INSTITUTO DE COMPUTACIÓN - FACULTAD DE INGENIERÍA
UNIVERSIDAD DE LA REPÚBLICA

Tesis de Maestría en Ingeniería en Computación

Interoperabilidad entre Sistemas de Workflow utilizando eXtensible Markup Language

ING. ADELA CASERO SOULIER

SUPERVISOR:

DR. HERMANN STEFFEN

FEBRERO 2004

Índice

Resumen

Debido a que los ambientes de los sistemas de información actuales son heterogéneos, ampliamente distribuidos, y dada la creciente necesidad de integración entre organizaciones, entorno al comercio electrónico, la idea de empresa virtual y/o procesos ínter-organizacionales, -facilitados por la expansión de Internet- existe un mayor requerimiento de interoperabilidad, es decir que los distintos sistemas de información puedan cooperar intercambiando datos y sincronizando las ejecuciones de sus aplicaciones.

Está interoperabilidad involucra la operación y el desarrollo de procesos de negocios que se ejecutan a través y entre organizaciones, donde encontramos que los sistemas de workflow son el soporte tecnológico obvio para lograrlo.

La propuesta es considerar a los sistemas de workflow, como marco para esta interoperabilidad, considerándolos el sustento de los "procesos virtuales" de la "empresa virtual", donde cada empresa participa integrando sus servicios, usando potencialmente distintos sistemas de workflow.

Debemos lograr una verdadera interoperabilidad, no solamente a nivel de integración de aplicaciones, sino además de integración de información. Por lo que es necesario que las organizaciones que intervienen adhieran a una semántica compartida, y encontramos en el estándar emergente XML, eXtensible Markup Language, un lenguaje simple e independiente de la implementación y con costo efectivo, para representar la información estructurada compartida entre las organizaciones.

Se plantea como objetivo realizar una modelización, para formalizar esta interoperabilidad, basándose en la interoperabilidad entre sistemas de workflow y en la utilización del estándar de intercambio de XML.

La modelización definida incluye formas de definir, activar, y monitorear los procesos de negocios de la empresa virtual, comercio electrónico o procesos ínter organizacionales, además de soportar actividades de coordinación y consultas. Incluyendo consideraciones de seguridad, calidad de servicio y garantías de ejecución.

1. Introducción

Para conocer las necesidades y los problemas relacionados con la integración entre organizaciones, entorno al comercio electrónico, la idea de empresa virtual y/o procesos ínter-organizacionales, se analizará la evolución de las relaciones y se modelarán estas relaciones a través de las nociones de comunidad de comercio, empresa virtual y proceso virtual.

1.1 Evolución de las relaciones entre organizaciones

Históricamente, las organizaciones han encontrado formas de trabajar en conjunto con proveedores, clientes, asociados, etc. En los comienzos, se ha logrado sin el soporte de la computación, los miembros de cada organización se reunían en un espacio común para trabajar sobre un proyecto en particular.

Solo algunas organizaciones grandes podían afrontar la inversión requerida para mapear éstos vínculos comerciales a una infraestructura electrónica, las que en un principio, fueron basadas usualmente en líneas dedicadas costosas, mainframes y desarrollos ad hoc. Incluso para estas compañías, el costo sólo podía ser justificado sobre ciertos niveles de comercio y con socios que debían poder costear la infraestructura, y con los cuales, la cooperación debería continuar a largo plazo.

Dentro de estas prácticas, se encuentra la implementación de cadenas de suministro o stock a través y entre organizaciones, para ello se ha utilizado, a partir de los ’80, el estándar de intercambio electrónico de datos (EDI), como forma de integración de datos y automatización de transacciones, éste ha proporcionado un régimen de mensajería confiable para soportar relaciones comerciales entre organizaciones.
Estas relaciones comerciales son efectuadas a través de mensajes que contiene objetos de negocio estándares (documentos como facturas, órdenes de compra, fondos electrónicos), que son tratados como entradas al sistema de información de la organización receptora.
EDI ha sido bueno para soportar aplicaciones transaccionales de relaciones de comercio seguras, con grandes volúmenes. Sin embargo la experiencia muestra que es costoso en términos de instalación o puesta en marcha y que de alguna manera es inflexible una vez en operación.

Muchas de las barreras de acceso a estas prácticas, dadas por el costo y sobrecarga que involucran, han sido efectivamente eliminadas por el abaratamiento del hardware y por el surgimiento de Internet, el cuál ofrece un medio de comunicación de bajo costo y posibilita iniciar los emprendimientos de integración de una manera más dinámica. Estos factores han disparado el surgimiento de interés en las integraciones entre organizaciones, que han dado origen en los años recientes, al comercio electrónico y negocio electrónico.

Entendemos por negocio electrónico, (E-Business) el uso de Internet para conducir las transacciones de negocios entre proveedores, vendedores y otros partes para mejorar el servicio al cliente, reducir costos y abrir nuevos canales para incrementar los beneficios de los inversores. Llamamos comercio electrónico al marketing, venta y compra de productos y servicios a través de Internet.

La tecnología de Internet, ha transformado el modelo de negocio tradicional. La emergente economía web requiere organizaciones ágiles, que puedan trabajar más directamente con proveedores y clientes y puedan responder más rápido e inteligentemente al cambio.

El modelo de negocio actual demanda integraciones más dinámicas y flexibles entre los distintos participantes de los procesos de negocio.
La corporación integrada verticalmente, donde una única organización se expande y controla todas las áreas de negocio, se está convirtiendo en una excepción más que en una norma y en el competitivo mercado global, las presiones en el negocio, - proliferación de canales, el aumento de la expectativa de los clientes, competencias, más rápida transformación de los productos-, han incrementado el énfasis en cómo las organizaciones operan y se relacionan entre sí.

Todo esto ha producido un incremento en la necesidad que tienen las empresas de expandir sus procesos de negocios a través de los límites de sus organizaciones, que tiene como objetivo el ahorro de tiempo de transmisión, ganancias en la calidad de datos y mejoras en sus productos y servicios.

Hoy en día, las integraciones entre organizaciones, como el comercio electrónico, son prácticas ya establecidas las cuales usan tecnología de información y comunicación para manejar sus transacciones de negocios diarias. Se desarrolla en una amplia variedad de formas: cadenas de suministros, ventas por catálogo, subastas, ventas directas, ventas al por menor, marketing, servicios personalizados, etc. De todas las variedades el comercio electrónico es el más exitoso, lo cual es fácilmente explicable por el volumen de comercio involucrado y la rápida adopción de tecnología por las organizaciones.

Analizando con más detalle las integraciones entre organizaciones actuales, cobra mayor importancia la integración basada en la noción de comunidad de comercio, donde distintas compañías reúnen sus servicios para ofrecer un producto o servicio de mayor valor agregado y más complejo.
La comunidad de comercio representa un gran desafío, dado por la naturaleza dinámica y el mayor grado de cooperación requerido por las partes involucradas, además, es la aproximación más prometedora para pequeñas y medianas empresas y el modo natural para Internet.

Este tipo de integración se ve actualmente favorecido por el surgimiento de una variedad de tecnologías de información, que mejoran la infraestructura necesaria para construir comunidades de comercio altamente coordinadas, cada una con la habilidad de operar como una empresa virtual, como determina el nuevo modelo de negocio.

1.2 El modelado de las relaciones entre organizaciones

Para describir las integraciones entre organizaciones, se utilizarán las nociones de proceso de negocio virtual, empresa virtual y comunidad de comercio, basándonos en que la mayoría de las formas de relacionamiento entre organizaciones, pueden ser modeladas usando procesos de negocio.

Los procesos de negocio pueden ser tratados como un conjunto de procedimientos y reglas expresadas en un lenguaje más o menos formal, en una forma gráfica o textual, que describen sus pasos, los cuales deben ser tomados en un cierto orden para cumplir una tarea compleja o una meta de negocio.
Son a menudo usados para representar, modelar y formalizar las actividades más relevantes en una organización.

Los procesos de negocio son usados para documentar los procesos diarios y como base para la automatización y optimización de dichos procedimientos. Un ejemplo de estas tareas es abrir una nueva cuenta en un banco, obtener un crédito, comprar una computadora, encontrar la localización actual para una parcela, reabastecimiento de comercios, etc.

Un **proceso de negocio virtual** es usado para definir una meta concreta de negocio y describe las correspondientes actividades. A diferencia de un proceso de negocio normal, en uno virtual, la definición y activación no son referidas a una entidad organizacional única.
Usa información y tecnología de comunicación para permitir al proceso de negocio, ir más allá los límites corporativos.

La noción de proceso de negocio virtual permite calificar e identificar correctamente las actividades a realizar, por ejemplo, cómo las compañías participantes interactuarán entre sí. Permite además determinar cuando las actividades se extienden a través de los límites corporativos y dónde y cuáles son las interfases entre estos límites.

Las figuras 1 y 2 representan dos ejemplos de procesos de negocios virtuales. En ambos casos el proceso de negocio virtual aparece como un proceso normal excepto por el factor de que algunos pasos del proceso se corresponden con actividades individuales o subprocesos de diferentes organizaciones. Se puede ver al proceso virtual como un meta-proceso donde sus bloques de construcción son provistos por las compañías participantes.

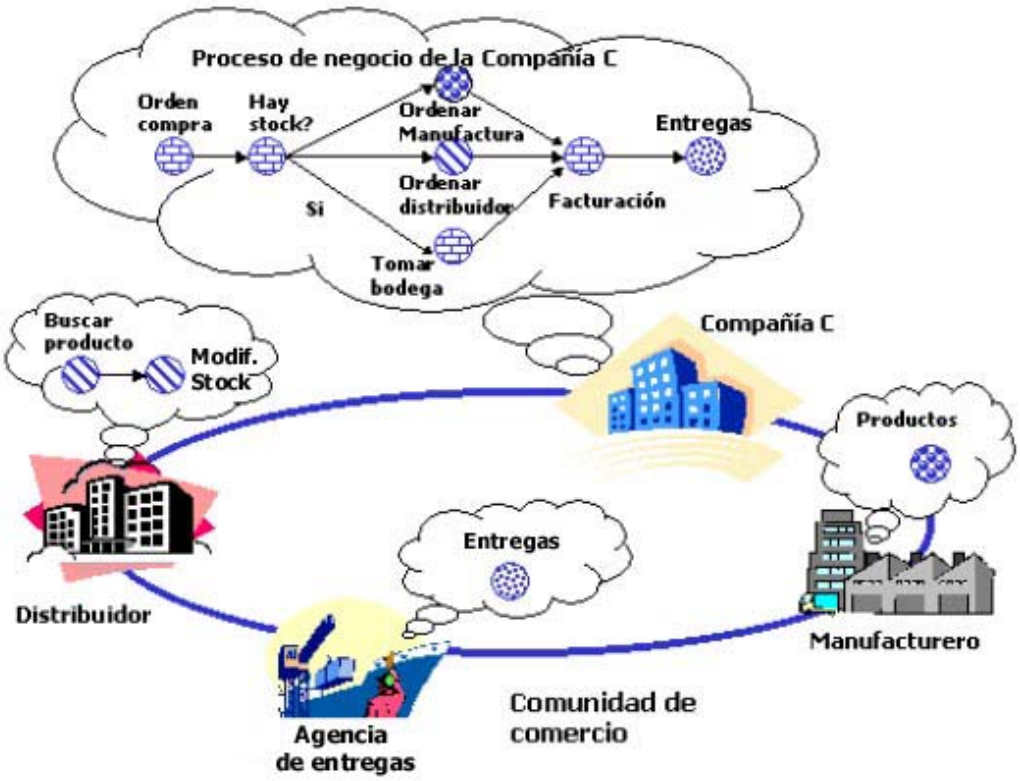


Figura 1: una compañía incorpora un proceso de negocio virtual como parte de su propio proceso de negocio.

En el ejemplo de la figura 1, una compañía incorpora como parte de su propio proceso actividades que serán realizadas por otras compañías. La compañía actúa como un distribuidor que comercia mercaderías que posee un su stock o que obtiene, directamente, de otros distribuidores o de los manufactureros. Utiliza además una cuarta compañía para la entrega de mercaderías. Como muestra el ejemplo, el proceso de negocio virtual se encuentra en la compañía C dado que es la única que alcanza todas las compañías participantes. Cabe destacar que no es el único proceso involucrado, el distribuidor, manufacturero y la agencia de entregas deben implementar también sus procesos de negocios.

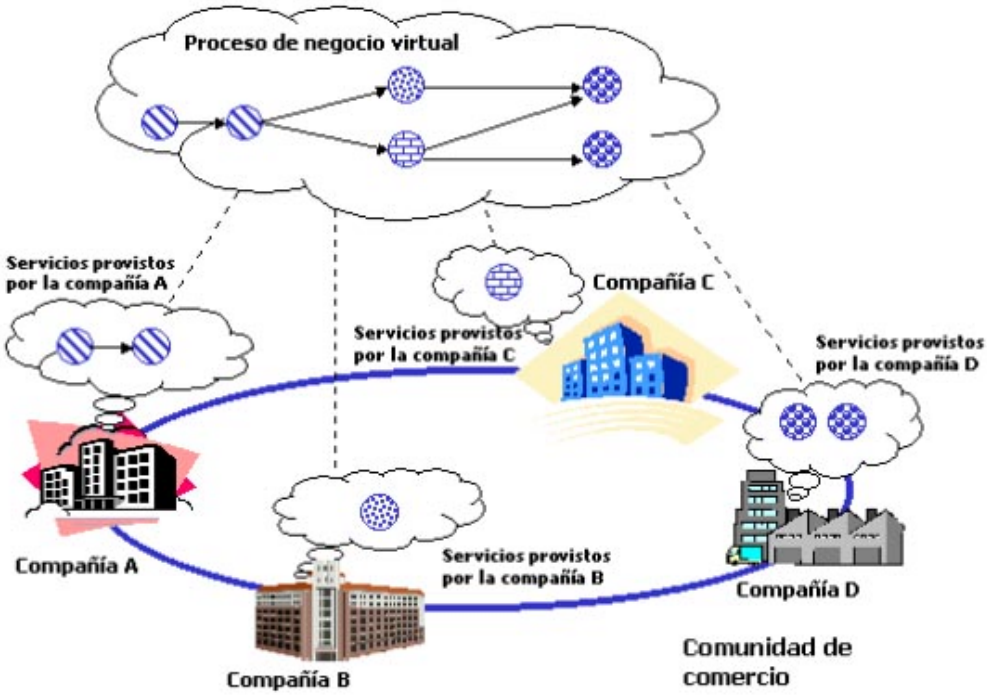


Figura 2: un proceso de negocio virtual como un proceso de valor agregado que combina servicios de diferentes compañías.

Estos procesos no tienen límite en el nivel de anidamientos de procesos, es decir un proceso componente puede estar formado, en si mismo, por un proceso de negocio virtual.

Para construir un proceso de negocio virtual, no necesitamos conocer en detalle los procesos que lo componen, solo necesitamos conocer las "interfaces" de los procesos componentes para incorporarlos en el proceso de negocio virtual.

Otro aspecto importante a considerar es, que los procesos de negocio virtuales son independientes del lenguaje, tanto del que utilizan para representar el proceso virtual, como del que usan los procesos independientes. Como solo necesitamos conocer las interfaces pueden estar formados por representaciones enteramente distintas.

Solo necesitamos un mecanismo para hacer pública su interfaz, junto con una especificación de las características del proceso componente y asegurar que la misma cumple con un formato de acuerdo preestablecido. Cómo lo que se necesita es la interfaz, las empresas participantes no necesitan exponer los detalles internos de sus actividades que por lo general son información propietaria.

Definir las interfaces no es un problema significativo ya que son usualmente especificadas como parte del contrato de relacionamientos entre compañías. Lo que sí sería importante es que se utilizara un método y forma estandarizada para representar estas interfaces y los contratos que las contiene, de forma de permitir que estas interacciones sean más fáciles de establecer y mantener, además de permitir que nuevas empresas puedan incorporarse.

Un proceso de negocio virtual no puede ser definido sin un contexto, es decir sin un conjunto de reglas, metas, requerimientos, restricciones, información y recursos. Este contexto es a lo que se llama **empresa virtual**.
Identifica y da significado al proceso de negocio virtual ubicándolo en el contexto de metas concretas y de un ambiente bien definido.

La empresa virtual es un factor determinante en término de su factibilidad, todo lo que no puede ser resuelto a nivel de los procesos componentes, debe ser resuelto a nivel de la empresa virtual, en el contexto del proceso virtual. Nombrando a este contexto explícitamente permite por ejemplo especificar que hacer en caso de excepciones a nivel del proceso virtual.

Alternativamente, una **empresa virtual** es definida como la empresa, cuyos procesos de negocios son procesos de negocios virtuales, independientemente si existe o no una empresa real detrás del la empresa virtual o no.

En el ejemplo de la figura 1, la empresa virtual es parte de la compañía C mientras que en el ejemplo de la figura 2 la empresa virtual es verdaderamente virtual en el sentido de que no necesariamente existe una organización física detrás de ella.

Para identificar o definir una empresa virtual es más directo como en el caso del ejemplo de la figura 1 y en otros casos requiere mayor esfuerzo desde el punto de vista organizacional y legal, como para el ejemplo de la figura 2. Por ejemplo quien es el propietario de la información sobre el proceso virtual, (uno de los participantes, todos, quien administra la información, quien tiene derecho a vender esta información como valor agregado del servicio).

Una vez que definimos que hacer (el proceso de negocio virtual) y el contexto en el cual se realiza (empresa virtual) necesitamos definir los actores. Para esto usamos la noción de **comunidad de comercio** que se describe como el conjunto de compañías participantes en la empresa virtual.
Provee el marco de trabajo de cooperación para las compañías participantes y establece cuáles de éstas son las partes involucradas en cada paso del proceso de negocio virtual.

Alternativamente puede ser definida como el conjunto de compañías las cuales proveen los bloques de construcción para los procesos de negocio virtuales. Cada empresa virtual tiene asociada una comunidad de comercio, y puede ejecutar varios procesos de negocio virtuales. Definir la comunidad de comercio es el primer paso para definir los derechos de acceso, responsabilidades, autenticaciones, mecanismos de encriptación y la configuración del sistema distribuido subyacente.

De todo lo anterior se desprende que, los procesos de negocio virtual, la empresa virtual y la comunidad de comercio son enfoques muy poderosos para interpretar e identificar un rango amplio de prácticas de relaciones entre organizaciones.

Podemos aplicarlas en el caso de ventas, donde una compañía puede proveer un producto más sofisticado, tercerizando aspectos de la operación, que no son centrales a sus actividades. Un ejemplo común son empresas que ofrecen un producto (libros, CD, flores) sin manejar los productos en si mismos (producirlos, almacenarlos o entregarlos). La mayor parte del manejo o manipulación es dejado a compañías que proveen servicios especializados, que permiten reducir significantemente los costos operacionales. La empresa virtual captura naturalmente estos escenarios teniendo los servicios de distribución y entregas incorporados como actividades en el proceso de negocio de la compañía que vende el producto, como lo veíamos en el ejemplo de la figura 1.

Los mismos escenarios pueden ser creados para ventas al por menor, servicios personalizados, y cadenas de suministros de comercio electrónico. Ejemplo de ventas: comprar una computadora usando una interfase en línea, que permite elegir sus componentes de un conjunto de componentes básicos. En este caso estamos interesados en implementar mecanismos que permitan a la compañía tercerizar partes de sus servicios. En el caso de servicios personalizados estamos interesados en escenarios donde una compañía cubre la brecha entre varias otras compañías y provee servicios de mayor valor agregado utilizando otros servicios provistos por otras empresas.

1.3 Ciclo de vida de un proceso de negocio virtual

El ciclo de vida se centra en el proceso, que es el factor determinante que da forma a la colaboración entre los participantes de la comunidad de comercio.

El ciclo de vida comienza con la comunidad de comercio que provee un número de servicios bien definidos los cuales se dan a conocer a los participantes. Usando estos servicios como bloques de construcción, se define un proceso de negocio virtual.

Una vez definido, puede ser ejecutado dentro del contexto de la empresa virtual. Basado en este contexto, la ejecución del proceso puede ser observada y el resultado de estas observaciones puede ser usado para mejorar la ejecución del proceso de negocio virtual.
El proceso de negocio virtual sirve como base para la comunicación y coordinación de los participantes.

Las interacciones pueden ser clasificadas en:

Interacciones entre la comunidad de comercio y el proceso de negocio virtual:

Involucra mecanismos donde las compañías pueden publicar sus servicios y otras compañías pueden buscarlos y finalmente incorporarlos a sus procesos de negocios. Los servicios deben aparecer en catálogos, en los cuales, las entradas deben representar las actividades que puedan ser incorporadas en un proceso de negocio, deben proveer información importante sobre estas actividades de manera que puedan ser incorporados directamente desde el catálogo al proceso de negocio virtual.

Interacciones entre el proceso de negocio virtual y la empresa virtual:

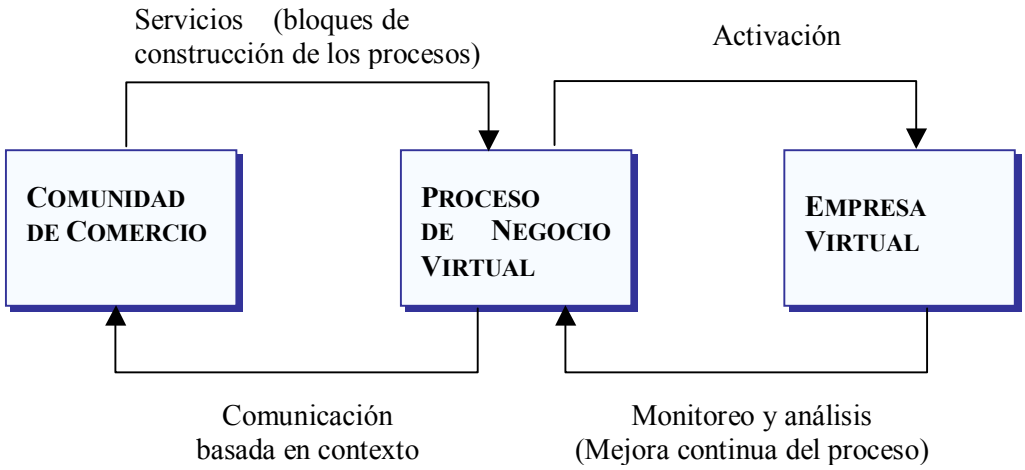
Involucra la ejecución del proceso y el subsanar las diferencias de interoperabilidad entre los sistemas participantes. Establecimiento contexto.

Interacciones entre la empresa virtual y el proceso de negocio virtual:

Permite observar el comportamiento del proceso cuando ejecuta para mejorar su diseño y obtener información de la empresa virtual, requiere un seguimiento en tiempo de ejecución, demoras, análisis de flujo, requerimientos de calidad de servicio, identificación de cuellos de botella, etc. Utilizando esta información el proceso de negocio virtual puede ser continuamente mejorado.

Interacciones entre el proceso de negocio virtual y la comunidad de comercio:

Permite a los integrantes de la comunidad de comercio establecer comunicaciones con otros integrantes basados en sus responsabilidades en el proceso de negocio virtual. Sería conveniente que puedan comunicarse con quien pueda ser responsable de una actividad dada del proceso virtual. Por lo que se debe resolver, quien es el responsable actual y establecer un canal de comunicación con él. Esta idea establece pautas específicas para el contexto de colaboración, por ejemplo, en el cual la comunicación no está basada en un ruteo punto a punto sino está basado en la dinámica de ejecución del proceso.



1.4 Problemas a resolver

A pesar de los avances de los últimos tiempos, muy pocas empresas virtuales, están actualmente operativas y algunas que lo son, no necesariamente hacen uso óptimo de la tecnología existente. Lo mismo ocurre para los otros tipos de integraciones entre organizaciones.

Las soluciones informáticas y las herramientas utilizadas son, en el mejor de los casos, solo soluciones parciales, propietarias y difíciles de extender e integrar en un todo coherente. Existen además importantes problemas en la integración de la información involucrada.

Para que sea posible soportar las integraciones entre organizaciones, como la idea de empresa virtual, procesos ínter-organizacionales o comercio electrónico, de forma exitosa, debemos resolver el problema de que los distintos sistemas de información de las diferentes organizaciones **puedan cooperar intercambiando datos y sincronizando las ejecuciones de sus aplicaciones**, es decir debemos lograr su **interoperabilidad**.

Para mejorar las probabilidades de suceso es necesario lograr una verdadera interoperabilidad, es decir resolver los problemas, tanto a nivel de **integración de aplicaciones** como de **integración de información**.

La integración entre organizaciones tienen como abstracción principal el proceso, el cuál se ejecuta a través y entre organizaciones, luego es necesario resolver como soportar la operación y el desarrollo de los mismos, donde encontramos que los sistemas de workflow son el soporte tecnológico obvio para lograrlo.

Es por ello que la propuesta es considerar a los sistemas de workflow como base para esta interoperabilidad, considerándolos el sustento de los "procesos virtuales" de la "empresa virtual", donde cada empresa participa integrando sus servicios individuales, cada una usando potencialmente distintos sistemas de workflow.

Además de lograr la solución al problema de la interoperabilidad en base a la interoperabilidad entre sistemas de workflow, se quiere definir una solución completa que abarque todo el ciclo de vida del proceso de negocio virtual, sus relaciones con la empresa virtual y la comunidad de comercio, que pueda ser usada para desarrollar estos conceptos sin un costo significante de experiencia o de desarrollo.

