

Conclusiones

El presente capítulo ofrece las conclusiones generales y los futuros trabajos que se considera se deberían realizar.

7.1 Conclusiones

- Aún cuando puede existir diferencia en el enfoque específico de cada uno de los estándares, si se considera la idea principal de la búsqueda de interoperabilidad, se concluye con claridad que es *WSBPEL* quien debe ser la primera opción, si se pretende la adopción de alguno de los dos estándares en el contexto de la PyME. Sintetizando las razones expuestas en el capítulo anterior, podemos decir que es el enfoque hacia servicios Web que le da un poder, generalidad e independencia que difícilmente se puede alcanzar con una base de fundamentos que permite etapas de asimilación, análisis e implementación relativamente sencillas si se comparan a las necesitadas para *ebXML BPSS*. Y en el contexto de PyME esa sencillez relativa es primordial.
- Aunque no se puede establecer con la claridad de la conclusión anterior, se puede inferir que en el contexto de grandes empresas también se debe considerarse inicialmente el estándar *WSBPEL*. Durante la presente investigación se observó una tendencia hacia la adopción de servicios *Web* y los estándares vinculados, incluso apreciada en las adaptaciones hechas en *ebXML BPSS* para trabajar con estos servicios, que hacen aún más relevante a *WSBPEL*. Es así que, la sencillez relativa de *WSBPEL*, aunque no tenga tanto peso como en la PyME, la naturaleza hacia servicios Web, y la no existencia de razones importantes para emplear *ebXML*, constituyen fundamentos para emplear *WSBPEL* en vez de *ebXML BPSS*.
- El tema de la interoperabilidad no es trivial, y se necesita un buen contexto antes de vincularse con los estándares analizados. Conceptos como los de procesos de negocios, esquemas *XML*, servicios *Web* por colocar unos ejemplos deben ser bien entendidos.

- Las posibilidades de implementación de una solución parcial ó completa para PyME, empleando *WSBPEL*, son considerablemente mayores que las de *ebXML BPSS*. Adicionalmente a puntos ya establecidos, como sencillez relativa, asimilación, existencia de herramientas de software y necesidad de negociación durante el desarrollo del presente trabajo en ocasiones se hizo la pregunta ¿Qué se necesitaría para implementar todo esto desde cero? Y la respuesta en prácticamente todos los escenarios para PyME es que se necesita menor esfuerzo y se tiene más al alcance lo necesario para *WSBPEL*.

7.2 Futuros Trabajos

- *WSBPEL* es un estándar relativamente nuevo cuyo estudio, ante las actuales tendencias tecnológicas, es necesario académica y económicamente hablando. La presente investigación ha concluido, sin lugar a duda, que *WSBPEL* es la primera opción en el contexto de PyME, y se considera que con este trabajo como base se puede desarrollar una investigación que conlleve la aplicación de *WSBPEL* en un caso real donde se formalice su análisis (procesos privados y públicos), se extraigan las variables necesarias para la puesta a tono de un sistema interoperable mediante *WSBPEL* y se tenga retro-alimentación de su completa implementación. La descripción de los procesos ejecutables (privados) y el empleo de un motor *WSBPEL* existente serían conceptos claves.
- Aún cuando el presente trabajo, adicional a la comparación de estándares, realiza un aporte significativo al establecer dos procedimientos para el empleo de cada estándar, se considera que el desarrollo de un método formal y detallado para cada caso, sería un proyecto interesante y de importancia para el área de descripción de procesos de negocios.

Glosario

A

Actividad de Negocio: Es usada para representar el estado del proceso de negocios de uno de los compañeros. Por ejemplo el solicitante está, ó en el estado enviando la solicitud, ó en el estado esperando por la respuesta, ó en el estado recibiendo (14).

B

BSI(Business Service Interface): La implementación de un compañero de la definición compartida de los estados de negocios y acciones relevantes, para una meta común de negocios; puede ser identificada como componente(s) de negocios que refuerza(n) la semántica y alcanza estados de alineación para los participantes (13).

C

Colaboración binaria: Un conjunto de Actividades de negocio entre dos compañeros abstractos o roles de alto nivel (13).

Colaboración de Negocios: Un conjunto de roles que compañeros de negocios asumen en una actividad de negocios, a través del intercambio de mensajes de negocios en un ambiente, que más que ser controlado es uno-a-uno, para alcanzar una meta de negocios (13).

Componentes base ebXML: Tipos de datos elementales para la construcción de documentos de negocio (3).

Coreografía: El ordenamiento y transición entre transacciones de negocios, o colaboraciones, dentro de colaboraciones de negocios (13).

CPA(Collaboration Protocol Agrement): Información acordada entre dos(o más) participantes que identifica o describe el protocolo de colaboración específico, que ellos han acordado usar. Un CPA indica que harán los participantes envueltos cuando llevan a cabo un proceso colaborativo. Un CPA es representable por un documento (14).

CPP(Collaboration Protocol Profile): Información acerca de un participante que puede ser usada para describir uno o más procesos colaborativos y los protocolos colaborativos asociados que el participante soporta. Un CPP indica que puede hacer un participante para llevar a cabo un proceso colaborativo. Un CPP es representable por un documento, Mientras lógicamente, un CPP es un

documento solo, en la práctica, el CPP puede ser un conjunto de documentos enlazados que expresan varios aspectos de las capacidades. Un CPP no es un acuerdo, sólo representa las capacidades de un participante (14).

D

Documento de Negocios: Una estructura lógica que puede ser compuesta desde más de una fuente y puede ser complementada por documentos no estructurados como anexos.

E

EDI(Electronic Data Interchange): Formato estándar para el intercambio de datos según la norma ANSI X12 (3).

F

Flujo de documentos de negocios: Una transacción de negocios es realizada como flujos de documentos de negocios entre el rol que solicita y el rol que responde (13).

P

Proceso de negocios: La manera por la cual una o más actividades son completadas en la práctica de cómo operan los negocios.

S

Servicios Web: Aplicaciones auto-descritas y modulares que pueden ser publicadas, localizadas e invocadas de cualquier parte en la Web o en la red de área local. El proveedor y el consumidor del servicio Web no tiene que preocuparse acerca del sistema operativo, lenguaje del ambiente, ó modelo de componentes usado para crear o acceder al servicio, pues ellos son basados en estándares de internet independientes y abiertos como XML, HTTP y SMTP (17).

SOAP(Simple Object Access Protocol): Es una especificación de un protocolo para acceder a servicios, funciones, componentes y objetos en servidores remotos. Utiliza XML y HTTP para ser independiente en cuanto a plataforma tecnológica.

T

Transacción de negocios: Una unidad de trabajo en un arreglo de intercambio entre dos participantes que juegan roles (abstractos pero ya declarados) opuestos; consiste de uno o dos flujos predefinidos de documentos de negocios; Traerá como resultado un estado de éxito ó falla (13).

U

UDDI(Universal Description, Discovery and Integration): Permite mantener un registro global de servicios Web (18). Uno de los servicios más importantes dentro de los Servicios Web al ser una especie de páginas amarillas electrónicas con un detalle alto en cuanto a la actividad de la empresa y su oferta de productos y servicios que permite una interrogación automática por otros sistemas electrónicos (3).

W

WSDL(Web Service Description Language): Un lenguaje desarrollado por Microsoft e IBM que permite describir los Servicios Web como una colección de operaciones y respuestas (18).

Índice De Términos

Bibliografía

1. **ebXML Standard Contributors.** *ebXML Business Process Specification Schema Technical Specification v2.0.4.* [pdf] Diciembre de 2006.

2. **WSBPEL Standard Contributors.** *Web Services Business Process Execution Language v2.0.* [pdf] Abril de 2007.

3. **Advanced Quality Solutions.** *ebXML Una Visión General.* [pdf] 2002.

4. **Media Wiley.** *An Overview Of E-Comerce And ebXML.* [pdf] 2002.

5. **0`Riordan, David.** *Business Process Standards For Web Services.* [pdf] Chicago : Tect, 2002.

6. **Muehlen, Michael y Nickerson, Jeffrey.** *Developing Web Sevices Choreography Standards.* [pdf] 2004.

7. PyMe. *Wikipedia.* [En línea] http://es.wikipedia.org/wiki/PYME.

8. **Box, D. y Ehnebuske, D.** Simple Object Access Protocol(SOAP) 1.1. *W3C ORG.* [En línea] http://www.w3.org/TR/2000/NOTE-SOAP-20000508.

9. **Christensen, E. y Curbera, F.** Web Services Definition Language(WSDL) 1.1. *W3C ORG.* [En línea] http://www.w3.org/TR/2001/NOTE-wsdl-20010315.

10. **Clement, L. y Hately, A.** UDDI Version 3.0.2. *UDDI ORG.* [En línea] http://uddi.org/pubs/uddi-v3.0.2-20041019.htm.

11. ebXML WebSite. [En línea] http://www.ebxml.org/.

12. **UN/CEFACT.** *Core Components Technical Specification V.2.01.* [pdf] 2003.

13. **ebXML Standard Contributors.** *ebXML BPSS Technical Specification Appendices v2.0.4.* [pdf] Diciembre de 2006.

14. **OASIS ebXML CPP/A Technical Committee.** *Collaboration-Protocol Profile And Agreement Specification V.2.0.* [pdf] Septiembre de 2002.

15. **Montilva, Jonas y Barrios, Judith.** *A Component-Based Method For Developing Web Applications.* [pdf]

16. **Montilva, Jonas y Hamzan, Khalid.** *The Watch Model For Developing Business Software In Small And Midsize Organizations.* [pdf]

17. **Cauldwell, P.** *Professional XML Web Services.* Wrox Press, 2001.

18. **Vallecillo, Antonio.** *Servicios Web y MDA.* [pdf] Dpto. Computación : Universidad de Málaga.

19. WSBPEL WebSite. [En línea] http://www.oasis-open.org/committees/wsbpel/.

20. BPEL4WS. *Cover Pages.* [En línea] http://xml.coverpages.org/bpel4ws.html.

21. ebXML Initiative. *Cover Pages.* [En línea] http://xml.coverpages.org/ebXML.html.

22. Una Política Moderna Para Las PYME. *Europa SCADPlus.* [En línea] http://europa.eu/scadplus/leg/es/lvb/n26106.htm.

23. **Ghalimi, Ismael Chang.** *BPM 2.0.* [pdf]

24. **Aissi, Selim y Malu, Pallavi.** *E-Business Process Modeling: The Next Big Step.* [pdf]

25. **Camara, Javier y Canal, Carlos.** *Issues in the formalization of Web Service Orchestrations.* [pdf]

26. **ebXML BP/CC Analysis Team.** *Business Process and Business Information Analysis Overview.* [pdf]

27. **ebXML Business Process Project Team.** *Business Process Analysis Worksheets & Guidelines v1.0.* [pdf]

28. **OASIS ebXML Messaging Services Technical Committee.** *Message Service Specification V.2.0.*

29. **Jurie, Matjaz.** *BPEL And Java.*

30. Como Organizar, Administrar y Perfeccionar Una PyME. *Portal Bioceánico.* [En línea] http://www.portalbioceanico.com/nuevasactividades_pymes.htm.

31. **ebXML BP-CC Analysis Team.** *ebXML E-Commerce Patterns v1.0.* [pdf]

