CENTRO DE INVESTIGACIÓN Y DE ESTUDIOS AVANZADOS DEL I.P.N.

DEPARTAMENTO DE INGENIERÍA ELÉCTRICA
SECCIÓN DE COMPUTACIÓN

Diseño y Construcción de un Intermediario para Comercio Electrónico B2B

Tesis que presenta el

M. en C. Giner Alor Hernández

para obtener el Grado de

Doctor en Ciencias en la Especialidad de Ingeniería Eléctrica Opción Computación

Director de la Tesis

Dr. José Oscar Olmedo Aguirre

MEXICO, D.F. A 2005.

Agradecimientos

A mi asesor de tesis, Dr. José Oscar Olmedo Aguirre, quien aporto excelentes ideas a esta tesis y por el apoyo otorgado en todo este tiempo que fue mi asesor.

A mis revisores de tesis, Dr. Pedro Mejía Álvarez, Dr. Jorge Buenabad Chávez, Dr. Adolfo Guzman Arenas y Dr. Miguel Angel Leon Chavez, quienes me ofrecieron su apoyo para revisar mi tesis y me señalaron errores que me ayudaron a mejorarla.

A Sofía Reza, quien es secretaria de la Sección de Computación, por dedicarme bastante tiempo para auxiliarme de excelente manera en cuestiones administrativas.

A los compañeros administrativos de la biblioteca del departamento de Ingeniería Eléctrica, quienes me auxiliaron de buena manera cuando necesité encontrar trabajos de investigación.

A todas aquellas personas quienes me brindaron su apoyo y su amistad en cada momento.

Al Centro de Investigación y de Estudios Avanzados del IPN, CINVESTAV-IPN, junto con la Sección de Computación del Departamento de Ingeniería Eléctrica, por ofrecerme las instalaciones y recursos necesarios para realizar este trabajo de investigación.

Al Consejo Nacional de Ciencia y Tecnología, CONACyT, por el apoyo económico otorgado en el periodo Enero 2002 - Diciembre 2005.

Dedicatorias

A *Dios*, divino ser quien me sigue apoyando en los pasos que doy en mi vida. Toda la felicidad y los beneficios que he recibido en mi vida te los debo sin duda alguna a ti *Dios*. No ha habido ocasión en que no estés conmigo. Gracias Dios mío por estar aquí siempre.

A mis amados padres Rosario Hernández Ortiz y Fidencio Alor Alor quienes han sido el motor de mi vida. Y de quienes me siento tremendamente orgulloso por su tenacidad ante ella. A ustedes les doy gracias por todos sus cuidados y porque siempre creyeron en mí. Les dedico este trabajo porque es algo que sin sus desvelos no hubiera podido ser.

A mis hermanos Janet y Juan Antonio Alor Hernández quienes en todo momento me apoyaron y motivaron para concluir mi tesis. Y quienes son parte de lo más valioso que la vida me ha dado.

A mi abuelitos, porque estén donde estén, sé que están contentos. Gracias por haber sido tan alegres durante mi niñez y parte de mi adolescencia. A ustedes Rafaela y Daniel, les dedico este trabajo como reconocimiento en su labor de padres y abuelos que fueron.

A mi cuñado Jorge Juan, quien es parte de la familia y considero como un hermano más y por estar junto a mi hermana.

A mi sobrina Monserrat Antonio Alor, con quien me divierto y recuerdo lo especial y necesario que es jugar y sonreir. Además me recuerda mi infancia junto a mis hermanos y amigos.

Índice General

Índice de Figuras

Capítulo 1 Motivación

Capítulo 2 Cómputo Orientado a Servicios

Capítulo 3 Sistema de Integración de Procesos de Negocio en la Cadena de Suministro

Lista de Figuras

Resumen

Una cadena de suministro es una red de distribución para la procuración de materiales que pueden transformarse en productos terminales para distribuirse a clientes finales. La procuración de materiales se puede realizar a través de diversos canales de distribución. Uno de los problemas fundamentales de la cadena de suministro es su integración con los procesos de negocio relevantes a la procuración oportuna. Esto se debe a que los procesos de negocio están generalmente construidos bajo diferentes tecnologías y residen en diversos ambientes de ejecución. Así también, las fuentes de información pueden residir en diversas organizaciones y autoridades de confianza.

Esta tesis propone un sistema de intermediación basado en servicios Web que facilita la integración de los procesos de negocio relevantes con una cadena de suministro. El sistema llamado BPIMS-WS, ofrece la procuración de productos en escenarios de la cadena de suministro usando la tecnología de servicios Web. La tecnología de servicios Web sigue los principios de SOA (*Service-Oriented Architecture*) para el desarrollo y despliegue de aplicaciones. El paradigma SOA incorpora la automatización de procesos y el intercambio automatizado de información entre organizaciones. Así también, BPIMS-WS incorpora características del paradigma EDA (*Event-Driven Architecture*) permitiendo la habilidad de supervisar, filtrar, analizar, correlacionar y responder a eventos en tiempo real. Por consiguiente, BPIMS-WS provee una arquitectura SOA combinada con una EDA, proporcionando las características necesarias para la integración de una cadena de suministro, cuyos procesos pueden supervizarse y administrarse.

Como contribuciones de este trabajo, BPIMS-WS proporciona un conjunto de mecanismos para la administración de patrones de procesos de negocio, monitoreo basado en diagramas UML de secuencia, administración basada en servicios Web, publicación/suscripción de eventos y un servicio de mensajería confiable para la continua operación de una cadena de suministro.

Abstract

A supply chain is a distribution network for procurement of materials both raw and finished that can be transformed into finished goods to be distributed to the end customer. This procurement of materials is carried out through various distribution channels. The business processes integration in supply chain is a critical issue. This is due so that business processes have been built using different technologies, and run in different execution environments. Also, information sources can reside in different organizations and trust authorities.

In this thesis, we propose a Web service-based system that offers a brokering service to facilitate the business processes integration in supply chains. Our brokering service named BPIMS-WS offers the procurement of products in SCM scenarios by using current Web services technology. Web services technology follows the SOA's (Service-Oriented Architecture) principles for developing and deploying applications. SOA development paradigm incorporates process automation and automated exchange of information between organizations. Furthermore, BPIMS-WS incorporates features from EDA paradigm allowing the ability to monitor, filter, analyze, correlate, and respond in real time to events. Therefore, BPIMS-WS provides a SOA combined with EDA architecture, providing the ability to create a SCM architecture that enables business.

As salient contributions, BPIMS-WS provides a set of mechanisms for business processes pattern management, monitoring based on UML sequence diagrams, Web services-based management, events publish/subscription and reliable messaging service for the good operation of the supply chain.

Capítulo 1
Motivación

Una cadena de suministro es un ambiente complejo debido a que frecuentemente se incluyen recursos de múltiples proveedores y plataformas. Uno de los problemas fundamentales de la cadena de suministro es la integración de los procesos de negocio involucrados en la procuración electrónica. En este capítulo se realiza un análisis de cómo las diferentes tecnologías que han surgido han propuesto un enfoque de solución para este problema de integración. Se abordan sus características principales, sus ventajas y desventajas. Con base a sus limitaciones de cada una, se define una propuesta de solución para lograr la integración la cadena de suministro. Finalmente, se definen las contribuciones principales del enfoque de solución propuesta.

1.1. Introducción

Una cadena de suministro es un modelo que coordina los procesos administrativos de proveedores, plantas y centros de almacenamiento y distribución, de manera que los bienes sean producidos y distribuidos en las cantidades adecuadas en los lugares y en los tiempos correctos. El objetivo principal de la cadena de suministro es minimizar el costo de operación y satisfacer la calidad de los productos y servicios ofrecidos.

Hace algunos años, la única manera de poder obtener toda la información necesaria para el óptimo manejo de la cadena de suministro, era mediante la integración vertical de todos los pasos involucrados en la misma. Hoy en día, la tecnología de Internet no sólo hace posible que una cadena de suministro pueda coordinarse en forma distribuida, sino que permite además que distintas porciones de la misma cadena se realicen por empresas completamente independientes. Un claro ejemplo de esto lo constituyen las empresas de

transporte que se adhieren con sus sitios en Internet al reparto de productos de cualquier compañía.

Una de las tecnologías de Internet que permite la integración de procesos de negocio en la cadena de suministro son los servicios Web. Los servicios Web proveen un enfoque basado en estándares de Internet para implementar componentes distribuidos ya que ofrecen mecanismos necesarios para la integración de aplicaciones con el uso de protocolos como HTTP, SOAP y XML [1]. El *Stencil Group* propuso una definición ampliamente utilizada de servicios Web: "Los servicios Web son componentes de software re-usables y bajamente acoplados que encapsulan su funcionalidad en una forma semántica, y que además están distribuidos y accesibles a través de protocolos estándares de Internet" [2]. El término "bajamente acoplado" (*loosely coupled*) implica que un servicio Web es independiente de la plataforma de soporte, del lenguaje de programación y del modelo de programación utilizado. Debido a la ubicuidad y al bajo costo de Internet, los servicios Web proveen interoperabilidad en Internet e incluso en una Intranet. Desde el punto de vista de las arquitecturas de sistemas distribuidos, los servicios Web proveen una arquitectura basada en componentes y orientada a servicios. Desde un punto de vista orientado a negocios, los servicios Web son la tecnología que se utiliza para construir nuevos modelos de comercio electrónico. Los procesos involucrados en estos nuevos modelos están compuestos de tareas las cuales están distribuidas a través de una red de servicios de valor agregado y se representan como servicios Web. Por consiguiente, el desarrollo de aplicaciones basadas en servicios Web reduce costos de integración mejorando los niveles de servicios y extendiendo los límites de la empresa que mediante otros mecanismos no serían posibles [3]. A continuación se da una breve historia del impacto de las tecnologías de información en la cadena de suministro.

1.2. Relación y Evolución entre Cadena de Suministro y Tecnologías de Información

Una cadena de suministro comúnmente involucra múltiples empresas incluyendo proveedores, fabricantes, centros de almacenamiento y transportación, clientes minoristas y mayoristas [4]. A continuación, se discute la evolución de las cadenas de suministro que

van desde las primeras cadenas estáticas basadas en el intercambio electrónico de datos (EDI) hasta las actuales cadenas dinámicas basadas en servicios Web.

1.2.1. EDI (*Electronic Data Interchange*)

El comercio electrónico, como intercambio electrónico de datos (EDI), se originó en los Estados Unidos en los 60's con iniciativas independientes en los sectores de ferrocarriles, comestibles y fábricas de automóviles. Se diseñó para asegurar la calidad de los datos que dichos sectores intercambiaban en la cadena de suministro así como para sus procedimientos internos. En los 70's, la transferencia electrónica de fondos (TEF) a través de redes de seguridad privadas dentro de las instituciones financieras expandió el uso de las tecnologías de telecomunicación para propósitos comerciales, permitiendo el desarrollo del intercambio de información operacional comercial en el área financiera, específicamente la transferencia de giros y pagos. El EDI usa documentos electrónicos con formato estándar que reemplazan los documentos comerciales comunes, tales como preparación de facturas, datos de embarque, órdenes de compra, cambios en órdenes de compra, requerimientos de cotizaciones y recepción de avisos, los cuales son los 6 tipos más comunes de documentos comerciales que constituyen el 85% de las transacciones comerciales oficiales en los Estados Unidos [5]. A pesar del tiempo transcurrido, EDI todavía se utiliza por las empresas con el fin de reducir los costos de transacción con sus socios comerciales. Incluso hace algunos años, grandes compañías integraron sus procesos de negocio usando EDI por lo que dieron origen a grandes sistemas de integración de la cadena de suministro [6, 7, 8, 9]. Sin embargo, estas aplicaciones no eran eficaces debido a que utilizaban el procesamiento por lotes como mecanismo de transmisión de documentos EDI lo que hacía difícil la consistencia de datos a lo largo de las diferentes fases de la cadena de suministro. Esta falta de sincronización se debe principalmente a que las empresas cambian constantemente la funcionalidad de sus procesos de negocio [10]. Por ejemplo, algunos procesos de producción pueden cambiar su plan de operación para mejorar la eficacia ajustando los precios y promociones de los productos con el fin de tener mejores oportunidades de mercado dentro de la cadena de suministro. Incluso, el diseño y las características técnicas de los productos cambian constantemente con base a la información proveniente de los proveedores. Como una solución a esto, la mayoría de las compañías

adoptaron las redes de valor agregado (VAN) para proveer los servicios de transmisión de datos, pero para ello era necesario el desarrollo de traductores EDI tanto en la parte receptora como en la emisora. En las empresas con base tecnológica, la ausencia de socios comerciales implica pérdidas sustanciales en la integración cadena de suministro, por lo que a través de los años se han desarrollado diversas tecnologías de información para las PYMES. Con el creciente uso de Internet, los costos de transmisión de documentos basados en EDI se han ido reduciendo [11]. Sin embargo, actualmente el desarrollo de traductores EDI todavía es costoso porque los estándares EDI para la codificación de documentos son complicados debido a su énfasis en la consistencia de los datos.

EDI fue propuesto por organizaciones comerciales con el propósito de intercambiar información. Desafortunadamente, en esta propuesta no se consideraron los desarrollos alcanzados en el área de los Sistemas Distribuidos de Información, la cual es probablemente el origen de la mayoría de los problemas mencionados antes. Una alternativa a esta propuesta que elimina varios de estos problemas, fue la iniciativa desarrollada por organizaciones con base tecnológica en computación conocida como CORBA.

1.2.2. CORBA (Common Object Request Broker Architecture)

CORBA es una especificación del OMG (*Object Management Group*) [12] para lograr interoperabilidad en nodos de cómputo distribuidos. CORBA permite a aplicaciones hacer peticiones y recibir respuestas en un entorno distribuido, lo cual es la base para construir aplicaciones distribuidas basadas en objetos y llevar a cabo su integración en entornos heterogéneos [13].

CORBA es un bus de comunicaciones para aplicaciones heterogéneas y automatiza tareas habituales en sistemas distribuidos como: registro, localización y activación de objetos, gestión de errores, multiplexación y demultiplexación de invocaciones, por mencionar algunas. El principal componente de CORBA es el Intermediario de Solicitudes a Objetos (*Object Request Broker, ORB*) que proporciona mecanismos de comunicación entre objetos y se encarga de los detalles de sus invocaciones remotas, realizando funciones de localización y de activación de objetos cuando se reciben invocaciones para ellos.

Sin embargo, CORBA no ofrece interoperabilidad para los procesos de negocio en una cadena de suministro. Esto se debe principalmente a que cada socio comercial (representado como un nodo en CORBA) debe ejecutar su Intermediario de Solicitudes de Objeto, el cual en la práctica es altamente dependiente de la implementación comercial de CORBA. Esto da lugar a un gran problema de interoperabilidad en el desarrollo de las actividades comerciales entre los participantes de la cadena de suministro.

Para resolver el problema de la interoperabilidad entre las implementaciones de CORBA, Microsoft propuso un enfoque alternativo conocido DCOM.

1.2.3. DCOM (Distributed Component Object Model)

DCOM (*Distributed Component Object Model*) es la extensión a COM (*Component Object Model*). Como DCOM es una evolución lógica de COM, se pueden utilizar los componentes creados en aplicaciones basadas en COM, y trasladarlas a entornos distribuidos. DCOM es un conjunto de conceptos e interfaces programables de Microsoft en el cual los objetos de un programa cliente pueden solicitar servicios de objetos de un programa servidor que se encuentren en otros nodos de cómputo [14]. DCOM define cómo los componentes y sus clientes pueden interactuar entre sí. Esta interacción permite que el cliente y el componente se conecten sin la necesidad de un sistema intermedio.

Sin embargo, esta tecnología de información presenta ciertas carencias de interoperabilidad en la cadena de suministro. Todos los procesos de negocio de los socios comerciales participantes en una cadena de suministro deben ejecutarse bajo la plataforma Windows. Además, en DCOM los mensajes que se envían entre un cliente y un servidor tienen un formato definido por el protocolo DCOM Object RPC (ORPC), por lo cual es necesario utilizar mecanismos que traduzcan los mensajes para que un sistema distinto pueda interpretar y actuar en las peticiones y en las respuestas de estos.

A pesar de que DCOM resuelve los problemas de interoperabilidad causados por las incompatibilidades entre las implementaciones desarrolladas por diferentes proveedores de tecnología, aún persisten los problemas de interoperabilidad que surgen cuando diversas plataformas operativas están involucradas. El lenguaje de programación Java se diseñó con el fin de resolver estos problemas de interoperabilidad.

1.2.4. RMI (Remote Method Invocation)

RMI (*Remote Method Invocation*) es un modelo de cómputo distribuido que surgió después que CORBA y DCOM. Es similar a CORBA y DCOM, pero trabaja sólo con objetos de Java [15]. Java/RMI se basa en un protocolo llamado JRMP (*Java Remote Method Protocol*) el cual usa la serialización de objetos en Java que permite transmitirlos como una cadena de caracteres. Para realizar esto, el cliente realiza una llamada a un método de un componente del servidor, serializando sus parámetros de entrada. Después, el servidor deserializa la información y reconstruye el objeto correspondiente. Luego, el servidor procesa la llamada y devuelve la respuesta siguiendo los mismos mecanismos de serialización y deserialización usados en la invocación.

Sin embargo al igual que las tecnologías de información mencionadas anteriormente, Java/RMI presenta dificultades de interoperabilidad. Estas dificultades residen en el uso de la serialización de objetos la cual es específica de Java. En el contexto de la cadena de suministro, esto significa que los procesos de negocio deben estar escritos en Java. Por lo cual hay una dependencia del lenguaje de programación. Para resolver el problema de la dependencia del lenguaje de programación, el consorcio de la Web (W3C) propuso un lenguaje de descripción independiente de la plataforma operativa conocido como XML.

1.2.5. XML (eXtensible Markup Language)

Por sus siglas en inglés, XML significa lenguaje de marcado extensible y es un lenguaje estándar para el intercambio de datos se desarrolló a finales de los 90's. Su característica de ser extensible se debe a que no tiene un formato fijo como EDI o HTML. Hay tres niveles de abstracción relacionadas con XML [16]:

1. Lenguajes XML de definición, los cuales tienen 2 estándares: XML 1.0 *Recommendation* [17] y XML *Schema* [18]
2. Estándares de dominio para documentos comerciales definidos en XML DTD o XSD
3. Tipos de documentos comerciales que están codificados en formato XML.

Adicionalmente, XML tiene soporte para estándares como XSL [19], XML DOM [20] y SAX [21] los cuales se utilizan para la transformación, traducción y validación de

documentos XML, respectivamente. Existen algunos beneficios con el uso de XML en vez de EDI, como los que se enumeran a continuación:

1. **Adopción de Unicode:** XML utiliza Unicode para la codificación en diferentes idiomas
2. **Comprobación de errores:** Los archivos XML DTD y XSD son básicamente definiciones de lenguajes de marcado. Ellos definen las estructuras y restricciones de documentos específicos y por consiguiente, pueden utilizarse para capturar y validar documentos comerciales lo cual es una función crítica para el comercio electrónico B2B y para la cadena de suministro
3. **Uso de herramientas de análisis sintáctico:** Muchas herramientas para el análisis de documentos XML están públicamente disponibles en diversos lenguajes de programación y se encuentran integradas con diferentes ambientes de desarrollo como .NET y J2EE

Diversas empresas han desarrollado grandes iniciativas basadas en XML para la planeación, previsión y reaprovisionamiento conjunto (*Collaborative Planning, Forecasting, and Replenishment, CPFR*) [22, 23, 24]. Sin embargo, estas iniciativas utilizan XML para representar documentos comerciales basados pero que siguen usando el mismo modelo de procesamiento por lotes que EDI. Como consecuencia, los datos de los socios comerciales pueden no estar disponibles en la toma de decisiones en tiempo real. Así que tanto EDI como XML ofrecen soluciones parciales para la integración de cadenas de suministro en una forma dinámica, particularmente para la representación y el intercambio de información.

1.2.6. Servicios Web

Los servicios Web utilizan XML como formato de datos y a SOAP (*Simple Object Access Protocol*) como mecanismo de transmisión de documentos XML para lograr interoperabilidad entre componentes de software. Los servicios Web se usan para desarrollar nuevos componentes de software o para construir un puente de comunicación con los sistemas internos de los socios comerciales en la cadena de suministro con el fin de exponer sus procesos de negocio de manera externa. También, los servicios Web pueden

utilizarse internamente para proveer interfaces programables a sistemas ya existentes y para integrar aplicaciones Web con estos sistemas [25, 26].

EDI y XML son dos enfoques que se basan en el intercambio de datos para integrar los procesos de negocio en la cadena de suministro, por lo que los datos se envían a los socios comerciales para que ellos los interpreten y tomen alguna decisión o acción al respecto. Incluso, con la función de prevención conjunta en CPFR, los datos se envían en forma bidireccional entre clientes y proveedores, algunas veces con modificaciones en los documentos XML para que los socios comerciales puedan actualizar sus archivos con información proveniente de alguna fase de la cadena de suministro. XML es ampliamente utilizado para el intercambio de datos bajo SOAP. Sin embargo, los servicios Web proveen un enfoque orientado a servicios y procesos, con el fin de solucionar algunos problemas de sincronización en la cadena de suministro por lo que aplicaciones basadas en servicios Web ofrecen mecanismos de integración con sistemas existentes de una empresa (por ejemplo verificación de los niveles de inventario) y procesos de negocio de proveedores (por ejemplo solicitud de precio de un producto).

SOAP, un servicio de mensajería para los servicios Web, sigue el modelo solicitud/respuesta de mensajes HTTP. Bajo este enfoque, los datos comerciales de la cadena de suministro se intercambian periódicamente permitiendo el acceso a información actualizada. Actualmente, la mayoría de las aplicaciones basadas en servicios Web son estáticas en el sentido de que usan o realizan invocaciones a servicios Web en tiempo de diseño más no en tiempo de ejecución. Por lo que hay una gran necesidad de sistemas dinámicos de integración de la cadena de suministro ya que las condiciones de mercado cambian constantemente. Por ejemplo, los niveles de inventario de un producto pueden incrementarse con base a la producción o decrecer con base a las compras solicitadas. Mediante el uso de servicios Web, un proveedor puede solicitar el nivel de inventario de un cliente para poder reabastecerlo de un producto de forma oportuna. Por lo que un enfoque de integración dinámica en la cadena de suministro basado en servicios Web es imperativo

1.3. Planteamiento del Problema

El desarrollo de las tecnologías de información ha permitido mejorar la comunicación dentro y fuera de las organizaciones de una manera significativa. Un objetivo importante de las empresas es lograr la integración y colaboración con sus proveedores. En este sentido, las tecnologías de información son parte fundamental de la nueva administración de negocios, sistemas que permiten la planeación, organización, comunicación e integración de los procesos y datos internos de la empresa (*Enterprise Resource Planning*), sistemas para la administración de la cadena de suministros (*Supply Chain Managment*), y de administración de las relaciones del cliente (*Customer Relationship Management*). La procuración electrónica (*e-procurement*) de productos y servicios es una de las diversas aplicaciones que han surgido para complementar las tecnologías de información anteriores y robustecer nuevos esquema de negocios electrónicos (*e-business*) utilizando las ventajas de la infraestructura de Internet.

De acuerdo a lo anterior, el problema que se plantea en esta tesis es la definición de una arquitectura de integración que permita la automatización de procesos de negocio internos y externos relacionados con la procuración electrónica en la cadena de suministro. Este problema es importante porque una solución eficaz provee un conjunto de beneficios entre los que destacan:

- **Reducción de la mano de obra:** Tradicionalmente, la gestión de pedidos ha sido un proceso comercial muy laborioso con numerosos representantes de ventas al cliente, tomando el pedido de un cliente por teléfono y fax, verificando el crédito de forma manual y codificando la información del pedido en los sistemas de gestión de pedidos. La automatización de estos procesos comerciales elimina gran parte de intervención humana, lo cual reduce el número de personal necesario para realizar esas actividades.
- **Reducción de errores:** Los procesos manuales con múltiples puntos de contacto de personal humano son muy susceptibles a los errores debido a la inserción manual de los datos. La automatización del proceso comercial reduce puntos de contacto que como consecuencia conlleva a la disminución de errores.
- **Reducción en el tiempo de ejecución:** Una vez que se establece un sistema que automatiza los procesos de negocio, los proveedores de la empresa responden mejor a

las necesidades del comprador final. La información que se reúne en el punto de venta puede cotejarse automáticamente y estar disponible para su estudio. Esto reduce los tiempos de ejecución y ayuda a mejorar la calidad y el diseño de los productos.

- **Disminución de costos en los procesos:** Debido a la integración de los proveedores en la cadena de suministro se redefinen los procesos para conseguir una mayor eficiencia y eficacia, y por tanto, la consecuente disminución de costos. Los procesos más afectados son los relacionados con el intercambio de información y documentación con proveedores, gestión de inventarios, gestión de pedidos, por mencionar solo algunos.
- **Reducción de costos de compra debido a la eliminación de intermediarios físicos:** Debido a la facilidad de intercambio de información entre las distintas empresas, puede redefinirse la cadena de suministro, lo que conlleva a la eliminación de intermediarios físicos que en la mayoría de los casos, retrasan el tiempo de entrega del producto/servicio.
- **Ampliación del número de proveedores potenciales y disminución del tiempo de localización:** El comprador tiene acceso rápido y económico a gran cantidad de proveedores potenciales tanto a nivel nacional como internacional, teniendo mucha información adicional sobre los mismos, lo que le posibilita su fácil localización y evaluación.
- **Disminución del tiempo de aprovisionamiento:** Debido a la facilidad de interacción con los proveedores disponibles para un producto/servicio determinado y la redefinición de los procesos, se reduce importantemente el tiempo de adquisición del producto/servicio.

Lo anterior se logra al mejorar la comunicación entre proveedores y clientes utilizando un mecanismo de integración para conocer las necesidades de cada uno y realizar los ajustes necesarios al interior de su organización para cumplir sus expectativas.

No obstante, para poder abordar este problema es necesario tomar en cuenta las siguientes consideraciones:

- Cada uno de los procesos de negocio de estos participantes pueden estar construidos bajo diferentes tecnologías y residir en diversos ambientes de ejecución.

- Las fuentes de información pueden residir en diversas organizaciones y autoridades de confianza.

Con base en estas consideraciones, considere el siguiente ejemplo. Suponga que un cliente desea buscar y comprar sus productos por Internet para reabastecerse oportunamente. Para lograr su propósito necesita acceder a los sitios Web de los diferentes proveedores y buscar cuál de todos ofrece los productos que necesita al precio que desea. Lo que parecía ser una tarea relativamente simple, se convierte en una pérdida de tiempo durante la búsqueda y desemboca en una serie de molestias para él. Para lograr la automatización de todos estos pasos es necesario un mecanismo de integración. Mediante este mecanismo, el cliente únicamente tendría que hacer una sola petición, lo que se traduciría en un ahorro significativo de tiempo al efectuar todos esos pasos (habiendo consultado diversos proveedores) ya que reduce de manera significativa el número de conexiones con los proveedores. Suponga el caso de que ahora son n el número de clientes que desean reabastecerse de un producto y m el número de proveedores que pueden satisfacer las demandas de los clientes. Sin un mecanismo de intermediación, el numero de conexiones sería n*m ya que cada cliente debe establecer una conexión con cada proveedor. Sin embargo, con un mecanismo de intermediación el numero de conexiones se reduce a n+m únicamente estableciendo todas las conexiones de los clientes y proveedores.

En la procuración electrónica, la integración es consecuencia de una buena comunicación entre proveedor y cliente. En este contexto, los servicios Web son una tecnología que facilita la integración debido a que permiten el intercambio electrónico de datos. Por consiguiente, aplicaciones de procuración electrónica basadas en servicios Web permiten la integración de diversas empresas con sus proveedores, facilitando la administración de un segmento importante de la cadena de suministro y colaborando con la automatización de procesos de procuración de materiales y los procesos de intercambio de información interna y externa.

1.4. Solución Propuesta

Para resolver el problema descrito anteriormente en este trabajo se propone una arquitectura orientada a servicios cuyo diseño esté organizado por capas y que brinde interoperabilidad y dinamismo en la integración de procesos de negocio de las empresas.

Para alcanzar esta solución, el objetivo principal de este trabajo es diseñar e implementar un sistema de intermediación de comercio electrónico B2B. El sistema de intermediación se basa en la tecnología de servicios Web que facilita la integración de los procesos de negocio relevantes a la procuración oportuna en una cadena de suministro. El sistema es capaz de descubrir, integrar, orquestar, supervisar y administrar los procesos de negocio de las empresas. Los procesos de negocio se describen como servicios Web y su comportamiento se describe por ontologías de servicios con el fin de optimizar la fase de procuración oportuna de la cadena de suministro.

Este trabajo presenta los siguientes objetivos específicos:

- Definir una arquitectura basada en capas para la integración de procesos de negocio referentes a la procuración oportuna en la cadena de suministro
- Identificar los componentes y sus interrelaciones de la arquitectura de integración
- Definir los niveles de prestación de servicio en cada capa de la arquitectura de integración
- Definir mecanismos de coordinación para el descubrimiento, integración, orquestación, administración y supervisión de servicios Web en la arquitectura de integración
- Validar la arquitectura mediante la implementación de un sistema de intermediación y la integración a él mediante un modelo de organización virtual para el comercio electrónico B2B

1.5. Contribuciones

Las contribuciones de este trabajo son:

1) Diseño de una arquitectura híbrida de coordinación distribuida para compartir información entre proveedores y compradores por intervención de un mediador con

soporte para colaboraciones comerciales en el comercio electrónico B2B y en la cadena de suministro

2) Diseño e implementación de un administrador de procesos de negocio genéricos descritos en BPEL4WS
3) Un mecanismo para supervisar procesos de negocio involucrados en las colaboraciones comerciales en el comercio electrónico B2B y en la cadena de suministro a través de diagramas UML de secuencia
4) Un mecanismo de administración mediante servicios Web para controlar y configurar procesos de negocio
5) Un mecanismo de publicación/suscripción de eventos para incorporar información en los procesos de negocio a través de su ocurrencia
6) Un servicio de mensajería para garantizar la entrega de información involucrada en las colaboraciones comerciales de manera segura para la operación continua de la cadena de suministro

La tecnología de servicios Web sigue los principios de las arquitecturas orientadas a servicios (*Service-Oriented Architecture, SOA*) para el desarrollo y despliegue de aplicaciones. El paradigma SOA incorpora la automatización de procesos y el intercambio automatizado de información entre organizaciones. Por otra parte, las arquitecturas manejadas por eventos (*Event-Driven Architecture, EDA*) permiten la habilidad de supervisar, filtrar, analizar, correlacionar y responder a eventos en tiempo real. Por consiguiente, una arquitectura SOA combinada con una EDA proporciona las características necesarias para la integración de una cadena de suministro, cuyos procesos pueden supervisarse y administrarse.

En la cadena de suministro, los procesos de negocio consisten en un conjunto de actividades determinadas en tiempo de diseño por algún experto en la generación de flujos de trabajo (*workflows*). Este *workflow* está dirigido a la solución de un problema específico. En este sentido, los procesos genéricos de negocio ofrecen una solución más general para un problema específico. En ellos se abstraen las actividades recurrentes de un proceso de negocio y en tiempo de ejecución se completan para resolver un problema en particular. Bajo este contexto, se puede ver a un proceso de negocio genérico como una lista de

actividades definidas en tiempo de diseño a las que sólo les hace falta incluir cuáles y cuántos servicios Web son agregados en tiempo de ejecución.

La supervisión de procesos de negocio es un factor importante en la cadena de suministro. A través del proceso de supervisión, los socios involucrados en un proceso de negocio pueden dar seguimiento y verificar el desarrollo de las actividades comerciales. A través de diagramas UML de secuencia es posible identificar de manera gráfica los identificadores de los participantes, su dirección, el orden relativo en que ocurrieron. Así también, los mensajes y su contenido que intervienen en una colaboración comercial.

La cadena de suministro es un ambiente complejo debido a que frecuentemente se incluyen recursos de múltiples proveedores y plataformas, por lo que comprender el status de un recurso, es comprender sólo una pequeña parte del ambiente. El hecho es que la complejidad de administrar recursos crece con el número y tipo de recursos desplegados, por lo que las compañías requieren de herramientas que soporten y automaticen sus esfuerzos de administración. La tecnología de servicios Web ofrece el despliegue de servicios para su correcta administración.

Finalmente, uno de los aspectos más críticos en la cadena de suministro es mantener su continua operación tanto como sea posible. Para brindar mecanismos efectivos de comunicación a lo largo de la cadena, es necesario un servicio de mensajería confiable para garantizar la entrega de la información involucrada en las colaboraciones de la cadena de suministro.

1.6. Organización de la Tesis

El material presentado en esta tesis se organiza de la siguiente manera. En el capítulo 2 se definen los principales conceptos sobre Cómputo Orientado a Servicios y se explica cómo las arquitecturas orientadas a servicios ayudan a proveer aplicaciones basadas en servicios Web. Además se presenta el diseño de BPIMS-WS (*Business Processes Integration and Monitoring System for Web Services*), la arquitectura orientada a servicios híbrida del sistema de intermediación que se desarrolló en este trabajo, la cual provee una composición gradual de servicios ofreciendo la funcionalidad y los roles necesarios para la integración de múltiples servicios en un servicio compuesto. En el capítulo 3 se describe la arquitectura

orientada a servicios implementada en BPIMS-WS, capaz de integrar los procesos de negocio de diferentes empresas en el contexto del comercio electrónico B2B y en la administración de la cadena de suministro. Además, se presenta el diseño de BPIMS-WS y los componentes situados en cada capa. También se argumenta la funcionalidad de cada capa así como de sus componentes. En el capitulo 4 se describe detalladamente los mecanismos de descubrimiento e invocación de servicios Web que utiliza BPIMS-WS presentando los algoritmos y técnicas utilizadas. De igual forma, se revisan los trabajos relacionados y se discuten sus ventajas y desventajas en comparación con éstos. En el capítulo 5 se describe el mecanismo utilizado para la composición de procesos de negocio en BPIMS-WS, es decir, se describen las técnicas empleadas para la creación de nuevos procesos de negocio descritos como servicios Web en tiempo de diseño y ejecución. También, se revisan los trabajos relacionados y se discuten sus ventajas y desventajas en comparación con éstos. En el capítulo 6 se describe el proceso de supervisión de procesos de negocio descritos como servicios Web por lo que se describen las técnicas utilizadas en BPIMS-WS. De igual forma, se revisan los trabajos relacionados y se discuten sus ventajas y desventajas en comparación con éstos. El capítulo 7 presenta el mecanismo empleado por BPIMS-WS para poder llevar a cabo la administración de procesos de negocio descritos como servicios Web. Así también, se revisan los trabajos relacionados y se discuten sus ventajas y desventajas en comparación con éstos. En el capítulo 8 se describen tres casos de estudio donde BPIMS-WS se utilizó para la integración de procesos de negocio en la cadena de suministro. El primer caso de estudio describe la integración de los procesos de negocio de un modelo de empresa virtual a través de BPIMS-WS. El segundo caso de estudio describe la funcionalidad de BPIMS-WS como un agente de compra distribuido. Finalmente, el último caso de estudio describe la funcionalidad de BPIMS-WS para el reaprovisionamiento oportuno en la cadena de suministro. Finalmente, en el capítulo 9 se proveen las conclusiones, se discute sobre las contribuciones de BPIMS-WS y se consideran algunos aspectos como trabajo a futuro.

Capítulo 2
Cómputo Orientado a Servicios

El cómputo orientado a servicios es el paradigma computacional que utiliza los servicios como elementos fundamentales para el desarrollo de aplicaciones y soluciones de integración. Para construir un modelo basado en servicios, el cómputo orientado a servicios utiliza las arquitecturas orientadas a servicios como una forma de reorganizar aplicaciones de software previamente desarrolladas incorporándolas en un conjunto de servicios interconectados, cada uno accesible a través de interfaces y protocolos de mensajería estándares. Sin embargo, una arquitectura orientada a servicios básica no abarca completamente todas las capacidades necesarias en una arquitectura de integración como la administración y orquestación de servicios, la cual debe aplicarse a todos los componentes en una arquitectura basada en servicios. No obstante, estas capacidades se abordan en una arquitectura basada en servicios Grid que provee abstracción de alto nivel y facilidades de administración que permiten a los servicios funcionar como una unidad integrada para colaborar con otros servicios. En este capítulo se propone una arquitectura híbrida que presenta características tanto de una arquitectura orientada a servicios básica y de una arquitectura basada en servicios Grid, con el fin de proveer funcionalidades de orquestar y administrar servicios lo cual reduce el costo y complejidad en la integración de aplicaciones a través del uso de servicios Web.

2.1. Introducción

El cómputo orientado a servicios (SOC, *Service-Oriented Computing*) es el paradigma computacional que utiliza los servicios como elementos fundamentales para el desarrollo de

aplicaciones y soluciones de integración [27]. En el cómputo orientado a servicios, los servicios son auto-descriptivos e independientes de la plataforma proporcionando un soporte para la composición de aplicaciones distribuidas. Los servicios realizan funciones que pueden ser desde simples solicitudes hasta procesos de negocio complejos. Mediante los servicios, las organizaciones exponen sus procesos de negocio en Internet (o en una Intranet) utilizando lenguajes y protocolos estándares basados en XML, los cuales se implementan a través de una interfaz auto-descriptiva basada en estándares abiertos.

Debido a que los servicios proveen una distribución uniforme y ubicua de la información en un amplio rango de dispositivos de cómputo (como computadoras portátiles, PDAs y teléfonos celulares) y plataformas de software (como UNIX o Windows), ellos constituyen la próxima generación de cómputo distribuido.

En SOC, las organizaciones ofrecen los servicios las cuales realizan la implementación, proporcionan su descripción y ofrecen el soporte técnico y comercial relacionado a los servicios que ofrecen. Debido a que diferentes empresas pueden ofrecer servicios en Internet, ellos proveen una infraestructura de cómputo distribuido para la integración y colaboración de aplicaciones intra e inter-organizacionales. En este sentido, los clientes de servicios pueden ser otras soluciones, procesos, usuarios o aplicaciones dentro o fuera de una empresa. Para satisfacer estos requisitos de integración, los servicios deben ser:

- **Independientes de la tecnología:** deben ser consumibles a través de tecnologías consideradas como estándares, las cuales estén disponibles en casi todos los ambientes de cómputo distribuido. Esto implica que los mecanismos de invocación (protocolos y mecanismos de descripción y descubrimiento) deben obedecer estándares ampliamente aceptados.
- **Bajamente acoplados:** no deben requerir de conocimiento, estructura interna o contexto alguno en el lado del cliente o del servicio.
- **Transparentes en su ubicación:** deben tener almacenada sus definiciones e información de localización en un repositorio de servicios el cual sea accesible por una variedad de clientes que pueden localizar e invocar los servicios independientemente de su ubicación.

En SOC se identifican dos niveles de integración, el primero de ellos es integración a nivel de servicios básicos o *e-services* [28]. Esto significa que se hace énfasis en la integración de servicios con otras aplicaciones que se encuentran públicamente disponibles en Internet. El segundo se refiere a los servicios compuestos, que hace énfasis en la integración de un servicio compuesto formado por más de un servicio básico. Estos también son conocidos como *e-workflows* [29]. Por ejemplo, una empresa que ofrece una colección de servicios simples que realizan tareas comerciales específicas como facturación, manejo de pedidos y relaciones con los clientes, puede orquestar estos servicios creando una aplicación comercial distribuida que proporciona la personalización de pedidos, servicio a clientes y facturación para una línea especializada de productos (por ejemplo, equipos de telecomunicación, seguros médicos, entre otros). Entonces, los servicios ayudan en la integración de aplicaciones distribuidas y permiten definir arquitecturas y técnicas para construir nuevas funcionalidades a través de la integración de aplicaciones existentes.

Las aplicaciones basadas en servicios se desarrollan como conjuntos de servicios que ofrecen interfaces bien definidas a sus potenciales usuarios. Esto se logra debido al bajo acoplamiento de las aplicaciones distribuidas de los socios comerciales, lo cual no exige acuerdos predeterminados en el uso de un servicio ofrecido.

Dado que los servicios encapsulan la funcionalidad de procesos de negocio, se requieren algunos mecanismos para llevar a cabo la comunicación e interacción de servicios. Estos mecanismos pueden ser diversos ya que los servicios pueden implementarse en una sola computadora, o de manera distribuida a través de una red de computadoras de área local (LAN, *Local Area Network*) o una red de área amplia (WAN, *Wire Area Network*). Un caso particularmente interesante es cuando los servicios utilizan estándares de Internet y éste se utiliza como medio de comunicación. A este tipo de servicios se les conoce como servicios Web. Un servicio Web es un tipo específico de servicio que se identifica por un URI (*Uniform Resource Identifier*) y exhibe las siguientes características:

1. Expone sus características funcionales en Internet a través de lenguajes y protocolos estándares de Internet, y
2. Puede implementarse por medio de una interfaz auto-descriptiva basada en estándares abiertos de Internet (por ejemplo, interfaces XML)

En las interacciones de servicios Web se utiliza SOAP [30] para transmitir datos XML y WSDL [31] para expresar las descripciones del servicio. WSDL se utiliza para publicar un servicio Web en términos de sus puertos (direcciones electrónicas donde se encuentra la implementación), tipos de puertos (definiciones abstractas de las operaciones e intercambio de mensajes que provee) y *"bindings"* (definiciones concretas de los protocolos de comunicación que soporta). El estándar UDDI [32] es un directorio de servicios que contiene las publicaciones de los servicios y permite a los clientes de servicios Web localizar servicios candidatos y descubrir sus detalles de implementación.

2.2. El concepto de Software como servicio

El concepto de software como servicio, el cual está implícito en el cómputo orientado a servicios (SOC), apareció por primera vez en el modelo de software de proveedor de servicio de aplicaciones (ASP, *Application Service Provider*). Un ASP es una empresa que permite a sus clientes el uso de aplicaciones de software a través de conexiones seguras a Internet mediante un modelo de alquiler o cuota periódica. Esto implica que los servidores y aplicaciones de software no están instalados en el lado del cliente, sino en la empresa proveedora del servicio o en un directorio de datos (también conocido como *data center*) contratado por ella. La clave del ASP es que el software es un servicio, no es un producto. En un principio, este modelo se utilizó en pequeñas y medianas empresas, pero hoy en día las empresas consideran este modelo como una forma de subcontratación (*outsourcing*).

Aunque el modelo de software ASP introdujo el concepto de software como servicio, sufría de limitaciones inherentes como era la incapacidad de desarrollar aplicaciones interactivas e incapacidad para proveer aplicaciones personalizadas [33]. Esto originó como resultado el desarrollo de arquitecturas monolíticas y específicas de los clientes, las cuales proveían una solución de integración no reutilizable ya que se basaban en los principios de un fuerte acoplamiento.

El paradigma de SOC permite extender el concepto de software como servicio incluyendo procesos y transacciones comerciales complejas y permitiendo que las aplicaciones se

construyan de una manera rápida mediante la reutilización de servicios no importando quién y dónde se provee el servicio [34]. Debido a los grandes beneficios de la tecnología de servicios Web, muchos ASPs están modificando sus modelos comerciales para ser más similares a los proveedores de servicios Web

2.3. Paradigmas Orientados a Objetos (OO) vs Paradigmas Orientados a Servicios (SO)

La metodología OOAD (*Object-Oriented Analysis and Design*) propuesta por Booch y Jacobson [35], provee un excelente punto de inicio en la definición de una arquitectura orientada a servicios (SOA, *Service-Oriented Architecture*). La metodología OOAD es un enfoque muy poderoso y probado, y como tal, en el diseño de una SOA se deben utilizar técnicas de análisis OO (*Object-Oriented*) tanto como sea posible. Para aplicar con éxito el análisis OO a una SOA, se debe examinar a más de un sistema en un momento, por lo que los modelos de caso de uso juegan un papel importante. Sin embargo, en una SOA el concepto predominante es el proceso en vez de manejar casos de uso.

En el nivel de diseño, la meta de OOAD es habilitar el rápido y eficiente diseño, desarrollo y ejecución de aplicaciones de tal forma que sean flexibles y extensibles. Los objetos son estructuras de software que presentan un comportamiento similar a entidades del mundo que ellos modelan. Por ejemplo, un objeto "Cliente" puede tener un nombre, información del contacto, y podría tener uno o más objetos "CuentaBancaria" asociados con él. Desde la perspectiva de OOAD, todo esto es un objeto. Los principios fundamentales de OO son: 1) Encapsulación, 2) Ocultación de información, 3) Clases e instancias, 4) Asociación y herencia, 5) Mensajería y 6) Polimorfismo.

Las técnicas de análisis orientado a objetos tienen un soporte completo para las diferentes fases del ciclo de vida (análisis, diseño y desarrollo) de aplicaciones:

- OOAD intenta encontrar el conjunto óptimo de objetos y la jerarquía de clase más natural para implementarlo.
- En la programación orientada a objetos, el desarrollo de aplicaciones se realiza de una sola manera, implementando un escenario comercial o un caso de uso.

- En un ambiente de ejecución orientado a objetos se ejecuta código, se proporcionan los servicios de aplicación como recolección de basura y se proveen mecanismos para que objetos que residen en diferentes servidores de aplicaciones puedan intercomunicarse.

El principal problema con las técnicas de diseño orientado a objetos con respecto a SOC es que su nivel de granularidad se enfoca en el nivel de clase, el cual se sitúa demasiado abajo para un nivel de abstracción para la modelación de un servicio comercial. Las asociaciones como herencia crean un fuerte acoplamiento y por consiguiente, una dependencia entre las partes involucradas. Por el contrario, el paradigma SOC promueve flexibilidad y agilidad a través de un bajo acoplamiento. Sin embargo, el paradigma OO todavía es un valioso enfoque para el diseño de clases y componentes dentro de un servicio definido. Además, muchas técnicas de OOAD como clases, responsabilidades, y tarjetas de colaboraciones (CRC) pueden utilizarse para el diseño del servicio, siempre y cuando se incremente el nivel de abstracción. En la figura 2.1 se ilustran los niveles de visibilidad y enfoque dados por el diseño OO, orientado a componentes (*CO, Component-Oriented*) y SOC. También se ilustra cómo ellos se relacionan dentro del contexto de una SOA.

Otro de los elementos importantes en el diseño de una SOA es el uso de notaciones UML. El Proceso Unificado Racional® (*RUP, Rational Unified Process*) reconocido como uno de los principales procesos OOAD que tiene soporte para el análisis, diseño y desarrollo de software de forma incremental, utiliza la modelación en UML. Sin embargo, RUP no provee los mecanismos necesarios para el diseño de aplicaciones basadas en una SOA debido a que se fundamenta en las técnicas de diseño de OOAD por consiguiente, el problema de RUP reside en el nivel de abstracción de modelación para un servicio comercial. En otras palabras, RUP sólo permite la modelación a nivel de clases y componentes, pero no a nivel de servicios, siendo que un servicio puede constituirse de componentes y clases, tal como se muestra en la Fig. 2.1. Desde el punto de vista de RUP, la arquitectura de un sistema es la estructuración de sus principales componentes que interactúan a través de interfaces definidas. Además, estos componentes se forman de manera decreciente por componentes más pequeños que tienen granularidad a nivel de clase.

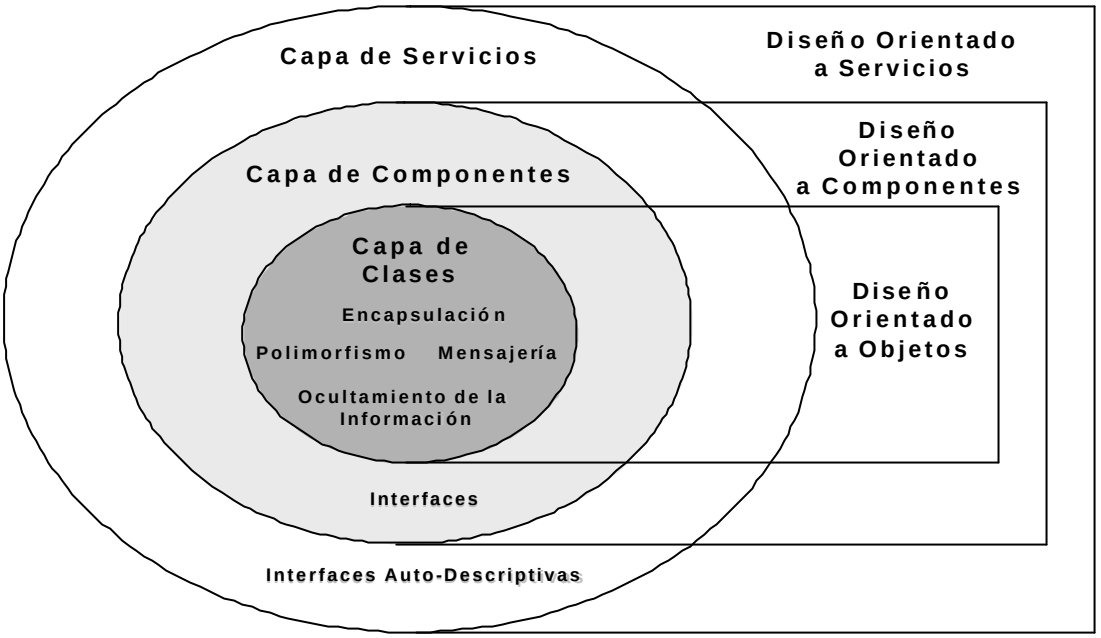


Fig. 2.1. Capas de Diseño de OOAD, CO y SOC

Por el contrario, la arquitectura de un sistema basado en una SOA considera servicios auto-descriptivos, totalmente encapsulados y sin estados que satisfacen un servicio comercial genérico. Estos servicios pueden formarse de un número de servicios orquestados o de colaboración. Esto no excluye el punto de vista de OO adoptado por RUP, sino que debe considerarse otra capa de abstracción sobre él. Por lo que, esta capa debe encapsular componentes que se diseñaron como artefactos de RUP (como servicios de software).

2.4. La arquitectura orientada a servicios básica

Comúnmente, una empresa se constituye de un conjunto de aplicaciones y diversas fuentes de información que realizan una serie de funciones y almacenan innumerables datos. Por lo general, estas aplicaciones y recursos de información residen en diferentes organizaciones las cuales se construyeron utilizando diferentes tecnologías de información y ejecutándose en diferentes plataformas de software. En este contexto, una SOA provee un mecanismo de integración de aplicaciones existentes sin tener en cuenta la plataforma o el lenguaje de

programación en que se desarrollaron, por lo que las soluciones basadas en una SOA se componen de servicios re-utilizables, bien definidos, públicos e interfaces basadas en estándares de Internet. Hay tres niveles de abstracción conceptual dentro de una SOA:

1. **Operaciones:** Son transacciones que representan unidades lógicas de trabajo. La ejecución de una operación influye en la persistencia de la información ya que puede representar funciones de lectura, escritura y modificación. Las operaciones de una SOA se comparan directamente a los métodos de la programación orientada a objetos (OOP, *Object-Oriented Programming*). De igual forma que los métodos, las operaciones tienen una interfaz específica, estructurada y devuelven respuestas estructuradas. Así también, la ejecución de una operación específica puede involucrar la invocación de operaciones adicionales
2. **Servicios:** Representan agrupaciones lógicas de operaciones. Por ejemplo, suponga que hay un servicio de Registro de Clientes, entonces, capturar, verificar, validar y guardar la información representan las operaciones asociadas que constituyen al servicio
3. **Procesos de negocio:** Es un conjunto de acciones o actividades realizadas con metas comerciales específicas. En un escenario comercial típico, los procesos de negocio realizan múltiples invocaciones de servicios. Algunos ejemplos de procesos de negocio son: Venta de Productos o Servicios, Órdenes de Pago, entre otros. En términos de una SOA, un proceso comercial consiste de una serie de operaciones que se ejecutan en una cierta secuencia según un conjunto de reglas comerciales. El orden, la selección y la ejecución de las operaciones se denomina orquestación del proceso comercial. Comúnmente, los servicios orquestados se invocan para responder a eventos comerciales.

Actualmente, existen diversas definiciones de una arquitectura orientada a servicios desde diferentes enfoques. Para el cómputo basado en Internet, una SOA es un nuevo paradigma computacional que permite crear, diseñar, administrar y acceder a recursos, funciones, aplicaciones y procesos de negocio vistos como servicios, por medio de interfaces auto-descriptivas sin importar su ubicación y su tecnología de implementación [36].

Desde el enfoque de comunicación, una SOA es una manera de reorganizar aplicaciones de software previamente desarrolladas incorporándolas en un conjunto de servicios

interconectados, cada uno accesible a través de interfaces y protocolos de mensajería estándares [37]. En una SOA, aplicaciones existentes y futuras pueden acceder a estos servicios sin necesidad de una integración punto a punto la cual se basa en protocolos propietarios. Este enfoque se aplica cuando múltiples aplicaciones que se ejecutan en diferentes plataformas necesitan comunicarse entre sí. De esta manera, las empresas pueden combinar servicios para realizar transacciones comerciales con un mínimo esfuerzo de programación.

Para la ingeniería de software, una SOA es un diseño lógico de un sistema de software que proporciona servicios a aplicaciones de usuarios finales y otros servicios que se encuentran distribuidos en una red a través interfaces de publicación y descubrimiento [38].

En cómputo distribuido una SOA básica define una interacción entre agentes de software y aplicaciones como un intercambio de mensajes entre solicitantes de servicio (clientes) y proveedores de servicio [39]. Los clientes son agentes de software o aplicaciones que solicitan la ejecución de un servicio. Los proveedores son agentes de software o aplicaciones que proporcionan el servicio. En este sentido, bs agentes de software o aplicaciones pueden ser tanto clientes como proveedores de servicios. Los proveedores son responsables de publicar la(s) descripción(es) del servicio que ellos proporcionan. Los clientes encuentran esa(s) descripción(es) y se integran con ellas analizando e invocando las descripciones de los servicios. Por lo que una SOA básica no es solo una arquitectura de servicios, es una relación entre tres tipos de participantes: el proveedor de servicio, el repositorio de servicios y el solicitante de servicio (cliente). Las interacciones entre ellos involucran mecanismos de publicación, consulta e integración [40], tal como se muestra en la Fig. 2.2. En un escenario típico de una SOA, un proveedor de servicio define una descripción (implementación) para un servicio y lo publica en un repositorio de servicio a través del cual los clientes pueden descubrir el servicio. El solicitante de servicio utiliza una operación de consulta para recuperar la descripción de servicio del repositorio de servicios, y utiliza la descripción del servicio para integrarse con el proveedor del servicio invocando o interactuando recíprocamente con la implementación del servicio. Los roles tanto de proveedor como de solicitante de servicio son estructuras lógicas y un servicio puede exhibir características de ambos.