Para poder formalizar la solución completa se plantea como objetivo realizar una modelación.

La Modelización definida incluye, además de lo necesario para satisfacer los requerimientos de interoperabilidad, tanto a nivel de aplicaciones como de información, soluciones técnicas a problemas como los de incorporar servicios de diferentes compañías como parte de un único proceso de negocio, como las empresas pueden publicar sus servicios y hacerlos disponibles a otras compañías o como los

procesos de negocio virtual pueden ser activados y monitoreada su ejecución. Además debe incluir consideraciones de seguridad, calidad de servicio y garantía de ejecución.

Analizando con más detalle la problemática general de la solución completa se destacan los siguientes problemas:

1) Características organizacionales:

Un problema clave en la transferencia de información entre organizaciones es que la estructura semántica de la información es frecuentemente perdida.
La filosofía de trabajo y los productos difieren de una organización a otra, lo mismo pasa con las reglas de negocio, estructuras organizacionales y división de responsabilidades. Por esta razón los acuerdos entre organizaciones son costosos y difíciles de establecer y consumen mucho tiempo de monitoreo, más si pensamos en expandir la interoperabilidad a varias organizaciones, gobiernos, etc.

Los problemas son menos acuciantes cuando se trata de organizaciones que tienen filosofías relativas a un dominio específico, pero en este caso los sinónimos, homónimos y otras relaciones específicas del dominio deben ser abordadas con cuidado.

2) Identificación y administración de los ítems de trabajo de un proceso:

Si pensamos en la interoperabilidad de workflow debemos pensar en procesos que son ejecutados por varios sistemas de workflow en varias organizaciones. Un ítem de trabajo va a pasar de un sistema a otro al recorrer un proceso Inter. Organizacional.
En el ámbito de herramientas de workflow la remoción de ítem de trabajo desde un workflow y la reinserción asincrónica en otro, tiene una serie de problemas a tener en cuenta. La auditoria de los ítems de trabajo son características necesarias en estos casos.

3) Consulta y monitoreo

Los sistemas de workflow deben permitir ser consultados activamente por las estadísticas de procesos o por los movimientos de trabajo. Alternativamente deben generar eventos que signifiquen el movimiento de trabajo a través de una actividad y prever que esto pueda ser capturado y tratado por otros elementos del proceso (por ejemplo las distintas organizaciones intervinientes, usuarios del proceso, etc). Del mismo modo el monitoreo centralizado y el control distribuido de los sistemas de workflow son actualmente problemáticos.

Para abordar los problemas a resolver nos basaremos en las siguientes consideraciones o puntos de partida para la investigación.

El problema de las "Características organizacionales" se soluciona demandando que los miembros que intervienen adhieran a una semántica compartida, la cuál puede ser mapeada a la semántica local de cada sistema interoperante.
Esta semántica compartida, se propone que sea definida a través del uso del lenguaje estándar XML.

En cuanto a los dos restante problemas identificados (Identificación y administración de los ítem de trabajo de un proceso, Consulta y monitoreo) se tomará como base el trabajo de estandarización de la Workflow Management Coalition, sobre interoperabilidad de sistemas de workflow, especialmente los referentes a la interface de interoperabilidad entre sistemas de workflow utilizando XML. (WfMC, Workflow Standard - Interoperability Abstract Specification y WfMC Standard interface 4 Wfmc-XML Binding).

2. Organización

Este trabajo se encuentra organizado básicamente en cuatro partes.

La primera parte "MARCO TEÓRICO", contiene la descripción del estado del arte de los Sistemas de Workflow, profundizando en la interoperabilidad de los sistemas de workflow, y del lenguaje XML (eXtensible Markup Language) que incluye una introducción al mismo, y describe especificaciones que han sido desarrolladas para potenciarlo.
Luego se incluye un capítulo que describe las estandarizaciones para la comunidad de comercio, finalizando con una Síntesis del estado del arte.

La segunda parte "MODELIZACIÓN", describe la modelización propuesta para formalizar las relaciones entre organizaciones, basadas en la interoperabilidad entre sistemas de workflow y en la utilización del lenguaje XML.

La tercera parte "CASO PRÁCTICO – APLICACIÓN DE LA MODELIZACIÓN", describe la aplicación de la modelización propuesta a un caso real.

La cuarta parte "CONCLUSIONES", donde se incluyen las conclusiones del trabajo, y finalizando se encuentran los capítulos de apéndices y bibliografía.

Parte I
MARCO TEÓRICO

3. Síntesis del estado del arte

3.1 Estándares de interoperabilidad para los sistemas de workflow

En el marco teórico describo los estándares de workflow más relevantes, como han ido evolucionando de uno a otro, hasta llegar a la especificación Wf-XML, la cuál adoptaré como estándar de interoperabilidad en la modelización propuesta. A continuación realizo una síntesis de los mismos.

Los productos de workflow tienen características en común, que los habilita potencialmente a alcanzar niveles de interoperabilidad a través del uso de estándares comunes para varias áreas funcionales, en la comunidad de workflow encuentro trabajos sobre interoperabilidad de consorcios industriales, los cuales identifican y clarifican características comunes, y administran el desarrollo de estándares, dentro de estos consorcios encuentro a la WfMC.

El primer trabajo de estandarización entre los sistemas de administración de workflow fue el realizado por la **WfMC** con su **Modelo de Referencia** definido en 1994.

El modelo de referencia de la WfMC tiene como objetivo establecer una arquitectura común para la tecnología workflow. Consiste en un marco de trabajo que identifica las características, terminología y componentes de un sistema de workflow y describe un modelo común para su construcción. Identifica cinco interfaces generales que deben ser estandarizadas.

Considero a este modelo especialmente importante ya que al ser desarrollado a partir de una estructura genérica de aplicación de workflow y al identificar las distintas interfaces, ha permitido la evolución ordenada de la interoperabilidad entre los sistemas de workflow en varios niveles, por ejemplo, algunos trabajos se han focalizado en la interface de definición del proceso (interface 1), otras en la interacción con aplicaciones clientes y aplicaciones invocadas (interface 2 y 3), en aspectos de administración y monitoreo (interface 5), y la interface de interoperabilidad entre sistemas de workflow (interface 4) que es a la que me refiero en este trabajo.

La WfMC ha identificado en el modelo de referencia las interfaces generales y ha desarrollado especificaciones apropiadas para ser implementadas en los productos de workflow. El objetivo de estas especificaciones es posibilitar la interoperabilidad entre los productos de workflow heterogéneos y mejorar la integración de los mismos con otros servicios, como el correo electrónico y la administración de documentos, mejorando así las oportunidades para un uso efectivo de la tecnología workflow, beneficiando a los vendedores y a los usuarios de estos sistemas.

La WfMC ha definido una especificación abstracta para cada una de las cinco interfaces del modelo de referencia, en particular ha definido **la interface 4 – especificación abstracta de interoperabilidad**.

Las especificaciones abstractas han permitido que los distintos productos comerciales puedan implementar algunas de las interfaces y demostrar su conformidad, esto es muy bueno en el momento de elegir o interoperar productos de workflow ya que ayudan a determinar las capacidades de los mismos.

En particular, la importancia que tiene la interface de interoperabilidad o interface 4 es que define un protocolo abstracto para las interacciones entre sistemas de workflow tanto a nivel de una misma organización como más allá de sus límites, esta definición es muy completa e incluye un conjunto muy amplio de operaciones e interacciones posibles.
Como describo en el marco teórico, la interface de interoperabilidad define los mecanismos necesarios para que distintos sistemas de workflow puedan seleccionar, instanciar y activar definiciones de procesos conocidos en otros sistemas de workflow.

Esta interface define distintos tipos de interacción, por ejemplo, a través de herramientas de software, mensajes, interacciones vía mecanismos de gateway o a través de bases de datos compartidas; los

distintos tipos de comunicación -sincrónica o asincrónica, y los distintos intercambios -atómicos o individuales y batch, los cuales se adecuan a los distintos requerimientos de integración. Un punto muy importante es la definición del conjunto de modelos de interoperabilidad.

Define un amplio conjunto de operaciones que incluye todos los relacionamientos que pueden efectuarse durante el tiempo de vida de un proceso que involucre la interoperabilidad entre sistemas de workflow. Estas operaciones permiten la creación de un proceso en otro sistema de workflow, setear sus atributos, crear una instancia del proceso, controlarla y realizar su seguimiento. Posee operaciones que permiten comunicar los cambios de estado de los procesos, entre los distintos sistemas de workflow interoperantes. Diferencia entre crear e instanciar una definición de proceso a través de dos operaciones, del mismo modo existen dos operaciones una para suspender una instancia y otra para reactivarla. Incluye además, una operación que permite obtener la lista de instancias activas con determinados criterios.

A partir de la interface abstracta de interoperabilidad, la WfMC ha realizado una serie de bindings, que son detalles de cómo las especificaciones son implementadas con un conjunto particular de herramientas, formatos y protocolos, el primero de ellos es el **binding Internet e-mail MIME**, para el correo electrónico de Internet utilizando como mecanismo de transporte la codificación MIME.

El binding Internet e-mail MIME describe como interoperan distintos sistemas de workflow a través del intercambio de mensajes de solicitudes y respuestas del correo electrónico de Internet, los sistemas de workflow están identificados a través de una casilla de correo a la cuál escuchan.

Esta forma de interoperar, aunque ha sido implementada en algunos productos, tiene como inconveniente que solo permite interacciones asincrónicas dadas las características de la mensajería, y además se debe implementar un mecanismo de interface entre el mensaje MIME y el motor de workflow, se deben realizar los seteos de casilla de correo y la forma de comunicarse entre el ingreso de un mail a la casilla y su pasaje al sistema de workflow.

En 1998 la **Object Management Group** publica su estándar **Workflow Management Facility**, conocido como la especificación JointFlow, desarrollado por el trabajo de un grupo numeroso de vendedores de productos de workflow y de usuarios

Este estándar está basado en el modelo de referencia de la WfMC, la interface 2 - aplicación cliente workflow, la interface 4 - especificación abstracta de interoperabilidad y los formatos de datos de auditoria, interface 5.

La especificación Workflow Management Facility satisface la necesidad de un marco de trabajo para la aplicación de sistemas de workflow distribuidos representados a través de un modelo de objetos, que permitiera a los distintos productos de workflow y aplicaciones relacionadas trabajar juntas.

La importancia de esta especificación es que describe un modelo de objetos unificado que cubre varias interfaces estándares de la WfMC (todas menos la interface de identificación de procesos).

Este modelo permite la representación de la interoperabilidad entre sistemas de workflow, el monitoreo de sus ejecuciones, y la asociación de los componentes de workflow con los recursos involucrados en el proceso.

Este estándar aporta un modelo de objetos Wf-Model y define interfaces que describen las relaciones y dependencias con los solicitantes y proveedores de servicios de workflow (Wf-Requester, Wf-Process, Wf-ProcessMgr, Wf-Activity, WfExecutionObject) e incluyen las estructuras de datos, operaciones, excepciones y relaciones con otras interfaces.

El estándar también define interfaces para el manejo de la lista de trabajos, las asignaciones y recursos (WfAssignment y WfResource) y procesos de auditoria (WfEventAudit).

Es un modelo claro, que tiene un pequeño conjunto de 8 clases de objetos que sirven para la interacción entre el solicitante y proveedor de un servicio de workflow y para el manejo de listas de trabajo y procedimientos de auditoria. Por lo reducido del modelo y sus claras interfaces provee un modelo sencillo y completo apto para basar especificaciones concretas sobre él.

Es tomado como modelo lógico de recursos por las siguientes evoluciones de los estándares tanto por el proyecto SWAP, como por la especificación Wf-XML, como describiré a continuación.

Aunque hasta ahora en las evoluciones de WMF solo se implemente las interacciones solicitante proveedor, quedan especificados los demás relacionamientos de forma estándar, debido a la naturaleza de los modelos de objetos. Esto permite el crecimiento ordenado, la evolución de un estándar puede ser enriquecido agregando las características incluidas en el modelo de objetos, relacionadas a la administración, monitoreo, y al manejo de recursos y asignaciones, sin tener que rescribir lo implementado.

La propuesta **SWAP (Simple Workflow Access Model)**, fue presentada a Internet Engineering Task Force en diciembre de 1998. Tiene como objetivo definir un protocolo de acceso para workflow basado en Internet, que permita para instanciar, controlar y monitorear instancias de proceso de workflow.

SWAP fue ideado como un binding del modelo de objetos de OMG Workflow Management Facility (jointFlow) y estándares WfMC, es un protocolo de interacción basado en HTTP, que pretende ser un modelo más "liviano" apto para Internet

La idea básica de la propuesta SWAP (traduciendo o interpretando interacciones entre componentes de aplicaciones workflow como mensajes codificados en XML), provee una excelente base para adaptar los estándares de la WfMC en el área de la integración de aplicaciones de empresas basadas en mensajes.

El modelo de objetos de SWAP define cuatros recursos: Observers, ProcessInstance, ProcessDefinitions y Activities. Estos se corresponden a las interfaces del estándar de la OMG, WfRequester, WfProcess, WfProcessMgr y WfActivity, que representan la interacción solicitante proveedor que residen en localizaciones distintas de Internet.

El proyecto SWAP se ha implementado exitosamente en varios proyectos como describí en el estado del arte.

Este proyecto representa un paso muy importante en la simplificación de la integración entre los sistemas de workflow, debido al uso de mensajes de solicitud/respuesta HTTP sobre protocolo TCP/IP de Internet, que no establece requerimientos sobre la plataforma o tecnología en la que residen las aplicaciones invocadas, el uso del modelo de objetos definido por la OMG WMF y la introducción del uso del lenguaje XML como lenguaje para los mensajes.

La WfMC se basó en la experiencia recogida en el proyecto SWAP para crear la especificación **Wf-XML**, que es un binding de la **especificación abstracta de interoperabilidad utilizando el lenguaje XML**, actualmente se encuentra disponible la versión final 1.1 de noviembre de 2001.

Es una evolución de las siguientes especificaciones:

- WfMC Interoperability Abstract (IF4) specification.
- OMG Workflow Management Facility (jointflow)
- WfMC IF4 Internet E-mail MIME binding specification.
- Propuesta de Simple Workflow Access Protocol (SWAP)

Se basa en los conceptos centrales y las funcionalidades descriptas en la especificación abstracta de interoperabilidad (interface 4), pero no posee una correlación directa entre las operaciones y parámetros

especificados en ella, dado que se ha desarrollado con el propósito de describir un lenguaje simple, fácil y que sea rápido demostrar la viabilidad de su implementación.

El estándar Wf-XML usa el modelo de objetos básico definido en la especificación WMF de la OMG. Soporta un subconjunto de las entidades definidas en este modelo de objetos y combina operaciones que están separadas en las interfaces OMG Workflow Management Facility en una única operación, consecuentemente mejora la performance no requiriendo operaciones tan finamente granulares. Del mismo modo que el proyecto SWAP implementa la interacción solicitante proveedor y deja sin representar la asignación de recursos y la administración y monitoreo.

La correspondencia entre el modelo de recursos es la siguiente:

- El recurso ProcessDefinition de la Wf-XML se corresponde con la interface WfProcessMgr de la OMG WMF.
- El recurso ProcesssInstance se corresponde a la interface WfProcess de la OMG WMF.
- El recurso Observer de la Wf-XML se corresponde con la interfaz WfRequester del OMG WMF.

La especificación utiliza mensajes para comunicarse lo que facilita una rápida implementación usando tecnologías actuales. Describe la sintaxis de los mensajes en una forma abierta, basada en el uso del lenguaje estándar Extensible Markup Languaje (XML), para permitir un formato de comunicación estructurado, robusto y personalizable, del mismo modo que el proyecto SWAP.

Encuentro muchas de ventajas en la aplicación de XML en los mensajes Wf-XML ya que:

1. Es posible separar el protocolo de mensajes del medio de comunicación, luego solucionamos las restricciones propietarias o limitaciones del ambiente en los que un producto de workflow puede trabajar.

2. XML es fácilmente transformado de una sintaxis a otra haciéndolo un protocolo flexible para interacciones en ambientes con modelos de datos heterogéneos.

3. XML es abierto y se está activamente proveyendo de mecanismos de transporte, interpretación y transformación.

4. La naturaleza extensiva de XML ofrece alguna protección contra la obsolescencia, por ejemplo, los protocolos de mensaje pueden ser definidos con esquemas que son referenciados en cada mensaje haciendo a XML un protocolo auto-descripto.

5. XML permite desarrollar una arquitectura tecnológica en capas sobre un formato de mensajería simple y extensible lo que nos ayuda a provee robustez, simplicidad, reusabilidad e interoperabilidad.

La especificación Wf-MXL fue descripta detalladamente en el marco teórico, posee varias características a resaltar, una de ellas es que considera que el concepto de interoperabilidad entre sistemas de workflow puede ser extendido para incluir interacciones entre otros tipos de sistemas y servicios. Considera a los sistemas de workflow como servicios genéricos en el modelo lógico de recursos, es decir, como un recurso el cual pueda tener variada implementación, con el requerimiento de ser identificable y poder interactuar uniformemente con otros recursos recibiendo solicitudes de servicio y retornando las respuestas adecuadas. Lo que permite aplicar Wf-XML a la implementación y diseño de otros sistemas integrados o distribuidos.

La especificación Wf-XML abarca un conjunto importante de las características necesarias para modelar los procesos de negocios virtuales ya que:

- Permite la existencia de los distintos tipos de procesos de negocios virtuales, dado que puede soportar modelos de workflow encadenados, anidados y paralelamente sincronizados, que cubren todas las posibilidades de conexión entre procesos.

- Posee diferentes modelos de comunicación, adaptables a las distintas necesidades de las organizaciones, permite interacciones sincrónicas y asincrónicas entre los sistemas involucrados. Además de soportar operaciones individuales y batch.

- Es sencillo e independiente de la implementación, (por ejemplo, del lenguaje de programación, mecanismo de transporte de datos, plataforma de OS/hardware). De todos modos, como HTTP se prevé que es el mecanismo de transporte de datos más utilizado para intercambiar mensajes Wf-XML, la especificación provee una descripción de cómo es el intercambio de los mensajes WF-XML mediante HTTP.

- Permite definir un protocolo liviano y fácil de implementar mediante la utilización del lenguaje XML, un mensaje Wf-XML es un documento XML. Provee una adaptación más sencilla del modelo abstracto de interoperabilidad.

- La especificación Wf-XML deja libre la especificación de los datos de contexto y resultado de los procesos (context data y result data). Lo que permite una mejor adaptación a distintas interacciones entre organizaciones y permite utilizar estándares XML o gramáticas XML comunes para la representación de los mismos logrando mayor flexibilidad.

- Existe un conjunto de productos comerciales que incluyen la implementación de las funcionalidades necesarias para su utilización. (WebFlow de Sap, Staffware entre otros) además dado su fácil implementación puede ser resuelto a través de las Workflow API's (WAPI) de los productos y a través de programación, como en el caso de proyecto AFRICA.

En resumen además de todas las características descriptas, considero que la especificación Wf-XML permite simplificar los esfuerzos de integración y maximiza la reutilización del diseño y codificación. Esto impacta en que se incrementará la rapidez en que se puedan implementar u obtener soluciones más oportunas, menos costosas, que beneficia directamente a la adopción del estándar en los productos de workflow comerciales. Esto es lo que se necesita para lograr generalizar la integración entre organizaciones a través de la interoperabilidad entre sistemas de workflow, facilitando la creación de empresas virtuales con sus procesos de negocio virtuales, comercio electrónico, entre otras integraciones.

3.2 XML – eXtensible Markup Language

Además de las ventajas mencionadas en el uso del lenguaje XML, para la estructura de los mensajes utilizados por Wf-XML, se deben agregar las características ya mencionadas en el marco teórico, como su extensibilidad, estructuración, ser auto-validable, simple, abierto, auto-descriptivo, etc., las cuales hacen que XML sea el lenguaje apropiado para utilizar en la resolución del problema "Características organizacionales", referido a la transferencia de información entre organizaciones. Este problema como ya he mencionado, lo propongo solucionar a través de uso de una semántica definida en XML, acordada entre las organizaciones involucradas en la integración.

Al utilizar XML cuento con la ventaja de que es un estándar, que presenta especificaciones complementarias que facilitan su aplicación, tiene una amplia aceptación en la industria, que es incorporado en estándares para la comunidad de comercio, y en las nuevas tecnologías, especialmente las utilizadas en Internet, como por ejemplo los Web Services.

Como utilizo XML para representar la gramática compartida entre las organizaciones interoperantes, a través de la definición de gramáticas XML, puedo optar por representar las gramáticas a través de esquemas XML Schemas o los DTD. Aunque tienen similar propósito, los XML Schemas ofrecen más precisión y flexibilidad para definir gramáticas XML.

Como ya he mencionado un XML-Schema es un conjunto de reglas predefinidas que describen una clase de documento XML. Define los elementos y atributos permitidos en un documento XML, especifica la relación entre ellos.

Las diferencias principales por las que he decido utilizar XML Schema frente a los DTD son:

- Modelo de contenido abierto.
  Los XML Schema pueden definir un modelo de contenido abierto, lo que permite que se puedan presentar elementos o atributos adicionales en un elemento XML sin tener que declararlos a cada uno de ellos en el XML Schema.
  En contraste, los DTD definen un modelo de contenido cerrado, lo que significa que un documento no puede presentar contenido adicional si no está explícitamente definido en el DTD.

- Tipos de datos complejos para elementos o atributos.
  Los XML Schemas permiten especificar tipos de datos para elementos o atributos, los cuales indican el formato, y esto permite que pueda ser validado por un parser XML.
  Los DTD no soportan tipos de datos complejos.

- Extensibles.
  Los XML Schemas son extensibles, dado que se pueden crear esquemas personalizados a partir de esquemas estándares. Por ejemplo, si existe un elemento <dirección> de otro esquema que es adecuado para mi esquema propio de elemento <dirección>, este esquema puede ser usado directamente en mi esquema, a través de una referencia al esquema original llamada namespace, en vez de copiar su definición.
  Los documentos XML solo pueden referenciar a un DTD, lo cual lo anterior no puede ser realizado.

- Aplicación a elementos individuales.
  Los esquemas XML Schemas son aplicados a elementos específicos. Generalmente, se quiere aplicar un XML Schema al elemento raíz de un documento XML para definir la gramática de todo el documento. De todas maneras, se puede aplicar un XML Schema a elementos aislados para definir la gramática de porciones de un documento XML.
  En un documento XML solo un DTD puede ser aplicado.

- Sintaxis XML regular.
  Los XML Schemas usan sintaxis XML en vez de sintaxis no XML como la utilizada en los DTD. Una de las consecuencias interesantes de esto es que se puede acceder a través de programación a los XML Schemas de un documento y así acceder a sus reglas gramaticales.
  Para escribir un DTD se necesita aprender otra sintaxis y un conjunto específico de marcas.

En especial, quiero resaltar la característica de XML, de los XML Namespaces, que proveen un método simple para calificar elementos y nombres de atributos usados en documentos XML mediante la asociación de ellos con identificadores de colecciones de nombres a través de una URI. Estos me permitirán nombrar e identificar las distintas semánticas compartidas de las distintas integraciones entre organizaciones e incluir gramáticas definidas en los estándares de la comunidad de comercio.

3.3 Estándares para la comunidad de comercio.

Han surgido un conjunto importante de emprendimientos de grupos de compañías e industrias para definir como trabajar en conjunto a través del comercio electrónico, cadenas de suministros, etc., conformando lo que se llama comunidad de comercio.
Estos emprendimientos tienen como objetivo mejorar y ordenar las interacciones entre los socios de negocio eliminando redundancias y mejorando el intercambio de información, y en particular, lo más destacable es la adopción generalizada del lenguaje XML.

Principalmente tienen como meta primaria establecer estándares para intercambiar información. Pero no solo buscan definir tipos de documentos XML sino que muchos de ellos implementan marcos de trabajo que describe detalles técnicos para la conectividad, vocabularios y diccionarios de datos que definen la información de negocio que intercambian y procesos estándares para definir las interacciones entre socios de negocio.

Las organizaciones de pequeña escala pueden integrarse solo definiendo estándares de vocabulario y diccionario de datos pero si nos referimos a redes de comercio medianas o grandes se necesita definir claramente los procesos e implementar todo el marco de trabajo.
En todos los casos se identifica claramente la necesidad de establecer estándares que permitan tener pautas de relacionamiento lo más claras posibles, porque la ambigüedad hace que se incremente la complejidad y causa incompatibilidades entre los socios.

Los estándares han evolucionado desde los comienzos con el estándar EDI hasta ahora que han surgido iniciativas desvinculadas de productos concretos como ebXML o RosettaNet o otras propietarias como xCBL de ComerceOne o cXML de Ariba, centradas en productos de comercio electrónico. Otros trabajos importantes que han contribuido a la evolución de los estándares son el trabajo de las organizaciones que apoyan el uso de estándares de XML como W3C y OASIS y los estándares de los Web Services (WSDL - Web Service Description Language, UDDI - Universal Description, Discovery and Integration).

Los **marcos de trabajo** definidos incluyen una serie de componentes y definen la infraestructura tecnológica que permite que los socios de negocio interactúen de forma segura y confiable, entiendan y procesen la información de negocio.