Anexos

A.1 Esquema WSBPEL Para Procesos Abstractos De Negocios

```
<?xml version="1.0" encoding="UTF-8" ?>
- <!--
   Copyright (c) OASIS Open 2006-2007. All Rights Reserved.
 -->
- <xsd:schema targetNamespace="http://docs.oasis-open.org/wsbpel/2.0/process/abstract" xmlns="http://docs.oasis-open.org/wsbpel/2.0/process/abstract" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsd-derived="http://docs.oasis-open.org/wsbpel/2.0/process/abstract" elementFormDefault="qualified" blockDefault="#all">
- <xsd:annotation>
  - <xsd:documentation>
    Schema for Abstract Process Common Base for WS-BPEL 2.0 OASIS Standard 11th April, 2007 NOTE: Here is the changes path from Exec BPEL XSD to this XSD: (a) have a global replace from minOccurs=1 to minOccurs=0 for element declaration (b) have a global replace from use=required to use=optional for attribute declaration (c) change "xsd-derived" from XSD NS to this Abstract BPEL NS and define the corresponding simple types (d) adding "##opaque" to BPELVariableName, tInitiate, tPattern, tRoles, tBoolean (e) add abstractProcessProfile to tProcess (f) add opaqueActivity (g) add tOpaqueBoolean type and add opaque attribute tExpression, tQuery, tFrom, tTo (h) add opaqueFrom element and introduce fromGroup (i) Change the sequence for "tOnAlarmEvent"
    - <![CDATA[
                                                    From:
                                                    ---------------------
                                                    <xsd:sequence>
                                                            <xsd:choice>
                                                                    <xsd:sequence>
                                                                            <xsd:group ref="forOrUntilGroup"
    minOccurs="1"/>
                                                                            <xsd:element ref="repeatEvery"
    minOccurs="0"/>
                                                                    </xsd:sequence>
                                                                    <xsd:element ref="repeatEvery" minOccurs="1"/>
                                                            </xsd:choice>
                                                            <xsd:element ref="scope" minOccurs="1"/>
                                                    </xsd:sequence>
                                                    ---------------------
                                                    To:
                                                    ---------------------
                                                    <xsd:sequence>
                                                            <xsd:group ref="forOrUntilGroup" minOccurs="0"/>
                                                            <xsd:element ref="repeatEvery" minOccurs="0"/>
                                                            <xsd:element ref="scope" minOccurs="0"/>
                                                    </xsd:sequence>
                                                    ---------------------

    ]]>
    </xsd:documentation>
    </xsd:annotation>
 <xsd:import namespace="http://www.w3.org/XML/1998/namespace" schemaLocation="http://www.w3.org/2001/xml.xsd" />
- <xsd:element name="process" type="tProcess">
- <xsd:annotation>
 <xsd:documentation>This is the root element for a WS-BPEL 2.0 process.</xsd:documentation>
    </xsd:annotation>
```

```
    </xsd:element>
- <xsd:complexType name="tProcess">
- <xsd:complexContent>
- <xsd:extension base="tExtensibleElements">
- <xsd:sequence>
  <xsd:element ref="extensions" minOccurs="0" />
  <xsd:element ref="import" minOccurs="0" maxOccurs="unbounded" />
  <xsd:element ref="partnerLinks" minOccurs="0" />
  <xsd:element ref="messageExchanges" minOccurs="0" />
  <xsd:element ref="variables" minOccurs="0" />
  <xsd:element ref="correlationSets" minOccurs="0" />
  <xsd:element ref="faultHandlers" minOccurs="0" />
  <xsd:element ref="eventHandlers" minOccurs="0" />
  <xsd:group ref="activity" minOccurs="0" />
    </xsd:sequence>
  <xsd:attribute name="name" type="xsd-derived:NCName" use="optional" />
  <xsd:attribute name="targetNamespace" type="xsd-derived:anyURI" use="optional" />
  <xsd:attribute name="queryLanguage" type="xsd-derived:anyURI" default="urn:oasis:names:tc:wsbpel:2.0:sublang:xpath1.0" />
  <xsd:attribute name="expressionLanguage" type="xsd-derived:anyURI" default="urn:oasis:names:tc:wsbpel:2.0:sublang:xpath1.0" />
  <xsd:attribute name="suppressJoinFailure" type="tBoolean" default="no" />
  <xsd:attribute name="exitOnStandardFault" type="tBoolean" default="no" />
  <xsd:attribute name="abstractProcessProfile" type="xsd:anyURI" use="required" />
    </xsd:extension>
    </xsd:complexContent>
    </xsd:complexType>
- <xsd:complexType name="tExtensibleElements">
- <xsd:annotation>
  <xsd:documentation>This type is extended by other component types to allow elements and attributes from other namespaces to be added
      at the modeled places.</xsd:documentation>
    </xsd:annotation>
- <xsd:sequence>
  <xsd:element ref="documentation" minOccurs="0" maxOccurs="unbounded" />
  <xsd:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded" />
    </xsd:sequence>
  <xsd:anyAttribute namespace="##other" processContents="lax" />
    </xsd:complexType>
  <xsd:element name="documentation" type="tDocumentation" />
- <xsd:complexType name="tDocumentation" mixed="true">
- <xsd:sequence>
  <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded" />
    </xsd:sequence>
  <xsd:attribute name="source" type="xsd-derived:anyURI" />
  <xsd:attribute ref="xml:lang" />
    </xsd:complexType>
- <xsd:group name="activity">
- <xsd:annotation>
```

```
<xsd:documentation>All standard WS-BPEL 2.0 activities in alphabetical order. Basic activities and structured activities. Addtional constraints: - rethrow activity can be used ONLY within a fault handler (i.e. "catch" and "catchAll" element) - compensate or compensateScope activity can be used ONLY within a fault handler, a compensation handler or a termination handler</xsd:documentation>
</xsd:annotation>
- <xsd:choice>
<xsd:element ref="assign" />
<xsd:element ref="compensate" />
<xsd:element ref="compensateScope" />
<xsd:element ref="empty" />
<xsd:element ref="exit" />
<xsd:element ref="extensionActivity" />
<xsd:element ref="flow" />
<xsd:element ref="forEach" />
<xsd:element ref="if" />
<xsd:element ref="invoke" />
<xsd:element ref="pick" />
<xsd:element ref="receive" />
<xsd:element ref="repeatUntil" />
<xsd:element ref="reply" />
<xsd:element ref="rethrow" />
<xsd:element ref="scope" />
<xsd:element ref="sequence" />
<xsd:element ref="throw" />
<xsd:element ref="validate" />
<xsd:element ref="wait" />
<xsd:element ref="while" />
<xsd:element ref="opaqueActivity" />
</xsd:choice>
</xsd:group>
<xsd:element name="extensions" type="tExtensions" />
- <xsd:complexType name="tExtensions">
- <xsd:complexContent>
- <xsd:extension base="tExtensibleElements">
- <xsd:sequence>
<xsd:element ref="extension" minOccurs="0" maxOccurs="unbounded" />
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:element name="extension" type="tExtension" />
- <xsd:complexType name="tExtension">
- <xsd:complexContent>
- <xsd:extension base="tExtensibleElements">
<xsd:attribute name="namespace" type="xsd-derived:anyURI" use="optional" />
<xsd:attribute name="mustUnderstand" type="tBoolean" use="optional" />
</xsd:extension>
```

```
        </xsd:complexContent>
        </xsd:complexType>
    <xsd:element name="import" type="tImport" />
  - <xsd:complexType name="tImport">
  - <xsd:complexContent>
  - <xsd:extension base="tExtensibleElements">
    <xsd:attribute name="namespace" type="xsd-derived:anyURI" use="optional" />
    <xsd:attribute name="location" type="xsd-derived:anyURI" use="optional" />
    <xsd:attribute name="importType" type="xsd-derived:anyURI" use="optional" />
        </xsd:extension>
        </xsd:complexContent>
        </xsd:complexType>
    <xsd:element name="partnerLinks" type="tPartnerLinks" />
  - <xsd:complexType name="tPartnerLinks">
  - <xsd:complexContent>
  - <xsd:extension base="tExtensibleElements">
  - <xsd:sequence>
    <xsd:element ref="partnerLink" minOccurs="0" maxOccurs="unbounded" />
        </xsd:sequence>
        </xsd:extension>
        </xsd:complexContent>
        </xsd:complexType>
    <xsd:element name="partnerLink" type="tPartnerLink" />
  - <xsd:complexType name="tPartnerLink">
  - <xsd:complexContent>
  - <xsd:extension base="tExtensibleElements">
    <xsd:attribute name="name" type="xsd-derived:NCName" use="optional" />
    <xsd:attribute name="partnerLinkType" type="xsd-derived:QName" use="optional" />
    <xsd:attribute name="myRole" type="xsd-derived:NCName" />
    <xsd:attribute name="partnerRole" type="xsd-derived:NCName" />
    <xsd:attribute name="initializePartnerRole" type="tBoolean" />
        </xsd:extension>
        </xsd:complexContent>
        </xsd:complexType>
    <xsd:element name="messageExchanges" type="tMessageExchanges" />
  - <xsd:complexType name="tMessageExchanges">
  - <xsd:complexContent>
  - <xsd:extension base="tExtensibleElements">
  - <xsd:sequence>
    <xsd:element ref="messageExchange" minOccurs="0" maxOccurs="unbounded" />
        </xsd:sequence>
        </xsd:extension>
        </xsd:complexContent>
        </xsd:complexType>
    <xsd:element name="messageExchange" type="tMessageExchange" />
  - <xsd:complexType name="tMessageExchange">
  - <xsd:complexContent>
```

```
- <xsd:extension base="tExtensibleElements">
  <xsd:attribute name="name" type="xsd-derived:NCName" use="optional" />
      </xsd:extension>
      </xsd:complexContent>
      </xsd:complexType>
  <xsd:element name="variables" type="tVariables" />
- <xsd:complexType name="tVariables">
- <xsd:complexContent>
- <xsd:extension base="tExtensibleElements">
- <xsd:sequence>
  <xsd:element ref="variable" minOccurs="0" maxOccurs="unbounded" />
      </xsd:sequence>
      </xsd:extension>
      </xsd:complexContent>
      </xsd:complexType>
  <xsd:element name="variable" type="tVariable" />
- <xsd:complexType name="tVariable">
- <xsd:complexContent>
- <xsd:extension base="tExtensibleElements">
- <xsd:sequence>
  <xsd:group ref="fromGroup" minOccurs="0" />
      </xsd:sequence>
  <xsd:attribute name="name" type="BPELVariableName" use="optional" />
  <xsd:attribute name="messageType" type="xsd-derived:QName" use="optional" />
  <xsd:attribute name="type" type="xsd-derived:QName" use="optional" />
  <xsd:attribute name="element" type="xsd-derived:QName" use="optional" />
      </xsd:extension>
      </xsd:complexContent>
      </xsd:complexType>
- <xsd:simpleType name="BPELVariableName">
- <xsd:union>
- <xsd:simpleType>
- <xsd:restriction base="xsd:NCName">
  <xsd:pattern value="[^\.]+" />
      </xsd:restriction>
      </xsd:simpleType>
- <xsd:simpleType>
- <xsd:restriction base="xsd:string">
  <xsd:enumeration value="##opaque" />
      </xsd:restriction>
      </xsd:simpleType>
      </xsd:union>
      </xsd:simpleType>
  <xsd:element name="correlationSets" type="tCorrelationSets" />
- <xsd:complexType name="tCorrelationSets">
- <xsd:complexContent>
- <xsd:extension base="tExtensibleElements">
```

```
- <xsd:sequence>
  <xsd:element ref="correlationSet" minOccurs="0" maxOccurs="unbounded" />
      </xsd:sequence>
      </xsd:extension>
      </xsd:complexContent>
      </xsd:complexType>
  <xsd:element name="correlationSet" type="tCorrelationSet" />
- <xsd:complexType name="tCorrelationSet">
- <xsd:complexContent>
- <xsd:extension base="tExtensibleElements">
  <xsd:attribute name="properties" type="QNames" use="optional" />
  <xsd:attribute name="name" type="xsd-derived:NCName" use="optional" />
      </xsd:extension>
      </xsd:complexContent>
      </xsd:complexType>
- <xsd:simpleType name="QNames">
- <xsd:restriction>
- <xsd:simpleType>
  <xsd:list itemType="xsd-derived:QName" />
      </xsd:simpleType>
  <xsd:minLength value="1" />
      </xsd:restriction>
      </xsd:simpleType>
  <xsd:element name="faultHandlers" type="tFaultHandlers" />
- <xsd:complexType name="tFaultHandlers">
- <xsd:complexContent>
- <xsd:extension base="tExtensibleElements">
- <xsd:sequence>
  <xsd:element ref="catch" minOccurs="0" maxOccurs="unbounded" />
  <xsd:element ref="catchAll" minOccurs="0" />
      </xsd:sequence>
      </xsd:extension>
      </xsd:complexContent>
      </xsd:complexType>
- <xsd:element name="catch" type="tCatch">
- <xsd:annotation>
    <xsd:documentation>This element can contain all activities including the activities compensate, compensateScope and
      rethrow.</xsd:documentation>
      </xsd:annotation>
      </xsd:element>
- <xsd:complexType name="tCatch">
- <xsd:complexContent>
- <xsd:extension base="tActivityContainer">
  <xsd:attribute name="faultName" type="xsd-derived:QName" />
  <xsd:attribute name="faultVariable" type="BPELVariableName" />
  <xsd:attribute name="faultMessageType" type="xsd-derived:QName" />
  <xsd:attribute name="faultElement" type="xsd-derived:QName" />
```

```
      </xsd:extension>
      </xsd:complexContent>
      </xsd:complexType>
- <xsd:element name="catchAll" type="tActivityContainer">
- <xsd:annotation>
    <xsd:documentation>This element can contain all activities including the activities compensate, compensateScope and
      rethrow.</xsd:documentation>
      </xsd:annotation>
      </xsd:element>
- <xsd:complexType name="tActivityContainer">
- <xsd:complexContent>
- <xsd:extension base="tExtensibleElements">
- <xsd:sequence>
  <xsd:group ref="activity" minOccurs="0" />
      </xsd:sequence>
      </xsd:extension>
      </xsd:complexContent>
      </xsd:complexType>
  <xsd:element name="eventHandlers" type="tEventHandlers" />
- <xsd:complexType name="tEventHandlers">
- <xsd:annotation>
  <xsd:documentation>XSD Authors: The child element onAlarm needs to be a Local Element Declaration, because there is another onAlarm
      element defined for the pick activity.</xsd:documentation>
      </xsd:annotation>
- <xsd:complexContent>
- <xsd:extension base="tExtensibleElements">
- <xsd:sequence>
  <xsd:element ref="onEvent" minOccurs="0" maxOccurs="unbounded" />
  <xsd:element name="onAlarm" type="tOnAlarmEvent" minOccurs="0" maxOccurs="unbounded" />
      </xsd:sequence>
      </xsd:extension>
      </xsd:complexContent>
      </xsd:complexType>
  <xsd:element name="onEvent" type="tOnEvent" />
- <xsd:complexType name="tOnEvent">
- <xsd:complexContent>
- <xsd:extension base="tOnMsgCommon">
- <xsd:sequence>
  <xsd:element ref="scope" minOccurs="0" />
      </xsd:sequence>
  <xsd:attribute name="messageType" type="xsd-derived:QName" use="optional" />
  <xsd:attribute name="element" type="xsd-derived:QName" use="optional" />
      </xsd:extension>
      </xsd:complexContent>
      </xsd:complexType>
- <xsd:complexType name="tOnMsgCommon">
- <xsd:annotation>
```

```
<xsd:documentation>XSD Authors: The child element correlations needs to be a Local Element Declaration, because there is another correlations element defined for the invoke activity.</xsd:documentation>
</xsd:annotation>
- <xsd:complexContent>
- <xsd:extension base="tExtensibleElements">
- <xsd:sequence>
<xsd:element name="correlations" type="tCorrelations" minOccurs="0" />
<xsd:element ref="fromParts" minOccurs="0" />
</xsd:sequence>
<xsd:attribute name="partnerLink" type="xsd-derived:NCName" use="optional" />
<xsd:attribute name="portType" type="xsd-derived:QName" use="optional" />
<xsd:attribute name="operation" type="xsd-derived:NCName" use="optional" />
<xsd:attribute name="messageExchange" type="xsd-derived:NCName" use="optional" />
<xsd:attribute name="variable" type="BPELVariableName" use="optional" />
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
- <xsd:complexType name="tCorrelations">
- <xsd:annotation>
<xsd:documentation>XSD Authors: The child element correlation needs to be a Local Element Declaration, because there is another correlation element defined for the invoke activity.</xsd:documentation>
</xsd:annotation>
- <xsd:complexContent>
- <xsd:extension base="tExtensibleElements">
- <xsd:sequence>
<xsd:element name="correlation" type="tCorrelation" minOccurs="0" maxOccurs="unbounded" />
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
- <xsd:complexType name="tCorrelation">
- <xsd:complexContent>
- <xsd:extension base="tExtensibleElements">
<xsd:attribute name="set" type="xsd-derived:NCName" use="optional" />
<xsd:attribute name="initiate" type="tInitiate" default="no" />
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
- <xsd:simpleType name="tInitiate">
- <xsd:restriction base="xsd:string">
<xsd:enumeration value="yes" />
<xsd:enumeration value="join" />
<xsd:enumeration value="no" />
<xsd:enumeration value="##opaque" />
</xsd:restriction>
</xsd:simpleType>
- <xsd:complexType name="tOnAlarmEvent">
```

```
- <xsd:complexContent>
- <xsd:extension base="tExtensibleElements">
- <xsd:sequence>
  <xsd:group ref="forOrUntilGroup" minOccurs="0" />
  <xsd:element ref="repeatEvery" minOccurs="0" />
  <xsd:element ref="scope" minOccurs="0" />
      </xsd:sequence>
      </xsd:extension>
      </xsd:complexContent>
      </xsd:complexType>
- <xsd:group name="forOrUntilGroup">
- <xsd:choice>
  <xsd:element ref="for" minOccurs="0" />
  <xsd:element ref="until" minOccurs="0" />
      </xsd:choice>
      </xsd:group>
  <xsd:element name="for" type="tDuration-expr" />
  <xsd:element name="until" type="tDeadline-expr" />
  <xsd:element name="repeatEvery" type="tDuration-expr" />
- <xsd:complexType name="tActivity">
- <xsd:complexContent>
- <xsd:extension base="tExtensibleElements">
- <xsd:sequence>
  <xsd:element ref="targets" minOccurs="0" />
  <xsd:element ref="sources" minOccurs="0" />
      </xsd:sequence>
  <xsd:attribute name="name" type="xsd-derived:NCName" />
  <xsd:attribute name="suppressJoinFailure" type="tBoolean" use="optional" />
      </xsd:extension>
      </xsd:complexContent>
      </xsd:complexType>
  <xsd:element name="targets" type="tTargets" />
- <xsd:complexType name="tTargets">
- <xsd:complexContent>
- <xsd:extension base="tExtensibleElements">
- <xsd:sequence>
  <xsd:element ref="joinCondition" minOccurs="0" />
  <xsd:element ref="target" minOccurs="0" maxOccurs="unbounded" />
      </xsd:sequence>
      </xsd:extension>
      </xsd:complexContent>
      </xsd:complexType>
  <xsd:element name="joinCondition" type="tCondition" />
  <xsd:element name="target" type="tTarget" />
- <xsd:complexType name="tTarget">
- <xsd:complexContent>
- <xsd:extension base="tExtensibleElements">
```

```
<xsd:attribute name="linkName" type="xsd-derived:NCName" use="optional" />
    </xsd:extension>
    </xsd:complexContent>
    </xsd:complexType>
<xsd:element name="sources" type="tSources" />
- <xsd:complexType name="tSources">
- <xsd:complexContent>
- <xsd:extension base="tExtensibleElements">
- <xsd:sequence>
<xsd:element ref="source" minOccurs="0" maxOccurs="unbounded" />
    </xsd:sequence>
    </xsd:extension>
    </xsd:complexContent>
    </xsd:complexType>
<xsd:element name="source" type="tSource" />
- <xsd:complexType name="tSource">
- <xsd:complexContent>
- <xsd:extension base="tExtensibleElements">
- <xsd:sequence>
<xsd:element ref="transitionCondition" minOccurs="0" />
    </xsd:sequence>
<xsd:attribute name="linkName" type="xsd-derived:NCName" use="optional" />
    </xsd:extension>
    </xsd:complexContent>
    </xsd:complexType>
<xsd:element name="transitionCondition" type="tCondition" />
<xsd:element name="assign" type="tAssign" />
- <xsd:complexType name="tAssign">
- <xsd:complexContent>
- <xsd:extension base="tActivity">
- <xsd:sequence>
- <xsd:choice maxOccurs="unbounded">
<xsd:element ref="copy" minOccurs="0" />
<xsd:element ref="extensionAssignOperation" minOccurs="0" />
    </xsd:choice>
    </xsd:sequence>
<xsd:attribute name="validate" type="tBoolean" use="optional" default="no" />
    </xsd:extension>
    </xsd:complexContent>
    </xsd:complexType>
<xsd:element name="copy" type="tCopy" />
- <xsd:complexType name="tCopy">
- <xsd:complexContent>
- <xsd:extension base="tExtensibleElements">
- <xsd:sequence>
<xsd:group ref="fromGroup" minOccurs="0" />
<xsd:element ref="to" minOccurs="0" />
```

```
        </xsd:sequence>
    <xsd:attribute name="keepSrcElementName" type="tBoolean" use="optional" default="no" />
    <xsd:attribute name="ignoreMissingFromData" type="tBoolean" use="optional" default="no" />
        </xsd:extension>
        </xsd:complexContent>
        </xsd:complexType>
  - <xsd:group name="fromGroup">
  - <xsd:choice>
    <xsd:element ref="opaqueFrom" />
    <xsd:element ref="from" />
        </xsd:choice>
        </xsd:group>
    <xsd:element name="opaqueFrom" type="tExtensibleElements" />
    <xsd:element name="from" type="tFrom" />
  - <xsd:complexType name="tFrom" mixed="true">
  - <xsd:sequence>
    <xsd:element ref="documentation" minOccurs="0" maxOccurs="unbounded" />
    <xsd:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded" />
  - <xsd:choice minOccurs="0">
    <xsd:element ref="literal" minOccurs="0" />
    <xsd:element ref="query" minOccurs="0" />
        </xsd:choice>
        </xsd:sequence>
    <xsd:attribute name="expressionLanguage" type="xsd-derived:anyURI" />
    <xsd:attribute name="variable" type="BPELVariableName" />
    <xsd:attribute name="part" type="xsd-derived:NCName" />
    <xsd:attribute name="property" type="xsd-derived:QName" />
    <xsd:attribute name="partnerLink" type="xsd-derived:NCName" />
    <xsd:attribute name="endpointReference" type="tRoles" />
    <xsd:attribute name="opaque" type="xsd-derived:tOpaqueBoolean" />
    <xsd:anyAttribute namespace="##other" processContents="lax" />
        </xsd:complexType>
    <xsd:element name="literal" type="tLiteral" />
  - <xsd:complexType name="tLiteral" mixed="true">
  - <xsd:sequence>
    <xsd:any namespace="##any" processContents="lax" minOccurs="0" maxOccurs="1" />
        </xsd:sequence>
        </xsd:complexType>
    <xsd:element name="query" type="tQuery" />
  - <xsd:complexType name="tQuery" mixed="true">
  - <xsd:sequence>
    <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded" />
        </xsd:sequence>
    <xsd:attribute name="queryLanguage" type="xsd-derived:anyURI" />
    <xsd:attribute name="opaque" type="xsd-derived:tOpaqueBoolean" />
    <xsd:anyAttribute namespace="##other" processContents="lax" />
        </xsd:complexType>
```

```
- <xsd:simpleType name="tRoles">
- <xsd:restriction base="xsd:string">
  <xsd:enumeration value="myRole" />
  <xsd:enumeration value="partnerRole" />
  <xsd:enumeration value="##opaque" />
      </xsd:restriction>
      </xsd:simpleType>
  <xsd:element name="to" type="tTo" />
- <xsd:complexType name="tTo" mixed="true">
- <xsd:sequence>
  <xsd:element ref="documentation" minOccurs="0" maxOccurs="unbounded" />
  <xsd:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded" />
  <xsd:element ref="query" minOccurs="0" />
      </xsd:sequence>
  <xsd:attribute name="opaque" type="xsd-derived:tOpaqueBoolean" />
  <xsd:attribute name="expressionLanguage" type="xsd-derived:anyURI" />
  <xsd:attribute name="variable" type="BPELVariableName" />
  <xsd:attribute name="part" type="xsd-derived:NCName" />
  <xsd:attribute name="property" type="xsd-derived:QName" />
  <xsd:attribute name="partnerLink" type="xsd-derived:NCName" />
  <xsd:anyAttribute namespace="##other" processContents="lax" />
      </xsd:complexType>
  <xsd:element name="extensionAssignOperation" type="tExtensionAssignOperation" />
- <xsd:complexType name="tExtensionAssignOperation">
- <xsd:complexContent>
  <xsd:extension base="tExtensibleElements" />
      </xsd:complexContent>
      </xsd:complexType>
  <xsd:element name="compensate" type="tCompensate" />
- <xsd:complexType name="tCompensate">
- <xsd:complexContent>
  <xsd:extension base="tActivity" />
      </xsd:complexContent>
      </xsd:complexType>
  <xsd:element name="compensateScope" type="tCompensateScope" />
- <xsd:complexType name="tCompensateScope">
- <xsd:complexContent>
- <xsd:extension base="tActivity">
  <xsd:attribute name="target" type="xsd-derived:NCName" use="optional" />
      </xsd:extension>
      </xsd:complexContent>
      </xsd:complexType>
  <xsd:element name="empty" type="tEmpty" />
- <xsd:complexType name="tEmpty">
- <xsd:complexContent>
  <xsd:extension base="tActivity" />
      </xsd:complexContent>
```

```
        </xsd:complexType>
  <xsd:element name="exit" type="tExit" />
- <xsd:complexType name="tExit">
- <xsd:complexContent>
  <xsd:extension base="tActivity" />
        </xsd:complexContent>
        </xsd:complexType>
  <xsd:element name="extensionActivity" type="tExtensionActivity" />
- <xsd:complexType name="tExtensionActivity">
- <xsd:sequence>
  <xsd:any namespace="##other" processContents="lax" />
        </xsd:sequence>
        </xsd:complexType>
  <xsd:element name="flow" type="tFlow" />
- <xsd:complexType name="tFlow">
- <xsd:complexContent>
- <xsd:extension base="tActivity">
- <xsd:sequence>
  <xsd:element ref="links" minOccurs="0" />
  <xsd:group ref="activity" minOccurs="0" maxOccurs="unbounded" />
        </xsd:sequence>
        </xsd:extension>
        </xsd:complexContent>
        </xsd:complexType>
  <xsd:element name="links" type="tLinks" />
- <xsd:complexType name="tLinks">
- <xsd:complexContent>
- <xsd:extension base="tExtensibleElements">
- <xsd:sequence>
  <xsd:element ref="link" minOccurs="0" maxOccurs="unbounded" />
        </xsd:sequence>
        </xsd:extension>
        </xsd:complexContent>
        </xsd:complexType>
  <xsd:element name="link" type="tLink" />
- <xsd:complexType name="tLink">
- <xsd:complexContent>
- <xsd:extension base="tExtensibleElements">
  <xsd:attribute name="name" type="xsd-derived:NCName" use="optional" />
        </xsd:extension>
        </xsd:complexContent>
        </xsd:complexType>
  <xsd:element name="forEach" type="tForEach" />
- <xsd:complexType name="tForEach">
- <xsd:complexContent>
- <xsd:extension base="tActivity">
- <xsd:sequence>
```

```
<xsd:element ref="startCounterValue" minOccurs="0" />
<xsd:element ref="finalCounterValue" minOccurs="0" />
<xsd:element ref="completionCondition" minOccurs="0" />
<xsd:element ref="scope" minOccurs="0" />
    </xsd:sequence>
<xsd:attribute name="counterName" type="BPELVariableName" use="optional" />
<xsd:attribute name="parallel" type="tBoolean" use="optional" />
    </xsd:extension>
    </xsd:complexContent>
    </xsd:complexType>
<xsd:element name="startCounterValue" type="tExpression" />
<xsd:element name="finalCounterValue" type="tExpression" />
<xsd:element name="completionCondition" type="tCompletionCondition" />
- <xsd:complexType name="tCompletionCondition">
- <xsd:complexContent>
- <xsd:extension base="tExtensibleElements">
- <xsd:sequence>
<xsd:element ref="branches" minOccurs="0" />
    </xsd:sequence>
    </xsd:extension>
    </xsd:complexContent>
    </xsd:complexType>
<xsd:element name="branches" type="tBranches" />
- <xsd:complexType name="tBranches">
- <xsd:complexContent>
- <xsd:extension base="tExpression">
<xsd:attribute name="successfulBranchesOnly" type="tBoolean" default="no" />
    </xsd:extension>
    </xsd:complexContent>
    </xsd:complexType>
<xsd:element name="if" type="tIf" />
- <xsd:complexType name="tIf">
- <xsd:complexContent>
- <xsd:extension base="tActivity">
- <xsd:sequence>
<xsd:element ref="condition" minOccurs="0" />
<xsd:group ref="activity" minOccurs="0" />
<xsd:element ref="elseif" minOccurs="0" maxOccurs="unbounded" />
<xsd:element ref="else" minOccurs="0" />
    </xsd:sequence>
    </xsd:extension>
    </xsd:complexContent>
    </xsd:complexType>
<xsd:element name="elseif" type="tElseif" />
- <xsd:complexType name="tElseif">
- <xsd:complexContent>
- <xsd:extension base="tExtensibleElements">
```

```
- <xsd:sequence>
  <xsd:element ref="condition" minOccurs="0" />
  <xsd:group ref="activity" minOccurs="0" />
      </xsd:sequence>
      </xsd:extension>
      </xsd:complexContent>
      </xsd:complexType>
  <xsd:element name="else" type="tActivityContainer" />
  <xsd:element name="invoke" type="tInvoke" />
- <xsd:complexType name="tInvoke">
- <xsd:annotation>
   <xsd:documentation>XSD Authors: The child element correlations needs to be a Local Element Declaration, because there is another
       correlations element defined for the non-invoke activities.</xsd:documentation>
      </xsd:annotation>
- <xsd:complexContent>
- <xsd:extension base="tActivity">
- <xsd:sequence>
  <xsd:element name="correlations" type="tCorrelationsWithPattern" minOccurs="0" />
  <xsd:element ref="catch" minOccurs="0" maxOccurs="unbounded" />
  <xsd:element ref="catchAll" minOccurs="0" />
  <xsd:element ref="compensationHandler" minOccurs="0" />
  <xsd:element ref="toParts" minOccurs="0" />
  <xsd:element ref="fromParts" minOccurs="0" />
      </xsd:sequence>
  <xsd:attribute name="partnerLink" type="xsd-derived:NCName" use="optional" />
  <xsd:attribute name="portType" type="xsd-derived:QName" use="optional" />
  <xsd:attribute name="operation" type="xsd-derived:NCName" use="optional" />
  <xsd:attribute name="inputVariable" type="BPELVariableName" use="optional" />
  <xsd:attribute name="outputVariable" type="BPELVariableName" use="optional" />
      </xsd:extension>
      </xsd:complexContent>
      </xsd:complexType>
- <xsd:complexType name="tCorrelationsWithPattern">
- <xsd:annotation>
   <xsd:documentation>XSD Authors: The child element correlation needs to be a Local Element Declaration, because there is another
       correlation element defined for the non-invoke activities.</xsd:documentation>
      </xsd:annotation>
- <xsd:complexContent>
- <xsd:extension base="tExtensibleElements">
- <xsd:sequence>
  <xsd:element name="correlation" type="tCorrelationWithPattern" minOccurs="0" maxOccurs="unbounded" />
      </xsd:sequence>
      </xsd:extension>
      </xsd:complexContent>
      </xsd:complexType>
- <xsd:complexType name="tCorrelationWithPattern">
- <xsd:complexContent>
```

```
- <xsd:extension base="tCorrelation">
  <xsd:attribute name="pattern" type="tPattern" />
      </xsd:extension>
      </xsd:complexContent>
      </xsd:complexType>
- <xsd:simpleType name="tPattern">
- <xsd:restriction base="xsd:string">
  <xsd:enumeration value="request" />
  <xsd:enumeration value="response" />
  <xsd:enumeration value="request-response" />
  <xsd:enumeration value="##opaque" />
      </xsd:restriction>
      </xsd:simpleType>
  <xsd:element name="fromParts" type="tFromParts" />
- <xsd:complexType name="tFromParts">
- <xsd:complexContent>
- <xsd:extension base="tExtensibleElements">
- <xsd:sequence>
  <xsd:element ref="fromPart" minOccurs="0" maxOccurs="unbounded" />
      </xsd:sequence>
      </xsd:extension>
      </xsd:complexContent>
      </xsd:complexType>
  <xsd:element name="fromPart" type="tFromPart" />
- <xsd:complexType name="tFromPart">
- <xsd:complexContent>
- <xsd:extension base="tExtensibleElements">
  <xsd:attribute name="part" type="xsd-derived:NCName" use="optional" />
  <xsd:attribute name="toVariable" type="BPELVariableName" use="optional" />
      </xsd:extension>
      </xsd:complexContent>
      </xsd:complexType>
  <xsd:element name="toParts" type="tToParts" />
- <xsd:complexType name="tToParts">
- <xsd:complexContent>
- <xsd:extension base="tExtensibleElements">
- <xsd:sequence>
  <xsd:element ref="toPart" minOccurs="0" maxOccurs="unbounded" />
      </xsd:sequence>
      </xsd:extension>
      </xsd:complexContent>
      </xsd:complexType>
  <xsd:element name="toPart" type="tToPart" />
- <xsd:complexType name="tToPart">
- <xsd:complexContent>
- <xsd:extension base="tExtensibleElements">
  <xsd:attribute name="part" type="xsd-derived:NCName" use="optional" />
```

```
<xsd:attribute name="fromVariable" type="BPELVariableName" use="optional" />
    </xsd:extension>
    </xsd:complexContent>
    </xsd:complexType>
<xsd:element name="pick" type="tPick" />
- <xsd:complexType name="tPick">
- <xsd:annotation>
<xsd:documentation>XSD Authors: The child element onAlarm needs to be a Local Element Declaration, because there is another onAlarm
      element defined for event handlers.</xsd:documentation>
    </xsd:annotation>
- <xsd:complexContent>
- <xsd:extension base="tActivity">
- <xsd:sequence>
<xsd:element ref="onMessage" minOccurs="0" maxOccurs="unbounded" />
<xsd:element name="onAlarm" type="tOnAlarmPick" minOccurs="0" maxOccurs="unbounded" />
    </xsd:sequence>
<xsd:attribute name="createInstance" type="tBoolean" default="no" />
    </xsd:extension>
    </xsd:complexContent>
    </xsd:complexType>
<xsd:element name="onMessage" type="tOnMessage" />
- <xsd:complexType name="tOnMessage">
- <xsd:complexContent>
- <xsd:extension base="tOnMsgCommon">
- <xsd:sequence>
<xsd:group ref="activity" minOccurs="0" />
    </xsd:sequence>
    </xsd:extension>
    </xsd:complexContent>
    </xsd:complexType>
- <xsd:complexType name="tOnAlarmPick">
- <xsd:complexContent>
- <xsd:extension base="tExtensibleElements">
- <xsd:sequence>
<xsd:group ref="forOrUntilGroup" minOccurs="0" />
<xsd:group ref="activity" minOccurs="0" />
    </xsd:sequence>
    </xsd:extension>
    </xsd:complexContent>
    </xsd:complexType>
<xsd:element name="receive" type="tReceive" />
- <xsd:complexType name="tReceive">
- <xsd:annotation>
<xsd:documentation>XSD Authors: The child element correlations needs to be a Local Element Declaration, because there is another
      correlations element defined for the invoke activity.</xsd:documentation>
    </xsd:annotation>
- <xsd:complexContent>
```

```
- <xsd:extension base="tActivity">
- <xsd:sequence>
  <xsd:element name="correlations" type="tCorrelations" minOccurs="0" />
  <xsd:element ref="fromParts" minOccurs="0" />
      </xsd:sequence>
  <xsd:attribute name="partnerLink" type="xsd-derived:NCName" use="optional" />
  <xsd:attribute name="portType" type="xsd-derived:QName" use="optional" />
  <xsd:attribute name="operation" type="xsd-derived:NCName" use="optional" />
  <xsd:attribute name="variable" type="BPELVariableName" use="optional" />
  <xsd:attribute name="createInstance" type="tBoolean" default="no" />
  <xsd:attribute name="messageExchange" type="xsd-derived:NCName" use="optional" />
      </xsd:extension>
      </xsd:complexContent>
      </xsd:complexType>
  <xsd:element name="repeatUntil" type="tRepeatUntil" />
- <xsd:complexType name="tRepeatUntil">
- <xsd:complexContent>
- <xsd:extension base="tActivity">
- <xsd:sequence>
  <xsd:group ref="activity" minOccurs="0" />
  <xsd:element ref="condition" minOccurs="0" />
      </xsd:sequence>
      </xsd:extension>
      </xsd:complexContent>
      </xsd:complexType>
  <xsd:element name="reply" type="tReply" />
- <xsd:complexType name="tReply">
- <xsd:annotation>
   <xsd:documentation>XSD Authors: The child element correlations needs to be a Local Element Declaration, because there is another
        correlations element defined for the invoke activity.</xsd:documentation>
      </xsd:annotation>
- <xsd:complexContent>
- <xsd:extension base="tActivity">
- <xsd:sequence>
  <xsd:element name="correlations" type="tCorrelations" minOccurs="0" />
  <xsd:element ref="toParts" minOccurs="0" />
      </xsd:sequence>
  <xsd:attribute name="partnerLink" type="xsd-derived:NCName" use="optional" />
  <xsd:attribute name="portType" type="xsd-derived:QName" use="optional" />
  <xsd:attribute name="operation" type="xsd-derived:NCName" use="optional" />
  <xsd:attribute name="variable" type="BPELVariableName" use="optional" />
  <xsd:attribute name="faultName" type="xsd-derived:QName" />
  <xsd:attribute name="messageExchange" type="xsd-derived:NCName" use="optional" />
      </xsd:extension>
      </xsd:complexContent>
      </xsd:complexType>
  <xsd:element name="rethrow" type="tRethrow" />
```

```
- <xsd:complexType name="tRethrow">
- <xsd:complexContent>
  <xsd:extension base="tActivity" />
      </xsd:complexContent>
      </xsd:complexType>
  <xsd:element name="scope" type="tScope" />
- <xsd:complexType name="tScope">
- <xsd:annotation>
  <xsd:documentation>There is no schema-level default for "exitOnStandardFault" at "scope". Because, it will inherit default from enclosing
        scope or process.</xsd:documentation>
      </xsd:annotation>
- <xsd:complexContent>
- <xsd:extension base="tActivity">
- <xsd:sequence>
  <xsd:element ref="partnerLinks" minOccurs="0" />
  <xsd:element ref="messageExchanges" minOccurs="0" />
  <xsd:element ref="variables" minOccurs="0" />
  <xsd:element ref="correlationSets" minOccurs="0" />
  <xsd:element ref="faultHandlers" minOccurs="0" />
  <xsd:element ref="compensationHandler" minOccurs="0" />
  <xsd:element ref="terminationHandler" minOccurs="0" />
  <xsd:element ref="eventHandlers" minOccurs="0" />
  <xsd:group ref="activity" minOccurs="0" />
      </xsd:sequence>
  <xsd:attribute name="isolated" type="tBoolean" default="no" />
  <xsd:attribute name="exitOnStandardFault" type="tBoolean" />
      </xsd:extension>
      </xsd:complexContent>
      </xsd:complexType>
- <xsd:element name="compensationHandler" type="tActivityContainer">
- <xsd:annotation>
  <xsd:documentation>This element can contain all activities including the activities compensate and compensateScope.</xsd:documentation>
      </xsd:annotation>
      </xsd:element>
- <xsd:element name="terminationHandler" type="tActivityContainer">
- <xsd:annotation>
  <xsd:documentation>This element can contain all activities including the activities compensate and compensateScope.</xsd:documentation>
      </xsd:annotation>
      </xsd:element>
  <xsd:element name="sequence" type="tSequence" />
- <xsd:complexType name="tSequence">
- <xsd:complexContent>
- <xsd:extension base="tActivity">
- <xsd:sequence>
  <xsd:group ref="activity" minOccurs="0" maxOccurs="unbounded" />
      </xsd:sequence>
      </xsd:extension>
```

```
        </xsd:complexContent>
        </xsd:complexType>
  <xsd:element name="throw" type="tThrow" />
- <xsd:complexType name="tThrow">
- <xsd:complexContent>
- <xsd:extension base="tActivity">
  <xsd:attribute name="faultName" type="xsd-derived:QName" use="optional" />
  <xsd:attribute name="faultVariable" type="BPELVariableName" />
        </xsd:extension>
        </xsd:complexContent>
        </xsd:complexType>
  <xsd:element name="validate" type="tValidate" />
- <xsd:complexType name="tValidate">
- <xsd:complexContent>
- <xsd:extension base="tActivity">
  <xsd:attribute name="variables" use="optional" type="BPELVariableNames" />
        </xsd:extension>
        </xsd:complexContent>
        </xsd:complexType>
- <xsd:simpleType name="BPELVariableNames">
- <xsd:restriction>
- <xsd:simpleType>
  <xsd:list itemType="BPELVariableName" />
        </xsd:simpleType>
  <xsd:minLength value="1" />
        </xsd:restriction>
        </xsd:simpleType>
  <xsd:element name="wait" type="tWait" />
- <xsd:complexType name="tWait">
- <xsd:complexContent>
- <xsd:extension base="tActivity">
- <xsd:choice>
  <xsd:element ref="for" minOccurs="0" />
  <xsd:element ref="until" minOccurs="0" />
        </xsd:choice>
        </xsd:extension>
        </xsd:complexContent>
        </xsd:complexType>
  <xsd:element name="while" type="tWhile" />
- <xsd:complexType name="tWhile">
- <xsd:complexContent>
- <xsd:extension base="tActivity">
- <xsd:sequence>
  <xsd:element ref="condition" minOccurs="0" />
  <xsd:group ref="activity" minOccurs="0" />
        </xsd:sequence>
        </xsd:extension>
```

```
        </xsd:complexContent>
        </xsd:complexType>
- <xsd:complexType name="tExpression" mixed="true">
- <xsd:sequence>
  <xsd:any minOccurs="0" maxOccurs="unbounded" processContents="lax" />
        </xsd:sequence>
  <xsd:attribute name="expressionLanguage" type="xsd-derived:anyURI" />
  <xsd:attribute name="opaque" type="xsd-derived:tOpaqueBoolean" />
  <xsd:anyAttribute namespace="##other" processContents="lax" />
        </xsd:complexType>
- <xsd:complexType name="tCondition">
- <xsd:complexContent mixed="true">
  <xsd:extension base="tExpression" />
        </xsd:complexContent>
        </xsd:complexType>
  <xsd:element name="condition" type="tBoolean-expr" />
- <xsd:complexType name="tBoolean-expr">
- <xsd:complexContent mixed="true">
  <xsd:extension base="tExpression" />
        </xsd:complexContent>
        </xsd:complexType>
- <xsd:complexType name="tDuration-expr">
- <xsd:complexContent mixed="true">
  <xsd:extension base="tExpression" />
        </xsd:complexContent>
        </xsd:complexType>
- <xsd:complexType name="tDeadline-expr">
- <xsd:complexContent mixed="true">
  <xsd:extension base="tExpression" />
        </xsd:complexContent>
        </xsd:complexType>
- <xsd:simpleType name="tBoolean">
- <xsd:restriction base="xsd:string">
  <xsd:enumeration value="yes" />
  <xsd:enumeration value="no" />
  <xsd:enumeration value="##opaque" />
        </xsd:restriction>
        </xsd:simpleType>
      - <!--
       SCHEMA NOTE: new types and element introduced for Abstract WS-BPEL
       -->
- <xsd:simpleType name="tOpaqueStr">
- <xsd:restriction base="xsd:string">
  <xsd:enumeration value="##opaque" />
        </xsd:restriction>
        </xsd:simpleType>
- <xsd:simpleType name="QName">
```

```
  <xsd:union memberTypes="xsd:QName tOpaqueStr" />
      </xsd:simpleType>
- <xsd:simpleType name="NCName">
  <xsd:union memberTypes="xsd:NCName tOpaqueStr" />
      </xsd:simpleType>
- <xsd:simpleType name="anyURI">
  <xsd:union memberTypes="xsd:anyURI tOpaqueStr" />
      </xsd:simpleType>
- <xsd:simpleType name="tOpaqueBoolean">
- <xsd:restriction base="xsd:string">
  <xsd:enumeration value="yes" />
      </xsd:restriction>
      </xsd:simpleType>
  <xsd:element name="opaqueActivity" type="tOpaqueActivity" />
- <xsd:complexType name="tOpaqueActivity">
- <xsd:complexContent>
  <xsd:extension base="tActivity" />
      </xsd:complexContent>
      </xsd:complexType>
      </xsd:schema>
```