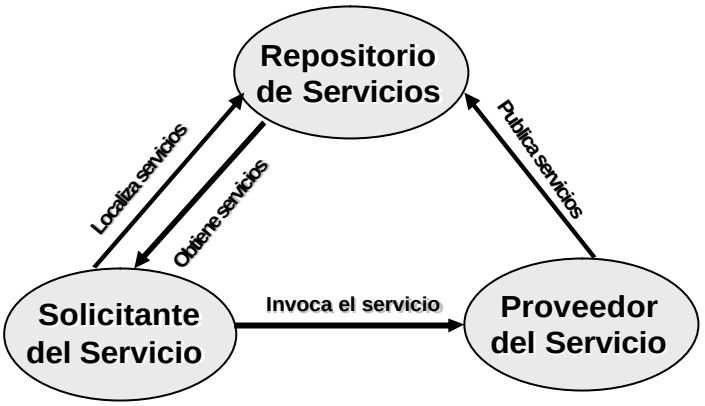


Fig. 2.2. La arquitectura orientada a servicios básica

Usualmente, un servicio es una función comercial implementada y encapsulada en una interfaz que es bien conocida para que pueda encontrarse no sólo por agentes que diseñaron el servicio sino también por agentes que no conocen cómo el servicio se diseñó y que desean accederlo y utilizarlo. Este enfoque de encapsulación de "caja negra" es una característica de los principios de modularidad en Ingeniería de Software, por ejemplo, módulos, objetos y componentes. Sin embargo, los servicios son diferentes de todas estas formas de modularidad ya que representan funciones comerciales completas que pueden ser reutilizadas formando nuevas transacciones comerciales de tipo simple o compuesto. Además, los servicios representan una funcionalidad comercial que puede utilizarse para construir nuevas y más grandes configuraciones que dependan de la necesidad de los clientes o usuarios.

En SOC, la interfaz de un servicio proporciona los mecanismos para que los servicios se comuniquen con otras aplicaciones y servicios. Técnicamente, la interfaz de servicio es la descripción de un conjunto de operaciones que están disponibles a un cliente de servicio para su invocación. La especificación de servicio debe describir explícitamente todas las interfaces que un cliente de éste servicio espera, así como las interfaces del ambiente en que el servicio reside. Dado que las interfaces de servicio de servicios compuestos se proporcionan por otros servicios, la especificación del servicio funciona como un medio para definir cómo una interfaz de servicio compuesta puede relacionarse con otras interfaces de servicios importados y cómo puede implementarse fuera de las interfaces de los servicio importados tal como se muestra en la Fig. 2.3. En este sentido, la especificación

de un servicio tiene una misión idéntica a un meta-modelo de composición que proporciona una descripción de cómo las interfaces de servicios Web actúan recíprocamente entre sí y cómo definen una nueva interfaz de servicio Web (elementos *PortType*) como una colección (ensamblaje) de otras existentes (elementos *PortType* importados), como se muestra en la Fig. 2.3.

Entonces, una especificación de un servicio define los límites de encapsulamiento de un servicio y, por consiguiente, determina la granularidad de una composición de interfaces de servicios Web. Esto es una forma confiable de diseñar servicios utilizando la importación de servicios sin conocimiento de sus implementaciones.

Como el desarrollo de servicios compuestos requiere la utilización de múltiples interfaces de servicio importados es útil introducir el concepto de una interfaz de uso de servicio en esta fase.

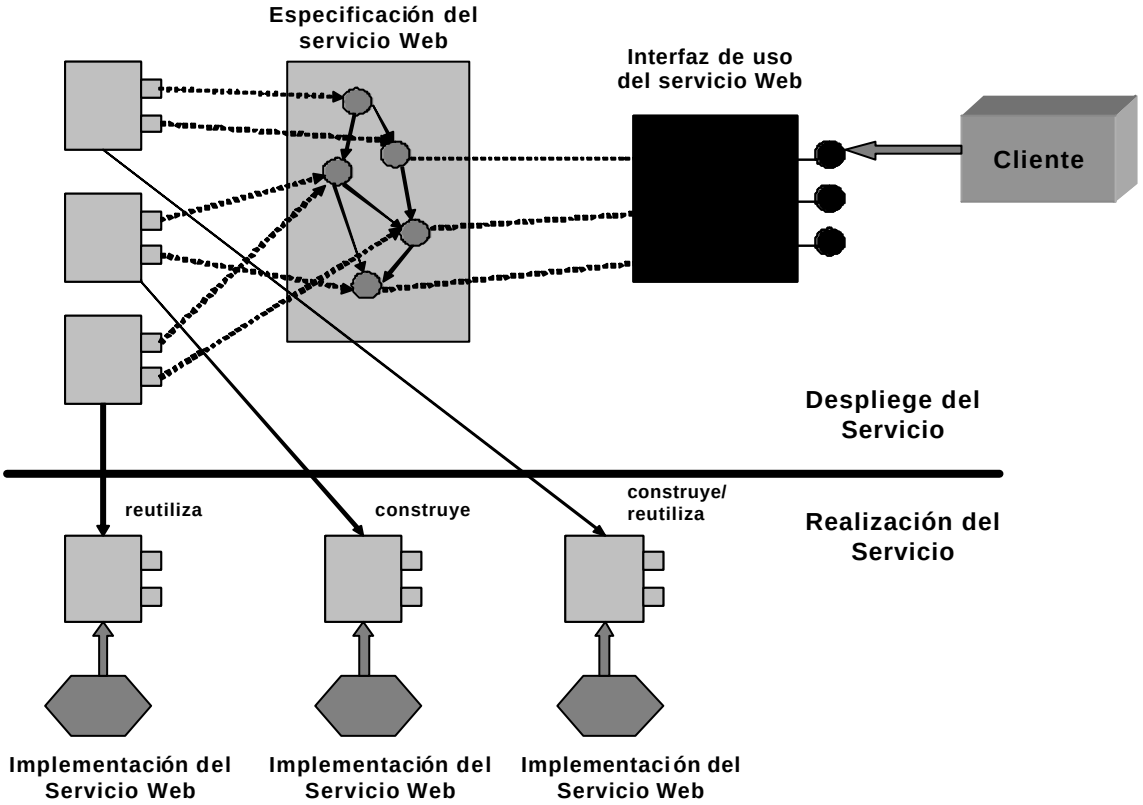


Fig. 2.3. Interfaces e Implementación de un Servicio

Una interfaz de uso de servicio es una interfaz que el servicio expone a sus clientes [41]. Esto significa que la interfaz de uso de un servicio es similar a la interfaz de un servicio importado como se muestra en la Fig. 2.3. Las descripciones de servicio se usan para publicar las capacidades, interfaces, conducta y calidad del servicio. La publicación esta información en un repositorio de servicio, provee los mecanismos necesarios para el descubrimiento, selección, integración y composición de servicios. En particular, la descripción de la interfaz de servicio publica cómo acceder y utilizar el servicio mientras la descripción de la capacidad de servicio define el propósito conceptual y los resultados esperados del servicio.

La conducta de un servicio durante su ejecución se describe en la descripción de conducta del servicio, como un proceso de flujo de trabajo (*workflow process*). Finalmente, la descripción de la calidad del servicio (*QoS*) describe atributos de calidad funcional y no funcionales como el costo, métricas de rendimiento (tiempo de respuesta), seguridad, integridad (transaccional), confiabilidad, escalabilidad y disponibilidad. Incluso, la implementación de un servicio puede involucrar la integración de aplicaciones, por ejemplo, programas monolíticos o programas que pertenecen a múltiples aplicaciones. Por lo que un servicio en una SOA, se diseña de tal forma que pueda invocarse por diversos clientes de servicio y sea lógicamente desacoplado (bajo acoplamiento). Es por esto que en una SOA, el servicio no se acopla con sus potenciales consumidores de servicio (clientes), de hecho no sabe nada de ellos. Por consiguiente, quienes deben acoplarse a un servicio son los consumidores del servicio por lo que se depende de la disponibilidad y versión del servicio.

A pesar de que una SOA básica provee algunos mecanismos de integración, no abarca todas las capacidades necesarias en una arquitectura de integración como por ejemplo la administración de servicios, la cual debe aplicarse a cada uno de los componentes en una arquitectura basada en servicios. Las funciones de administración de servicios en una SOA también pueden residir en Computación Grid (GC, *Grid Computing*). Uno de los objetivos de la Computación Grid es proveer los mecanismos necesarios para administrar en forma creciente el ciclo de vida de redes de computadoras. La arquitectura de servicios Grid es un modelo de abstracción de alto nivel que describe conductas, atributos, operaciones e interfaces comunes para permitir a una colección de servicios funcionar como una unidad

integrada para colaborar con otros servicios en un ambiente totalmente distribuido y heterogéneo [42]. Los servicios Grid constituyen un componente importante en la administración de servicios distribuidos teniendo un alcance más allá de los límites de una empresa abarcando un amplio rango de socios comerciales, como en los mercados abiertos [43, 44]. Para este propósito, los servicios Grid proveen la funcionalidad de la capa de administración de servicios de la SOA híbrida del sistema de intermediación que se desarrolló en este trabajo.

Además, una SOA básica no es capaz de responder a eventos de tiempo real. Un evento puede ser la solicitud de una orden o un cambio en el precio de un producto o servicio. Estos eventos pueden ser fijos (por ejemplo, un mensaje de confirmación cada vez que se realice un pedido) o no programados (por ejemplo, el cambio de la cantidad de productos a solicitarse en un pedido). Bajo este contexto, las compañías requieren la habilidad de enviar, recibir y responder a información comercial imprevisible y a eventos de manera asíncrona, por lo que los servicios Grid proveen un modelo de interacción asíncrono que permite a los usuarios y aplicaciones iniciar e inspeccionar múltiples tareas y datos de forma simultánea, e inmediatamente alertar cuando ocurra una transacción completa o un evento crítico. A continuación se describen las características funcionales que provee una arquitectura basada en servicios Grid.

2.5. El Bus de Servicios Grid

Los objetivos de los servicios Web y servicios Grid se complementan ya que los servicios Web se enfocan en la descripción, descubrimiento e invocación de servicios independientes de la plataforma, mientras que los servicios Grid se enfocan en el descubrimiento dinámico y el uso eficaz de recursos de cómputo distribuidos. Este punto de intersección de servicios Web y servicios Grid ha dado lugar a la Arquitectura Abierta de Servicios Grid (OGSA, *Open Grid Services Architecture*) propuesta en [45, 46] la cual provee los mecanismos necesarios para que la funcionalidad de los servicios Grid pueda encapsularse a través de interfaces de servicios Web. Los servicios Grid proporcionan un conjunto de interfaces bien definidas que siguen convenciones específicas para facilitar la coordinación

y administración de servicios Web [47]. Un servicio Grid indica cómo un cliente puede interactuar con un servicio Web a través de su especificación definida en un documento WSDL. Los servicios Grid se sitúan en la capa de administración de servicios de la SOA híbrida que se propone en este capítulo y que se describe en la siguiente sección. En esta capa de administración se provee una infraestructura a los sistemas y aplicaciones que requieren la integración y administración de servicios en el contexto de mercados virtuales dinámicos. En esta capa, los servicios Grid proveen la posibilidad de lograr calidad en los servicios ya que se enfocan en la administración de sistemas y aplicaciones. Con este fin, los servicios de Grid proveen la infraestructura necesaria para que los servicios interactúen y se coordinen a través de arquitecturas Grid distintas. A esta infraestructura se le conoce como Bus de Servicios Grid. El Bus de Servicios Grid (SGB, *Service Grid Bus*) provee una arquitectura abstracta de alto nivel y facilidades de administración que permiten a los servicios (dentro de un mercado abierto) funcionar como una unidad integrada para colaborar con otros servicios. La arquitectura SGB provee los mecanismos necesarios para el registro, descubrimiento, selección/enrutamiento, aplicación de reglas comerciales, filtrado, asignación de ruta y mapeos topológicos de instancias de servicios [48].

El bus de servicios es una estructura lógica que no toma en cuenta dónde o en qué plataforma un proveedor de servicio se ejecuta. Una interfaz de usuario, diseñada para coincidir de manera exacta los requisitos comerciales puede consumir servicios proporcionados por el bus de servicio, delegando a la empresa la tarea de seleccionar eficazmente el proveedor de servicio.

Un SGB se diseña para proveer conectividad y administración de servicios logrando los siguientes aspectos:

1. **Alcance y Robustez:** El SGB permite localizar servicios en cualquier parte de un mercado abierto y aislarlos de errores de conexión y barreras como corta fuegos, *proxies* y caches. Esto garantiza que cada servicio tiene una conexión libre de errores a cualquier otro servicio. La aplicación debe ser en conjunto robusta bajo cualquier tipo de fallos a largo plazo de uno o más componentes de servicio.
2. **Políticas y Administración de seguridad:** Los servicios deben describir sus capacidades y requisitos a su ambiente y sus usuarios potenciales. Una colección de capacidades y requisitos es una política [49]. En este contexto, una política puede

expresar diversas características como las capacidades transaccionales del servicio, la seguridad, el tiempo de respuesta, entre otras. Por ejemplo, una política de un servicio puede especificar que todas las interacciones deben invocarse bajo cierto protocolo de comunicación, que los mensajes entrantes tienen que cifrarse y que se firmarán los mensajes salientes, que sólo pueden aceptarse respuestas dentro de un intervalo de tiempo específico, por mencionar solo algunas.

3. **Costo y Tiempo de desarrollo:** El SGB proporciona los mecanismos que permiten agregar fácilmente servicios a un servicio compuesto. Nuevos clientes y servicios pueden agregarse en cualquier parte del ciclo de vida de la aplicación.
4. **Escalabilidad y Rendimiento:** El SGB debe tener un buen rendimiento a pesar de la existencia de componentes lentos e impredecibles y de largas latencias de la red.

El SGB permite a los componentes de servicio interactuar sobre cualquier conexión de la red, administrar los errores, barreras y condiciones de desconexión de la red. Un SGB provee un soporte para un modelo de transacción comercial y los mecanismos de soporte para conductas transaccionales avanzadas de procesos de negocio complejos orientados a servicios [50]. El modelo de transacción comercial permite expresar el criterio de atomicidad no convencional, por ejemplo, atomicidad del pago, atomicidad de la conversación, atomicidad del contrato, y además, permite expresar acuerdos de colaboración y secuencias de conversación de negocios. El modelo es un enfoque escalonado para transacciones comerciales ya que todo el intercambio de información entre los socios de negocio se realiza bajo las condiciones que soportan.

Esto ofrece flexibilidad, reducción de latencias, compensaciones de transacción y "*rollback*" en las interacciones comerciales. Las comunicaciones asíncronas del SGB lo hacen más robusto bajo cualquier fallo en los nodos de servicios, y permite un re-direccionamiento transparente en caso de un fallo prolongado. La ventaja principal del SGB es que permite a las compañías identificar y responder a eventos que necesitan direccionarse por uno o más sistemas a través de un manejador de eventos. Los eventos, coleccionados en un SGB, pueden analizarse para identificar patrones de negocio relevantes, para que las compañías puedan tener un comportamiento pro-activo y respondan a escenarios del mundo real en tiempo real.

Además, el SGB provee servicios estándares y de alto nivel para seguridad, administración, interacción de servicio, desarrollo y despliegue rápido de aplicaciones. El SGB también es consistente con flujos de trabajo emergentes de servicios Web y transacciones estándares como WS-T (*Web Services Transactions*) [51], WS-C (*Web Services Coordination*) [52] y BTP (*Business Transactions Protocol*) [53]. Finalmente, el SGB tiene soporte para una escalabilidad lineal y alto rendimiento ofreciendo soporte nativo para interacciones asíncronas y permitiendo administrar los servicios con balance de carga y distribución de carga de trabajo. En la Fig. 2.4 se describe un servicio bajo el modelo SGB para un mercado abierto.

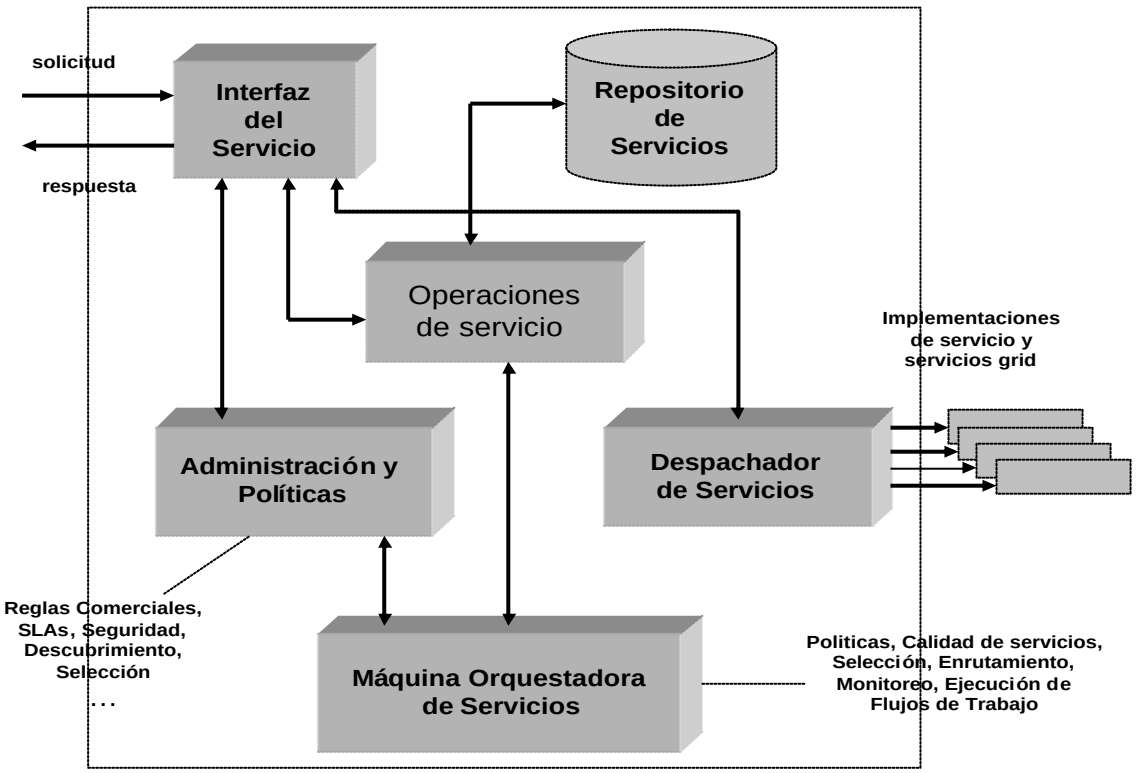


Fig. 2.4. El Bus de Servicios Grid

El modelo SGB permite compartir servicios y recursos dentro de mercados abiertos de servicios. Bajo este enfoque, el sistema de intermediación actúa como un mercado de servicios comerciales que automáticamente atiende el mejor servicio disponible de un conjunto de proveedores de servicio ensamblados dinámicamente para satisfacer las

necesidades del usuario. La selección de un servicio no solo se basa en la disponibilidad, sino también en características de *QoS* y acuerdos comerciales específicos. En el sistema de intermediación, la selección del servicio se basa en la disponibilidad, por lo que se realiza automáticamente en base a las características proporcionadas por el sistema de intermediación, permitiendo a los proveedores de servicio ocultar la complejidad de la aplicación para manejar múltiples solicitudes de clientes bajo ambientes heterogéneos. Para esto, el sistema de intermediación contiene un despachador de servicios, al igual que un SGB, el cual atiende las solicitudes externas de servicios. Este módulo también realiza la selección para encontrar una instancia de servicio para ejecutar una solicitud de servicio.
El despachador de servicios puede descomponerse en unidades funcionales modulares para su personalización, con el fin de satisfacer necesidades especiales de diversas plataformas y de la lógica de las operaciones.

2.6. Limitaciones de los trabajos propuestos

Recientemente se han desarrollado algunos trabajos que proponen SOAs con el fin de integrar, orquestar, supervisar y agregar calidad a los procesos de negocio en el contexto de Computación Grid y servicios Web. No obstante, estas arquitecturas presentan algunas limitaciones. En el contexto de servicios Web, Chung et. al. [54] propone un sistema llamado eXFlow para la integración de procesos de negocio en el comercio electrónico B2B y EAI. eXFlow solo tiene soporte para la invocación, descubrimiento, orquestación y supervisión de servicios mientras que la administración de servicios se contempla como trabajo a futuro. Además, eXFlow esta todavía en la etapa de desarrollo. Además, debido a que eXFlow se basa en una SOA, no tiene soporte de algún tipo de mensajería asíncrona.
Lakhal et. al [55]. propone un sistema llamado THROWS para la ejecución distribuida de servicios Web compuestos. La arquitectura de THROWS solo tiene soporte para el descubrimiento, invocación, y orquestación de servicios, pero no se presenta alguna implementación.
Arpinar et. al. [56] propone una arquitectura para la orquestación semi-automática de servicios Web tanto para un enfoque centralizado como para uno punto-a-punto. La

arquitectura solo tiene soporte para el descubrimiento, invocación, orquestación y supervisión de servicios. La administración de servicios no se considera y tampoco se tiene alguna implementación hecha.

Turner et. al. [57] propone un sistema llamado IBHIS (*Integration Broker for Heterogeneous Information Sources*) para la integración de datos procedentes de fuentes de información heterogéneas. IBHIS es un sistema ya implementado pero no considera la supervisión de la información, ya que solo tiene soporte para la integración, orquestación y administración de la información.

Radetzki et. al. [58] propone un sistema llamado IRIS (*Interoperability and Reusability of Internet Services*) para la creación de servicios Web compuestos mediante un conjunto de interfaces gráficas. El sistema IRIS tiene soporte para el descubrimiento y orquestación de servicios basándose en un modelo de repositorio de ontologías. Sin embargo, IRIS se orienta a la simulación de servicios Web compuestos por lo que no se considera la ejecución. La supervisión y administración de servicios no se contempla.

Howard et. al. [59] propone un marco de trabajo llamado KDSWS (*Knowledge-based Dynamic Semantic Web Services*) que tiene soporte para la especificación semántica, descubrimiento, invocación, orquestación y administración de servicios bajo un modelo basado en agentes. La supervisión de servicios no se considera y la implementación está en desarrollo.

Srinivasan et. al. [60] propone una arquitectura para el descubrimiento de servicios Web bajo un enfoque orientado a metas. El descubrimiento se realiza por medio del encadenamiento de servicios a través de la satisfacción de restricciones. No se tiene soporte para la supervisión y administración de servicios y además, se encuentra en la etapa de diseño.

Yu et. al [61] propone un arquitectura para el descubrimiento dinámico de servicios en nodos UDDI mediante mecanismos de publicación/suscripción y notificación, pero solo se tiene un prototipo experimental básico ya que no provee la orquestación, supervisión y administración de servicios.

Aggarwal et. al. [62] propone un sistema llamado METEOR-S (*Managing End-To-End OpeRations for Semantic Web services*) para la orquestación de servicios Web basado en

restricciones. La arquitectura de METEOR-S no contempla la administración de servicios. No obstante, se tiene una implementación funcionando correctamente.

En el contexto de computación Grid, Zang et. al. [63] propone una infraestructura de servicios Grid abierta (OGSI, *Open Grid Services Infrastructure*). La infraestructura propuesta tiene soporte para la administración de flujos de trabajo y ejecución de servicio. Sin embargo, no se consideran la supervisión y administración de servicios.

Yang et. al. [64] propone un sistema llamado S-WFMS (*Service-Based Workflow Management System*) para la administración y orquestación de servicios en ambientes Grid. S-WFMS no considera la supervisión de servicios, pero aborda otros aspectos importantes como confiabilidad, tolerancia a fallos y balance de carga de trabajo.

Younas et. al. [65] propone un sistema multi-agentes para el descubrimiento y orquestación dinámica de servicios Grid. Esta arquitectura no provee soporte para la supervisión y administración, sin embargo, presenta características de seguridad y confiabilidad.

Finalmente, Benkner et. al. [66] propone un sistema llamado VGE (*Vienna Grid Environment*) para el descubrimiento y orquestación de servicios bajo demanda (*On Demand*). Actualmente, VGE es un prototipo experimental que incluye aspectos de calidad de servicio. La administración y supervisión están siendo consideradas.

En resumen, la Fig. 2.5 muestra un análisis comparativo de las diversas características que soportan las arquitecturas propuestas mencionadas anteriormente.

	Invocación y Descubrimiento	**Orquestación**	**Supervisión**	**Administración**	**Contexto Computacional**
Chung et. al.	**X**	**X**	**X**		SW
Lakhal et. al	**X**	**X**			SW
Arpinar et. al	**X**	**X**	**X**		SW
Turner et. al.	**X**	**X**		**X**	SW
Radetzki et. al.	**X**	**X**			SW
Howard et. al.	**X**	**X**		**X**	SW
Srinivasan et. al.	**X**	**X**			SW
Yu et. al	**X**				SW
Aggarwal et. al.	**X**	**X**	**X**		SW

	Invocación y Descubrimiento	Orquestación	Supervisión	Administración	Contexto Computacional
Zang et. al.	**X**	**X**			CG
Yang et. al.	**X**	**X**		**X**	CG
Younas et. al.	**X**	**X**			CG
Benkner et. al	**X**	**X**	**X***	**X***	CG

Fig. 2.5 Análisis Comparativo de las arquitecturas propuestas con la arquitectura SOA híbrida

SW= Servicios Web

CG= Computación Grid

X*= Lo considera, pero no lo implementa.

Como se puede observar, ninguno de estos trabajos aborda completamente todas las características necesarias de una arquitectura de integración para la cadena de suministro. Incluso el trabajo más reciente de Benkner et. al. [66] considera la supervisión y administración de procesos de negocio pero no los implementa. Es por esto, que en este trabajo se propone una arquitectura de integración de procesos de negocio para la cadena de suministro la cual aborde características de integración, orquestación, supervisión y administración de servicios Web. Esta arquitectura de integración se describe a continuación en la siguiente sección.

2.7. La arquitectura orientada a servicios híbrida

Debido a la falta de administración de servicios de una SOA básica y a la robustez de las comunicaciones asíncronas del SGB, en este trabajo se propone una arquitectura orientada a servicios híbrida que presenta el sistema de intermediación que se desarrolló, el cual provee las funcionalidades de orquestar, coordinar y administrar servicios lo que reduce el costo y complejidad en la integración de aplicaciones, como se muestra en la Fig. 2.5. La arquitectura híbrida del sistema de intermediación la cual tiene un diseño en capas, utiliza

una SOA básica en su capa inferior. Así también, la arquitectura híbrida presenta una capa de composición de servicios que abarca la funcionalidad y papeles necesarios para la orquestación de múltiples servicios en un solo servicio compuesto. Los servicios compuestos resultantes pueden utilizarse por otros sistemas de intermediación o integradores de servicio como componentes (es decir, servicios básicos) en una composición de servicio más compleja o pueden utilizarse como aplicaciones o soluciones de integración por los clientes. Así, el sistema de intermediación actúa como un integrador de servicios el cual a su vez es un proveedor del servicio que publica las descripciones de los servicios compuestos que ellos crean. En este sentido, el sistema de intermediación consolida los servicios que proporcionan otros proveedores de servicio con un valor agregado de servicio distinto. Además, el sistema de intermediación desarrolla la especificación y/o codifica el servicio compuesto para llevar a cabo las siguientes funciones:

- **Coordinación:** controla la ejecución de los componentes de servicio y administra el flujo de datos así como el rendimiento del servicio. Por ejemplo, especifica un flujo de trabajo y utiliza un motor de ejecución para controlar el servicio en tiempo de ejecución, al igual que el bus de servicios Grid de la Fig. 2.4.
- **Supervisión:** permite inspeccionar eventos o información que los componentes de servicio producen.
- **Integridad:** asegura la integridad de los servicios compuestos haciendo coincidir sus tipos de parámetros con aquéllos de sus componentes e impone restricciones en los componentes de servicio. Por ejemplo, realiza actividades de fusión de datos.

Para esto, la SOA híbrida del sistema de intermediación provee estas capacidades en la capa de administración de servicios tal como se muestra en la Fig. 2.6. La funcionalidad de las operaciones de administración en una SOA híbrida da soporte para administrar las plataformas de servicio y el despliegue de los servicios y aplicaciones en esas plataformas. Este conjunto de operaciones provee estadísticas de rendimiento que permiten medir la efectividad y el funcionamiento global de las aplicaciones y servicios simples o compuestos. De esta forma, se pueden evitar errores típicos que ocurren cuando acuerdos individuales a nivel de servicio (SLA, *Service-Level Agreement*) no coinciden.

Otro objetivo de la capa de administración de servicio de la SOA híbrida es proveer el soporte a mercados abiertos de servicio. Actualmente, existen dos tipos de mercados: verticales y horizontales. Los mercados verticales corresponden a un dominio en particular, mientras que los mercados horizontales pueden abarcar diversos dominios. Los mercados abiertos de servicio operan casi de la misma manera que los mercados verticales, sin embargo, su propósito es crear nuevas oportunidades para compradores y vendedores encontrando y conduciendo electrónicamente sus negocios, y suministrando servicios de valor agregado, agrupando un fuerte poder de compra (similar a una cooperativa).

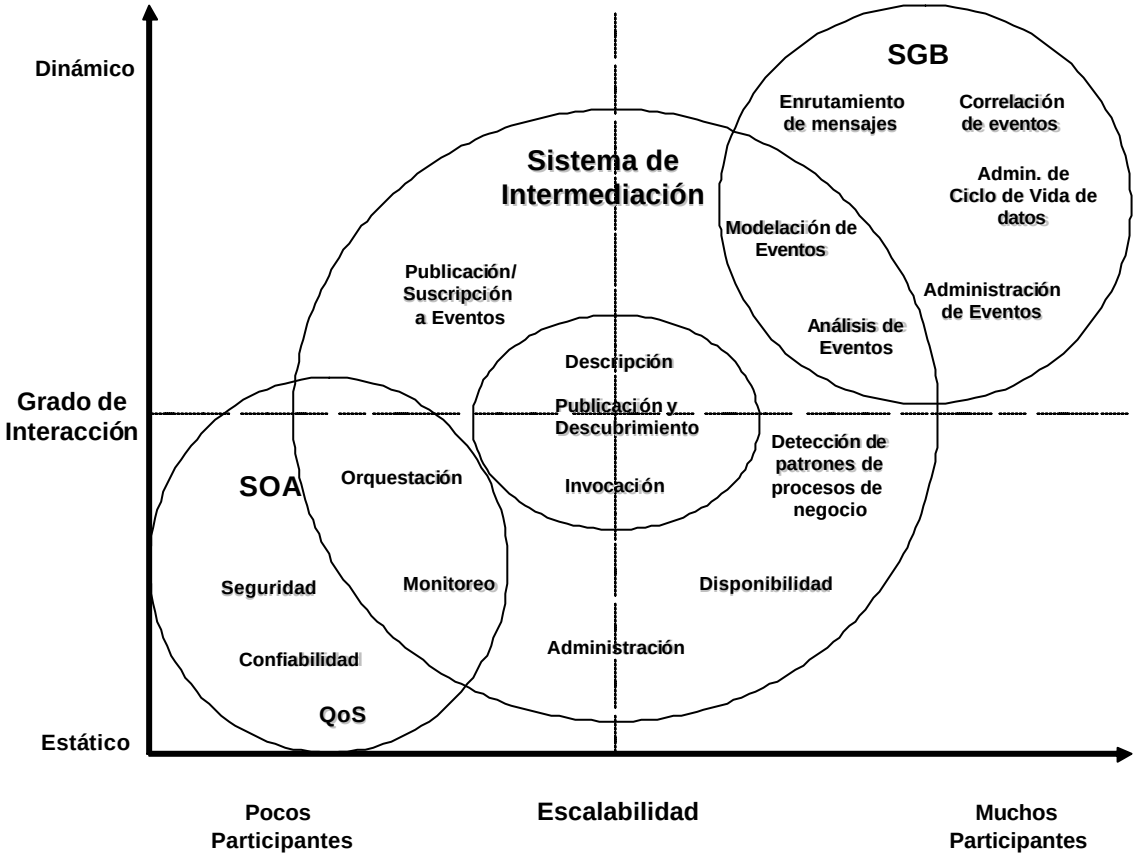


Fig. 2.6. Esquema General de la arquitectura orientada a servicios híbrida del sistema de intermediación

Sin embargo, el alcance de este tipo de mercados está limitado, ya que las empresas deben hacer visibles sus ofertas a otras empresas y establecer protocolos específicos para conducir

sus procesos de negocio. Los mercados abiertos tienen soporte para la cadena de suministro ya que proporciona a sus miembros una vista unificada de productos y servicios, terminología comercial estándar y descripciones detalladas de los procesos de negocio [67]. Además, los mercados de servicios deben ofrecer una amplia gama de servicios que soporten negociación y facilitación de transacciones comerciales como son pagos financieros, certificación y aseguramiento de calidad del servicio, popularidad (*rating*) de servicios y métricas de servicio tales como el número actual de solicitantes de un servicio.

La capa de administración de servicio de la SOA híbrida propuesta incluye funcionalidades de administración de mercado (como se ilustra en la Fig. 2.6) abarcando las funciones de administración descritas anteriormente. Los mercados abiertos se crean y mantienen a través de un consorcio de organizaciones que reúne tanto a proveedores como a vendedores. Este consorcio de organizaciones asume la responsabilidad de administrar el mercado y realiza tareas de mantenimiento para asegurar que la administración esté abierta para cualquier negocio y, en general, provee facilidades para el diseño y desarrollo de servicios integrados que satisfacen necesidades comerciales específicas conforme a estándares industriales.

Uno de los mayores beneficios de introducir una integración gradual, como sucede en la SOA híbrida propuesta, es la habilidad de acoplar servicios de valor agregado que proveen soluciones para ciertas necesidades de desarrollo comunes. Bajo este contexto, el sistema de intermediación que se desarrolló en este trabajo, provee una SOA híbrida, la cual combinada con un SGB, crea una arquitectura empresarial de tiempo real que habilita transacciones comerciales en tiempo real, tal como se muestra en la Fig. 2.6.

La justificación de diseño del sistema de intermediación de combinar características de SOA y SGB reside en que una SOA básica, no es lo suficientemente robusta o confiable debido a que no provee mecanismos para garantizar:

1) **Consistencia de la información a través de diversas organizaciones**
2) **Alta disponibilidad de servicios**
3) **Seguridad en la información y en servicios no-públicos**
4) **Orquestación de múltiples servicios para crear aplicaciones compuestas**
5) **Administración de meta-datos**

Por lo que en una SOA básica, el requisito primordial es definir cómo el sistema realiza su flujo de trabajo entre los servicios. En lo que respecta a SGB, éste provee el soporte de eventos y servicios de mensajería que habilitan la administración, análisis, correlación y planeación de eventos. Además, soporta la administración del ciclo de vida de los datos, procesos de negocio y detección de patrones de procesos de negocio. En este contexto, SGB provee dos características importantes dentro de la cadena de suministro:

- **Servicio de mensajería asíncrona:** Asegura que las aplicaciones provean la habilidad de enviar mensajes asíncronos cuando ocurran eventos. Debido a que la mensajería asíncrona no requiere una contestación inmediata del destinatario y se asegura la entrega del mensaje, esto garantiza la generación y consumo de eventos por entidades apropiadas aún cuando un sistema esté temporalmente no disponible.
- **Administración de Eventos:** Provee la habilidad para identificar y añadir eventos para que sean administrados en la misma forma que la información y los procesos de negocio de una empresa se administran. Esto implica la habilidad de capturar los eventos cuando ocurran y expiren, y determinar patrones bien definidos de procesos de negocio.

2.8. Resumen

El mejoramiento de los procesos involucrados en el reaprovisionamiento oportuno es uno de los aspectos críticos en la gestión de la cadena de suministro. Para esto, se necesita conocer en tiempo real el estado de los inventarios de los proveedores de productos y servicios para poder anticipar los pedidos. Una SOA básica no es capaz de responder a eventos de tiempo real, por lo que el reaprovisionamiento oportuno no puede considerarse bajo esta arquitectura. Para esto, se requiere añadir capacidades adicionales como un servicio de mensajería asíncrono y el soporte a eventos, los cuales son característicos de un SGB. En este capitulo se presenta una arquitectura híbrida para un sistema de intermediación que presenta características tanto de una arquitectura orientada a servicios básica y de una arquitectura basada en servicios Grid, con el fin de proveer funcionalidades de orquestar, supervisar y administrar servicios lo cual reduce el costo y complejidad en la

integración de aplicaciones a través del uso de servicios Web. Así, el sistema de intermediación actúa como un integrador de servicios el cual a su vez es un proveedor del servicio que publica las descripciones de los servicios compuestos que ellos crean. En este sentido, el sistema de intermediación consolida los servicios que proporcionan otros proveedores de servicio con un valor agregado de servicio distinto.

Capítulo 3
Sistema de Integración de Procesos de Negocio en la Cadena de Suministro

Un servicio Web es esencialmente la infraestructura de comunicaciones que permite invocar un método público ofrecido por un servidor. Mediante los servicios Web es posible llevar a cabo la integración de organizaciones comerciales ya que los servidores son independientes de la plataforma operativa, siendo necesario conocer solamente la especificación del servicio Web para realizar su invocación. Así, uno de los mayores usos de los servicios Web reside en el comercio electrónico B2B y en la administración de la cadena de suministro. En este capítulo se presenta un sistema llamado BPIMS-WS (*Business Processes Integration and Monitoring System for Web Services*) capaz de integrar los procesos de negocio de diferentes empresas en el contexto del comercio electrónico B2B y en la administración de la cadena de suministro. Además, se presenta el diseño de BPIMS-WS y los componentes situados en cada capa. También se argumenta la funcionalidad de cada capa, así como de sus componentes.

3.1. BPIMS-WS: Sistema de Integración y Supervisión de Procesos de Negocio basados en Servicios Web

BPIMS-WS contiene un nodo UDDI [32], en donde se tienen registradas a las empresas, los servicios (procesos de negocio) y los productos que ofrecen. Para llevar a cabo esta clasificación de negocios, productos y servicios en el repositorio, BPIMS-WS utiliza ontologías ampliamente aceptadas como NAICS [68], UNSPSC [69] y RosettaNet [70].

RosettaNet es un consorcio de compañías dedicadas a la tecnología de la información, componentes electrónicos, manufactura de semiconductores, telecomunicaciones, entre otras. Este consorcio tiene como objetivo crear e implementar estándares de procesos de negocio abiertos y que sean ampliamente aceptados en la industria. Estos estándares sirven fundamentalmente para formar un lenguaje de negocio común en la que los socios de un proceso de negocio se puedan incorporar libremente a una cadena de suministro [70]. RosettaNet ha establecido lo que se conoce como *Partner Interface Processes* (PIPs) los cuales definen procesos de negocio entre socios comerciales. Los PIPs son interfaces basadas en XML especializadas en la comunicación entre sistemas. Se han clasificado en siete grupos de proceso de negocio capaces de crear una red comercial. En esta tesis, se utiliza solo el grupo 3 de Administración de Ordenes. Este grupo ayuda en el área de administración de órdenes de los negocios a partir del precio y cantidad de un producto. El grupo está subdividido en segmentos dentro de los que destaca el segmento 3A. Este segmento especifica la entrada para obtener la cantidad y orden de un producto. Permite el intercambio de información sobre el precio y disponibilidad, cantidad, órdenes de compra y estado de una orden entre los socios así como el envío de órdenes. Finalmente, el segmento 3A está conformado por varios PIPs. Los que resultan de interés en esta tesis son el 3A1 y el 3A2. El primero es conocido como Petición de Cantidad y, como su nombre lo indica, permite a un comprador solicitar a un proveedor la cantidad de productos disponibles y a un proveedor regresar en respuesta la cantidad. Una vez que el comprador ha establecido el proveedor con el que desea hacer la petición de compra, la única información que es necesario que envíe es el código del producto. Mientras que el segundo es conocido como Petición de Precio y Disponibilidad. Su objetivo es el de proveer un proceso automatizado rápido para los socios comerciales que les permita solicitar y proveer el precio de un producto y su disponibilidad. En esta tesis, RosettaNet permitió la clasificación de los servicios provistos por BPIMS-WS.

Por otro lado, el *United Nations Standard Products and Services Code* (UN-SPSC) es una ontología promovida por las Naciones Unidas que facilita la clasificación de productos y servicios a través de estándares globales y abiertos. UN-SPSC se usó para clasificar los productos que ofrece BPIMS-WS.

NAICS (*North American Industry Classification System*) es un sistema de clasificación industrial que agrupa a los comercios en industrias, basándose en las actividades principales a que se dedican los mismos. Posee 20 sectores y 1,170 industrias en el NAICS de Estados Unidos. NAICS fue desarrollado para proveer definiciones estándares de industrias para Canadá, México y los Estados Unidos, facilitando los análisis de la economía de los tres países de Norte América. NAICS se utilizó para la clasificación de negocios en BPIMS-WS.

Al igual que un nodo UDDI [32], BPIMS-WS ofrece servicios de publicación y consulta. Los servicios de publicación son servicios Web que permiten registrar a las empresas, los servicios y productos que ofrecen, mientras que los servicios de consulta son servicios Web que permiten realizar operaciones de búsqueda de empresas, procesos de negocio y productos. BPIMS-WS ofrece adicionalmente servicios que no se presentan en ningún nodo UDDI. Estos servicios son llamados servicios Web compuestos. Un servicio Web compuesto es un servicio Web formado de 2 o más servicios Web simples. Un servicio compuesto se crea, concretiza y ejecuta de manera dinámica en una máquina BPEL4WS [71]. Dentro de los servicios Web compuestos está la búsqueda de productos en base a sus características técnicas como precio, tiempos de entrega, cantidades en existencia, entre otros. Así también, está la compra de productos bajo restricciones de mercado como comprar por mejor precio, tiempo de entrega o compra distribuida, por mencionar algunos. De igual forma, BPIMS-WS presenta servicios Web intra-worflows e inter-workflows. Un servicio Web intra-workflow son orquestaciones estructuradas de servicios Web compuestos y representan el comportamiento interno de una empresa. Mientras que los servicios Web inter-workflows son composiciones de servicios Web intra-workflow y representan el comportamiento de una federación de empresas.

Además, BPIMS-WS presenta una arquitectura híbrida basada en servicios Web tomando características de las arquitecturas orientadas a servicios (SOA) y del bus de servicios Grid (SGB) presentadas en el capítulo anterior. En el contexto de las arquitecturas orientadas a servicios, BPIMS-WS actúa como una plataforma de administración de procesos de negocio basada en el paradigma de SOA para facilitar la creación y ejecución de aplicaciones orientadas a procesos en una forma modular y transparente. En este sentido,

BPIMS-WS provee un conjunto de servicios Web para la publicación, búsqueda e invocación, los cuales se explican a continuación:

- **Publicación** comprende servicios Web que realizan operaciones de almacenamiento de información en el repositorio de servicios acerca de: (i) potenciales socios de negocio (*businessName*, *description*, *discoveryURL*, *contactName*, *phone* y *e-mail*), (ii) productos (*productCode* y ontology), y (iii) servicios (*serviceName*, *description*, *accessSOAP*, *accessWSDL* y *serviceCode*). Ejemplos de estos servicios son *save_Business*, *save_Services* y *save_Products*.
- **Búsqueda** constituye servicios Web que realizan operaciones de búsqueda acerca de información técnica de productos. Ejemplos de estos servicios son *get_ProviderURLBuy*, *get_ProviderURLPriceDelivery* y *get_ProviderURLQuantity*. Así también, comprende un conjunto de servicios Web enfocados en la recuperación de información tanto de servicios como de meta-información de BPIMS-WS. Ejemplos de estos servicios son *get_BusinessOntology*, *get_ServicesOntology*, *get_ProductsOntology*, *find_Product*, *get_ServicesList*, *get_RegisteredBusiness*, *find_Service*, *find_Business* y *get_PasswordRecovery*.
- **Invocación** comprende servicios Web que invocan los procesos de negocio relacionados con la procuración de productos y servicios. Ejemplos de estos servicios son *get_Quantity*, *get_Price*, *get_DeliveryTime*, *get_Warranty*, get_PriceandDeliveryTime, *get_ProductsByProvider* y *get_ProviderQuantity*.

En el contexto del bus de servicios Grid, BPIMS-WS proporciona una infraestructura de software con mecanismos de integración para aplicaciones conducidas por eventos las cuales ocurren a lo largo de las aplicaciones existentes y principalmente se define por su significado comercial y su granularidad. Sin tener en cuenta la granularidad del evento, BPIMS-WS se asegura que las partes involucradas, comúnmente otras aplicaciones, sean notificadas inmediatamente cuando un evento ocurre. En este sentido, BPIMS-WS contiene un conjunto de servicios Web los cuales realizan operaciones de suscripción y notificación de eventos. Estos tipos de servicios se explican a continuación:

- **Suscripción** comprende un servicio donde diversas partes interesadas publican sus eventos de forma automática con el fin de incorporar información en los procesos de

negocio. Este servicio se implementó con los siguientes servicios Web: *subscribe*, *getSubscribeId* y *deleteSubscribeId.*

- **Notificación** introduce comunicaciones asíncronas en la cual la información se envía sin esperar una respuesta inmediata o el requisito de mantener una conexión entre dos sistemas mientras se espera por una respuesta. Este servicio se implementó utilizando los siguientes servicios Web: *NotifyAvailability* y *NotifyCancelPurchaseOrder.* El servicio Web *NotifyAvailability* envía un mensaje al comprador cuando un proveedor está disponible en referencia con un producto requerido. El servicio Web *NotifyCancelPurchaseOrder* envía un mensaje al comprador como respuesta de una cancelación de una orden de compra.

Estas operaciones se ejecutan por el servicio de intermediación provisto por BPIMS-WS, el cual, a continuación se describe en detalle su arquitectura.

3.2. Arquitectura de BPIMS-WS

BPIMS-WS se base en una arquitectura de 7 capas. En la Fig. 3.1 se muestra la arquitectura conceptual de BPIMS-WS. En esta figura, cada capa provee las prestaciones de servicios para la siguiente capa superior, resguardando la capa de los detalles de cómo las prestaciones se implementan. Las capas son abstraídas de tal manera que cada empresa, sistema o agente de software se comunique con cualquier capa. Así, si una federación de empresas desea comunicarse con el sistema, la información que envíe la federación debe de pasar por todas las capas inferiores. En este sentido, cada capa tiene una función bien definida. En las siguientes secciones se explica la funcionalidad de cada capa.

3.2.1 Capa de Dominio de las Ontologías

Esta capa define una taxonomía para representar la información provista por diversa información de productos, servicios y negocios. En esta capa se provee un conjunto de repositorios de ontologías que representan la terminología, el diccionario de datos y el dominio de la información. De acuerdo a [72], una terminología es la pila completa de los conceptos, sus definiciones y nombres en un dominio específico.

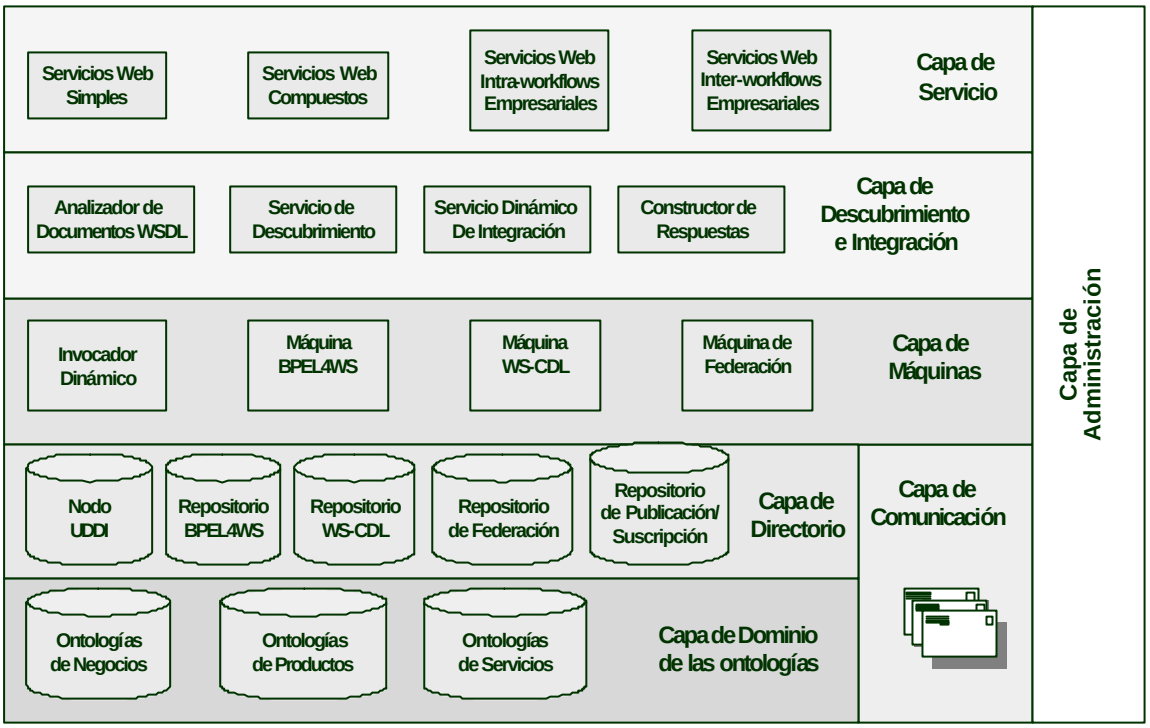


Fig. 3.1 Arquitectura conceptual de BPIMS-WS

Un diccionario de datos es una colección de datos los cuales pueden describirse e interpretarse como conceptos con contexto. En este sentido, el diccionario de datos se utiliza para describir diferentes dominios de la información como productos electrónicos, de computación, entre otros; y para describir diferentes escenarios donde la información se puede aplicar como sucede en los procesos de negocio donde cada empresa puede utilizar diferentes tipos de actividades comerciales para llevar a cabo un proceso de negocio.

También en esta capa se utiliza la noción de ontología de dominio de acuerdo con Gruber [73]. Una ontología es una jerarquía de conceptos con atributos y relaciones, que define una terminología consensuada para definir redes semánticas de unidades de información interrelacionadas. Una ontología proporciona un vocabulario de clases y relaciones para describir un dominio, poniendo énfasis en la compartición del conocimiento y el consenso en la representación de éste. Por ejemplo, una ontología sobre arte podría incluir clases como Pintor, Cuadro, Estilo o Museo, y relaciones como autor de un cuadro, pintores pertenecientes a un estilo artístico u obras localizadas en un museo. Entonces, una ontología de dominio provee las especificaciones formales y las definiciones estándares de

los términos utilizados para representar el conocimiento en dominios específicos de tal forma que puedan ser computacionalmente entendibles para poder llevar acabo la interacción y comunicación con otros dominios. Para representar el conocimiento en una ontología, la Web semántica provee diferentes lenguajes entre los que destacan SHOE, DAML y OWL. SHOE es el acrónimo de *Simple HTML Ontology Extension* y es un lenguaje compatible con XML para la representación del conocimiento en la Web [74]. DAML es el acrónimo de *Darpa Agent Markup Language* y es un lenguaje basado en XML para la descripción de ontologías en la Web [75]. OWL es el acrónimo de *Web Ontology language* y provee un marco de trabajo para la administración de recursos, integración de empresas, el reuso y la compartición de datos en la Web [76].

3.2.2 Capa de Directorio

Esta capa comprende un conjunto de repositorios que almacenan las descripciones de los servicios e información de meta-datos. Dentro de la información de meta-datos están la categoría, propiedades, capacidades, localización e información de acceso de los servicios disponibles. Las capas superiores utilizan éstos meta-datos para localizar y consultar los repositorios que contienen las descripciones de los servicios Web. Para esto, BPIMS-WS provee componentes de software que permiten publicar y descubrir servicios Web para la integración de los procesos de negocio. BPIMS-WS usa el nodo UDDI para el descubrimiento de los servicios Web. Los componentes de software consultan la información guardada en el nodo UDDI mediante ODBC/JDBC.

Además, en esta capa BPIMS-WS utiliza un repositorio de procesos de negocio para la creación y ejecución de servicios Web compuestos. Este repositorio presenta ciertas similitudes a un nodo UDDI. La estructura del repositorio BPEL4WS en comparación con el nodo UDDI, se muestra en la Fig. 3.2.