Cada marco de trabajo incluye:

- Un **protocolo** de comunicación, que especifica el comunicación a nivel de redes
- **Estructura de mensajes** que se intercambian.
- **Conversación** que detalla la secuencia de mensajes y los tipos de intercambio, etc.
- **Seguridad**
- **Contrato entre socios**, incluyen las condiciones para establecer el vínculo electrónico entre los socios.

Ejemplos de marcos de trabajo:

- **EbXML:** Define un marco de trabajo que contiene un protocolo, estructura de mensaje ebXML, seguridad, conversación y contrato, este último es un documento XML llamado CPA collaboration protocol agreement.

- **RosettaNet:** Define un marco de trabajo RNIF, que consiste en una infraestructura, que plantea el relacionamiento con intercambio de mensajes con una arquitectura de capas:
    - Capa de Negocio: Capa de mensajes de negocio.
    - PIP-Capa: Intercambio de mensajes según lógica especificada.
    - Capa RNIF: Empaquetado, ruteo y encriptado.

  Define la estructura de los mensajes representados por documentos XML, y utiliza protocolo http o SMTP para transmisión de los mensajes sobre TCP/IP.

Otro componente a estandarizar son los **vocabularios** que proveen sintaxis, definen la estructura de la información de negocio y permite tener descripciones entendibles para humanos y esquemas legibles para aplicaciones en especial utilizando XML. Se definen a través de XML Schema y DTD.

Los **diccionarios** son los componentes que proveen semántica a los elementos y atributos. Es crítico el acuerdo sobre los diccionarios entre los socios de negocio tanto como el de los vocabularios para entender información de intercambio.

Ejemplos de vocabularios y diccionarios:

- **RosettaNet:**
  Diccionario de negocio (Business Dictionary) incluye términos relacionados con transacciones (factura, cuenta bancaria).
  Diccionario técnico (Technical Dictionary): incluye términos para la descripción de los objetos de las transacciones, (elementos para la descripción de productos y servicios.)

- **EbXML:**
  Vocabulario y Diccionario de datos: Componentes básicos de negocios reutilizables (core components): incluyen distintos niveles de complejidad, los básicos representan tipos de datos, incluyen un contexto, y son usados para estandarizar documentos de más alto nivel.

Los **estándares para procesos** especifican las actividades, decisiones, roles para cada socio de negocio para poder cumplir un conjunto de funciones de negocio.

Ejemplos de estandarizaciones para procesos:

- **EbXML:**
  Process Specification: Contiene información acerca de cada una de las transacciones que pertenecen a los procesos.
  Existe un elemento central en que las empresas pueden acudir para obtener la información de interés sobre potenciales socios de negocio, llamado Registro ebXML. En este registro se publican los servicios que ofrece una empresa en los términos de ebXML, sus perfiles de actividad, llamados Collaboration Protocol Profile – CPP, para que sea accesible a los posibles interesados.

- **RosettaNet:**
  Especificación de procesos: Organiza los procesos de negocios en 8 grandes grupos llamados Cluster, que se subdividen en unidades llamadas Segmentos los cuales contienen una descripción detallada de los procesos identificados en cada Cluster. Los segmentos a su vez se subdividen en PIPs Partner Interface Process que describen las acciones de cada socio de negocios en más detalle, representados como documentos XML.

3.3.1 Estandarización para contratos electrónicos en XML

Como ya he mencionado, un contrato o acuerdo electrónico de socios de negocios define como los socios de negocios interactuarán, se definen claramente cada expectativa y requerimiento de todas las áreas que puedan impedir la interoperabilidad. Están modelados por documentos XML que contienen la información esencial que los socios de negocio deben acordar para que sus aplicaciones y procesos de negocio puedan interoperar.

Existen propuestas para estandarizar estos acuerdos, como el usado por ebXML en los Collaboration Protocol Agreements o en el lenguaje de marcas para especificar contratos, Trading partner agreements Markup Language creado por IBM, y que fue presentado a OASIS (XML.ORG) para su estandarización.

Al aplicar el estándar Wf-XML necesito que los socios de negocio acuerden todo lo necesario para efectivizar la interoperabilidad, incluyendo la información relacionada con la semántica compartida establecida y utilicen un contrato de negocio o acuerdo de interoperabilidad para documentar el relacionamiento y contar con las siguientes ventajas:

- Estos documentos XML son importantes ya que capturan la información esencial que los socios de negocios deben acordar, para comunicar sus aplicaciones y procesos de negocio. Cuando se involucran muchos socios es importante distinguir la información de interacción de otra información perteneciente a una particularidad de diseño o implementación de un socio.

- Pautan un único punto de control para los acuerdos consolidados y orientados a negocios, esta facilidad ayudará a tener modelos de interacciones de negocios, para asistir en el aceleramiento de la conexión y comienzo de nuevas interacciones con nuevos socios.

- El implementar los acuerdos de negocios fácilmente y rápidamente posibilitará, a través de proveer conectividad dinámica y registrar en lugares públicamente accesibles la información de los acuerdos, ayudar para identificar y localizar nuevos potenciales socios de negocios.

- El acuerdo de interoperabilidad y las herramientas asociadas para su autorización y registro pueden ayudar para acelerar el proceso de definir, cómo, las interacciones de negocio de un nuevo socio, pueden llevarse a cabo. Por ejemplo, las herramientas de validación de XML pueden detectar inmediatamente errores. En muchos casos la definición del estilo de interacción se puede basar a partir del modelo genérico o personalizar un acuerdo basado en uno de los estándares de comercio electrónico como RosettaNet, CPA u OBI (Open Buying on the Internet)

4. Sistemas de Workflow

4.1 Desde los procesos a los sistemas de workflow

La estructuración de las organizaciones a través de sus procesos ha sido común desde los años treinta. A pesar de estos primeros esfuerzos, la separación de tareas en base a las funciones, y el resultado de estructuras funcionales o divisionales ha dominado las prácticas corporativas hasta la década del 80, cuando cambiaron las condiciones de mercado y se incrementó la competitividad, lo que condujo a las compañías a investigar sobre la eficiencia de la estructura de sus procesos.

Siguiendo los movimientos de los 80 de la administración de Calidad Total, emergieron numerosas prácticas de administración de procesos de los 90, entre ellas se encuentran la mejora de procesos, innovación de procesos de negocios, y el rediseño de procesos de negocio. Cada una de estas aproximaciones perciben el rol facilitador de la tecnología de información para reestructurar sus organizaciones. En consecuencia las compañías que se comprometieron en estas actividades buscaron sistemas de información adecuados para soportar la administración de sus procesos. La tecnología de administración de Workflow está diseñada para soportar exactamente estos problemas.

A pesar de las afirmaciones que los desarrollos de aplicaciones de workflow están fuertemente entrelazadas con los movimientos de reingeniería de procesos de principios de los 90, el origen de la tecnología de workflow puede ser rastreada a los finales de 1970, donde uno de los primeros conceptos de sistemas de información para soportar procesos organizacionales fue un sistema de automatización de oficinas que usaba redes Petri para representar los procesos de negocio. (SCOOP system).

La investigación en la automatización de oficinas, que floreció entre 1975 y 1985, dejó el trabajo de base para el desarrollo de aplicaciones de workflow industriales a través del análisis de la tecnología para soportar procesos administrativos. Mientras que los intereses en la investigación de la automatización de oficinas se redujeron a mediados de los 80, dos desarrollos surgieron el Groupware y los sistemas administradores de Workflow. Mientras que la investigación de groupware se focalizó en el soporte de actividades de colaboración no estructuradas, la investigación en los sistemas de workflow se focalizó en la coordinación de actividades a lo largo de un modelo de proceso común, sin la automatización de las actividades en sí mismas.

La Administración del Flujo de trabajo o workflow es una tecnología en creciente evolución que esta siendo explotada en una variedad de industrias.
Su principal característica es la automatización de procesos considerando actividades humanas y basadas en máquinas, particularmente aquellas que involucran interacción con aplicaciones y herramientas de tecnología de información.
Algunas de las áreas de mayor aplicación son: bancaria, legal, administrativa, industrial, manufacturera.

Para describir el marco teórico de los sistemas de workflow, nos basaremos principalmente en el trabajo de la Workflow Management Coalition, y en los trabajos de estandarización asociados a la misma, como el de la Object Management Group OMG Workflow Management Facility Standard y del trabajo del proyecto SWAP. Centraremos la atención en la Interface abstracta de interoperabilidad de sistemas de workflow y en las implementaciones propuestas.

La **Workflow Management Coalition, WfMC** (http://www.wfmc.org/) fue fundada en 1993, es una organización internacional sin fines de lucro, con más de 300 miembros, que incluyen de vendedores de Workflow, usuarios, analistas y grupos de investigación de universidades.

Su misión es promover y desarrollar el uso de los sistemas de workflow, estableciendo estándares para la terminología, interoperabilidad, y conectividad entre productos de workflow.

Dentro de sus objetivos se incluyen, el incrementar el valor de la inversión de los usuarios de la tecnología workflow, reducir el riesgo de uso de estos productos, y expandir el mercado de workflow incrementando su conocimiento.

Entre sus objetivos específicos se encuentra el de lograr la interoperabilidad entre productos de workflow heterogéneos y mejorar la integración de las aplicaciones de workflow con otros servicios información tales como correo electrónico y administración de documentos.

Los estándares de la WfMC permiten que las definiciones de procesos y herramientas de modelado sean usadas para especificar procesos de negocios implementados en diferentes motores de workflow, que un sistema de workflow pueda invocar y ser invocado por otros sistemas de workflow y que múltiples sistemas de workflow puedan operar conjuntamente (en una organización o en múltiples).

Existen tres niveles de estándares de workflow de la WfMC, el modelo de referencia, las especificaciones abstractas y los bindings.
El **Modelo de Referencia** (Reference Model) es el modelo de cómo los estándares se relacionan entre si y define un conjunto de interfaces. Las **especificaciones abstractas** (Abstract Specifications) identifican cada una de las funcionalidades requeridas y que datos involucran. Los **bindings** son los detalles de cómo las especificaciones son implementadas con un conjunto particular de herramientas, formatos y protocolos.
Por ejemplo, como se verá más adelante, la Wf-XML es un binding de la Especificación Abstracta de Interoperabilidad de la Interface 4 de Modelo de referencia de WfMC.

4.2 ¿Qué es un workflow?

La WfMC (Workflow Management Coalition) define workflow como:

> **La automatización total o parcial de procedimientos, en donde cada documento, información o tarea es pasada de un participante a otro para su procesamiento, de acuerdo a un conjunto de reglas procedurales.**

El Workflow es a menudo asociado a la Reingeniería de los Procesos del Negocio (**B**usiness **P**rocess **R**e-engineering) análisis, modelado e implementación operacional del núcleo de los procesos del negocio de una organización.

La tecnología de workflow separa la lógica de los procesos del negocio de su soporte operacional IT, permitiendo la incorporación de cambios en las reglas procedurales que los definen.

Un sistema de administración de workflow (WfMS) provee la automatización de un proceso a través de la administración de una secuencia de actividades e invocación de recursos humanos y/o informáticos, asociados a los distintos pasos del proceso. Este sistema define, administra y ejecuta flujos de trabajo a través de un software, cuyo orden de ejecución es manejado por la representación computacional de la lógica del workflow.

Desde una perspectiva conceptual, el propósito de los sistemas de administración de workflow es coordinar todas las entidades involucradas en la ejecución de un proceso (de negocio o de software). La coordinación puede ser definida como la administración de dependencias entre actividades. Los sistemas de administración de workflow resuelven dos tipos de problemas de coordinación: las dependencias de los datos entre las actividades (una actividad necesita de los resultados de una o más actividades), que son administradas a través del control y el flujo de datos, y la coordinación de recursos compartidos (por ejemplo el recurso participante de workflow puede realizar una tarea en un mismo momento), las cuales están administradas a través de mecanismos de agendas y de asignación de participantes.

A través de la automatización de estas funciones de coordinación, los sistemas de administración de workflow soportan varias metas de eficiencia de las organizaciones:

- Eficiencia en los procesos: Optimización de los criterios de los procesos como tiempo de procesamiento (a se minimizado) o cumplimiento de plazos (a ser maximizado).
  - Logrado por el soporte de workflow a la coordinación de actividades a través del control del flujo y de plazos, etc.

- Eficiencia en los recursos: Uso eficiente de los recursos (humanos tanto como sistemas de aplicaciones) disponibles para la ejecución del proceso:
  - Logrado por el soporte del workflow de la asignación de recurso.

- Eficiencia en el mercado: La propia posición de la organización con sus relaciones con los socios de mercado. Esto incluye una predicción confiable de los tiempos de demora, comunicación transparente con los proveedores, clientes y procesos de adquisición y distribución optimizados.
  - Logra a través de interfaces de procesos bien definidas para los web services (que definen comportamientos externos) , comportamiento interno previsible a través de procesos estandarizados.

- Eficiencia en la delegación: Adecuado uso de las competencias de las unidades organizacionales superiores (visión mayor del alcance de los procesos) y de la unidades subordinadas (conocimiento detallado sobre una actividad determinada).
  - Coordinación de la asignación de los participantes, conceptos de roles.

- Eficiencia en la motivación: Motivar a los participantes para actuar de forma congruente con las metas de negocio de la organización.
  - Guía para realizar actividades a lo largo del modelo de workflow, monitoreo del progreso y información de actividades previas.

Los participantes de un workflow pueden ser humanos, recursos maquinas (procesos de producción automatizados) o componentes de software (existen workflow que se utilizan para propósitos de integración de aplicaciones).

La estructura de los procesos automatizados pueden ser predefinidas o flexibles (de workflow de producción hasta aplicaciones de workflow ah hoc).

La estructura de los procesos automatizados pueden ser restringidos a una sola organización, o extenderse más allá de los límites de una organización como en el caso de una aplicación de comercio electrónico (business to business).

La granularidad de los objetos de datos manejados en el workflow pueden ser grandes (documentos, o objetos pasados a lo largo de un proceso) o finos (como atributos).
La granularidad de las aplicaciones que puede invocar un workflow pueden ser grandes a nivel de procesos como llamada a un web services en el caso de B2B, o llamado a nivel de aplicación, como por ejemplo un programa, o finos como una llamada a método o componente.

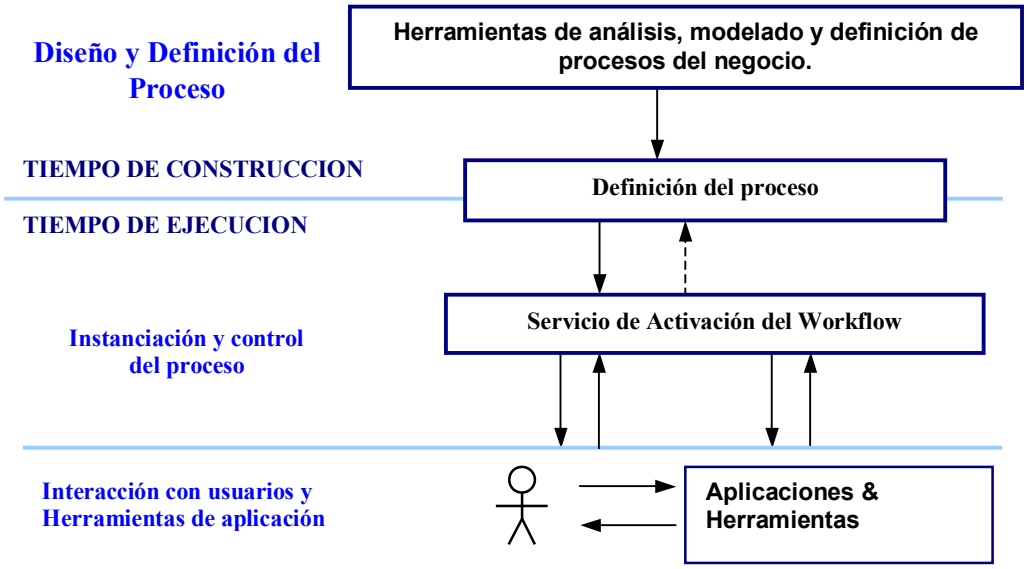


Figura 1 – Características de los sistemas de Workflow

A grandes rasgos los sistemas de Administración de Workflow (WfMS) pueden caracterizarse como aquellos que proveen soporte en tres áreas básicas:

4.2.1 Funciones de Diseño del Proceso en tiempo de construcción

Son aquellas funciones que traducen un proceso del negocio a una **"Definición del proceso"** especificada en un lenguaje formal procesable por una computadora. Para ello se utilizan una o más técnicas de análisis, y modelado de procesos. El resultado obtenido se llama "Modelo de proceso", "Plantilla de Proceso", "Metadata de Proceso" o "Definición de proceso" y puede ser expresado en forma textual, gráfica o en algún lenguaje formal.
La "Definición de proceso" incluye un número discreto de pasos de actividades asociados a un conjunto de operaciones computacionales y/o humanas y reglas que rigen el progreso del proceso. Algunos sistemas de workflow permiten la modificación dinámica de este modelo en tiempo de ejecución.
Esta es un área potencial de estandarización para lograr el intercambio de datos entre las distintas herramientas de diseño de procesos.

4.2.2 Funciones de Control en Tiempo de Ejecución

Son funciones relacionadas a la administración y el mantenimiento del proceso de workflow en un ambiente operacional así como al secuenciado de las actividades que forman parte de cada proceso.
Durante la ejecución la "definición del proceso" es interpretada por un software que es el responsable de crear y controlar las instancias operacionales del proceso y programar los pasos de las actividades que forman parte del mismo, así como de invocar los recursos necesarios.
Estas funciones actúan como enlace entre la "definición del proceso" y el proceso en el mundo real reflejado en las interacciones de usuarios y de aplicaciones IT.
El componente núcleo es el **Software de control de Administración del Workflow o "Motor de Workflow"** responsable de la creación y borrado del proceso, así como, de controlar la programación de las actividades dentro de un proceso operacional y de la interacción con herramientas de aplicación o recursos humanos. Este software es a menudo distribuido a través de un número de plataformas de computación para llevar a cabo procesos que operan sobre una amplia base geográfica.
Las funciones de control tienen dos aspectos: almacenamiento persistente y la navegación del proceso. El almacenamiento persistente de datos permite al sistema la recuperación de fallas sin pérdida de información y permite también contar con los medios para mantener y auditar el proceso en ejecución.

La lógica navegacional controla la ejecución del proceso cambiando el estado de actividades de acuerdo a la evaluación de condiciones. Por esta razón existen dos componentes básicos que son: **el servidor de almacenamiento** y el servidor de navegación o **motor de Workflow** (Workflow Control Data y Workflow Engine).

4.2.3 Funciones de Interacción en Tiempo de Ejecución

Las actividades dentro de un proceso de workflow están relacionadas a operaciones humanas a menudo realizadas con alguna herramienta de tecnología de información particular como es el llenado de un formulario o con operaciones de procesamiento que requieren la ejecución de una aplicación particular como la actualización de una base de datos de pedidos con un nuevo registro.
La interacción con el software de control del proceso es necesaria para transferir el control entre actividades, asegurar el estado operacional de los procesos, invocar herramientas de aplicación y pasar los datos apropiados.
Existen varios beneficios en tener un marco de referencia estándar que soporte este tipo de interacciones, incluyendo el uso de una interface consistente para múltiples sistemas workflow y la habilidad de desarrollar herramientas de aplicación comunes para trabajar con distintos productos de workflow.

4.2.4 Distribución e Interface de Sistemas

La habilidad para distribuir tareas e información entre participantes es una característica distinguible de la infraestructura de ejecución de los Sistemas de workflow. La función de distribución puede operar en varios niveles, desde grupos de trabajo a inter-empresas, dependiendo del alcance del workflow; puede usar una gran variedad de mecanismos de comunicación: mail electrónico, pasaje de mensajes, tecnología de distribución de objetos.

El flujo de trabajo puede involucrar la transferencia de tareas entre productos de workflow de distintos proveedores para permitir que distintas partes del proceso sean ejecutadas en distintas plataformas o subredes, usando productos particulares adecuados a esa etapa del proceso.
En este escenario el flujo pasa entre dos o más productos de workflow. Por ejemplo, si tengo un flujo con 5 actividades, las actividades 1,2 y 5 pueden ser ejecutadas por un sistema de workflow y las actividades 3 y 4 por un sistema distinto con el pasaje del control entre ellas en puntos apropiados dentro del workflow.
Estándares que soporten este pasaje de control permiten el desarrollo de aplicaciones de workflow compuestas usando productos de workflow distintos que operan juntos como una única entidad lógica.

5. Modelo de referencia de Workflow de WfMC

El Modelo de Referencia de Workflow ha sido desarrollado a partir de una estructura genérica de una aplicación de workflow, identificando las interfaces dentro de la misma. Este modelo permite a los productos interoperar en varios niveles. Los sistemas de workflow están formados por un número genérico de componentes.
Para lograr la interoperabilidad entre productos de workflow se ha definido un conjunto estándar de interfaces y formatos de intercambio de datos entre sus componentes.

Las interfaces definidas por la WfMC (Workflow Management Coalition) son:

- Especificación para los datos que forman parte del modelo de proceso y su intercambio.
- Interfaces que soporten interoperabilidad entre distintos sistemas de workflow.
- Interfaces que soporten interacción con una variedad de tipos de aplicaciones IT.
- Interfaces que soporten interacción con funciones de escritorio de las interfaces de usuarios.
- Interfaces que provean monitoreo de sistemas y funciones de métricas que faciliten la administración de un ambiente de aplicaciones de workflow compuesto (con más de un producto de workflow).

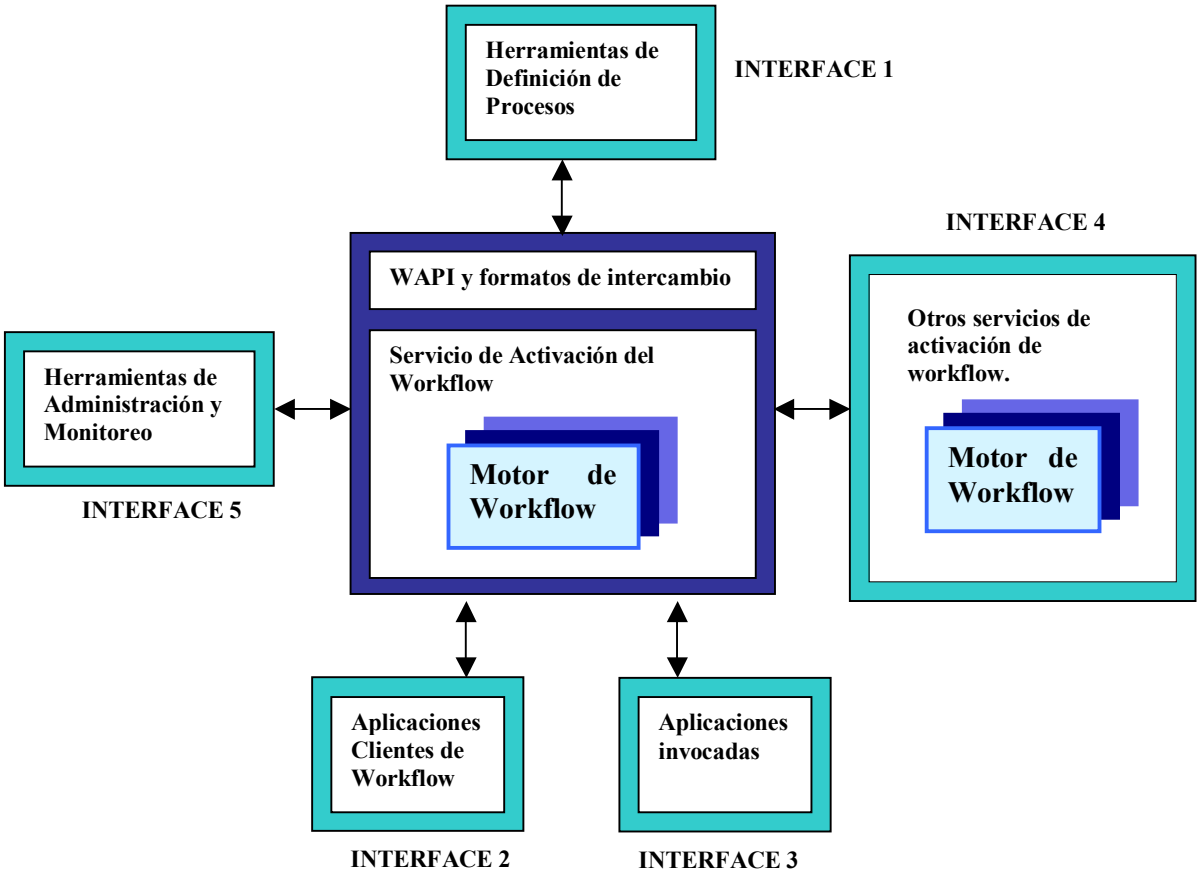


Fig.2 . Modelo de Referencia de Workflow. Componentes e Interfaces

La interfaz alrededor del Servicio de Activación de Workflow se designa WAPI (Workflow API), que junto a los formatos de intercambio, pueden ser considerados como un conjunto de construcciones a través de los cuales pueden ser accedidos los servicios del sistema de workflow y son los que regulan la interacción entre el software de control de workflow y las demás componentes del sistema.
Muchas de las funciones dentro de las cinco áreas de interfaces son comunes a dos o más servicios de interfaces por lo cual es más apropiado considerar a WAPI como una interfaz de servicio unificado que es usada para soportar las funciones de administración del workflow a través de cinco áreas funcionales en lugar de cinco interfaces individuales.