A.2 Esquema ebXML BPSS Para Procesos De Negocios

```
<?xml version="1.0" encoding="UTF-8" ?>
- <!--
  Metadata: Owner: ebxml-bp (OASIS ebXML Business Process TC)
  Product: ebxmlbp (aka ebBP)
  Product Version: 2.0.4
  Artifact Type: Schema
  Stage: os (OASIS Standard)
  Descriptive Name: None required
  Revision: None
  Language: en (English)
  Form: xsd (schema)
  Date: 20061221 (21 December 2006)
-->
- <!--
  OASIS takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on OASIS's procedures with respect to rights in OASIS specifications can be found at the OASIS website. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementors or users of this specification, can be obtained from the OASIS Executive Director.
  OASIS invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to implement this specification. Please address the information to the OASIS Executive Director.
  Copyright © OASIS Open 2005, 2006. All Rights Reserved.
  This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to OASIS, except as needed for the purpose of developing OASIS specifications, in which case the procedures for copyrights defined in the OASIS Intellectual Property Rights document must be followed, or as required to translate it into languages other than English.
  The limited permissions granted above are perpetual and will not be revoked by OASIS or its successors or assigns.
  This document and the information contained herein is provided on an "AS IS" basis and OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.
-->
- <xsd:schema xmlns="http://docs.oasis-open.org/ebxml-bp/ebbp-2.0" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xi="http://www.w3.org/2001/XInclude" targetNamespace="http://docs.oasis-open.org/ebxml-bp/ebbp-2.0"
  elementFormDefault="qualified" attributeFormDefault="unqualified">
 <xsd:import namespace="http://www.w3.org/XML/1998/namespace" schemaLocation="http://www.w3.org/2001/xml.xsd" />
- <xsd:element name="ProcessSpecification">
- <xsd:annotation>
  <xsd:documentation>Root element of a Process Specification document that has a globally unique identity. The Process Specification element can specify the version of the technical specification used and the process instance version related to the target ebBP (schema).</xsd:documentation>
  </xsd:annotation>
- <xsd:complexType>
- <xsd:complexContent>
- <xsd:extension base="ProcessSpecificationType">
- <xsd:attribute name="specificationVersion" type="xsd:NMTOKEN" use="optional">
- <xsd:annotation>
```

```
<xsd:documentation>Is the technical specification version of the Process Specification. Note: This attribute was added in v2.0.</xsd:documentation>
</xsd:annotation>
</xsd:attribute>
- <xsd:attribute name="instanceVersion" type="xsd:string" use="optional">
- <xsd:annotation>
<xsd:documentation>Is the version of the Process Specification or artifact instance. An example would be the Australian Wheat Board v2.1. Note: This attribute was added in v2.0.</xsd:documentation>
</xsd:annotation>
</xsd:attribute>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
- <xsd:unique name="ProcessSpecification-ID">
<xsd:selector xpath="." />
<xsd:field xpath="nameID" />
</xsd:unique>
</xsd:element>
- <xsd:complexType name="ProcessSpecificationType">
- <xsd:annotation>
<xsd:documentation>Type for the root element of a Process Specification document.</xsd:documentation>
</xsd:annotation>
- <xsd:sequence>
<xsd:element ref="Documentation" minOccurs="0" maxOccurs="unbounded" />
<xsd:element ref="AttributeSubstitution" minOccurs="0" maxOccurs="unbounded" />
<xsd:element ref="ExternalRoles" minOccurs="0" />
<xsd:element ref="Signal" minOccurs="0" maxOccurs="unbounded" />
<xsd:element ref="Variable" minOccurs="0" maxOccurs="unbounded" />
- <xsd:choice minOccurs="0" maxOccurs="unbounded">
<xsd:element ref="Package" minOccurs="0" maxOccurs="unbounded" />
<xsd:element ref="BusinessDocument" minOccurs="0" maxOccurs="unbounded" />
<xsd:element ref="BusinessTransactionHead" minOccurs="0" maxOccurs="unbounded" />
<xsd:element ref="BinaryCollaboration" minOccurs="0" maxOccurs="unbounded" />
<xsd:element ref="MultiPartyCollaboration" minOccurs="0" maxOccurs="unbounded" />
<xsd:element ref="BusinessCollaboration" minOccurs="0" maxOccurs="unbounded" />
<xsd:element ref="OperationMapping" minOccurs="0" maxOccurs="unbounded" />
</xsd:choice>
</xsd:sequence>
<xsd:attributeGroup ref="name" />
- <xsd:attribute name="uuid" type="xsd:string" use="required">
- <xsd:annotation>
<xsd:documentation>Defines a string identification mechanism for a Process Specificiation. The uuid is not used for the purpose of versioning, so that even a change introduced by AttributeSubstitution (to business documents' schemas, for example), would be marked by a new uuid.</xsd:documentation>
</xsd:annotation>
</xsd:attribute>
</xsd:complexType>
```

```
- <xsd:element name="Package">
- <xsd:annotation>
  <xsd:documentation>Defines a hierarchical name scope containing reusable elements.</xsd:documentation>
      </xsd:annotation>
- <xsd:complexType>
- <xsd:complexContent>
  <xsd:extension base="PackageType" />
      </xsd:complexContent>
      </xsd:complexType>
- <xsd:unique name="Package-ID">
  <xsd:selector xpath="." />
  <xsd:field xpath="nameID" />
      </xsd:unique>
      </xsd:element>
- <xsd:complexType name="PackageType">
- <xsd:annotation>
  <xsd:documentation>Type for a hierarchical name scope containing reusable elements.</xsd:documentation>
      </xsd:annotation>
- <xsd:choice maxOccurs="unbounded">
  <xsd:element ref="Documentation" minOccurs="0" maxOccurs="unbounded" />
  <xsd:element ref="AttributeSubstitution" minOccurs="0" maxOccurs="unbounded" />
  <xsd:element ref="ExternalRoles" minOccurs="0" />
  <xsd:element ref="Signal" minOccurs="0" maxOccurs="unbounded" />
  <xsd:element ref="Variable" minOccurs="0" maxOccurs="unbounded" />
- <xsd:choice minOccurs="0" maxOccurs="unbounded">
  <xsd:element ref="Package" minOccurs="0" maxOccurs="unbounded" />
  <xsd:element ref="BusinessDocument" minOccurs="0" maxOccurs="unbounded" />
  <xsd:element ref="BusinessTransactionHead" minOccurs="0" maxOccurs="unbounded" />
  <xsd:element ref="BinaryCollaboration" minOccurs="0" maxOccurs="unbounded" />
  <xsd:element ref="MultiPartyCollaboration" minOccurs="0" maxOccurs="unbounded" />
  <xsd:element ref="BusinessCollaboration" minOccurs="0" maxOccurs="unbounded" />
  <xsd:element ref="OperationMapping" minOccurs="0" />
      </xsd:choice>
      </xsd:choice>
  <xsd:attributeGroup ref="name" />
- <xsd:attribute name="parentRef" type="xsd:IDREF" use="optional">
- <xsd:annotation>
  <xsd:documentation>Defines the nameID reference for a Package.</xsd:documentation>
      </xsd:annotation>
      </xsd:attribute>
      </xsd:complexType>
- <xsd:element name="Documentation" type="DocumentationType">
- <xsd:annotation>
  <xsd:documentation>Defines user documentation for any element. Must be the first element of its container. Note: The xml:lang was added
      in v2.0.</xsd:documentation>
      </xsd:annotation>
      </xsd:element>
```

```
- <xsd:complexType name="DocumentationType">
- <xsd:annotation>
  <xsd:documentation>Type for the user documentation for any element.</xsd:documentation>
      </xsd:annotation>
- <xsd:simpleContent>
- <xsd:extension base="xsd:string">
- <xsd:attribute name="uri" type="xsd:anyURI">
- <xsd:annotation>
  <xsd:documentation>Defines the address of the Documentation object. A URL can be a URI.</xsd:documentation>
      </xsd:annotation>
      </xsd:attribute>
  <xsd:attribute ref="xml:lang" />
      </xsd:extension>
      </xsd:simpleContent>
      </xsd:complexType>
    - <!--
     AttributeSubstitution allowed to "edit" references (IDREFS)  or other attribute values.
      -->
- <xsd:element name="AttributeSubstitution">
- <xsd:annotation>
  <xsd:documentation>Attribute or document value should be used in place of some value in an existing Process Specification. Attribute
        substitution could be used for document substitution. These substititution changes were made in v2.0.</xsd:documentation>
      </xsd:annotation>
- <xsd:complexType>
- <xsd:complexContent>
  <xsd:extension base="AttributeSubstitutionType" />
      </xsd:complexContent>
      </xsd:complexType>
      </xsd:element>
- <xsd:complexType name="AttributeSubstitutionType">
- <xsd:annotation>
  <xsd:documentation>Type for the attribute or document value used for substitution.</xsd:documentation>
      </xsd:annotation>
- <xsd:sequence>
  <xsd:element ref="Documentation" minOccurs="0" maxOccurs="unbounded" />
      </xsd:sequence>
- <xsd:attribute name="nameIDRef" type="xsd:IDREF" use="required">
- <xsd:annotation>
  <xsd:documentation>Is the nameID reference to the Documentation related to a particular element.</xsd:documentation>
      </xsd:annotation>
      </xsd:attribute>
- <xsd:attribute name="attributeName" type="xsd:NMTOKEN" use="required">
- <xsd:annotation>
  <xsd:documentation>Is the name of an attribute of any element within the scope of the substitution set.</xsd:documentation>
      </xsd:annotation>
      </xsd:attribute>
- <xsd:attribute name="value" type="xsd:string" use="required">
```

```
- <xsd:annotation>
  <xsd:documentation>Is the value, which shall replace the current value of the attribute.</xsd:documentation>
      </xsd:annotation>
      </xsd:attribute>
      </xsd:complexType>
            - <!--
            Include element is replaced by XInclude's include element.
                  This include element is not imported into our schema, but should
                  be processed (and replaced by the referenced XML) prior to validation of instances
      -->
- <xsd:element name="ExternalRoles" type="ExternalRolesType">
- <xsd:annotation>
  <xsd:documentation>External role element maps to the actual roles used in a Business Collaboration (for example, an external role maps to a
        Business Collaboration). Note: This element was added in v2.0.</xsd:documentation>
      </xsd:annotation>
      </xsd:element>
- <xsd:complexType name="ExternalRolesType">
- <xsd:annotation>
  <xsd:documentation>Types for the external role that maps to actual roles in a Business Collaboration. Performs elements are needed when
        the values of Roles declared in a Business Collaboration differ from the values declared with the @name attribute. Note: This
        complexType was added in v2.0.</xsd:documentation>
      </xsd:annotation>
- <xsd:sequence>
- <xsd:element name="BusinessPartnerRole" minOccurs="2" maxOccurs="unbounded">
- <xsd:annotation>
  <xsd:documentation>Each business partner plays one or more abstract partner roles in the Business Collaboration.</xsd:documentation>
      </xsd:annotation>
- <xsd:complexType>
- <xsd:sequence>
  <xsd:element ref="Documentation" minOccurs="0" />
  <xsd:element name="Performs" type="PerformsType" minOccurs="0" maxOccurs="unbounded" />
      </xsd:sequence>
  <xsd:attributeGroup ref="name" />
      </xsd:complexType>
      </xsd:element>
      </xsd:sequence>
      </xsd:complexType>
- <xsd:group name="collaborationGroup">
- <xsd:annotation>
  <xsd:documentation>The group that includes the various types of Collaborations. Note: The Business Collaboration will replace the Binary
        and MultiParty Collaboration in a future version. Note: This group was added in v2.0.</xsd:documentation>
      </xsd:annotation>
- <xsd:choice>
  <xsd:element ref="BusinessTransactionActivity" minOccurs="0" />
  <xsd:element ref="CollaborationActivity" minOccurs="0" />
  <xsd:element ref="ComplexBusinessTransactionActivity" minOccurs="0" />
  <xsd:element ref="Success" minOccurs="0" />
  <xsd:element ref="Failure" minOccurs="0" />
```

```
<xsd:element ref="Transition" minOccurs="0" />
<xsd:element ref="Fork" minOccurs="0" />
<xsd:element ref="Join" minOccurs="0" />
<xsd:element ref="Decision" minOccurs="0" />
    </xsd:choice>
    </xsd:group>
- <xsd:element name="BinaryCollaboration" type="BinaryCollaborationType">
- <xsd:annotation>
 <xsd:documentation>Two roles - Defines the interaction between two top-level or abstract partner roles. Binary Collaboration is a
      choreographed state of two Business Collaboration roles. This Business Collaboration is being deprecated.</xsd:documentation>
    </xsd:annotation>
- <xsd:unique name="BinaryCollaboration-ID">
<xsd:selector xpath="." />
<xsd:field xpath="nameID" />
    </xsd:unique>
- <xsd:unique name="BinaryCollaborationRole-ID">
<xsd:selector xpath=".//Role" />
<xsd:field xpath="nameID" />
    </xsd:unique>
    </xsd:element>
- <xsd:complexType name="BinaryCollaborationType">
- <xsd:annotation>
<xsd:documentation>The type related to Binary Collaboration.</xsd:documentation>
    </xsd:annotation>
- <xsd:sequence>
<xsd:element ref="Documentation" minOccurs="0" maxOccurs="unbounded" />
<xsd:element name="Role" type="RoleType" minOccurs="2" maxOccurs="2" />
<xsd:element ref="TimeToPerform" />
<xsd:element ref="Start" />
<xsd:element name="BeginsWhen" type="ConditionExpressionType" minOccurs="0" maxOccurs="unbounded" />
<xsd:element name="PreCondition" type="ConditionExpressionType" minOccurs="0" maxOccurs="unbounded" />
<xsd:element name="PostCondition" type="ConditionExpressionType" minOccurs="0" maxOccurs="unbounded" />
<xsd:element name="EndsWhen" type="ConditionExpressionType" minOccurs="0" maxOccurs="unbounded" />
<xsd:group ref="collaborationGroup" maxOccurs="unbounded" />
    </xsd:sequence>
<xsd:attributeGroup ref="name" />
- <xsd:attribute name="pattern" type="xsd:anyURI">
- <xsd:annotation>
<xsd:documentation>May point to the pattern on which an activity or collaboration is based. This attribute is used only when not using the
      concrete BT patterns provided.</xsd:documentation>
    </xsd:annotation>
    </xsd:attribute>
- <xsd:attribute name="isInnerCollaboration" type="xsd:boolean" default="false">
- <xsd:annotation>
<xsd:documentation>Indicates whether or not this Business Collaboration definition can only be used within a Collaboration Activity (as a
      sub collaboration) or initiated directly by a party.</xsd:documentation>
    </xsd:annotation>
```

```
    </xsd:attribute>
    </xsd:complexType>
- <xsd:element name="MultiPartyCollaboration" type="MultiPartyCollaborationType">
- <xsd:annotation>
  <xsd:documentation>More than two roles - Defines the interaction between more than two top-level or abstract partner roles. Binary Collaboration is a choreographed state of more than two Business Collaboration roles. This collaboration is being deprecated.</xsd:documentation>
    </xsd:annotation>
    </xsd:element>
- <xsd:complexType name="MultiPartyCollaborationType">
- <xsd:annotation>
  <xsd:documentation>The type related to MultiParty Collaboration.</xsd:documentation>
    </xsd:annotation>
- <xsd:sequence>
  <xsd:element ref="Documentation" minOccurs="0" maxOccurs="unbounded" />
  <xsd:element name="Role" type="RoleType" minOccurs="3" maxOccurs="unbounded" />
  <xsd:element ref="TimeToPerform" />
  <xsd:element ref="Start" />
  <xsd:element name="BeginsWhen" type="ConditionExpressionType" minOccurs="0" maxOccurs="unbounded" />
  <xsd:element name="PreCondition" type="ConditionExpressionType" minOccurs="0" maxOccurs="unbounded" />
  <xsd:element name="PostCondition" type="ConditionExpressionType" minOccurs="0" maxOccurs="unbounded" />
  <xsd:element name="EndsWhen" type="ConditionExpressionType" minOccurs="0" maxOccurs="unbounded" />
  <xsd:group ref="collaborationGroup" maxOccurs="unbounded" />
    </xsd:sequence>
  <xsd:attributeGroup ref="name" />
- <xsd:attribute name="pattern" type="xsd:anyURI">
- <xsd:annotation>
  <xsd:documentation>May point to the pattern on which an activity or collaboration is based. This attribute is used only when not using the concrete BT patterns provided.</xsd:documentation>
    </xsd:annotation>
    </xsd:attribute>
- <xsd:attribute name="isInnerCollaboration" type="xsd:boolean" default="false">
- <xsd:annotation>
  <xsd:documentation>Indicates whether or not this Business Collaboration definition can only be used within a Business Collaboration activity (as a sub collaboration) or initiated directly by a party.</xsd:documentation>
    </xsd:annotation>
    </xsd:attribute>
    </xsd:complexType>
- <xsd:element name="BusinessCollaboration" type="BusinessCollaborationType">
- <xsd:annotation>
  <xsd:documentation>Two or more roles - Two or more roles - Defines the interaction between two or more top-level or abstract partner roles. Binary Collaboration is a choreographed state of two or more Business Collaboration roles. This element will replace Binary and MultiParty Collaboration elements in a future version.</xsd:documentation>
    </xsd:annotation>
    </xsd:element>
- <xsd:complexType name="BusinessCollaborationType">
- <xsd:annotation>
```

```
<xsd:documentation>The type related to Business Collaboration.</xsd:documentation>
    </xsd:annotation>
- <xsd:sequence>
<xsd:element ref="Documentation" minOccurs="0" maxOccurs="unbounded" />
<xsd:element name="Role" type="RoleType" minOccurs="2" maxOccurs="unbounded" />
<xsd:element ref="TimeToPerform" />
<xsd:element ref="Start" />
<xsd:element name="BeginsWhen" type="ConditionExpressionType" minOccurs="0" maxOccurs="unbounded" />
<xsd:element name="PreCondition" type="ConditionExpressionType" minOccurs="0" maxOccurs="unbounded" />
<xsd:element name="PostCondition" type="ConditionExpressionType" minOccurs="0" maxOccurs="unbounded" />
<xsd:element name="EndsWhen" type="ConditionExpressionType" minOccurs="0" maxOccurs="unbounded" />
<xsd:group ref="collaborationGroup" maxOccurs="unbounded" />
    </xsd:sequence>
<xsd:attributeGroup ref="name" />
- <xsd:attribute name="pattern" type="xsd:anyURI">
- <xsd:annotation>
<xsd:documentation>May point to the pattern on which an activity or collaboration is based. This attribute is used only when not using the
      concrete BT patterns provided.</xsd:documentation>
    </xsd:annotation>
    </xsd:attribute>
- <xsd:attribute name="isInnerCollaboration" type="xsd:boolean" default="false">
- <xsd:annotation>
<xsd:documentation>Indicates whether or not this Business Collaboration definition can only be used within a Collaboration Activity (as a
      sub collaboration) or initiated directly by a party.</xsd:documentation>
    </xsd:annotation>
    </xsd:attribute>
    </xsd:complexType>
- <xsd:element name="DocumentEnvelope" type="DocumentEnvelopeType">
- <xsd:annotation>
<xsd:documentation>Conveys business information between two roles in a business transaction. One document envelope conveys the
      request from the Requesting to the Responding role and another the response from the Responding role back to the Requesting one
      (where applicable).</xsd:documentation>
    </xsd:annotation>
    </xsd:element>
- <xsd:complexType name="DocumentEnvelopeType">
- <xsd:annotation>
<xsd:documentation>The Type related to envelope that conveys business information.</xsd:documentation>
    </xsd:annotation>
- <xsd:sequence>
<xsd:element ref="Documentation" minOccurs="0" maxOccurs="unbounded" />
<xsd:element ref="Attachment" minOccurs="0" maxOccurs="unbounded" />
    </xsd:sequence>
<xsd:attributeGroup ref="name" />
- <xsd:attribute name="businessDocumentRef" type="xsd:IDREF" use="required">
- <xsd:annotation>
<xsd:documentation>Indicates the nameID reference to the logical business document.</xsd:documentation>
    </xsd:annotation>
```

```
    </xsd:attribute>
- <xsd:attribute name="isPositiveResponse" type="xsd:boolean">
- <xsd:annotation>
  <xsd:documentation>May evaluate to TRUE or FALSE. If TRUE, the DocumentEnvelope is intended as a positive response to a request. The value for this parameter is used to evaluate a Business Success or Failure of the corresponding Business Transaction.</xsd:documentation>
    </xsd:annotation>
    </xsd:attribute>
  <xsd:attributeGroup ref="documentSecurity" />
    </xsd:complexType>
- <xsd:element name="BusinessDocument" type="BusinessDocumentType">
- <xsd:annotation>
  <xsd:documentation>A generic name of a document. A Business Document may have 0..n Condition Expressions.</xsd:documentation>
    </xsd:annotation>
    </xsd:element>
- <xsd:complexType name="BusinessDocumentType">
- <xsd:annotation>
  <xsd:documentation>The type related to a Business Document.</xsd:documentation>
    </xsd:annotation>
- <xsd:sequence>
  <xsd:element ref="Documentation" minOccurs="0" maxOccurs="unbounded" />
  <xsd:element ref="ConditionExpression" minOccurs="0" maxOccurs="unbounded" />
  <xsd:element ref="Specification" maxOccurs="unbounded" />
    </xsd:sequence>
  <xsd:attributeGroup ref="name" />
    </xsd:complexType>
- <xsd:element name="Attachment">
- <xsd:annotation>
  <xsd:documentation>An optional unstructured document associated with a Business Document.</xsd:documentation>
    </xsd:annotation>
- <xsd:complexType>
- <xsd:sequence>
  <xsd:element ref="Documentation" minOccurs="0" maxOccurs="unbounded" />
  <xsd:element ref="Specification" maxOccurs="unbounded" />
    </xsd:sequence>
  <xsd:attributeGroup ref="name" />
  <xsd:attributeGroup ref="documentSecurity" />
- <xsd:attribute name="businessDocumentRef" type="xsd:IDREF" use="required">
- <xsd:annotation>
  <xsd:documentation>Indicates the nameID reference to the logical business document.</xsd:documentation>
    </xsd:annotation>
    </xsd:attribute>
- <xsd:attribute name="mimeType" type="xsd:string" use="optional">
- <xsd:annotation>
    <xsd:documentation>Defines the valid MIME (Multipurpose Internet Mail Extensions) type of this Attachment. Example: 'application/pdf'.</xsd:documentation>
    </xsd:annotation>
```

```
    </xsd:attribute>
- <xsd:attribute name="minOccurs" type="xsd:integer" use="optional">
- <xsd:annotation>
  <xsd:documentation>Defines the minimum occurrences of an Attachment. Note: This attribute was added in v2.0.</xsd:documentation>
    </xsd:annotation>
    </xsd:attribute>
- <xsd:attribute name="maxOccurs" type="xsd:integer" use="optional">
- <xsd:annotation>
  <xsd:documentation>Defines the maximum occurrences of an Attachment. Note: This attribute was added in v2.0.</xsd:documentation>
    </xsd:annotation>
    </xsd:attribute>
    </xsd:complexType>
- <xsd:unique name="Attachment-ID">
  <xsd:selector xpath="." />
  <xsd:field xpath="nameID" />
    </xsd:unique>
    </xsd:element>
- <xsd:element name="Specification">
- <xsd:annotation>
  <xsd:documentation>A specification element that can associate many references to a particular ebBP element. For example, multiple
      specifications associated with a logical Business Document. Note: This element was added in v2.0.</xsd:documentation>