En un nodo UDDI, el principal elemento es *BusinessEntity* mientras que en el repositorio BPEL4WS es *ProcessEntity*. El elemento *ProcessEntity* presenta los siguientes elementos:

- ***personName*** es el nombre de la persona a la que pertenece el proceso. Esta es la persona de contacto.
- ***phone*** corresponde al número telefónico de la persona de contacto

- ***email*** es la dirección de correo electrónico de la persona de contacto
- ***address*** corresponde a la ubicación física de la persona de contacto
- ***discoveryURL*** es la dirección electrónica del sitio Web de la persona de contacto
- ***name*** corresponde al nombre del proceso de negocio
- ***descritpion*** es una breve descripción de la funcionalidad del proceso de negocio
- ***processKey*** corresponde al identificador único del proceso de negocio en el repositorio
- ***subProcessKey*** corresponde al identificador de los subprocesos de negocio que forman al proceso. Este es en el caso de que un proceso de negocio esté formado por dos o más procesos
- ***ontology*** es la ontología a la que pertenece el proceso de negocio
- ***codeOntology*** corresponde al Código de la ontología a la que pertenece el proceso de negocio
- ***classOntology*** corresponde a la Clase de la ontología a la que pertenece el proceso de negocio

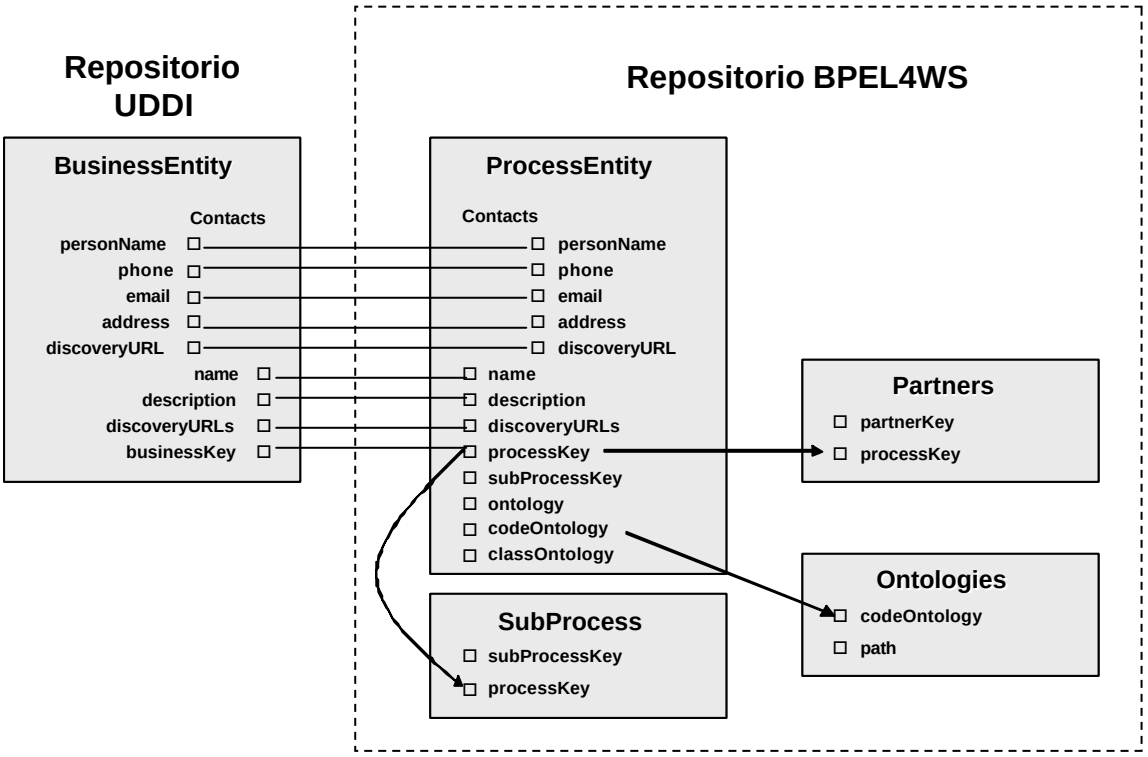


Fig. 3.2 Estructura del repositorio BPEL4WS de BPIMS-WS y su comparación con el repositorio UDDI

Dentro del repositorio de BPEL4WS existen 3 relaciones con otros elementos. La primer relación es entre el elemento *processKey* de *ProcessEntity* y su correspondiente en el elemento *SubProcess*. Esta relación establece que un proceso puede constituirse de dos o más subprocesos identificándolos con el elemento *subProcessKey*. La segunda relación es entre el elemento *codeOntology* con su correspondiente en el elemento *Ontologies*. Esta relación establece la ubicación electrónica en dónde se encuentra la ontología a la que pertenece un proceso. Finalmente, la ultima relación, es la que se da entre el elemento *processKey* y su correspondiente en el elemento *Partners*. Esta relación establece a qué socio comercial pertenece un proceso de negocio.

Finalmente, como se aprecia en la Fig. 3.1, esta capa contiene un repositorio de publicación y suscripción donde se almacenan las solicitudes que BPIMS-WS atiende en forma asíncrona para realizar un proceso comercial. Este mecanismo de publicación/suscripción es característico de las arquitecturas manejadas por eventos (EDA, *Event-Driven Architectures*) y del Bus de Servicios Grid. BPIMS-WS utiliza este repositorio para almacenar las solicitudes que necesitan atenderse mediante una comunicación asíncrona. Para esto, BPIMS-WS utiliza a SOAP el cual soporta cuatro patrones de interacción: solicitud-respuesta, notificación-respuesta, notificación del cliente al servidor y viceversa. En este sentido, SOAP permite interacciones síncronas y asíncronas.

3.2.3 Capa de Comunicación

En esta capa se proporciona el servicio de mensajería que utiliza BPIMS-WS. El servicio de mensajería permite almacenar y trasmitir los mensajes involucrados en los procesos de negocio que realiza BPIMS-WS con el fin de proveer capacidades de extracción, verificación de integridad y auditoria de mensajes. La extracción de mensajes implica la recuperación de datos comerciales para la toma de decisiones. BPIMS-WS determina las características y las cantidades de nueva información de los clientes que están procesándose. Por ejemplo, BPIMS-WS es capaz de identificar y desplegar los participantes y la interacción que éstos tienen durante la ejecución del servicio Web ya que los mensajes SOAP de petición contienen esencialmente la información relativa al nombre del servicio Web solicitado así como del tipo y valor de los parámetros requeridos, mientras

que los correspondientes mensajes de respuesta contienen el tipo y valor de los resultados producidos por el servicio. En este sentido, BPIMS-WS da seguimiento a un proceso de negocio de manera visual, permitiendo no solo a los programadores de servicios Web depurar sus aplicaciones, sino también a los usuarios verificar el desarrollo de los servicios. Para esto, BPIMS-WS intercepta los mensajes SOAP que son intercambiados por los servicios Web, los cuales se almacenan en un búfer. La Fig. 3.3 muestra la arquitectura de BPIMS-WS en la capa de comunicación. Básicamente un servicio Web utiliza los protocolos SOAP, HTTP y TCP/IP, en un modelo organizado en capas. En esta figura, los mensajes SOAP (de solicitud y respuesta) se envuelven en sobres correspondientes a cada capa de acuerdo al modelo de la OSI.

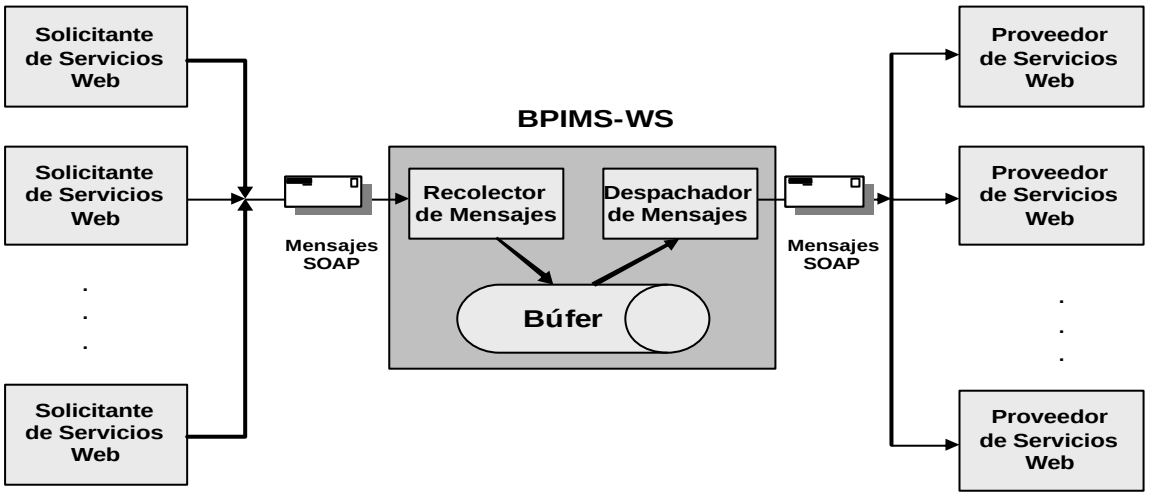


Fig. 3.3 Arquitectura de BPIMS-WS en la capa de comunicación

El contexto en el que se desarrollan los servicios Web se muestra en la Fig. 3.4. En ambos lados de esta figura se sitúan las aplicaciones de flujo de trabajos y una máquina virtual BPEL4WS/VM de documentos BPEL4WS. BPIMS-WS está situado en el nivel de aplicaciones Web y aplicaciones de procesos de negocios descritos en BPEL4WS. En caso de que BPIMS-WS dejara de funcionar por algún factor externo (corte en el suministro de energía, mal funcionamiento del servidor de aplicaciones, por mencionar solo algunos), un búfer persistente podría guardar los mensajes que se pudieran perder.

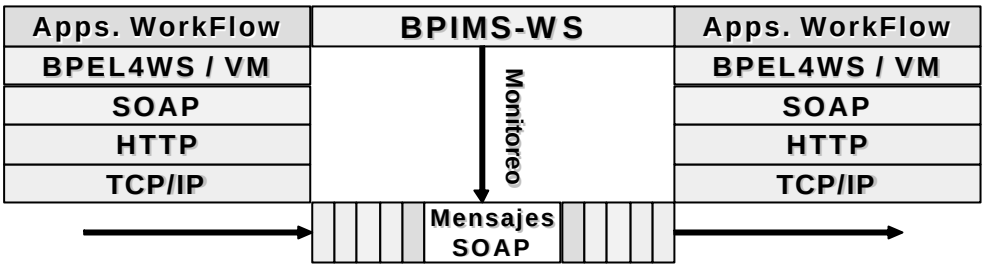


Fig. 3.4 Contexto de los servicios Web

Los mensajes podrían compararse con otros mecanismos de almacenamiento de mensajes para asegurar la integridad de la transmisión de los mensajes. Esto permitiría mantener los estados entre dos o más componentes. Actualmente, BPIMS-WS provee este tipo de mensajería segura debido a que se utiliza WS-RM para la transmisión de los mensajes [77]. Debido a que SOAP no provee mecanismos que garanticen la entrega de mensajes, fue necesaria una migración del servicio de mensajería basado en WS-RM para solucionar este problema. WS-RM proporciona los mecanismos para el intercambio de información de manera fiable sobre Internet entre socios de negocio y recibe una notificación de entrega, una capacidad que no se consideró previamente por las especificaciones existentes de servicios Web [77]. Así también, el uso de mecanismos de almacenamiento persistentes permitiría la evaluación del tráfico de los mensajes. Por ejemplo, podría examinarse la carga de tráfico de los mensajes y modificaciones de los contenidos de los mensajes en un periodo para poder restaurar la información o para realizar un análisis para una toma de decisión. BPIMS-WS no provee esta característica dentro de la capa de comunicación.

3.2.4 Capa de Máquinas

Esta capa provee el soporte para las interacciones entre los servicios Web. En esta capa se provee un conjunto de máquinas que ejecutan las invocaciones de los procesos de negocio descritos como servicios Web. En esta capa, los negocios pueden compartir sus procesos de negocio de acuerdo a estándares B2B específicos como EDI (*Electronic Data Interchange*), RosettaNet o cXML los cuales definen formatos comunes y la semántica para los mensajes

(como una orden de compra) y las conversaciones de procesos de negocio para ejecutarlas en la máquina correspondiente. Dentro de esta capa se tiene la siguiente lista de máquinas:

1. **Invocador Dinámico**. Permite la ejecución de servicios Web descritos en WSDL. Para esto, el invocador dinámico construye las solicitudes de los mensajes SOAP.
2. **Máquina de Ejecución BPEL4WS.** Permite la ejecución de flujos de trabajos que representan procesos de negocio compuestos. Un proceso comercial compuesto está constituido por dos o más servicios Web descritos en WSDL que representan actividades comerciales simples como la obtención del precio o la obtención de la cantidad en existencia de un producto específico.
3. **Máquina de Ejecución WS-CDL.** Permite la ejecución de flujos de trabajos que representan el comportamiento y las relaciones de una entidad empresarial descritas en flujos de trabajos BPEL4WS. En este contexto, una entidad empresarial puede entenderse como un departamento de facturación en donde se llevan a cabo un conjunto de transacciones, incluyendo el intercambio de órdenes de pedido, facturas, información de envío, etc., a través de una red como Internet o una red privada virtual (VPN).
4. **Máquina de Federación.** Permite la ejecución de flujos de trabajos que representan el comportamiento, los acuerdos, las políticas y relaciones comerciales entre organizaciones representadas como flujos de trabajos WS-CDL [78]. Bajo este contexto, interviene la compartición de información estática o dinámica, incluyendo niveles de inventario, planificaciones, previsiones, políticas de mercado y comunidades de subastas así como las licitaciones, para permitir el desarrollo de los negocios y la integración a las empresas.

En esta capa, cada máquina de ejecución provee una funcionalidad específica de acuerdo al grado de complejidad y granularidad del proceso comercial que ejecuta. Por ejemplo, si se desea invocar una solicitud de precio el cual es un proceso comercial simple, el invocador dinámico de servicios Web atendería esta solicitud. En caso de que se quiera llevar a cabo un proceso comercial más elaborado, como es el caso de una orden de compra donde comúnmente involucra 3 procesos de negocio (solicitud de precio, solicitud de cantidad en existencia y formulación de la orden), la maquina BPEL4WS es la indicada para realizar

este proceso. Para el caso de las máquinas WS-CDL y de federación, la cuales no se tiene alguna implementación, se espera obtener los mismos resultado debido a que el modelo de orquestación de WS-CDL es un complemento para lenguajes como BPEL4WS y Java. WS-CDL ofrece a estos lenguajes un modelo global que necesitan para asegurar que su comportamiento, relaciones y políticas sea coherente a través de servicios en cooperación.
Bajo este enfoque de orquestación gradual, máquinas con una granularidad gruesa de ejecución de flujos de trabajos podrían desplegar, ejecutar y administrar procesos de negocio de aquéllas máquinas que coordinan procesos de negocio e interacciones en los flujos de las colaboraciones comerciales con una granularidad fina. Aquí, las colaboraciones comerciales pueden ser síncronas o asíncronas, por lo que cada máquina debe almacenar las instancias correspondientes en su base de datos, en espera de un llamado por evento (*callback*) de los socios comerciales remotos y otras actividades involucradas en la colaboración comercial.
Finalmente, cada máquina verifica la compatibilidad de la versión de los procesos de negocio y administra la entrega de mensajes SOAP a los destinos remotos.

3.2.5. Capa de Descubrimiento e Integración

En esta capa se realiza el descubrimiento e integración de servicios Web. Para realizar el proceso de descubrimiento de servicios Web, se utilizan mecanismos que transforman los mensajes que se intercambian en un proceso de negocio, al formato correspondiente de la aplicación receptora. En este sentido, se tiene un diccionario común que contiene información de cómo cada componente se comunica con otro, así como el significado de la información. Los mecanismos de transformación de mensajes incluyen análisis gramatical y métodos de emparejamiento que describen la estructura de cualquier formato de mensaje. Los formatos de mensajes se construyen por piezas de información que representan una solicitud de una funcionalidad encapsulada dentro del mensaje. El proceso de transformación de mensajes permite almacenar información de las aplicaciones en la capa de directorios con el fin de inspeccionar el funcionamiento de BPIMS-WS. Así también, el proceso de transformación de mensajes incluye la conversión de esquemas y de datos. Esto es importante para asegurar la consistencia de la información entre todos los componentes

involucrados cuando se realiza un proceso de negocio. Una conversión de esquema es el proceso de cambiar la estructura de un mensaje para que se acepte por un componente destino. BPIMS-WS realiza este proceso de forma dinámica a través de reglas de procesamiento predeterminadas las cuales transforman los datos dependiendo de su contenido y esquema. En este contexto, una regla de procesamiento provee el flujo de control y distribución de los mensajes. En el proceso de conversión de datos, BPIMS-WS determina el formato de los datos de los componentes que participan para llevar a cabo un proceso de negocio para saber qué elementos deben extraerse y convertirse, y finalmente dónde ellos necesitan ponerse. BPIMS-WS convierte tipos de datos a otro. Por ejemplo, BPIMS-WS es capaz de convertir información numérica en alfanumérica (y viceversa), información numérica o alfanumérica a sentencias SQL, por mencionar solo algunas. Para esto, BPIMS-WS utiliza tablas de mapeo para la conversión de datos en tiempo de ejecución. También, esta capa se provee un mecanismo de asignación de ruta interna de los mensajes basado en su contenido, identificando un mensaje proveniente de una aplicación, enviándolo a la aplicación destino y transformándolo en caso de que se requiera. Por ejemplo, cuando un mensaje llega a BPIMS-WS, se analiza y se identifica, Luego, se aplican las reglas de procesamiento predeterminadas, incluyendo la transformación del mensaje. Una vez procesado, BPIMS-WS envía el mensaje a la aplicación correcta.

La arquitectura para el descubrimiento e integración de servicios en BPIMS-WS se muestra en la Fig. 3.5 En esta figura, BPIMS-WS recibe una solicitud de un mensaje SOAP y la transforma a sentencias SQL para realizar una consulta al nodo UDDI (Pasos 1-3 en Fig. 3.5). El resultado de la consulta son las direcciones electrónicas de los documentos WSDL, BPIMS-WS analiza estos documentos de cada negocio encontrado en la consulta, extrae la información necesaria de los servicios Web de los negocios y construye las solicitudes de invocación a los servicios Web de los negocios (Pasos 4-6 en Fig. 3.5). BPIMS-WS recibe las respuestas a las solicitudes, extrae la información que le es útil y crea una nueva respuesta (Pasos 7 y 8 en Fig. 3.5). Finalmente, BPIMS-WS envía esta nueva respuesta a quien le formuló la solicitud (Paso 9 en Fig. 3.5).

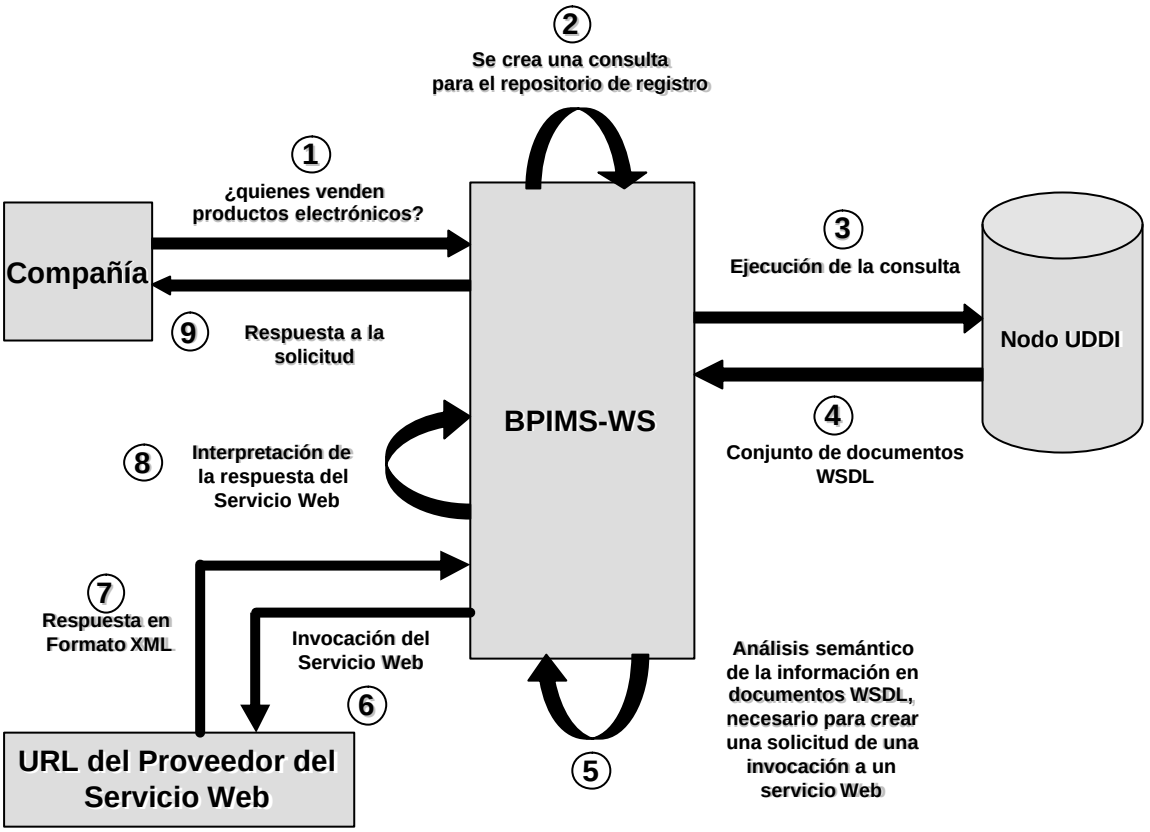


Fig. 3.5 Esquema general de la invocación e integración de servicios Web en BPIMS-WS

3.2.6. Capa de Servicio

La constituye el conjunto de servicios Web ofrecidos por BPIMS-WS. En BPIMS-WS, los servicios Web se clasifican de acuerdo al nivel de funcionalidad que ofrecen, es decir, a la capacidad del servicio. BPIMS-WS clasifica los servicios Web en: (1) simples, (2) compuestos, (3) intra-worflows empresariales, y (4) inter-workflows empresariales. A continuación se explica en detalle la funcionalidad de cada uno.

3.2.6.1. Servicios Web simples

Los servicios Web simples representan operaciones básicas de:

a) **Registro.** Agrupa servicios Web que permiten almacenar información dentro del nodo UDDI de BPIMS-WS sobre: (i) los potenciales socios comerciales (*businessName*, *description*, *discoveryURL*, *contactName*, *phone* y *email*), (ii) productos (*productCode* y *ontology*), y (iii) servicios (*serviceName*, *description*, *accessSOAP, accessWSDL* y *serviceCode*). Ejemplos de estos servicios Web son *save_Business, save_Services* y *save_Products*.

b) **Eliminación.** Incluye servicios Web que permiten eliminar información previamente registrada de negocios, productos y servicios dentro del nodo UDDI. Ejemplos de estos servicios Web son *delete_Business, delete_Products* y *delete_Services.*

c) **Búsqueda.** Contiene servicios Web que permiten realizar operaciones de consulta al nodo UDDI de BPIMS-WS. Dentro de estas operaciones está la búsqueda de productos, negocios y servicios, la recuperación de la descripción de los servicios de solicitud de información técnica de un producto, precio de un producto, tiempo de entrega, cantidad en existencia, orden de compra, por mencionar solo algunos. Algunos ejemplos de estos servicios Web son: *get_Quantity*, *get_Price*, *get_DeliveryTime* y *get_Warranty*.

d) **Información.** Agrupa servicios Web que permiten recuperar información de servicio y meta-información de BPIMS-WS. Dentro de estas operaciones está la recuperación de las ontologías de negocios, productos y servicios, la recuperación de los servicios Web provistos por BPIMS-WS, entre otros. Algunos de estos servicios Web son *get_BusinessOntology*, *get_ServicesOntology*, *get_ProductsOntology*, *get_ServicesList*, *get_RegisteredBusiness*, *find_Product*, *find_Service*, *find_Business* y *get_PasswordRecovery*.

El objetivo de los servicios Web simples es proveer un conjunto de operaciones comerciales básicas en la administración de la cadena de suministro con el fin de proveer el soporte para flujos de trabajos más complejos bajo el enfoque de una composición gradual en base a la funcionalidad de los servicios. Por ejemplo, en un escenario típico de la cadena de suministro, diversas compañías necesitan conocer la información actualizada de la información técnica de los productos ofrecidos por un proveedor, para esto los servicios Web simples ofrecen una solución proporcionando información acerca del producto como el precio y tiempo de entrega, además de información de contacto del los proveedores como su nombre, ubicación y la clase de servicio que ofrecen.

La funcionalidad del uso de servicios Web simples en BPIMS-WS se muestra en la Fig. 3.6.

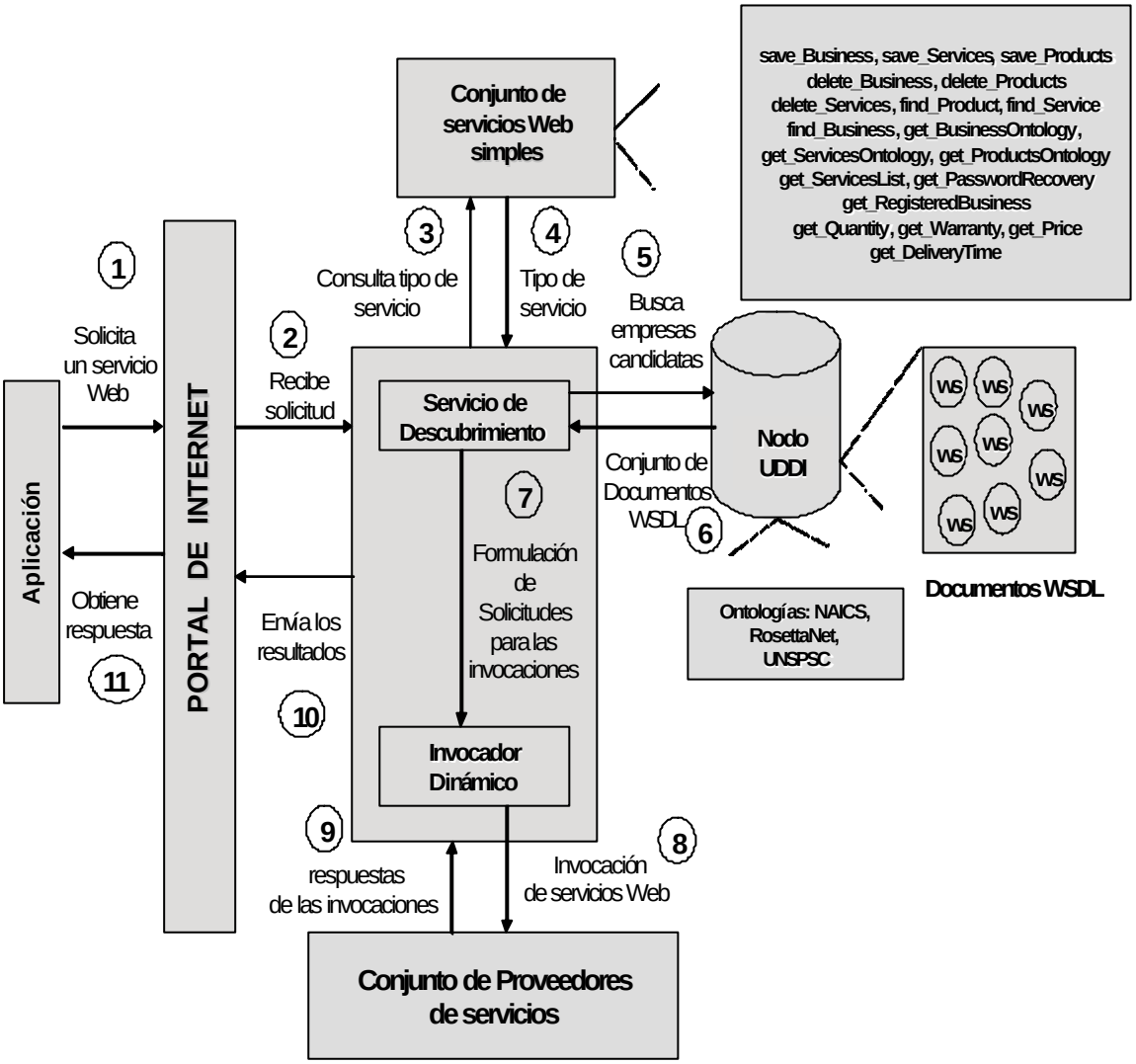


Fig. 3.6 Estructura y Funcionalidad de BPIMS-WS con servicios Web simples

La estructura y comportamiento de esta funcionalidad puede entenderse con el siguiente ejemplo el cual servirá para mostrar el flujo de datos en BPIMS-WS. Suponga que un cliente desea encontrar la información técnica disponible de un producto de su preferencia. Primero, el cliente debe seleccionar el tipo de producto dentro de un rango de opciones ofrecidas a través del portal de Internet (Paso 1 en Fig. 3.6). Entonces, BPIMS-WS obtiene la solicitud e identifica el tipo de servicio Web que se le solicita (Paso 2-4 en Fig. 3.6). En

base al tipo de servicio Web solicitado, BPIMS-WS formula una consulta al nodo UDDI (Paso 5 en Fig. 3.6). El resultado de la consulta es una lista de todos los proveedores que tienen el producto pedido en sus inventarios. Entonces, para cada uno de estos proveedores, BPIMS-WS formula otra consulta al nodo UDDI, para recuperar la dirección electrónica (*URL*) del servicio Web correspondiente al PIP 2A5 de RosettaNet donde se puede solicitar la información técnica del producto (Paso 6 en Fig. 3.6). Una vez localizado el URL, BPIMS-WS construye las solicitudes SOAP para la invocación de los servicios Web asociados a las empresas encontradas (Paso 7 en Fig. 3.6). Entonces, BPIMS-WS envía estas solicitudes SOAP a las empresas y obtiene las respuestas correspondientes (Paso 8-9 en Fig. 3.6). Luego BPIMS-WS extrae la información requerida y construye un documento XML. Este documento XML se presenta en formato HTML usando una hoja de estilos. La respuesta contiene información técnica correspondiente al producto (según el servicio Web invocado) y la dirección electrónica (*discoveryURL*) de la empresa que ofrece el producto (Paso 10-11 en Fig. 3.6).

Por medio del uso de los servicios Web simples, un cliente puede obtener el precio, el tiempo de entrega o la cantidad disponible en inventario de cualquier empresa registrada en el nodo UDDI.

3.2.6.2. Servicios Web compuestos

Este tipo de servicios son orquestaciones de los servicios Web simples y representan operaciones más elaboradas. Entre estas operaciones está la búsqueda de proveedores bajo restricciones de precio, tiempo de entrega, cantidad en existencia, entre otros. A diferencia de los servicios Web simples, los servicios Web compuestos se crean en tiempo de ejecución. El objetivo de los servicios Web compuestos es poder utilizar operaciones comerciales básicas representadas en los servicios Web simples con el fin de crear flujos de trabajos comerciales con el fin de integrar las actividades comerciales que describen el comportamiento de un departamento de una empresa. Los servicios Web compuestos habilitan la interoperabilidad entre las aplicaciones que constituyen un departamento comercial. La arquitectura de BPIMS-WS para la creación de servicios Web compuestos se muestra en la Fig. 3. 7.

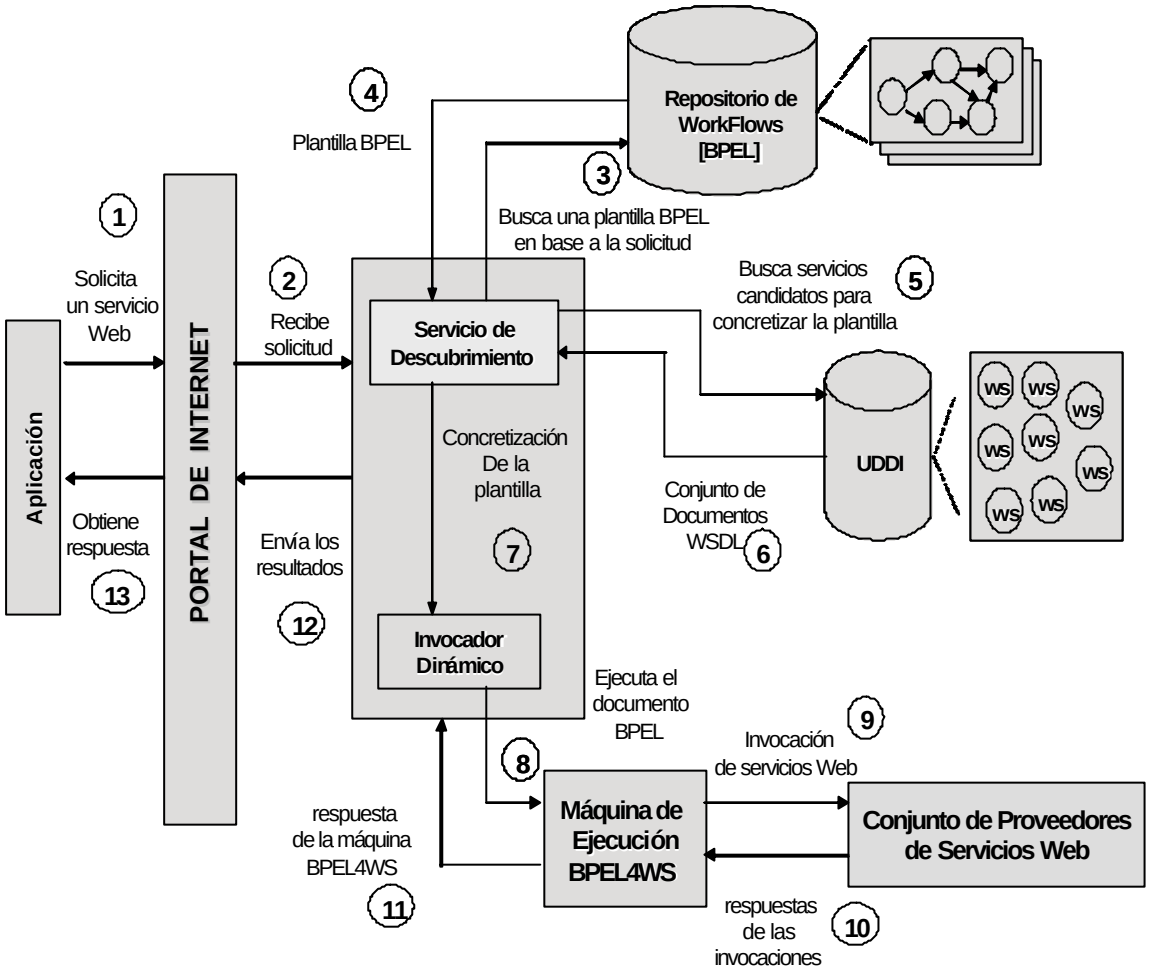


Fig. 3.7 Arquitectura de BPIMS-WS en la creación de servicios Web compuestos

Para la ejecución de un servicio Web compuesto primero es necesario localizar una plantilla del repositorio BPEL4WS que describe la actividad comercial. Suponga un escenario de una compra de libros, donde el cliente podría estar interesado en comprar un libro en la tienda que ofrece el precio más bajo o el tiempo de entrega mínimo (Paso 1 en Fig. 3.7). Si un cliente quiere comprar varios libros al precio más bajo, BPIMS-WS recupera de una base de datos la localización de la plantilla BPEL4WS que representa el criterio de compra seleccionado (Paso 2). Una vez que la plantilla se localiza, BPIMS-WS usa los documentos WSDL y todos los archivos de configuración relacionados para llevar a cabo el proceso de concretización. BPIMS-WS obtiene las plantillas adecuadas que pueden

usarse para encontrar a los proveedores que ofrecen el producto en base a los requerimientos del cliente.

Luego, BPIMS-WS realiza una consulta a una base de datos que contiene los documentos WSDL y recupera los servicios Web apropiados para obtener información comercial como precio, tiempo de entrega, cantidad en inventario, y la dirección electrónica de acceso a la compra del producto (Paso 3). Estos servicios, basados en las ontologías UNSPSC y RosettaNet, son *get_PriceandDeliveryTime*, *get_ProviderQuantity*, y *get_ProviderURLBuy*, respectivamente. Después, BPIMS-WS analiza los documentos WSDL relacionados, y recupera toda la información pertinente para concretizar las plantillas. Posteriormente, las plantillas concretizadas se almacenan en una máquina de inferencia BPEL4WS para su ejecución (Paso 4). Para la comunicación con el flujo de trabajo que se está ejecutando, BPIMS-WS construye mensajes SOAP con información proporcionada por el cliente. Después, el cliente envía al flujo de trabajo el código del libro y la cantidad requerida en un mensaje SOAP. El flujo de trabajo verifica que la suma de todas las cantidades es por lo menos la cantidad pedida por el cliente. Si no se cumple esta condición, una lista vacía se envía al cliente como respuesta, lo cual significa que la solicitud del cliente no puede cumplirse separadamente o en conjunto por las empresas registradas. En caso contrario, envía al cliente una lista de proveedores que satisfacen sus condiciones. Entonces, el flujo de trabajo se elimina de la máquina de ejecución. Después de que el cliente selecciona a un proveedor, BPIMS-WS recupera una plantilla BPWL4WS para hacer un pedido de la compra, la plantilla se completa y se ejecuta como se describió anteriormente.

3.2.6.3. Servicios Web *intra-workflows* empresariales

Este tipo de servicios son orquestaciones de los servicios Web compuestos. Esta orquestación se realiza en base a su documento WS-CDL [78]. Un documento WS-CDL representa un *intra-worflow*. Un *intra-workflow* define los comportamientos y las relaciones de las diversas entidades que constituyen una empresa. Una entidad puede ser un departamento de facturación o un departamento de mercadeo. Además, habilitan un mecanismo eficaz mediante el cual se pueden crear aplicaciones que constituyan soluciones de flujo de trabajo integrales. Estas soluciones son apropiadas para escenarios de mediana

ejecución, como las transacciones comerciales entre las diferentes entidades de una empresa. Para llevar cabo esta orquestación, se sigue un enfoque similar al descrito en la capa de servicios Web compuestos, donde primero se localiza la plantilla del repositorio de documentos WS-CDL que representa al servicio, después se concretiza en tiempo de ejecución para crear un documento WS-CDL con información proveniente de los servicios Web compuesto y simples, luego se ejecuta en la máquina de inferencia WS-CDL. Finalmente se obtienen y se muestran los resultados.

3.2.6.4. Servicios Web *inter-workflows* empresariales

Los servicios Web *inter-workflows* empresariales son composiciones de los servicios Web empresariales *intra-workflows*. Un *inter-workflow* define la conducta y las relaciones de las diversas empresas que constituyen una federación. Una federación puede verse como un conjunto de empresas que colaboran conjuntamente para lograr un interés o una meta en común. Los servicios Web empresariales *inter-workflows* están orientados a la satisfacción de necesidades de la sociedad como reducir el consumo de recursos naturales no renovables o aumentar la producción de alimentos sujeta a un presupuesto limitado. Además, permiten a las empresas integrar, administrar y automatizar flujos de trabajos empresariales mediante el intercambio de documentos comerciales (por ejemplo, órdenes de compra y facturas) entre aplicaciones que se encuentran fuera de los límites de la empresa definiendo el comportamiento de los servicio Web en la composición de procesos empresariales en los que intervienen varias entidades

3.2.7 Capa de Administración

En esta capa se lleva a cabo la administración de servicios que ofrece BPIMS-WS. BPIMS-WS realiza la administración de servicios Web usando la especificación *WS-Manageability*. La tecnología utilizada para la implementación fue JMX (*Java Management eXtension*). La arquitectura JMX consiste de tres niveles: instrumentación, agencia y servicios distribuidos. JMX proporciona interfaces y servicios adecuados para supervisar y administrar requerimientos de sistemas [79]. Esta funcionalidad involucra abstracción de recursos usando componentes llamados *MBeans* (*Managed Beans*) y accesibilidad de recursos

instrumentados remotamente a través de conectores JMX. Un *MBean* es un objeto de Java que representa un recurso administrado, como una aplicación, un servicio, un componente, o un dispositivo [79].

El componente principal para realizar la administración de servicios es un *JMX Bridge* que actúa como un puente entre los recursos administrados por JMX y los servicios Web. En vez de proveer una interfaz JMX de servicio Web específica, BPIMS-WS proporciona una interfaz de servicio Web a un recurso administrado. Bajo este enfoque, los recursos pueden implementarse en diferentes tecnologías ya que sólo es necesario definir una interfaz de servicio Web para un recurso. Para lograr esto, se utilizaron *MBeans* para representar un recurso que puede administrarse. El *JMX Bridge* es información que identifica o describe una instancia *MBean* que representa un recurso administrado específico. El *JMX Bridge* genera documentos WSDL que describen el servicio Web, una clase de Java que actúa como la implementación del servicio Web y un descriptor de despliegue para el servidor de aplicaciones Tomcat. Luego de esto, la clase de Java resultante debe compilarse y desplegarse como un servicio Web en el servidor de aplicaciones Tomcat. El documento WSDL generado puede utilizarse para realizar llamadas dinámicas, validar llamadas estáticas, o generar un *proxy*. En la Fig. 3.8 se muestra la arquitectura general de BPIMS-WS en la capa de administración. En la Fig. 3.8, se implementó un administrador genérico el cual se añadió al servidor de aplicaciones con el fin de reunir datos estadísticos.

El administrador invoca a un *proxy* de administración que almacena o actualiza los datos estadísticos en *MBeans* los cuales se concretizan y administran a través del servidor *JMX MBean*. El servidor *MBean* rastrea información global y estadística sobre los servicios Web como el *hostname* y el puerto específico donde residen los servicios. Lo más significativo de este administrador es la implementación de un servicio Web de administración basado en JMX que permite almacenar *MBeans* dentro de un servidor *JMX MBean* que puede accederse y manipularse desde una página Web. Bajo este enfoque es posible supervisar el estado de los servicios y componentes a través de un cliente SOAP. En forma conjunta, esta funcionalidad representa una administración activa de datos y recursos basados en JMX mediante el uso de interfaces estándares (SOAP). Además, cada *MBean* rastrea información global con respecto a la configuración y el uso del servidor. BPIMS-WS

rastrea y recupera esta información global desde páginas Web con el propósito de supervisar y administrar los servicios Web de forma remota.

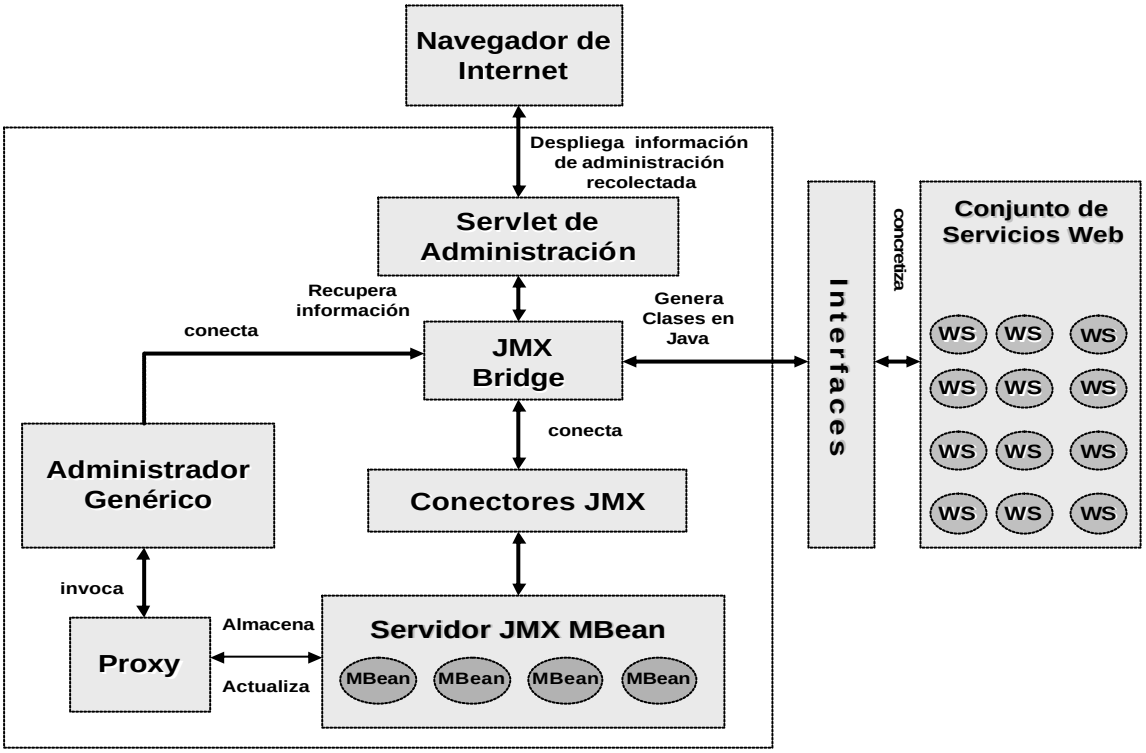


Fig. 3.8. Arquitectura de BPIMS-WS en la capa de administración

A continuación se presenta la arquitectura general de BPIMS-WS describiendo cada componente y sus relaciones, así como la funcionalidad de cada componente.

3.3. Descripción de los componentes de BPIMS-WS

Bajo la arquitectura de capas de BPIMS-WS, cada componente tiene una función definida. En la Fig. 3.9 se describe la arquitectura general de BPIMS en donde el:

Analizador de Mensajes SOAP

Determina la estructura y contenido de los documentos que se intercambian en los procesos de negocio involucrados en las colaboraciones de la cadena de suministro. Debido a que BPIMS-WS se basa en servicios Web como tecnología de información, este componente

determina la información involucrada en los mensajes SOAP entrantes por medio de herramientas y analizadores XML. Este componente recupera la información útil empaquetada en los mensajes SOAP y la transforma a una representación de un árbol de nodos DOM. Con esta representación se determina la lógica de las operaciones a realizar contenidas en los mensajes SOAP.

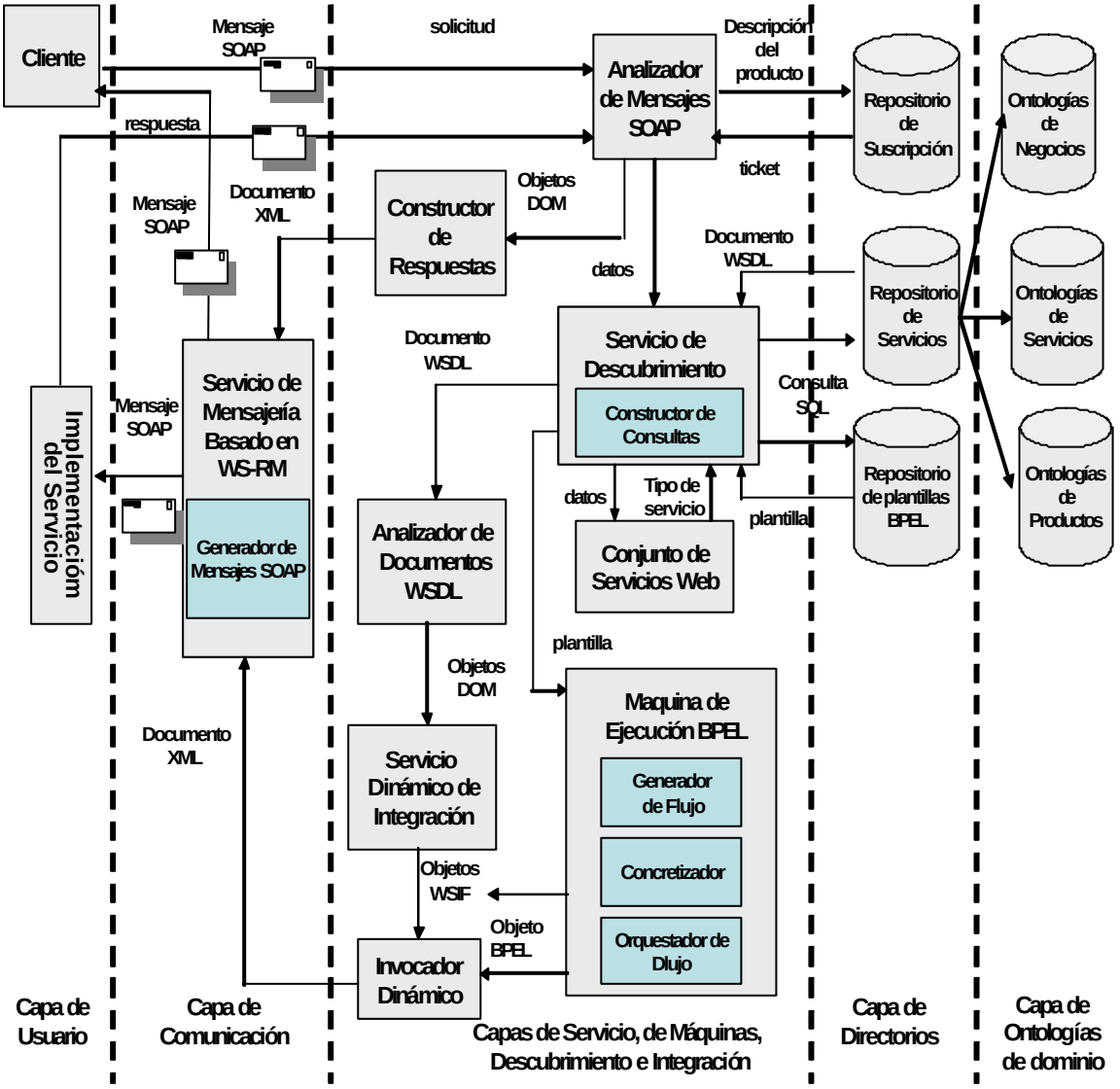


Fig. 3.9. Arquitectura General de BPIMS-WS incluyendo sus componentes

Para la generación de la estructura de árbol y para determinar la lógica de la aplicación para cada nodo de los mensajes SOAP, BPIMS-WS utiliza las APIs de DOM [20] y SAX [21], respectivamente. Para esto, BPIMS-WS contiene un conjunto de clases en Java basadas en JAXP [80] para el análisis sintáctico de documentos XML.

Repositorio de servicios

Es el mecanismo para registrar y publicar información referente a los procesos de negocio, productos y servicios entre los socios de negocio para actualizarla y adaptarla en escenarios de la cadena de suministro. Si la cadena de suministro incluye socios de negocio internos, es decir, todos los participantes forman parte de la misma organización, una base de datos centralizada con menos problemas de seguridad es preferible para mantener de manera fácil y eficiente la información. Sin embargo, si la cadena de suministro consiste de múltiples organizaciones en donde cada organización debe de mantener su propia base de datos y compartir la información con otros socios de negocio, un nodo UDDI donde las empresas registren sus productos y servicios es recomendable. Para la implementación, se utilizó una base de datos relacional comercial debido a las fuertes limitaciones que presenta la versión 3.0 de UDDI, la cual no provee el soporte suficiente para información no funcional, como las clasificaciones de productos y procesos de negocio organizados en ontologías. Además, UDDI está diseñado principalmente para aplicaciones de razonamiento automatizado no intensivo por lo que no ofrece un soporte para descripciones semánticas. Estas limitaciones fueron superadas agregando sólo el soporte a ontologías de procesos de negocio, permitiendo descubrimiento dinámico de servicios Web de una forma rápida y sencilla. Para la clasificación de procesos de negocio, productos y servicios en el repositorio de servicios, se usaron ontologías ampliamente aceptadas como NAICS, UNSPSC y RosettaNet. NAICS es un sistema estándar de clasificación para la industria de Norte América; UNSPSC proporciona un estándar abierto y global en diversos sectores para una clasificación eficiente de productos y servicios, y; RosettaNet define los diccionarios técnicos y de negocio.

Repositorio de Suscripción

Es el mecanismo para registrar las interacciones que publican las aplicaciones acerca de la información de un evento con el fin de que otras que estén suscritas y autorizadas puedan recibir estos mensajes para tomar una decisión. De acuerdo a la causalidad del evento, se consideraron tanto causalidades verticales como horizontales. En lo que respecta a la causalidad vertical, el repositorio tiene soporte para el almacenamiento de eventos de ejecución lo cual representa ocurrencias en tiempo de ejecución como la invocación de servicios o componentes. El ciclo de vida (tal como la terminación o el inicio de un proceso de negocio) y administración de eventos (cuando un umbral excede limites o rangos definidos) que también son características de la causalidad vertical se consideran como trabajo a futuro. En lo que respecta a la causalidad horizontal, el repositorio tiene soporte a la ocurrencia de eventos a nivel de capas. Eventos en la capa de plataforma significa la ocurrencia de actividades a este nivel, tal como la modificación de fuentes de información o la agregación de un nuevo servicios. Eventos en la capa de componentes significa la ocurrencia de actividades entre componentes, tal como la transformación de la vista de un objeto o una transición de un estado a otro. Finalmente, eventos en la capa de negocio significa la ocurrencia de actividades entre servicios, tal como el servicio para crear un nuevo usuario o eliminar uno existente.

Servicio de descubrimiento

Es un componente utilizado para descubrir las implementaciones de los servicios en tiempo de ejecución. Debido al ambiente dinámico de la cadena de suministro donde la información está cambiando constantemente, es necesario proveer la capacidad para encontrar procesos de negocio en tiempo de ejecución. Estos procesos de negocio se obtienen de proveedores de servicios candidatos que se encuentran registrados en el repositorio de servicios de BPIMS-WS. Cuando existe más de un servicio que provee una misma función, este componente selecciona un servicio basado en los requisitos del cliente. Dentro del servicio del descubrimiento, hay un constructor de consultas que construye las consultas basadas en la ontología del dominio que se enviarán al repositorio de servicios. Este módulo recupera un conjunto de servicios seleccionados de un paso anterior y crea conjuntos compatibles de servicios listos para su integración. El servicio de descubrimiento

utiliza técnicas sofisticadas para descubrir dinámicamente servicios Web y para formular consultar a los nodos UDDI. Estas técnicas sofisticadas se describen en [81].

Servicio de Integración Dinámico

Es un componente que integra los servicios compatibles. La integración de un Servicio Web se refiere a cómo puede y debe acoplarse con otro servicio Web. Por ejemplo, la tecnología de un proveedor de servicio podría ser incompatible con otro aunque las capacidades de los dos coincidan con algunos requisitos. En este sentido, el módulo actúa como un adaptador API que traduce la interfaz fuente a una interfaz común soportada por BPIMS-WS.

Invocador Dinámico

Transforma los datos de un formato a otro. Este componente puede verse como un objeto que transforma la información (es decir, las solicitudes o respuestas) que fluye entre los solicitantes o clientes y las aplicaciones del proveedor de servicios Web. El invocador dinámico crea las solicitudes de invocación para los servicios Web. Este componente recibe como parámetro un árbol de nodos DOM que representa la descripción de un servicio Web, y como parámetro de salida devuelve su correspondencia en objetos WSIF. WSIF es una tecnología para invocar servicios descritos en el lenguaje WSDL en una variedad de protocolos de red, tales como SOAP, JMS y RMI. Para la invocación de servicios Web, BPIMS-WS utiliza WSIF, la cual es una API en Java para invocar servicios Web, no importando cómo o dónde los servicios se proporcionan [82].