5.1 Servicio de Activación del Workflow

Es un servicio que brinda el ambiente en tiempo de ejecución en el cual ocurren la activación e instanciación del proceso, utilizando uno o más motores de administración del workflow, responsables de interpretar y activar toda o parte de la definición del proceso e interactuar con los recursos externos necesarios para procesar cada una de las actividades.

5.1.1 def.: Servicio de Activación del Workflow

Es un servicio de software que puede consistir en uno o más motores de workflow, para crear, manejar y ejecutar instancias de workflow, las aplicaciones se comunican con este servicio a través de las WAPI.

En el modelo adoptado existe una separación lógica entre el proceso y la lógica de control de las actividades que constituye el Servicio de Activación de Workflow y las herramientas de aplicación y las tareas de usuarios finales que constituye el procesamiento asociado con cada actividad.

La interacción con los recursos externos accesibles con un Servicio de Activación de Workflow particular ocurre a través de una o más interfaces:

- **La interface con aplicaciones cliente**, a través de la cual el motor de workflow interactúa con el Manejador de Listas de Trabajo, responsable de organizar el trabajo para el recurso de un usuario. Es responsabilidad del manejador seleccionar y avanzar un ítem de trabajo individual de una Lista de Ítem de Trabajo. La activación de las herramientas de aplicación pueden estar bajo el control del Manejador de listas de trabajo o del usuario final.

- **La interface de aplicación invocadas**, que permite al motor de workflow llamar directamente una herramienta específica para llevar a cabo una actividad particular. Esto sería una aplicación basada en el servidor sin interface de usuario; donde una actividad particular usa una herramienta que requiere interacción del usuario final sería normalmente invocada a través de la interface de la lista de trabajos para proveer una mayor flexibilidad a la asignación de las tareas de trabajo. Al usar una interface estándar para la invocación de herramientas, las futuras herramientas de aplicación pueden ser **'workflow enabled'** en forma estándar.

En esta sección el Servicio de Activación del Workflow ha sido descripto como una sola entidad lógica, aunque físicamente puede ser centralizada o distribuida funcionalmente.

En un Servicio de Activación de Workflow distribuido, varios motores de workflow, cada uno controla una parte de la activación del proceso e interactúa con un subconjunto de usuarios, y herramientas de aplicación relacionadas a las actividades dentro del proceso de las cuales es responsable. Tal servicio de activación tiene igual nombre y alcance administrativo, para que las definiciones del proceso (o subconjuntos) y nombre de usuario / aplicación puedan ser manejadas con bases consistentes. Los sistemas de workflow distribuidos hacen uso de protocolos específicos y formatos de intercambio entre motores de workflow para sincronizar sus operaciones e intercambiar procesos e información de control de actividades. Los datos relevantes del workflow también pueden ser transferidos entre motores de workflow. Dentro de un único Servicio de Activación de workflow tal operación es específica del proveedor del producto.

Donde existen productos heterogéneos, es necesario un intercambio estándar entre motores de workflow . Usando la interface 4 de interoperabilidad entre sistemas de workflow, el servicio de activación puede transferir actividades o subprocesos a otro Servicio de Activación de workflow para su ejecución. También será necesario la administración y el monitoreo de funciones comunes.

5.2 El motor de workflow

Un motor de workflow es responsable de, parte o todo, el ambiente de control de ejecución dentro de un servicio de activación.

def.: Es un servicio de software o "motor", que provee el ambiente en tiempo de ejecución para una instancia de un flujo de trabajo.

Típicamente este software provee facilidades para soportar:

- Interpretación del proceso.
- Control de instancias del proceso - creación , activación, suspensión, finalización , etc.
- Navegación entre actividades de proceso, que pueden involucrar operaciones secuenciales o paralelas, "deadline scheduling", interpretación de los datos relevantes del workflow, etc.
- Registración y borrado de participantes nuevos
- Identificación de ítems de trabajo para la atención de usuarios y una interface que soporte interacción de usuarios.
- Mantenimiento de los datos de control del workflow y de los datos relevantes del workflow, pasando los datos relevantes del workflow desde y hacia usuarios y/o aplicaciones.
- Una interfaz para invocar aplicaciones externas y unir todos los datos relevantes del workflow.
- Acciones de supervisión para control, administración y auditoria.

Un motor de workflow puede controlar la ejecución de un conjunto de procesos, o subprocesos, instancias con un alcance definido, determinado por el rango de tipos de objetos, y sus atributos que pueden ser interpretados dentro de la definición del proceso.

5.3 INTERFACE 1 - Intercambio de Definición de Workflow

Es la interface de las herramientas de modelado y definición, y el software de administración en tiempo de ejecución (Runtime Workflow management) es definida como interface de importación / exportación de definición de procesos. Esta interface es un intercambio de formatos y llamadas a APIs, la cual puede soportar el intercambio de información de definición de procesos sobre una variedad de medios de intercambio físico y electrónico. La interface puede soportar el intercambio de una definición completa de procesos o un subconjunto, por ejemplo, un conjunto de cambios de una definición de procesos o de los atributos de una actividad particular entre el proceso de definición.

5.4 INTERFACE 2 - Interface de aplicación cliente Workflow

La idea es contener la variedad detrás de un conjunto de APIs estándar (el WAPI) el cual debe ser usado de manera consistente para acceder de una aplicación Workflow al motor de workflow y listas de trabajos, sin importar la naturaleza de la implementación actual del producto.
Las Apis y sus parámetros serán mapeados en varios mecanismos alternativos de comunicación para encajar en la variedad de modelos de implementación de Workflow.

En el caso de comunicaciones basadas en e-mail es posible, para un manejador de listas de trabajos acceder directamente a la casilla de entrada (inbox) para ítems de trabajo entrantes usando cualquier interface local de acceso al mail, en vez de un llamado específico a WAPI. En el caso de una aplicación de manejador de listas de trabajos será responsable de filtrar los ítems de e-mail que no sean de Workflow y trabajar con ellos de una manera apropiada. Comandos similares o respuestas directas al motor de workflow por la aplicación Workflow, debe ser suministrados directamente al manejador de mail de salida. En este escenario un nivel simple de interoperabilidad es logrado a través del uso del intercambio de mensajes estándar, en vez del uso completo de WAPI.

5.5 INTERFACE 3 - Interface de aplicaciones invocadas por Workflow (Invoked Applications Interface)

El alcance de la interface es aplicable a agentes de aplicaciones y aplicaciones las cuales hallan sido designadas como "Workflow enabled" (es decir capaces de interactuar directamente con el motor de Workflow) las cuales a través de APIs e intercambio estandarizados podrán interactuar con los sistemas de activación de workflow.

5.6 INTERFACE 4 - WAPI Funciones de interoperabilidad

Esta interface será referida en profundidad en el punto "Interface 4 – Especificación estándar de interoperabilidad".
La naturaleza general de la información y control de flujos entre sistemas de Workflow heterogéneos es mostrada en la siguiente figura:

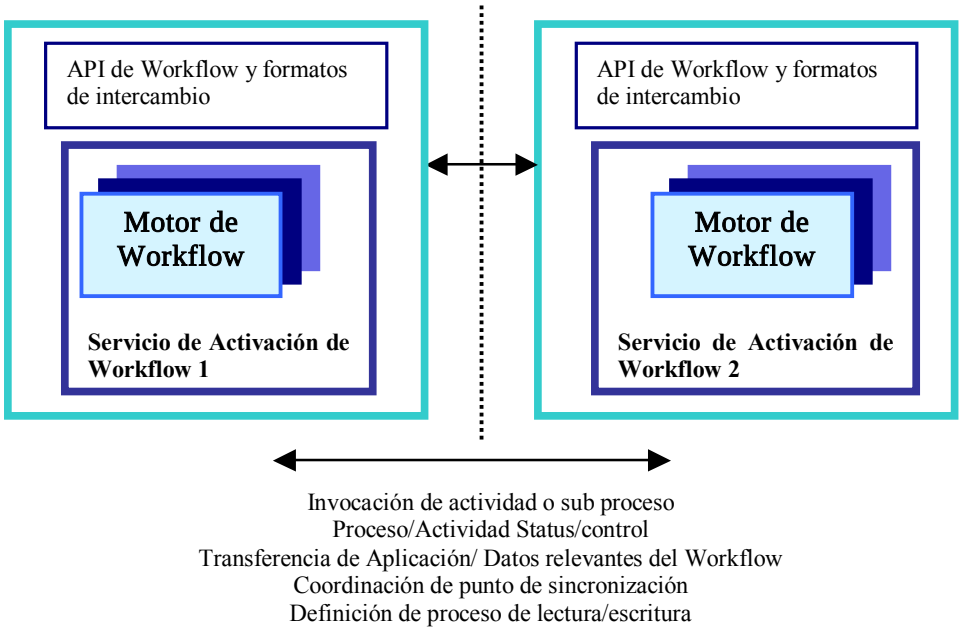


Fig. 3 Interface de interoperabilidad de Workflow.

Hay dos aspectos principales para la interoperabilidad:

- La extensión de las interpretaciones comunes de definiciones de procesos (o a un subconjunto) que se necesita y la cual puede ser alcanzada.
- Soporte en tiempo de ejecución para el intercambio de varios tipos de información de control y transferencia de datos relevantes del Workflow y/o datos de las aplicaciones (Workflow relevant y / o application data) entre los distintos servicios de representación.

Uso de la definición de un proceso a través de múltiples dominios.

Cuando ambos servicios de activación pueden interpretar una definición de proceso común, posibilita compartir una sola visión de los objetos de la definición del proceso y sus atributos. Esto incluye actividades, aplicaciones, nombres de organizacionales y roles, condiciones de ruteo, etc. Esto permite potencialmente que motores de Workflow individuales le transfieran la ejecución de actividades o procesos a motores de Workflow heterogéneos dentro del contexto del nombrado común y el modelo de objetos. Esta aproximación es particularmente aplicable para la interoperabilidad cuando varios sistemas cooperan a un nivel puntual.
Cuando esta visión común no es posible existe la alternativa de "exportar" los detalles de subconjuntos de la definición de procesos como parte de un intercambio en tiempo de ejecución. Las APIs de intercambio de definición de procesos provee un requerimiento de objeto y atributo de dato desde un servicio de Workflow particular, que posibilita al motor de Workflow obtener los datos relevantes de la definición del proceso para la ejecución de una actividad individual o un subproceso asignado a él en un ambiente de representación cooperativo.
Cuando los intercambios de definición de procesos no pueden ser posible por las aproximaciones anteriores, la interoperabilidad esta dada por la aproximación de un gateway, en el cuál (típicamente un subconjunto de) nombres de objetos y atributos son mapeados entre los dos ambientes vía una aplicación de interrelación gateway. En este simple caso, dos servicios de representación separados usan sus propios formatos de definición de procesos con un mapeo entre ambos a través de un gateway.

Interacciones de control en tiempo de ejecución.

En tiempo de ejecución los llamados a WAPI son usados para transferir el control entre servicios de Workflow para representar subprocesos o actividades individuales sobre servicios específicos. Cuando ambos servicios soportan un nivel común de llamados a WAPIs y una visión común de los objetos de la definición de procesos (incluso convenciones de nombrado y los datos relevantes para el Workflow y la aplicación) esto será hecha directamente entre los motores de Workflow – aunque esto requerirá la conformidad del soporte de un protocolo común para las primitivas WAPI.
Cuando este no es el caso, las llamadas a WAPI pueden ser usadas para construir una función gateway que provea interrelación entre los dos servicios de Workflow mapeando los diferentes objetos y visiones de datos entre los servicios de Workflow.
Esto se ilustra en el diagrama:

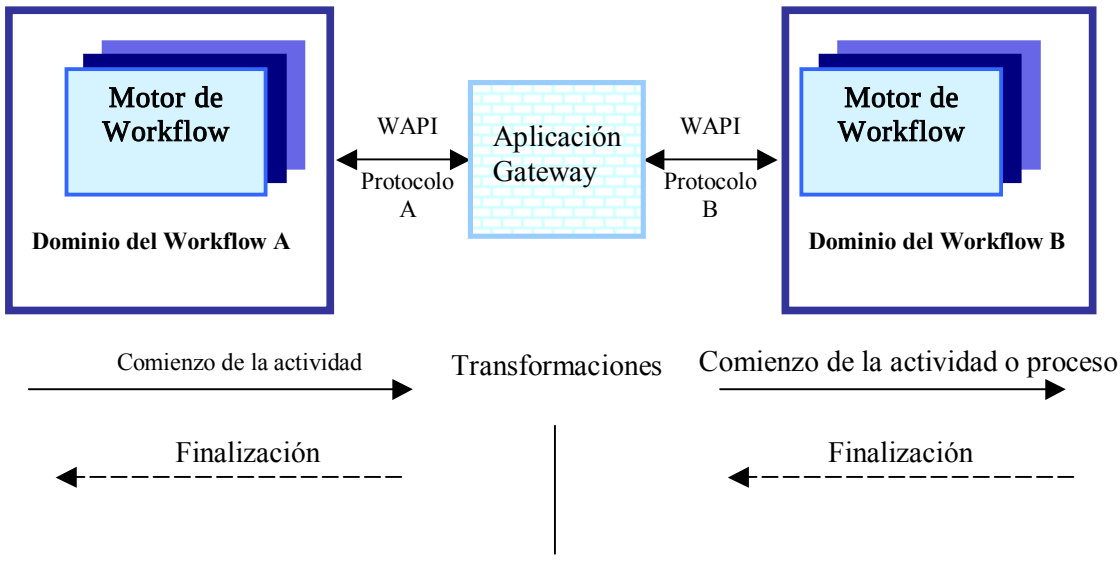


Fig. 4 Operación de Gateway usando WAPI

Probablemente un gran número de comandos WAPI sean explotados para soportar la interoperabilidad a través de llamadas directas entre dos servicios de Workflow o vía una función gateway. Varios de los comandos WAPI discutidos anteriormente son potencialmente aplicables a la interoperabilidad:

- Establecimiento de sesión
- Operaciones sobre las definiciones de procesos y sus objetos
- Funciones de control de procesos y de estado
- Funciones de manejo de actividades
- Operaciones de manejo de datos

5.7 INTERFACE 5 - Interface de administración y monitoreo

La interface permite que la aplicación de administración interactúe con diferentes dominios de Workflow, pueden ser posibles escenarios alternativos, por ejemplo que la aplicación de administración sea parte integral de uno de los servicios, y sea capaz de manejar varias funciones a través de dominios de Workflow adicionales (heterogéneos).

6. Interface 4 WfMC – Especificación Abstracta de Interoperabilidad

[WfMC, Workflow Standard - Interoperability Abstract Specification, Doc. Nro. WFMC-TC-1012, 20 Octubre 1996, Versión 1.0]

6.1 Alcance de la especificación de Interoperabilidad

El estándar de Interoperabilidad de la WfMC define los mecanismos requeridos por los proveedores de productos de Workflow para implementar motores de workflow, a fin de que éstos puedan realizar requerimientos a otros motores a efectos de:

- seleccionar
- instanciar
- activar

definiciones de procesos conocidos por esos otros motores.

El motor de workflow que realiza los requerimientos (opcionalmente) puede además tener la posibilidad de recibir información de estado y el resultado de la activación de la definición del proceso. Hasta donde esto sea posible debe ser realizado de manera "transparente al usuario". Cómo efecto colateral para facilitar esta comunicación entre motores de Workflow, hay una marcada necesidad de producir datos de auditoría.

6.2 Definición y modelos de interoperabilidad entre workflow

La **Interoperabilidad entre Workflow** se describe como:

> **"La habilidad de dos o más motores de Workflow de comunicarse e interoperar, a fin de coordinar y ejecutar instancias de procesos de workflow entre ellos".**

Se puede concluir que la interoperabilidad entre workflows puede ocurrir entre dos o más motores de workflow: (figura 5)

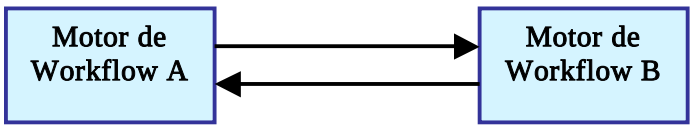


Fig. 5 Interacción directa entre motores de Workflow

Dos o más motores de workflow que operan dentro de un mismo servicio de activación de workflow (Figura 6)

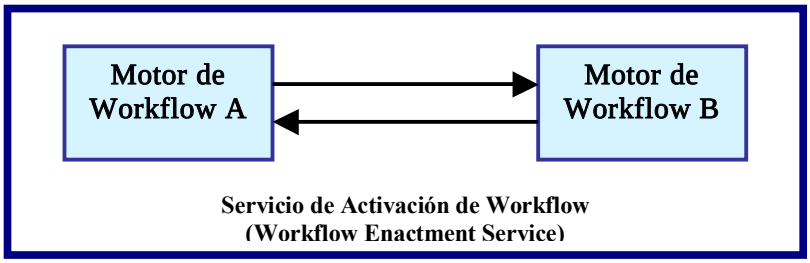


Fig. 6. Interacción entre motores de workflow dentro de un servicio de activación.

Dos o más servicios de activación de workflow (por ejemplo uno o dos motores de workflow operando desde dentro de dos o más servicios de activación de workflow) dentro de los límites de un sistema de administración de Workflow (Workflow Management System), (figura 7)

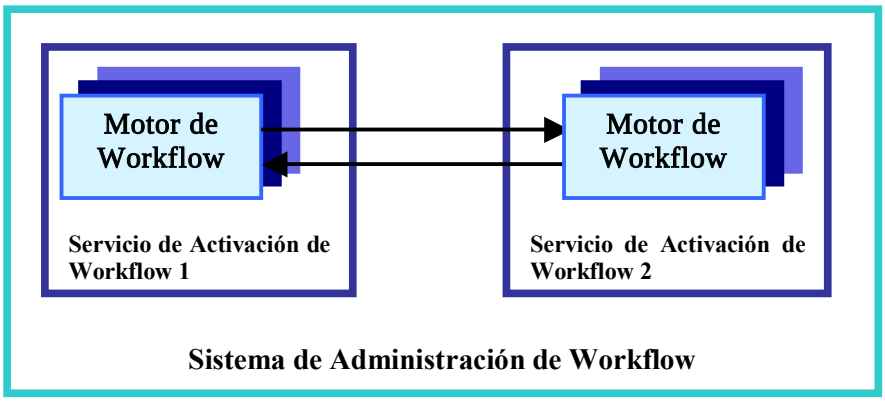


Fig.7. Interoperabilidad entre motores de workflow en diferentes servicios de activación.

El Modelo de Referencia de la WfMC [WfMC003] extiende la definición de un **servicio de activación de workflow (Workflow Enactment Service)** de la siguiente manera:

> "... provee un ambiente en tiempo de ejecución en el cuál ocurre la instanciación y la activación de procesos, utilizando uno o más motores de workflow, responsables de interpretar o activar una parte o en forma total la definición del proceso, e interactuar con recursos externos para procesar diversas actividades."

De lo anterior podemos inferir que dos motores de procesos que tengan distintos ambientes de tiempo de ejecución, pueden tomarse como si fueran dos servicios de activación de workflow diferentes.

Los recursos externos pueden ser tomados como:

- Agentes humanos (a través de las aplicaciones cliente workflow)
- Herramientas de software invocadas para realizar tareas particulares
- Otros motores de workflow (que pueden constituir individualmente o colectivamente servicios de activación de workflow).

6.2.1 Efectivizar la Interoperabilidad

La interoperabilidad entre herramientas de software significa la habilidad de compartir datos y/o funcionalidades entre dos o más herramientas. Una herramienta de software puede ser una pieza que realiza una o un conjunto de funciones específica, como un editor de texto, mail, un servidor corporativo de base de datos o un sistema de administración de workflow. La interoperabilidad normalmente se logra utilizando una de las siguientes estrategias:

1.- Interacción directa entre herramientas (ver figura 8)

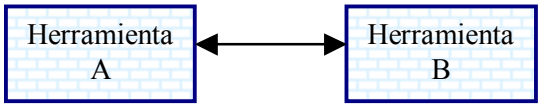


Fig. 8 Interacción directa entre herramientas de software

2. Pasaje de mensajes (Figura 9)

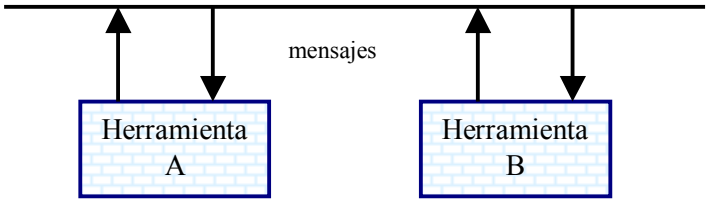


Fig. 9 Herramientas de Software interactuando a través del pasaje de mensajes

3. Bridging (usando alguna forma de encapsulamiento, traducción o mecanismo de gateway como en la figura 10)

Fig.10 Herramientas de Software interactuando vía un gateway que realiza la trascripción de protocolo. (protocol transliteration)

4. Usa un almacenamiento de datos compartido (repositorio común, figure 11).

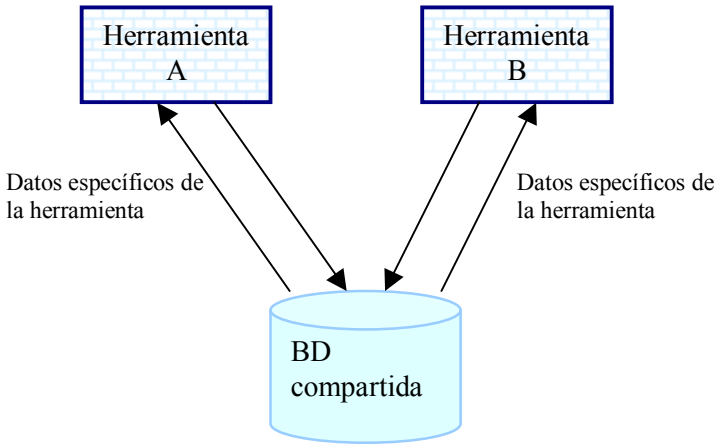


Fig.11 Herramientas de Software integradas vía el uso de un repositorio común.

Esta última aproximación no es específicamente consideraba por el modelo de referencia, pero dado que existen productos de workflow los cuáles mueven el trabajo desde una actividad a otra a través de una base de datos interna, usando una base de datos común compartida para mover paquetes de trabajo entre productos de workflow, es admisible, como una manera alternativa de efectuar la interoperabilidad entre estos productos. En un nivel abstracto esta aproximación puede ser vista como otro mecanismo de almacenamiento (store-forward mechanism)

6.2.2 Niveles de interoperabilidad

Existen ocho niveles desde la no interoperabilidad hasta interoperabilidad extrema.

6.2.3 Modelos de Interoperabilidad

Los productos de Workflow son diversos en naturaleza, desde aquellos los cuales usan ruteo ad-hoc para sus tareas o datos o aquellos que son para procesos de producción.
El trabajo de la Coalición, se ha focalizado en desarrollar una variedad de escenarios de interoperabilidad los cuales, pueden operar desde un simple pasaje de tarea, a una aplicación de Workflow interoperable con intercambio completo de definición de Workflow, y los datos relevantes del Workflow (Workflow relevant data). En ésta área, se espera que escenarios de interoperabilidad simple sean soportados en una etapa inicial hasta que situaciones más complejas, requieran mayor trabajo sobre definición de interoperabilidades.
Se han identificado cuatro modelos posibles de interoperabilidad, los cuales cubren varios niveles de capacidades.

A. Procesos encadenados (Chained processes)

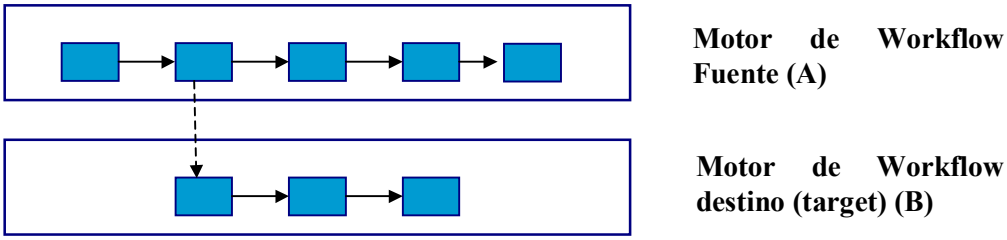


Fig. 12 El modelo encadenado de interoperabilidad

Este modelo de interoperabilidad asume que la instancia de proceso que está activa en el motor de workflow A dispara la creación y la activación de una instancia subproceso en el motor de workflow B. Cuando comienza la activación de la instancia del subproceso en el motor B, el motor A puede terminar o continuar con la activación de su propia instancia de proceso. Este no se interesa mas por la instancia de subproceso creada recientemente.

B. Subprocesos anidados (Nested sub-process)

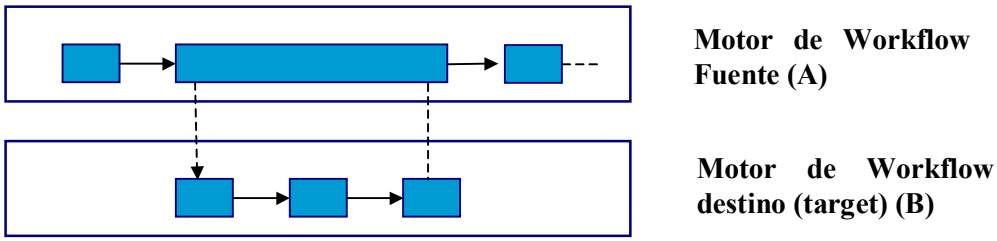


Fig.13. Modelo de interoperabilidad de subprocesos anidados

El modelo de interoperabilidad de subprocesos anidados asume que una instancia de proceso activada en un motor de workflow, causa la creación y activación de una instancia de subproceso en un segundo motor, y luego espera por su terminación antes de continuar con su propia activación.