    </xsd:annotation>
- <xsd:complexType>
- <xsd:attribute name="type" type="DocumentSpecificationType" use="optional" default="schema">
- <xsd:annotation>
  <xsd:documentation>This attribute defines the type of the Specification of the particular ebBP element. Note: This attribute was added in
      v2.0.</xsd:documentation>
    </xsd:annotation>
    </xsd:attribute>
- <xsd:attribute name="location" type="xsd:anyURI" use="required">
- <xsd:annotation>
   <xsd:documentation>The location of the Specification of the particular ebBP element. Note: This attribute was added in
      v2.0.</xsd:documentation>
    </xsd:annotation>
    </xsd:attribute>
- <xsd:attribute name="targetNamespace" type="xsd:anyURI" use="optional">
- <xsd:annotation>
  <xsd:documentation>The target namespace of the Specification of the particular ebBP element. Note: This attribute was added in
      v2.0.</xsd:documentation>
    </xsd:annotation>
    </xsd:attribute>
- <xsd:attribute name="externalDocumentDefRef" type="xsd:normalizedString" use="optional">
- <xsd:annotation>
  <xsd:documentation>A name, URN or other reference that cites an external specification or reference that defines the document. This
      attribute was added during v2.0.1 Public Review and included in v2.0.2.</xsd:documentation>
    </xsd:annotation>
    </xsd:attribute>
```

```
<xsd:attributeGroup ref="name" />
    </xsd:complexType>
    </xsd:element>
- <xsd:element name="BusinessTransactionActivity">
- <xsd:annotation>
<xsd:documentation>Defines a Business Transaction Activity within a Business Collaboration. A Business Transaction Activity is a business activity that executes a Business Transaction. Note in v2.0, isLegallyBinding was replaced by hasLegalIntent.</xsd:documentation>
    </xsd:annotation>
- <xsd:complexType>
- <xsd:complexContent>
<xsd:extension base="BusinessTransactionActivityType" />
    </xsd:complexContent>
    </xsd:complexType>
    </xsd:element>
- <xsd:complexType name="BusinessTransactionActivityType">
- <xsd:annotation>
<xsd:documentation>The type related to a Business Transaction Activity. The BusinessTransactionActivityType reuses previously defined Business Transactions. Performs elements are required to bind Role values to the Requesting and Responding activities. The older initiatingRoleIDRef attribute is removed as insufficiently versatile. Note: This complexType was added in v2.0.</xsd:documentation>
    </xsd:annotation>
- <xsd:complexContent>
- <xsd:extension base="BusinessActivityType">
- <xsd:sequence>
<xsd:element ref="Documentation" minOccurs="0" maxOccurs="unbounded" />
<xsd:element ref="TimeToPerform" />
<xsd:element ref="Performs" minOccurs="2" maxOccurs="unbounded" />
<xsd:element name="BeginsWhen" type="ConditionExpressionType" minOccurs="0" maxOccurs="unbounded" />
<xsd:element name="PreCondition" type="ConditionExpressionType" minOccurs="0" maxOccurs="unbounded" />
<xsd:element name="PostCondition" type="ConditionExpressionType" minOccurs="0" maxOccurs="unbounded" />
<xsd:element name="EndsWhen" type="ConditionExpressionType" minOccurs="0" maxOccurs="unbounded" />
    </xsd:sequence>
- <xsd:attribute name="businessTransactionRef" type="xsd:IDREF" use="required">
- <xsd:annotation>
<xsd:documentation>The nameID reference for the Business Transaction. This attribute is used to reference the Business Transaction reused by the BusinessTransactionActivityType.</xsd:documentation>
    </xsd:annotation>
    </xsd:attribute>
- <xsd:attribute name="hasLegalIntent" type="xsd:boolean" default="false">
- <xsd:annotation>
<xsd:documentation>Indicates that a particular activity that could represent a statement or commitment between trading partners, and their shared intent. Note: This attribute was renamed to hasLegalIntent from isLegallyBinding in v2.0.</xsd:documentation>
    </xsd:annotation>
    </xsd:attribute>
- <xsd:attribute name="isConcurrent" type="xsd:boolean" default="true">
- <xsd:annotation>
<xsd:documentation>A parameter that governs the flow of transactions. Unlike the security and timing parameters it does not govern the internal flow of a transaction, rather it determines whether at run-time multiple instances of that Business Transaction Activity can
```

```
be 'open' at the same time within any Business Collaboration instance performed between any two partners. isConcurrent limits the ability to execute multiple BTA of the same BT across Business Collaboration instances (with the same party), or within the same Business Collaboration if multiple paths are open. As a result, when isConcurrent is set to false, the BSIs of each party are responsible for serializing these Business Transaction activities.</xsd:documentation>
      </xsd:annotation>
      </xsd:attribute>
      </xsd:extension>
      </xsd:complexContent>
      </xsd:complexType>
- <xsd:element name="ComplexBusinessTransactionActivity">
- <xsd:annotation>
  <xsd:documentation>Defines a new descriptive element that holds an embedded activity that allows recursive embedded activities. This construct is restricted to 'black box' visibility (i.e. the embedded activity is used for visibility only not for a Multiparty Collaboration). The subparties in the ComplexBTA are auxiliary partners (not constrained by the Business Collaboration). Note: The ComplexBTA and other linking constructs replaced the onInitiation flag in v2.0.</xsd:documentation>
      </xsd:annotation>
- <xsd:complexType>
- <xsd:complexContent>
- <xsd:extension base="BusinessTransactionActivityType">
- <xsd:choice minOccurs="0" maxOccurs="unbounded">
- <xsd:sequence>
  <xsd:element ref="ComplexBusinessTransactionActivity" />
  <xsd:element ref="StatusVisibility" />
      </xsd:sequence>
- <xsd:sequence>
  <xsd:element ref="BusinessTransactionActivity" />
  <xsd:element ref="StatusVisibility" />
      </xsd:sequence>
      </xsd:choice>
      </xsd:extension>
      </xsd:complexContent>
      </xsd:complexType>
      </xsd:element>
- <xsd:element name="StatusVisibility">
- <xsd:annotation>
  <xsd:documentation>Information (which can be aggregated) returned by the subparties of an embedded Business Transaction Activity or ComplexBTA for visibility purposes to the outermost ComplexBTA. For example, a subparty (requester in an embedded BTA that is responder in ComplexBTA) returns aggregated supplier information to the ComplexBTA prior to the responder issuing an order response. The Status Visibility element specifies which status values and which Document Envelope events of the embedded processes are considered, if any, when returning the status value to the context of the ComplexBTA. If no status values or DocumentEnvelope events can be monitored, then both BusinessDocumentList and SubstateVisibility are omitted. Note, this element was added in v2.0.</xsd:documentation>
      </xsd:annotation>
- <xsd:complexType>
- <xsd:sequence>
  <xsd:element name="BusinessDocumentList" type="BusinessDocumentValueList" minOccurs="0" />
  <xsd:element name="SubstateVisibility" type="ConditionGuardValueList" minOccurs="0" />
```

```
<xsd:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded" />
    </xsd:sequence>
<xsd:attributeGroup ref="name" />
    </xsd:complexType>
    </xsd:element>
- <xsd:element name="CollaborationActivity" type="CollaborationActivityType">
- <xsd:annotation>
<xsd:documentation>The activity of performing a Business Collaboration within another Business Collaboration.</xsd:documentation>
    </xsd:annotation>
    </xsd:element>
- <xsd:complexType name="CollaborationActivityType">
- <xsd:annotation>
<xsd:documentation>The type related to a Collaboration Activity.</xsd:documentation>
    </xsd:annotation>
- <xsd:complexContent>
- <xsd:extension base="BusinessActivityType">
- <xsd:sequence>
<xsd:element ref="Documentation" minOccurs="0" maxOccurs="unbounded" />
<xsd:element ref="Performs" minOccurs="0" maxOccurs="unbounded" />
    </xsd:sequence>
- <xsd:attribute name="collaborationRef" type="xsd:IDREF" use="required">
- <xsd:annotation>
<xsd:documentation>The nameID reference for the Business Collaboration performed by the Collaboration Activity.</xsd:documentation>
    </xsd:annotation>
    </xsd:attribute>
    </xsd:extension>
    </xsd:complexContent>
    </xsd:complexType>
- <xsd:element name="FromLink">
- <xsd:annotation>
<xsd:documentation>A linking construct that indicates a state that can be transitioned from in the current context (containing element). The FromLink/@fromBusinessStateRef attribute references the state that is transitioned from by its ID value. FromLinks can have 0..n Condition Expressions associated with them. The conditionGuard attribute can contain status values obtained from the state pointed to by the FromLink; matching the value governs whether a transition is made at run time. Note: This element was added in v2.0.</xsd:documentation>
    </xsd:annotation>
- <xsd:complexType>
- <xsd:complexContent>
- <xsd:extension base="BusinessStateLinkType">
- <xsd:attribute name="fromBusinessStateRef" type="xsd:IDREF" use="required">
- <xsd:annotation>
<xsd:documentation>The nameID reference of the Business State of the link transitioned from.</xsd:documentation>
    </xsd:annotation>
    </xsd:attribute>
- <xsd:attribute name="conditionGuard" type="ConditionGuardValue" use="optional">
- <xsd:annotation>
<xsd:documentation>The condition that guards the transition from a Business State.</xsd:documentation>
```

```
    </xsd:annotation>
    </xsd:attribute>
    </xsd:extension>
    </xsd:complexContent>
    </xsd:complexType>
    </xsd:element>
- <xsd:element name="ToLink">
- <xsd:annotation>
  <xsd:documentation>A linking construct that indicates states that the current context (containing element) can transition to. The ToLink/@toBusinessStateRef attribute value references the state through its ID value. Completion States can have 0..n ConditionExpressions that are checked at runtime to determine whether the transition to a state is actually made. Note: This element was added in v2.0.</xsd:documentation>
    </xsd:annotation>
- <xsd:complexType>
- <xsd:complexContent>
- <xsd:extension base="BusinessStateLinkType">
- <xsd:attribute name="toBusinessStateRef" type="xsd:IDREF" use="required">
- <xsd:annotation>
  <xsd:documentation>The nameID reference of the Business State of the link transitioned to.</xsd:documentation>
    </xsd:annotation>
    </xsd:attribute>
    </xsd:extension>
    </xsd:complexContent>
    </xsd:complexType>
    </xsd:element>
- <xsd:complexType name="BusinessStateLinkType">
- <xsd:annotation>
  <xsd:documentation>The type related to the linking constructs (TO and FROM). The type can have 0..n Condition Expression associated with it. Note: The linking constructs on a transition replaced the onInitiation flag in v2.0.</xsd:documentation>
    </xsd:annotation>
- <xsd:sequence>
  <xsd:element ref="Documentation" minOccurs="0" />
  <xsd:element ref="ConditionExpression" minOccurs="0" maxOccurs="unbounded" />
    </xsd:sequence>
    </xsd:complexType>
- <xsd:element name="ConditionExpression">
- <xsd:annotation>
  <xsd:documentation>An expression element that can be evaluated and provide a TRUE or FALSE. Multiple Condition Expression languages and expressions can be used. Note: The capability whereby multiple Condition Expression languages and expressions can be used was expanded (as well as its use with variables) in v2.0.</xsd:documentation>
    </xsd:annotation>
- <xsd:complexType>
- <xsd:complexContent>
  <xsd:extension base="ConditionExpressionType" />
    </xsd:complexContent>
    </xsd:complexType>
    </xsd:element>
```

```
- <xsd:complexType name="ConditionExpressionType">
- <xsd:annotation>
  <xsd:documentation>The type related to the ConditionExpression element. A BSI supports at least the XPath language, as well as the DocumentEnvelope (expressionLanguage of ExpressionLanguageType) which is the nameID of a Document Envelope. In previous versions, this was known as DocumentEnvelopeNotation. When variables are used, XPath and XSLT could be beneficial. Note: This complexType was added in v2.0.</xsd:documentation>
    </xsd:annotation>
- <xsd:sequence>
  <xsd:element ref="Documentation" minOccurs="0" maxOccurs="unbounded" />
    </xsd:sequence>
- <xsd:attribute name="expressionLanguage" type="ExpressionLanguageType" use="required">
- <xsd:annotation>
  <xsd:documentation>Defines the language used for the Condition Expression.</xsd:documentation>
    </xsd:annotation>
    </xsd:attribute>
- <xsd:attribute name="expression" type="xsd:string" use="required">
- <xsd:annotation>
  <xsd:documentation>Defines the value for the Condition Expression.</xsd:documentation>
    </xsd:annotation>
    </xsd:attribute>
- <xsd:attribute name="defaultValue" type="xsd:string" use="optional">
- <xsd:annotation>
  <xsd:documentation>Allows a default value to be specified for the Condition Expression. This default value is expressed as an xsd:string. To acquire an xsd:boolean, an XSLT constant boolean function of TRUE or FALSE may be used.</xsd:documentation>
    </xsd:annotation>
    </xsd:attribute>
    </xsd:complexType>
- <xsd:element name="Start" type="StartType">
- <xsd:annotation>
  <xsd:documentation>The specific Collaboration started with to traverse a path through a graph to a Completion State.</xsd:documentation>
    </xsd:annotation>
- <xsd:unique name="Start-ID">
  <xsd:selector xpath="." />
  <xsd:field xpath="nameID" />
    </xsd:unique>
    </xsd:element>
- <xsd:complexType name="StartType">
- <xsd:annotation>
  <xsd:documentation>The type related to the Start of a specific Collaboration type within the Collaboration Group.</xsd:documentation>
    </xsd:annotation>
- <xsd:sequence>
  <xsd:element ref="Documentation" minOccurs="0" maxOccurs="unbounded" />
  <xsd:element ref="ToLink" maxOccurs="unbounded" />
    </xsd:sequence>
  <xsd:attributeGroup ref="optname" />
    </xsd:complexType>
- <xsd:element name="Transition" type="TransitionType">
```

```
- <xsd:annotation>
  <xsd:documentation>A link between business states in a Business Collaboration. Choreography is expressed as transitions between business states. Transition to the same business state is allowed.</xsd:documentation>
    </xsd:annotation>
    </xsd:element>
- <xsd:complexType name="TransitionType">
- <xsd:annotation>
  <xsd:documentation>The type related to the traverse between business states.</xsd:documentation>
    </xsd:annotation>
- <xsd:sequence>
  <xsd:element ref="FromLink" />
  <xsd:element ref="ToLink" />
    </xsd:sequence>
  <xsd:attributeGroup ref="optname" />
    </xsd:complexType>
- <xsd:element name="Decision" type="DecisionType">
- <xsd:annotation>
  <xsd:documentation>A particular pattern of transition between business states. For example, a choice. This is similar to a Decision in a UML activity diagram, although precise semantics differ. One incoming link and many outgoing links, only one of which is taken.</xsd:documentation>
    </xsd:annotation>
- <xsd:unique name="Decision-ID">
  <xsd:selector xpath="." />
  <xsd:field xpath="nameID" />
    </xsd:unique>
    </xsd:element>
- <xsd:complexType name="DecisionType">
- <xsd:annotation>
  <xsd:documentation>The type related to a Decision construct.</xsd:documentation>
    </xsd:annotation>
- <xsd:sequence>
  <xsd:element ref="Documentation" minOccurs="0" maxOccurs="unbounded" />
  <xsd:element ref="FromLink" />
  <xsd:element ref="ToLink" minOccurs="2" maxOccurs="unbounded" />
    </xsd:sequence>
  <xsd:attributeGroup ref="optname" />
    </xsd:complexType>
- <xsd:element name="Fork" type="ForkType">
- <xsd:annotation>
  <xsd:documentation>A choreography construct with one incoming transition and many outgoing transitions. All outgoing transitions are considered to happen either in parallel or, when indicated, an exclusive OR is operative.</xsd:documentation>
    </xsd:annotation>
- <xsd:unique name="Fork-ID">
  <xsd:selector xpath="." />
  <xsd:field xpath="nameID" />
    </xsd:unique>
    </xsd:element>
```

```
- <xsd:complexType name="ForkType">
- <xsd:annotation>
  <xsd:documentation>The type related to the Fork construct</xsd:documentation>
      </xsd:annotation>
- <xsd:sequence>
  <xsd:element ref="Documentation" minOccurs="0" maxOccurs="unbounded" />
  <xsd:element ref="TimeToPerform" minOccurs="0" />
  <xsd:element ref="FromLink" />
  <xsd:element ref="ToLink" minOccurs="2" maxOccurs="unbounded" />
      </xsd:sequence>
  <xsd:attributeGroup ref="optname" />
- <xsd:attribute name="type" use="optional" default="OR">
- <xsd:annotation>
  <xsd:documentation>Defines the type of Fork. OR: An OR value will mean that any business activity pointed to by a transition coming from
      the fork might be initiated. All activities may run in parallel. XOR: Only one of the possible activities will run.</xsd:documentation>
      </xsd:annotation>
- <xsd:simpleType>
- <xsd:restriction base="xsd:NMTOKEN">
  <xsd:enumeration value="OR" />
  <xsd:enumeration value="XOR" />
      </xsd:restriction>
      </xsd:simpleType>
      </xsd:attribute>
      </xsd:complexType>
- <xsd:element name="Join" type="JoinType">
- <xsd:annotation>
  <xsd:documentation>A choreography construct that defines the point where one or more forked activities join. Can define that the
      completion of all state occur.</xsd:documentation>
      </xsd:annotation>
- <xsd:unique name="Join-ID">
  <xsd:selector xpath="." />
  <xsd:field xpath="nameID" />
      </xsd:unique>
      </xsd:element>
- <xsd:complexType name="JoinType">
- <xsd:annotation>
  <xsd:documentation>The type related to the Join construct</xsd:documentation>
      </xsd:annotation>
- <xsd:sequence>
  <xsd:element ref="Documentation" minOccurs="0" maxOccurs="unbounded" />
  <xsd:element ref="FromLink" minOccurs="2" maxOccurs="unbounded" />
  <xsd:element ref="ToLink" minOccurs="0" />
      </xsd:sequence>
  <xsd:attributeGroup ref="optname" />
- <xsd:attribute name="waitForAll" type="xsd:boolean" default="true">
- <xsd:annotation>
```

```
<xsd:documentation>Indicates that all transitions coming into the Join are executed in order for the Business Collaboration to reach the Join state (AND-join). By default, the Join is an AND-join.</xsd:documentation>
</xsd:annotation>
</xsd:attribute>
</xsd:complexType>
- <xsd:element name="Success" type="CompletionType">
- <xsd:annotation>
<xsd:documentation>Defines a successful completion of a Collaboration as a transition from an activity.</xsd:documentation>
</xsd:annotation>
- <xsd:unique name="Success-ID">
<xsd:selector xpath="." />
<xsd:field xpath="nameID" />
</xsd:unique>
</xsd:element>
- <xsd:element name="Failure" type="CompletionType">
- <xsd:annotation>
<xsd:documentation>Defines a failure completion of a Business Collaboration as a transition from an activity.</xsd:documentation>
</xsd:annotation>
- <xsd:unique name="Failure-ID">
<xsd:selector xpath="." />
<xsd:field xpath="nameID" />
</xsd:unique>
</xsd:element>
- <xsd:complexType name="CompletionType">
- <xsd:annotation>
<xsd:documentation>The type related to the Success for Failure completion of a Business Collaboration as a transition from an activity.</xsd:documentation>
</xsd:annotation>
- <xsd:sequence>
<xsd:element ref="Documentation" minOccurs="0" maxOccurs="unbounded" />
<xsd:element ref="FromLink" minOccurs="0" maxOccurs="unbounded" />
</xsd:sequence>
<xsd:attributeGroup ref="optname" />
</xsd:complexType>
- <xsd:element name="BusinessAction" type="BusinessActionType" abstract="true">
- <xsd:annotation>
<xsd:documentation>An abstract superclass that holds the attributes common to the Requesting and Responding Business Activity.</xsd:documentation>
</xsd:annotation>
</xsd:element>
- <xsd:complexType name="BusinessActionType">
- <xsd:annotation>
<xsd:documentation>The type related to the Business Action.</xsd:documentation>
</xsd:annotation>
- <xsd:sequence>
<xsd:element ref="Documentation" minOccurs="0" maxOccurs="unbounded" />
<xsd:element ref="DocumentEnvelope" minOccurs="0" maxOccurs="unbounded" />
```

```
<xsd:element name="ReceiptAcknowledgement" type="ReceiptAcknowledgementType" minOccurs="0" />
<xsd:element name="ReceiptAcknowledgementException" type="ReceiptAcknowledgementExceptionType" minOccurs="0" />
<xsd:element name="AcceptanceAcknowledgement" type="AcceptanceAcknowledgementType" minOccurs="0" />
<xsd:element name="AcceptanceAcknowledgementException" type="AcceptanceAcknowledgementExceptionType" minOccurs="0" />
    </xsd:sequence>
<xsd:attributeGroup ref="name" />
<xsd:attributeGroup ref="quality" />
    </xsd:complexType>
- <xsd:element name="RequestingBusinessActivity" type="RequestingBusinessActivityType">
- <xsd:annotation>
<xsd:documentation>A Business Action performed by the Requesting role within a Business Transaction.</xsd:documentation>
    </xsd:annotation>
    </xsd:element>
- <xsd:complexType name="RequestingBusinessActivityType">
- <xsd:annotation>
<xsd:documentation>The type related to the Requesting Business Action.</xsd:documentation>
    </xsd:annotation>
- <xsd:complexContent>
<xsd:extension base="BusinessActionType" />
    </xsd:complexContent>
    </xsd:complexType>
- <xsd:element name="RespondingBusinessActivity" type="RespondingBusinessActivityType">
- <xsd:annotation>
<xsd:documentation>A Business Action performed by the Responding role within a Business Transaction.</xsd:documentation>
    </xsd:annotation>
    </xsd:element>
- <xsd:complexType name="RespondingBusinessActivityType">
- <xsd:annotation>
<xsd:documentation>The type related to the Responding Business Action.</xsd:documentation>
    </xsd:annotation>
- <xsd:complexContent>
<xsd:extension base="BusinessActionType" />
    </xsd:complexContent>
    </xsd:complexType>
- <xsd:complexType name="BusinessActivityType">
- <xsd:annotation>
<xsd:documentation>The type related to the Business Transaction Activity or Collaboration Activity types (and business state).</xsd:documentation>
    </xsd:annotation>
<xsd:attributeGroup ref="name" />
    </xsd:complexType>
- <xsd:element name="Performs">
- <xsd:annotation>
<xsd:documentation>Performs elements are required whenever referencing the RequestingBusinessActivity or RespondingBusinessActivity in a BTA or within the BTAs of a ComplexBTA. Also Performs elements are required when the Role values in a referring context differ from or need to be switched between the Role values in the referenced context. (The main referring contexts for Business Collaborations are the content models of the CollaborationActivity and ExternalRoles elements.The BTAs and ComplexBTAs are the
```