Analizador de Documentos WSDL

Este componente verifica la correctitud de la sintaxis de los documentos WSDL que describen los procesos de negocio a través de sus interfaces las cuales se utilizan por los socios de negocio en la cadena de suministro. Los documentos WSDL emplean esquemas XML para la especificación de elementos de información que describen información técnica de productos y operaciones de los procesos de negocio. Este componente extrae la información necesaria para crear una solicitud de invocación de un servicio Web. Esta información comprende el conjunto de operaciones, protocolos de comunicación,

parámetros de entrada y salida, así como los tipos de datos que ofrece un servicio Web. Una vez recuperada esta información, se realiza una transformación para representarla como un árbol de nodos DOM con el fin de manipularla como objetos Java. BPIMS-WS utiliza WSDL4J [83] para la transformación de nodos DOM a objetos en Java.

Servicio de Mensajería basado en WS/RM

Es el mecanismo de comunicación para la colaboración entre las partes involucradas a lo largo de la cadena de suministro. Uno de los aspectos más críticos en la cadena de suministro es mantener su continua operación tanto como sea posible. Para brindar mecanismos efectivos de comunicación a lo largo de la cadena, se consideran a las tecnologías de información como una solución ideal para resolver los problemas relacionados a la confiabilidad. Diversos factores impactan la confiabilidad en los servicios Web, entre los que se encuentran: (1) la confiabilidad entre los servidores donde las aplicaciones residen; (2) las características de rendimiento y tolerancia a fallos; (3) la capacidad para soportar accesos concurrentes; entre otros. BPIMS-WS utiliza un servicio de mensajería confiable basado en *Web Services Reliable Messaging* (WS-RM), el cual es un protocolo que proporciona una forma estándar e interoperable para garantizar la entrega de mensajes entre aplicaciones o servicios Web [84]. La especificación de WS-RM define cómo un usuario o aplicación puede conocer si sus mensajes SOAP [30] han arribado o no a sus destinos. Así también, WS-RM utiliza otra especificación: *WS-Addressing*. La especificación de *WS-Addressing* define una forma estándar para especificar la localidad de los servicios Web dentro de los mensajes SOAP [85]. Bajo este contexto, BPIMS-WS proporciona un procesamiento y una entrega segura de mensajes que permite en una forma confiable la entrega de la información entre aplicaciones distribuidas en presencia de fallos en los componentes de software, sistemas o redes de computadoras a través de WS-RM.

Además, en BPIMS-WS se proveen dos tipos de comunicaciones: síncronas las cuales son características de las arquitecturas orientadas a servicios y asíncronas las cuales son características del bus de servicios Grid. Para las comunicaciones síncronas, BPIMS-WS contiene un conjunto de clases en Java basadas en la API de SAAJ [86]. Para las comunicaciones asíncronas, BPIMS-WS contiene otro conjunto de clases en Java basadas en la API de JAX-RPC [87]. Debido a que WS-RM se sitúa en un nivel de aplicación por

encima de SOAP [30], BPIMS-WS utiliza a SOAP como protocolo base de comunicación para enviar mensajes y realizar llamados a procedimientos remotos. No obstante, la mensajería RPC pudo utilizarse para un mejor rendimiento, pero debido a las restricciones inherentes del modelo RPC donde se requiere que el cliente y el servidor estén fuertemente acoplados, el modelo RPC es incapaz de adaptarse a cambios. Por ejemplo, al usar mensajería RPC, si una interfaz de servicio cambia, las aplicaciones de lado del cliente también deben cambiarse. Además, la mensajería RPC requiere de la existencia de un *proxy* del lado del cliente para cada servicio. En contraste, un mensaje SOAP simplemente es un bloque contiguo de datos, típicamente combinado con un encabezado que contienen los detalles logísticos de transporte.

Constructor de respuestas

Recibe las respuestas de los proveedores sobre un producto solicitado. Este módulo recupera la información útil de las respuestas y construye un documento XML con información proveniente del nodo UDDI extendido y las respuestas de las invocaciones. Luego, este documento XML se presenta en formato HTML utilizando XSL. El proceso de transformación de documentos de XML al formato HTML se describe detalladamente en [88]. La respuesta contiene información de producto (nombre, código, precio, tiempo de entrega, cantidad en existencia, entre otros) e información de la empresa que ofrece el producto (nombre, descripción, e-mail, teléfono, dirección electrónica de la página Web de la empresa, por mencionar solo algunos).

Máquina de Flujos de Trabajo

Coordina los servicios Web utilizando el lenguaje de procesos de negocio BPEL4WS. Consiste en construir en tiempo de diseño y ejecución una descripción de un flujo de trabajo completamente concretizado dónde se definen dinámicamente los socios comerciales al momento de la ejecución del flujo. En la administración de la cadena de suministro, los flujos de trabajo no pueden determinarse completamente debido a que los socios comerciales no se conocen a priori y además por que ellos cambian constantemente sus papeles cliente-proveedor durante una colaboración comercial. Por esta razón, se diseñó e implementó un repositorio de procesos de negocio genéricos descritos en

BPEL4WS que describen formas complejas de situaciones recurrentes en los diversos estados de la cadena de suministro. Este repositorio contiene patrones de flujos de trabajo involucrados en escenarios de procuración oportuna en la cadena de suministro. Estos patrones describen los tipos de interacciones detrás de cada proceso de negocio, y los tipos de mensajes que se intercambian en cada interacción.

3.4. Modalidades de Interacción de BPIMS-WS

En base al énfasis de automatización, BPIMS-WS presenta 2 modalidades de interacción: como servidor *proxy* y como portal de Internet. La figura 3.10 muestra la arquitectura de BPIMS-WS mediante la modalidad de servidor *proxy* y de portal de Internet.

Mediante la modalidad de servidor *proxy*, BPIMS-WS puede interoperar con otros sistemas o agentes de software. Al igual que un *proxy*, BPIMS-WS recibe solicitudes SOAP de los clientes, redirigiendo las solicitudes a otros sistemas y enviando las respuestas SOAP a clientes o a otros destinos. Así, BPIMS-WS recibe en un puerto específico las solicitudes de servicio escritas como documentos XML. Bajo esta modalidad, BPIMS-WS permite a proveedores y socios comerciales el acceso a la información de su base de datos sin necesidad de acceder a las páginas Web en la modalidad de portal de Internet. Sin embargo, los proveedores y socios comerciales necesitan conocer la ubicación de las descripciones de los servicios Web que provee BPIMS-WS.

En la modalidad de portal de Internet, BPIMS-WS fue construido en JSP para traducir peticiones HTTP a solicitudes SOAP. Bajo esta modalidad, los clientes no necesitan tener conocimientos sobre servicios Web para acceder a las diversas funcionalidades de BPIMS-WS ya que ofrece un conjunto de interfaces gráficas para tener acceso a los diferentes servicios Web proporcionados. Mediante este conjunto de interfaces, los clientes pueden realizar consultas al nodo UDDI, invocar los servicios Web registrados y visualizar los resultados en una forma legible haciéndolo transparente a los usuarios. El portal JSP se ejecuta sobre el servidor de aplicaciones Tomcat [89]. Bajo esta modalidad, BPIMS-WS

puede verse como un sitio Web de comercio electrónico B2C (*Business-to-Consumer*) que se realiza entre empresas y particulares a través de la venta de productos y servicios.

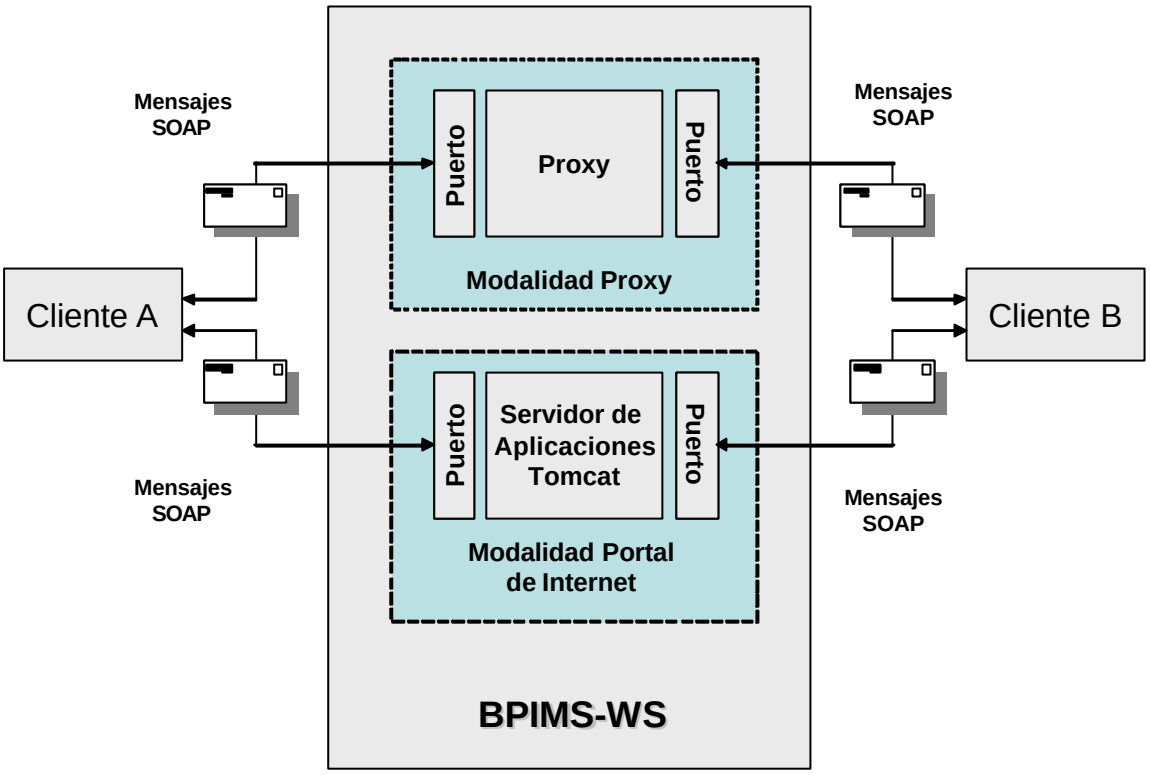


Fig. 3.10. Arquitectura de BPIMS-WS mediante la modalidad de servidor proxy y de portal de Internet

3.5. Resumen

Uno de los mayores usos de los servicios Web reside en el comercio electrónico B2B y en la administración de la cadena de suministro ya que mediante su uso es posible llevar a cabo la integración de organizaciones comerciales debido a que los servidores son independientes de la plataforma operativa, siendo necesario conocer solamente la especificación del servicio Web para realizar su invocación. En este capítulo se describe la arquitectura orientada a servicios implementada en BPIMS-WS, un sistema de intermediación basado en servicios Web capaz de integrar los procesos de negocio de

diferentes empresas en el contexto del comercio electrónico B2B y en la administración de la cadena de suministro. Además, se presenta el diseño de BPIMS-WS y los componentes situados en cada capa. También se argumenta la funcionalidad de cada capa así como de sus componentes.

Capítulo 4
Descubrimiento e Invocación Dinámica de Servicios Web en BPIMS-WS

Uno de los problemas centrales en la cadena de suministro y en el comercio electrónico B2B es cómo automatizar el proceso de descubrimiento de servicios Web, es decir, cómo localizar los servicios que cumplan ciertos requisitos comerciales con el fin de llevar a cabo el procesos de integración. En este capítulo se describe detalladamente los mecanismos de descubrimiento e invocación de servicios Web que utiliza BPIMS-WS presentando los algoritmos y técnicas utilizadas. Así también, se revisan los trabajos relacionados y se discuten sus ventajas y desventajas en comparación con éstos.

4.1. Descubrimiento Dinámico de Servicios Web con UDDI

El descubrimiento de los servicios Web se basa principalmente en encontrar los servicios que ofrecen las organizaciones para satisfacer ciertas necesidades. Para esto, es necesario utilizar un registro distribuido, conocido como UDDI [32], como mecanismo común para la publicación de las descripciones de los servicios Web. Sin embargo, UDDI presenta deficiencias como la dificultad de localizar a proveedores que ofrezcan productos y servicios que cumplan ciertas especificaciones como modelo, precio, tiempo de entrega, forma de pago y otras características técnicas. Esto se debe principalmente a que UDDI sólo está enfocado a localizar servicios que los negocios ofrecen y no a cómo estos servicios operan. Por ejemplo, suponga que se desea encontrar a una empresa que venda componentes electrónicos, en especial, microprocesadores Pentium 4 de 2.0 Ghz. UDDI

como respuesta solo devuelve una lista de todos los negocios referentes a componentes electrónicos, los servicios que ofrece dicho negocio, el punto de contacto y la especificación del servicio, solo por mencionar algunos datos. Más no se establece qué es lo que se vende, cómo se vende, en qué cantidades se vende, características del producto que se vende, formas de pago, formato de las órdenes, y todo lo referente a los procesos comerciales de la compañía. Por lo tanto UDDI, no puede responder a peticiones tan sencillas como: ¿Quién vende X producto con Y características?, ¿En qué cantidades se vende el producto X?, ¿Cuál es la forma de pago del producto X?, ¿Cuáles son las características adicionales del producto X?, etc. Además UDDI, presenta una pobre descripción de los negocios y servicios, ya que ésta está en lenguaje natural, lo cual se necesitan técnicas de extracción de información para realizar algunas inferencias de conocimiento. Esto es un punto importante, ya que no da grandes posibilidades para llevar a cabo intermediación electrónica.

Como una solución a esto, BPIMS-WS contiene un repositorio central en donde se mantienen registrados y clasificados los negocios de las empresas, los procesos de negocio que llevan a cabo, los tipos de servicios que ofrecen y los productos que venden. Para llevar a cabo esta clasificación de negocios, productos y servicios en el repositorio, se utilizaron ontologías ampliamente aceptadas como NAICS [68], UNSPSC [69] y RosettaNet [70]. Este repositorio puede considerarse como una extensión de UDDI en el sentido en que no solamente se registran los tipos de negocio, sino que además los tipos de servicios y productos que ofrecen las empresas. Sin embargo, el repositorio de BPIMS-WS no es interoperable con otros nodos registradores UDDI, como los de IBM [90,91], Microsoft [92, 93], SAP [94] y Quick UDDI [95].

Como se mencionó en el capítulo anterior, BPIMS-WS ofrece 2 modalidades de interacción. Entonces, si una empresa desea registrarse en BPIMS-WS, lo puede hacer mediante el portal de Internet o invocando el servicio Web *save_Business*. Para esta última, es necesario construir una solicitud SOAP y enviar en un documento XML los siguientes datos: *login*, *password*, *businessCode*, *name*, *description*, *personName*, *phone*, *email* y *discoveryURL*. Los datos *login* y *password* corresponden al método de autenticación que BPIMS-WS utiliza. El dato *businessCode* corresponde a la clasificación del negocio en la ontología NAICS. Mientras que la demás información corresponde a la empresa.

Una vez enviada la solicitud SOAP, BPIMS-WS registra el negocio, y devuelve un ticket (*businessKey*) con el cual, BPIMS-WS identifica de manera única al negocio dentro del repositorio.

Si la empresa desea registrarse utilizando el portal de Internet, debe seleccionar la opción "Registration" del menú principal de BPIMS-WS. Luego, seleccionar la opción "Business Registration". Aquí, BPIMS-WS despliega una interfaz gráfica mostrando un formulario para que sea llenado por la empresa. En la Fig. 4.1 se muestra una esta interfaz gráfica. La información de este formulario corresponde a la información de entrada del servicio Web *save_Business* mencionado anteriormente. Una vez llenado el formulario, BPIMS-WS construye una solicitud SOAP para invocar el servicio Web *save_Business*. BPIMS-WS despliega una interfaz gráfica mostrando el resultado de la invocación, el cual es el ticket (*businessKey*) que identifica al negocio de forma única.

De la misma manera, para que una empresa registre sus productos, es necesario construir una solicitud SOAP para invocar el servicio Web save_Products y enviar en un documento XML los siguientes datos: *login, password, businessKey y ProductsList.*

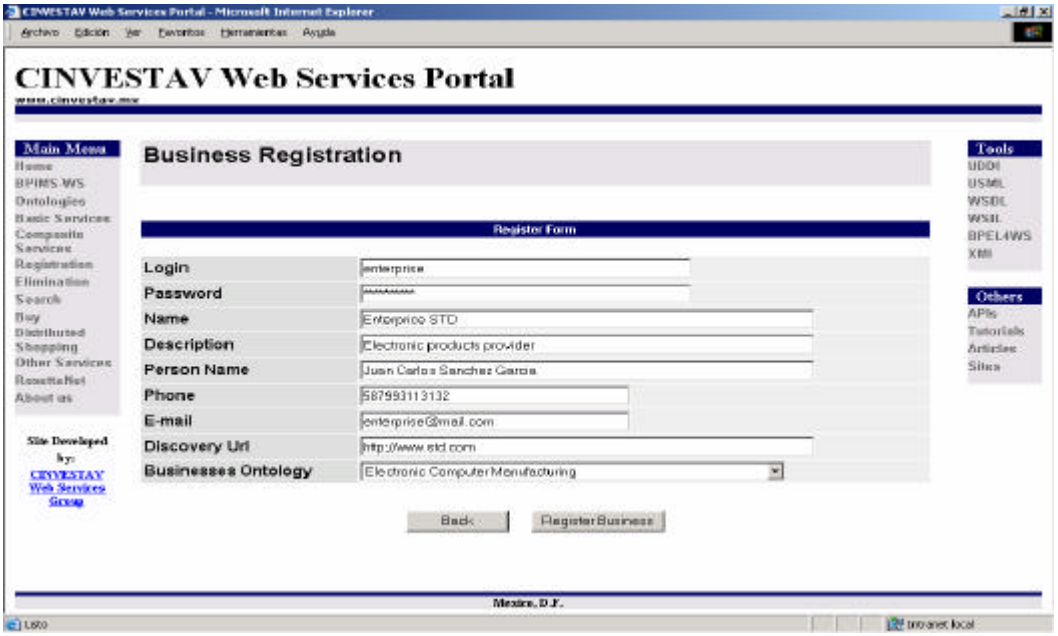


Fig. 4.1. Interfaz gráfica del registro de empresas en BPIMS-WS en la modalidad de portal de Internet

El dato *ProductsList* es un arreglo de elementos *productCode* y *ontology*, en el cual se especifica la ontología de los productos y el código asociado. Hasta este momento, BPIMS-WS soporta 2 ontologías de productos: UNSPSC que comprende una clasificación de componentes electrónicos y Amazon que abarca un catálogo de libros, revistas, películas dvd y cds de música.

Una vez invocado el servicio Web, BPIMS-WS registra la lista de productos y devuelve una lista de los productos que fueron registrados de forma exitosa. Así también, si la empresa desea registrar sus productos utilizando el portal de Internet, debe seleccionar la opción "Registration" del menú principal de BPIMS-WS. Luego, seleccionar la opción "Products Registration". Para poder registrar los productos, BPIMS-WS utiliza el mecanismo de autenticación de *login* y *password*, por lo cual antes de poder registrar algún producto es necesario especificar esta información. Si la autentificación es exitosa, BPIMS-WS despliega una interfaz gráfica mostrando una lista de las ontologías y los productos asociados a éstas que soporta. Aquí, la empresa debe agregar los productos que vende, a una lista que se encuentra en la parte izquierda de esta interfaz, tal como se muestra en la Fig. 4.2.

Fig. 4.2. Interfaz gráfica del registro de productos en BPIMS-WS en la modalidad de portal de Internet

Una vez creada la lista de productos, la empresa debe seleccionar "Save Products", entonces BPIMS-WS construye una solicitud SOAP para invocar el servicio Web save_Products. BPIMS-WS despliega una interfaz gráfica mostrando la lista de productos que registró de forma exitosa. Cabe señalar que si un producto ya se encuentra registrado, BPIMS-WS no lo registra.

Finalmente, para que una organización registre sus servicios, es necesario construir una solicitud SOAP para invocar el servicio Web save_Services y enviar en un documento XML los siguientes datos: *login*, *password*, *businessKey* y *ServicesList.* El dato *ServicesList* es un arreglo de elementos *serviceCode*, *description*, *accessPoint* y *overviewURL.* El elemento *serviceCode* corresponde a la clasificación del servicio en la taxonomía de RosettaNet. Los elementos *accessPoint* y *overviewURL* corresponden a las direcciones electrónicas, de dónde se debe invocar y dónde se encuentra la especificación del servicio, respectivamente. BPIMS-WS registra la lista de servicios, devolviendo una lista de los servicios que fueron registrados de forma exitosa. Así también, si la empresa desea registrar sus servicios utilizando el portal de Internet, debe seleccionar la opción "Registration" del menú principal de BPIMS-WS. Luego, seleccionar la opción "Services Registration". Al igual que en el registro de productos, BPIMS-WS utiliza el mecanismo de autenticación de *login* y *password*, por lo cual antes de poder registrar algún servicio es necesario especificar esta información. Si la autentificación es exitosa, BPIMS-WS despliega una interfaz gráfica mostrando una serie de formularios para que la empresa agregue los servicios que ofrece En la Fig. 4.3 se muestra una pantalla de esta interfaz. Una vez llenados los formularios, la empresa debe seleccionar "Register Services", entonces BPIMS-WS construye una solicitud SOAP para invocar el servicio Web *save_Services*. BPIMS-WS despliega una interfaz gráfica mostrando la lista de los servicios que registró de forma exitosa. De igual forma que en el registro de productos, BPIMS-WS no registra servicios repetidos.

En el proceso de registro de servicios es de suma importancia especificar el *accessPoint* y el *overviewURL* de un servicio. El *accessPoint* especifica dónde se debe invocar el servicio Web, mientras que el *overviewURL* especifica dónde se encuentra la descripción del servicio.

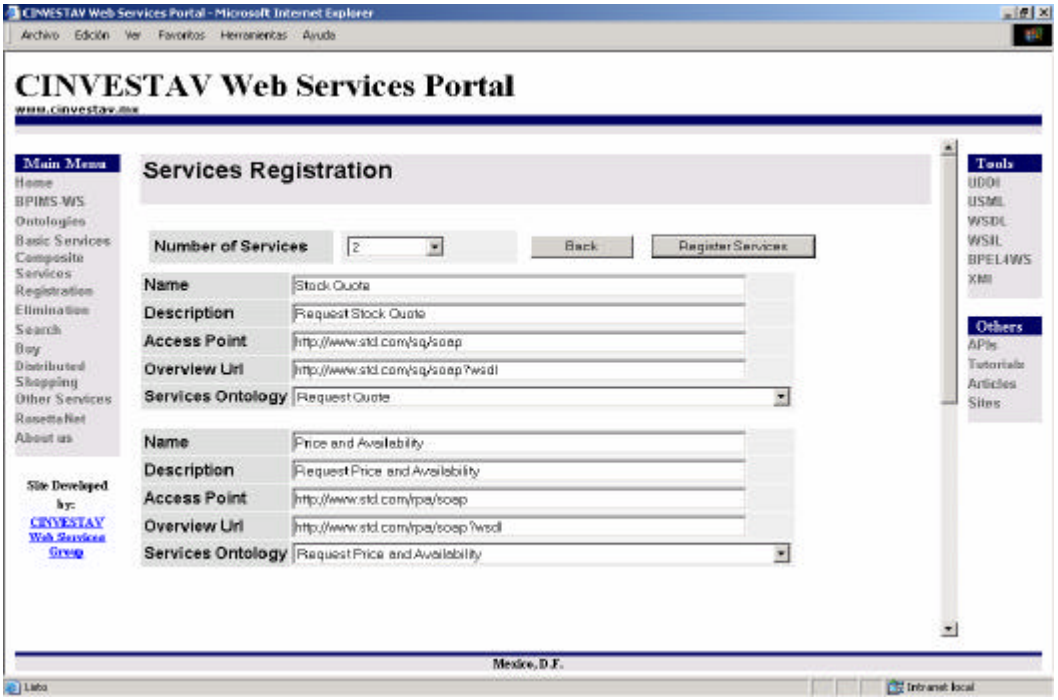


Fig. 4.3. Interfaz gráfica del registro de servicios en BPIMS-WS en la modalidad de portal de Internet

Estos elementos son necesarios para llevar a cabo el proceso de invocación dinámica de los servicios Web que ofrecen las empresas que se describe en la siguiente sección.

4.2. Invocación Dinámica de Servicios Web

Con el registro de los negocios, productos y sobre todo los tipos de servicios de las diversas empresas, BPIMS-WS es capaz de invocar sus servicios Web. Para la invocación dinámica de los servicios a partir de una solicitud, BPIMS-WS formula una consulta al nodo UDDI, buscando los elementos *acccessPoint* y *overviewURL* de aquéllos negocios que pueden satisfacer las condiciones de la solicitud. El resultado de la consulta son los elementos *accessPoint*, que indica la dirección electrónica dónde se debe invocar el servicio Web, y *overviewURL*, que indica la dirección electrónica del documento WSDL. La forma en que BPIMS-WS lleva a cabo la invocación dinámica es mediante el análisis de los documentos WSDL. Con el análisis de dichos documentos, BPIMS-WS invoca de forma automática a

los servicios Web, lo cual es una capacidad adicional que no se ofrece en UDDI. Para llevar a cabo el proceso de invocación automática, BPIMS-WS analiza los documentos WSDL de cada negocio encontrado en la consulta, extrae la información necesaria de los servicios Web de los negocios y construye las solicitudes de invocación a los servicios Web de los negocios. Estos documentos XML están empaquetados en SOAP. Una vez construidos los documentos XML empaquetados bajo SOAP, se envían los mensajes a los diferentes servicios Web por medio de Java WSDP (*Java Web Service Developer Pack*) que permite construir, probar y desplegar servicios y aplicaciones Web [96]. Después de enviados los mensajes, se reciben las respuestas de los diferentes servicios Web que fueron invocados. BPIMS-WS analiza estas respuestas, extrae la información que le es útil y construye un nuevo documento XML el cual será la respuesta a la solicitud hecha a BPIMS-WS por una organización a través de un servicio Web provisto, como por ejemplo: *get_PriceandDeliveryTime*, *get_Quantity*, *get_ProviderQuantity*, por mencionar sólo algunos. Luego, BPIMS-WS envía entonces la respuesta al URL donde le fue hecha la solicitud. Para esto, se empaqueta el documento XML bajo SOAP y envía dicho documento.

4.3. Búsqueda y Localización de Servicios Web utilizando WSIL

La búsqueda, localización e invocación de un servicio Web consiste en realizar búsquedas de las descripciones de los servicios Web que ofrecen diversas compañías, en diferentes nodos UDDI, y recuperar el documento WSDL donde se describe el servicio para poder efectuar su invocación. Sin embargo, no todos los servicios Web están publicados en nodos UDDI. Para esto, se necesitan mecanismos para publicar y conocerlos, con el fin de tener acceso a ellos. Para solucionar esto, BPIMS-WS ofrece un explorador de documentos WSIL con el fin de localizar de forma más rápida la especificación de un servicio Web y llevar a cabo su invocación. WSIL *(Web Services Inspection Language)* [97] ayuda en gran medida a realizar referencias a descripciones de servicios Web que no se encuentran publicadas en un nodo UDDI. Además, provee mecanismos para hacer referencias a diferentes tipos de descripciones de servicios Web. Mediante el siguiente caso de estudio se

muestra cómo BPIMS-WS facilita el descubrimiento de los servicios Web que ofrece una compañía, que se dedica a la venta de libros *on-line*.

Suponga el siguiente escenario:

1. Suponga que se tiene una compañía, la cual vende de manera *on-line,* libros mediante un portal de Internet. Además, esta compañía provee una serie de servicios Web que todavía no se encuentran registrados en algún nodo UDDI público.
2. Los potenciales clientes de la compañía, desean realizar pedidos, mediante el uso de los servicios Web que ofrece la compañía, con el fin de llevar a cabo el proceso de integración.
3. La compañía describió sus servicios Web en un documento WSIL que se encuentra publicado en el sitio Web de la compañía.

En este escenario, ¿Cómo los clientes potenciales pueden localizar e invocar los servicios Web de la compañía, para poder realizar pedidos de libros?

Para poder contestar la pregunta descrita en el escenario, BPIMS-WS ofrece una solución a los clientes. En BPIMS-WS hay una opción en el menú principal llamada "WSIL". En esta opción, se despliega una interfaz gráfica en donde los clientes pueden realizar la búsqueda y localización de los servicios Web de la compañía, con solo escribir el URL donde se encuentra el documento WSIL de la compañía. Una vez establecido el URL donde se encuentra el documento WSIL de la compañía, BPIMS-WS analiza el documento y despliega mediante una interfaz gráfica, las descripciones de los servicios Web encontrados. En la Fig. 4.4 se muestra la interfaz gráfica del resultado del análisis del documento WSIL. En esta interfaz gráfica, las descripciones de los servicios Web encontradas aparecen en forma de hipervínculos. Aquí, los clientes pueden seleccionar una descripción, y al seleccionar el hipervínculo correspondiente, BPIMS-WS analiza la descripción del servicio Web y crea una interfaz gráfica para efectuar su invocación. Al analizar la descripción del servicio Web, BPIMS-WS despliega las operaciones provistas por el servicio Web seleccionado.

Entonces, los clientes pueden seleccionar una operación del servicio Web. Posteriormente de que el cliente seleccionó una operación, BPIMS-WS despliega los parámetros de entrada que correspondan a la operación seleccionada.

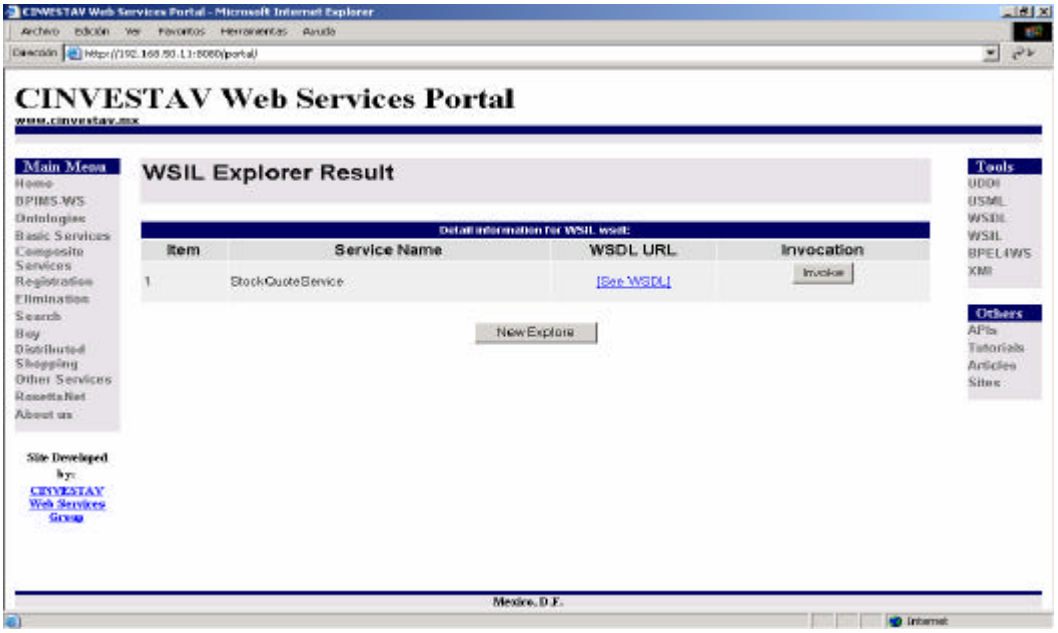


Fig. 4.4 Interfaz gráfica del resultado del análisis de un documento WSIL

El cliente debe llenar con información estos parámetros para poder efectuar la invocación del servicio Web. Después de definir la información de los parámetros, BPIMS-WS realiza la invocación. En la Fig. 4.5, se muestra la interfaz gráfica donde es posible llevar a cabo la invocación de un servicio Web que proporciona el precio de un libro dado su ISBN. Para poder llevar a cabo la invocación, BPIMS-WS utiliza WSIF. WSIF (*Web Services Invocation Framework*) es una API en Java para llevar a cabo invocaciones de servicios Web, no importando donde los servicios Web sean provistos [82]. Con WSIF, es posible trabajar con representaciones abstractas de los servicios Web descritos en WSDL e invocarlos mediante las APIs de SOAP, las cuales constituyen el modelo usual de invocación [82]. Finalmente, después de efectuarse la invocación del servicio Web, BPIMS-WS muestra los resultados de la invocación realizada, mediante una interfaz gráfica. En la Fig. 4.5 se muestra la interfaz gráfica donde se muestran los resultados de la invocación al servicio Web que proporciona el precio de un libro a partir de su ISBN.

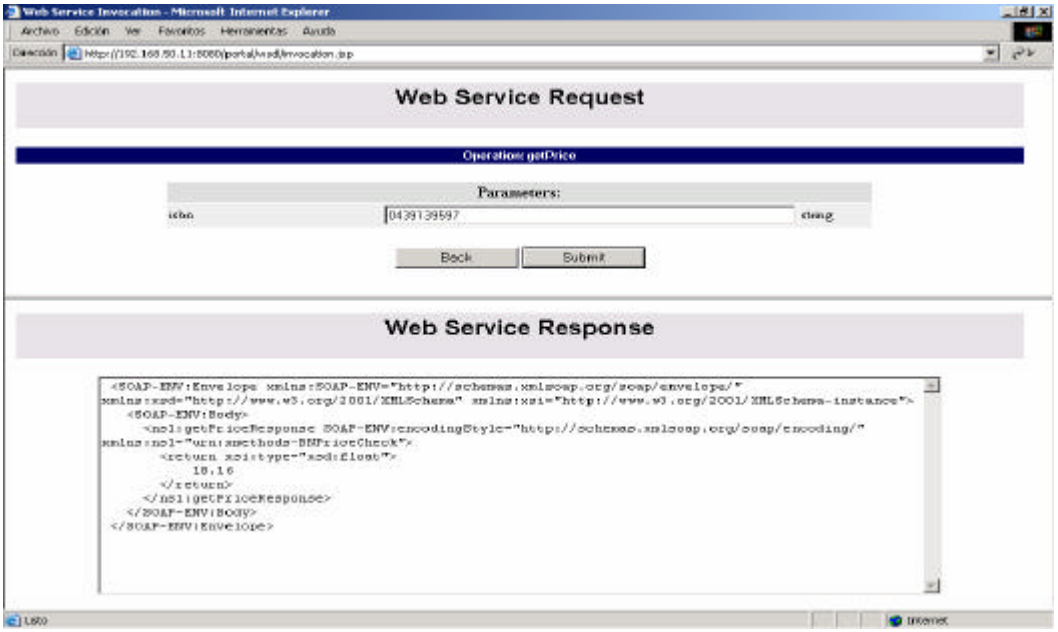


Fig. 4.5 Interfaz gráfica para la invocación de un servicio Web

En la búsqueda, localización e invocación de servicios Web, BPIMS-WS es de gran ayuda a los clientes de la compañía, ya que ésta no tiene publicados sus servicios Web en nodos UDDI, donde habitualmente los clientes buscarían los servicios Web.

4.4. Descubrimiento Dinámico de Servicios Web en nodos UDDI mediante USML

Para la publicación de servicios Web, organizaciones, consorcios y grandes empresas desarrollaron UDDI (*Universal Description, Discovery and Integration*), un servicio de búsqueda y recuperación de información sobre servicios Web. Desafortunadamente, las consultas actuales en nodos UDDI se enfocan en un solo criterio de búsqueda como el nombre de negocio, situación comercial, categorías comerciales o tipos de servicio. Para resolver esto, BPIMS-WS permite realizar consultas en diversos nodos UDDI utilizando múltiples criterios de búsqueda para el descubrimiento dinámico de servicios Web mediante la utilización de USML (*UDDI Search Markup Language*) [98].

Mediante el siguiente caso de estudio se muestra cómo BPIMS-WS facilita el descubrimiento dinámico de los servicios Web que ofrecen las empresas registradas en nodos UDDI.

A continuación se describe el siguiente escenario:

1. Suponga que una compañía desea comprar componentes electrónicos, sin embargo, desconoce quienes podrían ser sus potenciales proveedores.
2. Suponga que una serie de proveedores de componentes electrónicos registraron sus servicios Web de manera dispersa en diferentes nodos UDDI.

Bajo estas condiciones, ¿cómo puede la compañía buscar y encontrar los servicios Web de todos los potenciales proveedores dado que éstos tienen registrados sus servicios en forma dispersa en diferentes nodos UDDI?

BPIMS-WS ofrece una solución a esto. En BPIMS-WS hay una opción en el menú principal llamada USML. En esta opción, la compañía puede realizar la búsqueda y localización de los servicios Web de los potenciales proveedores mediante algunos criterios de búsqueda como son: nombre del negocio, nombre del servicio y tipo de servicio. Para ello, BPIMS-WS ofrece 2 modalidades de consulta: 1) consulta en todos los nodos UDDI, y 2) consulta en un nodo UDDI particular.

En la primera modalidad, la compañía no necesita saber cuáles son los nodos UDDI y los respectivos URLs de dichos nodos. En este sentido, la compañía delega a BPIMS-WS la tarea de buscar los servicios Web de los proveedores en los diferentes nodos UDDI. Para ello, BPIMS-WS primero realiza una búsqueda de los nodos UDDI existentes en su archivo de configuración. En este archivo se encuentran registrados los nombres y URLs de los diversos nodos UDDI en los que se pueden llevar a cabo las consultas. Mediante este archivo se permite almacenar un gran número de URLs asociados con diversos nodos UDDI. A este archivo, pueden agregarse nuevos nodos UDDI sin necesidad de modificar la programación. La Fig. 4.6 muestra el archivo de configuración que utiliza BPIMS-WS para conocer los diversos nodos UDDI donde se realizarán las consultas. El archivo de configuración de esta figura es un documento XML que contiene básicamente el nombre del nodo UDDI (elemento *Name*) y la dirección electrónica del servicio de consulta del nodo UDDI (elemento *URL*).

```
<?xml version="1.0"?>
<!DOCTYPE Registries SYSTEM "config.dtd">
<Registries>
   <Registry>
      <Name>Microsoft UDDI</Name>
      <URL> http://uddi.microsoft.com/inquire </URL>
   </Registry>
   <Registry>
      <Name>Private UDDI</Name>
      <URL> http://127.0.0.1/servlet/uddi </URL>
   </Registry>
   <Registry>
      <Name>IBM Public UDDI</Name>
      <URL>
   http://www-w3.ibm.com/services/uddi/testregistry/inquiryapi
   </URL>
   </Registry>
</Registries>
```

Fig. 4.6 Archivo de configuración para la búsqueda de servicios Web en USML

Una vez recuperados los nombres de los nodos UDDI y sus respectivos URLs, y establecidos los criterios de búsqueda de la compañía, BPIMS-WS construye un documento USML para cada nodo UDDI en donde se va a realizar una consulta. Después, empaqueta en SOAP cada documento USML. Luego, BPIMS-WS envía los documentos a los diferentes nodos UDDI por medio de SAAJ (*SOAP with Attachments API for Java*) [86]. Cabe señalar, que el envío de los documentos USML a los diversos nodos UDDI, se realiza en forma paralela, para esto, BPIMS-WS crea un *thread* o hilo de ejecución por cada consulta en un nodo UDDI. Ya realizadas las consultas, es decir, enviados los documentos USML, BPIMS-WS obtiene las respuestas de las consultas de cada nodo UDDI. Después, analiza estas respuestas y extrae la información que le es útil y construye un nuevo documento USML, el cual será la respuesta a la solicitud hecha por la compañía. Una vez creada la respuesta, BPIMS-WS despliega los resultados en una interfaz gráfica y le asocia un hipervínculo a cada negocio, servicio o tipo de servicio encontrado. Todo esto, con el fin de obtener información adicional en el nodo UDDI en donde se encuentra registrado el negocio, servicio o tipo de servicio. Aquí, la compañía ya conoce los servicios Web que ofrecen los potenciales proveedores registrados en los diferentes nodos UDDI.

En lo que respecta a la segunda modalidad, ésta se puede ver como un caso particular de la primera. En esta modalidad, la compañía debe conocer los diversos nodos UDDI existentes.

Para esto, BPIMS-WS realiza una búsqueda en el archivo de configuración con el fin de obtener nombres y URLs de los diversos nodos UDDI existentes. Una vez recuperados los nombres y URLs de los nodos UDDI, BPIMS-WS despliega los resultados en una interfaz gráfica, para que la compañía seleccione el nodo UDDI donde desea realizar la consulta. Una interfaz gráfica de la selección del nodo UDDI se muestra en la Fig. 4.7.

Fig. 4.7 Interfaz gráfica para la selección del nodo UDDI de consultas en USML

Una vez seleccionado el nodo UDDI, la compañía debe establecer los criterios de búsqueda mediante una interfaz gráfica. Al establecer un criterio de búsqueda, BPIMS-WS despliega información pertinente a ese criterio, es decir, si la compañía selecciona el criterio de búsqueda por negocio, se despliega información como: nombre del negocio a buscar, identificador del nombre de negocio a buscar, categoría del nombre de negocio a buscar, entre otros. La información que BPIMS-WS despliega varía de acuerdo al criterio de búsqueda que se está seleccionando. Una interfaz gráfica del establecimiento de los criterios de búsqueda se observa en la Fig. 4.8. Luego de establecer los criterios de búsqueda, BPIMS-WS muestra una interfaz gráfica donde la compañía debe seleccionar el operador lógico a utilizar en la consulta (OR o AND).

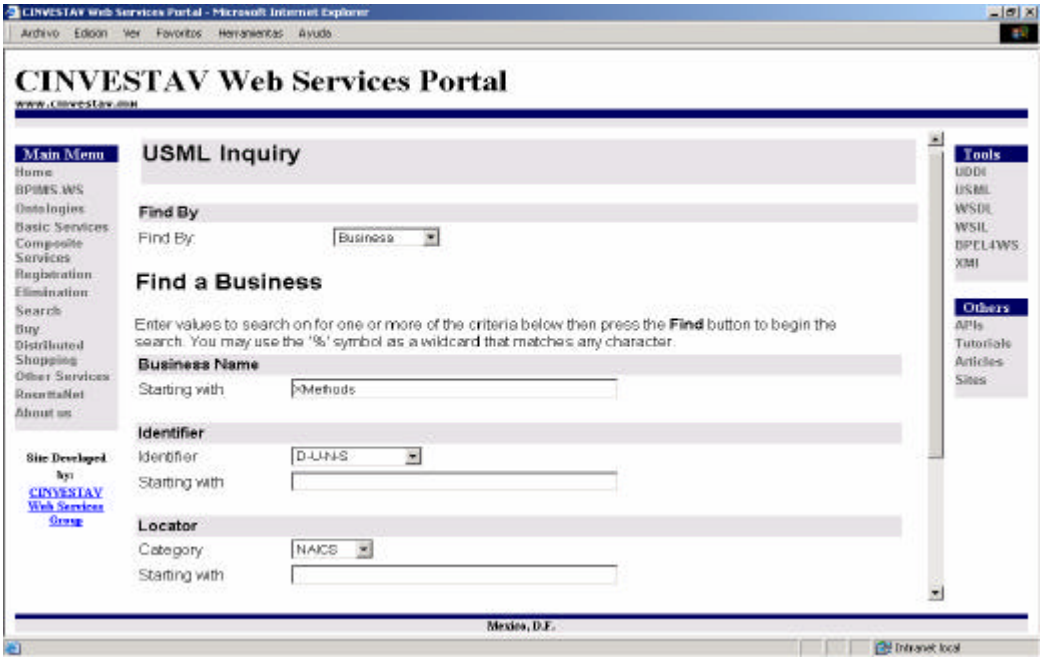


Fig. 4.8 Interfaz gráfica de la selección del tipo de búsqueda en USML

Posteriormente, BPIMS-WS construye el documento USML que corresponde a la solicitud de consulta de la compañía y lo muestra en otra interfaz gráfica. Después, BPIMS-WS empaqueta en SOAP el documento USML y lo envía al nodo UDDI. Finalmente, BPIMS-WS recibe la respuesta y despliega el resultado en una interfaz gráfica. En la Fig. 4.9 se muestra el resultado de la búsqueda de negocios bajo el criterio "XMethods" al nodo UDDI de Microsoft. De igual forma que en la primera modalidad, BPIMS-WS asocia un hipervínculo a cada negocio, servicio o tipo de servicio encontrado. Todo esto, con el fin de obtener información adicional en el nodo UDDI en donde se encuentra registrado el negocio, servicio o tipo de servicio.

En la búsqueda y localización de servicios Web, BPIMS-WS es de gran ayuda a la compañía, ya que la compañía desconoce el nombre, la ubicación geográfica y electrónica, los servicios y los tipos de servicios que ofrecen los potenciales proveedores.

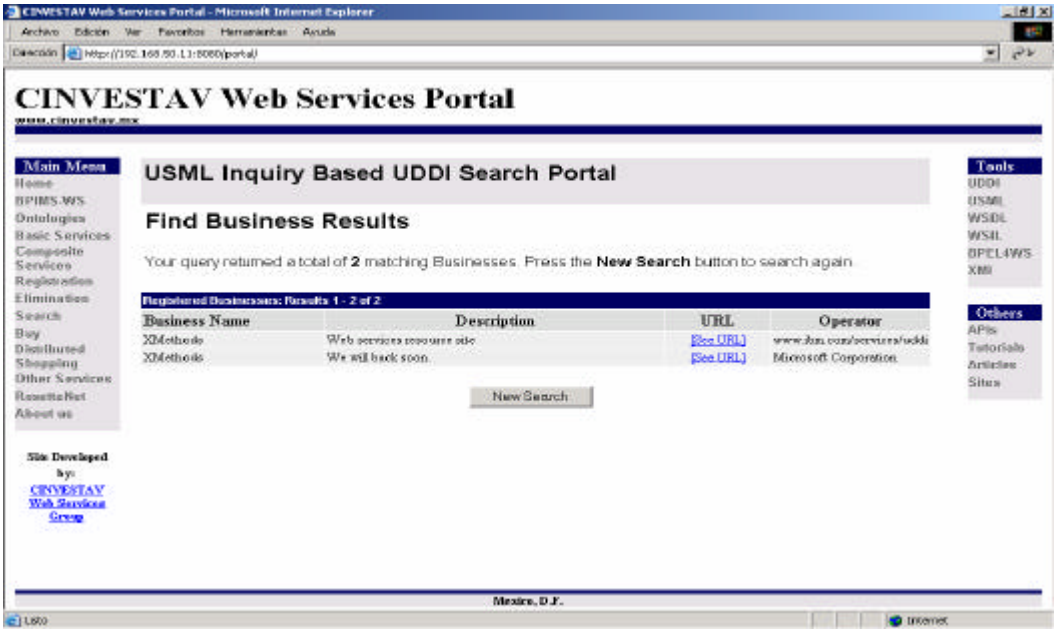


Fig. 4.9 Interfaz gráfica que muestra los resultados de una consulta en USML

4.5. Trabajos Relacionados con BPIMS-WS en el descubrimiento Dinámico de Servicios Web

El problema de descubrimiento dinámico de servicios Web en tiempo de ejecución ha motivado el desarrollo de diversos trabajos. En estos trabajos se aborda este problema mediante el uso de diversas técnicas y mecanismos. McIlraith et. al. [99] propone un lenguaje semántico de servicios Web basado en DAML el cual provee una API declarativa para la captura de datos y meta-datos asociados con un servicio junto con las especificaciones de sus propiedades y capacidades, las interfaces de su ejecución, y los prerrequisitos y consecuencias de su uso. Este lenguaje permite la definición de tipos especiales de servicios en ontologías específicas que contienen información especial de productos. Las ontologías se utilizan para facilitar la compartición, re-uso, composición y mapeo de la información. La idea de este lenguaje es crear una base de conocimientos distribuidas con el fin de que agentes poblen sus bases de conocimientos locales de tal forma que puedan razonar sobre ellas para llevar a cabo el descubrimiento, ejecución, composición e interoperabilidad de servicios Web. Además, el lenguaje tiene soporte para

el uso de tecnología de agentes a través de un lenguaje de primer orden del cálculo de situaciones y de una versión extendida del lenguaje de programación de agentes ConGolog. Mediante esta tecnología de agentes, se localizan, ejecutan y orquestan los servicios Web candidatos en base a las restricciones impuestas por el usuario. En comparación, BPIMS-WS no provee mecanismos para el razonamiento automático debido a que no tiene soporte para tecnologías de agentes debido a las limitaciones inherentes: el lenguaje de comunicación y en el formato de mensajes que se intercambian. En la comunidad de agentes, existen diversos lenguajes de comunicación por lo que hasta el momento no hay un estándar para la comunicación entre agentes. Para poder llevar a cabo una comunicación entre agentes, es necesario que ambos adopten un lenguaje específico para poder realizar sus funcionalidades. Sin esto, cada agente tendría que tener un soporte para poder llevar a cabo la comunicación multi-lenguaje. También, en lo que respecta al formato de los mensajes, los agentes deben adoptar un formato de mensajes para poder interpretar las solicitudes de servicio. Por el contrario, BPIMS-WS supera estas limitaciones debido a la tecnología estándar inherente de los servicios Web. Para la comunicación entre servicios, BPIMS-WS utiliza a SOAP el cual es un servicio de mensajería estándar que permite el intercambio de información sobre diferentes protocolos de comunicación como HTTP, SMTP y FTP lo cuales también son estándares de Internet. Además, BPIMS-WS utiliza a XML para poder describir y representar la información que se intercambia entre los servicios Web. XML es un lenguaje estándar, el cual es la base de la tecnología de servicios Web.

Dumas et. al. [100] propone una clasificación y caracterización de tipos de servicios (tangibles e intangibles) independientemente del dominio. Para cada característica de un servicio se identifica el rango de sus posibles valores en diferentes configuraciones y, cuando es aplicable, se consideran enfoques alternativos para representar estos valores. La idea es combinar estos enfoques individuales y mapearlos en una notación unificada, con el fin de diseñar lenguajes de descripción de servicios con el fin de publicarlos y descubrirlos en configuraciones de aplicaciones específicas. La propuesta de Dumas se enfoca en el diseño de lenguajes que permitan categorizar servicios y que ofrezcan mecanismos o servicios de consultas a estos catálogos, además de establecer una base para el razonamiento acerca de estos servicios. En particular, dadas dos descripciones de servicios

conocer si pueden orquestarse y derivar algunas propiedades de la orquestación. BPIMS-WS utiliza lenguajes estándares de Internet para la descripción y composición de servicios. WSDL permite describir las capacidades que ofrece un servicio Web y BPEL4WS permite orquestar estas capacidades.

Paolucci et. al. [101] propone un algoritmo para el descubrimiento semántico que permite descubrir las capacidades que ofrecen los servicios Web descritos en DAML-S. El algoritmo utiliza ontologías basadas en DAML, las cuales permiten definir funciones de ranking con el fin de distinguir diferentes niveles de descubrimiento. Estos niveles de descubrimiento se determinan en base a la distancia mínima entre conceptos en el árbol taxonómico de las ontologías. Aquí se diferencian cuatro niveles de descubrimiento: exacto el cual indica que hay un nodo en el árbol con la misma funcionalidad requerida, *plug in* el cual indica que hay un nodo en el árbol que provee una funcionalidad similar (no igual) que puede sustituir a la funcionalidad que se está buscando, subsume el cual indica que hay un nodo en el árbol el cual solo representa una parcialidad de la funcionalidad requerida y; fallo el cual indica que no hay ningún nodo en el árbol que represente la funcionalidad requerida. Para el descubrimiento semántico, primero se recibe una solicitud de servicio y el componente llamado *matching engine* selecciona los servicios candidatos de una base de datos donde se encuentran publicados. Luego, un componente llamado razonador DAML+OIL calcula el nivel de correspondencia. Para esto, el razonador utiliza una base de datos que contiene las ontologías que se utiliza para el proceso de descubrimiento. Sin embargo, una fuerte limitación de esta propuesta es que el descubrimiento de servicios sólo se realiza verificando la compatibilidad en los tipos de los parámetros de entrada y salida de un servicio publicado con respecto a una solicitud de servicio, es decir, solo se verifica la coincidencia de los mensajes de entrada y salida, por lo que no considera operaciones o conjunto de operaciones, los cuales constituyen la funcionalidad de los servicios Web.