C. Paralelamente sincronizado (Parallel synchronized)

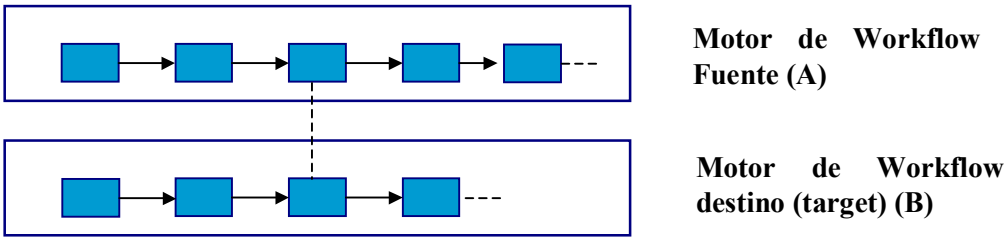


Fig. 14. El modelo de interoperabilidad paralelamente sincronizado.

El modelo de interoperabilidad paralelamente sincronizado asume que dos motores de workflow simultáneamente activan instancias de procesos, y que en algún punto de la definición de cada instancia de proceso, se ha especificado un rendezvous. Una vez alcanzado el punto del rendezvous, el primer motor de workflow (no se especifica cual es el primero) espera hasta que el otro alcance este punto. Una vez que la activación de ambas instancias de procesos han alcanzado el respectivo punto de rendezvous, hay un intercambio (no especificado) entre los motores de workflow, antes de que ambos continúen con la activación de sus respectivas instancias de proceso.
Este modelo de interoperabilidad está fuera del alcance actual que el estándar pretende alcanzar.

6.3 Suposiciones sobre diseño

La suposición que recalca la especificación es que dos o más motores de workflow existen (pueden ser dos o más instancias del mismo producto de workflow o instancias de distintos productos) los cuales pueden comunicarse entre si a efectos de seleccionar, instanciar y activar una definición de procesos conocida y (opcionalmente) retornar los resultados de la ejecución de las definiciones de procesos anidados al motor de workflow invocante. No hay suposiciones acerca de cómo es efectuada la comunicación, solo que se realiza tampoco sobre características de la arquitectura o operativas de los productos de workflow. Solo que existen comunicación sincrónica y asincrónica.
El principio de transparencia a través de la interface asume que la definición de proceso esta especificada usando protocolo común como WPDL[1] (Workflow Process Definition Language) descripto en la interface 1 [WfMC0020].

El término Motor de Workflow A se refiere al motor de workflow que activa la instancia del proceso (padre) que causa que otro Motor de Workflow B inicie la activación de una instancia de subproceso (hijo).

6.3.1 Comienzo de un subproceso encadenado

Este escenario describe la activación de un subproceso de tipo encadenado descripto anteriormente. Para ejemplificar se asume la existencia de dos motores de workflow A y B.

[1] WPDL provee un lenguaje formal para la definición y el intercambio de definiciones de proceso usando los objetos y atributos definidos con el meta-modelo incluido en la interface 1.

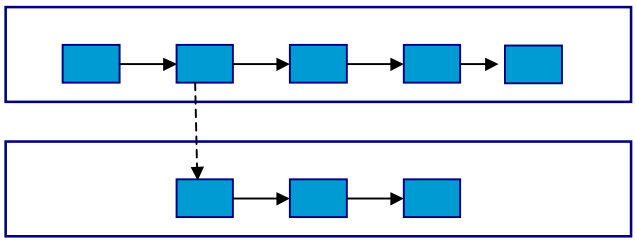


Fig. 15 Subproceso encadenado

El motor de workflow A debe:

1. Seleccionar una definición de proceso administrada o accesible al motor de workflow B.
2. Pasar al motor B los datos relevantes del workflow (relevant data) (el cuál debe incluir la localización de la application data) para instanciar la definición seleccionada.
3. Requerir al motor B activar la definición de proceso seleccionada.

Transferir los datos de las aplicaciones (application data) entre motores de workflow interoperables se estima fuera del alcance de la especificación y del WAPI. La notificación de la localización de los datos de las aplicaciones (la application data) a ser procesada por las herramientas invocadas durante la activación de las actividades del workflow, es tratada como pasaje de procesos de los datos relevantes del workflow (relevant data).

Con respecto a las posibilidades de selección de la definición del proceso por el motor de workflow B, se tiene que:

1. La definición pertenece al motor A y es pasada al motor B cuando es requerida.
2. La definición es administrada por el motor de workflow B y requerida por el motor de workflow A cuando es requerida (esto implica que el motor de workflow A conoce la existencia y localización de la definición).
3. Todas las definiciones están almacenadas en un espacio común (repositorio) y pueden ser obtenidos por cualquier motor de workflow involucrado.

Para la especificación del estándar se reduce a dos opciones:

1. La definición del subproceso a ser activado es pasado desde un motor de workflow a otro.
2. La localización de la definición del subproceso a ser activado es conocida y accesible a los motores de workflow.

El estándar asume que en el caso 1 debe ocurrir cuando el motor de workflow B requiera instrucciones al A y será efectuado usando otros mecanismos de intercambio posiblemente, uno que involucre Process Definition Interchange como esta definido en [WfMC0020, interface 1], aquí solo se considera la opción 2.

Una característica clave para la especificación de bindings concretos es cuando la interoperabilidad entre los motores de workflow es efectuada usando algún modelo de comunicación sincrónica, como el requerido por la conexión directa a través de TCP/IP, o comunicación asincrónica la cual puede ser usada alguna forma de mecanismos de store-forward como la mensajería electrónica.

[La suposición de trabajo es, que la interoperabilidad realizada a través de un negociador de objetos (Object Request Brokers ORB) puede ser sincrónica y asincrónica. Pero un ORB soporta un tercer modo de trabajo, llamado sincronismo diferido donde la aplicación invocante suspende su mensaje, continúa

con su propio trabajo y reclama la respuesta al ORB tiempo después; este modo está fuera del alcance de la especificación.]

Luego, hay dos casos distintos que debemos considerar:

- Interoperabilidad sincrónica
- Interoperabilidad asincrónica

La diferencia central entre los dos modos es el requerimiento de ambos motores de estar "en línea" al mismo tiempo para efectuar el diálogo de la interoperabilidad.

Lista de operaciones usadas para dialogar entre dos motores de workflow y activar un subproceso encadenado:

Motor de Workflow	***Operaciones***
A	*Create Process Instance*
B	*Response to Create Process Instance*
A	*Set Process Instance Attributes*
B	*Response to Set Process Instance Attributes*
A	*Get Process Instance Attributes*
B	*Response to Get Instance Attributes*
A	*Start Process Enactment*
B	*Response to Start Process Enactment*
A	*Relinquish Process Instance*
B	*Response to Relinquish Process Instance*

Nótese que el diálogo entre los dos motores del workflow está basado en la noción de pares de mensajes de requerimiento/respuesta, donde la respuesta retorna un estado que indica el éxito, fracaso o otro resultado del funcionamiento pedido. Las respuestas son distintas de las notificaciones hechas por el motor de Workflow B al tener cambios de estado o cambios en los valores de atributos de instancias de proceso que ocurren durante la vida de los subprocesos activados. No se envían notificaciones a las instrucciones recibidas, dado que se envían mensajes de respuesta con la información necesaria.

Ejemplo :

Se asume que un motor de Workflow A requiere comenzar la activación de un subproceso en el motor de Workflow B.
El motor A se conectaría al motor de Workflow B y le pasaría una instrucción para crear una nueva instancia de proceso, basada en una definición del proceso conocida usando la operación *Create Process Instance*.
El motor de Workflow B responde notificando al motor A del identificador de proceso (PID) del proceso creado.

El motor de Workflow A puede setear valores de los datos relevantes en la definición usando la operación Set *Process Instance Attributes*. El motor de Workflow B responde notificando al motor A que la operación a sido exitosa o ha fracasado.

Cuando es necesario el motor B puede solicitar al motor A que asigne valores a los datos relevante del workflow usando la operación *Set Workflow Instance Attributes*. El motor de workflow A le responde proporcionando los valores requeridos.

El motor A le requerirá o le dará instrucciones la motor B para empezar la activación de la instancia del proceso usando la operación *Start Process Enactment*. El motor B notificará al A cuando esto ha ocurrido.

Si los tiempos de respuesta no son los adecuados para soportar o sostener la comunicación atómica, el motor A puede requerir mediante mensajes batch la transmisión al motor B. El motor de workflow B devolverá mensajes batch de respuesta, uno para cada mensaje de la requerido en forma batch por el motor A.
Asumiendo transmisiones batched, los requerimiento y respuestas podrían ser como sigue:

Motor de Workflow A	→	Create Process Instance
Motor de Workflow B	→	Respuesta a Create Process Instance
Motor de Workflow A	→	Setea los atributos de la instancia del proceso Set Process Instance Attributes Start Process Enactment Relinquish Process Instance
Motor de Workflow B	→	Respuesta a Set Process Instance Attributes Respuesta a Start Process Enactment Respuesta a Relinquish Process Instance

Si ocurre una falla de un mensaje de requerimiento en el medio del batch de requerimientos, el motor de workflow B devolverá un batch de respuesta en el que:

- Aquellas operaciones que fueron exitosas previas a la falla
- Las operaciones que fallaron devuelven un estado de falla
- Las operaciones requeridas que seguidas de operaciones que fallaron retornan un estado de operación no realizado.

Pueden construirse procesos encadenados involucrando la creación de muchos casos del proceso, ej. el proceso central comienza un subproceso encadenado en otro motor el cuál a su vez crea una instancia de subproceso encadenado en un tercer motor de workflow.

6.3.2 Comienzo de un subproceso anidado.

En este escenario, la instancia de proceso padre espera por la finalización del proceso hijo, posiblemente recuperando datos relevantes del proceso o de los datos de aplicación antes de realizar el siguiente paso del proceso. En este caso, es necesario proveer de un mecanismo de notificación de finalización de la activación del subproceso.

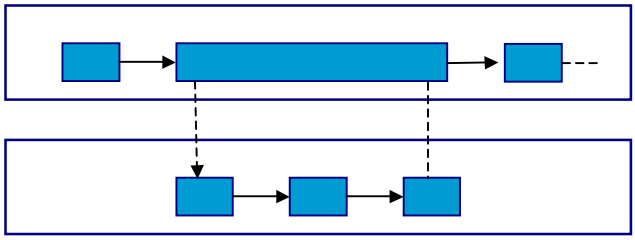


Fig. 16 Subproceso anidado

Operaciones requeridas para activar un subproceso anidado:

Motor de Workflow	*Operaciones*
A	*Create Process Instance*
B	*Response to Create Process Instance*
A	*Set Process Instance Attributes*
B	*Response to Set Process Instance Attributes*
A	*Get Process Instance Attributes*
B	*Response to Get Instance Attributes*
A	*Start Process Enactment*
B	*Response to Start Process Enactment*
A	*Notify Process Instance Attribute Changed*
B	*Response to Notify Process Instance Attribute Changed*
A	*Get Process Instance Attributes*
B	*Response to Get Process Instance Attributes*
A	*Relinquish Process Instance*
B	*Response to Relinquish Process Instance*

Ejemplo:

Asumimos que el motor de workflow A requiere comenzar la activación de un subproceso anidado en el motor de workflow B, y necesita los resultados de la activación para poder continuar su propia activación.

El motor de A se conectará al motor B y le pasará una instrucción para crear una nueva instancia de proceso basada en una definición del proceso conocida, usando la operación *Create Process Instance*. El motor B responde notificando al motor A del PID de la instancia de proceso creada.

El motor A puede setear valores a los datos relevantes del workflow de la definición usando la operación *Set Process Instance Attributes*. El motor B responde notificando A que la operación ha sido exitosa o si ha fallado.

Cuando sea necesario el motor B puede preguntar la motor A por la asignación de valores a los ítems de datos relevantes del workflow usando la operación *Get Workflow Instance Attribute.* El motor A responde con los valores requeridos.

El motor A requiere o le da instrucciones al motor B para iniciar la activación de la instancia del proceso con la operación *Start Process Enactment.* Una vez que la activación halla comenzado el motor B se lo notifica al A.

Cuando la activación del subproceso se termina, el motor de workflow B es obligado a comunicar el producto de lo requerido y se exige comunicar el producto del subproceso al motor de workflow A (o al menos lo notifica que está ahora disponible). Esto puede ser logrado a través de la operación *Notify Process Instance Completed.*
Luego el motor de workflow A le pide al B los valores de los ítems de datos relevantes del workflow, usando la operación *Get Workflow Instance Attributes.* El motor de workflow B responde proporcionándole de los valores pedidos.
Una vez ha recuperado todo los valores que requiere, el motor A le comunica al motor B que puede liberar de forma segura toda la memoria relativa a la estructura relacionando a la activación de la instancia del subproceso.
Esto que puede lograrse a través de la operación *Relinquish Process Instance.*

Asumiendo transmisiones batched, los requerimiento y respuestas podrían ser como sigue:

Motor de Workflow A	→	Create Process Instance
Motor de Workflow B	→	Respuesta a Create Process Instance
Motor de Workflow A	→	Setea los atributos de la instancia del proceso Set Process Instance Attributes Start Process Enactment
Motor de Workflow B	→	Respuesta a Set Process Instance Attributes Respuesta a Start Process Enactment
Motor de Workflow B	→	Notify Process Instance Attribute Changed
Motor de Workflow A	→	Respuesta a Notify Process Instance Attribute Changed
Motor de Workflow A	→	Get Process Instance Attributes
Motor de Workflow B	→	Respuesta a Get Process Instance Attributes
Motor de Workflow A	→	Relinquish Process Instance
Motor de Workflow B	→	Respuesta a Relinquish Process Instance

6.4 Convenciones de nombrado.

Las operaciones están especificadas usando el lenguaje de definición de interface de OMG (IDL) definido en [OMG93]. Se usan tipos abstractos de datos que pueden ser mapeados a los definidos en [WfMC1013, Interface 2 Convenciones de nombrado] para producir bindings al lenguaje C. Para bindings de otros lenguajes, éstos deben proveer sus propias definiciones de tipos. Los estados y valores resultados usados son derivados de [WfMC015, Interface 5 Audit Data Specification].

6.5 Especificación de operaciones para efectivizar la interoperabilidad

Se definen las especificaciones IDL que son representaciones abstractas de las operaciones requeridas, y las especificaciones de mensajes que son representaciones abstractas de la información. Es posible para motores de workflows interoperando sincrónicamente (a través de un negociador de objetos ORB) que la especificación IDL provea una base para su implementación. Se espera que la especificación de formatos de mensaje sean aplicados en implementaciones que trabajen tanto sincrónicamente como asincrónicamente.

Se describen tres clases distintos de mensajes:

- Mensajes de requerimientos
- Mensajes de respuestas
- Mensajes de notificación

Los mensajes de requerimientos son usados cuando un motor de workflow necesita que otro motor de workflow realice alguna acción en su beneficio. Cada mensaje de este tipo es respondido por un mensaje de respuesta, que le comunica el resultado de su requerimiento, por ejemplo, la acción requerida ha sido llevado a cabo o no ha sido posible realizar la acción requerido y el porque.
Los mensajes de notificación se usan para que el motor de workflow B, que está activando el subproceso a cuenta del motor de workflow A, lo informe de eventos significativos que ocurren durante la activación del subproceso. Cada mensaje de notificación, es respondido por un mensaje respuesta que indica al motor notificante de que ha sido recibido y entendido.

6.5.1 Funciones

- Creación de una instancia de proceso (*Create Process Instance*): Identifica la definición de proceso que se requiere activar en el motor de workflow receptor.

- Obtener el estado de la instancia de proceso (*Get Process Instance State*): Obtener el estado actual de una instancia de proceso dada, la cuál está siendo activada por otro motor de workflow.

- Setear atributos de una instancia de proceso (*Set Process Instance Attributes*): setea un conjunto de valores de atributos (process relevant data) de la instancia del proceso en una definición de proceso seleccionada. El motor de workflow destino seteará los valores correspondientes.

- Obtener atributos de una instancia de proceso (*Get Process Instance Attributes*)

- Comenzar una instancia de proceso (*Start Process Instance*): comienzo de la activación de una instancia de proceso identificada en otro motor de workflow. Ambos motores de workflow crean información de auditoría.

- Notificación de Instancia de proceso comenzada (*Process Instance Started*)

- Cancelación de una instancia de proceso (*Abort Process Instance*): cancelación de una instancia de proceso activa en un motor de workflow.

- Notificación de cancelación de una instancia de proceso (*Process Instance Aborted*): comunica al motor de workflow invocante que la activación de la instancia del proceso ha sido cancelada localmente.

- Terminación de una instancia de proceso (*Terminate Process Instance*): terminación de la activación de una instancia de proceso en otro motor de workflow.

- Notificación de terminación de una instancia de proceso (*Process Instance Terminated*) notificación que una instancia de proceso activada ha terminado prematuramente.

- Instancia de proceso finalizada (*Process Instance Completed*) notifica al motor de workflow involucrado que la representación de la instancia de proceso a finalizado normalmente.

- Lista de instancias de proceso (*List Process Instances*): retorna la lista de identificaciones de instancias de proceso seleccionadas por un criterio.

- Cambio de estado de un proceso (*Process State Changed*): notifica el cambio de estado de un subproceso en el que hay un interés registrado.

- Cambiar atributos de un proceso (*Process Attributes Changed*): setea valores de atributos de una instancia de proceso (process relevant data) de una definición de proceso seleccionada.

- Resignar de una instancia de proceso (*Relinquish Process Instance*): notifica de la resignación de una instancia de proceso el motor de workflow, liberará la memoria donde contienen la estructura de datos de la instancia correspondiente y no enviará más mensajes de notificación al respecto de esta instancia de proceso.

6.6 Detalle de Implementación

6.6.1 Administración de las sesiones y manejo de mensajes.

Para mecanismos de transporte como MAPI, las sesiones y el manejo de mensajes tienen que ver con la capa de transporte, para otros mecanismos como el correo básico de Unix será necesario un esquema el cuál garantice la integridad de diálogo entre los motores de workflow interoperantes.

Se asume que de alguna forma los mensajes, que contienen requerimientos de operaciones a ser realizadas o respuestas a requerimientos, son transmitidos desde dos o más motores de workflow interoperantes. Los mensajes son intercambiados entre los motores de workflow que interoperan durante una sesión, se asume que para cada mensaje existe respuesta.

Hay dos estilos de interoperabilidad:

1. **TRANSMISIÓN ATÓMICA** (atomic transmission) donde el diálogo entre dos motores de workflow es logrado por el intercambio de mensajes uno a uno.

Ejemplo:

A realiza un requerimiento a B
B responde informándole a A si el requerimiento ha sido exitoso o no
A realiza un requerimiento a B
B responde informándole a A si el requerimiento ha sido exitoso o no
B realiza un requerimiento a A
A responde informándole a B si el requerimiento ha sido exitoso o no

Este estilo es característico de la interoperabilidad entre dos motores de workflow que se comunican sincrónicamente. Puede ser utilizado también para una comunicación asincrónica.
La restricción central aquí es el requerimiento de un adecuado tiempo de respuesta.

2. **TRASMISIÓN BATCHED** (batched transmission) los motores de workflow que se comunican asincrónicamente (por ej., vía e-mail) pueden agregar a un batch grupos de mensajes de respuestas y enviar a ellos como una sesión al motor objetivo de workflow para procesarlo. (existen dos tipos de sesiones los batches asociados de requerimientos y los batchs de respuestas).

El motor de workflow destino procesará casa mensaje en turno y retornar una series de mensajes de respuesta una por cada mensaje enviado o retornará un mensaje batch de respuestas.

Si el motor de workflow que recibe, falla en llevar a cabo exitosamente una operación requerida, no se puede asumir el uso de transacciones y o la habilidad de deshacer un batch (roll back), lo mejor que se puede hacer es:

- Se detiene el procesamiento del batch de requerimientos.
- Se construye un mensaje reportando la falla de requerimientos y lo agrega a esta lista de mensajes de respuesta de requerimientos de operaciones exitosas.
- Se construye un mensaje que indica que la operación no se realizó y lo agrega a la lista de mensajes de respuesta de requerimientos aún no realizados.
- Se construye un mensajes de cese de sesión y lo agrega a la lista de mensajes de respuesta que será luego retornada al motor de workflow iniciador.

Es deseable que el diseñador de workflow administre el número de pedido de operación en batch, para que la complejidad de reparar el estado del workflow activado no sea tan complejo.

Cada sesión tiene un identificador de sesión de dos partes. El motor de workflow inicial asigna un identificador de sesión fuente, y el motor destino un identificador de sesión destino (el motor con el que se desea iniciar una sesión). Si solo lo asignara uno de ellos, podría ocurrir, que un motor, que trabaje con múltiples motores fuentes reciba el mismo identificador de varias fuentes, igualmente si lo nombrara solo el motor destino.
La combinación del par identificación de sesión fuente / destino es la garantía de unicidad para ambos motores. Debe ser mantenida en alguna estructura de datos concreta, representada por un tipo abstracto de datos (WMAEngineID).

Los mensajes de una sesión son numerados únicamente. Una forma de hacer esto es tratar la activación de un subproceso, como el único sujeto de la sesión e incrementar el número de mensajes a través de la vida de la instancia de subproceso. Para protegerse contra la pérdida de mensajes o la distribución fuera de secuencia.

El motor de workflow que recibe debe ser capaz de identificar:

- Mensajes que ha recibido y actuado sobre el mismo.
- Mensajes recibidos pero como están fuera de secuencia no han sido elegibles para procesar
- Mensajes que no han sido recibidos (por ejemplo si solamente se han recibido los mensajes 1, 2, y 5 se deduce que los mensajes 3 y 4 están perdidos) .

La semántica de la numeración de mensajes es “mensaje n que envié al motor X durante la sesión S” o “mensaje n que recibí desde el motor X durante la sesión S”. Para posibilitar a cada motor distinguir mensajes que ha enviado o recibido, ambos motores numeran los mensajes que envían al otro motor con incrementos de a dos, comenzando desde 0, las respuestas a mensajes de requerimientos / notificaciones son numerados agregando uno al mismo mensaje de requerimiento / notificación.

En el caso de escenarios de mensajería atómica, para evitar la posibilidad que en una sesión dos mensajes de requerimientos / notificaciones puedan tener igual número, cada motor es requerido para establecer una sesión con el otro (uno por requerimientos entrantes y uno por requerimientos de salida).

Usando los conceptos es posible de garantizar:

1. Pese a que es un ambiente donde puede haber varios motores de workflow manteniendo sesiones de interoperabilidad entre ellos, cada sesión es identificable únicamente.
2. En este ambiente cada mensaje es únicamente identificable.

6.6.2 Consideraciones de seguridad

Se asume que las implementaciones de políticas de seguridad para la administración de motores de workflow interoperantes está fuera del alcance del estándar.

6.7 Criterios de evaluación

Se presenta un esquema para que los usuarios puedan evaluar qué productos de workflow, pueden interoperar y a que niveles de interoperabilidad.
Los proveedores de workflow deberán publicar las capacidades de interoperabilidad de sus productos indicando:

- El/los mecanismos de transporte usados para efectuar la interoperabilidad con otros motores.
- El/los estilos de diálogo de interoperabilidad que pueden soportar (batched, atómico, ambos).
- El/los modelos de diálogo de interoperabilidad que emplean (half duplex o full duplex).

6.7.1 Capacidades

El factor central que afecta la habilidad de interoperar de dos motores es la capacidad esgrimida de cada uno de ellos en responder los mensajes que recibe, e iniciar requerimientos y el pasaje de datos dentro del diálogo de interoperabilidad.

Se pueden diferenciar los motores de workflow que:

- Son enteramente socios pasivos en la interoperabilidad, solo capaces de recibir instrucciones para crear e iniciar la activación de una nueva instancia de proceso y actuar en ellas.
- Son capaces del pasaje / recibir workflow relevant data, tanto como de recibir instrucciones de crear o instanciar la activación de nuevas instancias de procesos y actuar en ellas.
- Son capaces de preguntar por workflow relevant data tanto como de recibir instrucciones para crear e iniciar la activación de nuevas instancias de procesos y actuar en ellas.

Los proveedores realizarán una matriz de capacidades para cada uno de las operaciones definidas, enunciando cuándo puede el motor de workflow iniciar un mensaje asociado con esas operaciones y cuándo puede responder a ellas.

7. Internet e-mail MIME - Binding de la Especificación abstracta de interoperabilidad

7.1 Definición de MIME

Es la abreviatura de ***M**ultipurpose **I**nternet **M**ail **E**xtensions*, una especificación para formatear mensajes no-ASCII para que puedan ser enviados a través de Internet. MIME fue definido en 1992 por el Internet Engineering Task Force (IETF).
Muchos clientes de correo electrónico actualmente soportan MIME, lo que les posibilita a enviar y recibir archivos de gráficos, audio y vídeo a través del correo de Internet. Soporta mensajes en otros conjuntos de caracteres distintos al ASCII.
Hay varios tipos predefinidos de MIME, como los archivos gráficos GIF, los archivos PostScript. Es posible definir tipos propios.
Al respecto de las aplicaciones de correo electrónico los Web Browser soportan varios tipos MIME. Esto posibilita que los browser puedan desplegar archivos que no están en formato HTML.
Una nueva versión llamada S/MIME soporta mensajes encriptados.