other referring contexts.) For example, in a Business Collaboration between two parties is related to another Business Collaboration (also of two parties) using a Collaboration Activity, the roles may change for the involved parties. Those roles are traced and associated with the parties. This functionality supports tracing and binding of roles of the Business Collaboration across and within multiple levels of nesting. Where allowed, the Performs element may be omitted if the actual values of Roles in the referring and referred-to context are the same (i.e. string identical) and they match. Note: This element was added in v2.0.</xsd:documentation>
</xsd:annotation>
- <xsd:complexType>
- <xsd:complexContent>
<xsd:extension base="PerformsType" />
</xsd:complexContent>
</xsd:complexType>
- <xsd:unique name="Performs-ID">
<xsd:selector xpath="." />
<xsd:field xpath="nameID" />
</xsd:unique>
</xsd:element>
- <xsd:complexType name="PerformsType">
- <xsd:annotation>
<xsd:documentation>The type related to the role binding Performs element.</xsd:documentation>
</xsd:annotation>
- <xsd:sequence>
<xsd:element ref="Documentation" minOccurs="0" />
</xsd:sequence>
- <xsd:attribute name="currentRoleRef" type="xsd:IDREF" use="required">
- <xsd:annotation>
<xsd:documentation>The currentRoleRef attribute defines the nameID reference of the specific role currently used in an activity. If roles change, this name ID reference for the currently used role is the referring role that is linked to the referred-to role in the performsRoleRef.</xsd:documentation>
</xsd:annotation>
</xsd:attribute>
- <xsd:attribute name="performsRoleRef" type="xsd:IDREF" use="optional">
- <xsd:annotation>
<xsd:documentation>The performsRoleRef attribute defines nameID reference of the referred-to role for the specific role used in an activity. This referred-to role is bound to the referring role (currentRoleRef).</xsd:documentation>
</xsd:annotation>
</xsd:attribute>
</xsd:complexType>
- <xsd:complexType name="RoleType">
- <xsd:annotation>
<xsd:documentation>The type related to the Role of a Business Collaboration. For example, in a Business Collaboration, two or more top-level roles apply. The Role Type is a global element, that allows Role elements to be defined of Role Type.</xsd:documentation>
</xsd:annotation>
- <xsd:sequence>
<xsd:element ref="Documentation" minOccurs="0" />
</xsd:sequence>
<xsd:attributeGroup ref="name" />
</xsd:complexType>

```
- <xsd:element name="BusinessTransactionHead" type="BusinessTransactionBaseType" abstract="true">
- <xsd:annotation>
  <xsd:documentation>The abstract superclass associated with the concrete set of defined Business Transaction patterns. A Business Transaction is a set of business information and Business Signal exchanges amongst business partners that occurs in an agreed upon format and sequence (as supported by the patterns). Through the Business Transaction Head, the concrete set of BT patterns and the Data Exchange element (which allows pattern specialization) enable business exchange through a defined pattern. The Business Transaction Head in essence is a placeholder where the concrete pattern is substituted (i.e. a Request-Confirm is substituted and used). Note: The Business Transaction Head replaced the Business Transaction element in v2.0.</xsd:documentation>
  </xsd:annotation>
  </xsd:element>
- <xsd:complexType name="BusinessTransactionBaseType">
- <xsd:annotation>
  <xsd:documentation>The type related to the abstract superclass associated with Business Transactions.</xsd:documentation>
  </xsd:annotation>
- <xsd:sequence>
  <xsd:element ref="Documentation" minOccurs="0" />
- <xsd:element name="RequestingRole" type="RoleType">
- <xsd:annotation>
  <xsd:documentation>Allows definition of the Requesting declarative role on the Business Transaction. This explicit, yet abstract, role facilitates role mapping. This element was added in v2.0.1.</xsd:documentation>
  </xsd:annotation>
  </xsd:element>
- <xsd:element name="RespondingRole" type="RoleType">
- <xsd:annotation>
  <xsd:documentation>Allows definition of the the Responding declarative role on the Business Transaction. This explicit, yet abstract, role facilitates role mapping. This element was added in v2.0.1.</xsd:documentation>
  </xsd:annotation>
  </xsd:element>
  </xsd:sequence>
  <xsd:attributeGroup ref="name" />
- <xsd:attribute name="pattern" type="xsd:anyURI">
- <xsd:annotation>
  <xsd:documentation>May point to the pattern on which the Business Transaction is based. This attribute is used only when the concrete BT patterns provided are not used. This attribute is retained for backward compatibility. The concrete Business Transaction patterns are also specified. This attribute is not be used if one of the concrete, extensible (Data Exchange) or Legacy Business Transaction (used for conversion purposes only) patterns are used. Note: These changes were made in v2.0.</xsd:documentation>
  </xsd:annotation>
  </xsd:attribute>
- <xsd:attribute name="isGuaranteedDeliveryRequired" type="xsd:boolean" default="false">
- <xsd:annotation>
  <xsd:documentation>Refers to the expectation that the underlying messaging service used to implement the Business Transaction protocol provides guaranteed delivery, i.e. reliable messaging.</xsd:documentation>
  </xsd:annotation>
  </xsd:attribute>
  </xsd:complexType>
- <xsd:element name="BusinessTransaction" type="BusinessTransactionType" substitutionGroup="BusinessTransactionHead">
- <xsd:annotation>
```

```
<xsd:documentation>The Business Transaction of type BusinessTransactionType is based on the Commercial Transaction pattern. This pattern (and the two elements that map to it) is typically a formal obligation between parties. Historically, the Commercial Transaction (as one of the defined patterns) has been colloquially known as the Business Transaction. Note: This element that maps to the Commercial Transaction pattern (i.e. the complex type BusinessTransactionType) was added in v2.0.</xsd:documentation>
</xsd:annotation>
</xsd:element>
- <xsd:element name="CommercialTransaction" type="BusinessTransactionType" substitutionGroup="BusinessTransactionHead">
- <xsd:annotation>
<xsd:documentation>The Commercial Transaction is based on the Commercial Transaction pattern. This pattern (and the two elements that map to it) is typically a formal obligation between parties. Historically, the Commercial Transaction (as one of the defined patterns) has been colloquially known as the Business Transaction. Note: This element that maps to the Commercial Transaction pattern (i.e. the complex type BusinessTransactionType) was added in v2.0.</xsd:documentation>
</xsd:annotation>
</xsd:element>
- <xsd:complexType name="BusinessTransactionType">
- <xsd:annotation>
<xsd:documentation>Either the Commercial Transaction or the Business Transaction elements map to type BusinessTransactionType that relates to the Commercial Transaction pattern. The type BusinessTransactionType alllows different communities to use the Commercial Transaction pattern via the two elements that map to it - Commercial Transaction or Business Transaction. Note: This complexType was added in v2.0.</xsd:documentation>
</xsd:annotation>
- <xsd:complexContent>
- <xsd:extension base="BusinessTransactionBaseType">
- <xsd:sequence>
- <xsd:element name="RequestingBusinessActivity">
- <xsd:complexType>
- <xsd:complexContent>
- <xsd:restriction base="RequestingBusinessActivityType">
- <xsd:sequence>
<xsd:element ref="Documentation" minOccurs="0" maxOccurs="unbounded" />
<xsd:element ref="DocumentEnvelope" />
<xsd:element name="ReceiptAcknowledgement" type="ReceiptAcknowledgementType" />
<xsd:element name="ReceiptAcknowledgementException" type="ReceiptAcknowledgementExceptionType" />
<xsd:element name="AcceptanceAcknowledgement" type="AcceptanceAcknowledgementType" minOccurs="0" />
<xsd:element name="AcceptanceAcknowledgementException" type="AcceptanceAcknowledgementExceptionType" minOccurs="0" />
</xsd:sequence>
<xsd:attribute name="isAuthorizationRequired" type="xsd:boolean" />
<xsd:attribute name="isIntelligibleCheckRequired" type="xsd:boolean" />
<xsd:attribute name="isNonRepudiationRequired" type="xsd:boolean" />
<xsd:attribute name="isNonRepudiationReceiptRequired" type="xsd:boolean" />
<xsd:attribute name="timeToAcknowledgeReceipt" type="xsd:duration" />
<xsd:attribute name="timeToAcknowledgeAcceptance" type="xsd:duration" />
<xsd:attribute name="retryCount" type="xsd:int" />
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
</xsd:element>
```