Una extensión a este trabajo lo propone Sivashanmugam et. al [102]. Sivashanmugam propone un algoritmo para el descubrimiento semántico de servicios el cual no solo considera parámetros de entrada y salida, sino también especificaciones funcionales de operaciones y efectos. Para esto, se combinan un número de términos predefinidos para expresar pre y post-condiciones. Para la descripción de estos términos, se extiende a WSDL y UDDI con capacidades de descripción semántica. Mediante este tipo de descripciones es

posible distinguir servicios muy similares con efectos distintos, es decir, servicios que presentan parámetros de entrada y salida similares pero con diferente funcionalidad. En comparación con los trabajos de Paolucci y Sivashanmugam, BPIMS-WS no utiliza lenguajes semánticos para la descripción de servicios Web debido a que no son estándares y no tienen soporte por la industria. A pesar de que DAML-S permite descripciones semánticas para la descripción de las capacidades de los servicios Web, presenta una fuerte limitación debido a que no provee mecanismos para describir y representar niveles y protocolos de comunicación entre servicios Web, lo cual es un factor importante en el proceso de integración, es por esto que BPIMS-WS utiliza a WSDL para la descripción de servicios Web el cual es una estándar para la industria. El enfoque que utiliza BPIMS-WS para el descubrimiento de servicios Web no es extender a WSDL dotándolo de capacidades semánticas, como se caracterizan los anteriores trabajos. En vez de esto, BPIMS-WS extiende el esquema del repositorio donde los servicios Web se publican, que en este caso es UDDI. En este sentido, hay trabajos reportados que agregan mecanismos a la descripción de UDDI. Akkiraju et. al. [103] propone una extensión al servicio de consulta de UDDI que permite a los solicitantes de servicios especificar las capacidades requeridas de un servicio. Así también, refuerza el descubrimiento de servicios en nodos UDDI realizando un *matching* semántico y una composición automática de servicios Web a través de algoritmos de planeación. Esto se debe a que los estándares actuales no son aptos para resolver este problema. SOAP y WDSL, por ejemplo, se preocupan más en describir mecanismos de transporte e interfaces; UDDI describe entidades por atributos y servicios que provee. Sin embargo, esta descripción es para humanos. Por lo que se requiere un nivel semántico porque el solicitante de un servicio no sabe a priori que servicios están disponibles, y además un proveedor y un solicitante de un servicio pueden tener distinta perspectiva sobre éste. Para el descubrimiento de servicios, un módulo de búsqueda recibe las solicitudes y ejecuta las consultas pertinentes. Si se especifica la categoría correspondiente a la información de servicios soportados por UDDI, entonces se emplea un algoritmo de 2 pasos para el descubrimiento de los servicios. Primero, se emplea un filtro para identificar la categoría de los servicios especificada en una solicitud. El filtro realiza una búsqueda para recuperar a aquellos servicios que caen en el conjunto de taxonomías soportadas por UDDI. Luego, los servicios filtrados son los parámetros de entrada de la

máquina que realiza la correspondencia semántica. Esta máquina ejecuta un algoritmo de encadenamiento hacia atrás para finalmente, encontrar los servicios que satisfacen a la solicitud. En comparación con el servicio de descubrimiento de BPIMS-WS, no se utilizan algoritmos de planeación para realizar una correspondencia semántica que garantize que la respuesta de una solicitud sea entregada. En BPIMS-WS, se tiene un repositorio de ontologías de procesos de negocio que garantiza la respuesta a una solicitud en una forma rápida en vez del uso de algoritmos de planeación. Sin embargo, debido al dinamismo del comercio electrónico es necesario mantener actualizada lo mejor posible la información de las ontologías de los procesos de negocios.

Doshi et al. [104] propone un motor semántico parametrizado para realizar búsquedas de servicios en nodos UDDI añadiendo anuncios (*advertises*). La máquina está esencialmente diseñada para funcionar en el contexto de la integración de procesos de negocio. La máquina habilita el análisis de descripciones de servicios declarativos representados en lenguajes semánticos como DAML-S y, combina criterios de emparejamiento con un algoritmo intuitivo para producir los resultados. El descubrimiento semántico utiliza la noción de indexación de conceptos el cual permite realizar razonamiento sobre las ontologías que especifican los conceptos del dominio. La máquina también ejecuta búsquedas de servicios en nodos UDDI. Para ello, se extiende la especificación de UDDI utilizando apuntadores a la información semántica que utiliza la máquina durante el proceso de descubrimiento de servicios. La extensión a UDDI consiste en una clasificación basada en servicios modificadores y no modificados del estado de UDDI. Los servicios modificadores son aquellos que realizan operaciones de creación, actualización, eliminación y suscripción para cambiar el estado de un nodo UDDI. Los servicios no modificadores son aquellos que realizan operaciones de verificación, búsqueda y publicación pero que no cambian el estado de un nodo UDDI. Al igual que la propuesta de Doshi, BPIMS-WS extiende a UDDI en cuanto a la funcionalidad de los servicios. Sin embargo, los enfoques son distintos ya que BPIMS-WS utiliza una clasificación en base a las funcionalidades de los servicios desde el punto de vista de los usuarios, es decir, se extiende a UDDI para describir cómo los usuarios y bajo qué contexto pueden utilizar la funcionalidad de los servicios publicados. En cambio Doshi, extiende a UDDI para

identificar a aquellos servicios que tienen o no cierta repercusión en el estado de UDDI en base a la funcionalidad que proveen.

ShaikhAli [105] propone un enfoque distinto para extender a UDDI. Este enfoque provee el soporte de páginas azules para almacenar las propiedades definidas por los usuarios asociadas con un servicio y, para habilitar el descubrimiento de servicios basadas en estas propiedades. Esta extensión consiste en poder registrar servicios por periodos de tiempo limitado con el fin de resolver el problema de inconsistencia o pérdida de referencias de los servicios publicados. Así también, provee el soporte para la búsqueda basada en otros atributos de servicios basado en rangos numéricos o lógicos. Las extensiones se basan en cuatro tipos de información: (1) de los negocios; (2) de los servicios; (3) de las especificaciones de los servicios; y (4) de la integración de los servicios. Estas extensiones tienen implicaciones en los métodos *saveService*, *findService* y *getServiceDetails* del servicio de consulta del nodo UDDI. En el método *saveService* se debe a que se permiten almacenar servicios bajo un cierto periodo de tiempo, en el método *findService* se permiten la búsqueda de servicio por diversos atributos. Finalmente, en el método *getServiceDetails* se permite la búsqueda de las especificaciones de los servicios. En comparación con BPIMS-WS, se extiende a UDDI proveyéndolo de capacidades para la búsqueda y registro de productos. En este sentido, se agregaron dos métodos adicionales para esta capacidades: *saveProducts* y *findProducts*. El método *saveProducts* permite a las empresas poder registrar los productos que soportan en base a un conjunto de ontologías de productos que provee el repositorio UDDI de BPIMS-WS. El método *findProducts* permite la búsqueda y localización de los productos previamente registrados por las empresas. Bajo este enfoque, BPIMS-WS permite la localización rápida de productos mediante los servicios Web publicados que representan procesos de negocio los cuales siguen un comportamiento establecido por las ontologías de procesos de negocio. Es por esto que BPIMS-WS extiende a UDDI incorporando una relación simbiótica entre los productos ofrecidos por las empresas y los procesos de negocio que soportan. Esta relación simbiótica define qué procesos de negocio soporta una empresa y qué tipo de productos se proveen en esos procesos de negocio. Por ejemplo, una empresa que provee componentes electrónicos puede definir que provee la solicitud de precio, verificación de disponibilidad y solicitud de compra para ese tipo de productos.

Youzhi [106] propone un marco de trabajo de servicios Web basado en Internet para que empresas virtuales puedan descubrir e integrar los procesos de negocios ofrecidos por otros proveedores. En este marco de trabajo se utilizan ontologías para proveer un formato uniforme para el intercambio de mensajes y un sistema multi-agente para la comunicación dinámica. La arquitectura de la plataforma se basa en tres tipos de agentes: (1) de solicitud; (2) de servicio; (3) de ontología. El agente de solicitud recibe y analiza las solicitudes provenientes de los clientes para transformarlas en formato estándar. Este agente interactúa con el agente de servicio para encontrar el servicio que el cliente solicita. A diferencia del agente de solicitud, el cual está en el lado del cliente, el agente de servicio se encuentra en el lado de la aplicación. Este agente valida la solicitud proveniente del agente de solicitud, para verificar si pertenece a un cliente autorizado. En caso afirmativo, construye la consulta al nodo UDDI y selecciona el servicio Web apropiado y ayuda al agente de solicitud a integrarse con el. El agente de ontología es quien contesta la solicitud del agente de servicio y ejecuta las consultas al nodo UDDI. Este es el trabajo más relacionado con BPIMS-WS, sin embargo, el marco de trabajo se encuentra en su fase conceptual ya que todavía no se tiene una implementación reportada que refuerce esta idea. Por el contrario, BPIMS-WS es un sistema ya implementado el cual presenta características similares a la propuesta de Youzhi. Los agentes de solicitud, servicio y de ontología tienen una correspondencia natural a los componentes: (1) Analizador de mensajes SOAP y, (2) Servicio de descubrimiento. En BPIMS-WS, el analizador de mensajes SOAP es quien analiza la solicitud proveniente de los clientes y determina mediante la estructura y el contenido de los mensajes, el tipo de servicio que se está solicitando. Esto es similar a la funcionalidad del agente de solicitud en la propuesta de Youzhi. El agente de servicio presenta una funcionalidad similar al servicio de descubrimiento de BPIMS-WS. El servicio de descubrimiento permite localizar las implementaciones de los servicios en tiempo de ejecución. A diferencia del agente de ontología, el servicio de descubrimiento construye y realiza las consultas basadas en la ontología del dominio que se envían al repositorio de servicios de BPIMS-WS recuperando un conjunto de servicios seleccionados de un paso anterior y creando conjuntos compatibles de servicios listos para su integración. Finalmente, BPIMS-WS presenta componentes que permiten la integración de los servicios Web, los cuales no están claramente definidos en la propuesta de Youzhi. Ejemplo de estos

componentes son: (1) Servicio de Integración Dinámico e, (2) Invocador Dinámico. En BPIMS-WS estos componentes determinan cómo puede y debe acoplarse un servicio Web con otro. Por ejemplo, la tecnología de un proveedor de servicio podría ser incompatible con otro aunque las capacidades de los dos coincidan con algunos requisitos. En este sentido es necesario realizar algunas transformaciones de la información que fluye entre los solicitantes o clientes y las aplicaciones del proveedor de servicios Web.

4.6. Trabajos Relacionados con BPIMS-WS en la invocación dinámica de servicios Web

Actualmente, los sistemas de información empresariales son aplicaciones basadas en servicios Web desarrolladas bajo arquitecturas orientadas a servicios. Uno de los aspectos importantes de los sistemas de información empresariales es mantener su funcionamiento continuo tan largo como sea posible. Para lograr esto, se deben proveer mecanismos para la invocación de servicios Web. Debido al dinamismo del comercio electrónico, la especificación de un servicio puede cambiar por lo que es necesario emplear técnicas en tiempo de ejecución para la ejecución e invocación de servicios Web. En esta área se han desarrollado diversos trabajos para afrontar este problema. VanderMeer et. al [107] propone una sistema llamado FUSION el cual es un portal de servicios que expone un conjunto de servicios Web a los usuarios, permite la especificación y ejecución de tareas complejas definidas a través de servicios Web individuales. Para la ejecución de estas tareas se utilizan algoritmos de planeación que transforman las metas de los usuarios en planes óptimos de ejecución. Una vez determinado un plan de ejecución, se realiza su ejecución y se verifican los resultados contra el criterio de satisfacción de los usuarios y, en caso de una no satisfacción se inician procedimientos de recuperación. Para la descripción de un plan se propone un lenguaje para la invocación de servicios Web llamado WSESL (*Web Services Execution Specification Language*) el cual solo se encuentra en su fase conceptual. El lenguaje, según los autores, permite una descripción formal de la noción de correctitud y optimalidad con respecto a los planes de ejecución. La arquitectura de FUSION consta de los siguientes subsistemas: (1) Receptor de solicitudes del usuario; (2) Generador dinámico del plan; (3) Ejecutador del plan; (4) Recuperador de fallos; (5)

Generador de respuestas. En comparación, BPIMS-WS provee una máquina de flujos de trabajo la cual coordina las invocaciones de los servicios Web participantes utilizando el lenguaje de procesos de negocio BPEL4WS. En este sentido, BPIMS-WS utiliza un lenguaje estándar en vez de un lenguaje propietario como WSESL, el cual incluso no se tiene reportada alguna máquina de ejecución desarrollada que provea el soporte para la especificación de WSESL. Además, una máquina BPEL provee características similares a la arquitectura de FUSION. En la especificación de BPEL se tienen los elementos *recieve* los cuales proveen la funcionalidad de recibir las solicitudes por parte de los usuarios u otros servicios Web. Mediante los elementos *invoke* se realiza las ejecuciones de los servicios Web las cuales mediante operadores como *flow*, *sequence* y *while* pueden realizarse en forma paralela, secuencial o iterativa. Así también, el elemento *reply* (mediante una combinación de asignaciones) puede actuar como un constructor de respuestas. Finalmente, la especificación de BPEL4WS provee algunos mecanismos para la compensación de fallos los cuales se encuentran en los elementos *faultHandlers*, *eventHandlers* y *compensationHandler.*

Migliardi et. al. [108] propone un modelo de integración para la invocación de servicios Web basado en una codificación XDR para mejorar el rendimiento de WSIF (*Web Services Invocation Framework*). La idea principal de este enfoque consiste en modificar las descripciones de los protocolos de comunicación contenida en los elementos binding de un documento WSDL por una descripción reducida codificada en XDR. Esta solución está orientada a disminuir el *overhead* en las comunicaciones de servicios Web que utilizan protocolos como TCP/HTTP/SOAP los cuales producen un *overhead* alto. Para llevar a cabo esto, se implementó un elemento de extensibilidad XDR para la API WSDL4J y sus correspondientes serializadores/deserializadores de objetos que utiliza WSIF. Para probar la extensión se implementó un servidor de aplicaciones escrito en Java el cual recibe los datos XDR y los convierte a sus correspondientes tipos de Java. Luego, invoca el método solicitado a través de la reflexión de Java. Sin embargo, la implementación tiene fuerte limitaciones debido a que solo tiene soporte para tipos de datos numéricos primitivos, así como para arreglos de tipos de datos numéricos primitivos. En BPIMS-WS, sólo la invocación de algunos servicios Web se realiza a través de WSIF debido a que WSIF no tiene soporte para tipos de datos complejos como listas (arreglos) de descripciones de

productos y listas de proveedores de productos. Esto es importante ya que algunos servicios que provee BPIMS-WS necesitan como parámetros de entrada listas de productos, para la búsqueda y compra de productos de forma distribuida. Es por esto, que cuando un servicio Web tiene como parámetros de entrada un tipo de dato primitivo (string, int, entre otros), BPIMS-WS utiliza a WSIF para realizar la invocación al servicio, en caso contrario, BPIMS-WS construye la solicitud de invocación mediante SOAP, evitando los problemas antes mencionados.

Nagano et. al. [109] propone una solución para llevar a cabo invocaciones de servicios Web en tiempo de ejecución cuando dos servicios presentan estructuras similares de sus esquemas XML donde se definen los tipos de datos que se intercambian en una invocación. La solución utiliza relaciones de suposición *(subsumption*) para traducir los tipos de esquemas de los parámetros de invocación sin modificar los procedimientos de invocación del solicitante. Para esto, se estudian cuatro casos de similitud: (1) Que los servicios contengan nombres de elementos iguales; (2) Que los servicios utilicen sinónimos en los elementos; (3) Que en uno de los servicios uno de los elementos tenga una correspondencia a dos elementos en el otro servicio y (4) Que cada servicio presenta elementos únicos. En este sentido, BPIMS-WS no necesita mecanismos de traducción para poder detectar similitud en las descripciones de los procesos de negocio. Esto se debe a que BPIMS-WS utiliza ontologías las cuales proveen un vocabulario de conceptos y relaciones para describir el comportamiento de los procesos de negocio. Así también, las ontologías proporcionan un diccionario de datos es una colección de datos los cuales pueden describirse e interpretarse como conceptos con contexto. En este sentido, el diccionario de datos se utiliza para describir diferentes dominios de la información como productos electrónicos, de computación, entre otros; y para describir diferentes escenarios donde la información se puede aplicar como sucede en los procesos de negocio donde cada empresa puede utilizar diferentes tipos de actividades comerciales para llevar a cabo un proceso de negocio.

Yu et. al. [110] propone un mecanismo de adaptación de servicios Web para diferentes variaciones presentadas en el ambiente de ejecución de tal forma que no se altere el diseño original de los servicios y además provee un soporte para diferentes modelos de comunicación (síncrona y asíncrona) y para diferentes arquitecturas de *middleware* como

DCOM, CORBA, JXTA y Jini. Además, se propone una arquitectura de *middleware* poliárquica con soporte para la adaptación autogenerativa de los servicios Web. El mecanismo de adaptación restaura las comunicaciones des aplicaciones distribuidas con servicios remotos a través de la generación de código móvil (en forma de adaptadores, filtros y *wrappers*). Como resultado, las aplicaciones distribuidas tienen adaptación en tiempo de ejecución y características de interoperabilidad con soporte e interacción con servicios obicuos y de lagado en una variedad de estilos y estándares. La generación automática del código móvil, se utilizan plantillas predefinidas donde la única información que se inserta es el protocolo de comunicación de las aplicaciones distribuidas. En comparación, BPIMS-WS no provee algún mecanismo de adaptación para diferentes modelos de comunicación. El conjunto de servicios que provee BPIMS-WS son servicios Web por lo que no se tiene alguna interoperabilidad con funcionalidades descritas en DCOM, CORBA, JXTA o Jini. En BPIMS-WS se tiene el soporte para comunicación síncrona y asíncrona pero no bajo diferentes arquitecturas de *middleware*.

JianJun et. al. [61] propone una arquitectura orientada a servicios como un marco de trabajo basado en servicios Web, el cual permite la invocación dinámica de servicios Web publicados en repositorios UDDI. Este marco de trabajo esta orientado a resolver el problema de obtener los cambios que se suscitan en los servicios Web como son el cambio de la descripción, interfaz o disponibilidad. Para esto, se utilizan mecanismos de publicación/suscripción y notificación para obtener listas de servicios Web de repositorios UDDI privados donde únicamente se almacena información de los servicios provenientes de repositorios UDDI públicos. Cuando un repositorio UDDI público agrega, elimina o actualiza un servicio Web, el repositorio UDDI privado recibe todos estos cambios mediante el mecanismo de notificación y realiza los cambios pertinentes de su información. Este no es un buen enfoque para la obtención de los cambios de los servicios Web publicados en un repositorio UDDI a través de un repositorio espejo ya que el mantenimiento es costoso cuando se tiene un número grande de servicios publicados debido a que se debe tener un repositorio espejo para cada nodo UDDI público. Para resolver esto, BPIMS-WS provee un mecanismo de actualización para las descripciones de los servicios Web. Este mecanismo es muy burdo ya que para actualizar una descripción de un servicio, primero es necesario eliminarla y posteriormente volver a registrarla. BPIMS-WS provee

un mecanismo de notificación, sin embargo, este se utiliza bajo otro contexto. En BPIMS-WS, el mecanismo de notificación se utiliza cuando hay un proveedor para el producto, pero el producto no está disponible en ese momento. En este caso, BPIMS-WS se compromete en obtener el producto solicitado de los proveedores registrados.

Gorton et. al. [111] propone una arquitectura para la integración dinámica de fuentes de información. La arquitectura emplea técnicas basadas en conocimiento para ejemplificar y clasificar el contenido de una fuente de información y los datos de interés del usuario. De esta forma, los usuarios son notificados cuando nuevos datos que son similares a sus intereses puedan integrarse al *middleware* de la arquitectura. A través de esto, los usuarios se conectan rápidamente a nuevas fuentes de información. La arquitectura contiene los siguientes subsistemas: (1) Descubridor de fuentes de información el cual localiza e integra las fuentes de información con el *middleware* de la arquitectura; (2) Integrador de fuentes información el cual interactúa con el descubridor para realizar el proceso de integración a través de un conjunto de interfaces basadas en servicios Web; (3) Clasificador de las fuentes de información el cual clasifica los datos y meta-datos a través de algoritmos que extraen un resumen representativo del contenido de la fuente de información y; (4) Firmador del conocimiento el cual crea una firma de conocimiento que captura las relaciones del contenido de los datos a un rango de aplicaciones expresadas como ontologías de dominio. Para todo esto se desarrolló un prototipo que integra fuentes heterogéneas de información. El prototipo se sustenta en un proyecto de investigación llamado *Data Concierge*, y provee una API genérica para la descripción de meta datos para facilitar la integración dinámica. La invocación de servicios Web se realiza mediante el uso de la API genérica a través de la adjunción y análisis de la información. La arquitectura propuesta por Gordon es similar a la de BPIMS-WS, solo que para servicios Web y no fuentes de información. En comparación BPIMS-WS presenta un servicio de descubrimiento el cual localiza los servicios Web, el servicio de integración junto con el invocador dinámico realizan la integración de los servicios Web. Esta funcionalidad es similar al descubridor e integrador de fuentes de información en la propuesta de Gordon. El clasificador de las fuentes de información puede verse como el nodo UDDI extendido que provee BPIMS-WS donde se encuentran clasificados los negocios, productos y servicios Web. Finalmente, BPIMS-WS también utiliza ontologías para definir las relaciones de los

procesos de negocio y los productos a un rango de aplicaciones expresadas como ontologías de dominio.

4.7. Resumen

Uno de los problemas centrales en la cadena de suministro es la localización de los procesos de negocio que cumplan ciertos requisitos comerciales con el fin de llevar a cabo el proceso de integración con ellos. A este proceso de localización se le conoce como descubrimiento de servicios Web. El descubrimiento de los servicios Web se basa principalmente en encontrar los procesos de negocio que ofrecen las organizaciones para satisfacer ciertas necesidades. En este capitulo se describe detalladamente los mecanismos de descubrimiento e invocación de servicios Web que utiliza BPIMS-WS presentando los algoritmos y técnicas utilizadas. Dentro de estas técnicas se encuentran el uso de USML y WSIL los cuales son especificaciones que permiten descubrir servicios con un valor agregado. En el caso de USML, permite construir y realizar consultas a nodos UDDI mediante diferentes criterios de búsqueda. En el caso de WSIL, permite buscar y localizar servicios Web que se encuentren o no publicados en nodos UDDI. Finalmente, se revisan los trabajos relacionados tanto en el descubrimiento como en la invocación de servicios Web y se discuten sus ventajas y desventajas en comparación con éstos.

Capítulo 5
Orquestación de Servicios Web en BPIMS-WS

Debido al dinamismo inherente del comercio electrónico B2B y de la cadena de suministro, las organizaciones requieren de la habilidad para adaptarse a los requerimientos de los clientes y a las condiciones cambiantes del mercado por lo que se requiere de mecanismos para su integración con el fin de ofrecer servicios Web más completos, que una sola entidad no podría prestar. En este capítulo se describe el mecanismo utilizado para la composición de procesos de negocio en BPIMS-WS, es decir, se describen las técnicas empleadas para la creación de nuevos procesos de negocio descritos como servicios Web en tiempo de diseño y ejecución. Así también, se revisan los trabajos relacionados y se discuten sus ventajas y desventajas en comparación con éstos.

5.1 Orquestación de Servicios Web

Los servicios Web por sí mismos proveen una forma eficaz de integrar aplicaciones de negocio orientadas a la Internet. Sin embargo, debido a que un negocio es en esencia la suma de sus procesos el verdadero valor de los servicios Web radica en la conexión de sus servicios, lo que provee un mayor valor a la organización. Existen dos perspectivas desde las cuales se puede abordar el problema de conectar un conjunto de servicios Web para que colaboren entre sí. Estas perspectivas son la orquestación y la coreografía. La orquestación de servicios Web consiste en proveer los medios necesarios para establecer un proceso de negocio ejecutable que permita la interacción entre servicios Web internos o externos. El proceso de negocio es dirigido por una de las partes del negocio. Mientras que la coreografía es más distribuida en el sentido de que ninguna de las partes controla el proceso

del negocio. De hecho, en la coreografía, cada parte involucrada en el proceso describe el papel que juega en la interacción. Se puede decir que en la orquestación se plantea la ejecución de un proceso desde un punto de vista único y general, en tanto que en la coreografía se deja a cada una de las partes del proceso ejecutarse desde su perspectiva sin que ninguna de ellas controle la conversación.

En la actualidad existen principalmente dos lenguajes estándares para realizar orquestación y coreografía. El primero es el *Web Service Choreography Interface* (WSCI) el cual describe desde la perspectiva de cada uno de los servicios Web su participación en el intercambio de los mensajes, es decir, más apegado a la coreografía. Mientras que el segundo es el *Business Process Execution Language for Web Services* (BPEL4WS) el cual se apega tanto a la coreografía como a la orquestación.

BPEL4WS define una notación para especificar el comportamiento de un proceso de negocio basado en los Servicios Web, es decir, modela un *workflow*. Un *workflow* consiste en la automatización de procesos de negocio, parcial o total, durante la cual los documentos, información o tareas se transfieren de un participante a otros, de acuerdo a un conjunto de reglas de procedimiento. BPEL4WS provee también un modelo basado en una alta interoperabilidad que facilita la integración de procesos automatizados en los espacios internos de una corporación y entre las empresas.

BPIMS-WS permite la orquestación de servicios Web en tiempo de ejecución mediante la concretización de documentos BPEL4WS. Un documento BPEL4WS es un patrón que define un proceso genérico de negocio que involucra 2 o más procesos simples de negocio, en este caso, un proceso simple es un servicio Web.

Para el proceso de orquestación, BPIMS-WS utiliza el repositorio de documentos BPEL4WS que contienen procesos genéricos de negocio.

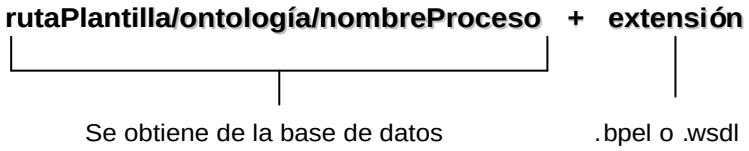
5.1.1. Descripción de Procesos Genéricos de Negocio

Un proceso genérico de negocio es una abstracción de un proceso de negocio que comúnmente se lleva a cabo al interior o exterior de una organización. En esta abstracción se pretende diferenciar las partes de un proceso de negocio que no cambian de las que sí lo hacen. La forma en que se procede para crear las plantillas o procesos de negocio genéricos

es, primeramente, construir un flujo de trabajo funcional en BPEL4WS. Posteriormente, se analiza cuáles son las partes del flujo de trabajo que pueden cambiar en tiempo de ejecución (como los socios comerciales) y cuáles son inherentes al proceso. Las partes que cambian se dejan incompletas obteniendo así una plantilla con base en las partes del flujo de trabajo que no cambian. Estas partes se completan en tiempo de ejecución cuando el cliente decida lo socios comerciales con los que desea interactuar.

5.1.2. Recuperación dinámica del flujo de trabajo genérico escrito en BPEL4WS

Para la recuperación de la plantilla, primero se determina la ubicación concatenando la información de la ontología de servicios con el nombre del proceso seguida de la extensión. La ruta de la plantilla queda determinada por:



Una vez que sabe dónde se encuentra la plantilla, BPIMS-WS procede a recuperarla junto con la plantilla WSDL y las plantillas de configuración necesarias para la máquina virtual que interpreta los documentos BPEL4WS. Los documentos WSDL y BPEL4WS se obtienen usando las APIs de JAXP [80] las cuales se encuentran en el *Java Web Service Developer Pack*.

5.1.3. Concretización de la plantilla

La implementación utilizada en este trabajo de la máquina de ejecución de BPEL4WS es el BPEL4WS Server de Oracle. Para poder agregar participantes en un proceso de negocio se requiere crear un documento XML llamado *Wrapper* por cada participante y cuyo nombre de archivo es el nombre del servicio Web concatenado con la palabra *Wrapper*. Este documento permite establecer un enlace entre el proceso de negocio y sus participantes

usando un elemento *PartnerLinkType* para cada uno. Un ejemplo de un documento *Wrapper* construido se muestra en la Fig. 5.1.

```
<?xml version="1.0" encoding="utf-8"?> <definitions
name="get_PriceandDeliveryTime"
targetNamespace="http://www.cinvestav.com/get_PriceandDeliveryTime.wsdl"
xmlns="http://schemas.xmlsoap.org/wsdl/"
xmlns:plnk="http://schemas.xmlsoap.org/ws/2003/05/partner-link/"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:tns="http://www.cinvestav.com/get_PriceandDeliveryTime.wsdl"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsd1="http://www.cinvestav.com/get_PriceandDeliveryTime.xsd1">
   <import location="get_PriceandDeliveryTime.wsdl"/>
   <plnk:partnerLinkType name="partner0PLT">
     <plnk:role name="partner0Provider">
       <plnk:portType name="tns:get_PriceandDeliveryTimePortType"/>
     </plnk:role>
   </plnk:partnerLinkType>
 </definitions>
```

Fig. 5.1 Ejemplo de un documento Wrapper

BPIMS-WS analiza los documentos WSDL de los participantes y la información más importante se recupera y se utiliza para generar el *Wrapper*. El espacio de nombres del servicio Web participante se incluye como atributo del elemento *definitions*. En este caso se agrega el atributo *targetNamespace* y *xmlns:tns* con el espacio de nombres del servicio Web. Luego se agrega un elemento *import* que tiene la localización del documento WSDL del participante. Finalmente, se agrega un elemento *partnerLinkType* al elemento *definitions* junto con los correspondientes elementos *role* y *portType*. Este último corresponde al ofrecido por el participante en su documento WSDL. El nombre del elemento *portType* es dividido en dos partes: el espacio de nombres y el nombre del elemento *portType* del servicio remoto. Los documentos *Wrappers* de los otros participantes en el flujo comercial compuesto se construyen de la misma manera.

Por otro lado, la concretización de un documento BPEL4WS ejecutable consiste en incluir o modificar algunos de sus elementos. La Figura 5.2 muestra una plantilla BPEL4WS concretizada. Evidentemente, en esta figura no se puede mostrar el código completo de la

plantilla concretizada debido a que es bastante largo y resultaría confuso. Sin embargo, el código presentado es suficientemente ilustrativo para los fines de esta sección.

```
<process name="lowpriceprocess"
xmlns="http://schemas.xmlsoap.org/ws/2003/03/business-process/"
xmlns:pn0=http://www.cinvestav.com/get_PriceandDeliveryTime.wsdl"
…
xmlns:tns="http://www.cinvestav.com/getLowPrice.wsdl">
<partnerLinks>
  <partnerLink name="User" partnerLinkType="tns:UserSLT"
       partnerRole="InformationProvider"/>
  <partnerLink name="partner0LT"
       partnerLinkType="tns:partner0PLT"
       partnerRole="partner0Provider"/>
        …
</partnerLinks>
<variables>
  <variable name="uRequest" messageType="tns:getLowPriceRequest"/>
 <variable name="uResponse" messageType="tns:getLowPriceResponse"/>
 <variable name="partner0Request" messageType="pn0:get_ArrayRequest"/>
 <variable name="partner0Response" messageType="pn0:get_ArrayResponse"/>
       …
</variables>
   <sequence name="sequence">
       <receive name="receiveRequest" operation="getLowPrice" partnerLink="User"
       portType="tns:getLowPricePortType" createInstance="yes"
       variable="uRequest"/>
       <assign name="CreateLowPriceInput">
       <copy>
         <from variable="uRequest"/>
         <to variable="partner0Request"/>
       </copy>
       …
       </assign>
       <invoke name="invokeService" inputVariable="partner0Request"
       operation="get_Array" outputVariable="partner0Response"
        partnerLink="partner0LT"portType="pn0:get_ArrayPortType"/>
         …

       <reply name="sendInformation" operation="getLowPrice" partnerLink="User"
       portType="tns:getLowPricePortType" variable="uResponse">
       </reply>
   </sequence>
</process>
```

Fig. 5.2 Plantilla BPEL4WS concretizada. Las instrucciones en negritas son añadidas o modificadas dinámicamente

Como se muestra en la figura, los espacios de nombre de los servicios Web, los elementos *partnerLinks*, las variables de entrada y salida y las invocaciones a los servicios externos se agregan a la plantilla BPEL4WS. Nótese que un espacio de nombres y un elemento

partnerLink deben ser añadidos para cada servicio Web junto con sus variables de entrada y salida. También se puede apreciar que a las variables se les asigna la información que el socio expone en su documento WSDL. De ahí se obtiene el tipo del mensaje con el que se va a intercambiar información (entrada y salida).

Por otra parte, los elementos *partnerLinks* se construyen con la información que proviene de los documentos WSDL que describen la funcionalidad de cada socio comercial. En particular, el nombre del elemento *partnerLinkType* y sus correspondiente elemento *partnerRole* deben coincidir con el que se agrega en el elemento *partnerLink* del documento BPEL4WS. El nombre del elemento *portType* asociado con el elemento *partnerLinkType* también se obtiene de los documento WSDL. La descripción del proceso de negocio escrito en BPEL4WS se ejecuta en la máquina de flujos de trabajo que se encarga de orquestar las invocaciones a los servicios Web,

5.1.4. Programación de la Máquina de Ejecución BPEL

Para la orquestación de servicios Web, BPIMS-WS utiliza el servidor BPEL de Oracle que proporciona un ambiente escalable y confiable para desplegar, ejecutar y administrar procesos BPEL [112]. BPIMS-WS programa el servidor BPEL a través de la definición de proyectos BPEL. Un proyecto BPEL está compuesto de varios tipos de archivos:

- **Un archivo con extensión WSDL** describe la interfaz pública del proceso comercial compuesto.
- **Un archivo con extensión BPEL** captura la declaración de los elementos *partnerLinks* que participan en el flujo comercial compuesto, las variables intercambiadas como parte del proceso comercial y la lógica que coordina las interacciones con los elementos *partnerLinks.*
- **Un conjunto de URLs** indican la localización de los documentos WSDL de los elementos *services/partnerLinks* participantes.
- **Un conjunto de documentos *XML Schema*** describen los tipos de mensajes y documentos XML usados en el flujo comercial compuesto. Esto es optativo ya que los

tipos de mensaje pueden definirse directamente en las definiciones WSDL del flujo comercial compuesto y sus socios comerciales.

- **Un archivo bpel.xml** es un descriptor de despliegue que nombra el proceso y liga la implementación BPEL, su interfaz pública y la localización de los servicios participantes en el flujo.

Cuando BPIMS-WS recupera la plantilla que corresponde a un servicio Web compuesto solicitado, ésta tiene asociados 2 archivos: bpel.xml y build.xml. BPIMS-WS concretiza el archivo bpel.xml con los documentos WSL que representan los servicios Web de los participantes en el flujo. Un ejemplo de antes y después del proceso de concretización se muestra en la Fig. 5.3.

ANTES

```
<?xml version="1.0" encoding="UTF-8"?>
<bpel-process id="" src="" wsdlLocation="">
</bpel-process>
```

DESPUES

```
<?xml version="1.0" encoding="UTF-8"?>
<bpel-process id="LPriceAndQuantity" src="LPriceAndQuantity.bpel"
wsdlLocation="LPriceAndQuantity.wsdl">
   <properties id="partner0LT">
     <property name="wsdlLocation">
         services/get_PriceandDeliveryTimeWrapper.wsdl
     </property>
   </properties>
   <properties id="partner1LT">
     <property name="wsdlLocation">
         services/get_ProviderQuantityWrapper.wsdl
     </property>
   </properties>
   <properties id="partner2LT">
     <property name="wsdlLocation">
         services/get_ProviderURLBuyWrapper.wsdl
     </property>
   </properties>
 </bpel-process>
```

Fig. 5.3 Archivo bpel.xml antes y después del proceso de concretización

Luego, BPIMS-WS concretiza el archivo build.xml con el nombre del proyecto BPEL y su localización. Un ejemplo de antes y después del proceso de concretización se muestra en la Fig. 5.4

ANTES

```
<?xml version="1.0"?>
<project name="" default="main" basedir=".">
  <property name="deploy" value="default"/>
  <property name="rev" value="1.0"/>
  <target name="main">
   <bpelc  input="${basedir}/bpel.xml" rev="${rev}" deploy="${deploy}"/>
  </target>
</project>
```

DESPUES

```
<?xml version="1.0"?>
 <project basedir="." default="main" name="LPriceAndQuantity">
   <property name="deploy" value="default"/>
   <property name="rev" value="1.0"/>
   <target name="main">
    <bpelc deploy="${deploy}" input="${basedir}/bpel.xml" rev="${rev}"/>
   </target>
 </project>
```

Fig. 5.4 Archivo build.xml antes y después del proceso de concretización

5.1.5. Ejecución de los Flujos de Trabajo

Una vez que BPIMS-WS concretiza el flujo de trabajo y crea los archivos de configuración correspondientes para la programación de la máquina de ejecución BPEL, se procede a su ejecución. Las plantillas concretizadas se almacenan en una máquina BPEL4WS para su ejecución. En particular, para la máquina BPEL4WS de Oracle se usa el siguiente comando que ejecuta el flujo de trabajo:

obant - f /ruta/ServiceN/build.xml

donde N representa el número del identificador del usuario que corresponde al número de sesión que automáticamente se le asigna cuando se establece la comunicación con el servidor de aplicaciones Tomcat. Para la comunicación con el flujo de trabajo, BPIMS-WS construye mensajes SOAP conteniendo la información proporcionada por el cliente.

Dentro de las lecciones aprendidas es preciso señalar que en la especificación 1.1 de BPEL4WS, no existen ciclos *for* ni instrucciones *if*, por lo que se tuvieron que simular con las actividades *while* y *switch* respectivamente. Dicho lo anterior, todos los flujos de trabajo

en BPIMS-WS tienen el orden de complejidad O(n), donde n representa al número de empresas que ofrecen el producto. Finalmente, se previno que los flujos de trabajo no cayeran en abrazos mortales ya que en caso de que no haya ninguna empresa con el producto, no se ejecuta ninguno de los ciclos *while* y si existe un número de empresas, entonces los ciclos *while* realizan un número limitado de iteraciones correspondientes al número de empresas que ofrecen un producto en base al tipo de solicitud.

5.2. Trabajos Relacionados con BPIMS-WS en la orquestación de servicios Web

Con el advenimiento de los servicios Web, la composición automática de flujos de trabajo basado en servicios Web se ha convertido en un punto importante de investigación. Esto ha dado origen a numerosas propuestas y técnicas reportadas en el área. En la década de los 90s, diversos trabajos [113, 114, 115] se desarrollaron como solución a problemas de modelación y ejecución de actividades a alto nivel. Estos problemas incluían la concretización incremental de modelos gráficos de alto nivel, administración de transacciones comerciales duraderas, adaptación dinámica, captura de procesos, por mencionar solo algunos. Trabajos más recientes abordan el problema de encontrar un servicio Web que satisfaga ciertas condiciones de un cliente. Para esto, se utilizan algunos mecanismos para la orquestación de servicios bajo un enfoque centralizado.

Verma et. al. [116] presenta un sistema para la integración dinámica de servicios Web para especificaciones abstractas de flujos de procesos de negocio utilizando un mecanismo de descubrimiento semántico basado en restricciones. Las especificaciones de los flujos de procesos de negocio se definen en niveles de tareas abstractas no importando los detalles de los protocolos de comunicación de servicios específicos y los flujos de ejecución, de tal forma que el sistema los descubra automática o semi-automáticamente. Los flujos de procesos de negocio se describen en BPEL4WS agregando semánticas para facilitar el descubrimiento dinámico de servicios Web. Para la composición es preciso determinar las dependencias entre los servicios. El sistema presenta los siguientes componentes: (1) Nodo UDDI semántico el cual es un repositorio de servicio con soporte para descripciones de

servicios descritas en DAML-S; (2) Verificador de restricciones el cual verifica que las pre y post-condiciones de los servicios se cumplan; (3) Invocador dinámico el cual construye y realiza las invocaciones a los servicios Web. Todos estos componentes forman un Proxy Genérico de servicios Web el cual descubre los servicios Web candidatos, automáticamente los integra, los invoca y regresa el control a la máquina BPEL que ejecuta el flujo de trabajo. En comparación, BPIMS-WS orquesta servicios Web descritos en flujos de trabajo BPEL4WS pero sin agregar semánticas para facilitar el descubrimiento de los servicios Web. Bajo el enfoque de BPIMS-WS, no es necesario determinar dependencias de servicios ya que los procesos de negocio tienen un comportamiento bien definido en base a ontologías, que en este caso, los procesos de negocio participantes en un flujo de trabajo siguen el comportamiento de la ontología de procesos de negocio de RosettaNet. Al igual que la propuesta de Verma [116], BPIMS-WS provee un *proxy* el cual descubre, automáticamente integra, orquesta e invoca servicios Web regresando el control a una máquina BPEL que ejecuta el flujo de trabajo. La diferencia estriba en que BPIMS-WS no provee un verificador de restricciones dado que los socios comerciales junto con sus procesos de negocio se conocen a priori.

Estublier et. al. [117] extiende las capacidades de flujos de trabajo para coordinación y orquestación de servicios Web. Para la orquestación, se desarrolló un sistema llamado Melusine el cual provee un modelo gráfico que define las actividades y el flujo de datos. El sistema Melusine tiene una diseño de tres capas para realizar la composición y coordinación de los servicios Web, las cuales son: (1) Capa de Mediación en la cual se distinguen los papeles (*roles*) y *proxies*. Un papel (*role*) es un servicio abstracto el cual define funcionalidades ofrecidas (interfaces) y utilizadas (receptáculos). Un *proxy* es una pieza de código que realiza una funcionalidad específica, pero que también actúa como un mediador el cual invoca a un componente de servicio. (2) Capa de componentes en donde se provee un conjunto herramientas que contienen *wrappers* que funcionan como adaptadores e integradores de servicios Web los cuales verifican la compatibilidad de los parámetros de entrada y salida, así como los protocolos de comunicación que se utilizan y; (3) Capa de coordinación en donde se contiene un conjunto de adaptadores los cuales sustituyen componentes por otros que ofrecen un funcionalidad similar o que ofrezcan un mismo papel (role). BPIMS-WS no provee un modelo gráfico para definir las actividades y

el flujo de los datos con el fin de realizar la composición de servicios Web. Sin embargo, la máquina BPEL que utiliza BPIMS-WS, la cuales es el *Workflow Manager* de Oracle, provee un conjunto de herramientas gráficas que permiten modelar orquestaciones de servicios Web. No obstante, BPIMS-WS permite inspeccionar y modelar con diagramas UML de secuencia orquestaciones de servicios Web. En BPIMS-WS lo importante no es cómo se modela una orquestación de servicios, sino conocer cuáles son los participantes involucrados y los mensajes que se están intercambiando. Un diagrama UML de secuencia provee estas características.

Anane et. al [118] propone un marco de trabajo compuesto para asegurar el óptimo uso y coordinación de servicios Web. Este marco de trabajo tiene soporte para tecnologías de servicios Web, servicios Grid y Web semántica para una plataforma basada en BPEL4WS. La composición se realiza de forma híbrida debido a que se pueden orquestar servicios Web y servicios Grid incorporando integración dinámica a través de la mediación de agentes. Para la composición de servicios Grid es necesario realizar una trasformación a servicios Web y establecer una relación entre agentes y servicios Web en la plataforma BPEL4WS. Para esto, los servicios Grid se transforman en servicios Web virtuales los cuales ofrecen un medio para desacoplar el proceso de composición de los elementos *binding* descritos en los documentos WSDL. En un documento WSDL, los elementos *binding* definen los protocolos de comunicación que soporta un servicio Web. Con esto, todas las interfaces definidas por los servicios Grid se redefinen en *JavaBeans* como un tipo complejo XML con un atributo público de servicio Grid en el documento WSDL. El código WSDL asociado con el servicio Web virtual incluye un identificador para los espacios de nombres y los elementos *PartnerLinks*. Los elementos *PartnerLinks* establecen la relación en términos conversacionales entre dos servicios a través de la definición de los papeles que cada servicio tiene en la conversación y de la especificación de sus operaciones en los elementos *portType* expuestos por cada servicio para recibir mensajes. El papel (*role*) del servicio Web virtual en el documento WSDL que representa el servicio Web compuesto, se genera a través de la combinación de un nombre y un valor aleatorio, para que pueda identificarse de manera única en el flujo de trabajo descrito en BPEL. También, se define e implementa un operador adicional en el servicio Web virtual para crear nuevas instancias de los servicios Grid. En el agente de mediación, el servicio Web virtual incluye operadores

que activan a un agente cuando se recibe la solicitud de la máquina BPEL, o se provee una entrada. En este sentido, un servicio Web virtual no contiene operaciones, solo interfaces a los agentes. La composición se realiza a través del uso de plantillas de agentes las cuales contienen dos funciones elementales: (1) Escuchar los eventos activados por un servicio Web virtual y (2) Enviar la respuesta de otros agentes o servicios Web virtuales. La plantilla solo incluye un número predefinido de protocolos de comunicación genéricos. Este es un trabajo muy similar al proceso de orquestación de servicios que utiliza BPIMS-WS ya que también se utilizan plantillas preestablecidas que representan funcionalidades genéricas. En BPIMS-WS, los roles de los participantes se encuentran preestablecidos, entonces, lo que se determina es el servicios Web apropiado que satisfaga el role. Es por esto que en BPIMS-WS, los espacios de nombre de los servicios Web, los socios involucrados, las variables de entrada y salida y las invocaciones a los servicios externos son agregados a la plantilla BPEL4WS. Así también, a las variables se les asigna la información que el socio expone en su WSDL. De ahí se obtiene el tipo del mensaje con el que se va a intercambiar información (entrada y salida). Por otro parte, los elementos *partnerLinks* se construyen con la información que proviene del WSDL, y se le asigna un nombre correspondiente al elemento *partnerRole* los cuales deben coincidir con el que se agrega en el elemento *partnerLink* del BPEL4WS. En este sentido, la asignación de espacios de nombres, nombres y roles en el flujo de trabajo no se realiza de manera aleatoria, tal como sucede en el enfoque de Anane.

Tai et. al. [119] propone un enfoque para la composición de servicios Web coordinados basado en políticas introduciendo un nuevo modelo y *middleware* que habilita la integración flexible de diversos tipos de coordinación en la composición de servicios Web que representan procesos de negocio. Para esto, se toman características de BPEL4WS para la parte de composición y *WS-Coordination* para la coordinación de protocolos para las actividades distribuidas. El *middleware* consiste de un compilador BPEL, un conjunto de políticas y procesador de transacciones de servicios Web el cual soporta tipos de coordinación *WS-AT* y *WS-BA*. *WS-AT* (*Web Services Atomic Transaction*) y *WS-BA* (*Web Services Business Activity*) son especificaciones que definen acuerdos para los protocolos de comunicación. Las políticas se basan en *WS-Policy* (*Web Service Policy*) el cual define un modelo y sintaxis de propósito general para expresar propiedades funcionales o no

funcionales de un servicio Web en una forma declarativa. En este sentido, una política es una expresión XML que combina una o mas sentencias que especifican características abstractas o concretas de servicios Web tal como un esquema de autenticación requerida o una calidad de servicios deseada. Por ejemplo, en un servicio Web se podrían definir esquemas de encriptación, es decir, que la información que recibe y que envía el servicio Web debe estar cifrada; selección del protocolo de transporte en donde se define que el protocolo de comunicación es SMTP, entre otros. En comparación, BPIMS-WS no provee el soporte para la descripción de políticas. No obstante, en versiones futuras de BPIMS-WS se esta considerando incorporar estas características, por lo que se podrían definir políticas de seguridad a los servicios Web provistos por BPIMS-WS. Para esto, cada descripción de servicio WSDL que ofrece BPIMS-WS bajo un documento WSDL debe sufrir algunas modificaciones al respecto ya que las políticas de seguridad pueden incluirse en los elementos *portype*, *operation* o *message* de un documento WSDL. Un elemento *message* define los parámetros (de entrada o salida) que son necesarios para la invocación de una operación en un servicio Web. Un intercambio de mensajes entre un proveedor de servicio y un solicitante se describe como un elemento *operation*. Por otra parte, un elemento *portType* define un conjunto de operaciones.

Xiulan et. al. [120] introduce dos conceptos para la composición de servicios Web: socio virtual e inspector. Un socio virtual es un pseudo servicio Web que tiene la misma interfaz que el servicio Web actual pero con diferentes protocolos de comunicación e integración. Un socio virtual se invoca directamente desde un proceso de negocio descrito en BPEL para que pueda supervisarse el rendimiento funcional y no funcional durante el ciclo de desarrollo con el fin de evitar problemas en tiempo de ejecución, o para seleccionar los socios comerciales en tiempo de diseño. Los socios virtuales proveen un conjunto de técnicas para explorar cada aspecto de sus programas. El inspector se utiliza bajo un esquema de intermediación externa, es decir, el inspector funciona como un servicio Web que provee operaciones de intermediación como la búsqueda y localización de servicios. Un inspector es un servicio Web donde cualquier programador puede registrar cualquier información de salida. Estos dos conceptos se incluyen en un prototipo de modelación automática y de ambiente de simulación. Bajo el ambiente de simulación, la composición se realiza mediante un conjunto de herramientas que permiten inspeccionar y simular el

proceso de composición utilizando una máquina BPEL externa. En el ambiente se permiten dos estados: de simulación y modelación. Como se discutió con el trabajo de Estublier [117] et. al., BPIMS-WS no provee mecanismo de modelación para la orquestación de servicios Web. Lo que provee BPIMS-WS es un marco de trabajo para la ejecución de flujos de procesos de negocio, más no un ambiente de simulación como es el caso del enfoque propuesto por Xiulan.

Jiamao [121] propone un enfoque de composición semántica y dinámica de servicios Web. En este enfoque, los servicios Web se modelan con algunas reglas descritas en XML cuyos encabezados y cuerpos se relacionan con una ontología semántica para eliminar los conflictos semánticas en el proceso de orquestación. Para ello, se utiliza un algoritmo de encadenamiento hacia atrás sin *backtrace*. A partir de ciertos parámetros de entrada (precondiciones) y de salida (postcondiciones), se genera automáticamente un plan de composición de servicios Web descrito en BPEL4WS para que se ejecute y entregue la respuesta correspondiente. El motivo de no incluir *backtrace* en el algoritmo de composición es no afectar la eficiencia cuando sea el caso en que se tengan un número de reglas demasiado grande. Como se ha discutido anteriormente, BPIMS-WS no provee el soporte para algún tipo de razonamiento basado en reglas, y mucho menos se utiliza para descubrir los servicios participantes en una composición de servicios Web. BPIMS-WS sigue un modelo de descubrimiento basado en ontologías de procesos de negocio, por lo cual este modelo se sigue para la orquestación de servicios Web. Bajo este enfoque, los socios comerciales se conocen a priori, lo cual no es necesario utilizar un algoritmo como encadenamiento hacia atrás o hacia delante en el proceso de descubrimiento. BPIMS-WS solo realiza consultas a las ontologías contenidas en el nodo UDDI extendido para determinar cuál o cuáles son los servicios Web candidatos. Esta forma de descubrir es rápida en comparación cuando se utilizan algoritmos de planeación los cuales podrían no entregar una respuesta satisfactoria.

Bajo un enfoque de composición punto-a-punto, Benatallah et. al. [122] propone un modelo para la composición de servicios Web multiempresariales. Este modelo de composición llamado *Self-Serv* habilita la composición declarativa de nuevos servicios a partir de otros existentes, la selección dinámica de multi-atributos de servicios en una composición y una comunicación punto-a-punto en las ejecuciones de los servicios Web compuestos. Para la

composición se introducen dos componentes: un orquestador de servicios y un contenedor de servicios. Para expresar la lógica de negocio de una operación de un servicio compuesto se utilizan diagramas de estado que codifican el flujo de las invocaciones a las operaciones de los componentes de servicio. Un diagrama de estados se constituye de estados simples y compuestos, y de transiciones las cuales se etiquetan de acuerdo a reglas ECA (Evento-Condición-Acción). En la composición se utilizan mecanismos de membresía y calificación de los servicios. Para la composición se utilizan coordinadores de estado y tablas de enrutamiento. Un coordinador de estado se responsabiliza de recibir notificaciones de otro coordinador y determina cuál es el siguiente estado. Además, invoca los servicios Web con sus precondiciones correspondientes y notifica a los coordinadores de los resultados y que la ejecución finalizó. Las tablas de enrutamiento se utilizan para determinar las transiciones de los mensajes entrantes y salientes en una composición de servicio. El enfoque que utiliza BPIMS-WS para la orquestación de servicios es un modelo centralizado. Bajo este enfoque, BPIMS-WS tiene una solo unidad de procesamiento para los flujos de trabajo, que en esta caso en la máquina BPEL, por lo que BPIMS-WS presenta problemas de rendimiento debido a la generación de cuellos de botella. A pesar de esto, el mantenimiento es más fácil que en un enfoque punto-a-punto, sobre todo cuando se traduce en cada nodo debe conocer todos los servicios Web asociados con sus nodos vecinos, por lo cual, debe de conocer perfectamente las descripciones de servicios para realizar el proceso de integración. Este factor es muy importante, ya que cuando se tiene un gran número de nodos se traduce en un alto costo de desarrollo y mantenimiento debido a la gran cantidad de servicios. Además, se deben proveer los mecanismos necesarios para el control de concurrencia y recuperación de fallos, que en esta caso no sucede con una máquina BPEL que se encarga de estos procesos.

5.3. Resumen

La orquestación de servicios Web consiste en proveer los medios necesarios para establecer un proceso de negocio ejecutable que permita la interacción entre servicios Web internos o externos de una organización. Esto es un factor importante debido a que la naturaleza dinámica de la cadena de suministro, las organizaciones requieren de la habilidad para adaptarse a los requerimientos de los clientes y a las condiciones cambiantes del mercado

por lo que se requiere de mecanismos para su integración con el fin de ofrecer servicios Web más completos, que una sola entidad no podría prestar. En este capítulo se describe el mecanismo utilizado para la composición de procesos de negocio en BPIMS-WS tanto en tiempo de diseño y ejecución. Dicho mecanismo consiste en la concretización de documentos BPEL4WS los cuales representan procesos de negocio genéricos. Para esto, se propone un repositorio de procesos de negocio genéricos los cuales son localizados, recuperados y concretizados en tiempo de ejecución. Finalmente, se revisan los trabajos relacionados en la composición automática y semi-automática de servicios Web y se discuten sus ventajas y desventajas en comparación con éstos.

Capítulo 6
Supervisión de Servicios Web en BPIMS-WS

La supervisión de procesos de negocio es un factor importante en la cadena de suministro. A través del proceso de supervisión, los socios involucrados en un proceso de negocio pueden dar seguimiento y verificar el desarrollo de las actividades comerciales con el fin de identificar problemas y oportunidades, y así puedan manejar respuestas apropiadas asegurando ganancias significativas y ventajas competitivas. En este capítulo se describe el proceso de supervisión de procesos de negocio, por lo que se describen las técnicas utilizadas en BPIMS-WS. Así también, se revisan los trabajos relacionados y se discuten sus ventajas y desventajas en comparación con éstos.

6.1. Proceso de Supervisión de Servicios Web

Debido a que un servicio Web esencialmente sigue un protocolo de coordinación y comunicación entre las entidades participantes, BPIMS-WS utiliza diagramas UML [123] de secuencia para modelar el ordenamiento de actividades comerciales descritas como servicios Web. BPIMS-WS da seguimiento a un proceso de negocio de manera visual, permitiendo no solo a los programadores de servicios Web depurar sus aplicaciones, sino también a los usuarios verificar el desarrollo de los servicios.