WfMC Workflow Standard – interoperability, Internet e-mail MIME Binding WfMC-TC- 1018, 25 Set, 1998 Versión 1.1

Este documento define, dentro del alcance de la especificación de interoperabilidad [WfMC1012], un binding que provee un tipo concreto de definiciones y formatos de mensajes para la especificación abstracta con correo electrónico de Internet, usando como mecanismo de transporte la codificación MIME.
La especificación brinda definiciones concretas sobre los mensajes que fluyen entre dos motores de workflow con el propósito de efectuar la interoperabilidad.
Las definiciones abstractas de los mensajes dadas en la interface 4 [WfMC1012], son mapeadas en interfaces simples de texto, que usa al correo electrónico de Internet como método de transporte.

7.2 Mensajes MIME

Los mensajes MIME pueden usar tipos de contenido multipart/mixed o text/plain dependiendo si posee archivos adjuntos (attachments). Si éstos no son usados son del tipo text/plain.

Se describe cómo es ubicada la información de interoperabilidad en el mensaje de correo electrónico.

Estructura de los mensajes MIME:

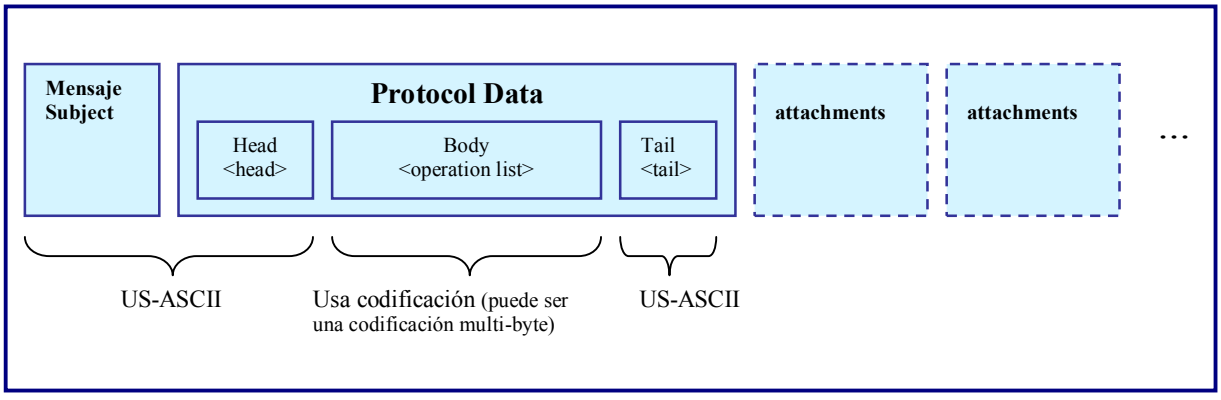


Fig. 17 Estructura mensaje MIME

Mensaje subject:

Está conformado por la sintaxis Backus Naur Form siguiente:

```
<message Subject> ::= <message type> <sequence> <conversation id> "&&"

<message type> ::= "wfmc-if4-request"
                 | "wfmc-if4-response"
                 | "wfmc-if4-error" "(" <message error> ")"

<sequence> ::= "[" <message id> "]"
```

Tipos de mensajes: indica que tipo de mensaje contiene, éstos pueden contener operaciones o respuestas pero no ambos, los mensajes más importantes son los de requerimientos y respuestas.

Los mensajes que contienen operaciones a ser realizadas por otro motor de workflow son de tipo wfmc-if4-request. El workflow destino usa las operaciones para notificar al motor de workflow de su progreso, en ambos casos se usa un mensaje de tipo wfmc-if4-request.

Los mensajes de respuesta (wfmc-if4-response) son usados por los motores de workflow fuente y destino para responder a las operaciones que le sean requeridas.

Otro tipo de mensaje son los de error (wfmc-if4-error) son los que se envían cuando el motor receptor no puede procesar la operación o mensaje de respuesta debido a que el mensaje no es válido.

Protocolo de datos (Protocol data):

En un mensaje multiparte la primera parte debe ser texto plano y contener el protocolo de datos. El resto del mensaje son consideradas archivos adjuntos. En un mensaje con una única parte (single-part) contiene solo text/plain y el protocolo de datos.
El cuerpo de la primera parte contiene una o más operaciones de requerimiento o respuesta definidas en codificación CGI (Common Gateway Interface) o esquema de codificación URL.
El final o tail de mensaje contiene la palabra "end" seguida de un chequeo de suma (cheksum)

Archivos adjuntos, (attachments) pueden ser usados para intercambiar archivos entre motores de workflow, los mensajes que los contienen deben tener una primera parte de texto donde se define el protocolo y una segunda parte (multipart/mixed) donde contiene los datos adjuntos. Pueden ser referidos por números o archivos,

7.3 Protocolo para la confiabilidad

Este protocolo de interoperabilidad es usado sobre correo electrónico, el cual es un canal de comunicación poco fiable.
Los motores de workflow están únicamente identificados a través de una casilla de correo a la cuál "escuchan". Cada mensaje que un motor de workflow envía es numerado, y es únicamente identificado por su tipo y por el lugar en la secuencia de mensajes de la sesión. Incluso con estas precauciones, pueden ocurrir varios problemas en el canal de comunicación. Este protocolo se responsabiliza de mensajes que:

- Están fuera de secuencia
- Duplicados
- Perdidos
- Truncados
- Modificados

El protocolo provee de una numeración, para el caso de los mensajes fuera de secuencia y duplicados. Incluye una estrategia de retransmisión que recupera mensajes perdidos. Un mecanismo de suma de chequeo (checksum) recupera los mensajes perdidos, además maneja los mensajes modificados y truncados.
El estándar describe la retransmisión, el manejo de mensajes duplicados, el establecimiento y fin de una sesión de conversación, el flujo de control y la recuperación de fallas.

7.4 Conformidad de proveedores con el estándar.

Un proveedor de workflow para demostrar la conformidad al estándar debe definir:

- Mecanismos/binding requeridos para efectivizar la interoperabilidad, por ejemplo Internet e-mail MIME 1.1
- El estilo de diálogo de interoperabilidad que soporta: atómico, batched o ambos.
- El estilo de diálogo soportado: half duplex, full duplex
- Tipos de procesos soportados: encadenados, anidados
- Implementación de datos de auditoría: datos auditoría de WfMC, datos de auditoría específicos del producto, si no los posee.

Debe notarse que la interoperabilidad de un motor de workflow implica más que la sola habilidad de un motor de workflow de realizar cosas requeridas por otro. Requiere que ambos motores de workflow sean capaces de realizar cosas requeridas por ambos.

7.4.1 Escenarios

Esta versión de interoperabilidad con binding MIME soporta la implementación los siguientes escenarios:

- Simplemente encadenados
- Subprocesos anidados.

7.5 Operaciones:

A continuación se detallan las operaciones incluidas en la interface:

- Cambiar el estado de una instancia de proceso (Change Process Instance State)
- Crear una instancia de proceso (Create Process Instance)
- Setear atributos de una instancia de proceso (Get Process Instance Attributes)
- Obtener el estado de una instancia de proceso (Get Process Instance State)
- Notificación de cambio de atributos de una instancia de proceso (Process Instance Attributes Changed)
- Notificación de cambio de estado de una instancia de proceso (Process Instance State Changed)
- Comienzo de una conversación (Start Conversation): conecta al motor de workflow designado, comienza una nueva conversación, ambos motores de workflow crean un registro de ella.
- Fin de conversación (Stop Conversation)

7.6 Sugerencias para la implementación (no normativas).

7.6.1 Contratos de interoperabilidad.

Los motores de workflow que cooperan en la misma organización o en organizaciones separadas, colectivamente conforman una cadena de valor. Se enumeran un conjunto de consideraciones a tener en cuenta en estos contratos de interoperabilidad:

1. Que motores de workflow, en un servicio de dominio, son visibles o capaces de interoperar con que otros motores de workflow en otros dominios de servicios.
2. Que definiciones de procesos de workflow pueden ser activadas dentro de un servicio de dominio, a requerimiento de otros motores de workflow en otros servicios de dominios.
3. Que servicio de transporte es soportado (por ej. MIME)
4. Para cada definición de proceso identificada en la cooperación:
   4.1. Que escenarios de conformidad son requeridos para efectuar la interoperabilidad
   4.2. Requerimientos de activación, por ejemplo, el ProcessDefinitionID, los atributos requeridos para activar un proceso.
   4.3. Para cada elemento compartido de datos relevantes del workflow (workflow relevant data)
      4.3.1. Derechos de acceso (escritura / lectura)
      4.3.2. Restricciones de valores (valores máximos, mínimos, número permitido de modificaciones y accesos).
   4.4. Elementos de resultados, salidas, retornados de los datos relevantes del workflow (workflow relevant data)
   4.5. Políticas de auditoría de datos.
   4.6. Cambios de política de control.
5. Políticas de seguridad e implementación.
   5.1. Autenticación
   5.2. Soporte para políticas sobre no repudio.
   5.3. Claves de criptografía y claves de administración compartidas.
   5.4. Manejo de brechas de seguridad
6. Manejo de protocolos de excepciones y recuperación, comportamiento transaccional.
7. El manejo de mensajes perdidos.
   7.1. El tiempo en que un motor de workflow debe esperar por mensajes perdidos.
   7.2. El número de intentos de reenvíos antes de desistir.
   7.3. Que hacer en el caso de desistir.
8. Acuerdos sobre niveles de servicios, métricas, escalas y penalización de desempeño.

Los contratos de interoperabilidad individuales deben tener un identificador único ContractID, determinado por las organizaciones, el cuál es usado para soportar mecanismos de autenticación.

7.6.2 Información de proceso y contrato.

Para que un workflow pueda estar bajo un contrato de interoperabilidad es necesario que el administrador configure su motor de workflow. Esta configuración incluye información de requerimientos entrantes tanto como la información requerida por requerimientos salientes. Esta información de configuración será mantenida en una base de datos, y se debe proveer de herramientas de configuración.

Se detalla a continuación un esquema simple de configuración:

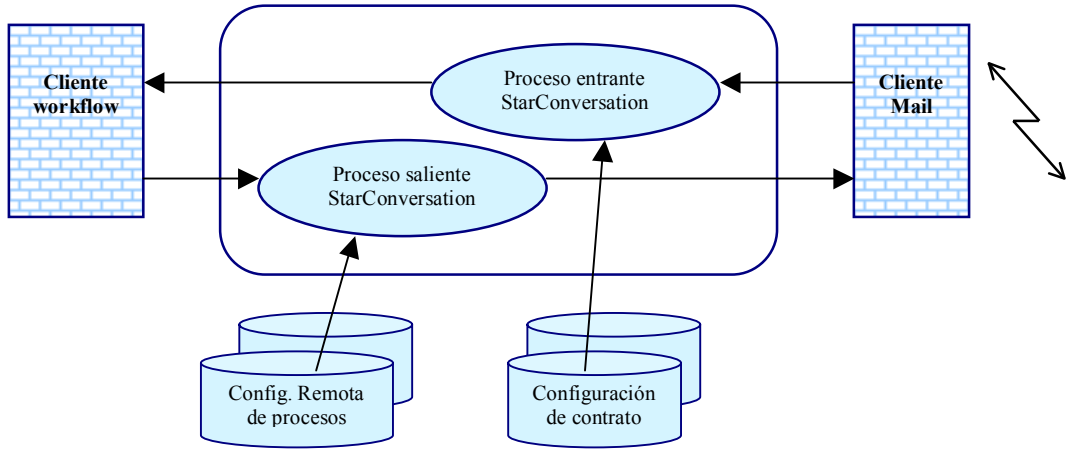


Fig. 18 Configuración del motor de workflow

Requerimientos entrantes:

Un requerimiento de entrada comienza con la operación StarConversation, el procesamiento incluye una validación de la información de contrato. Hay un archivo de configuración para cada contrato, contienen información de cuales motores de workflow acceden al sistema.

Requerimientos salientes:

Comienzan con la creación de StartConversation. Para crear un StartConversation es necesario tener información de cómo contactar el motor de workflow remoto y que acuerdo de negociación usar, esta información está contenida en el archivo de configuración remota de procesos.

7.6.3 Seteo de la casilla de correo

En este esquema cada motor de workflow es únicamente identificado por la casilla de correo a la cuál escucha (por ejemplo su dirección de correo).

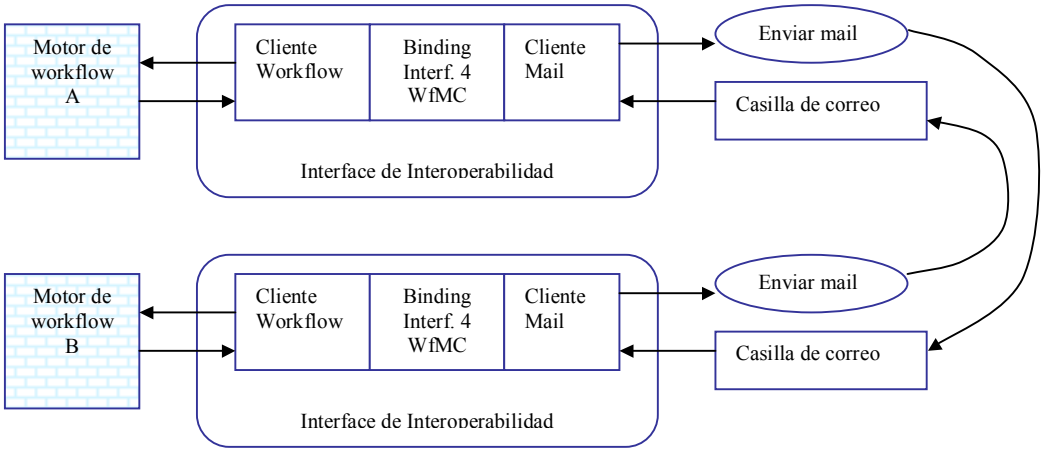


Fig. 19 Seteo de la casilla de correo

8. OMG Workflow Management Facility Standard

El estándar de la OMG Workflow Management Facility (conocido también como la especificación JointFlow) está basada en los estándares de la Wfmc.
Esta especificación describe un marco de trabajo para aplicaciones de workflow distribuidas en el mundo de los objetos de negocios. Posibilita la interoperabilidad entre componentes de procesos de workflow, el monitoreo de una ejecución de workflow y la asociación de componentes de workflow con los recursos involucrados en un proceso.

El modelo básico caracteriza a un solicitante de un servicio de workflow y a un proveedor de servicios de workflow, describe un conjunto de interfaces.

Las interfaces están basadas en un modelo de objetos que incluyen las relaciones y dependencias con los solicitantes, asignaciones y recursos.

Las interfaces centrales de workflow son definidas en el módulo **WorkflowModel**.

8.1 WorkflowModel

Algunas interfaces primarias son:

- **WfRequester**: Solicita hacer un trabajo. Puede registrar el interés en un workflow y provee operaciones que son usadas por el proceso para propagar modificaciones del estado. Recibe eventos importantes como "complete".

- **WfProcess**: realiza el trabajo solicitado por un usuario o actores automáticos como WfActivity o WfRequester, a menudo delegándolo a otras entidades. Provee operaciones para el control de ejecución del trabajo solicitado y para observar su estado.

- **WfProcessMgr**: representa la definición de un proceso de workflow, y sirve como fábrica para los procesos de workflow, y de un proceso de localización de un WfProcess.

- **WfActivity**: Es un paso en el proceso asociado, representa un ítem de trabajo en el contexto del WfProcess que la contiene. Provee un adaptador para integrar objetos de negocio existentes, en un proceso de workflow. En conjunción con la interface WfRequester, permite además, la integración del workflow principal (que contiene la actividad de workflow) y otra aplicación de workflow (que implementa la tarea definida por el workflow). Es un paso de un WfProcess y puede ser un WfRequester.

- **WfExecutionObject**: Es la clase abstracta base de WfProcess y de WfActivity, define los atributos, estados y operaciones comunes.

El estándar también define interfaces para el manejo de la lista de trabajos (WfAssignment y WfResource) y procesos de auditoría (WfEventAudit).

- **WfAssignment**: Une las actividades WfActivity con los WfResource potenciales o actuales.

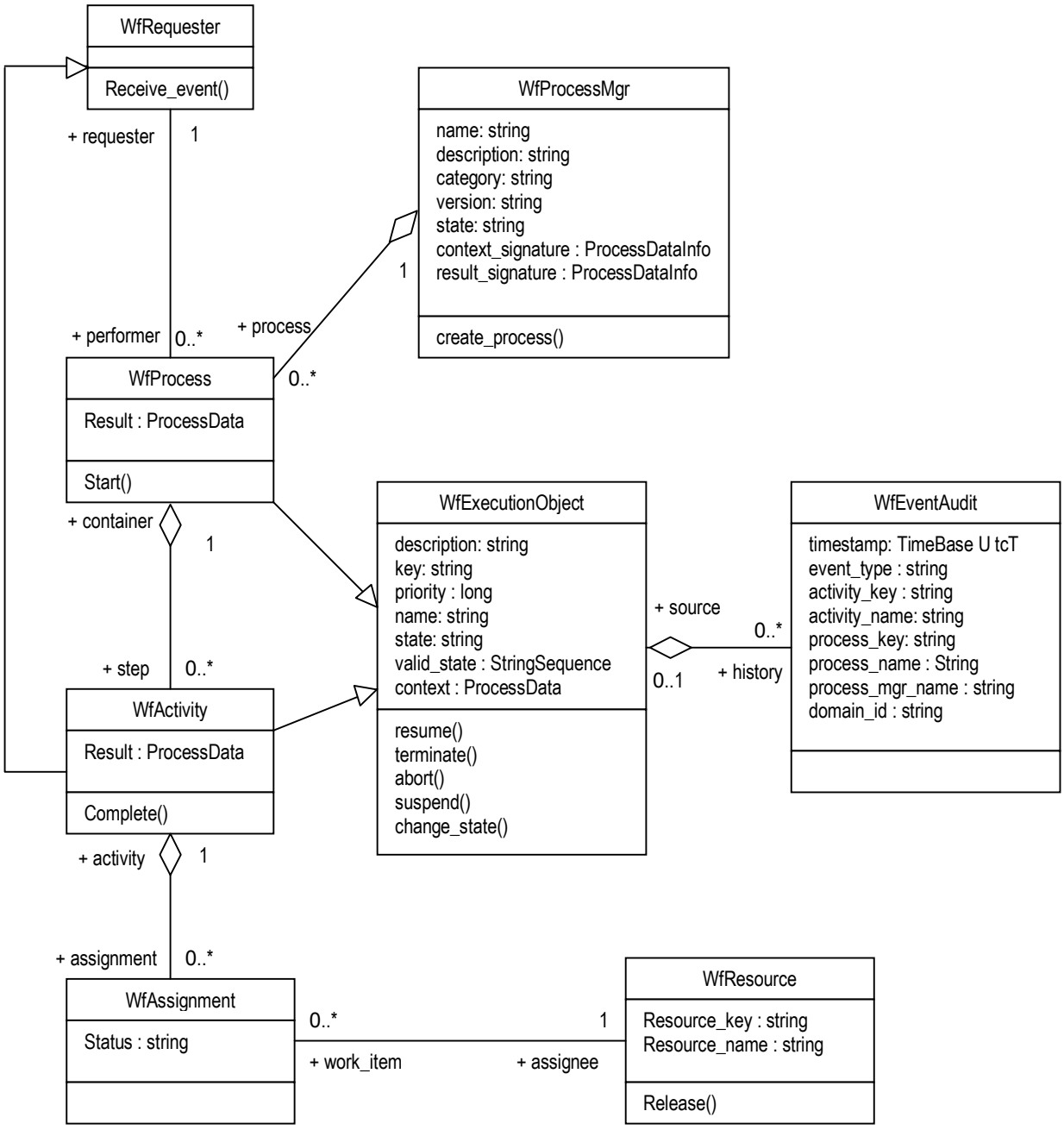
- **WfResource**: Una persona o cosa que puede realizar o aceptar una WfActivity.

- **WfEventAudit**: Es una interface común para registrar los eventos de un workflow. Se definen varios subtipos de esta interface para registrar el cambio de estado de un objeto de workflow, datos del proceso asociado con él, y cambios en la asignación de recursos en las WfActivities.

El estándar de workflow de la OMG define un modelo de objetos unificado que cubre varios estándares de la WfMC. (excepto definición del proceso, lo que sería la interface 1 de la WfMC). No pretende estandarizar las definiciones de proceso de workflow, de todas maneras, está en desarrollo un llamado por propuestas (RFP) para extender la Workflow Management Facility para estandarizar las definiciones de procesos.

En modelo de objetos se utiliza como base para la evolución de estándares de la WfMC futuros.

El modelo WorkflowModel se describe cómo clases de UML y los diagramas de interacción de objetos son especificadas por interfaces IDL. A continuación se detalla el modelo de clases:



8.2 Activación de un proceso.

Para iniciar una activación de un proceso de workflow particular, se identificará al solicitante (requester) que es responsable por un proceso. Se creará un solicitante nuevo o si ya existe será reutilizado, para observar el proceso creado.

Se identifica el administrador de procesos WfProcessMgr y se crea el proceso usando la operación create-process de este administrador.

El solicitante se asocia con el proceso en el momento de la creación del mismo y recibirá las notificaciones de cambio de estado del proceso.
Cuando el proceso es instanciado debe crear el conjunto de Actividades que representan los pasos que posee el proceso.
El proceso se inicializa seteando los datos del contexto (context data) estos datos se utilizarán para parametrizar un proceso de workflow genérico, identifica los recursos que utilizará el proceso, etc.

Para instanciar un proceso se invoca la operación start (que pertenece a WfProcess), el proceso utilizará los datos del contexto y la lógica del proceso para determinar por que actividades estará compuesto. Puede a su vez iniciar otros procesos (que serían sus subprocesos).

Cuando se inicia una actividad, se setea su contexto y se le asignan los recursos mediante la creación de asignaciones (instancias de la clase WfAssignment) que relacionan la actividad con los recursos (instancias de la clase WfResource).
Aquí no se define el mecanismo de selección de recursos, esto debe hacerse en una implementación, por ejemplo, utilizando otro proceso que determine que proceso asignar a una solicitud particular, o usando información de contexto y otros parámetros del proceso.

Una actividad a su vez puede ser implementada como un subproceso, es decir puede registrarse como un solicitante de un proceso; el subproceso se iniciará cuando la actividad sea iniciada y finalizará cuando el subproceso lo haga y su resultado será el resultado del subproceso.
Una actividad también puede ser realizada por una aplicación que usa el set-result[2] de la actividad y la operación complete para retornar los resultados y serializar la finalización de la actividad.

Cuando una actividad es finalizada (complete), sus resultados serán usados por la lógica del workflow para determinar las siguientes Actividades; y también para obtener el resultado global del proceso que la contiene.

El proceso es finalizado (complete) cuando no quedan más actividades por ser activadas, y se notifica al solicitante asociado de la completitud. El resultado del proceso queda disponible, mientras que los resultados intermedios pueden ser accedidos solo mientras el proceso está ejecutando.

8.3 Monitoreo de un proceso:

El estado global de un proceso puede ser obtenido mediante las operaciones state, get-context y get-result de WfProcess.
El solicitante asociado al proceso recibe notificaciones sobre los cambios de estado del mismo.
Puede obtenerse información más detallada sobre los estados de los pasos del proceso, si se navega por la relación de "step" entre el proceso y sus Actividades, usando el pedido de estados (get_result) a la interface de WfActivity.

La navegación entre workflows anidados puede ser soportada por la relación process entre la instancia de WfActivity (que en este caso es un solicitante WfRequester) y los subprocesos.

[2] Cuando se mencionan set-"nombre atributo" y get-"nombre atributo" se refiere a las operaciones para asignar y obtener el valor de los atributos de las clases respectivas.

Siempre que un objeto de ejecución (proceso o actividad) realiza (relevante del workflow) cambio de estado se registra en el EventAudit.
Por cada objeto que ejecuta la historia de los ítems de EventAudit pueden ser accesibles para analizar la ejecución histórica del objeto.
El EventAudit puede ser publicado usando el servicio de notificación de la OMG (OMG Notification Service).

8.4 Modulo de WorkflowModel.

Define las interfaces centrales de la Workflow Management Facility contiene las interfaces, estructuras de datos, las excepciones y los patrones que representan los atributos y relaciones de las interfaces. Se describen a continuación las interfaces del modelo.

WfRequester:

Es la interface que tiene que ver con la ejecución y los resultados de un proceso de workflow, representa la solicitud de la realización de algún trabajo. Quien lo ejecuta, (un WfProcess), administra la solicitud y comunica de los cambios de estado, en particular le informa al solicitante cuando fue finalizado el trabajo requerido. Una sola solicitud puede tener asociados varios procesos.

Usualmente un WfRequester será el objeto que inicie un proceso. Cuando un proceso se inicia algunas de las acciones de control sobre el proceso incluyen la asignación del contexto, iniciar el proceso, obtener el resultado y los estados.