- <xsd:element name="RespondingBusinessActivity">
- <xsd:complexType>
- <xsd:complexContent>
- <xsd:restriction base="RespondingBusinessActivityType">
- <xsd:sequence>
<xsd:element ref="Documentation" minOccurs="0" maxOccurs="unbounded" />
<xsd:element ref="DocumentEnvelope" maxOccurs="unbounded" />
<xsd:element name="ReceiptAcknowledgement" type="ReceiptAcknowledgementType" />
<xsd:element name="ReceiptAcknowledgementException" type="ReceiptAcknowledgementExceptionType" />
<xsd:element name="AcceptanceAcknowledgement" type="AcceptanceAcknowledgementType" minOccurs="0" />
<xsd:element name="AcceptanceAcknowledgementException" type="AcceptanceAcknowledgementExceptionType" minOccurs="0" />
</xsd:sequence>
<xsd:attribute name="isAuthorizationRequired" type="xsd:boolean" />
<xsd:attribute name="isIntelligibleCheckRequired" type="xsd:boolean" />
<xsd:attribute name="isNonRepudiationRequired" type="xsd:boolean" />
<xsd:attribute name="isNonRepudiationReceiptRequired" type="xsd:boolean" />
<xsd:attribute name="timeToAcknowledgeReceipt" type="xsd:duration" />
<xsd:attribute name="timeToAcknowledgeAcceptance" type="xsd:duration" />
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
</xsd:element>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
- <xsd:element name="RequestConfirm" substitutionGroup="BusinessTransactionHead">
- <xsd:annotation>
<xsd:documentation>A concrete Business Transaction Pattern where an initiating party requests information about their status of a previous request or a Responder's business rules. Typically no residual obligation between parties applies. For example, an initiating party could request authorization to sell specific products where a confirmation response is expected. Note: This concrete pattern was added in v2.0.</xsd:documentation>
</xsd:annotation>
- <xsd:complexType>
- <xsd:complexContent>
- <xsd:extension base="BusinessTransactionBaseType">
- <xsd:sequence>
- <xsd:element name="RequestingBusinessActivity">
- <xsd:complexType>
- <xsd:complexContent>
- <xsd:restriction base="RequestingBusinessActivityType">
- <xsd:sequence>
<xsd:element ref="Documentation" minOccurs="0" maxOccurs="unbounded" />
<xsd:element ref="DocumentEnvelope" />
<xsd:element name="ReceiptAcknowledgement" type="ReceiptAcknowledgementType" />
<xsd:element name="ReceiptAcknowledgementException" type="ReceiptAcknowledgementExceptionType" />
<xsd:element name="AcceptanceAcknowledgement" type="AcceptanceAcknowledgementType" minOccurs="0" maxOccurs="0" />

```
    <xsd:element name="AcceptanceAcknowledgementException" type="AcceptanceAcknowledgementExceptionType" minOccurs="0"
      maxOccurs="0" />
    </xsd:sequence>
  <xsd:attribute name="isAuthorizationRequired" type="xsd:boolean" />
  <xsd:attribute name="isIntelligibleCheckRequired" type="xsd:boolean" />
  <xsd:attribute name="timeToAcknowledgeReceipt" type="xsd:duration" />
  <xsd:attribute name="isNonRepudiationReceiptRequired" type="xsd:boolean" />
  <xsd:attribute name="isNonRepudiationRequired" type="xsd:boolean" />
  <xsd:attribute name="retryCount" type="xsd:int" />
    </xsd:restriction>
    </xsd:complexContent>
    </xsd:complexType>
    </xsd:element>
- <xsd:element name="RespondingBusinessActivity">
- <xsd:complexType>
- <xsd:complexContent>
- <xsd:restriction base="RespondingBusinessActivityType">
- <xsd:sequence>
  <xsd:element ref="Documentation" minOccurs="0" maxOccurs="unbounded" />
  <xsd:element ref="DocumentEnvelope" maxOccurs="unbounded" />
  <xsd:element name="ReceiptAcknowledgement" type="ReceiptAcknowledgementType" />
  <xsd:element name="ReceiptAcknowledgementException" type="ReceiptAcknowledgementExceptionType" />
  <xsd:element name="AcceptanceAcknowledgement" type="AcceptanceAcknowledgementType" minOccurs="0" maxOccurs="0" />
    <xsd:element name="AcceptanceAcknowledgementException" type="AcceptanceAcknowledgementExceptionType" minOccurs="0"
      maxOccurs="0" />
    </xsd:sequence>
  <xsd:attribute name="isAuthorizationRequired" type="xsd:boolean" />
  <xsd:attribute name="isIntelligibleCheckRequired" type="xsd:boolean" />
  <xsd:attribute name="isNonRepudiationRequired" type="xsd:boolean" />
  <xsd:attribute name="isNonRepudiationReceiptRequired" type="xsd:boolean" />
  <xsd:attribute name="timeToAcknowledgeReceipt" type="xsd:duration" />
    </xsd:restriction>
    </xsd:complexContent>
    </xsd:complexType>
    </xsd:element>
    </xsd:sequence>
    </xsd:extension>
    </xsd:complexContent>
    </xsd:complexType>
    </xsd:element>
- <xsd:element name="RequestResponse" substitutionGroup="BusinessTransactionHead">
- <xsd:annotation>
  <xsd:documentation>A concrete Business Transaction Pattern where there is typically no residual obligation between the parties. The
      Request-Response Business Transaction Pattern can be used when an initiating party requests information the Responding party
      already has. This pattern is used when a complex set of interrelated results are required; otherwise use Query-Response. Typically
      no residual obligation between parties applies. For example, a request for price and availability. Note: This concrete pattern was
      added in v2.0.</xsd:documentation>
```

```
      </xsd:annotation>
- <xsd:complexType>
- <xsd:complexContent>
- <xsd:extension base="BusinessTransactionBaseType">
- <xsd:sequence>
- <xsd:element name="RequestingBusinessActivity">
- <xsd:complexType>
- <xsd:complexContent>
- <xsd:restriction base="RequestingBusinessActivityType">
- <xsd:sequence>
 <xsd:element ref="Documentation" minOccurs="0" maxOccurs="unbounded" />
 <xsd:element ref="DocumentEnvelope" />
 <xsd:element name="ReceiptAcknowledgement" type="ReceiptAcknowledgementType" minOccurs="0" />
 <xsd:element name="ReceiptAcknowledgementException" type="ReceiptAcknowledgementExceptionType" minOccurs="0" />
 <xsd:element name="AcceptanceAcknowledgement" type="AcceptanceAcknowledgementType" minOccurs="0" maxOccurs="0" />
   <xsd:element name="AcceptanceAcknowledgementException" type="AcceptanceAcknowledgementExceptionType" minOccurs="0"
     maxOccurs="0" />
   </xsd:sequence>
 <xsd:attribute name="isAuthorizationRequired" type="xsd:boolean" />
 <xsd:attribute name="isIntelligibleCheckRequired" type="xsd:boolean" />
 <xsd:attribute name="timeToAcknowledgeReceipt" type="xsd:duration" />
 <xsd:attribute name="isNonRepudiationReceiptRequired" type="xsd:boolean" />
 <xsd:attribute name="isNonRepudiationRequired" type="xsd:boolean" />
 <xsd:attribute name="retryCount" type="xsd:int" />
   </xsd:restriction>
   </xsd:complexContent>
   </xsd:complexType>
   </xsd:element>
- <xsd:element name="RespondingBusinessActivity">
- <xsd:complexType>
- <xsd:complexContent>
- <xsd:restriction base="RespondingBusinessActivityType">
- <xsd:sequence>
 <xsd:element ref="Documentation" minOccurs="0" maxOccurs="unbounded" />
 <xsd:element ref="DocumentEnvelope" maxOccurs="unbounded" />
 <xsd:element name="ReceiptAcknowledgement" type="ReceiptAcknowledgementType" minOccurs="0" />
 <xsd:element name="ReceiptAcknowledgementException" type="ReceiptAcknowledgementExceptionType" minOccurs="0" />
 <xsd:element name="AcceptanceAcknowledgement" type="AcceptanceAcknowledgementType" minOccurs="0" maxOccurs="0" />
   <xsd:element name="AcceptanceAcknowledgementException" type="AcceptanceAcknowledgementExceptionType" minOccurs="0"
     maxOccurs="0" />
   </xsd:sequence>
 <xsd:attribute name="isAuthorizationRequired" type="xsd:boolean" />
 <xsd:attribute name="isIntelligibleCheckRequired" type="xsd:boolean" />
 <xsd:attribute name="timeToAcknowledgeReceipt" type="xsd:duration" />
 <xsd:attribute name="isNonRepudiationReceiptRequired" type="xsd:boolean" />
 <xsd:attribute name="isNonRepudiationRequired" type="xsd:boolean" />
   </xsd:restriction>
```

```
    </xsd:complexContent>
    </xsd:complexType>
    </xsd:element>
    </xsd:sequence>
    </xsd:extension>
    </xsd:complexContent>
    </xsd:complexType>
    </xsd:element>
- <xsd:element name="QueryResponse" substitutionGroup="BusinessTransactionHead">
- <xsd:annotation>
  <xsd:documentation>A concrete Business Transaction Pattern where the Requester queries for information the Responder already has. The
      response meets the specified constraining criteria. If a complex set of interrelated results are required, use Request-Response
      Business Transaction Pattern. Typically no residual obligation between parties applies. Note: This concrete pattern was added in
      v2.0.</xsd:documentation>
    </xsd:annotation>
- <xsd:complexType>
- <xsd:complexContent>
- <xsd:extension base="BusinessTransactionBaseType">
- <xsd:sequence>
- <xsd:element name="RequestingBusinessActivity">
- <xsd:complexType>
- <xsd:complexContent>
- <xsd:restriction base="RequestingBusinessActivityType">
- <xsd:sequence>
  <xsd:element ref="Documentation" minOccurs="0" maxOccurs="unbounded" />
  <xsd:element ref="DocumentEnvelope" />
  <xsd:element name="ReceiptAcknowledgement" type="ReceiptAcknowledgementType" minOccurs="0" />
  <xsd:element name="ReceiptAcknowledgementException" type="ReceiptAcknowledgementExceptionType" minOccurs="0" />
  <xsd:element name="AcceptanceAcknowledgement" type="AcceptanceAcknowledgementType" minOccurs="0" maxOccurs="0" />
    <xsd:element name="AcceptanceAcknowledgementException" type="AcceptanceAcknowledgementExceptionType" minOccurs="0"
      maxOccurs="0" />
    </xsd:sequence>
  <xsd:attribute name="isAuthorizationRequired" type="xsd:boolean" />
  <xsd:attribute name="isIntelligibleCheckRequired" type="xsd:boolean" />
  <xsd:attribute name="timeToAcknowledgeReceipt" type="xsd:duration" />
  <xsd:attribute name="isNonRepudiationReceiptRequired" type="xsd:boolean" />
  <xsd:attribute name="isNonRepudiationRequired" type="xsd:boolean" />
  <xsd:attribute name="retryCount" type="xsd:int" />
    </xsd:restriction>
    </xsd:complexContent>
    </xsd:complexType>
    </xsd:element>
- <xsd:element name="RespondingBusinessActivity">
- <xsd:complexType>
- <xsd:complexContent>
- <xsd:restriction base="RespondingBusinessActivityType">
- <xsd:sequence>
```

```
<xsd:element ref="Documentation" minOccurs="0" maxOccurs="unbounded" />
<xsd:element ref="DocumentEnvelope" maxOccurs="unbounded" />
<xsd:element name="ReceiptAcknowledgement" type="ReceiptAcknowledgementType" minOccurs="0" />
<xsd:element name="ReceiptAcknowledgementException" type="ReceiptAcknowledgementExceptionType" minOccurs="0" />
<xsd:element name="AcceptanceAcknowledgement" type="AcceptanceAcknowledgementType" minOccurs="0" maxOccurs="0" />
  <xsd:element name="AcceptanceAcknowledgementException" type="AcceptanceAcknowledgementExceptionType" minOccurs="0"
    maxOccurs="0" />
  </xsd:sequence>
<xsd:attribute name="isAuthorizationRequired" type="xsd:boolean" />
<xsd:attribute name="isIntelligibleCheckRequired" type="xsd:boolean" />
<xsd:attribute name="timeToAcknowledgeReceipt" type="xsd:duration" />
<xsd:attribute name="isNonRepudiationReceiptRequired" type="xsd:boolean" />
<xsd:attribute name="isNonRepudiationRequired" type="xsd:boolean" />
    </xsd:restriction>
    </xsd:complexContent>
    </xsd:complexType>
    </xsd:element>
    </xsd:sequence>
    </xsd:extension>
    </xsd:complexContent>
    </xsd:complexType>
    </xsd:element>
- <xsd:element name="InformationDistribution" substitutionGroup="BusinessTransactionHead">
- <xsd:annotation>
  <xsd:documentation>A concrete Business Transaction Pattern where an informal exchange occurs between parties. Typically no residual
      obligation between parties applies. No Responding Business Document (response) applies. It is important to note that in this pattern
      there is a Responding Business Activity and corresponding Responding Role irrespective of the fact there is not a Responding
      Business Document. That Responding role receives and processes (in an abstract sense) the Request. The Responding Business
      Activity binds the associate role to the business action. Each activity, Requesting or Responding, has roles bound and linked to it.
      Note: This concrete pattern was added in v2.0.</xsd:documentation>
    </xsd:annotation>
- <xsd:complexType>
- <xsd:complexContent>
- <xsd:extension base="BusinessTransactionBaseType">
- <xsd:sequence>
- <xsd:element name="RequestingBusinessActivity">
- <xsd:complexType>
- <xsd:complexContent>
- <xsd:restriction base="RequestingBusinessActivityType">
- <xsd:sequence>
  <xsd:element ref="Documentation" minOccurs="0" maxOccurs="unbounded" />
  <xsd:element ref="DocumentEnvelope" />
  <xsd:element name="ReceiptAcknowledgement" type="ReceiptAcknowledgementType" minOccurs="0" />
  <xsd:element name="ReceiptAcknowledgementException" type="ReceiptAcknowledgementExceptionType" minOccurs="0" />
  <xsd:element name="AcceptanceAcknowledgement" type="AcceptanceAcknowledgementType" minOccurs="0" maxOccurs="0" />
    <xsd:element name="AcceptanceAcknowledgementException" type="AcceptanceAcknowledgementExceptionType" minOccurs="0"
      maxOccurs="0" />
```

```
      </xsd:sequence>
    <xsd:attribute name="isAuthorizationRequired" type="xsd:boolean" />
    <xsd:attribute name="isIntelligibleCheckRequired" type="xsd:boolean" />
    <xsd:attribute name="timeToAcknowledgeReceipt" type="xsd:duration" />
    <xsd:attribute name="retryCount" type="xsd:int" />
      </xsd:restriction>
      </xsd:complexContent>
      </xsd:complexType>
      </xsd:element>
  - <xsd:element name="RespondingBusinessActivity">
  - <xsd:complexType>
  - <xsd:complexContent>
  - <xsd:restriction base="RespondingBusinessActivityType">
  - <xsd:sequence>
    <xsd:element ref="Documentation" minOccurs="0" maxOccurs="unbounded" />
    <xsd:element ref="DocumentEnvelope" minOccurs="0" maxOccurs="0" />
    <xsd:element name="ReceiptAcknowledgement" type="ReceiptAcknowledgementType" minOccurs="0" maxOccurs="0" />
    <xsd:element name="ReceiptAcknowledgementException" type="ReceiptAcknowledgementExceptionType" minOccurs="0" maxOccurs="0"
      />
    <xsd:element name="AcceptanceAcknowledgement" type="AcceptanceAcknowledgementType" minOccurs="0" maxOccurs="0" />
      <xsd:element name="AcceptanceAcknowledgementException" type="AcceptanceAcknowledgementExceptionType" minOccurs="0"
        maxOccurs="0" />
      </xsd:sequence>
    <xsd:attribute name="isAuthorizationRequired" type="xsd:boolean" />
    <xsd:attribute name="isIntelligibleCheckRequired" type="xsd:boolean" />
    <xsd:attribute name="timeToAcknowledgeReceipt" type="xsd:duration" />
    <xsd:attribute name="isNonRepudiationReceiptRequired" type="xsd:boolean" />
    <xsd:attribute name="isNonRepudiationRequired" type="xsd:boolean" />
      </xsd:restriction>
      </xsd:complexContent>
      </xsd:complexType>
      </xsd:element>
      </xsd:sequence>
      </xsd:extension>
      </xsd:complexContent>
      </xsd:complexType>
      </xsd:element>
  - <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. It is important to note that in this pattern there is a Responding Business Activity
        and corresponding Responding Role irrespective of the fact there is not a Responding Business Document. That Responding role
        receives and processes (in an abstract sense) the Request. The Responding Business Activity binds the associate role to the business
        action. Each activity, Requesting or Responding, has roles bound and linked to it. Note: This concrete pattern was added in
        v2.0.</xsd:documentation>
```

```
    </xsd:annotation>
- <xsd:complexType>
- <xsd:complexContent>
- <xsd:extension base="BusinessTransactionBaseType">
- <xsd:sequence>
- <xsd:element name="RequestingBusinessActivity">
- <xsd:complexType>
- <xsd:complexContent>
- <xsd:restriction base="RequestingBusinessActivityType">
- <xsd:sequence>
  <xsd:element ref="Documentation" minOccurs="0" maxOccurs="unbounded" />
  <xsd:element ref="DocumentEnvelope" />
  <xsd:element name="ReceiptAcknowledgement" type="ReceiptAcknowledgementType" />
  <xsd:element name="ReceiptAcknowledgementException" type="ReceiptAcknowledgementExceptionType" />
  <xsd:element name="AcceptanceAcknowledgement" type="AcceptanceAcknowledgementType" minOccurs="0" />
  <xsd:element name="AcceptanceAcknowledgementException" type="AcceptanceAcknowledgementExceptionType" minOccurs="0" />
    </xsd:sequence>
  <xsd:attribute name="isAuthorizationRequired" type="xsd:boolean" />
  <xsd:attribute name="isIntelligibleCheckRequired" type="xsd:boolean" />
  <xsd:attribute name="isNonRepudiationRequired" type="xsd:boolean" />
  <xsd:attribute name="isNonRepudiationReceiptRequired" type="xsd:boolean" />
  <xsd:attribute name="timeToAcknowledgeReceipt" type="xsd:duration" />
  <xsd:attribute name="timeToAcknowledgeAcceptance" type="xsd:duration" />
  <xsd:attribute name="retryCount" type="xsd:int" />
    </xsd:restriction>
    </xsd:complexContent>
    </xsd:complexType>
    </xsd:element>
- <xsd:element name="RespondingBusinessActivity">
- <xsd:complexType>
- <xsd:complexContent>
- <xsd:restriction base="RespondingBusinessActivityType">
- <xsd:sequence>
  <xsd:element ref="Documentation" minOccurs="0" maxOccurs="unbounded" />
  <xsd:element ref="DocumentEnvelope" minOccurs="0" maxOccurs="0" />
  <xsd:element name="ReceiptAcknowledgement" type="ReceiptAcknowledgementType" minOccurs="0" maxOccurs="0" />
   <xsd:element name="ReceiptAcknowledgementException" type="ReceiptAcknowledgementExceptionType" minOccurs="0" maxOccurs="0"
      />
  <xsd:element name="AcceptanceAcknowledgement" type="AcceptanceAcknowledgementType" minOccurs="0" maxOccurs="0" />
    <xsd:element name="AcceptanceAcknowledgementException" type="AcceptanceAcknowledgementExceptionType" minOccurs="0"
      maxOccurs="0" />
    </xsd:sequence>
  <xsd:attribute name="isAuthorizationRequired" type="xsd:boolean" />
  <xsd:attribute name="isIntelligibleCheckRequired" type="xsd:boolean" />
  <xsd:attribute name="timeToAcknowledgeReceipt" type="xsd:duration" />
  <xsd:attribute name="isNonRepudiationReceiptRequired" type="xsd:boolean" />
  <xsd:attribute name="isNonRepudiationRequired" type="xsd:boolean" />
```

```
        </xsd:restriction>
        </xsd:complexContent>
        </xsd:complexType>
        </xsd:element>
        </xsd:sequence>
        </xsd:extension>
        </xsd:complexContent>
        </xsd:complexType>
        </xsd:element>
- <xsd:element name="DataExchange" substitutionGroup="BusinessTransactionHead">
- <xsd:annotation>
  <xsd:documentation>The element is open and allows definition of other patterns unspecified in the concrete set of Business Transaction
      patterns. The DataExchange element was not constrained to support state alignment. The semantics related to DataExchange are
      partner-specific and therefore left unspecified. Note: This element was added in v2.0.</xsd:documentation>
      </xsd:annotation>
- <xsd:complexType>
- <xsd:complexContent>
- <xsd:extension base="BusinessTransactionBaseType">
- <xsd:choice minOccurs="0" maxOccurs="2">
- <xsd:element name="RequestingBusinessActivity" minOccurs="0">
- <xsd:complexType>
- <xsd:complexContent>
- <xsd:restriction base="RequestingBusinessActivityType">
- <xsd:sequence>
  <xsd:element ref="Documentation" minOccurs="0" maxOccurs="unbounded" />
  <xsd:element ref="DocumentEnvelope" />
  <xsd:element name="ReceiptAcknowledgement" type="ReceiptAcknowledgementType" minOccurs="0" />
  <xsd:element name="ReceiptAcknowledgementException" type="ReceiptAcknowledgementExceptionType" minOccurs="0" />
      </xsd:sequence>
  <xsd:attribute name="isAuthorizationRequired" type="xsd:boolean" />
  <xsd:attribute name="isIntelligibleCheckRequired" type="xsd:boolean" />
  <xsd:attribute name="isNonRepudiationRequired" type="xsd:boolean" />
  <xsd:attribute name="isNonRepudiationReceiptRequired" type="xsd:boolean" />
  <xsd:attribute name="timeToAcknowledgeReceipt" type="xsd:duration" />
  <xsd:attribute name="timeToAcknowledgeAcceptance" type="xsd:duration" />
  <xsd:attribute name="retryCount" type="xsd:int" />
      </xsd:restriction>
      </xsd:complexContent>
      </xsd:complexType>
      </xsd:element>
- <xsd:element name="RespondingBusinessActivity" minOccurs="0">
- <xsd:complexType>
- <xsd:complexContent>
- <xsd:restriction base="RespondingBusinessActivityType">
- <xsd:sequence>
  <xsd:element ref="Documentation" minOccurs="0" maxOccurs="unbounded" />
  <xsd:element ref="DocumentEnvelope" minOccurs="0" maxOccurs="unbounded" />
```

```
<xsd:element name="ReceiptAcknowledgement" type="ReceiptAcknowledgementType" minOccurs="0" />
<xsd:element name="ReceiptAcknowledgementException" type="ReceiptAcknowledgementExceptionType" minOccurs="0" />
    </xsd:sequence>
<xsd:attribute name="isAuthorizationRequired" type="xsd:boolean" />
<xsd:attribute name="isIntelligibleCheckRequired" type="xsd:boolean" />
<xsd:attribute name="isNonRepudiationRequired" type="xsd:boolean" />
<xsd:attribute name="isNonRepudiationReceiptRequired" type="xsd:boolean" />
<xsd:attribute name="timeToAcknowledgeReceipt" type="xsd:duration" />
<xsd:attribute name="timeToAcknowledgeAcceptance" type="xsd:duration" />
<xsd:attribute name="retryCount" type="xsd:int" />
    </xsd:restriction>
    </xsd:complexContent>
    </xsd:complexType>
    </xsd:element>
    </xsd:choice>
    </xsd:extension>
    </xsd:complexContent>
    </xsd:complexType>
    </xsd:element>
- <xsd:element name="LegacyBusinessTransaction" substitutionGroup="BusinessTransactionHead">
- <xsd:annotation>
  <xsd:documentation>The previous Business Transaction (now called LegacyBusinessTransaction) was retained for conversions only. The previous Business Transaction element is being replaced by the Business Transaction Head abstract superclass, BusinessTransactionBaseType and the concrete set of Business Transaction Patterns. Note: The Business Transaction (LegacyBusinessTransaction) will be deprecated in a future version. This element was retained in v2.0.x versions.</xsd:documentation>
    </xsd:annotation>
- <xsd:complexType>
- <xsd:complexContent>
- <xsd:extension base="BusinessTransactionBaseType">
- <xsd:sequence>
- <xsd:element name="RequestingBusinessActivity">
- <xsd:complexType>
- <xsd:complexContent>
- <xsd:restriction base="RequestingBusinessActivityType">
- <xsd:sequence>
  <xsd:element ref="Documentation" minOccurs="0" maxOccurs="unbounded" />
  <xsd:element ref="DocumentEnvelope" />
  <xsd:element name="ReceiptAcknowledgement" type="ReceiptAcknowledgementType" />
  <xsd:element name="ReceiptAcknowledgementException" type="ReceiptAcknowledgementExceptionType" />
  <xsd:element name="AcceptanceAcknowledgement" type="AcceptanceAcknowledgementType" minOccurs="0" />
  <xsd:element name="AcceptanceAcknowledgementException" type="AcceptanceAcknowledgementExceptionType" minOccurs="0" />
    </xsd:sequence>
  <xsd:attribute name="isAuthorizationRequired" type="xsd:boolean" />
  <xsd:attribute name="isIntelligibleCheckRequired" type="xsd:boolean" />
  <xsd:attribute name="isNonRepudiationRequired" type="xsd:boolean" />
  <xsd:attribute name="isNonRepudiationReceiptRequired" type="xsd:boolean" />
```

```
<xsd:attribute name="timeToAcknowledgeReceipt" type="xsd:duration" />
<xsd:attribute name="timeToAcknowledgeAcceptance" type="xsd:duration" />
<xsd:attribute name="retryCount" type="xsd:int" />
    </xsd:restriction>
    </xsd:complexContent>
    </xsd:complexType>
    </xsd:element>
- <xsd:element name="RespondingBusinessActivity" minOccurs="0">
- <xsd:complexType>
- <xsd:complexContent>
- <xsd:restriction base="RespondingBusinessActivityType">
- <xsd:sequence>
<xsd:element ref="Documentation" minOccurs="0" maxOccurs="unbounded" />
<xsd:element ref="DocumentEnvelope" maxOccurs="unbounded" />
<xsd:element name="ReceiptAcknowledgement" type="ReceiptAcknowledgementType" />
<xsd:element name="ReceiptAcknowledgementException" type="ReceiptAcknowledgementExceptionType" />
<xsd:element name="AcceptanceAcknowledgement" type="AcceptanceAcknowledgementType" minOccurs="0" />
<xsd:element name="AcceptanceAcknowledgementException" type="AcceptanceAcknowledgementExceptionType" minOccurs="0" />
    </xsd:sequence>
<xsd:attribute name="isAuthorizationRequired" type="xsd:boolean" />
<xsd:attribute name="isIntelligibleCheckRequired" type="xsd:boolean" />
<xsd:attribute name="isNonRepudiationRequired" type="xsd:boolean" />
<xsd:attribute name="isNonRepudiationReceiptRequired" type="xsd:boolean" />
<xsd:attribute name="timeToAcknowledgeReceipt" type="xsd:duration" />
<xsd:attribute name="timeToAcknowledgeAcceptance" type="xsd:duration" />
    </xsd:restriction>
    </xsd:complexContent>
    </xsd:complexType>
    </xsd:element>
    </xsd:sequence>
    </xsd:extension>
    </xsd:complexContent>
    </xsd:complexType>
    </xsd:element>
- <xsd:element name="Signal" type="SignalType">
- <xsd:annotation>
<xsd:documentation>As a Business Action, this element defines the identification structure for Business Signal messages to be sent to a
    trading partner. Note: This element was explicitly added for Business Signals to mirror structure of the Business Document
    specification. This Business Action is non-substantive. A Business Signal is used for state alignment. Note: This element was added in
    v2.0.</xsd:documentation>
    </xsd:annotation>
    </xsd:element>
- <xsd:complexType name="SignalEnvelopeType">
- <xsd:annotation>
<xsd:documentation>The type of a Signal Envelope definition that conveys Business Action information. This type for Business Signals
    mirrors the Document Envelope structure (where applicable). A Business Signal is used for state alignment. Note: This type was
    added in v2.0.</xsd:documentation>
```

```
    </xsd:annotation>
- <xsd:sequence>
  <xsd:element ref="Documentation" minOccurs="0" maxOccurs="unbounded" />
    </xsd:sequence>
  <xsd:attributeGroup ref="name" />
- <xsd:attribute name="signalDefinitionRef" type="xsd:IDREF" use="required">
- <xsd:annotation>
  <xsd:documentation>The nameID reference of the Business Signal definition for the Business Signal. Note: This attribute was added in
      v2.0.</xsd:documentation>
    </xsd:annotation>
    </xsd:attribute>
    </xsd:complexType>
- <xsd:complexType name="SignalType">
- <xsd:annotation>
  <xsd:documentation>The type of a Signal element. A Business Signal is used for state alignment. This construct allows specification
      references (such as those used for context), and a Signal Type may have 0..n Condition Expression. Note: This type was added in
      v2.0.</xsd:documentation>
    </xsd:annotation>
- <xsd:sequence>
  <xsd:element ref="Documentation" minOccurs="0" maxOccurs="unbounded" />
  <xsd:element ref="ConditionExpression" minOccurs="0" maxOccurs="unbounded" />
  <xsd:element ref="Specification" maxOccurs="unbounded" />
    </xsd:sequence>
  <xsd:attributeGroup ref="name" />
    </xsd:complexType>
- <xsd:complexType name="ReceiptAcknowledgementType">
- <xsd:annotation>
  <xsd:documentation>The type of Business Action Business Signal of positive Receipt Acknowledgement. A Business Signal is used for state
      alignment. Note: This type was added in v2.0.</xsd:documentation>
    </xsd:annotation>
- <xsd:complexContent>
- <xsd:extension base="SignalEnvelopeType">
- <xsd:attribute name="isPositiveSignal" type="xsd:boolean" fixed="true">
- <xsd:annotation>
  <xsd:documentation>ReceiptAcknowledgementType specifies whether positive receipt of a Business Signal is required (i.e. the Business
      Signal is not an exception). Note: This attribute was added in v2.0.</xsd:documentation>
    </xsd:annotation>
    </xsd:attribute>
    </xsd:extension>
    </xsd:complexContent>
    </xsd:complexType>
- <xsd:complexType name="ReceiptAcknowledgementExceptionType">
- <xsd:annotation>
  <xsd:documentation>The type of a BusinessAction Business Signal of exception Receipt Acknowledgement. A Business Signal is used for
      state alignment. Note: this type was added in v2.0.</xsd:documentation>
    </xsd:annotation>
- <xsd:complexContent>
```