BPIMS-WS utiliza el lenguaje SVG (*Scalar Vector Graphics*) [124] para representar a los diagramas UML, debido a que los documentos de este lenguaje pueden visualizarse en los navegadores comunes de Internet, propiciando con ello la disponibilidad de la herramienta.

El motivo principal que originó su desarrollo fue que actualmente no se cuentan con suficientes herramientas capaces de realizar la tarea de supervisar procesos de negocio con

diagramas UML de secuencia utilizando lenguajes estándares de la Web. BPIMS-WS da seguimiento a la ejecución de procesos de negocios escritos en los lenguajes WSDL y BPEL4WS. Como se ha mencionado en capítulos anteriores, BPEL4WS es un lenguaje de descripción de procesos que describe la composición de servicios Web complejos a partir de otros más elementales usando estructuras convencionales de control de procesos.

Para la visualización de los diagramas se utilizó un dialecto de XML denominado SVG que actualmente atraviesa por un proceso de estandarización llevado acabo por el Consorcio de la Web. Debido a la creciente aceptación de SVG, su soporte queda a cargo de los navegadores de Internet generalmente disponibles en la forma de *plug-ins*.

La metodología empleada por BPIMS-WS consiste en extraer información de los mensajes SOAP intercambiados por servicios Web, identificando los participantes y la interacción que estos tienen durante la ejecución del servicio Web.

BPIMS-WS intercepta los mensajes SOAP que son intercambiados por los servicios Web, los cuales se almacenan en un búfer. A partir de esto, BPIMS-WS es capaz de generar un diagrama UML de secuencia haciendo un mapeo de los mensajes SOAP al diagrama ya que cuenta con toda la información requerida para este fin: los identificadores de los participantes, su dirección, el orden relativo en que ocurrieron los mensajes y el contenido de ellos. Con esto, se puede observar gráficamente la transmisión de los mensajes SOAP entre las entidades involucradas.

Para asegurar que BPIMS-WS no interfiera con el desarrollo del proceso observado, se crearon procesos ligeros o hilos (*threads*) los cuales fueron sincronizados por la disponibilidad de los mensajes recibidos. Dado que la ejecución de los servicios Web se desarrolla rápidamente y que la computación requerida para generar diagramas UML se incrementa de acuerdo al número de mensajes SOAP recibidos, se usó un área de almacenamiento temporal (*buffer*) para disminuir la diferencia entre las velocidades relativas en la producción y consumo de mensajes.

En la Fig. 6.1 presentan algunos ejemplos de diagramas UML de secuencia generados. Este ejemplo corresponde al PIP 3A2 de RosettaNet el cual consiste de una petición de precios y disponibilidades de productos. Se emplearon estas especificaciones de RosettaNet para utilizarlas como plantillas en la generación de los diagramas UML, los cuales son etiquetados con información extraída de los mensajes SOAP. En BPIMS-WS se usaron

animaciones en SVG. En los diagramas se animaron las flechas de los mensajes con desplazamientos sugiriendo la dirección y el sentido en que éstos se mueven. Estas animaciones no retrasan en forma importante el cómputo requerido para la generación del diagrama por lo que las seguiremos incluyendo en futuras versiones.

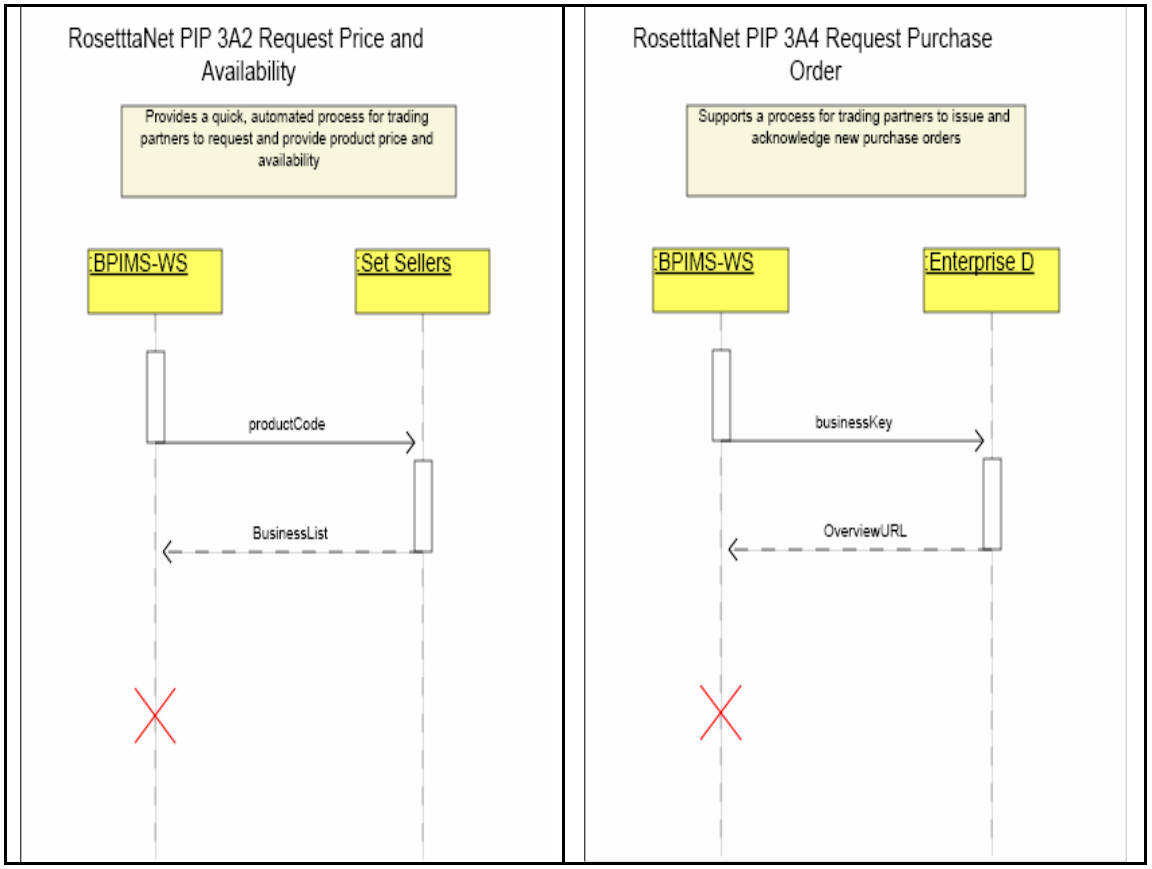


Fig. 6.1. Ejemplos de diagramas UML de secuencia generados por BPIMS-WS

Sin embargo, un inconveniente presente en la metodología, es la visualización de un documento SVG. Para visualizar un documento SVG en un navegador es necesario almacenar continuamente el documento SVG en disco duro debido a la presencia de elementos dinámicos que se incorporan o se eliminan continuamente en el diagrama, degradando con ello el rendimiento del sistema. Como una solución a esto, se implementaron algunos *scripts* y programas JSP para generar diagramas al vuelo,

manejando el documento únicamente en memoria. Otro inconveniente fue el *scrolling*, debido a que nuestros diagramas UML de secuencia crecen hacia abajo. La mayoría de los navegadores no soportan *scrolling* para visualizar documentos SVG, con lo que también incluimos *scripts* que emulan el *scrolling* de un navegador.

En resumen, BPIMS-WS ofrece la capacidad de supervisar los procesos de negocio mostrando los participantes, el flujo de lo mensajes y los valores de lo tipos de datos involucrados en las colaboraciones comerciales.

6.2. Trabajos Relacionados con BPIMS-WS en la supervisión de servicios Web

La necesidad de conducir los negocios en tiempo real todavía es un desafío importante que enfrentan las empresas hoy en día. Las empresas que operan eficazmente en tiempo real pueden detectar eventos significativos al instante, pueden identificar problemas y oportunidades, y pueden manejar respuestas apropiadas asegurando ganancias significativas y ventajas competitivas. La supervisión de las actividades comerciales (BAM, *Business Activity Monitoring*) es un concepto interesante que Gartner propuso. Gartner lo define como, el concepto de proporcionar el acceso en tiempo real a indicadores de rendimiento para mejorar la velocidad y efectividad de las actividades comerciales [125]. BAM ha surgido como un paradigma en el contexto de sistemas de tiempo real, originando nuevas propuestas y mecanismos para la supervisión de actividades comerciales. En este sentido, dentro de los primeros trabajos Geppert et al. [126] introduce un enfoque basado en anotaciones y análisis post-mortem de las ejecuciones de flujos de trabajo en EvE el cual es un sistema distribuido manejado por eventos para la administración de flujos de trabajo. En EvE se utiliza la tecnología de administración de bases de datos activas para expresar ciertos aspectos de administración de un flujo de trabajo como la sincronización de agentes o el flujo de control. El análisis post-mortem se realiza a través de consultar el historial de los eventos que ocurren en un flujo de trabajo. Para esto, los eventos se almacenan en una base de datos activa con soporte de reglas ECA (ECA, *Event-Condition-Action*). EvE presenta un conjunto predefinido de consultas especificadas en OQL, el cual es un lenguaje

de consultas para OODBMS. Para el análisis post-mortem se tienen clasificadas dos tipos de consultas en base a su propósito: (1) de detección de errores semánticos que determina si un flujo de trabajo finalizó con un resultado no deseado y (2) de optimización, que determina qué tan a menudo se ejecuta una instancia de un flujo de trabajo o qué tan a menudo un flujo de trabajo termina con un evento de terminación de cierto tipo. Estos tipos de consultas se utilizan para detectar ocurrencias de excepciones y fallos así como la detección de cuellos de botella e ineficiencias las cuales degraden el rendimiento de EvE.

Koksal et. al. [127] propone un enfoque para la administración de la historia de flujos de trabajo. La administración de la historia, es decir, la estructura y consulta se realiza a través del almacenamiento de los objetos que representan las actividades involucradas en un flujo de trabajo. Este almacenamiento se lleva a cabo mediante el uso de repositorios de datos los cuales pueden ser bases de datos o archivos de texto plano. Para esto, se describe la estructura de los objetos según la naturaleza de los datos y las necesidades de procesamiento, así también las estrategias de procesamiento de consulta en estos objetos. Para el procesamiento de consultas se utiliza el Servicio de Consulta de Objetos (*Object Query Service*) propuesto por la OMG, el cual provee un conjunto de operaciones para la selección, inserción, actualización y eliminación de colecciones de objetos. En este sentido, se desarrolló un Servicio de Consulta el cual provee interoperabilidad entre diferentes sistemas de consulta y soporte de diferentes lenguajes de consulta como SQL y OQL. Este servicio de consulta provee un conjunto de estrategias que muestran cómo formular las consultas para recuperar la información auditada. En particular, este conjunto de estrategias se enfoca en la reducción del costo de procesamiento considerando las siguientes métricas: (1) número de sitios involucrados; (2) número de niveles de anidación en un proceso; (3) costo de comunicación y (4) número de objetos en los repositorios de datos.

Muth et. al [128] presenta un sistema llamada Mentor-lite que provee una arquitectura para la administración de la historia de los flujos de trabajo. En la arquitectura de Mentor-lite se presentan los siguientes componentes: (1) Un interpretador el cual verifica las especificaciones de los flujos de trabajo; (2) Un administrador de comunicación el cual es responsable del envío de los mensajes entre diferentes máquina de ejecución de flujos de trabajo y (3) Un administrador de accesos el cual provee mecanismos de recuperación y de acceso para las diferentes máquina de ejecución. Como mecanismo de almacenamiento de

las instancias de los flujos de trabajo se propone una base de datos temporal como mecanismo de almacenamiento Esta base de datos puede consultarse en tiempo de ejecución. Para esto, se provee un conjunto predefinido de operaciones de consultas a través de un lenguaje de especificación basado en SQL. En comparación con los trabajos de Geppert [126], Koksal [127] y Muth [128], BPIMS-WS no provee mecanismos para realizar anotaciones y algún tipo de análisis (a priori, en ejecución o a posteriori) de los procesos de negocio descritos en los flujos de trabajo. Así también, BPIMS-WS no utiliza algún dispositivo de almacenamiento persistente como una base de datos para almacenar los eventos que ocurren en una actividad comercial. Por consiguiente, BPIMS-WS no provee algún soporte de diferentes lenguajes de consulta como SQL y OQL.

BPIMS-WS almacena en memoria las interacciones involucradas en una actividad comercial. Para cada actividad comercial, BPIMS-WS almacena en memoria a los participantes, los datos y los eventos participantes. Sin embargo, en caso de que BPIMS-WS pudiera utilizar un dispositivo persistente, podría examinarse la carga de tráfico de los mensajes y modificaciones de los contenidos de los mensajes en un periodo para poder restaurar la información o para realizar un análisis para una toma de decisión.

Recientes trabajos proponen otros mecanismos más sofisticados para la supervisión de procesos de negocio. Jeng et. al. [129] propone una arquitectura basada en agentes que soporta la supervisión, interpretación, predicción, automatización y respuesta de procesos de negocio, disminuyendo el tiempo en la toma de decisiones comerciales. Esta arquitectura habilita el análisis para procesos de negocio corporativos a través de notificaciones, recomendaciones y la activación automática de operaciones comerciales. La arquitectura presenta cinco componentes principales: (1) Agentes inteligentes de negocio los cuales realizan el procesamiento analítico, (2) El contenedor de procesamiento de evento en donde se almacenan los eventos comerciales que representan procesos de negocio, (3) Fábrica de información de procesos el cual almacena las métricas de los procesos de negocio, (4) Un administrador de políticas el cual inspecciona y entrega eventos a los agentes que están interesados en los eventos que tienen relación con una política específica y (5) Una pizarra para la visualización de las métricas de los procesos de negocio y los resultados analíticos. Bajo esta arquitectura, la supervisión de los procesos de negocio se realiza por el conjunto de agentes que provee la arquitectura. Estos agentes proveen: (1) mecanismos de

reactividad para responder a situaciones y excepciones en un escenario comercial; (2) mecanismos deliberativos para ejecutar tareas que requieren razonamiento y más procesamiento de cómputo; (3) mecanismos reflectivos para adaptar y extender la información.

Schiefer et. al. [130] propone otra arquitectura para la supervisión de proceso comerciales basado en un almacenamiento de los datos para la toma de decisiones tácticas y operacionales proporcionando el acceso en tiempo real a indicadores de rendimiento de procesos de negocio para mejorar la velocidad y la efectividad de los flujos de trabajo. Para esto se presenta una arquitectura la cual provee los siguientes mecanismos: (1) Propagación de datos en tiempo real para proporcionar información exacta y detallada de las situaciones actuales de los procesos de negocio; (2) Agregación de contexto a los procesos de negocio para calcular las métricas de los flujos de trabajo y; (3) Respuestas automatizadas con retroalimentación directa o indirecta para la evaluación de las métricas de los flujos de trabajo y para la activación de las operaciones comerciales.

En comparación con los trabajos de Jeng [129] y Schiefer [130], BPIMS-WS utiliza servicios Web y no agentes inteligentes para representar y realizar un proceso comercial. De igual forma, BPIMS-WS no provee acceso en tiempo real a indicadores de rendimiento de procesos de negocio para mejorar la velocidad y la efectividad de los procesos de negocio. BPIMS-WS solo provee el acceso, más no el análisis y/o modificación, de la información que se está intercambiando en un proceso comercial. Sin embrago, mediante la combinación de tecnologías de agentes y el uso de un dispositivo persistente, BPIMS-WS podría realizar algunas actividades como la identificación de eventos inusuales y su correspondiente reacción a la ocurrencia de excepciones que pudieran suceder en los flujos de trabajos. Estas actividades podrían realizarse utilizando el repositorio de publicación/suscripción que BPIMS-WS posee con el fin de notificar y/o adaptar el estado actual de una actividad comercial. Finalmente, registrando cada tarea, acontecimiento y proceso, el repositorio podría utilizarse para verificar las etapas del proceso o para analizar procesos de cara a su optimización y modelado.

6.3. Resumen

La supervisión de procesos de negocio es un factor importante en la cadena de suministro. A través del proceso de supervisión, los socios involucrados en un proceso de negocio pueden dar seguimiento y verificar el desarrollo de las actividades comerciales con el fin de identificar problemas y oportunidades, y así puedan manejar respuestas apropiadas asegurando ganancias significativas y ventajas competitivas. En este capítulo se describe el proceso de supervisión de procesos de negocio descritos como servicios Web en BPIMS-WS. BPIMS-WS da seguimiento a un proceso de negocio de manera visual, permitiendo no solo a los programadores de servicios Web depurar sus aplicaciones, sino también a los usuarios verificar el desarrollo de los servicios. Para esto, BPIMS-WS intercepta los mensajes SOAP que son intercambiados por los servicios Web, los cuales se almacenan en un búfer. A partir de esto, BPIMS-WS es capaz de generar un diagrama UML de secuencia haciendo un mapeo de los mensajes SOAP al diagrama ya que cuenta con toda la información requerida para este fin: los identificadores de los participantes, su dirección, el orden relativo en que ocurrieron los mensajes y el contenido de ellos. Con esto, se puede observar gráficamente la transmisión de los mensajes SOAP entre las entidades involucradas. Finalmente, se revisan los trabajos relacionados en la supervisión de procesos de negocio y se discuten sus ventajas y desventajas en comparación con éstos.

Capítulo 7
Administración de Servicios Web en BPIMS-WS

La cadena de suministro es un ambiente complejo debido a que frecuentemente se incluyen recursos de múltiples proveedores y plataformas, por lo que comprender el status de un recurso, es comprender sólo una pequeña parte del ambiente. El hecho es que la complejidad de administrar recursos crece con el número y tipo de recursos desplegados, por lo que las compañías requieren de herramientas que soporten y automaticen sus esfuerzos de administración. En este capítulo se presenta el mecanismo empleado por BPIMS-WS para poder llevar a cabo la administración de procesos de negocio descritos como servicios Web. Así también, se revisan los trabajos relacionados y se discuten sus ventajas y desventajas en comparación con éstos.

7.1. Administración de Servicios Web

La complejidad de administrar servicios crece con el número y tipo de servicios desplegados en BPIMS-WS, por lo que se requirió el desarrollo de una herramienta que soportara y automatizara los esfuerzos de administración. En la Fig. 7.1 se muestra la interfaz gráfica donde BPIMS-WS despliega los recursos como servicios Web. BPIMS-WS utiliza un *JMX Bridge* que actúa como un puente entre los recursos administrados por JMX y los servicios Web para administrar los servicios Web. Para esto, BPIMS-WS proporciona una interfaz de servicio Web a un recurso administrado. Bajo este enfoque, los recursos pueden implementarse en diferentes tecnologías ya que sólo es necesario definir una interfaz de servicio Web para un recurso. Para lograr esto, se utilizaron *MBeans* para

representar un recurso que puede administrarse. El *JMX Bridge* genera: (1) documentos WSDL que describen el servicio Web; (2) una clase de Java que actúa como la implementación del servicio Web y; (3) un descriptor de despliegue para el servidor de aplicaciones Tomcat. Luego de esto, la clase de Java resultante se compila y despliega como un servicio Web en el servidor de aplicaciones Tomcat.

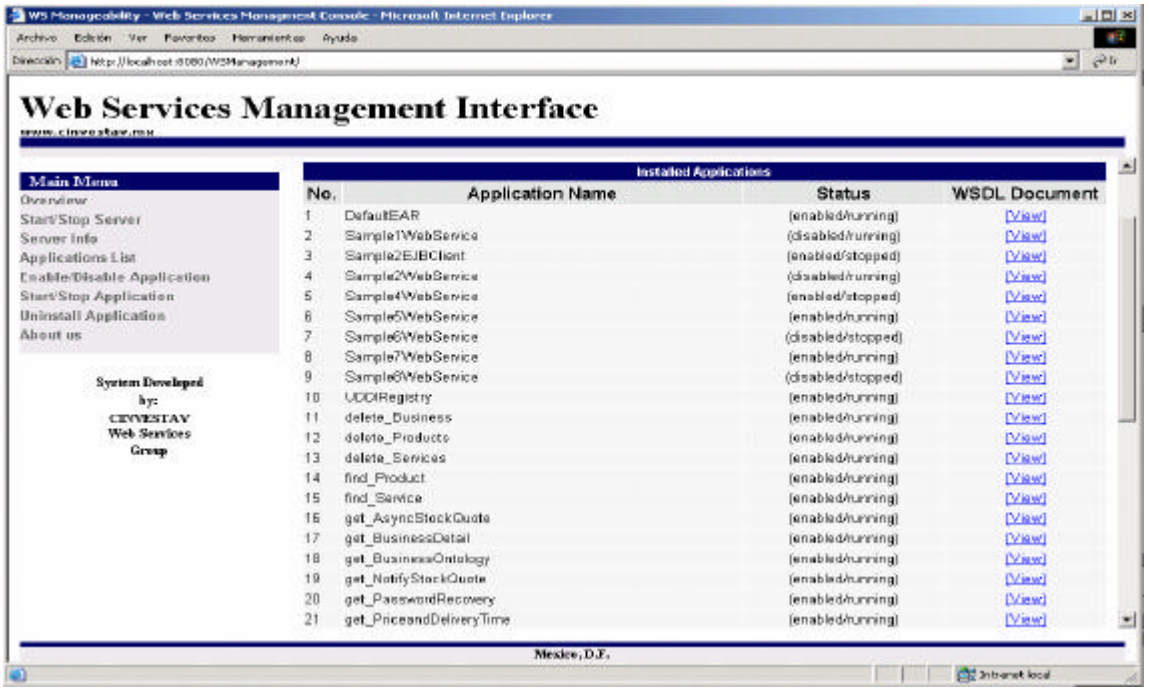


Fig. 7.1 Recursos desplegados por el servidor JMX MBean en BPIMS-WS

BPIMS-WS rastrea información global y estadística sobre los servicios Web. BPIMS-WS colecciona información la cual incluye:

- El nombre del servidor de aplicaciones
- La versión del servidor de aplicaciones
- El *hostname* del servidor
- El numero de puerto HTTP que está utilizando el servidor de aplicaciones
- El número total de servicios administrados que se desplegaron en el servidor de aplicaciones
- El número total de llamadas RPC a todos los servicios administrados en forma conjunta

BPIMS-WS rastrea y recupera esta información global desde páginas Web con el propósito de supervisar y administrar los servicios Web de forma remota.

Finalmente, en BPIMS-WS se implementó un administrador genérico el cual se añadió al servidor de aplicaciones Tomcat con el fin de reunir datos estadísticos cuando los servicios Web se invocan. Este administrador invoca a un *proxy* de administración que almacena o actualiza los datos estadísticos en *MBeans* los cuales se concretizan y administran a través del servidor *JMX MBean*. En la Fig. 7.2 se muestra la interfaz gráfica en donde se observa la información estadística que despliega BPIM-WS cuando los servicios Web se invocan.

Para cada servicio Web invocado, BPIMS-WS muestra la siguiente información estadística:

- El estado actual
- El número total de invocaciones exitosas
- El número de invocaciones fallidas (solicitudes recibidas por el servidor pero que producen una excepción)
- Promedio del tiempo de respuesta/transacción de las solicitudes exitosas

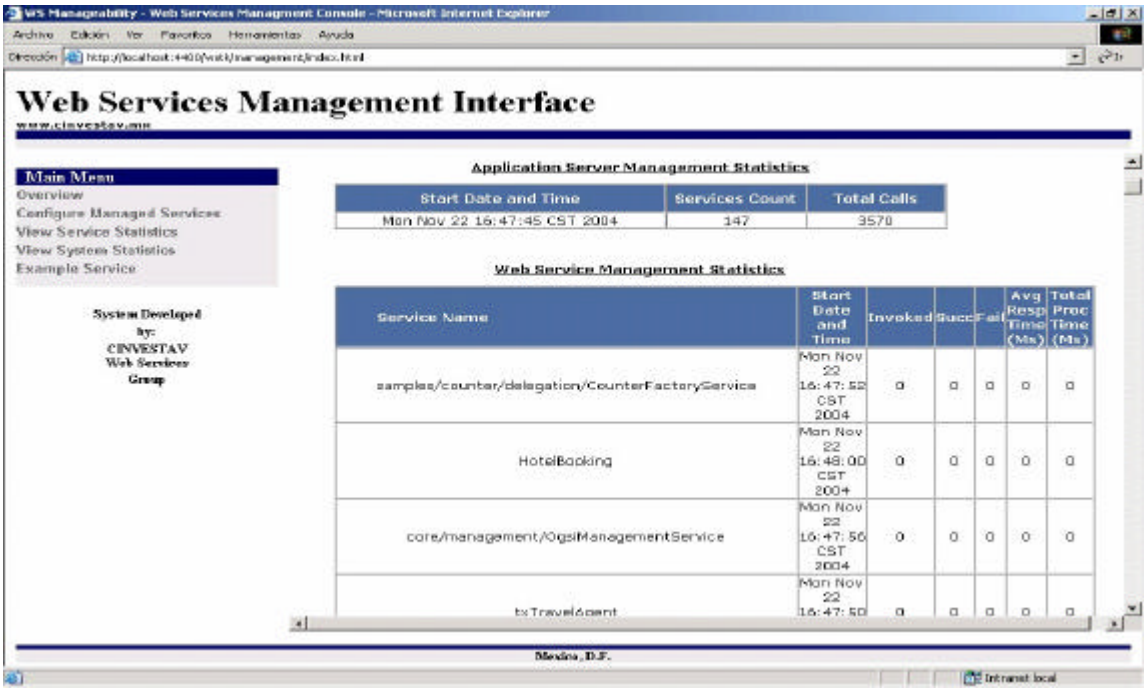


Fig. 7.2. Información estadística desplegada por el servidor JMX MBean en BPIMS-WS

7.2. Trabajos Relacionados con BPIMS-WS en la administración de servicios Web

Hoy en día, el desarrollo y la investigación en servicios Web se enfoca en las infraestructuras de *middleware* básicas. Sin embargo, una vez que la tecnología de servicios Web se consolide y consecuentemente un número importante de servicios Web estén disponibles, la atención se enfocará en el despliegue de servicios para su correcta administración. El hecho es que la complejidad de administrar servicios crece con el numero y tipo de servicios desplegados, por lo que las compañías requieren de herramientas que soporten y automaticen sus esfuerzos de administración. El interés en esta área ha permitido la publicación de algunas propuestas de estandarización y el desarrollo de trabajos de investigación. Dentro de los primeros trabajos, Jennings et al. [131] propone una infraestructura basada en agentes llamada ADEPT (*Advanced Decision Environment for Process Tasks*) para la administración de procesos de negocio. En el sistema ADEPT se propone un enfoque para estructurar el diseño y desarrollo de sistemas de administración de procesos de negocio. El enfoque consiste en utilizar tecnología de agentes para la negociación, aprovisionamiento de servicios, acuerdos a nivel de servicios, administración de recursos y compartición de la información. En este sentido, los procesos de negocio se asocian con uno o más agentes los cuales son responsables de administrarlos y ejecutarlos. Los agentes contienen módulos para el enrutamiento de mensajes entre el agente y su agencia y entre agentes, para aprovisionar servicios a través de negociación, y para evaluar y supervisar la habilidad del agente de encontrar los niveles de servicios apropiados. La principal aportación del sistema ADEPT es que la responsabilidad para promulgar diversos componentes del proceso comercial se delega a agentes que resuelven varios problemas en forma autónoma. Estos agentes actúan recíprocamente y negocian con otros agentes para coordinar sus acciones.

Yan et. al. [132] combina las capacidades de la tecnología de agentes y flujos de trabajo para la administración de procesos de negocio. Para esto, Yan identifica tres fases en el ciclo de vida de las administración de procesos de negocio: (1) creación; (2) aprovisionamiento y; (3) la puesta en marcha. La fase de creación involucra el análisis y la definición de los procesos de negocio, los cuales pueden representarse como un diagrama o

un formato verbal, para representar la lógica de las actividades en los procesos de negocio. La fase de aprovisionamiento involucra la asignación de recursos, incluyendo personas, equipo y tiempo de procesamiento. En esta fase se requiere negociación, planeación y calendarización para asegurar que existen los suficientes recursos para un proceso de negocio. Esta fase se realiza en tiempo de ejecución. La fase de puesta en marcha considera la administración de actividades requeridas para asegurar que cada instancia de un proceso de negocio se ejecuta de forma efectiva. En esta fase se incluye el enrutamiento de trabajo, paso de la información, activación automática de actividades y el manejo de listas de trabajo. También, esta fase se realiza en tiempo de ejecución. En estas tres fases del ciclo de vida de la administración de procesos de negocio, Yan considera que la tecnología de flujos de trabajo es apta para el estado de la puesta en marcha debido a que se proveen mecanismos de control y supervisión de las ejecuciones de los procesos de negocio. Además, se proveen herramientas para definir procesos de negocio como plantillas de flujos de trabajo. Así también, Yan argumenta que ésta tecnología no provee un buen soporte para la fase de aprovisionamiento. Para esto, Yan considera que la tecnología basada en agentes ofrece un buen soporte debido a que la negociación, la calendarización y la asignación de recursos son comunes de los sistemas multi-agentes. Así también, Yan argumenta que incluso esta tecnología podría considerarse para la fase de puesta en marcha ene el contexto de proveer mecanismos de automatización de actividades. En este sentido, los agentes pueden representar elementos de trabajo y a través de negociación decidir la ruta que estos elementos seguirán. Incluso, si algún proceso de negocio sufre modificaciones, los agentes pueden reaccionar ante esto. Finalmente, para la fase de creación, los agentes pueden crear nuevos procesos de negocio a través de negociación.

En comparación con los trabajos de Jennings [131] y Yan [132], BPIMS-WS no utiliza la tecnología de agentes para llevar a cabo la administración de los procesos de negocio ya que como se ha dicho, los procesos de negocio se describen mediante servicios Web. En comparación con Jennings [131], BPIMS-WS en cierta forma realiza el aprovisionamiento de servicios a través de su nodo UDDI, la administración de servicios a través de un servidor JMX y la compartición de la información a través de las invocaciones de los diferentes servicios provistos. Sin embargo, BPIMS-WS no provee mecanismos para realizar acuerdos a nivel de servicios. En comparación con Yan [132], BPIMS-WS

también provee algunas fases de administración. Sin embargo, estas fases no se refieren al ciclo de vida de la administración de los procesos de negocio sino al ciclo de vida de éstos. En BPIMS-WS, se identifican 4 fases del ciclo de vida de los procesos: (1) Activo, cuando el proceso se encuentra en ejecución; (2) Inactivo, que indica que el proceso no puede realizar alguna actividad; (3) Disponible, que el proceso se encuentra habilitado para una posible ejecución y; (4) No Disponible, que el proceso no tiene se encuentra habilitado más no inactivo para realizar alguna actividad.

En trabajos más recientes, Zhao et. al. [133] propone una arquitectura basada en servicios Web para sistemas de administración de flujos de trabajo con un énfasis en la interoperabilidad, flexibilidad, confiabilidad y escalabilidad a nivel inter-organizacional. Bajo esta arquitectura se propone tres componentes principales: (1) un servicio de administración basado en acuerdos; (2) un conjunto de servicio de datos de flujos de trabajo y; (3) un servicio de biblioteca de plantillas para la definición de las colaboraciones de los procesos de negocio entre las organizaciones participantes, incluyendo esquemas organizacionales horizontales y verticales. El servicio de biblioteca para la definición de los flujos de trabajo es responsable de almacenar las plantillas y las definiciones de los flujos de trabajo. El servicio de datos de flujos de trabajos incluye servicios para el control de datos y la administración de información relevante de los flujos de trabajo. La funcionalidad de estos servicios es garantizar la integridad y validez de los datos, especialmente en un proceso de migración. También, este conjunto de servicios provee mecanismos de respaldo y replicación de datos con el fin de habilitar la recuperación hacia atrás para propósitos de confiabilidad. El servicio de administración basado en acuerdos es responsable de la comunicación inter-organizacional. Esto involucra conversiones de datos, verificación de versión, verificación de socios comerciales, entre otros.

Martin-Flatin et. al. [134] propone un prototipo de código abierto de una plataforma de administración llamada JAMAP para la administración de servicios Web. JAMAP presenta una arquitectura de administración integrada basada en Web (WIMA) la cual está implementada en Java. En la arquitectura WIMA, los agentes publican la supervisión de los datos y notificaciones, y las aplicaciones de administración (administradores) se suscriben a ellos en una forma automatizada. En WIMA, una organización es dividida en múltiples dominios de administración los cuales se definen en base a criterios como: localización

geográfica, área de administración, cliente, entre otros. De esta forma, cada dominio los administra un conjunto de agentes. La arquitectura de WIMA presenta cinco componentes principales: (1) analizador de datos el cual analiza y supervisa los datos al vuelo y detecta problemas enviándolos al correlacionador; (2) colector de datos el cual colecta y filtra los datos para su supervisión; (3) el colector de notificaciones se encarga de recibir los eventos entrantes, filtrarlos y reenviarlos al correlacionador de eventos, los eventos se archivan en un repositorio de datos; (4) administrador de configuración que se encarga de la configuración de los agentes; (5) analizador del background el cual ejecuta minería de datos y análisis de los datos que se almacenan en ele repositorio de datos.

Jeng et. al. [135] propone un marco de trabajo basado en políticas para la administración de actividades comerciales. El marco de trabajo presenta tres entidades principales: (1) políticas de agentes de administración las cuales definen que actividades debe realizar un agente las cuales se activan mediante eventos comerciales y se interpretan por una agente; (2) políticas de unidades virtuales de negocio definen las restricciones y obligaciones que una entidad de negocio necesita obedecer ya sea sobre aspectos de autorización o políticas de seguridad; (3) políticas de habilitadores de administración definen el acceso, control y la supervisión sobre las entidades virtuales de negocio. Un administrador de políticas administra el ciclo de vida de las políticas de administración proveyendo acceso a: (1) Un repositorio de modelos BAM el cual almacena los meta modelos de las actividades administradas incluyendo información como la semántica y las interfaces de invocación; (2) Un registro de ejecución BAM que almacena los valores en tiempo real de todas las políticas como son restricciones, acuerdos o compromisos.

Shegalov et. al. [136] propone una arquitectura para la administración de flujos de trabajos distribuidos. Esta arquitectura se implementó sobre un prototipo llamado Mentor-lite el cual es un mediador basado en XML que se encarga del intercambio de mensajes, el flujo de datos de control entre los flujos de trabajo y servidores y de la actividades del cliente encapsuladas en los mensajes XML sobre HTTP.

En comparación con los trabajos de Zhao [133], Martin-Flatin [134], Jeng [135] y Shegalov [136], BPIMS-WS realiza la administración de servicios Web. Sin embargo, no utiliza políticas para la administración ni provee múltiples dominios de administración y mucho

menos se hace énfasis en la interoperabilidad, flexibilidad, confiabilidad y escalabilidad. El enfoque de BPIMS-WS es muy simple: la recopilación de datos estadísticos.

Por otra parte, diferentes consorcios han definido algunas propuestas de estandarización en esta área. Bullen et. al. [137] propone una interfaz abierta de administración (OMI, *Open Management Interface*) como una manera fácil y abierta para acceder y administrar los recursos y sus procesos de negocio asociados en una plataforma de integración. OMI es una interfaz genérica y extensible la cual es accesible como un servicio Web. A través de esta interfaz, los consumidores de servicios pueden manipular un conjunto de objetos OMI administrados que representan los recursos disponibles. La especificación OMI también define un conjunto de atributos estándares, operaciones y notificaciones para cada tipo de recurso OMI administrado y también un conjunto de relaciones que pueden existir entre los objetos OMI administrados. OASIS (*Organization for the Advancement of Structured Information Standards*) propone a la especificación WSDM (*Web Services Distributed Management*) para la integración de recursos heterogéneos en software de administración heterogéneo utilizando protocolos y plataformas heterogéneas [138]. WSDM utiliza servicios Web para solucionar el problema de integración de administración. WSDM consiste de dos especificaciones: la administración mediante servicios Web (MUWS, *Management Using Web Services*) y la administración de servicios Web (MOWS, *Management Of Web Services*). WSDM MUWS define cómo representar y acceder las interfaces de manejabilidad de recursos como servicios Web. También define un conjunto básico de capacidades de manejabilidad, como la identidad del recurso, métrica, configuración, y relaciones que pueden formarse para expresar la capacidad de instrumentación de la administración. Además, WSDM MUWS proporciona un formato estándar para la administración de eventos para mejorar la interoperabilidad y correlación. WSDM MOWS define cómo administrar los servicios Web vistos como recursos y cómo describir y acceder la manejabilidad utilizando MUWS. Además, MOWS proporciona los mecanismos y metodologías que habilitan el desarrollo de aplicaciones de servicios Web administrados para interoperar dentro y fuera de los límites de una empresa.

BPIMS-WS utiliza la especificación *WS-Manageability* para administrar los servicios Web. La especificación permite la administración de recursos locales y remotos a través de servicios Web. En este contexto, *WS-Manageability* es una especificación similar a OMI y

WSDM. Pero, BPIMS-WS no realiza administración a través de servicios Web sino que realiza la administración de los servicios en sí mismos.

7.3. Resumen

En la cadena de suministro, la complejidad de administrar servicios crece con el número y tipo de servicios involucrados por lo que las compañías requieren de herramientas que soporten y automaticen sus esfuerzos de administración. En este capítulo se presenta el mecanismo empleado por BPIMS-WS para poder llevar a cabo la administración de procesos de negocio descritos como servicios Web. Dicho mecanismo consiste en utilizar *MBeans* para representar un recurso que puede administrarse. Para esto, se desarrolló un *JMX Bridge* que actúa como un puente entre los recursos administrados por JMX y los servicios Web para administrarlos, por lo que BPIMS-WS proporciona una interfaz de servicio Web a un recurso administrado. Bajo este enfoque, los recursos pueden implementarse en diferentes tecnologías ya que sólo es necesario definir una interfaz de servicio Web para un recurso. Finalmente, se revisan los trabajos relacionados en la administración de procesos de negocio descritos como servicios Web y se discuten sus ventajas y desventajas en comparación con éstos.

Capítulo 8
Casos de Estudio de BPIMS-WS

BPIMS-WS permite descubrir, integrar, orquestar, administrar y supervisar los procesos de negocio de diversas empresas descritas como servicios Web. En este capítulo se describen tres casos de estudio donde BPIMS-WS se utilizó para la integración de procesos de negocio en la cadena de suministro. El primer caso de estudio describe la integración de los procesos de negocio de un modelo de empresa virtual a través de BPIMS-WS. El segundo caso de estudio describe la funcionalidad de BPIMS-WS como un agente de compra distribuido. Finalmente, el último caso de estudio describe la funcionalidad de BPIMS-WS para el reaprovisionamiento oportuno en la cadena de suministro.

8.1. Integración de Procesos de negocio de un Modelo de Empresa Virtual a través de BPIMS-WS

Para describir la integración de los procesos de negocio registrados en BPIMS-WS, se propone el siguiente caso de estudio. El caso de estudio describe como BPIMS-WS facilita el descubrimiento de información de un modelo de empresa virtual llamada STD (Surte Tu Despensa) la cual ofrece productos de primera necesidad. Algunas interfaces gráficas de esta empresa se muestran en la Fig. 8.1.

Suponga el siguiente escenario:

1. Hay una empresa llamada STD la cual ofrece en línea productos de primera necesidad. Esta empresa registró sus productos y procesos de negocio como servicios Web en BPIMS-WS.
2. La empresa tiene soporte para los procesos de negocio descritos bajo la ontología de RosettaNet.

3. Hay diversos clientes potenciales que desean realizar solicitudes de compra de productos ofrecidos por STD.

En éste escenario, ¿cómo los clientes pueden buscar y encontrar los servicios Web que ofrece STD para comprar productos?

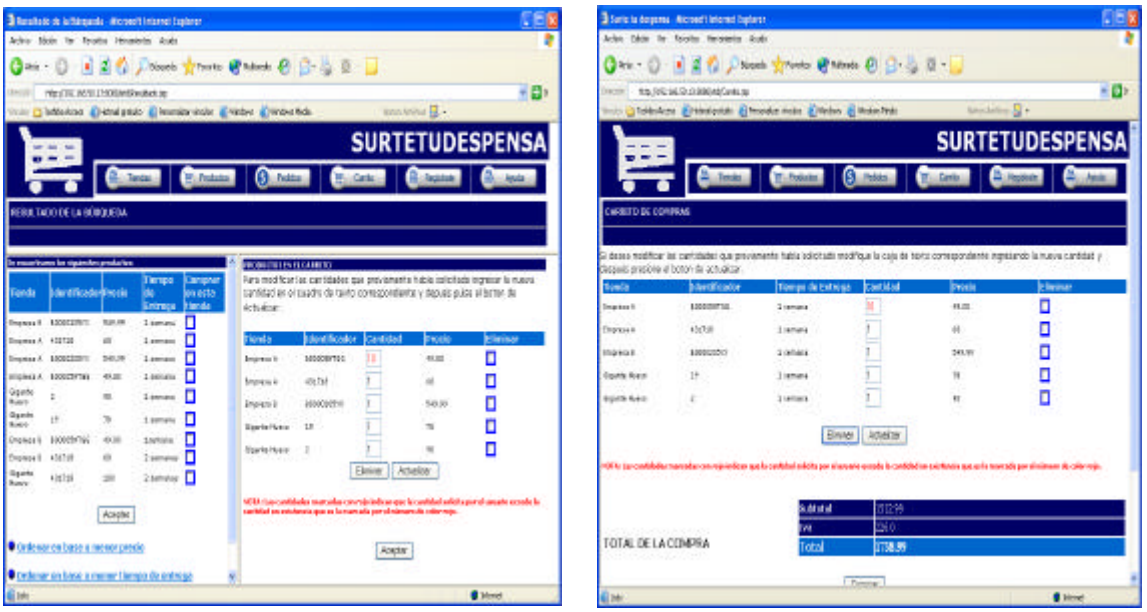


Fig. 8.1. Interfaces gráficas de la empresa STD

En la modalidad de portal de Internet, BPIMS-WS tiene en su menú principal una opción llamada "Comprar". En esta opción, BPIMS-WS despliega una interfaz gráfica donde los clientes pueden realizar la búsqueda de productos de las empresas registradas. Esta interfaz gráfica muestra las diversas ontologías de productos que BPIMS-WS soporta. En la Fig. 8.2 se muestra la interfaz gráfica de la selección de ontología y producto. Una vez seleccionada la ontología y el producto, BPIMS-WS despliega otra interfaz gráfica mostrando los diferentes criterios de búsqueda para la compra de un producto. Entre estos criterios de búsqueda están: Mostrar Todos los Proveedores, Precio más Bajo, Precio más Bajo y Cantidad, y Mínimo Tiempo de Entrega. En la Fig. 8.3 muestra la interfaz gráfica de los diferentes criterios de búsqueda para la compra de un producto.

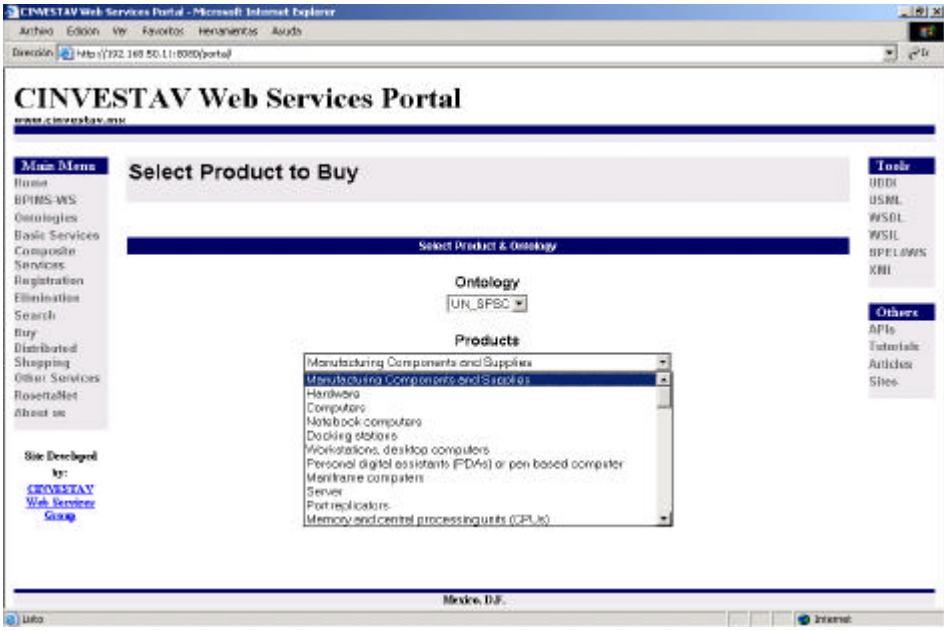


Fig. 8.2. Interfaz gráfica de la selección del producto y la ontología en BPIMS-WS

Fig. 8.3. Interfaz gráfica de los criterios de búsqueda para la compra de un producto en BPIMS-WS

Suponga que el cliente selecciona el criterio de búsqueda “Mostrar Todos los Proveedores”. Luego, BPIMS-WS formula una consulta al nodo UDDI para encontrar todos los proveedores candidatos del producto. Para cada proveedor candidato, BPIMS-WS formula otra consulta al nodo UDDI para encontrar el URL donde se encuentra la especificación del servicio Web correspondiente al PIP 3A2 (*Request Price and Availability*) de RosettaNet. Una vez localizado el URL, BPIMS-WS analiza la especificación del servicio y construye una solicitud para cada proveedor encontrado con el fin de invocar su correspondiente servicio Web. Para esto, BPIMS-WS envía a cada proveedor los mensajes SOAP que contienen las solicitudes de invocación y obtiene las respectivas respuestas. BPIMS-WS analiza cada respuesta y extrae la información útil para construir una nueva respuesta. El resultado de esta nueva respuesta es una lista de todos los proveedores del producto más información adicional del producto como el precio y tiempo de entrega. En la Fig. 8.4 se muestra el resultado final de invocar los PIP 3A2 de cada proveedor candidato. Aquí, la empresa STD aparece como un proveedor del producto. Luego, suponga que el cliente desea comprar el producto en la empresa STD debido a que ofrece el precio mas bajo o porque ofrece el menor tiempo de entrega.

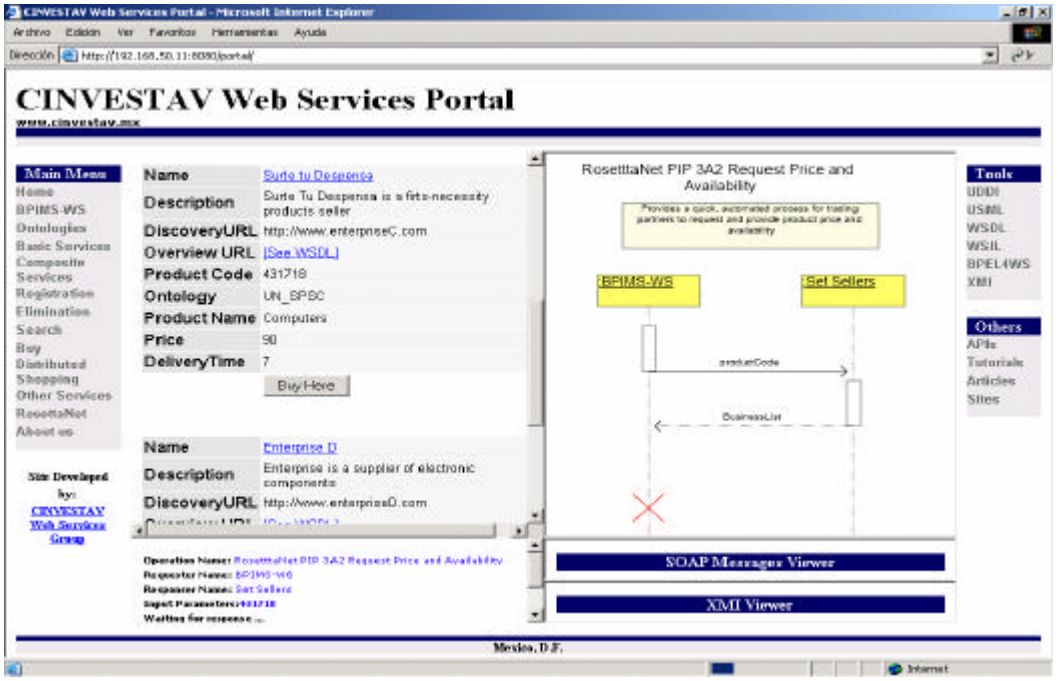


Fig. 8.4. Interfaz gráfica del resultado de invocar los PIP 3A2 de los proveedores

Una vez hecho esto, BPIMS-WS formula una nueva consulta al nodo UDDI para encontrar el URL del servicio Web que corresponde al PIP 3A1 (*Request Quote*) de la empresa STD. Una vez localizado el URL, BPIMS-WS analiza la especificación del servicio y construye una solicitud para invocar el servicio. El resultado de esa invocación es la cantidad en existencia del producto que tiene STD en sus almacenes. En la Fig. 8.5 muestra la respuesta a la invocación del PIP 3A1 de STD.

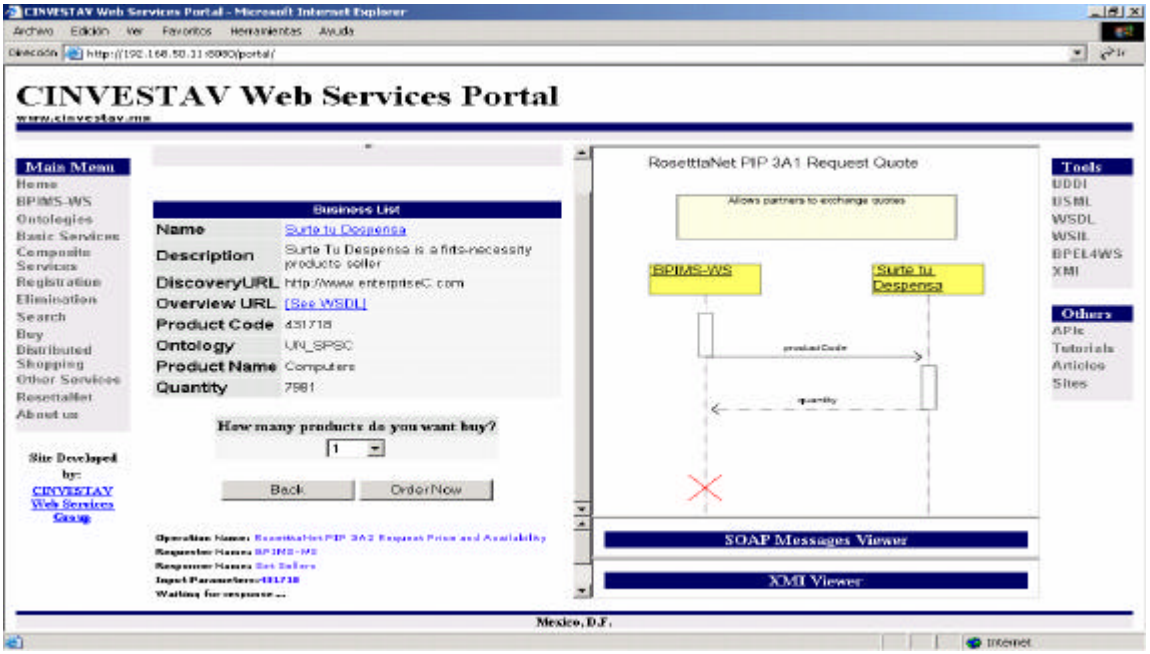


Fig. 8.5. Interfaz gráfica del resultado de invocar el PIP 3A1 de STD

En este punto, el cliente debe seleccionar la cantidad del producto que desea comprar a STD. Una vez seleccionada la cantidad, BPIMS-WS formula una consulta al nodo UDDI para encontrar el URL del servicio Web de STD que corresponde al PIP 3A4 (*Request Order*) de RosettaNet. Una vez localizado, BPIMS-WS analiza la especificación del servicio Web y despliega una interfaz gráfica del servicio, tal como se muestra en la Fig. 8.6. En este punto, el cliente debe llenar la información correspondiente al servicio Web para realizar el proceso de compra. Una vez hecho esto, BPIMS-WS construye una solicitud e invoca el servicio Web de STD. BPIMS-WS obtiene la respuesta a la invocación del servicio y despliega los resultados. La respuesta contiene un identificador de pedido del producto, como la muestra la Fig. 8.7

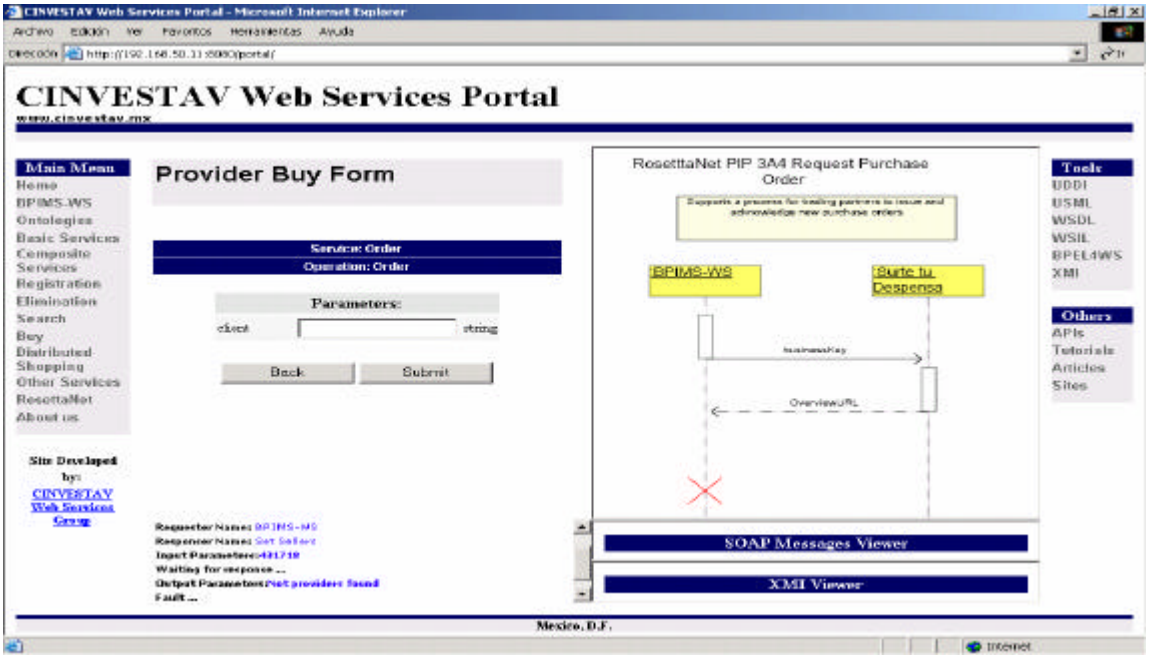


Fig. 8.6. Interfaz gráfica del resultado de analizar el servicio Web del PIP 3A4 de STD

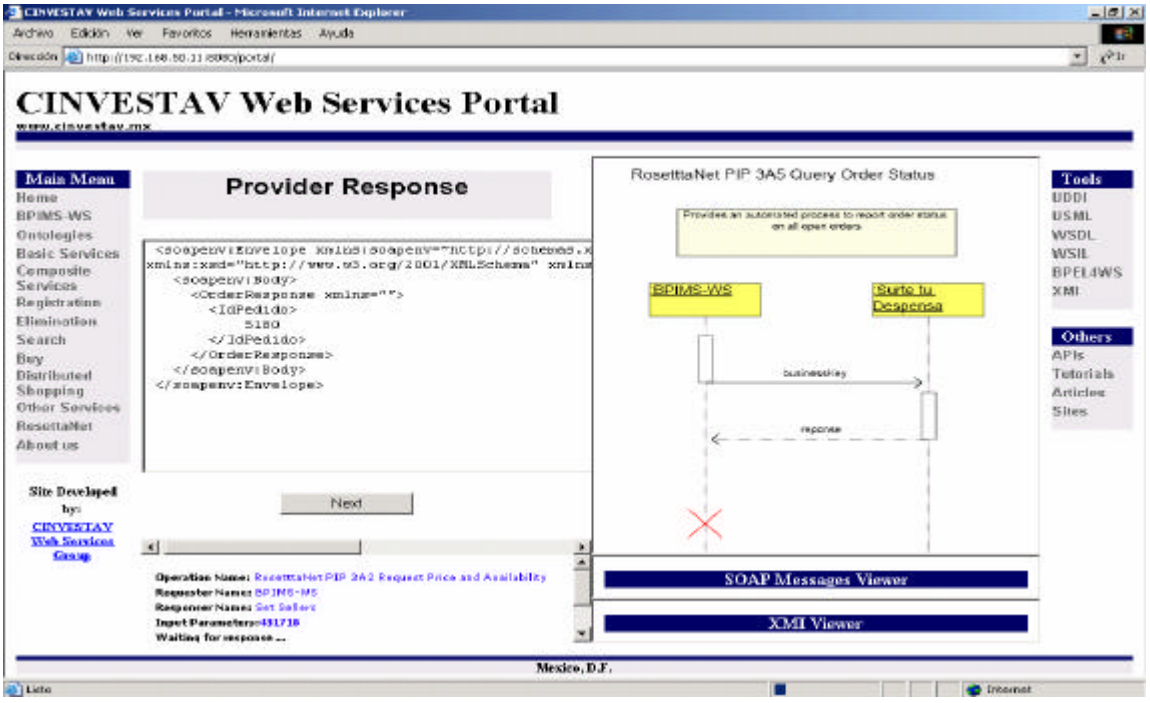


Fig. 8.7. Interfaz gráfica del resultado de invocar el PIP 3A4 de STD

Como se pudo observar en este caso de estudio, BPIMS-WS facilita el descubrimiento de servicios Web que representan los procesos de negocio de una empresa llamada STD que ofrece productos de primera necesidad, con el fin de que clientes potenciales puedan comprar productos a través de la modalidad de portal de Internet que ofrece BPIMS-WS.