Hay dos escenarios para la asociación entre un WfProcess con un WfRequester:

- Procesos anidados: Una WfActivity es un WfRequester y puede solicitar ejecutar un WfProcess. En este caso, la WfActivity será registrada como el solicitante de este subproceso cuando el WfProcess sea creado, recibirá las notificaciones de cambio de estado y cuando finalice la WfActivity estará en estado finalizado (Complete).

- Uniendo un proceso de workflow con otra aplicación, (iniciándolo o controlándolo). Cuando se utiliza para unir procesos, el solicitante debe ser un WfRequester que no sea la WfActivity de unión. Los solicitantes que no son actividades son roles o adaptadores para clientes externos.

Relaciones:

Performer: Asocia el trabajo realizado con su ejecutante. (cardinalidad=0..n)

Operaciones:

Receive_event: Es utilizada por el WfProcess para notificar al solicitante del workflow de los eventos ocurridos. El WfProcess debe notificar al solicitante los eventos de finalización, terminación o cancelación (complete, terminate, abort) o la transmisión a un estado cerrado.

WfExecutionObject:

Es una clase base abstracta de la interface que define atributos, estados y operaciones comunes a WfProcess y WfActivity.

Provee la capacidad para asignar y obtener valores de atributos y estados internos. Posee operaciones para asignar el estado actual y realizar la transmisión de estado actual a otro estado. Estas operaciones

son suspend, resume, terminate y abort. El estado retornado por estas operaciones no debe ser confundido con el estado del proceso, el cuál es calculado por el WfProcess. Los estados retornados por estas operaciones solo perteneces a los objetos de donde son retornados. Por ejemplo una actividad puede estar disponible (enabled) y el proceso globalmente estar pausado o resumido.

La interface incluye los atributos de nombre, descripción, prioridad y clave. Provee además un a operación para monitorear las ejecuciones de WfExecutionObject retornando basado en filtros específicos, registros de auditoria de eventos que representan la historia de ejecución. Otras operaciones incluyen métodos de asignar y obtener el contexto.

Atributos:

Name: describe el nombre del objeto de ejecución workflow.
Key: Identifica un objeto de ejecución único en el alcance de su objeto padre.
Context: datos relevantes del proceso que definen el contexto de definición del proceso.
Description: información que describe el objeto ejecución.
Priority: Un número que representa la prioridad de ejecución del objeto.

Relaciones:

History: Asocia datos de auditoria de eventos con su objeto de ejecución. (cardinalidad=0..n)

Operaciones:

Todas las operaciones disparan la creación de eventos de cambio de estado (WfStateEventAudit).

Resume: Solicita a la activación de una ejecución suspendida a ser resumida.

Terminate: Solicita a la activación de una ejecución que termine antes de su normal finalización.

Suspend: Solicita a la activación de una ejecución que sea suspendido.

Abort: Solicita a la activación de una ejecución que sea cancelada antes de su normal finalización.

Change_state: Modifica el estado actual de la ejecución .

WfProcessMgr

Representa la planilla de un proceso de workflow específico, es usado para crear instancias de procesos de workflow. Lógicamente es la fábrica y la localización para las instancias de WfProcess. Provee acceso a la información de metadatos sobre el contexto de un proceso y del resultado

Atributos:

Name: el nombre de un administrador de proceso
Description: describe el tipo de proceso de workflow
Category: Provee un indicador de el dominio de la aplicación para la cuál es proceso fue designado.
Versión: Define la versión del administrador del proceso.
Context_signature: Describe la estructura del contexto de los datos para el proceso.
Result_signature: Describe al estructura de los datos de resultado del proceso.

Relaciones:

Process: Localiza las instancias de proceso creado usando el WfProcessMgr. (cardinalidad=0..n)

Operaciones:

Create_process: Esta operación es usada para crear instancias del modelo de proceso y asociarlo a su solicitante.

WfProcess:

Un WfProcess es el ejecutante de la solicitud de workflow. Esta interface permite realizar trabajo asincrónico mientras es monitoreado y controlada.
Esta interface especializa la interface WfExecutionObject agregando la operación de comienzo de ejecución de un proceso, una operación para obtener el resultado producido por un proceso y relaciones con WfRequester y WfActivity.

Atributos:

Result: Resultado provisto por el proceso.

Relaciones:

Requester: Asocia el solicitante del proceso (cardinalidad=1)
Step: Contiene las actividades del proceso. (cardinalidad=0..n)
Manager: Identifica el template de la instancia. (cardinalidad=1)

Operaciones:

Start: Es usada para iniciar la activación de un WfProcess.

WfActivity

Es un paso en el proceso que esta asociado, como parte de una agregación a un único WfProcess. Representa una solicitud de trabajo en el contexto de WfProcess que la contiene. Puede haber varios objetos WfActivity en un WfProcess en un momento dado.
La interface WfActivity especializa a WfExecutionObject con operaciones que indican la finalización de un paso, y con una operación para asignar el resultado de una WfActivity. Además agrega una relación entre WfProcess y WfAssignment.

Atributos:

Result: El resultado producido por la realización de una actividad.

Relaciones:

Assignment: Une la actividad a los recursos potenciales o actuales (cardinalidad= 0..n)
Container: Une al proceso una actividad que lo compone. (cardinalidad=1)

Operaciones:

Complete: Esta operación es usada por una aplicación para indicar la finalización de WfActivity. Será usada en conjunto con la operación set_result para enviar los resultados de la actividad la proceso de workflow.

WfAssignment:

WfAssignment une las WfActivity solicitadas con los recursos WfResources potenciales o actuales. Esta interface debe ser especializada por las facilidades de administración de recursos que interpretan el contexto de las actividades para crear y negociar asignaciones con los recursos.
Las asignaciones son creadas como parte del proceso de selección de recursos antes de que una actividad se pueda ejecutar.

Relaciones:

Activity: Asocia la actividad
Assignee: Une el recurso a su asignación.

WfResource:

Es una abstracción que representa a una persona o objeto que pueda aceptar ser asignado a una actividad. Las WfAssignment potenciales y/o aceptadas son unidad entre las WfActivities solicitadas y los objetos WfResource.

Atributos:

Resource_key: Identificador único de recursos.
Resource_name: Nombre del recurso.

Relaciones:

Work_item: Provee la unión de los recursos a las asignaciones de trabajo asignadas o potenciales (cardinalidad = 0..n)

Operaciones:

Release: Es usada para notificar la recurso que no se necesita más para una asignación específica.

WfEventAudit:

Provee el registro de auditoria para la información de eventos de workflow. Provee información sobre la fuente del evento y contiene datos específicos del evento, estos eventos incluyen cambios de estado, cambios de asignación de recursos y cambios de datos.
Los eventos de workflow son persistentes y pueden ser accedidos navegando por la relaciones de la historia del WfExecutionObject.
Un objeto de evento de auditoria de workflow es creado cuando un objeto workflow cambia de estado.

Atributos:

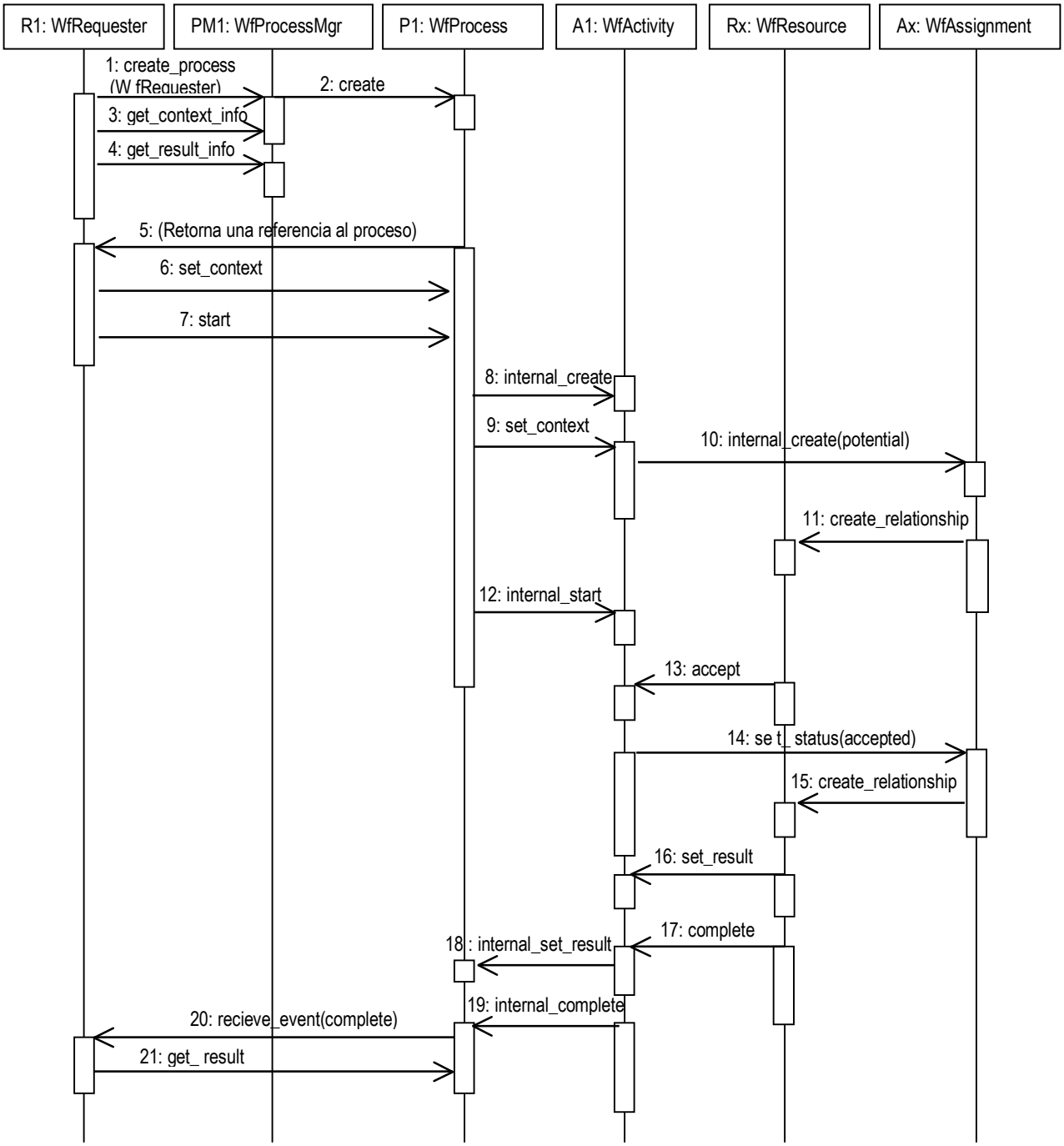
Timestamp: Registra el tiempo del evento.
event_type: Describe el tipo de evento de auditoria.
activity_key: Identifica la WfActivity asociada con el evento, es Null para los eventos de los procesos.
activity_name: Nombre de la WfActivity asociada con el evento, es Null para los eventos de los procesos.
process_key: Identifica el WfProcess asociado con el evento.
process_name: Nombre del proceso asociado con el evento.
process_mgr_name: Nombre del administrador del proceso asociado con el evento.
process_mgr_version: Versión del administrador del proceso.

Relaciones:

Source: Asocia el evento a su fuente de origen. (cardinalidad = 0..1)

8.5 Ejemplo de utilización de la interface.

El siguiente diagrama de interacción muestra uno de los posibles conjuntos de iteraciones que muestra la activación de un proceso desde su creación hasta su completitud.



8.6 Relación con los estándares existentes.

La especificación se basa en la los estándares definidos por la Wfmc: el modelo de referencia de Workflow, la interface 2 - aplicación cliente workflow y la interface 4 - especificación abstracta de interoperabilidad y los formatos de datos de auditoria.

WfMC Especificación aplicación cliente workflow (interface 2):

Esta especificación de la WfMC define las operaciones que permiten a un aplicación cliente workflow obtener la listas de entidades de workflow en tiempo de ejecución para obtener y setear sus atributos y estados.
Las interfaces de WfProcess, WfActivity, WfAssignment y WfResource representan las entidades definidas en la especificación de la WfMC y soportan sus operaciones.

WfMC Especificación de datos de auditoria (interface 5):

Esta especificación de la WfMC define los cambios de estado que se reportan por un sistema de workflow y el formato de los registros de auditoria que se producen cuando ocurre un cambio de estado.
El formato de la interface WfEventAudit y sus subtipos reflejan el formato de los datos de auditoria de la especificación de la wfmc.

WfMC Especificación abstracta de interoperabilidad (interface 4):

Esta especificación de la WfMC define las interfaces que posibilitan la colaboración entre dos sistemas de workflow.
Las interacciones entre el proceso de workflow principal y los subprocesos son soportados por la interface WfRequester heredada por la interface WfActivity, esto permite a una Activity de un Proceso principal interactuar con otro subproceso que implementa ella misma actuando como un WfRequerter. El solicitante (la actividad) es registrado en el subproceso cuando es creado y el subproceso le informará el cambio de estado (más notablemente su completitud) al mismo.

8.7 Evolución

La iniciativa de Simple Workflow Access Protocol (SWAP) y los mensajes Wf-XML son ejemplos de esta evolución.

9. SWAP - Simple Workflow Access Protocol.

La propuesta SWAP, intenta definir un protocolo de acceso para workflow basado en Internet para instanciar, controlar y monitorear instancias de proceso de workflow.

SWAP fue ideado como un binding del modelo de objetos de OMG Workflow Management Facility (jointFlow) y estándares WfMC hacia un protocolo de interacción basado en http, que pretende ser un modelo más "liviano" apto para Internet

La idea básica de la propuesta SWAP (traduciendo o interpretando interacciones entre componentes de aplicaciones workflow como mensajes codificados en XML), provee una excelente base para adaptar los estándares de la WfMC en el área de la integración de aplicaciones de empresas basado en mensajes.

Es consistente con HTTP 1.1, codifica en XML la información estructurada y no establece requerimientos sobre la plataforma o tecnología en la que residen las aplicaciones invocadas.

El modelo de objetos de SWAP define cuatros recursos de Internet: **Observers**, **ProcessInstance**, **ProcessDefinitions** y **Activities**. Estas corresponden a las interfaces del estándar de la OMG, WfRequester, WfProcess, WfProcessMgr y WfActivity.

Un sistema de workflow representa una instancia de proceso como una colección de tareas. Conceptualmente hay dos partes de una tarea, una representando la solicitud y la descripción de la tarea y la otra representando la realización o cumplimiento de la tarea.

Si el sistema de workflow, coordina trabajo realizado por un rol, cumplido por humanos, el sistema describe el pedido de la tarea mientras que la persona realiza tarea. Para tareas automáticas, el sistema de workflow realiza el requerimiento a un programa externo. Trata el invocar a un subproceso en otro sistema de workflow como invocar una tarea a un programa externo, el hecho de que realizar una tarea involucre un conjunto de tareas más finas no es problema del nivel superior (de quien lo invoca). Por esto último, para lograr la integración de workflow y de herramientas en Internet, SWAP define un protocolo donde cada una de las partes de la tarea (la parte de solicitud y quien la cumple) pueden residir en localizaciones distintas de Internet.

Todas las interacciones de SWAP involucran pares de mensajes solicitud/respuesta HTTP codificando la información en XML para ambas direcciones.

La propuesta SWAP utiliza las extensiones WEBDAV de HTTP para comunicar los pedidos de operaciones workflow. (HTTP Extensions for Distributed Authoring - WEBDAV IETF RFC 2528, febrero 1999. Y. Goland).

La propuesta SWAP fue presentada a Internet Engineering Task Force en el encuentro de diciembre de 1998.

9.1 Implementaciones del proyecto SWAP.

Se ha implementado exitosamente mediante un prototipo para procesar requerimientos de información de los vendedores uniendo el motor de workflow del programa del Departamento de Defensa de U.S. Joint Computer-aided Acquisition and Logistics Support (JCALS) con el motor de workflow MQSeries de IBM en varias áreas de las oficinas del General Dynamics Electric Boat División.

Netscape implementó una versión preliminar en su Process Manager., si se le solicita utilizando un método particular HTTP, el Process Manager retorna una estructura XML describiendo el estado del proceso.

Fujitsu implementó una versión de SWAP en un producto basado en EDI. Este producto extiende la interacción básica de proceso a proceso para incluir la creación asincrónica de instancias de procesos, un pedido batch para crear un proceso y un especificador de procesos para procesos encadenados.

10. Wf- XML - Binding de la Especificación abstracta de interoperabilidad

WfMC, Workflow Standard – Interoperability, Wf-XML Binding, Document Number WFMC-TC-1023, Document Status – Final Draft , 14 November 2001. Version 1.1.

Este documento es una especificación para un lenguaje basado en eXtensible Markup Language, diseñado para modelar el conjunto de requerimientos de transferencia de datos de la especificación abstracta de interoperabilidad para el lenguaje XML. (WfMC-TC-1012).
Este lenguaje se utilizará para implementaciones concretas de dicha especificación abstracta.

En un alto nivel, los propósitos de la interface son:

- Soportar modelos de workflow encadenados, anidados y paralelamente sincronizados.
- Proveer interacciones sincrónicas y asincrónicas
- Soportar operaciones individuales y batch.
- Continuar siendo independiente de la implementación, (pe. lenguaje de programación, mecanismos de transporte de datos, plataforma, hardware, etc.)
- Definir un protocolo liviano y fácil de implementar.

La especificación utilizará una aproximación basada en mensajes para facilitar una rápida implementación usando tecnologías actuales. Se describirá la sintaxis de los mensajes en una forma abierta, basada en estándares, para permitir un formato de comunicación estructurado, robusto y personalizable. Es por lo último que utilizará el lenguaje Extensible Markup Languaje (XML). Se basa en la experiencia obtenida en el proyecto SWAP (simple workflow access protocol) que contienen una aproximación basada en XML.

Esta especificación se basa en los conceptos centrales y las funcionalidades descriptas en la especificación abstracta de interoperabilidad (interface 4), no posee una correlación directa entre las operaciones y parámetros especificados en ella, dado que se ha desarrollado con el propósito de describir un lenguaje simple, fácil y que sea rápido demostrar la viabilidad de su implementación.
Describe un lenguaje que es independiente de un mecanismo particular de implementación, por ejemplo del lenguaje de programación, mecanismo de transporte de datos, plataforma de OS/hardware. De todos modos, como http se prevé que es el mecanismo de transporte de datos prevalente para intercambiar mensajes Wf-XML, esta especificación provee una descripción de cómo es el intercambio de los mensajes WF-XML mediante http.

Se aplica a la interoperabilidad entre sistemas de workflow pero puede aplicarse al diseño e implementación de sistemas integrados o distribuidos, utilizándose como un protocolo para la interacción de servicios genéricos (posiblemente remotos).

Se basa en los siguientes especificaciones:

- WfMC Interoperability Abstract (IF4) specification.
- OMG Workflow Management Facility (jointflow)
- WfMC IF4 Internet E-mail MIME binding specification.
- Propuesta de Simple Workflow Access Protocol (SWAP)

10.1 Especificaciones técnicas

10.1.1 Modelo lógico de recursos.

Los conceptos de interoperabilidad entre sistemas de workflow pueden ser naturalmente extendidos para ser aplicados a interacciones entre otros tipos de sistemas o servicios. Estos otros sistemas son denominados "servicios genéricos" y pueden representar cualquier recurso identificable con el cual puede ocurrir una interacción. Un servicio genérico puede ser visto como el conjunto de un cierto número de recursos diferentes. Estos recursos pueden ser implementados de varias maneras, pero deben poder ser identificados únicamente y deben poder interactuar con otros recursos de forma uniforme, recibir requerimientos para activar servicios y enviar las respuestas apropiadas a los solicitantes.

Una función de interoperabilidad individual se llama "operación". Cada operación debe pasar un conjunto de parámetros de pedido y retornar un conjunto de parámetros de respuesta.
Las operaciones están divididas en diferentes grupos, los grupos primarios de operaciones requeridos para la interoperabilidad son: **ProcessDefinition, ProcessInstance y Observer.** Existe un grupo adicional llamado **Control** que es utilizado para soportar las operaciones opcionales en esta especificación.
Un recurso implementa un grupo de operaciones si, soporta las operaciones definidas en ese grupo, del mismo modo un recurso puede implementar más de un grupo de operaciones, por ejemplo ProcessInstance y Observer.

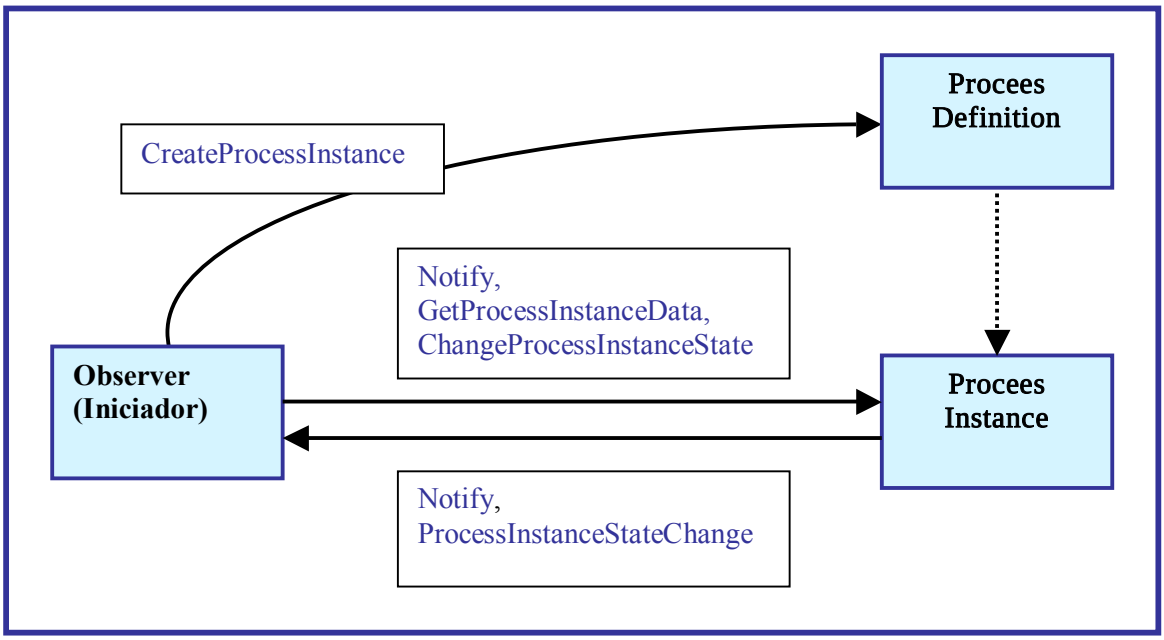
El grupo de operaciones **Control** sirve para soportar funciones del nivel de protocolo requeridas para mantener interoperabilidad con servicios genéricos. Actualmente este grupo se utiliza solo para permitir el monitoreo y control de mensajes batch. En futuras versiones de la especificación será utilizada para proveer formas más dinámicas de interoperabilidad.

El grupo **ProcessDefinition** es fundamental para la interacción con servicios genéricos. Representa la descripción de las funciones básicas del servicio y es el recurso de donde son creadas las instancias de un servicio. Dado que cada servicio a ser activado debe ser identificado unívocamente por un servicio de interoperabilidad o un solicitante de un servicio, la definición del proceso proveerá un identificador de recurso. Cuando un servicio deba ser activado este identificador de recurso será usado para referenciar el proceso deseado para ser ejecutado.

El grupo **ProcessInstance** representa la activación de una definición de proceso dada y tendrá su propio identificador de recurso distinto al de definición. Cuando un servicio es activado, el solicitante referenciará a un identificador de recurso de definición de proceso y creará una instancia de esa definición. Como una nueva instancia es creada para cada activación, la definición del proceso debe ser invocada (o instanciada) varias veces simultáneamente. De todas formas, cada instancia de proceso será única y existe solo una vez. Una vez creada una instancia de proceso será comenzada (posiblemente pausada o resumida) y eventualmente será completada o terminada.

El grupo **Observer** provee un método por el cual una instancia de proceso comunicará información sobre eventos que ocurren durante su ejecución, como su completitud o su finalización. En subprocesos anidados, debe haber un mecanismo por el cual el solicitante de la activación determine o sea informado cuando el subproceso haya finalizado. Para procesos paralelamente sincronizados (donde cada proceso cumplirá el rol de observador) debe haber una forma para que cada procesos sea informado de eventos y cambios del otro proceso. Finalmente recursos de terceros deben tener interés en el status de una instancia de proceso dada por varias razones organizacionales. El grupo observer proveerá de esta información dándole a la instancia de proceso el identificador de recurso del solicitante, que será el observador de la instancia de proceso. Si otros recursos deben ser notificados de eventos que ocurran en una instancia, es incumbencia de los observadores el pasar esta información sobre los eventos que recibe, a otros recursos.

El siguiente diagrama muestra la relaciones entre los grupos primarios de operaciones:



10.1.2 Modelo lógico de interacción

Una "interacción" en esta especificación es considerada como el intercambio de información relativa a un protocolo entre dos servicios genéricos. Wf-XML usa los "mensajes" como vehículo para lograr interacción entre servicios genéricos. Hay tres tipos de interacciones: "Request", "Acknowledgement" y "Response" que son utilizadas en los mensajes intercambiados entre servicios compatibles con la especificación Wf-XML.