- <xsd:extension base="**SignalEnvelopeType**">
- <xsd:attribute name="**isPositiveSignal**" fixed="**false**">
- <xsd:annotation>
<xsd:documentation>**For ReceiptAcknowledgementExceptionType, specifies whether positive receipt of a Business Signal is not allowed (i.e. the Business Signal is an exception). Note: This attribute was added in v2.0.**</xsd:documentation>
</xsd:annotation>
</xsd:attribute>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
- <xsd:complexType name="**AcceptanceAcknowledgementType**">
- <xsd:annotation>
<xsd:documentation>**The type of Business Action Business Signal of positive Acceptance Acknowledgement. A Business Signal is used for state alignment. Note: This type was added in v2.0.**</xsd:documentation>
</xsd:annotation>
- <xsd:complexContent>
- <xsd:extension base="**SignalEnvelopeType**">
- <xsd:attribute name="**isPositiveSignal**" fixed="**true**">
- <xsd:annotation>
<xsd:documentation>**For AcceptanceAcknolwedgmentType, specifies whether positive acceptance of a Business Signal is required (i.e. the Business Signal is not an exception) Note: This attribute was added in v2.0.**</xsd:documentation>
</xsd:annotation>
</xsd:attribute>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
- <xsd:complexType name="**AcceptanceAcknowledgementExceptionType**">
- <xsd:annotation>
<xsd:documentation>**The type of a BusinessAction Business Signal of exception Acceptance Acknowledgement. A Business Signal is used for state alignment. Note: This type was added in v2.0.**</xsd:documentation>
</xsd:annotation>
- <xsd:complexContent>
- <xsd:extension base="**SignalEnvelopeType**">
- <xsd:attribute name="**isPositiveSignal**" fixed="**false**">
- <xsd:annotation>
<xsd:documentation>**For AcceptanceAcknowledgementExceptionType, specifies whether positive acceptance of a Business Signal is not allowed (i.e. the Business Signal is an exception). Note: This attribute was added in v2.0.**</xsd:documentation>
</xsd:annotation>
</xsd:attribute>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
- <xsd:complexType name="**GeneralExceptionType**">
- <xsd:annotation>
<xsd:documentation>**The type of a Business Action of general exception (exceptions other than Receipt and Acceptance Acknowledgement). During an interaction, if specified by the parties, the general exception may be used when a party has to trigger an exception, for example, for a general communication of catastrophic failure. A Business Signal is typically used for state**