8.2. BPIMS-WS como un agente de compra distribuida

El siguiente caso de estudio describe cómo BPIMS-WS facilita el descubrimiento, integración y composición de servicios Web ofrecidos por algunas empresas que venden componentes electrónicos para realizar compras distribuidas. Suponga el siguiente escenario:

1. Hay diversas empresas que venden componentes electrónicos en línea. Las empresas registraron sus productos y procesos de negocio como servicios Web en BPIMS-WS.
2. Las empresas tiene soporte de sus procesos de negocio bajo la ontología de RosettaNet.
3. Un cliente potencial (empresa) tiene una cadena de suministro y necesita realizar compras distribuidas de productos usando servicios Web.

En este escenario, ¿cómo el cliente puede integrar sus procesos de negocio para localizar e invocar los servicios Web disponible para comprar los productos en una forma distribuida?
Como ya se mencionó en capítulos anteriores, BPIMS-WS ofrece la modalidad de interacción como un portal de Internet. En este modo, hay la opción "Compras Distribuidas" en el menú principal de BPIMS-WS. En esta opción, BPIMS-WS despliega una interfaz gráfica dónde los clientes pueden seleccionar algunos productos registrados y sus cantidades respectivas que desean encontrar, como se muestra en la Fig. 8.8.
Una vez seleccionada la lista de productos con sus respectivas cantidades, BPIMS-WS despliega otra interfaz gráfica dónde el cliente debe seleccionar un criterio de ordenamiento. Esto criterio de ordenamiento indica la forma en que BPIMS-WS desplegará el resultado de la búsqueda.

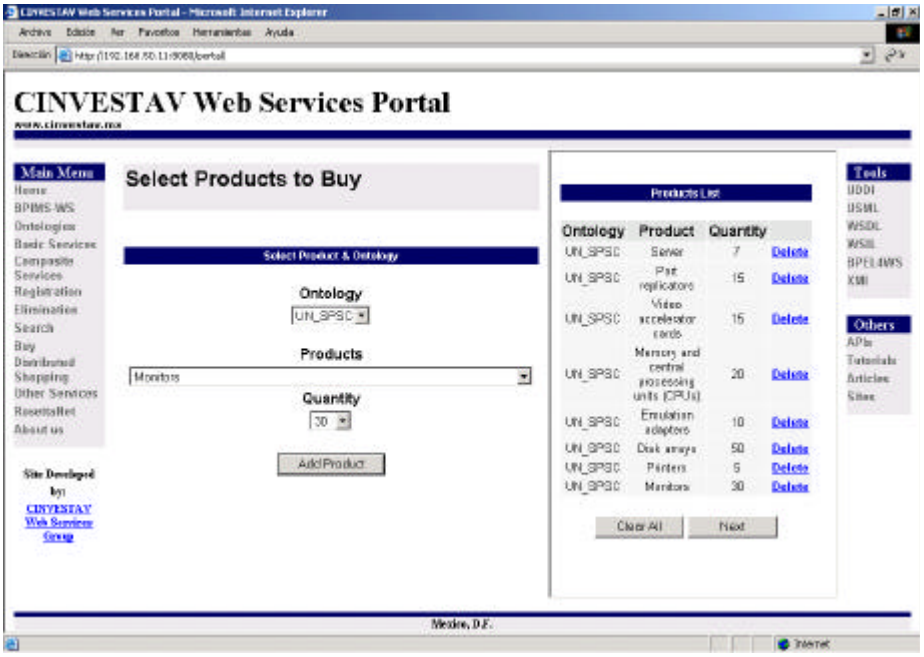


Fig.8.8 Interfaz Gráfica de selección de productos en BPIMS-WS

Entre los criterios disponibles están el precio más bajo, el mínimo tiempo de entrega y todas sus combinaciones posibles. Otros criterios como calidad del servicio y promociones u ofertas no se consideraron. La interfaz gráfica de usuario para seleccionar el criterio de ordenamiento se muestra en la figura 8.9. Luego, BPIMS-WS construye una solicitud para invocar el servicio Web *get_SortedProductsList*. El resultado de la invocación de este servicio Web devuelve una lista de proveedores del producto según el criterio de ordenamiento seleccionado. El resultado se muestra como un documento HTML. En este punto, una lista de empresas aparece como proveedores del producto, como lo muestra Fig. 8.10.

Posteriormente, el cliente selecciona un proveedor de esta lista para comprar diversos productos. Una vez seleccionado el proveedor, BPIMS-WS formula una consulta al nodo UDDI para localizar, obtener y analizar el URL del servicio Web correspondiente al PIP 3A4 de RosetaNet (Solicitud de Orden de Compra) del proveedor. Aquí, BPIMS-WS despliega una interfaz gráfica de usuario de la especificación del servicio Web, permitiendo la visualización de las actividades involucradas en el proceso de orden de compra. Entonces, el cliente debe proporcionar la información requerida para completar el proceso de compra. Esta interfaz gráfica es similar a la de la Fig. 8.6.

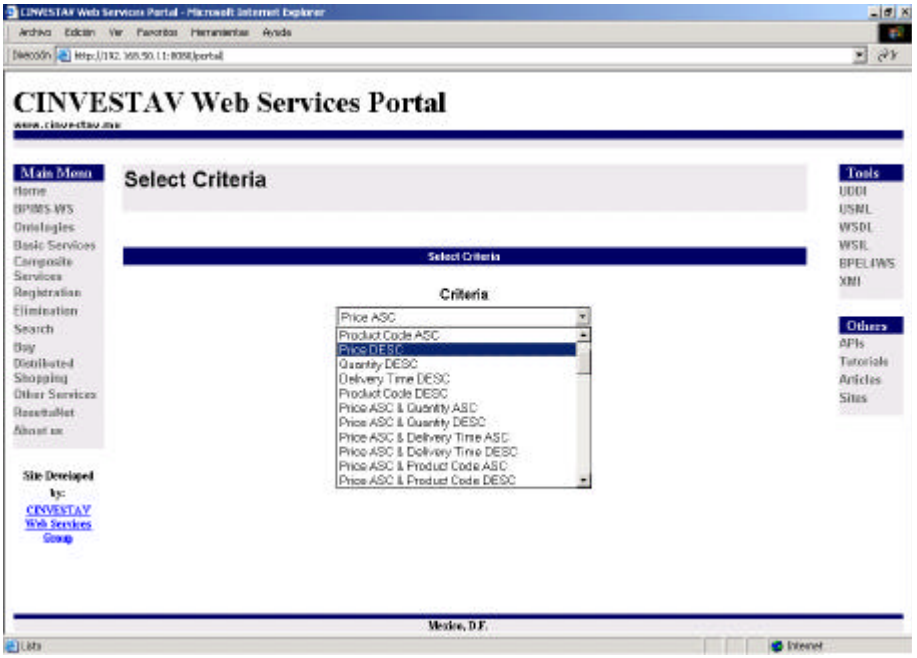


Fig. 8.9 Interfaz Gráfica del criterio de ordenamiento para la búsqueda de productos en BPIMS-WS

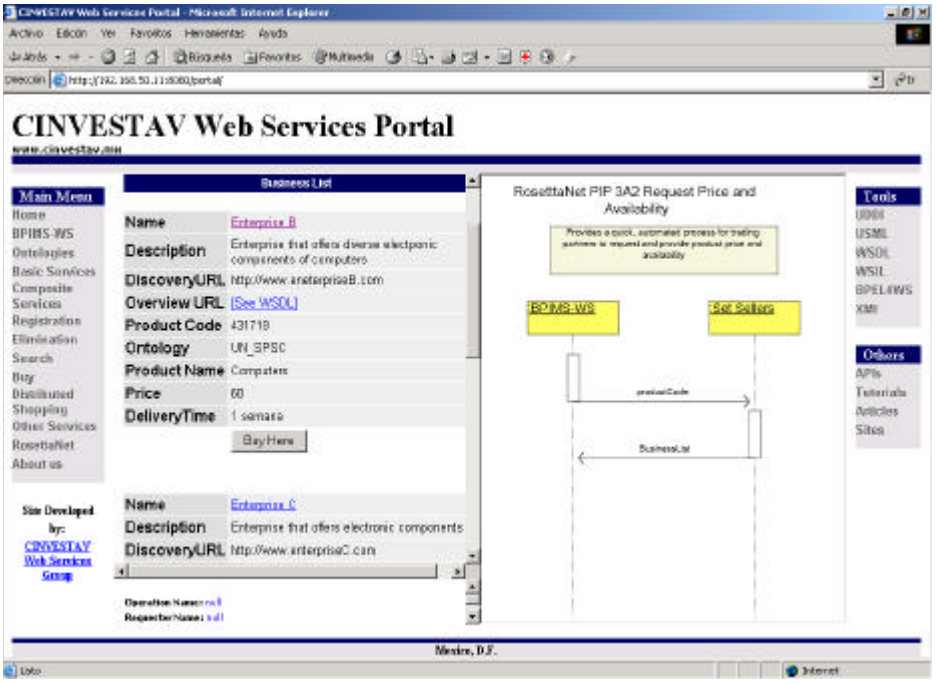


Fig. 8.10 Interfaz Gráfica que muestra el resultado de invocar el servicio Web get_SortedProductsList

Una vez que el cliente proporciona los datos, BPIMS-WS invoca el servicio Web correspondiente. Finalmente, BPIMS-WS despliega el identificador de pedido del producto como se muestra en la Fig. 8.7.

En este caso de estudio, sólo se muestra un ejemplo que ilustra la integración de los procesos de negocio en BPIMS-WS. Sin embargo, una variedad de otros casos de estudio que involucran diversos criterios de optimización se desarrollaron y probaron. Ejemplos de ellos son el mínimo tiempo de entrega, el precio más bajo, la cantidad exacta especificada, por mencionar solo algunos.

8.3. Procuración Oportuna usando BPIMS-WS

El último caso de estudio describe cómo BPIMS-WS facilita el descubrimiento de servicios Web que ofrecen diferentes empresas que venden componentes electrónicos.

Suponga el siguiente escenario:

1. Existen diversas empresas que venden componentes electrónicos, la cuales han registrado en BPIMS-WS sus productos y sus procesos de negocio descritos como servicios Web.
2. Un cliente potencial (empresa) inicia una cadena de suministro para procurar productos solicitando una orden de compra por medio de servicios Web.

En este escenario, cómo un cliente puede descubrir e invocar los servicios Web disponibles para llevar a cabo la procuración oportuna

Primeramente, se asume que los procesos de negocio registrados de las empresas en BPIMS-WS basan sus conductas comerciales descritas en PIPs de RosettaNet. Por otro lado, no se considera que todos los proveedores usan los mismos campos de datos para las descripciones de los productos y sigan el mismo protocolo para sus interacciones con los clientes. Bajo este escenario pueden observarse cuatro tipos de respuestas a través de BPIMS-WS:

1. La solicitud no se entiende. BPIMS-WS no puede procesar la solicitud del cliente porque no esta bien formulada. Algunos factores como la falta de parámetros o la

invocación de un servicio Web con un nombre incorrecto de operación producen una solicitud mal formulada. Bajo este caso, BPIMS-WS despliega un mensaje de error indicando que la solicitud no puede ser procesada.

2. No hay algún proveedor para el producto solicitado. BPIMS-WS devuelve como respuesta una lista vacía que indica que no se encontró algún proveedor que proporcione el producto. Esta situación se deriva principalmente de dos factores: (1) el producto solicitado no se encuentra registrado en BPIMS-WS y por consiguiente no existen proveedores disponibles; y (2) el producto solicitado en BPIMS-WS se encuentra registrado pero no está asociado con un proveedor.

3. Hay un proveedor para el producto solicitado. BPIMS-WS devuelve como respuesta una lista de las empresas que aparecen como proveedores del producto. Este es el mejor caso debido a que el producto solicitado se encuentra registrado y asociado con un proveedor. Bajo este caso, BPIMS-WS despliega una lista de proveedores que ofrecen el producto requerido en sus respectivos inventarios.

4. Hay un proveedor para el producto, pero el producto no está disponible en ese momento. En este caso, BPIMS-WS se compromete en obtener el producto solicitado de los proveedores registrados.

En este caso de estudio, sólo se considera el cuarto tipo de respuesta cuando se encuentran algunos proveedores pero ninguno de ellos tiene el suficiente número de productos en ese momento para cumplir la solicitud. En la figura 8.11 se muestra el escenario completo de procuración basada en el cuarto tipo de respuesta.

La solución propuesta se basa en el uso de un enfoque de patrones de procesos de negocio después de identificarse el modelo de suscripción basado en eventos que ocurre en el cuarto tipo de respuesta. El modelo de suscripción basado en eventos se usa para obtener una lista de proveedores que ofrecen el producto solicitado en cantidades suficientes. A continuación, se explica el modelo de suscripción basado en eventos:

- Primeramente, BPIMS-WS crea nuevas instancias tanto para el repositorio de servicios como el de suscripción (Pasos 1-4 en Fig. 8.11). Este paso es una fase de inicialización en la cual BPIMS-WS se encuentra listo para escuchar las solicitudes de los clientes.

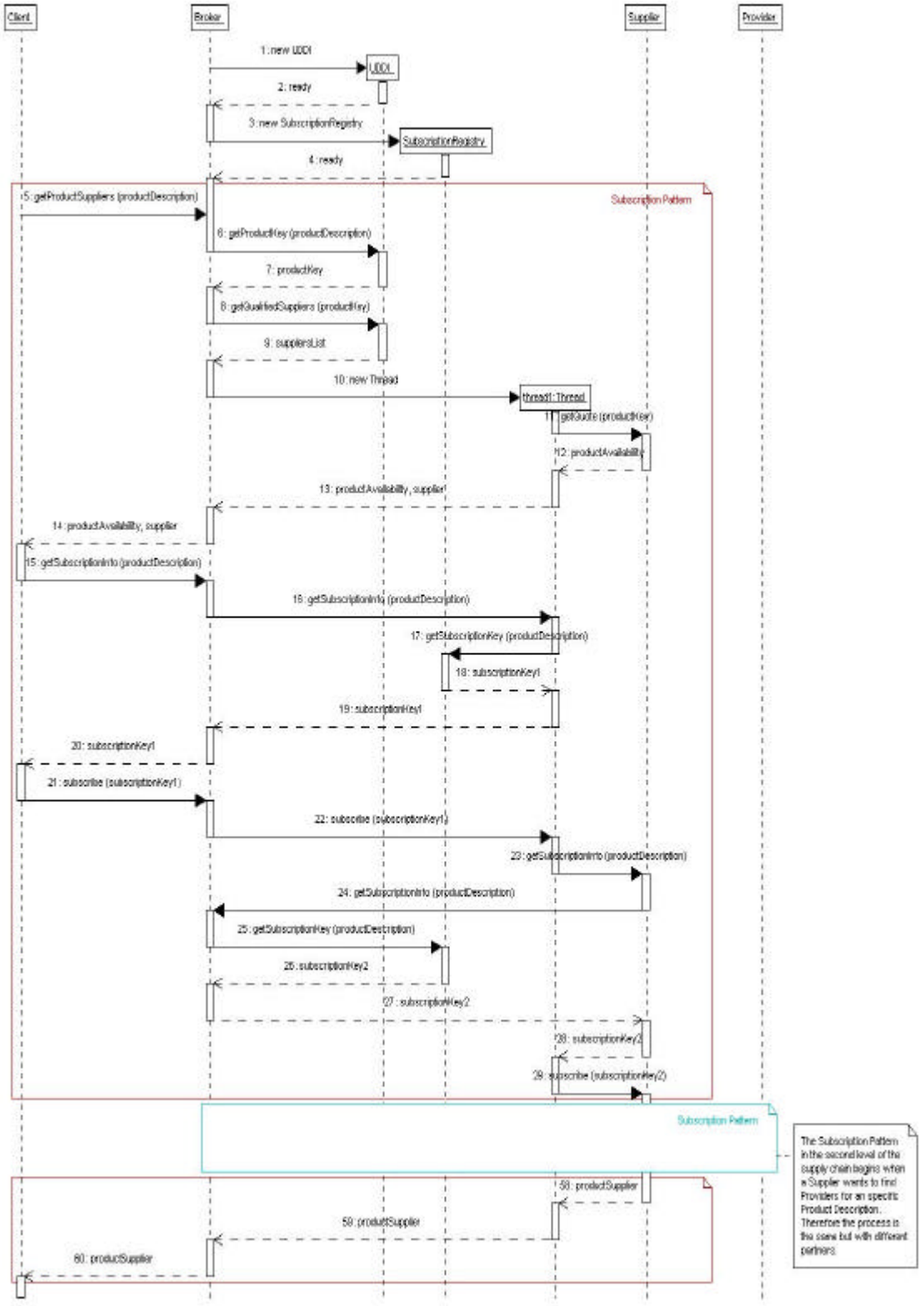


Fig. 8.11 Interacciones involucradas en el proceso de procuración

El cliente realiza una solicitud a BPIMS-WS enviando un mensaje SOAP que contiene la descripción del producto (Paso 5 en Fig. 8.11). Esta solicitud corresponde al PIP 3A5 de RosettaNet (*Query Technical Information*). BPIMS-WS extrae la descripción del producto y utiliza su servicio de descubrimiento. BPIMS-WS formula una consulta al nodo UDDI para conocer a los proveedores candidatos. Luego, BPIMS-WS transforma la descripción del producto a sentencias SQL y construye una consulta al repositorio de servicios. La consulta se ejecuta y la respuesta es un conjunto de documentos WSDL de las descripciones de los servicios de los proveedores candidatos (Pasos 6-9 en Fig. 8.11). Cada documento WSDL representa el PIP 2A5 de RosettaNet.

- Para cada documento WSDL, BPIMS-WS los analiza y recupera el punto de acceso, operaciones, mensajes y parámetros de entrada y salida. Entonces, BPIMS-WS identifica el tipo de protocolo de comunicación especificado en cada documento WSDL e invoca el servicio Web de cada proveedor candidato (Pasos 10-11 en Fig. 8.11)
- BPIMS-WS recibe la respuesta de cada proveedor (Pasos 12 y 13 en Fig. 8.11). Luego, determina cual de las cuatro tipos de mencionadas anteriormente se incluye en el mensaje.
- Si la contestación corresponda al cuarto caso, BPIMS-WS pregunta al cliente si desea subscribirse para un proveedor de producto cuando se encuentre disponible (Paso 14 en Fig. 8.11). Si el cliente acepta, BPIMS-WS utiliza su mecanismo de publicación/suscripción para publicar la descripción del producto solicitada por el cliente y genera un ticket (*subscriptionKey*) para el cliente (Pasos 15-20 en Fig. 8.11).
- Posteriormente, el cliente invoca el servicio de suscripción de BPIMS-WS enviando un mensaje SOAP que contiene el dato *subscriptionKey* y esperará por la respuesta (Paso 21 en Fig. 8.11). Durante la espera, BPIMS-WS pedirá al proveedor su *subscriptionKey* correspondiente enviando la descripción del producto solicitada por el cliente (Pasos 22 y 23 en Fig. 8.11). El proveedor guardará la descripción del producto y generará otro ticket (*subscriptionKey*) como una respuesta a la solicitud de BPIMS-WS (Pasos 24-26 en Fig. 8.11). Luego, BPIMS-WS invocará el servicio de suscripción del proveedor y esperará por la respuesta del proveedor (Pasos 27-29 en Fig. 8.11). Durante la espera de BPIMS-WS, el patrón de suscripción basado en eventos en el segundo nivel de la cadena de suministro empieza cuando un proveedor quiere encontrar otros proveedores

para una descripción del producto. Por consiguiente, el proceso es el mismo pero los socios comerciales toman papeles diferentes, es decir, ahora el proveedor actúa como un cliente y el proveedor del proveedor actúa como un proveedor para BPIMS-WS.

- Cuando BPIMS-WS recibe la respuesta del proveedor del proveedor en el segundo nivel de la cadena de suministro, analiza y recupera la información útil sobre el producto solicitado. Con esta información, BPIMS-WS puede enviar una respuesta a la solicitud del servicio de suscripción del proveedor. Aquí concluye el segundo nivel de la cadena de suministro. De una manera similar, el proveedor responderá al servicio de suscripción de BPIMS-WS y su espera terminará (Pasos 58 y 59 en Fig. 8.11).
- Finalmente, BPIMS-WS puede contestar a la solicitud de servicio de suscripción del y como consecuencia, la espera del cliente terminará y obtendrá la información sobre su producto pedido (Paso 60 en Fig. 8.11).

En este punto, el cliente puede seleccionar la cantidad del producto solicitado y puede realizar la compra de productos de manera similar como el primer caso de estudio.
En este caso de estudio se revelan las complejidades que BPIMS-WS puede manejar, tal como no sucede en los anteriores dos casos de estudio.

8.4. Resumen

En este capítulo se describen tres casos de estudio donde BPIMS-WS se utilizó para la integración de procesos de negocio en la cadena de suministro. El primer caso de estudio describe la integración de los procesos de negocio de un modelo de empresa virtual a través de BPIMS-WS. El segundo caso de estudio describe la funcionalidad de BPIMS-WS como un agente de compra distribuido. Finalmente, el último caso de estudio describe la funcionalidad de BPIMS-WS para el reaprovisionamiento oportuno en la cadena de suministro.

Capítulo 9
Conclusiones

9.1. Impacto de BPIMS-WS

Los servicios Web son independientes de la plataforma de cómputo, sistema operativo y lenguaje de programación. Por consiguiente, cualquier proveedor puede integrarse a BPIMS-WS siempre y cuando proporcione un número mínimo de servicios Web de acuerdo al conjunto de especificaciones de servicios provistos por BPIMS-WS, como por ejemplo solicitud de precio, tiempo de entrega, cantidad y características técnicas de un producto. En BPIMS-WS, las colaboraciones comerciales se realizan en tiempo de ejecución al momento que se inicia un proceso de compra o de reaprovisionamiento oportuno. Así también, BPIMS-WS realiza dinámicamente el descubrimiento de servicios a través del nodo UDDI extendido y la invocación de los servicios localizados. Dado que múltiples proveedores son fuentes de información de sus productos registrados, el proceso de compra en BPIMS-WS provee información actualizada en tiempo real a los potenciales compradores y mejora la conducción de los procesos de negocio dentro de una cadena de suministro. En esta tesis se abordó el problema de mejorar el funcionamiento del reaprovisionamiento oportuno, y además se demostró cómo en el proceso de reaprovisionamiento oportuno, la disponibilidad de información actualizada así como el procesamiento de órdenes de compra en tiempo real es de crucial importancia entre los socios comerciales en la cadena de suministro.

Trabajos recientes en el área describen las características técnicas de los servicios Web y discuten cómo los servicios Web pueden utilizarse para mejorar las colaboraciones comerciales [139]. En [139] se argumenta que los servicios Web deben adoptarse para la integración de procesos de negocio en la cadena de suministro debido a sus ventajas técnicas y económicas. En el contexto económico, la adopción los servicios Web reduce costos, aumenta las oportunidades para nuevos clientes quienes requieren el uso de

servicios Web para la integración con ellos, y se pueden obtener ganancias ya que una aplicación basada en servicios Web puede ser útil a otros clientes por lo cual ellos podrían pagar por este servicio.

También, los servicios Web mejoran la integración entre sistemas heterogéneos y distribuidos, por lo que existen beneficios adicionales al extender las fronteras comerciales de una empresa. En este contexto, éstos pueden medirse mediante dos criterios generales, satisfacción del cliente en un enfoque externo y eficacia de costos en un enfoque interno, los cuales son consistentes con el conjunto de atributos propuesto en el modelo SCOR (*Supply Chain Operations Reference*) [140]. El modelo SCOR provee una taxonomía y estructura de procesos de negocio a través de cinco elementos funcionales: planeación, fuentes de información, fabricación, entrega y retorno. Este modelo clasifica el rendimiento en cinco métricas: costo, administración de los recursos, fiabilidad, grado de respuesta y flexibilidad. El rendimiento en cada uno de estos atributos puede mejorar con la implementación de servicios Web en el proceso de compra de los casoa de estudio presentados en esta tesis. En este contexto, poca investigación se ha hecho acerca del impacto de servicios Web en el funcionamiento del reaprovisionamiento oportuno en la cadena de suministro, por lo cual esta sigue siendo un área importante de investigación a futuro. Recientes trabajos discuten los beneficios de EDI en los procesos de compra y muestran que la producción y costos de transacción se reducen tal como se expone en [141]. Cuando las colaboraciones comerciales de los socios comerciales se realizan en Internet, se observa una disminución de costos y mejoras operacionales dando como resultado el rediseño de flujos de trabajos [142]. Además, la óptima utilización de los recursos mejora las relaciones comerciales en la cadena de suministro [143, 144, 145]. Así también se mejora la satisfacción del cliente en un enfoque externo utilizando los servicios Web. Por ejemplo, la fiabilidad se logra con la adecuada selección de tecnologías de información mejorando los flujos de información en la cadena del suministro ya que la información se encuentra públicamente disponible a los proveedores [146, 147].

El grado de respuesta de una orden de compra también puede mejorarse mientras la información esté más disponible en las cadenas de suministro [148].

9.2. Trabajo a Futuro

A pesar de todo lo anterior, existen problemas en los servicios Web que necesitan abordarse para la integración de procesos de negocio basados en servicios Web. Entre estos problemas están:

1) La necesidad de desarrollar estándares únicos consistentes con los servicios Web registrados en nodos UDDI públicos. Nuevos estándares industriales de servicios Web para transacciones comerciales o documentación surgen día con día, por ejemplo, el Lenguaje Comercial Universal (*Universal Business Language*) de OASIS [149], el cual no es consistente con los servicios publicados en los nodos UDDI.
2) La seguridad es el problema principal que enfrentan los servicios Web. Actualmente, los servicios Web no han adoptado los protocolos y estándares de seguridad propuestos por la comunidad. Muchas empresas todavía utilizan estándares de seguridad existentes como HTTPS o Encriptación en los mensajes SOAP para garantizar la comunicación segura en los servicios Web. Estándares y protocolos de seguridad para servicios Web como SAML [150], *WS-Security* [151], *WS-Trust* [152] y *WS-SecurityPolicy* [153] todavía se encuentran en proceso de aceptación por la industria.
3) La carencia de aspectos de calidad en los servicios Web es un problema adicional. Actualmente, las especificaciones de los servicios Web no presentan aspectos de calidad referentes a la fiabilidad, rendimiento y disponibilidad. Aquí, la fiabilidad global del Internet impacta la confiabilidad de los servicios Web, así también el rendimiento de los servidores de aplicaciones donde los servicios Web residen. Una solución prometedora a este problema es el uso de mallas de servicios Web (*Web Services Grid*) dónde los servicios Web se construyen sobre mallas de computadoras [154].

Sin embargo, hay la necesidad de realizar más investigación para entender bien los beneficios de los sistemas de integración de la cadena de suministro basados en Servicios Web. Diversos problemas de investigación en el área de servicios Web y extensiones potenciales a BPIMS-WS se identificaron y se discuten a continuación:

- BPIMS-WS usa un intercambio privado de documentos para soportar únicamente el proceso de procuración mediante un conjunto de servicios Web. No obstante, servicios Web adicionales pueden desarrollarse así como sus respectivas interfaces gráficas para el soporte de otros procesos de negocio involucrados en la cadena de suministro tales como la administración de inventarios, manufacturación, soporte, distribución, solo por mencionar algunas.
- BPIMS-WS utiliza ontologías como catálogos de productos y servicios. Hasta el momento de la escritura de esta tesis, BPIMS-WS solo tiene soporte para 2 ontologías de productos: UNSPSC y Amazon; y una ontología de servicios: RosettaNet. Sin embargo, el diseño del nodo UDDI extendido que utiliza BPIMS-WS provee soporte para diversas ontologías de productos y servicios sin necesidad de fuertes cambios de diseño y programación. El único requisito es que las ontologías deben estar descritas en DAML [75], DAML-S [155], OWL [76] u OWL-S [156].
- El diseño actual de los nodos UDDI únicamente se enfoca al descubrimiento de los servicios de otras compañías, mientras que pasa a segunda importancia, cómo estas compañías exponen sus servicios. En este trabajo se propone una solución alterna que complementa la funcionalidad de los nodos UDDI encapsulando en ontologías la conducta de los servicios que ofrecen las diversas compañías y los diferentes tipos de productos que ofrecen.

Una vez que las compañías consideren los beneficios de las aplicaciones de integración de la cadena de suministro basadas en servicios Web, como se discute en esta tesis, las compañías considerarán el desarrollo de nuevos sistemas de información inter e intra-organizacionales para fortalecer la integración de sus cadenas de suministro.

9.3. Conclusiones Generales

En este trabajo se argumentó que las tecnologías de información recientes no soportan las necesidades dinámicas y de tiempo real de algunos procesos involucrados en la cadena de suministro. En este trabajo se demostró que los servicios Web pueden proveer ese soporte

con un prototipo desarrollado para la integración de procesos de negocio de la cadena de suministro el cual ilustra las capacidades de los servicios Web. Para el proceso de reaprovisionamiento oportuno de la cadena de suministro, la naturaleza dinámica de los servicios Web permite la identificación de los proveedores con información de productos y servicios actualizada. Adicionalmente, en el prototipo desarrollado se permite la variación de criterios de búsqueda de productos para acceder de una forma rápida y eficaz la información de las bases de datos de los diferentes proveedores a través de los servicios Web. Además, se resumen los beneficios obtenidos de la arquitectura de coordinación orientada a servicios propuesta la cual abarca múltiples tecnologías de información y diversos mecanismos de integración.

Finalmente, en este trabajo de investigación describimos la arquitectura de coordinación orientada a servicios propuesta argumentando que el prototipo desarrollado el cual es un sistema de integración de procesos de negocio de la cadena de suministro en donde se plasman los conceptos e ideas de la arquitectura que reduce costos de integración, amplía las fronteras comerciales de los socios comerciales involucrados y mejora los niveles de servicio.

Apéndice A
Interfaces de Programación de BPIMS-WS

A.1. Lista de Servicios

Nombre: delete_Business
Descripción: Operación que permite eliminar un negocio registrado en el nodo UDDI mediante un proceso de autenticación

Parámetros	**Entrada:**	password	**Type**:xsd:string
		login	**Type**:xsd:string
	Salida:	deletedServicesList	**Type**: xsd1:DeletedServicesList
		deletedServicesNumber	**Type**: xsd:string
		deletedProductsList	**Type**: xsd1:DeletedProductsList
		deletedProductsNumber	**Type**: xsd:string
		Details	**Type**: xsd:string

Nombre: delete_Products
Descripción: Operación que permite eliminar una lista de productos especificada pertenecientes a un negocio. Es necesario un proceso de autenticación.

Parámetros	**Entrada:**	password	**Type**:xsd:string
		Login	**Type**:xsd:string
		ProductsList	**Type**: xsd1:ProductsList
	Salida:	deletedProductsList	**Type**: xsd1:DeletedProductsList
		deletedProductsNumber	**Type**: xsd:string
		Details	**Type**: xsd:string

Nombre: delete_Services
Descripción: Operación que permite eliminar una lista de servicios referentes a un negocio registrado. Es necesario un proceso de autenticación.

Parámetros	**Entrada:**	password	**Type**:xsd:string
		login	**Type**:xsd:string
		ServicesList	**Type**: xsd1:ServicesList
	Salida:	deletedServicesList	**Type**: xsd1:DeletedServicesList
		deletedServicesNumber	**Type**: xsd:string
		Details	**Type**: xsd:string

Nombre: find_Product
Descripción: Operación que permite encontrar los negocios registrados que tienen un producto asociado a partir de su código y ontología.

Parámetros	**Entrada:**	Product	**Type**: xsd1:ProductInfo
	Salida:	BusinessList	**Type**: xsd1:BusinessList

Nombre: find_Service
Descripción: Operación que permite encontrar los negocios registrados que tienen un servicio a partir de su código que corresponde al PIP de RosettaNet.

Parámetros	**Entrada:**	serviceCode	**Type**: xsd:string
	Salida:	BusinessList	**Type**: xsd1:BusinessList

Nombre: Get_AsyncStockQuote
Descripción: Operación que permite almacenar una solicitud para un producto requerido. Dicha solicitud se almacena hasta que el producto este disponible por un proveedor.

Parámetros	**Entrada**:	businessKey	**Type**:xsd:string
		ProductCode	**Type**:xsd:string
		ontology	**Type**:xsd:string
		quantity	**Type**:xsd:string
		timeToPerform	**Type**:xsd:string
	Salida:	IdRequest	**Type**:xsd:string
		status	**Type**:xsd:string

Nombre: get_BusinessDetail
Descripción: Operación que permite encontrar información detallada de un negocio a partir de su identificador único.

Parámetros	**Entrada:**	businessKey	**Type**: xsd:string
	Salida:	Business	**Type**: xsd1:Business
		ProductsList	**Type**: xsd1:ProductsList
		ServicesList	**Type**: xsd1:ServicesList

Nombre: get_BusinessOntology
Descripción: Operación que permite obtener la ontología de negocios que soporta BPIMS-WS.

Parámetros	**Entrada:**		
	Salida:	BusinessOntology	**Type**: xsd1:BusinessOntology

Nombre: get_NotifyStockQuote
Descripción: Operación que permite obtener una notificación cuando una cantidad específica de un producto esté disponible

Parámetros	**Entrada:**	IdRequest	**Type**:xsd:string
		businessKey	**Type**:xsd:string
	Salida:		

Nombre: get_PasswordRecovery
Descripción: Operación que permite obtener el login y password vía e-mail para procesos de autenticación.

Parámetros	**Entrada:**	email	**Type**: xsd:string
	Salida:	ok	**Type**: xsd:boolean

Nombre: get_PriceandDeliveryTime
Descripción: Operación que permite obtener una lista de proveedores de un producto a partir de su código y ontología.

Parámetros	**Entrada:**	Product	**Type**: xsd1:ProductInfo
	Salida:	ProvidersList	**Type**: xsd1:ProvidersList

Nombre: get_ProductsByProvider
Descripción: Operación que permite obtener una lista de productos y sus respectivas cantidades de un negocio registrado específico.

Parámetros	**Entrada:**	Business	**Type**: xsd1:Business
	Salida:	businessKey	**Type**: xsd:string
		BusinessName	**Type**: xsd:string
		Description	**Type**: xsd:string
		DiscoveryURL	**Type**: xsd:string
		OverviewURL	**Type**: xsd:string
		ProductsList	**Type**: xsd1:ResultProductsList

Nombre: get_ProductsOntology
Descripción: Operación que permite obtener la ontología de productos que soporta BPIMS-WS.

Parámetros	**Entrada:**		
	Salida:	ProductsOntology	**Type**: xsd1:ProductsOntology

Nombre: get_ProductWarranty
Descripción: Operación que permite obtener una lista de proveedores y la información de garantía a partir del código de un producto.

Parámetros	**Entrada:**	productCode	**Type**: xsd:string
	Salida:	ProvidersList	**Type**: xsd1:ProvidersList

Nombre: get_ProviderQuantity
Descripción: Operación que permite obtener la cantidad en existencia de un producto requerido de un negocio registrado específico a partir de su código de producto y ontología.

Parámetros	**Entrada:**	businessKey	**Type**: xsd:string
		productCode	**Type**: xsd:string
		ontology	**Type**: xsd:string
	Salida:	productCode	**Type**: xsd:string
		ontology	**Type**: xsd:string
		quantity	**Type**: xsd:string
		productName	**Type**: xsd:string
		businessName	**Type**: xsd:string
		description	**Type**: xsd:string
		discoveryURL	**Type**: xsd:string
		overviewURL	**Type**: xsd:string

Nombre: get_ProviderURLBuy
Descripción: Operación que permite obtener la dirección electrónica del proceso de compra de un negocio específico a partir de su identificador único.

Parámetros	**Entrada:**	businessKey	**Type**: xsd:string
	Salida:	overviewURL	**Type**: xsd:string

Nombre: get_ProviderURLPriceDelivery
Descripción: Operación que permite obtener la dirección electrónica del proceso de obtención de precio y tiempo de entrega de un producto referente a un negocio específico a partir de su identificador único.

Parámetros	**Entrada:**	businessKey	**Type**: xsd:string
	Salida:	overviewURL	**Type**: xsd:string

Nombre: get_ProviderURLQuantity
Descripción: Operación que permite obtener la dirección electrónica del proceso de obtención de cantidad en existencia de un producto referente a un negocio específico a partir de su identificador único.

Parámetros	**Entrada:**	businessKey	**Type**: xsd:string
	Salida:	overviewURL	**Type**: xsd:string

Nombre: get_Quantity
Descripción: Operación que permite obtener una lista de proveedores y la información en cantidad en existencia de un producto requerido a partir de su código y ontología.

Parámetros	**Entrada:**	Product	**Type**: xsd1:ProductInfo
	Salida:	ProvidersList	**Type**: xsd1:ProvidersList

Nombre: get_RegisteredBusiness
Descripción: Operación que permite obtener la lista completa de todos los negocios registrados.
Parámetros **Entrada:**
Salida: BusinessList **Type**: xsd1:BusinessList

Nombre: get_ServicesList
Descripción: Operación que permite obtener la lista completa de servicios Web provistos por BPIMS-WS.
Parámetros **Entrada:**
Salida: ServicesList **Type**: xsd1:ServicesList

Nombre: get_ServicesOntology
Descripción: Operación que permite obtener la ontología de servicios que soporta BPIMS-WS.
Parámetros **Entrada:**
Salida: ServicesOntology **Type**: xsd1:ServicesOntology

Nombre: get_SortedList
Descripción: Operación que permite ordenar una lista de productos en base a un criterio. BPIMS-WS soporta criterios como menor precio, tiempo de entrega más corto, entre otros.
Parámetros **Entrada:** List **Type**: xsd1:List
Salida: SortedList **Type**: xsd1:Sorted

Nombre: save_Business
Descripción: Operación que permite registrar información acerca de un negocio. Un identificador único se asocia a cada negocio registrado.
Parámetros **Entrada:** Business **Type**: xsd1:Business
Salida: businessKey **Type**: xsd:string
Details **Type**: xsd:string

Nombre: save_Products
Descripción: Operación que permite registrar a un negocio una lista de productos que soporta. Se requiere un mecanismo de autenticación.
Parámetros **Entrada:** login **Type**: xsd:string
Password **Type**: xsd:string
ProductsList **Type**: xsd1:ProductsList
Salida: registeredProductsList **Type**: xsd1:RegisteredProductsList
RegisteredProductsNumber **Type**: xsd:string
Details **Type**: xsd:string

Nombre: save_Services

Descripción: Operación que permite registrar a un negocio una lista de servicios Web que soporta. Se requiere un mecanismo de autenticación.

Parámetros	**Entrada:**	login	**Type**: xsd:string
		Password	**Type**: xsd:string
		ServicesList	**Type**: xsd1:ServicesList
	Salida:	registeredServicesList	**Type**: xsd1:RegisteredServicesList
		RegisteredServicesNumber	**Type**: xsd:string
		Details	**Type**: xsd:string

Apéndice B
Tecnologías de Servicios Web

B.1. XML (Extensible Markup Language)

XML (*Extensible Markup Language*) es un lenguaje desarrollado por el W3C (*World Wide Web Consortium*), principalmente para superar las limitaciones de expresividad en HTML. XML es un conjunto de reglas para definir etiquetas que permiten organizar un documento. XML es un metalenguaje que define la sintaxis utilizada para definir familias de lenguajes de etiquetas estructurados [17].

Los servicios Web usan XML por tres razones principales:

1. **Es un estándar abierto.** XML es respaldado por muchas compañías con base tecnológica como IBM, Microsoft, Sun Microsystems, entre otros; quienes lo integran en sus aplicaciones y herramientas de desarrollo
2. **Simplicidad de sintaxis.** XML representa cualquier tipo de datos y hace fácil su codificación y entendimiento.
3. **Independencia del protocolo de transporte.** XML no necesita de ningún protocolo de trasporte especial, solo necesita de un protocolo que pueda transferir texto o documentos simples.

Un ejemplo de un documento XML que contiene la información principal referente a un libro se muestra en la Fig. A.1.

XML tiene las ventajas de ser independiente de la plataforma, de ser legible, y de contar con apoyo extenso de la industria. Así, XML se ha vuelto un formato estándar para el intercambio de datos estructurados y es la tecnología base para otras tecnologías de servicios Web como SOAP, WSDL, UDDI y WSIL que se describen en las secciones siguientes.

```
<?xml version="1.0"?>
<book>
  <title>Xml: Extensible Markup Language</title>
  <availability time="24" unit="hours"/>
  <author>Elliotte Rusty Harold</author>
  <format>Paperback</format>
  <publication>1998</publication>
  <price quantity="31.99" currency="dollar"/>
  <discount quantity="20"/>
  <isbn>15566332881</isbn>
  <link ref="/exec/obidos/ASIN/0764531999/qid=919015337"/>
</book>
```

Fig. B.1 Ejemplo de un documento XML con información referente a un libro

B.2. SOAP (Simple Object Access Protocol)

SOAP (*Simple Object Access Protocol*) permite el intercambio libre y estructurado de mensajes a través de Internet, definiendo un mecanismo estándar de comunicación e intercambio de datos entre aplicaciones [30]. La especificación SOAP describe [157]:

- **Estructura del Mensaje.** El contenido de un mensaje SOAP. Un mensaje SOAP es un documento XML que consiste de los siguientes elementos:
  - **Sobre (*Envelope*).** Es el elemento principal del documento XML. Provee un contenedor para información de control, la dirección de un mensaje, y el propio mensaje.
  - **Encabezado (*Header*).** Es un mecanismo genérico para agregar características adicionales a un mensaje SOAP. Puede contener información de control como atributos de calidad de un servicio, entre otros.
  - **Cuerpo (*Body*).** Es un contenedor para información requerida por el destinatario del mensaje. Contiene la identificación del mensaje y sus parámetros.
- **Estilos de Comunicación.** El estilo en que se intercambian los mensajes SOAP. Existen 2 estilos de comunicación que soporta SOAP:
  - **Llamadas Procedimientos Remotos (*RPC*).** Es una invocación síncrona a un método que devuelve un resultado. Primero, el cliente envía una solicitud SOAP a un servidor de aplicaciones. Entonces el servidor, devuelve una respuesta a la solicitud SOAP del cliente.

  - **Documento.** Es también conocido como el estilo orientado a mensaje. Con este estilo, un documento XML se empaqueta como el contenido del mensaje SOAP.

- **Reglas de Codificación.** Reglas que definen un mecanismo de serialización para el intercambio de los tipos de datos definidos en una aplicación. En un entorno distribuido, el estilo de codificación define cómo los valores de los datos definidos en una aplicación se traducen en un formato particular, esto es, serializado/deserializado. En el caso de SOAP, el estilo de codificación determina cómo se representan los valores de los datos en los mensajes SOAP. SOAP soporte 2 estilos de codificación que son:
  - **Codificación SOAP (*SOAP Encoding*).** Define un conjunto de reglas para representar la estructura de datos permitiendo serializar/deserializar los valores de los tipos de datos a y desde un modelo de datos SOAP.
  - **XML literal.** Permite convertir directamente un árbol de nodos DOM-XML a un mensaje SOAP y viceversa. XML literal sólo usa los tipos de datos definidos en XSD (*XML Schema*).

Un ejemplo de un mensaje SOAP se muestra en la Fig. A.2.

```
<Envelope>
   <Header>
      <transId>345</transId>
   </Header>
   <Body>
      <getPriceAndQuantity>
         <price>300</price>
         <quantity>4</quantity>
      </Add>
   </Body>
</Envelope>
```

Fig. B.2. Ejemplo de un mensaje SOAP

B.3. WSDL (Web Services Description Language)

WSDL (*Web Services Description Language*) es un dialecto de XML para describir formalmente interfaces de servicios Web y sus localizaciones, para que puedan encontrarse automáticamente [31].

WSDL permite a un proveedor de servicio especificar las siguientes características de un servicio Web:

1. El nombre del servicio Web y su información de localización.
2. El protocolo y estilo de codificación que utilizan las operaciones del servicio Web.
3. Las operaciones, parámetros, y tipos de datos correspondientes a la interfaz del servicio Web, y el nombre de la interfaz.

Un documento de WSDL consiste de dos partes o documentos [31]:

1. **Interfaz del servicio.** Describe la interfaz y sus protocolos de comunicación de manera abstracta. La interfaz del servicio de un documento WSDL incluye los siguientes elementos:
    a. **<portType>:** define un conjunto abstracto de una o más operaciones soportadas por uno o más puertos. Cada elemento hijo <*operation*> define un mensaje de entrada y salida así como un mensaje opcional de fallo.
    b. **<message>:** define de manera abstracta los datos intercambiados. Un mensaje puede tener uno o más elementos hijos <*part*>. Este elemento esencialmente define un conjunto de parámetros para una operación.
    c. **<type>:** define un contenedor para la definición de tipos de datos en *XML Schema*.
    d. **<bindings>:** define un protocolo y especificación del formato de datos para un elemento <*portType*> particular.
2. **Implementación del servicio.** Describe la información de acceso al servicio (localización). La implementación del servicio de un documento WSDL incluye los siguientes elementos:
    a. **<port>:** define una dirección URL para un solo *endpoint*. Una dirección de un protocolo específico se asocia con un solo elemento <*binding*> en la descripción de la interfaz del servicio.

b. **<service>:** es una colección de puertos relacionados.

La Fig. A.3 resume los elementos de la interfaz e implementación del servicio en un documento WSDL

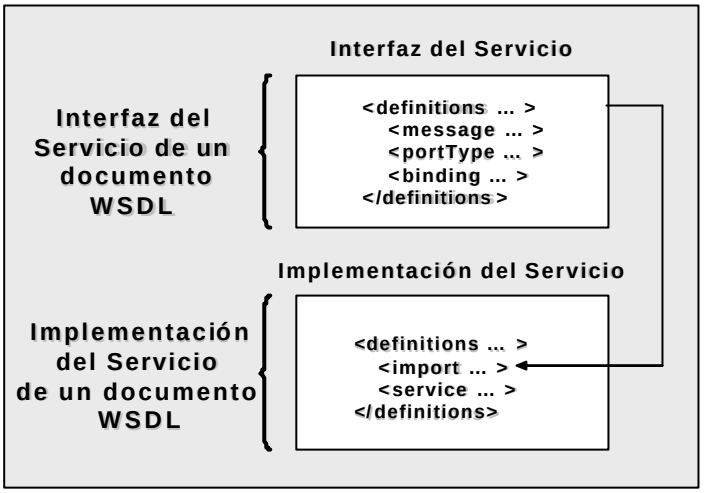


Fig. B.3. Elementos de la interfaz e implementación del servicio en un documento WSDL

B.4. UDDI (Universal Description Discovery and Integration)

UDDI (*Universal Description, Discovery and Integration*) es un esfuerzo industrial que comenzó en Septiembre del 2000 por Ariba, IBM, Microsoft, y otras 33 compañías. La idea de UDDI es muy simple: dado que existen varias compañías que proveen servicios Web, ¿cómo encontrar un servicio que satisfaga las necesidades de un cliente? La solución que ofrece UDDI es utilizar un registro distribuido como un mecanismo común para la publicación de las descripciones de los servicios Web [32]. Para esto, UDDI hace uso de estándares establecidos, entre los cuales, HTTP, XML y SOAP poseen la mayor importancia. La principal función de UDDI es anunciar cuáles son los servicios que ofrece una empresa, mientras que cómo dichas organizaciones ofrecen sus servicios es de menor importancia. UDDI se basa en el concepto de registro distribuido que proporciona la funcionalidad comercial de organizar la información en páginas Amarillas, Blancas y Verdes, similar a un directorio telefónico.

La información de las compañías se almacena en un documento XML llamado *BusinessEntity*. La estructura y elementos que contiene este documento se muestra en la Fig. A.4.

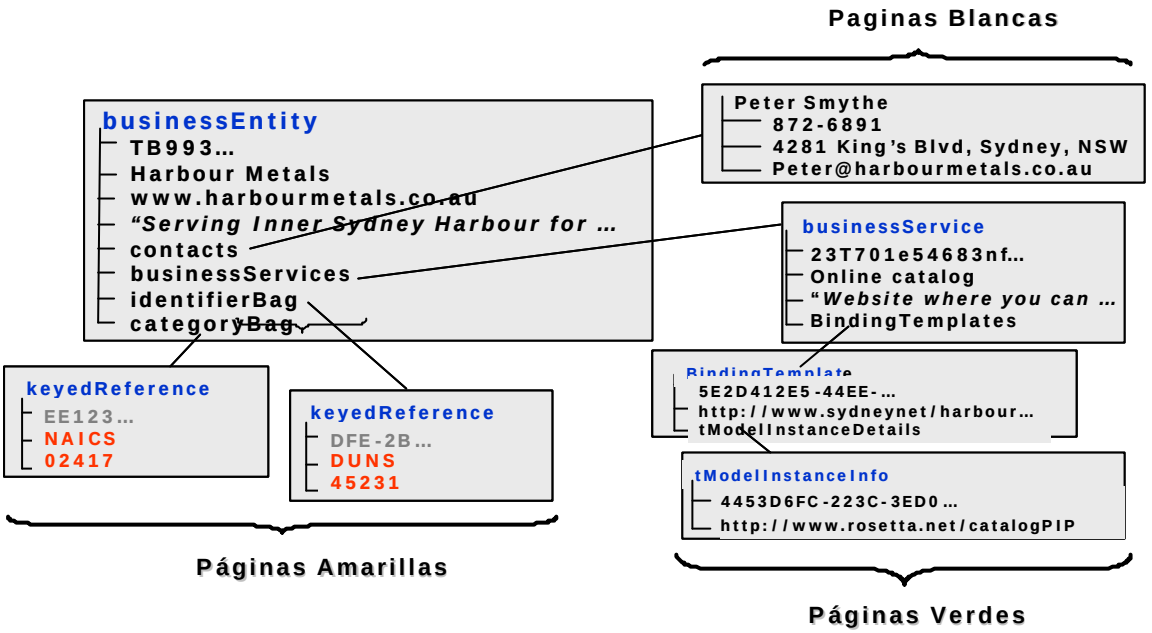


Fig. B.4. Estructura y elementos de la entidad BusinessEntity en UDDI

Como se observa en la Fig. A.4., el elemento *BusinessEntity* organiza la información de las compañías como un directorio telefónico, clasificando la información en páginas amarillas, blancas y verdes. A continuación se describe el significado de cada una de ellas:

- **Páginas Blancas:** Incluye descripción de la compañía, información de contacto e identificadores de la compañía como DUNS (*Data Universal Numbering System*).
- **Paginas Amarillas:** Contiene la categoría de negocios al cual pertenece la compañía.
- **Páginas Verdes:** Contiene la descripción del servicio, es decir, cómo se realiza comercio electrónico con otra compañía.

B.5. WSIL (Web Service Inspection Language)

WSIL (*Web Services Inspection Language*) proporciona un descubrimiento distribuido de las descripciones de servicios Web, especificando cómo inspeccionar un sitio Web para localizar a los servicios Web disponibles [97].

La especificación de WSIL define los elementos necesarios para saber donde buscar en un sitio Web, las descripciones de los servicios Web. Dado que WSIL se enfoca en el descubrimiento distribuido de servicios, la especificación de WSIL, complementa a UDDI facilitando el descubrimiento de servicios disponibles en sitios Web, pero que no están todavía publicados en un nodo UDDI.

La especificación WSIL no define un nuevo lenguaje de descripción de servicios Web. Los documentos WSIL proporcionan un método para realizar referencias a diferentes tipos de descripciones de servicios Web. Dentro de un documento WSIL, un solo servicio Web puede tener más de una referencia a una descripción de servicio. Por ejemplo, un solo servicio Web puede describirse tanto en un documento WSDL como en un nodo UDDI. Para esto, deben ponerse referencias a estas dos descripciones de servicio en un documento WSIL. Esto es de gran beneficio para el solicitante de servicio, ya puede seleccionar los tipos de descripción de servicios que es capaz de entender y usar.

En un documento WSIL se pueden agregar referencias a las descripciones de los servicios Web. Estas descripciones de servicio pueden definirse en cualquier formato de descripción de servicio, como WSDL, UDDI, o HTML [97].

Un documento WSIL puede contener una lista de referencias a las descripciones de los servicios, así como referencias a otros documentos WSIL. Para ello, se utilizan los elementos <service> y <link>, respectivamente. Un elemento <service> puede tener una o más referencias a diferentes tipos de descripciones de servicio para un mismo servicio Web. El elemento <link> puede contener referencias a sólo un tipo de descripción de servicio, pero esta descripción de servicio no tiene referencias al mismo servicio Web. En la Fig. A.5 se muestra un ejemplo simple de un documento WSIL. En la Fig. A.5, el documento contiene dos referencias a descripciones de servicio diferentes, y una referencia a otro documento WSIL. El primer elemento <service> contiene sólo una descripción de servicio, y es una referencia a un documento WSDL. El segundo elemento <service> también

contiene sólo una referencia de descripción de servicio. Esta referencia es a un servicio en un nodo UDDI.

```
<?xml version="1.0"?>
<inspection xmlns="http://schemas.xmlsoap.org/ws/2001/10/inspection/">
  <service>
    <description referencedNamespace="http://schemas.xmlsoap.org/wsdl/"
                 location="http://example.com/exampleservice.wsdl" />
  </service>
  <service>
    <description referencedNamespace="urn:uddi-org:api">
       <wsiluddi:serviceDescription
location="http://example.com/uddi/inquiryapi">
     <wsiluddi:serviceKey>52946BB0-BC28-11D5-A432-0004AC49CC1E
</wsiluddi:serviceKey>
       </wsiluddi:serviceDescription>
    </description>
  </service>
  <link
referencedNamespace="http://schemas.xmlsoap.org/ws/2001/10/inspection/"
        location="http://example.com/tools/toolservices.wsil"/>
</inspection>
```

Fig. B.5. Ejemplo de un documento WSIL

El elemento <serviceKey> identifica de forma única a un servicio Web en un nodo UDDI. Finalmente, el elemento <link> hace referencia a una colección de descripciones de servicio, en este caso, a otro documento WSIL.

B.6. USML (UDDI Search Markup Language)

USML (*UDDI Search Markup Language*) es un lenguaje basado en XML que permite buscar y localizar servicios Web con múltiples criterios en diversos nodos UDDI [98]. Según la especificación de UDDI [32], hay tres tipos de datos principales que pueden consultarse en un nodo UDDI: Negocio, Servicio, y Tipo de Servicio (*tModel, Technology Model*). Un *tModel* define los tipos de servicios que utiliza el servicio Web, incluyendo las definiciones de las operaciones, la estructura y protocolos de los mensajes, y protocolos de seguridad de dichos servicios. Partiendo de un *tModel* concreto, se puede saber qué operaciones de servicio Web realiza la entidad que implementa dicho *tModel* y cómo obtener acceso a ellas. Además, en un *tModel* se especifica información como: nombre del

tModel, nombre de la organización que publicó el *tModel*, una lista de categorías que describen el tipo de servicio. En USML, estos tres tipos de datos pueden consultarse de forma conjunta o independiente. El esquema general de un documento USML se muestra en la Fig. A.6.

```
<Query>
   <Source/>
   <SourceURL/>
   <BusinessName/>
   <Identifier/>
   <Category/>
   <ServiceName/>
   <ServiceTypeName/>
   <DiscoveryURL/>
   <FindBy/>
   <AggOperator/>
   <RequestTypeName/>
</Query>
```

Fig. B.6. Esquema general de un documento en USML

Donde:

- **Query** especifica las condiciones de búsqueda. Combina la búsqueda de la palabra clave, la búsqueda basada en los identificadores, y la búsqueda basada en las categorías.
- **Source** es el nodo UDDI donde se realiza la búsqueda. Puede ser un nodo público o privado.
- **SourceURL** es el URL del nodo UDDI donde se realiza la búsqueda.
- **BusinessName** es el nombre del negocio a buscar.
- **Identifier** es el nombre del identificador y su valor asociado. Se aceptan dos tipos de identificadores: DUNS (*Data Universal Numbering System*) [158] y Thomas Register [159].
- **Category** es el nombre de la categoría y su valor asociado. Actualmente, se aceptan cinco tipos de categorías: NAICS (*North American Industry Classification System*) [68], UNSPSC (*United Nations Standard Products and Services Code*) [69], GEO [160], UDDITYPE, y SIC (*Standard Industrial Classification*) [161].
- **ServiceName** es el nombre del servicio. Se usa en una búsqueda por el nombre de servicio.

- **ServiceTypeName** es el nombre del tipo de servicio (*tModel*). Se usa cuando la búsqueda es por el tipo de servicio.
- **DiscoveryURL** es el URL para el descubrimiento.
- **FindBy** especifica el tipo de datos a buscar (Negocio, Servicio o Tipo de Servicio).
- **AggOperator** especifica el tipo de operador lógico a utilizar en la búsqueda. Se soportan dos tipos de operadores lógicos: AND y OR. Si se utiliza el operador OR, la información especificada en la etiqueta *FindBy* de cada búsqueda es devuelta. En caso de utilizar el operador AND, solo información relacionada al tipo de datos especificado en la etiqueta *RequestTypeName* será devuelta. Más de 3 búsquedas (una búsqueda para cada uno de los tipos de datos) no se permite.
- **RequestTypeName** especifica el tipo de datos que se desean como resultado en la consulta (Negocio, Servicio o Tipo de Servicio).

Con la ayuda de USML se pueden combinar diferentes criterios de búsqueda en una consulta y así conseguir eficacia y exactitud al realizar una sola búsqueda. Esto se debe a que en USML se pueden utilizar dos tipos de operadores lógicos en las consultas: AND y OR. Estos operadores están situados en la etiqueta *AggOperator.*

B.7. WSIF (Web Service Invocation Framework)

WSIF (*Web Services Invocation Framework*) es una API en Java para la invocación de los servicios *Web*, no importando dónde y cómo los servicios se proveen [82].
Web Services Invocation Framework (WSIF) es una tecnología para invocar servicios descriptos en WSDL en una variedad de protocolos de red, tales como SOAP, JMS y RMI. WSIF permite desplegar servicios Web en forma flexible, en tanto que el lenguaje WSDL describe un servicio Web y sus capacidades.
WSIF es expansible y puede dar soporte fácilmente a protocolos de transporte adicionales, tales como mensajería instantánea. También brinda soporte a múltiples modelos de programación y permite a los desarrolladores interactuar con las aplicaciones de servicios Web, sea cual fuere el modo de implantación y acceso del servicio Web.

Acrónimos y Términos Usados

1. **Arquitectura:** La arquitectura de software de un programa o sistema de cómputo es la estructura o estructuras del sistema. Esta estructura incluye componentes de software, las propiedades externas de visibilidad de estos componentes, las relaciones entre ellos y las restricciones sobre su uso.
2. **Bajo Acoplamiento:** Acoplamiento es la dependencia entre la interacción de sistemas. Esta dependencia se descompone en dependencia real y artificial.
    a. La dependencia real es el conjunto de características o servicios que un sistema consume de otros sistemas. La dependencia real siempre existe y no puede reducirse.
    b. La dependencia artificial es el conjunto de factores que un sistema tiene que cumplir para consumir las características o servicios provistos por otros sistemas. Algunos factores de dependencia artificial son: dependencia del lenguaje, de la plataforma, entre otros. La dependencia artificial siempre existe pero puede reducirse.

    Entonces, el bajo acoplamiento describe la configuración en el cual la dependencia artificial se reduce al mínimo.
3. **Binding:** Una asociación entre una interfaz, un protocolo concreto y un formato de datos. Especifica el protocolo y formato de datos que debe usarse en la transmisión de mensajes definidos por la interfaz asociada.
4. **BPEL4WS:** El *Business Process Execution Language for Web Services* es un lenguaje composicional basado en XML que permite definir los procesos de negocio en términos de actividades de coordinación y comunicación.
5. **Componente:** Es un objeto de software que interactúa con otros componentes encapsulando cierta funcionalidad un conjunto de funcionalidades. Un componente tiene una interfaz bien definida y sigue una conducta común prescrita a todos los componentes dentro de una arquitectura.
6. **Correlación de mensajes:** Es la asociación de un mensaje con un contexto. La correlación de mensajes asegura que el solicitante de un servicio identifique la

respuesta apropiada con respecto a una solicitud hecha, especialmente cuando existen múltiples respuestas.

7. **DAML-S:** El *DARPA Agent Markup Language* es un lenguaje ontológico basado en DAML+OIL orientado a los servicios Web. Describe semánticamente las propiedades y capacidades de los servicios Web de tal forma que sean interpretables por las computadoras.
8. **Descubrimiento:** Es el acto de localizar una descripción procesable por una máquina de un recurso basado en servicios Web que previamente era desconocido y que cumple un cierto criterio funcional. Esto involucra hacer coincidir un conjunto de criterios funcionales con un conjunto de descripciones de recursos. La meta es encontrar un apropiado recurso basado en servicios Web.
9. **DOM:** Es una forma de representar documentos estructurados (tales como una página Web HTML o un documento XML) que es independiente de cualquier lenguaje orientado a objetos.
10. **End point:** Es una asociación entre un *binding* y una dirección de red especificada por un URI que se usa para comunicarse con una instancia de un servicio. Esto indica una localización específica para acceder un servicio usando un protocolo específico y un formato de datos.
11. **Interfaz de Servicio:** Es el límite abstracto que un servicio expone. Define los tipos de mensajes y los patrones de intercambio de mensajes involucrados para interactuar con el servicio, junto con algunas condiciones implícitas por esos mensajes.
12. **Mensaje:** Es la unidad básica de datos que envía un agente de servicios Web a otro. Es la unidad básica de comunicación entre un servicio Web y un cliente: los datos transmitidos a o desde un servicio Web.
13. **Proxy:** Programa intermediario que actúa a la vez como servidor y cliente para realizar demandas de otros clientes. Las demandas se tratan o bien de manera interna o pasándolas, con posible traducción, a otros servidores. Un *proxy* debe interpretar si es necesario, re-escribir un mensaje de pedido antes de enviarlo.
14. **Servicio:** Es un recurso abstracto que representa una capacidad de ejecutar tareas que forman una funcionalidad coherente desde el punto de vista de entidades proveedoras y solicitantes.
15. **Servicio de Descubrimiento:** Es un servicio que habilita a agentes la recuperación de descripciones de recursos basados en servicios Web.

16. **SGB:** Es una arquitectura abstracta de alto nivel que provee facilidades de administración que permiten a los servicios funcionar como una unidad integrada para colaborar con otros servicios. La arquitectura SGB provee los mecanismos necesarios para el registro, descubrimiento, selección/enrutamiento, aplicación de reglas comerciales, filtrado, asignación de ruta y mapeos topológicos de instancias de servicios.
17. **SOA:** Es un diseño lógico de un sistema de software que proporciona servicios a aplicaciones de usuarios finales y otros servicios que se encuentran distribuidos en una red a través interfaces de publicación y descubrimiento.
18. **SOAP:** El *Simple Object Access Protocol* es un protocolo ligero situado encima de protocolos de transporte como HTTP o SMTP. Se usa para enviar mensajes y hacer llamadas a procedimientos remotos. Su estructura está constituida por un sobre, una cabeza y un cuerpo. En este último se introduce la información enviada o recibida de un servicio Web.
19. **Transacción:** Es una característica de la arquitectura que soporta la coordinación de resultados u operaciones en una interacción que involucra múltiples pasos. La característica fundamental de una transacción es la habilidad de unir múltiples acciones en una misma unidad de trabajo.
20. **UDDI:** El *Universal Descrption, Discovery and Integration* define un registro y protocolos asociados para la búsqueda y localización de servicios Web. Provee dos tipos de servicio: publicación y consulta. El primero define operaciones para registrar, modificar y eliminar negocios y servicios. Mientras que el segundo define operaciones que permiten realizar búsquedas sobre negocios y servicios.
21. **WSIF:** El *Web Services Invocation Framwork* es una API escrita en Java para la invocación de servicios Web no importando dónde y cómo se proveen los servicios. Contiene utilerías que permiten la inspección de los documentos WSDL de un servicio Web.
22. **WSDL:** El *Web services Description Language* es un lenguaje basado en XML, desarrollado por Microsoft e IBM, para describir servicios Web como colecciones de puntos de comunicación donde los mensajes entre el proveedor de servicios y el cliente son intercambiados. En esencia, el WSDL describe las interfaces de un servicio Web y provee la información de contacto para sus usuarios.
23. **XML:** El *Extensible Markup Language* es un conjunto de reglas orientadas a la creación de lenguajes de marcado semánticamente ricos. Se ha consolidado como el

lenguaje estándar para la descripción de información promoviendo así una mayor interoperabilidad entre las aplicaciones.

24. **Coreografía:** La coreografía consiste en permitir a cada una de las partes involucradas en el proceso de negocio describir el rol que juegan en la interacción. En la coreografía se deja a cada una de las partes del proceso ejecutarse desde su perspectiva sin que ninguna de ellas controle la conversación.
25. **Ontología:** Es un conjunto de definiciones inteligibles para las máquinas que permiten crear una taxonomía de clases y subclases de objetos y relaciones entre ellas. Cada clase y subclase tiene un significado semántico claro de la aplicación cliente y el servicio Web.
26. **Orquestación:** La orquestación de servicios Web consiste en proveer los medios necesarios para establecer un proceso de negocio ejecutable que permita la interacción entre servicios Web internos o externos de una organización. El proceso de negocio es dirigido por una de las partes involucradas en dicho proceso.
27. **Proceso de negocio.** Un proceso de negocio consiste en la especificación del orden de ejecución de operaciones de una colección de servicios Web, la información compartida entre estos servicios Web, los socios involucrados y la forma en que éstos se involucran en el proceso de negocio.
28. **RosettaNet:** Es un consorcio que tiene como objetivo crear e implementar estándares que sirvan para formar un lenguaje de negocio común en la que los socios de un proceso de negocio puedan incorporarse libremente a una cadena de suministro.
29. **Socio de negocio:** Es una aplicación en forma de servicio Web que representa a un socio comercial de una organización.
30. **UN-SPSC:** El *United Nations Standard Products and Services Code* es una ontología promovida por las naciones unidas que facilita la clasificación de productos y servicios a través de estándares globales y abiertos.
31. **Workflow:** Un *workflow* consiste en la automatización de un proceso de negocio, parcial o total, durante la cual los documentos, información o tareas se transfieren de un participante a otro de acuerdo a un conjunto de reglas de procedimiento.

Artículos Publicados

En Congresos Nacionales

1. **Giner Alor Hernández**, José Oscar Olmedo Aguirre. *"Automatización de la Cadena de Suministro utilizando UDDI"*. XVI Congreso Nacional y II Congreso Internacional de Informática y Computación de la ANIEI, (CNCIIC-ANIEI 2003). Avances en Informática y Computación Vol. 1. ISBN: 970-36-0100-6, 970-36-0101-4. pp. 515-521.
2. **Giner Alor Hernández**, José Oscar Olmedo Aguirre. *"Lenguajes para la Composición e Integración de Servicios Web"*. I Congreso Nacional en Ciencias de la Computación. Proceedings CNCC 2003. ISBN: 968-863-711-4. pp. 59-68
3. **Giner Alor Hernández**, José Oscar Olmedo Aguirre. *"Automatización de la Cadena de Suministro en el Comercio Electrónico B2B"*. IX Conferencia de Ingeniería Eléctrica (CIE 2003). Proceedings CIE 2003.
4. **Giner Alor Hernández**, José Oscar Olmedo Aguirre. *"Búsqueda, Localización e Invocación Dinámica de Servicios Web utilizando WSIL"*. XVII Congreso Nacional y III Congreso Internacional de Informática y Computación de la ANIEI, (CNCIIC-ANIEI 2004).
5. Lucio Daniel Castelán, **Giner Alor Hernández**, José Oscar Olmedo Aguirre. *"MS4WS: Sistema de Monitoreo para Servicios Web con UML"*. XVII Congreso Nacional y III Congreso Internacional de Informática y Computación de la ANIEI, (CNCIIC-ANIEI 2004).

En Congresos Internacionales

6. **Giner Alor Hernández**, César Sandoval Hernández, Lucio D. Castelán, Paola E, Ramírez Santiago, José Oscar Olmedo Aguirre. *"BPIMS-WS: Business Processes Integration and Monitoring for B2B E-commerce"*. XI International Congress Computer Science Research (CIICC 2004). Proceedings CIICC 2004. ISBN: 968-5823-10-3. pp. 77-84.
7. Oscar Olmedo Aguirre, **Giner Alor Hernández**, Karina Escobar Vázquez, Guillermo Morales Luna. "ADM: An Active Deductive XML Database System". In R. Monroy, G. Arroyo-Figueroa, L.E. Sucar, H. Sossa (eds.): MICAI 2004: Advances in Artificial Intelligence, Third Mexican International Conference on Artificial Intelligence. México City, México, April 2004. Lecture Notes in Artificial Intelligence vol. 2972, pp.139-148. Springer-Verlag Berlin-Heidelberg 2004. ISBN: 3-540-21-459-3.
8. César Sandoval Hernández, **Giner Alor Hernández**, José Oscar Olmedo Aguirre. *"Dinamic Generation of Organizational BPEL4WS workflows"*. I International Conference on Electrical and Electronics Engineering and X Conference on Electrical Engineering (ICEEE-CIE 2004). Proceedings ICEEE-CIE 2004. IEEE Press. ISBN: 0-7803-8532-2.
9. **Giner Alor Hernández**, César Sandoval Hernández, José Oscar Olmedo Aguirre. *"BPIMS-WS: Brokering Architecture for Business Processes Integration for B2B E-*

commerce". IEEE International Conference on Electronics, Communications, and Computers. (IEEE CONIELECOMP 2005). Proceedings of the IEEE-CONIELECOMP 2005. IEEE Press. pp 160-165

10. Juan Miguel Gómez, **Giner Alor Hernandez**, José Oscar Olmedo, Christoph Bussler. *"A B2B conversational Architecture for Semantic Web Services based on BPIM-WS"*. In Proceedings of the 10th IEEE International Conference on Engineering of Complex Computer Systems. (IEEE ICECCS 2005). IEEE Press. pp 252-259.
11. **Giner Alor Hernández**, José Oscar Olmedo Aguirre. *"BPIMS-WS: A Service Oriented Architecture for Trading Partners Integration"*. In Proceedings of the 3td IEEE International Conference on Web Services 2005. IEEE Press. pp 150-155.
12. **Giner Alor Hernandez,** Pedro Ariza Acevedo, Lucio D. Catelán Vega, Jose Oscar Olmedo Aguirre. *"CRM Shopping Portal for B2C E-commerce based on Relevance Feedback"*. XXXI Latin American Informatics Conference (CLEI 2005). ISBN: 958-670-426-2. pp.37
13. **Giner Alor Hernández,** José Oscar Olmedo Aguirre. "Guaranteed Delivery and Processing in Web Services in BPIMS-WS using WS-RM". Proceedings of the 14th International Congress on Computing (CIC 2005). ISBN: 970-36-0266-5. pp 510-519.
14. **Giner Alor Hernández**, Jose Oscar Olmedo Aguirre. "STD: A Virtual Enterprise Model with a Web Services-based Brokering Service". Proceedings of the 14th International Congress on Computing (CIC 2005). ISBN: 970-36-0266-5. pp 520-529.

En Capítulos en Libros

15. **Giner Alor Hernández**, José Oscar Olmedo Aguirre. "Sistema de Intermediación para el Comercio Electrónico B2B basado en Servicios Web". In J. Díaz de León, G. Gonzalez, J. Figueroa (Eds). *Avances en Ciencias de la Computación. Research on Computing Science Series* Vol. 3. ISBN: 970-36-0098-0 pp. 330-334. CIC-IPN, México 2003.
16. **Giner Alor Hernández**, César Sandoval Hernández, José Oscar Olmedo Aguirre. "Descubrimiento Dinámico de Servicios Web en nodos UDDI utilizando USML". In Alexander Gelbukh, Grigori Sidorov, Wilbert A. Olán Cristobal, José A. Vera Felix (Eds). *Recientes Avances en la Ciencia de la Computación en México. Research on Computing Science Series* Vol. 7. ISBN: 970-36-0149-9. pp. 56-67. CIC-IPN, México 2004.
17. César Sandoval Hernández, **Giner Alor Hernández**, José Oscar Olmedo Aguirre. "Generación Dinámica de GUIs para la Invocación de Servicios Web publicados en nodos UDDI". In Alexander Gelbukh, Grigori Sidorov, Wilbert A. Olán Cristobal, José A. Vera Felix (Eds). *Recientes Avances en la Ciencia de la Computación en México. Research on Computing Science Series* Vol. 7. ISBN: 970-36-0149-9. pp. 68-79. CIC-IPN, México 2004.
18. **Giner Alor Hernández**, José Oscar Olmedo Aguirre. "BPIMS-WS: Service-Oriented Architecture for Business Processes Management. In Jesus Figueroa Nazuno, Alexander Gelbukh Khan, Cornelio Yañez Marquez, Oscar Camacho Nieto (Eds). *Advances in Artificial Intelligence, Computing Science and Computer Engineering. Research on Computing Science Series* Vol. 10. ISBN: 970-36-0194-4. ISSN: 1665-9899. pp. 255-264. CIC-IPN, México 2004.

19. **Giner Alor Hernández**, José Oscar Olmedo Aguirre. "Managed Web Services using WS-Manageability". In Alexander Gelbuck, Hiram Calvo (Eds). *Advances in Computing Science in Mexico. Research on Computing Science Series* Vol. 13. ISSN: 1665-9899. pp. 255-264. CIC-IPN, México 2005.
20. José Oscar Olmedo Aguirre, **Giner Alor Hernández**, Juan Miguel Gomez. "BPEL4WS Document Management with an Active Deductive XML Database". In Alexander Gelbukh, Cornelio Yáñez Márquez, Oscar Camacho Nieto (Eds). *Advances in Artificial Intelligence and Computer Science. Research on Computing Science Series* Vol. 14. ISSN: 1665-9899. pp. 211-220. CIC-IPN, México 2005.

En Revistas de Divulgación

21. **Giner Alor Hernández**, José Oscar Olmedo Aguirre. *"Sistema de Intermediación para el Comercio Electrónico B2B basado en Servicios Web"*. SPC Magazine. Perú, Año III Número 5. Enero-Febrero de 2004. pp. 10-14. Revista Editada por la Sociedad Peruana de Computación.
22. **Giner Alor Hernández**, José Oscar Olmedo Aguirre. *"Lenguajes para la Composición e Integración de Servicios Web"*. SPC Magazine. Perú, Año III Número 5. Enero-Febrero de 2004. pp. 15-20. Revista Editada por la Sociedad Peruana de Computación.

En Revista de Alto Impacto

23. **Giner Alor Hernández,** José Oscar Olmedo Aguirre. "BPIMS-WS: A Web Service-based Brokering Service for e-procurement in Supply Chains. IEEE Transactions on Network and Service Management. En revisión.

Referencias

[1] Curbera F., Duftler M., Khalaf R., Nagy W., Mukhi N., y Weerawarana S. “Unraveling the Web Services Web An introduction to SOAP, WSDL, and UDDI”, IEEE Internet Computing, 2002.

[2] Brent Sleeper, Bill Robins. “The Stencil Scope: Defining Web Services”. Stencil Group, pp. 2. 2001

[3] Adams, H., et al., “Custom Extended Enterprise Exposed Business Services Application Pattern Scenario,” IBM Alpha Works. Jan.1, 2003

[4] Chopra, S., P. Meindl, Supply Chain Management: Strategy, Planning and Operations, 2nd edition, Pearson Prentice-Hall, Upper Saddle River, New Jersey, 2004.

[5] Heath, N. "Taking Account of EDI", Management Accounting, February 1991, pp 14-15.

[6] Davis, D. “Third Parties Deliver,” Manufacturing Systems, 13, 66-68, 1995.

[7] Robins, G. “Pushing the limits of VMI,” Stores, pp 42-44, 1995.

[8] Mitchell, R.L. “Unilever Crosses the Data Streams,” Computerworld, December 17, 2001.

[9] Varon, E. “What you need to know about public and private exchanges,” CIO Magazine, September 1 2001.

[10] Turban, E., et. al., Electronic Commerce: A Managerial Perspective, Prentice Hall. 2002.

[11] Zilbert, A B., “A Comparative Study of Traditional Electronic Data Interchange versus Internet Electronic Data Interchange,” in D Colton, J Caouette, and B Raggad (Eds.), Proceedings ISECON, v 17 (Philadelphia): 501 AITP Foundation for Information Technology Education . 2000

[12] “Common Object Request Broker Architecture (CORBA/IIOP) Specification V 3.0.2”, Object Management Group, Final Report, 2002.

[13] “CORBA Component Model Specification V 3.0”. Object Management Group, Final Report, 2002.

[14] Markus Horstmann and Mary Kritland, "DCOM Architecture". MSDN Library, Specifications, Microsoft Corp. July 23, 1997.

[15] Merlin Hughes, et al. "Java Network Programming". Manning Publications, 1997.

[16] Chen, M., "Factors Affecting the Adoption and Diffusion of XML and Web Services Standards for E-business Systems ," International Journal of Human-Computer Studies, 58:3, 259-279, March 2003.

[17] W3C, Extensible Markup Language (XML) 1.0 (Second Edition), W3C Recommendation, http://www.w3.org/TR/REC-xml, October 6, 2000.

[18] W3C, XML Schema Part 0: Primer, W3C Recommendation, http://www.w3.org/TR/xmlschema-0/, May 2, 2001.

[19] W3C, Extensible Stylesheet Language (XSL) Version 1.0, W3C Recommendation, http://www.w3.org/TR/xsl/, October 15, 2001.

[20] W3C, Document Object Model (DOM) Level 2 Core Specification, Version 1.0, W3C Recommendation, http://www.w3.org/TR/2000/REC-DOM-Level-2-Core-20001113/13 November, 2000.

[21] SAX Project, About SAX, at http://www.saxproject.org/, June 2002.

[22] CPRF.ORG, CPFR Voluntary Guidelines, version 2.0, http://www.cpfr.org/documents /pdf/CPFR_Tab_7.pdf, June2002.

[23] Jouenne, T., Henkel-Eroski CPFR Pilot Case Study, http://www.cpfr.org/HenkelEroski.pdf, 2000.

[24] Koloszyc, G., "Retailers, Suppliers Push Joint Sales Forecasting," Stores, June 1998.

[25] Samtani, G. and D. Sadhwani, "Enterprise Application Integration and Web Services," in Web Services Business Strategies and Architectures, P. Fletcher and M. Waterhouse, Eds. Birmingham, UK: Expert Press, LTD, pp. 39-54, 2002.

[26] Vecchio, D. "Legacy Software: Junkyard Wars for Web Services?" Gartner Symposium/ITxpo, Orlando, FL, USA, 2001.

[27] S. Pallos Michael. "Service Oriented Architecture: A primer". Enterprise Application Integration Journal. December 2001. Pages 32-35.

[28] German Shegalov, Michael Gillmann, Gerhard Weikum. XML-enabled work? ows management for e-services across heterogeneous platforms. The VLDB Journal. pp. 91-103

[29] F. Leymann, D. Roller. Production Work?ow: Concepts and Techniques. Prentice-Hall, Englewood Cliffs, N.J.,USA.

[30] Simple Object Access Protocol Specification, SOAP Home, http://www.w3.org/TR/SOAP/

[31] Web Services Description Language Specification, WSDL Home, http://www.w3.org/TR/wsdl12

[32] UDDI, UDDI Version 3.0, Published Specification, July 19, 2002.

[33] J. Goepfert, M. Whalen. An Evolutionary View of Software as a Service. IDC White paper, http://www.idc.com, 2002.

[34] Munindar P. Singh, Michael N. Huhns. Service-Oriented Computing: Semantics, Processes, Agents, John Wiley & Sons, 2005.

[35] I. Jacobson. Object-Oriented Software Engineering: A Use Case Driven Approach. Addison-Wesley Professional. First Edition. ISBN: 0201544350. 30 June, 1992.

[36] Michael Huhns , Munindar P. Singh. Service-Oriented Computing: Key Concepts and Principles. IEEE Internet Computing. January-February 2005. pp 75-81.

[37] Mike P. Papazoglou, D. Georgakopoulos. Service Oriented Computing Communications of the ACM, October 2003.

[38] Ingolf H. Krüger, Reena Mathew. Systematic Development and Exploration of Service-Oriented Software Architectures. Proceedings of the Fourth Working IEEE/IFIP Conference on Software Architecture (WICSA'04). June 2004. pp. 177

[39] M. W. A. Steen, D. H. Akehurst, H. W. L. ter Doest, M. M. Lankhorst. Supporting Viewpoint-Oriented Enterprise Architecture. Proceedings of the Enterprise Distributed Object Computing Conference, Eighth IEEE International (EDOC'04). September 2004. pp. 201-211

[40] M. Champion, C. Ferris, E. Newcomer, and D. Orchard. Web Services Architecture W3C Working Draft, http://www.w3.org/TR/ws-arch/, November 2002.

[41] Mike P. Papazoglou, J. Yang Design Methodology for Web Services and Business Processes Proceedings of the 3rd VLDB-TES Workshop, Hong-Kong, 2002.

[42] I. Foster et al., "The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration," Globus, 17 Feb. 2002; http://www.globus.org/research/papers/ogsa.pdf.

[43] Martin C. Brown. The future of grid services: Initiatives, platforms, and companies. IBM Technical Report. September 2003.

[44] White Paper. Business Component Architecture: Unifying SOA and EDA. Fiorano Software, Inc. 2005

[45] I. Foster, C. Kesselman, J. M. Nick, and S. Tuecke. Grid Services for Distributed System Integration. IEEE Computer, 35(6), 2002.

[46] S. Tuecke, K. Czajkowski, I. Foster, J. Frey, S. Graham, and C. Kesselman. Grid Service Speci?cation. Technical Report. Open Grid Service Infrastructure WG, Global Grid Forum, 2002. Draft 5, November 5, 2002.

[47] White Paper. Delivering Real-time Enterprise: Enabling Real-Time Business Through A Service-Oriented and Event-Driven Architecture. TIBCO 2004.

[48] David A. Chappell. Enterprise Service Bus. O'Reilly Editorial. First Edition (June, 2004). ISBN: 0596006756

[49] Frank Leymann. Web Services: Distributed Applications without Limits -An Outline. Proceedings of Database Systems for Business, Technology and Web, 2003.

[50] Mike P. Papazoglou. Web Services and Business Transactions. World Wide Web Journal, vol.6, March 2003.

[51] Luis Felipe Cabrera, et. al. Web Services Atomic Transaction (WS-Transaction). BEA System, IBM, Microsoft Corporation November 2004, ftp://www6.software.ibm.com/software/developer/library/WS-AtomicTransaction.pdf

[52] Luis Felipe Cabrera, et. al. Web Services Coordination. WS-C Specification. BEA System, IBM, Microsoft Corporation. November 2004. ftp://www6.software.ibm.com/software/developer/library/WS-Coordination.pdf

[53] OASIS BTP. Business Transactions Protocol Specification version 1.0. Organization for the Advancement of Structured Information Systems. 3 June 2002.

[54] Nathan Chung-Nin Chung,Wen-Shih Huang,Tse-Ming Tsai and Seng-cho T.Chou. eXFlow:A Web Services-Compliant System to Support B2B Process Integration. Proceedings of the 37th Hawaii International Conference on System Sciences 2004.

[55] Neila Ben Lakhal, Takashi Kobayashi and Haruo Yokota. THROWS: An Architecture for Highly Available Distributed Execution of Web Services Compositions. Proceedings of the 14th International Workshop on Research Issues on Data Engineering: Web Services for E-Commerce and E-Government Applications (RIDE'04).

[56] I.Budak Arpinar, Boanerges Aleman-Meza, Ruoyan Zhang and Angela Maduko. Ontology-Driven Web Services Composition Platform. Proceedings of the IEEE International Conference on E-Commerce Technology.

[57] Mark Turner, Fujun Zhu, Ioannis Kotsiopoulos, Michelle Russell, David Budgen, Keith Bennett, Pearl Brereton, John Keane, Paul Layzell and Michael Rigby. Using Web Service Technologies to create an Information Broker: An Experience Report. Proceedings of the 26th International Conference on Software Engineering (ICSE'04).

[58] Uwe Radetzki and Armin B.Cremers. IRIS: A Framework for Mediator-Based Composition of Service-Oriented Software. Proceedings of the IEEE International Conference on Web Services (ICWS'04).

[59] Randy Howard and Larry Kerschberg. A Knowledge-based Framework for Dynamic Semantic Web Services Brokering and Management. Proceedings of the 15th International Workshop on Database and Expert Systems Applications (DEXA'04).

[60] Ananth Srinivasan and David Sundaram. Web Services for Enterprise Collaboration: A Framework and a Prototype. Proceedings of the 30th EUROMICRO Conference (EUROMICRO'04).

[61] JianJun Yu and Gang Zhou. Dynamic Web Service Invocation Based on UDDI. Proceedings of the IEEE International Conference on E-Commerce Technology for Dynamic E-Business (CEC-East'04).

[62] Rohit Aggarwal, Kunal Verma, John Miller and William Milnor. Constraint Driven Web Service Composition in METEOR-S. Proceedings of the 2004 IEEE International Conference on Services Computing (SCC'04).

[63] Tianyi ZANG, Wei JIE, Terence HUNG, Zhou LEI, Stephen J.Turner, Wentong CAI, Ming ZHU and Constantine Katsinis. An OGSI-compliant Grid Information Service - Its Architecture and Performance Study. Proceedings of the Seventh International Conference on High Performance Computing and Grid in Asia Pacific Region (HPCAsia'04).

[64] Mingkui Yang, Hongbing Liang and Bin Xu. S-WFMS: A Service-based Workflow Management System in Grid Environment. Proceedings of the 19th International Conference on Advanced Information Networking and Applications (AINA'05).

[65] M.Younas, K-M.Chao, R.Anane, S.Y.Yan, P.J.Lovett and A.N.Godwin. Grid Services Mediation. Proceedings of the 19th International Conference on Advanced Information Networking and Applications (AINA'05).

[66] Siegfried Benkner, Ivona Brandic, Gerhard Engelbrecht and Rainer Schmidt. VGE - A Service-Oriented Grid Environment for On-Demand Supercomputing. Proceedings of the Fifth IEEE/ACM International Workshop on Grid Computing (GRID'04).

[67] OECD. Open Services Markets Matter. Organization for Economic Cooperation and Development, October 2001.

[68] North American Industry Classification System, NAICS Homepage, http://www.naics.com/.

[69] United Nations Standard Products and Services Code, UNSPSC Homepage, http://www.unspsc.org/.

[70] RosettaNet, RosettaNet Homepage, http://www.rosettanet.org/

[71] Business Process Execution Language for Web Services. BPEL4WS 1.1 Specification. IBM May 5 2003.

[72] Deutsches Institut für Normung e.V. DIN 2342 Teil 1: Begriffe der Terminologielehre. Berlin: Deutsches Institut für Normung e.V.; 10/1992.

[73] Gruber TR. Toward Principles for the Design of Ontologies Used for Knowledge Sharing. International Journal of Human and Computer Studies 1995; 43(5/6): 907-928.

[74] Jeff Heflin, James Hendler, and Sean Luke. Reading Between the Lines: Using SHOE to Discover Implicit Knowledge from the Web. In AAAI-98 Workshop on AI and Information Integration. 1998.

[75] J. Hendler and D. L. McGuinness. The DARPA Agent Markup Language. IEEE Intelligent Systems. November-December, 2000. Vol. 15 Number 6 , pp. 67-73.

[76] Deborah L. McGuinness and Frank van Harmelen. OWL Web Ontology Language Overview. W3C Recommendation 10 February 2004

[77] Giner Alor Hernández, José Oscar Olmedo Aguirre. "Guaranteed Delivery and Processing in Web Services in BPIMS-WS using WS-RM". Proceedings of the 14th International Congress on Computing (CIC 2005). ISBN: 970-36-0266-5. pp 510-519.

[78] Web Service Choreography Description Language. WS-CDL Specification 1.0. WS-CDL Home, http://www.w3.org/TR/2004/WD-ws-cdl-10-20040427/

[79] Java Management Extensions Instrumentation and Agent Specification, v1.2. Sun Microsystems, Inc. October 2002

[80] Jeff Suttor, Norman Walsh, Kohsuke Kawaguchi. Java API for XML Processing JAXP) version 1.3. Sun Microsystems, Inc. Final Release, September 1, 2004

[81] Giner Alor Hernández, César Sandoval Hernández, José Oscar Olmedo Aguirre. Descubrimiento Dinámico de Servicios Web en nodos UDDI mediante USML. Research on Computing Science Series Vol. 7. ISBN: 970-36-0149-9. Pág.56-62

[82] Matthew J. Duftler, Nirmal K. Mukhi, Aleksander Slominski and Sanjiva Weerawarana. Web Services Invocation Framework (WSIF). IBM T.J. Watson Research Center. August 9, 2001

[83] Matthew J. Duftler, Paul Fremantle. Java APIs for WSDL (JWSDL) JSR110: JWSDL Final Release Version 1.0. IBM Corporation March 21, 2003

[84] Ruslan Bilorusets et al. Web Services Reliable Messaging Protocol (WS-ReliableMessaging). BEA Systems Inc., IBM Corporation, Microsoft Corporation, TIBCO Software Inc March 13, 2003

[85] Martin Gudgin, Marc Hadley. Web Services Addressing 1.0 – Core. W3C Candidate Recommendation 17 August 2005

[86] Phil Goodwin and Nick Kassem. SOAP with Attachments API for Java (SAAJ) Specification v1.2. Sun Microsystems, Inc. October 2003, Revision 01

[87] Marc Hadley and Roberto Chinnici. Java API for XML-Based RPC (JAX-RPC) Specification 2.0. Sun Microsystems, Inc. June 10, 2004

[88] César Sandoval Hernández, Giner Alor Hernández, José Oscar Olmedo Aguirre. "Generación Dinámica de GUI's para la invocación de servicios Web publicados en nodos UDDI". Research on Computing Science Series Vol. 7. ISBN: 970-36-0149- 9. Pág. 48-55.

[89] The Apache Jakarta Project. Apache Jakarta Tomcat Homepage http://jakarta.apache.org/tomcat/

[90] IBM UDDI Business Production Registry, https://uddi.ibm.com/ubr/registry.html

[91] IBM UDDI Business Test Registry, https://uddi.ibm.com/testregistry/registry.html

[92] Microsoft UDDI Business Registry Node, http://uddi.microsoft.com/default.aspx

[93] Microsoft UDDI Business Production Registry, http://uddi.microsoft.com/

[94] SAP UDDI Business Registry, http://uddi.sap.com/

[95] Quick UDDI Registration Service, qUDDI Home, http://www.quddi.com/

[96] Eric Armstrong, et al. "The Java Web Services Tutorial". SunMicrosystems Inc. February 2003.

[97] Keith Ballinger, Peter Brittenham, Ashok Malhotra, William A. Nagy, Stefan Pharies. Web Services Inspection Language 1.0. Specification. November 2001.

[98] Liang-Jie Zhang, Haifei Li, Henry Chang, Tian Chao, "XML-based Advanced UDDI Search Mechanism for B2B Integration". Proceedings of the Fourth IEEE International Workshop on Advanced Issues of E-Commerce and Web-Based Information Systems (WECWIS'02). June 26 - 28, 2002.

[99] S. McIlraith, T. C. Son and H. Zeng. Semantic Web Services. IEEE Intelligent Systems. Special Issue on the Semantic Web. Vol March/April 2001 (Vol. 16, No. 2). ISSN: 1094-7167. pp. 46-53

[100] M. Dumas, J. O'Sullivan, M. Heravizadeh, D. Edmond and A. Hofstede. Towards a Semantic Framework for Service Description. Proceedings of the IFIP Conference on Database Semantics. Kluwer Academic Publishers, April 2001.

[101] M. Paolucci, T. Kawmura, T. Payne and K. Sycara. Semantic Matching of Web Services Capabilities. Proceedings of the First International Semantic Web Conference. 2002. Lecture Notes in Computer Science; Vol. 2342. ISBN: 3-540-43760-6. Pages: 333 - 347

[102] K. Sivashanmugam, K.Verma, A. Sheth and J. Miller. Adding semantics to web services standards. Proceedings of the First International Conference on Web Services (ICWS 03). 2003.

[103] Rama Akkiraju, Richard Goodwin, Prashant Doshi and Sascha Roeder. A Method for Semantically Enhancing the Service Discovery Capabilities of UDDI. Proceeding of the Eighteenth International Joint Conference on Artificial Intelligence. 2003.

[104] Prashant Doshi, Richard Goodwin, Rama Akkiraju and Sascha Roeder. Extending Semantic Matching for Application in Business Process Integration

[105] Ali ShaikhAli, Omer F. Rana, Rashid Al-Ali, David W. Walker. UDDIe: An extended Registry for Web Services. Proceedings of the 2003 Symposium on Applications and the Internet Workshop (SAINT-W 03). IEEE Press.

[106] Youzhi Xu, Jie Shen, Zhimin Chen. Ontology-based Information Retrieval of Web Services in Virtual Enterprise. Proceedings of the 2004 IEEE International Conference on Services Computing (SCC 04). IEEE Press.

[107] Debra VanderMeer, Anindya Datta, Kaushik Dutta, Helen Thomas, Krithi Ramamritham and Shamkant B. Navathe. FUSION: A System Allowing Dynamic Web Service Composition and Automatic Execution. Proceedings of the IEEE International Conference on E-Commerce (CEC'03).

[108] Mauro Migliardi and Roberto Podesta. Performance Improvement in Web Services Invocation Framework. Proceedings of the 18th International Parallel and Distributed Processing Symposium (IPDPS'04).

[109] Shinichi Nagano, Tetsuo Hasegawa, Akihiko Ohsuga and Shinichi Honiden. Dynamic Invocation Model of Web Services Using Subsumption Relations. Proceedings of the IEEE International Conference on Web Services (ICWS'04).

[110] M.Yu, A.Taleb-Bendiab and D.Reilly. A Polyarchical Middleware for Self-Regenerative Invocation of Multi-Standard Ubiquitous Services. Proceedings of the IEEE International Conference on Web Services (ICWS'04).

[111] Ian Gorton, Justin Almquist, Kevin Dorow, Peng Gong and Dave Thurman. An Architecture for Dynamic Data Source Integration. Proceedings of the 38th Hawaii International Conference on System Sciences – 2005.

[112] "Programming BPEL". Collaxa Developer's Guide 2003.

[113] R. Conradi, C. Fernstrom, A. Fuggetta, and B. Snowdown. Towards a Reference Framework for Process Concepts. In Proc. EW SPT'92, 2nd European Workshop on Software Process Technology, Trondheim, Norway, Sept. 1992

[114] F. Leymann, D. Roller. Workflow-based application. IBM System Journal, vol.36, No.1, pp102-123, 1997

[115] Workflow Management Coalition. Interface 1: Process Definition Interface. WfMC TC-1016. August 1996

[116] Kunal Verma, Rama Akkiraju, Richard Goodwin, Prashant Doshi and Juhnyoung Lee. On Accommodating Inter Service Dependencies in Web Process Flow Composition. In Working Notes of the Spring Symposium on Semantic Web Services, AAAI, pp., Stanford, California, March 22-24, 2004.

[117] Jacky Estublier and Sonia Sanlaville. Business Processes and Workflow Coordination of Web Services. Proceedings of the IEEE International Conference on e-Technology, e-commerce and e-Service (EEE-05), Hong Kong 29 March, 1 April 2005.

[118] R. Anane, K-M Chao and Y. Li. Hybrid Composition of Web Services and Grid Services. Proceedings of the IEEE International Conference on e-Technology, e-Commerce and e-Service (EEE'05). pp. 426-431

[119] Stefan Tai, Rania Khalaf, and Thomas Mikalsen. Composition of Coordinated Web Services. IFIP International Federation for Information Processing 2004. In H.-A. Jacobsen (Ed.): Middleware 2004, Lecture Notes on Computer Science. Vol. 3231, pp. 294–310, 2004.

[120] Xiulan YU, Long ZHANG, Ying LI, Ying CHEN. WSCE: A Flexible Web Service Composition Enviroment. In Proceedings of the IEEE International Conference on Web Services. (ICWS 04)

[121] Jiamao Liu, Juntao Cui, Ning Gu. Composing Web Services Dynamically and Semantically. In Proceedings of the IEEE International Conference on E-Commerce Technology for Dynamic E-Business (CEC-East04).

[122] Boualem Benatallah, Marlon Dumas and Quan Z. Sheng. The Self-Serv Environment for Web Services Composition. IEEE Internet Computing. January-February 2003. pp. 40-48.

[123] "Unified Modeling Language 1.5 Specification" Object Management Group, Final Report, 2002.

[124] "Scalable Vector Graphics (SVG) 1.1 Specification". W3C Recommendation January 2003.

[125] Mo Klein and Francois Besson. Business Activity Monitoring: The End-Game of the Real-Time Enterprise. Business Integration Journal. December 2003. pp. 27-29

[126] A. Geppert and D. Tombros. Logging and Post-Mortem Analysis of Workflow Executions based on Event Histories. Proceedings of the Third International .Conference on Rules in Database Systems (RIDS), Lecture Notes on Computer Science Vol. 1312, Springer Verlag, Heidelberg, Germany, pages 67-82, 1997.

[127] P. Koksal, S. N. Alpinar and A. Dogac. Workflow History Management. ACM Sigmod Record 27(1): 67-75, 1998.

[128] P. Muth, J. Weissenfels, M. Gillmann and G. Weikum. Workflow History Management in Virtual Enterprises Using a Light-Weight Workflow Management System. Proceedings of the Ninth International Workshop on Research Issues on Data Engineering, 23 - 24 March, 1999, Sydney, Australia, pp. 148-155

[129] Jun-Jang Jeng, Josef Schiefer and Henry Chang. An Agent-based Architecture for Analyzing Business Processes of Real-Time Enterprises. Proceedings of the Seventh IEEE International Enterprise Distributed Object Computing Conference (EDOC'03). IEEE Press. p. 86.

[130] Josef Schiefer, Beate List, Robert M. Bruckner. Process Data Store: A Real-time Data Store for Monitoring Business Processes. In Vladimír Marík, Werner Retschitzegger, Olga Stepánková (Eds.). Database and Expert Systems Applications, 14th International Conference, DEXA 2003. Lecture Notes in Computer Science 2736 Springer 2003, ISBN 3-540-40806-1. pp. 760-770.

[131] N. R. Jennings, P. Faratin, M. J. Johnson, P. O'Brien and M. E. Wiegand. Using intelligent agents to manage business processes. Proceedings of the First International Conference on The Practical Application of Intelligent Agents and Multi-Agent Technology (PAAM96), pp.345-360.London, UK, 1996.

[132] Yuhong Yan, Zakaria Maamar, Weiming Shen. Integration of Workflow and Agent Technology for Business Process Management. In Proceedings of the Sixth International ACM Confererence on Computer Supported Cooperative Work in Design. 2001. London, Ontario, Canada.

[133] Xiaohui Zhao, Chengfei Liu and Yun Yang. Web Service Based Architecture for Workflow Management Systems. In F. Galindo et. al. (Eds). 15th International Conference on Database and Expert Systems Applications 2004, Lecture Notes in Computer Science 3180, pp. 34-43, 2004.

[134] Jean-Philippe Martin-Flatin, Pierre-Alain Doffoel and Mario Jeckle. Web Services for Integrated Management: A Case Study. In L.-J. Zhang and M. Jeckle (Eds). European Conference on Web Services 2004. Lecture Notes in Computer Science 3250, pp. 239-253, 2004.

[135] Jun-Jang Jeng, Henry Chang and Jen-Yao Chung. A Policy Framework for Business Activity Mnagement. In Proceddings of the IEEE International Conference on E-commerce (CEC' 03). IEEE Press.

[136] German Shegalov, Michael Gillmann and Gerhard Weikum. XML-enabled workflow management for e-services across heterogeneous platforms. The VLDB Journal 10:91 –103 (2001)/Digital Object Ide nti?er (DOI) 10.1007/s007780100038. pp. 91-103.

[137] Bullen, G. et al. Open Management Interface. Organization for the Advancement of Structured Information Standards. Version 1.0. 2001; http://www. webmethods.com

[138] Web Services Distributed Management (WSDM). Organization for the Advancement of Structured Information Standards. OASIS News. March 09, 2005.

[139] Samtani, G. and D. Sadhwani, "Return on Investment and Web Services," in Web Services Business Strategies and Architectures, P. Fletcher and M. Waterhouse, Eds. Birmingham, UK: Expert Press, LTD, pp. 9-24, 2002.

[140] Supply-Chain Council, Supply-Chain Operations Reference Model, version 6.0, June 10, 2003.

[141] Sriram, V. and S. Banerjee, "Electronic Data Interchange: Does Its Adoption Change Purchasing Policies and Procedures?" International Journal of Purchasing and Materials Management, 30:1, 31-40, 1994.

[142] Kaplan, S. and M. Sawhney, 2000, "E-Hubs: The New B2B Marketplaces," Harvard Business Review, May-June 2000.

[143] Gavirneni, S., R. Kapuscinski, and S. Tayur, "Value of Information in Capacitated Supply Chains," Management Science, 45:1, 16-24, 1999.

[144] Kekre, S., T. Mukhopadhyay, K. Srinivasan, "Modeling Impacts of Electronic Data Interchange Technology," in Quantitative Models for Supply Chain Management, S. Tayur, R. E. Ganeshan, and M. J. Magazine, Eds. Boston: Kluwer Academic Publishers, 1999.

[145] Lee, H. L., K. C. So, and C. S. Tang, "The Value of Information Sharing in a Two-Level Supply Chain". Management Science. 46:5, 626-643, 2000.

[146] Ahmad, S. and R. G. Schroeder, "The Impact of Electronic Data Interchange on Delivery Performance," Production and Operations Management, 10:1, 16-30, 2001.

[147] Walton, S. V. and A. S. Marucheck, "The Relationship Between EDI and Supplier Reliability," International Journal of Purchasing and Materials Management, pp 30-35, Summer 1997.

[148] Cachon, G. P. and M. Fisher, "Supply Chain Inventory Management and the Value of Information," Management Science, 46:8, 1032-1048, 2000.

[149] Meadows Bill, et al. Universal Business Language 1.0. OASIS Standard Specification. September15 2004.

[150] Hallam-Baker P., et al. Web Services Security: SAML Token Profile. OASIS Standard. December 1 2004.

[151] Web Services Security Language, WS-Security Specification Version 1.0, IBM Corp., Microsoft Corp., VeriSign Inc. April 5, 2002.

[152] Web Services Trust Language, WS-Trust Specification Version 1.0, IBM Corp., Microsoft Corp., RSA Security Inc., VeriSign Inc. December 18, 2002.

[153] Web Services Security Policy Language, WS-SecurityPolicy Specification Version 1.0, IBM Corp., Microsoft Corp., RSA Security Inc., VeriSign Inc. December 18, 2002.

[154] Tan, Y. S., B. Topol, V. Vellanki, and J. Xing, Business Service Grid, Part 1: Introduction, http://www-106.ibm.com/developerworks/ibm/library/i-servicegrid/, February 2003.

[155] A.Ankolenkar,M.Burstein,et al.,"DAML-S:Web Service Description for the Semantic Web". The First International Semantic Web Conference, Stanford, 2001.

[156] Sean Bechhofer, et al. OWL-S. Ontology Web Language for Services. Reference W3C Recommendation 10 February 2004.

[157] Gudgin M., Hadley M., Mendelsohn N. Moreau J., Nielsen H. Simple Object Access Protocol Recommendaion 1.2 (SOAP), W3C, 2003. http://www.w3.org/TR/soap12-part1/

[158] Data Universal Numbering System, D-U-N-S Homepage, http://www.dnb.com/

[159] Thomas Register, Thomas Register Homepage, http://www.thomasregister.com/

[160] International Organization for Standardization (ISO) - Language codes, ISO Homepage, http://www.iso.ch/

[161] Standard Industrial Classification, SIC Homepage, http://www.sec.gov/info/edgar/siccodes.htm