alignment. As an unforeseen event, this type of Exception is outside of the currently defined concrete BT patterns. Note: this type was added in v2.0.</xsd:documentation>
</xsd:annotation>
- <xsd:complexContent>
- <xsd:extension base="SignalEnvelopeType">
- <xsd:attribute name="isPositiveSignal" fixed="false">
- <xsd:annotation>
<xsd:documentation>For GeneralExceptionType, specifies whether positive receipt of a Business Signal is not allowed (i.e. the Business Signal is an exception). Note: This attribute was added in v2.0.</xsd:documentation>
</xsd:annotation>
</xsd:attribute>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
- <xsd:element name="TimeToPerform">
- <xsd:annotation>
<xsd:documentation>The expected time available to successfully complete a specified activity such as a substantive response to a request. The Time To Perform could be a variable, i.e. it can be specified at different points during the process lifecycle. Note: This element was added (previously was an attribute on specific elements) in v2.0; capabilities were also added for late binding.</xsd:documentation>
</xsd:annotation>
- <xsd:complexType>
- <xsd:sequence>
<xsd:element ref="Variable" minOccurs="0" />
</xsd:sequence>
- <xsd:attribute name="duration" type="xsd:duration" use="optional">
- <xsd:annotation>
<xsd:documentation>The duration of the maximum amount of time between the time at which the request is sent and the substantive response is received. Note: This attribute was added in v2.0.</xsd:documentation>
</xsd:annotation>
</xsd:attribute>
- <xsd:attribute name="type" type="TimeToPerformType" use="optional" default="design">
- <xsd:annotation>
<xsd:documentation>The type as specified by the simpleType of TimeToPerformType. Note: This attribute was added in v2.0.</xsd:documentation>
</xsd:annotation>
</xsd:attribute>
</xsd:complexType>
</xsd:element>
- <xsd:element name="Variable">
- <xsd:annotation>
<xsd:documentation>A semantic element that supports the effective use of conditional constraints. The variable can be accessed by external elements. The businessTransactionActivityRef and businessDocumentRef point to what context and documents are relevant to Condition Expression evaluation. Variable assumes type, if any, from expression evaluation. This element, for example, could be associated with a logical Business Document. Given whether simple or complex variables are used, XPath or XSLT could be used, for example. Note: This complexType was added in v2.0.</xsd:documentation>
</xsd:annotation>

```
- <xsd:complexType>
- <xsd:sequence>
- <xsd:annotation>
  <xsd:documentation>Exactly one ConditionExpression is used to provide values. If multiple ConditionExpressions are listed, each expressionLanguage value is different from others in the sequence and distinct.</xsd:documentation>
    </xsd:annotation>
 <xsd:element ref="ConditionExpression" maxOccurs="unbounded" />
    </xsd:sequence>
 <xsd:attributeGroup ref="name" />
- <xsd:attribute name="businessTransactionActivityRef" type="xsd:IDREF" use="optional">
- <xsd:annotation>
 <xsd:documentation>The nameID reference of the Business Transaction Activity associated with the semantic variable. Note: This attribute was added in v2.0.</xsd:documentation>
    </xsd:annotation>
    </xsd:attribute>
- <xsd:attribute name="businessDocumentRef" type="xsd:IDREF" use="optional">
- <xsd:annotation>
 <xsd:documentation>The nameID reference of the logical business document associated with the semantic variable. Note: This attribute was added in v2.0.</xsd:documentation>
    </xsd:annotation>
    </xsd:attribute>
    </xsd:complexType>
    </xsd:element>
- <xsd:element name="OperationMapping" type="OperationMappingType">
- <xsd:annotation>
 <xsd:documentation>An abstract element that allows mapping a BTA and its BusinessDocuments to Interface/Operation messages. Specifies input, output, fault, interface, operation map to role, BTA and document envelope or Business Signal references. Note: This element was added in v2.0.</xsd:documentation>
    </xsd:annotation>
    </xsd:element>
- <xsd:complexType name="OperationMappingType">
- <xsd:annotation>
 <xsd:documentation>The type related to the abstract element mapping Operations to Business Actions, either business messages or signals. Note: This complexType was added in v2.0.</xsd:documentation>
    </xsd:annotation>
- <xsd:sequence>
- <xsd:element name="MessageMap" maxOccurs="unbounded">
- <xsd:complexType>
 <xsd:attributeGroup ref="map" />
    </xsd:complexType>
    </xsd:element>
- <xsd:element name="SignalMap" minOccurs="0" maxOccurs="8">
- <xsd:complexType>
 <xsd:attributeGroup ref="map" />
    </xsd:complexType>
    </xsd:element>
    </xsd:sequence>
```

```
<xsd:attributeGroup ref="name" />
- <xsd:attribute name="roleRef" type="xsd:IDREF" use="required">
- <xsd:annotation>
<xsd:documentation>Maps to a Role element contained in collaboration.</xsd:documentation>
</xsd:annotation>
</xsd:attribute>
<xsd:attribute name="businessTransactionActivityRef" type="xsd:IDREF" use="required" />
</xsd:complexType>
- <xsd:simpleType name="TimeToPerformType">
- <xsd:annotation>
<xsd:documentation>The simpleType related to the definition of Time To Perform during the process lifecycle. Allows for late binding. Note: This simpleType was added in v2.0.</xsd:documentation>
</xsd:annotation>
- <xsd:restriction base="xsd:NMTOKEN">
<xsd:enumeration value="design" />
<xsd:enumeration value="configuration" />
<xsd:enumeration value="runtime" />
</xsd:restriction>
</xsd:simpleType>
- <xsd:simpleType name="BusinessDocumentValueList">
- <xsd:annotation>
<xsd:documentation>The simpleType related to the (future) enumerated list for the Business Document list associated with the Status Visibility element. This simpleType is used in exposing visibility to documents from more deeply nested BTA. Thoses values are to be available as the expression language listed. Note: This simpleType was added in v2.0.</xsd:documentation>
</xsd:annotation>
<xsd:list itemType="xsd:NMTOKEN" />
</xsd:simpleType>
- <xsd:simpleType name="ConditionGuardValueList">
- <xsd:annotation>
<xsd:documentation>The simpleType related to the content model for Status Visibility. Note: This simpleType was added in v2.0.</xsd:documentation>
</xsd:annotation>
<xsd:list itemType="ConditionGuardValue" />
</xsd:simpleType>
- <xsd:simpleType name="ConditionGuardValue">
- <xsd:annotation>
<xsd:documentation>The simpleType related to the enumerated list for the guard values associated with the FromLink in a Completion State. Each of the FromLinks in a Completion State can be specified to transition as a result of the ConditionGuardValues. This check is made every time a transition occurs from the real states. Because the FromLinks generally are from a specific state, only when that state has been reached is the check of conditionGuards from that state checked. Business Success and Failure relate to the business documents received. While conversely, any timeout is a business protocol failure, i.e. the state is not aligned. Note: This simpleType was added in v2.0.</xsd:documentation>
</xsd:annotation>
- <xsd:restriction base="xsd:NMTOKEN">
<xsd:enumeration value="ProtocolSuccess" />
<xsd:enumeration value="AnyProtocolFailure" />
<xsd:enumeration value="RequestReceiptFailure" />
```

```
<xsd:enumeration value="RequestAcceptanceFailure" />
<xsd:enumeration value="ResponseReceiptFailure" />
<xsd:enumeration value="ResponseAcceptanceFailure" />
<xsd:enumeration value="SignalTimeout" />
<xsd:enumeration value="ResponseTimeout" />
<xsd:enumeration value="BusinessSuccess" />
<xsd:enumeration value="BusinessFailure" />
<xsd:enumeration value="Success" />
<xsd:enumeration value="Failure" />
    </xsd:restriction>
    </xsd:simpleType>
- <xsd:simpleType name="DocumentSpecificationType">
- <xsd:annotation>
  <xsd:documentation>The simpleType related to the enumerated list of specification types for the Specification element. Note: This
      simpleType was added in v2.0.</xsd:documentation>
    </xsd:annotation>
- <xsd:restriction base="xsd:NMTOKEN">
<xsd:enumeration value="schema" />
<xsd:enumeration value="dtd" />
<xsd:enumeration value="wsdl" />
<xsd:enumeration value="relaxng" />
<xsd:enumeration value="schematron" />
<xsd:enumeration value="other" />
    </xsd:restriction>
    </xsd:simpleType>
- <xsd:simpleType name="StepType">
- <xsd:annotation>
  <xsd:documentation>The simpleType related to the enumerated list of operation types supported by OperationMapping and
      OperationMappingType. Note: This simpleType was added in v2.0.</xsd:documentation>
    </xsd:annotation>
- <xsd:restriction base="xsd:NMTOKEN">
<xsd:enumeration value="input" />
<xsd:enumeration value="output" />
<xsd:enumeration value="fault" />
    </xsd:restriction>
    </xsd:simpleType>
- <xsd:simpleType name="ExpressionLanguageType">
- <xsd:annotation>
<xsd:documentation>This simpleType relates to the types of Expression language. An enumerated list is provided. Note: This simpleType was
      added in v2.0.</xsd:documentation>
    </xsd:annotation>
- <xsd:restriction base="xsd:NMTOKEN">
<xsd:enumeration value="ConditionGuardValue" />
<xsd:enumeration value="DocumentEnvelope" />
<xsd:enumeration value="XPath1" />
<xsd:enumeration value="XSLT1" />
<xsd:enumeration value="CAM1" />
```

```
<xsd:enumeration value="XPath2" />
<xsd:enumeration value="XSLT2" />
<xsd:enumeration value="XQuery1" />
    </xsd:restriction>
    </xsd:simpleType>
- <xsd:attributeGroup name="documentSecurity">
- <xsd:annotation>
<xsd:documentation>The attributeGroup related to document security, quality of service attributes. These attributes relate to the BT patterns and the operational semantics surrounding their use.</xsd:documentation>
    </xsd:annotation>
- <xsd:attribute name="isAuthenticated">
- <xsd:annotation>
<xsd:documentation>The communications channel used to transport the Message provides transient authentication. The specific method will be determined by the communications protocol used. Persistent authentication means the Business Document signer's identity is verified at the receiving application level. Authentication assists in verification of role identity of a participating party.</xsd:documentation>
    </xsd:annotation>
- <xsd:simpleType>
- <xsd:restriction base="xsd:NMTOKEN">
<xsd:enumeration value="none" />
<xsd:enumeration value="transient" />
<xsd:enumeration value="persistent" />
<xsd:enumeration value="transient-and-persistent" />
    </xsd:restriction>
    </xsd:simpleType>
    </xsd:attribute>
- <xsd:attribute name="isConfidential">
- <xsd:annotation>
<xsd:documentation>Transient confidentiality is provided by a secure network protocol, such as SSL as the document is transferred between two adjacent ebXML Messaging Service or other transport messaging nodes. Persistent confidentiality is intended to preserve the confidentiality of the message such that only the intended party (application) can see it.</xsd:documentation>
    </xsd:annotation>
- <xsd:simpleType>
- <xsd:restriction base="xsd:NMTOKEN">
<xsd:enumeration value="none" />
<xsd:enumeration value="transient" />
<xsd:enumeration value="persistent" />
<xsd:enumeration value="transient-and-persistent" />
    </xsd:restriction>
    </xsd:simpleType>
    </xsd:attribute>
- <xsd:attribute name="isTamperDetectable">
- <xsd:annotation>
<xsd:documentation>Transient isTamperDetectable is the ability to detect if the information has been tampered with during transfer between two adjacent Message Service Handler nodes. Persistent isTamperDetectable is the ability to detect if the information has been tampered with after it has been received by messaging node, between the messaging node and the application. Tamper detection assists in verification of content integrity between and within a participating party.</xsd:documentation>
```

```
    </xsd:annotation>
- <xsd:simpleType>
- <xsd:restriction base="xsd:NMTOKEN">
  <xsd:enumeration value="none" />
  <xsd:enumeration value="transient" />
  <xsd:enumeration value="persistent" />
  <xsd:enumeration value="transient-and-persistent" />
    </xsd:restriction>
    </xsd:simpleType>
    </xsd:attribute>
    </xsd:attributeGroup>
- <xsd:attributeGroup name="map">
- <xsd:annotation>
   <xsd:documentation>The attributeGroup related to interface/operation types supported by OperationMapping and
      OperationMappingType. The abstract map includes the interface, operation, the step (see StepType) and the document reference.
      Note: This attributeGroup was added in v2.0.</xsd:documentation>
    </xsd:annotation>
- <xsd:attribute name="interfaceName" type="xsd:NMTOKEN" use="required">
- <xsd:annotation>
  <xsd:documentation>Interface is called portType in WSDL 1.1. The name of the interface or portType mapped logically in Operation
      Mapping. Note: This attribute was added in v2.0.</xsd:documentation>
    </xsd:annotation>
    </xsd:attribute>
- <xsd:attribute name="operationName" type="xsd:NMTOKEN" use="required">
- <xsd:annotation>
   <xsd:documentation>The name of the operation mapped in the OperationMapping. Note: This attribute was added in
      v2.0.</xsd:documentation>
    </xsd:annotation>
    </xsd:attribute>
- <xsd:attribute name="operationStep" type="StepType" use="required">
- <xsd:annotation>
  <xsd:documentation>The name of operation step from the Step Type enumeration list for OperationMapping. Note: This attribute was added
      in v2.0.</xsd:documentation>
    </xsd:annotation>
    </xsd:attribute>
- <xsd:attribute name="documentEnvelopeRef" type="xsd:IDREF" use="required">
- <xsd:annotation>
  <xsd:documentation>The nameID reference of the Document Envelope related to the OperationMapping. Note: This attribute was added in
      v2.0.</xsd:documentation>
    </xsd:annotation>
    </xsd:attribute>
    </xsd:attributeGroup>
- <xsd:attributeGroup name="name">
- <xsd:annotation>
  <xsd:documentation>The attributeGroup related to the identification information for most elements. For the name attribute, no white space
      restrictions are enforced. White space is not controlled but left to implementation to trigger faults or exceptions. Note: This group
      was enhanced to include not only name but nameID in v2.0.</xsd:documentation>
```

```
    </xsd:annotation>
- <xsd:attribute name="name" type="xsd:string" use="required">
- <xsd:annotation>
  <xsd:documentation>A designation that may be relevant to a business analyst but is not intended for referencing.</xsd:documentation>
    </xsd:annotation>
    </xsd:attribute>
- <xsd:attribute name="nameID" type="xsd:ID" use="required">
- <xsd:annotation>
  <xsd:documentation>Used for referencing, for example, for identification of elements within an ebBP instance.</xsd:documentation>
    </xsd:annotation>
    </xsd:attribute>
  <xsd:anyAttribute namespace="##other" processContents="lax" />
    </xsd:attributeGroup>
- <xsd:attributeGroup name="optname">
- <xsd:annotation>
  <xsd:documentation>The optname attributeGroup related to the identification information for some elements. This allows name or nameID
      to be optionally used. No white space restrictions are enforced. White space is not controlled but left to implementation to trigger
      faults or exceptions. Note: This group was added in v2.0.</xsd:documentation>
    </xsd:annotation>
  <xsd:attribute name="name" type="xsd:string" />
  <xsd:attribute name="nameID" type="xsd:ID" />
  <xsd:anyAttribute namespace="##other" processContents="lax" />
    </xsd:attributeGroup>
- <xsd:attributeGroup name="quality">
- <xsd:annotation>
   <xsd:documentation>The attributeGroup related to quality of service attributes. These attributes relate to the BT patterns and the
      operational semantics surrounding their use. Note: This attributeGroup was added in v2.0.</xsd:documentation>
    </xsd:annotation>
- <xsd:attribute name="isAuthorizationRequired" type="xsd:boolean">
- <xsd:annotation>
  <xsd:documentation>When a party uses isAuthorizationRequired on a Requesting and/or a Responding activity accordingly, the result that
      [the activity] will only be processed as valid if the party interpreting it successfully matches the stated identity of the activity's
      [Role] to a list of allowed values previously supplied by that party. Authorization typically relates to a signed business document
      and the association to the role identity of the party expected for that activity.</xsd:documentation>
    </xsd:annotation>
    </xsd:attribute>
- <xsd:attribute name="isIntelligibleCheckRequired" type="xsd:boolean">
- <xsd:annotation>
  <xsd:documentation>Allows partners to agree that a message is confirmed by a Receipt Acknowledgement only if it is also legible. Legible
      means that it has passed structure/schema validity check. The content of the receipt and the legibility of a business message (if
      required) are reviewed prior to the processing of the Business Document or the evaluation of Condition Expressions in the business
      message's Business Documents or Document Envelope.</xsd:documentation>
    </xsd:annotation>
    </xsd:attribute>
- <xsd:attribute name="isNonRepudiationRequired" type="xsd:boolean">
- <xsd:annotation>
```

```
<xsd:documentation>If non-repudiation of origin and content is required, then the Business Activity stores the business document in its original form for the duration mutually agreed to in an agreement.</xsd:documentation>
</xsd:annotation>
</xsd:attribute>
- <xsd:attribute name="isNonRepudiationReceiptRequired" type="xsd:boolean">
- <xsd:annotation>
<xsd:documentation>Both parties agree to mutually verify receipt of a Requesting Business Document and that the receipt is non-repudiable.</xsd:documentation>
</xsd:annotation>
</xsd:attribute>
- <xsd:attribute name="timeToAcknowledgeReceipt" type="xsd:duration">
- <xsd:annotation>
<xsd:documentation>The time a Responding or Requesting role has to acknowledge receipt of a Business Document.</xsd:documentation>
</xsd:annotation>
</xsd:attribute>
- <xsd:attribute name="timeToAcknowledgeAcceptance" type="xsd:duration">
- <xsd:annotation>
<xsd:documentation>The time a Requesting and Responding role has to non-substantively acknowledge business acceptance of a Business Document.</xsd:documentation>
</xsd:annotation>
</xsd:attribute>
- <xsd:attribute name="retryCount" type="xsd:int">
- <xsd:annotation>
<xsd:documentation>The business retry for a RequestingBusinessActivity identifies the number of retries allowed in addition to the initial request while the Time To Perform has not been exceeded.</xsd:documentation>
</xsd:annotation>
</xsd:attribute>
</xsd:attributeGroup>
</xsd:schema>
```