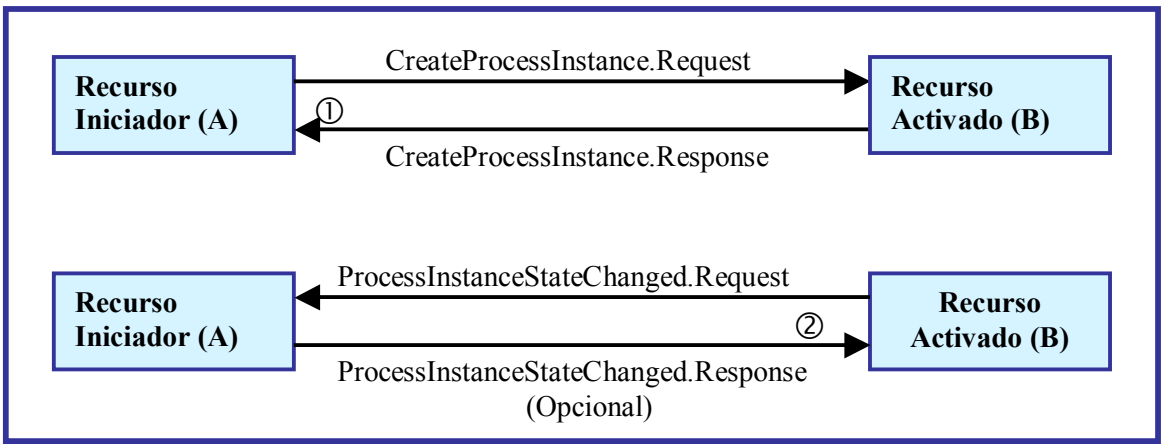
Un requerimiento "Request" es usado por un recurso (A) para iniciar una operación en un segundo recurso (B) y/o proveer datos de entrada a este recurso.

Un "Acknowledgement" es usado en las implementaciones asincrónicas por un recurso que recibe un mensaje Wf-XML para informar al emisor que el mensaje ha sido recibido. Es utilizado para dar un acuse de recibo de un mensaje, en oposición a las interacciones contenidas en el mensaje.

Un "Response" es usado por un recurso activado (B) para enviar los resultados de una operación a su demandante el recurso (A), proveyendo salidas. A pesar de que los tipos de interacciones request y response son claramente complementarias, no se requiere que sean utilizadas en conjunción. Es decir que diferente al modelo usado por http (que también utiliza los nombres Request/Response) no todos los mensajes request de Wf-XML requieren un mensaje response.

Mensajería Sincrónica

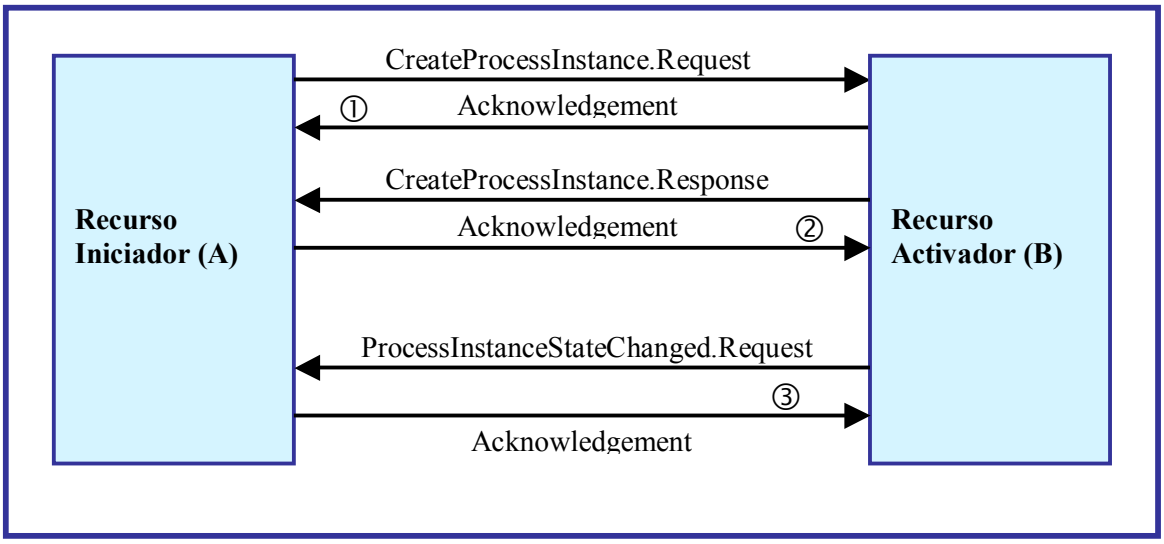
En un intercambio sincrónico un recurso (A) puede querer iniciar un subproceso en un segundo recurso (B) y suspender su proceso normal hasta que el subproceso finalice, en este punto se transforma en un observador del subproceso. Este ciclo de vida actualmente requiere dos intercambios sincrónicos separados.
Como se muestra en la figura el recurso inicializador (A) envía un mensaje request al recurso activado (B) el cuál le envía un mensaje de response (a A) indicando que el proceso ha sido iniciado. Cuando el recurso activado (B) completa el proceso le envía un mensaje de request al recurso iniciador (A), para informarle su finalización. Este mensaje puede no requerir respuesta si es meramente informativo, sin embargo es referenciado como un mensaje de Request.



Mensajería Asincrónica

En un intercambio asincrónico, el recurso iniciador (A) envía un requerimiento al recurso activado (B) para crear una nueva instancia de proceso. El recurso (B) luego envía un acknowledgement hacia el iniciador (A) informándolo que el requerimiento ha sido recibido. El acknowledgment positivo solo sirve para indicar que el mensaje ha sido recibido y no implica otra semántica adicional, como el estado de la operación. Las excepciones y estado deben ser retornadas a través de subsecuentes mensajes del protocolo.
Los requerimientos adicionales para acknowledgment negativos o otras semánticas de mensajería deben ser tratadas en el nivel de aplicación.
Más adelante en el tiempo, el recurso activante (B) envía una respuesta al recurso iniciador (A) indicando que la instancia del proceso requerido ha sido creado. El recurso iniciador (A) luego envía un acknowledgement (a B) indicando que este recibió la respuesta. Nuevamente este acknowledgement sirve solamente para indicar que la respuesta al mensaje ha sido recibida.

Cuando el proceso activado por el recurso (B) termina, el recurso (B) envía un requerimiento al recurso iniciador (A) para informarlo de su terminación. El recurso iniciante (A) luego envía un acknowledgement (a B) indicando que ha recibido el requerimiento. En este caso, el Requerimiento puede no requerir respuesta dado que este solo es informativo, y el intercambio termina aquí.



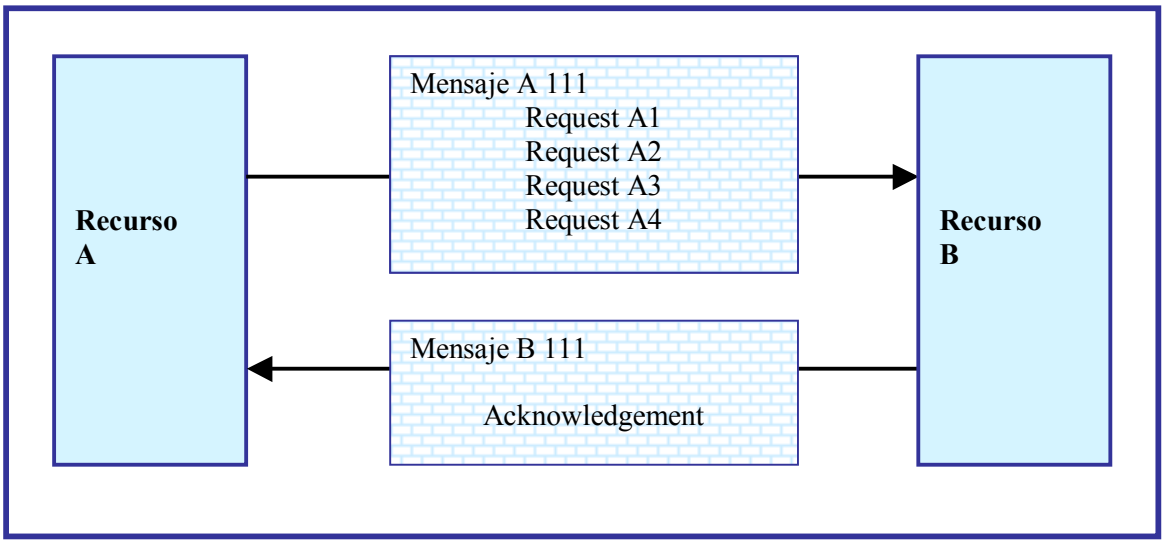
Mensajería Batch

Es deseable que en ciertas circunstancias se pueda intercambiar múltiples interacciones Wf-XML en un solo mensaje. Este tipo de procesamiento "batch" puede ser usado en situaciones de alto volumen transaccional, del estilo de transacciones EDI. Esta especificación describe un formato de datos adecuado para administrar procesamiento individual o batch.

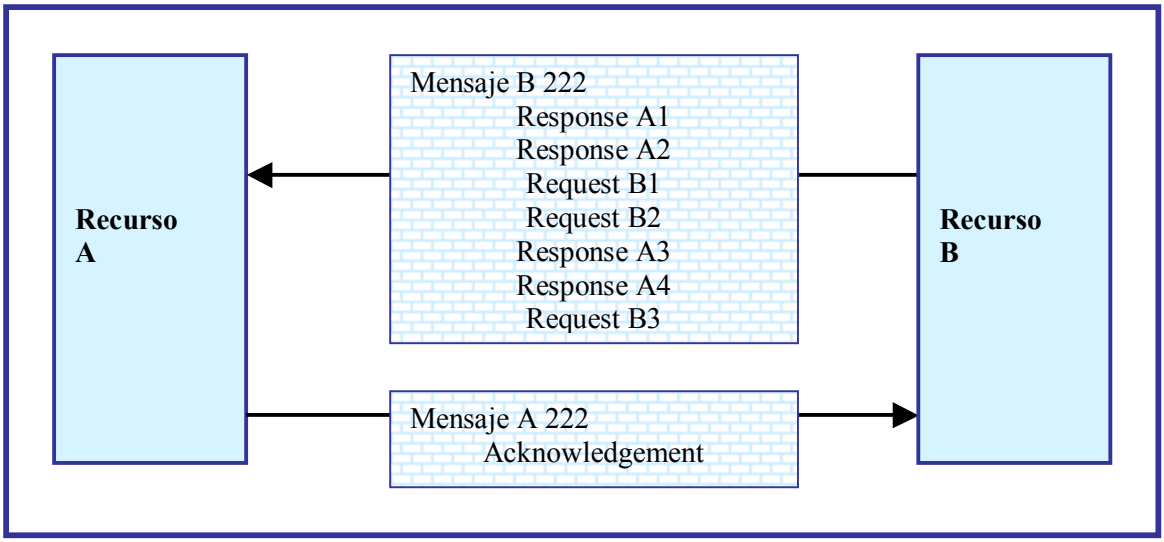
Cuando se procesan mensajes Wf-XML, los tipos de interacciones Request y Response se aplican a operaciones individuales dentro un mensaje batch, donde el tipo Acknowledgemet siempre se aplica a un mensaje en toda su totalidad. Esta es una distinción importante en el modelo de procesamiento batch, solo un mensaje de acknowledgement es requerido para un mensaje batch sin importar cuantas operaciones contenga.

Cuando intercambio interacciones batch, puedo contener solo Request, una combinación de Request y Response o solo Response. Mientras que este batching de interacciones puede ser conveniente, una implementación puede además elegir enviar Response individuales para las operaciones requeridas vía el mensaje batch. Esta aproximación puede ser usada para rastrear los progresos incrementales o resultados parciales de procesamiento.
La siguiente figura muestra un intercambio hipotético batch utilizando esta técnica. Este escenario utiliza procesamiento asincrónico para ilustrar la combinación del uso de estos modelos de procesamiento.

Mensaje batch inicial:

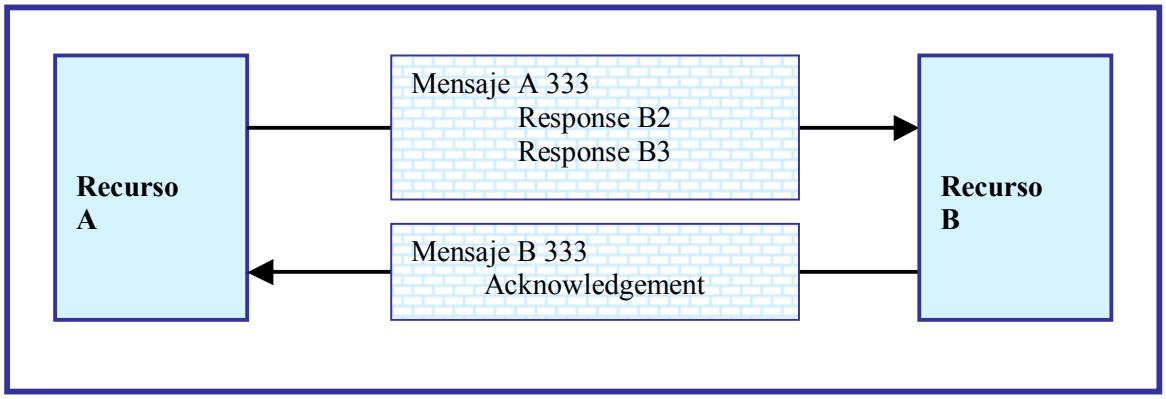


Mensaje batch con tipos de interacciones combinadas:

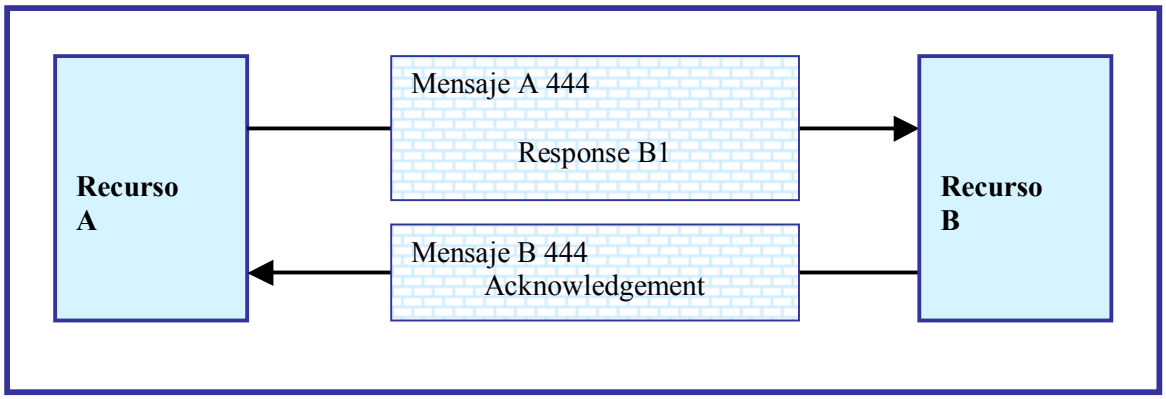


Nótese que en estos escenarios ningún recurso es explícitamente etiquetado como recurso iniciador o activador dado que cada uno de ellos puede cumplir ambos roles en algún punto en la ejecución de varios procesos de negocio.

Mensaje batch con resultados parciales:



Response individual para una operación Request batch:



Mientras que el escenario anterior muestra un intercambio de mensaje posible usando el modelo de procesamiento proporcionado, hay claramente otras formas que estos mensajes y tipos de interacciones puedan ser combinadas para acomodar diferentes requerimientos de administración de procesos.

10.1.3 Seguridad

La consideraciones de seguridad están fuera del alcance de la especificación, éstas dependen del mecanismo de transporte que use la implementación. Esto aplica a la identificación de usuarios, y autorización, encriptación y control de acceso a datos y funcional. En varios casos, los mecanismos de seguridad como SSL, PKI y LDAP pueden ser suficientes para algunas aplicaciones, pueden ser insuficientes para otros. Por lo tanto los mecanismos de seguridad usados entre dos o más servicios interoperantes deben ser identificados en el contrato de interoperabilidad existente entre ambos.

10.1.4 Definición de lenguaje Wf-XML

Cada mensaje Wf-XML en una instancia de documento XML, conforme con la especificación XML 1.0. Aunque no es requerido explícitamente por la especificación XML 1.0, el primer string que aparece en cada mensaje será la declaración XML según la forma: '<?xml version="1.0?">'. Esta declaración no contienen información explícita de codificación, e implica que ser usará la codificación por defecto de XML de UTF-8 o UTF-16.

Definición del Namespace de Wf-XML

La capacidad de interactuar con otros vocabularios de marcas XML, mezclando elementos entre ellos cuando sea necesario, es crucial para Wf-XML, tanto como para el intercambio entre sistemas de workflow y/o específicamente, para los intercambios de estos sistemas y las aplicaciones que invoque, que probablemente será enriquecido o recargado con lenguajes definidos fuera de esta especificación.
Por esta razón se crea la especificación del "Namespaces de XML" que debe ser usada en conjunto con esta especificación.

La URI del namespace que identifica a Wf-XML es :

"http://www.wfmc.org/standards/docs/Wf-XML"

Este namespace no implica la existencia o localización de ningún DTD o XML Schema, es solo para proveer un identificador único para un conjunto de elementos XML.

Para los documentos XML se debe declarar el namespace por defecto:

<WfMessage xmlns="http://www.wfmc.org/standards/docs/Wf-XML">

Alternativamente, este namespace puede ser declarado explícitamente en elementos relevantes de un mensaje, como por ejemplo:

<wf:WfMessage xmlns:wf="http://www.wfmc.org/standards/docs/Wf-XML">.

Si realizamos lo anterior en un elemento todos sus hijos que pertenezcan al namespace de Wf-XML deben poseer el prefijo " wf ". Utilizar este prefijo y no otro es recomendado para evitar colisiones.
Utilizando estas declaraciones, las aplicaciones podrán distinguir elementos definidos en esta especificación de otros, para lograr mejores niveles de interoperabilidad sin degradar la conformidad de esta especificación.

Sin embargo, debe notarse que dado la complejidad involucrada en validar documentos con múltiples namespace contra un DTD, no se provee soporte para esta funcionalidad en el Wf-XML DTD. En consecuencia los documentos Wf-XML que requieren múltiples namespaces se requiere solo que sean bien formados.

Tipos de datos

A pesar de que la sintaxis DTD no soporta tipos de datos robustos, se proveen varios tipos datos para usar en esta especificación. Una versión futura de la especificación utilizará el próximo XML Schema de la W3C, que posibilitará a estos tipos ser validados a nivel del parser XML.
Cuando se requieran, los campos de datos deberán ser de los siguientes tipos:

- Boolean: "True", "False"
- Integer: un valor numérico que no contenga decimales.
- String: secuencias de caracteres de cualquier largo.
- Date: deben contener una especificación date/time (descripta más adelante).
- URI: string conforme a la especificación Universal Resource Identification. Puede no ser absoluta, puede ser un identificador local que se resuelva por el servicio que procesa el mensaje. Adicionalmente, una implementación puede mantener internamente las URIs bases, consecuentemente solo requerir un URI relativo dentro de los mensajes Wf-XML. En estos casos, el mecanismo y la semántica para procesar URI relativas debe ser acordado en el contrato de interoperabilidad.
- UUID: string conforme a la especificación UUID[2].

El tipo por defecto de un valor es string. En los mensajes Wf-XML los datos específicos del contexto que están de acuerdo a otras especificaciones, deben ser intercambiados como datos del contexto del proceso o de resultado ("Representation of Process Context and Result Data"). Estos datos deben ser validados en base a la especificación a las que adhieren.

Valores de día y hora.

Para el tipo de datos "Date" se provee un formato Date/Time. Estos datos deben representarse en la hora de Greenwich (GMT) basado en timestamps para asegurar la interoperabilidad entre recursos de distintas zonas horarias.

Date/Time :

YYYY-MM-DDThh:mm:ssZ
Donde:
YYYY año en el calendario Gregoriano
MM mes (01 - 12)
DD día (01 - 31)
T separador entre día y hora
hh hora (00 - 24)
mm minuto (00 - 59)
ss Segundo (00 - 59)
Z es el símbolo que indica el Coordinated Universal Time (UTC) or GMT
La medianoche puede ser expresada por 00:00:00 o 24:00:00.

Existen conversiones que permiten pasar de un tiempo expresado en UTC o GMT a un tiempo en el tipo de zona de tiempo de recurso, usuario.

[2] Universal Unique Identifiers variante de DCE, es un identificador único a través del tiempo y espacio de todos los UUIDs. Consiste en un registro de 16 octetos y no debe contener caracteres de relleno entre los campos, el tamaño total es 128 bits. Puede ser usado para múltiples propósitos, para rotular objetos que poseen una vida extremadamente corta o para una identificación confiable de objetos muy persistentes a través de las redes. La generación no requiere de una autoridad de registración para cada identificador, sino que requiere un valor único sobre el espacio de cada generador de UUID.

Estructura global de un mensaje

El siguiente segmento de DTD define el elemento raíz (root) de un mensaje Wf-XML:

```
<!ELEMENT WfMessage ((WfTransport, (WfMessageHeader, WfMessageBody) *) | (WfMessageHeader,
WfMessageBody))>

<!ATTLIST WfMessage Version CDATA #FIXED "1.1"

                                    xml:space (default | preserve) #IMPLIED

                                    xml:lang NMTOKEN #IMPLIED>
```

El elemento raíz el llamado "WfMessage", tiene un atributo requerido "Versión", y los atributos reservados XML xml:space y xml:lang, con su correspondiente semántica, restricciones y significado.

Versión: Representa la versión particular de la especificación de la cual el mensaje se alinea. Esto determina cuando una implementación puede procesar un mensaje, si el servicio que recibe el mensaje no soporta la versión indicada debe retornar una respuesta conteniendo la información de la excepción apropiada de acuerdo al manejo de errores descripto en esta especificación.

xml:space: Este atributo es usado para indicar cuando un espacio en blanco en elemento es ignorado o preservado (como es especificado en la recomendación W3C XML 1.0.)

xml:lang: Este atributo es usado para indicar el lenguaje natural usado dentro del elemento. (como es especificado en la recomendación W3C XML 1.0.)

En el interior del elemento raíz WfMessage, cada mensaje Wf-XML contienen la siguiente estructura:

WfTransport:

Si es necesaria esta sección se usará para información relevante de la implementación de un protocolo de transporte. Para procesamiento asincrónico, se usará para la información convenida de acuse de recibo (acknowledgement). Para procesamiento batch, indicará que este procesamiento especial es requerido. Cuando el procesamiento es batch o asincrónico esta sección debe estar presente.

WfMessageHeader:

El mensaje puede contener 0 o más cabezales, que contendrán información relevante al ruteo y pre-procesamiento del mensaje. No está presente en un mensaje de acuse de recibo, dado que la información relevante estará en la sección de transporte.
Un mensaje con un único cabezal debe representar una operación individual, si posee varios cabezales será porque el mensaje será procesado como un batch. En este último caso, cada cabezal de mensaje será acompañado de un cuerpo de mensaje.

WfMessageBody:

La información específica de la operación es situada en el cuerpo del mensaje.
Los mensajes de acuse de recibo no contienen cuerpo del mensaje.
Cuando se quiere realizar una única operación los mensajes posee un único cuerpo de mensaje, cuando es un mensaje que se procesa como batch posee varios cuerpos de mensaje, los cuales deben ser acompañados de un cabezal correspondiente.

Por lo tanto un mensaje Wf-XML será:

```
<?xml version="1.0"?>
<WfMessage xmlns="http://www.wfmc.org/standards/docs/Wf-XML" Version="1.1">
        <WfTransport>
                …
        </WfTransport>
        <WfMessageHeader>
                …
        </WfMessageHeader>
        <WfMessageBody>
                …
        </WfMessageBody>
</WfMessage>
```

Ejemplo: Para una operación batch el mensaje será:

```
<?xml version="1.0"?>
<WfMessage xmlns="http://www.wfmc.org/standards/docs/Wf-XML" Version="1.1">
        <WfTransport>
                …
        </WfTransport>
        <WfMessageHeader>
                …
        </WfMessageHeader>
        <WfMessageBody>
                …
        </WfMessageBody>
        <WfMessageHeader>
                …
        </WfMessageHeader>
        <WfMessageBody>
                …
        </WfMessageBody>
                …
</WfMessage>
```

Ejemplo: Para un mensaje de acuse de recibo (usado en un procesamiento asincrónico) será:

```
<?xml version="1.0"?>
<WfMessage xmlns="http://www.wfmc.org/standards/docs/Wf-XML" Version="1.1">
        <WfTransport>
                …
        </WfTransport>
</WfMessage>
```

Mecanismo de transporte de los mensajes.

Uno de las metas de esta especificación es proveer un protocolo independiente de la implementación. Uno de los aspectos importantes de esta independencia es la habilidad de intercambiar mensajes Wf-XML sobre cualquier mecanismo de transporte. En consecuencia esta especificación no define ninguna de las características de los protocolos o mecanismos soportados, usados para el intercambio de mensajes Wf-XML. Como por ejemplo, detalles referidos a la integridad, confiabilidad (retrasmisión, detección de duplicados) de los mensajes, administración de sesiones, etc.
De todos modos, a menudo es necesario proveer información para soportar estas capacidades en los mensajes Wf-XML. Por esta razón se provee de la sección WfTransport.
La WfTransport, es la una sección opcional que contiene constructores de marcas designadas para facilitar la implementación de mecanismos de transporte asincrónico y batch.
Los detalles de la WfTransport, relativos a los requerimientos de un trasporte en particular deben ser especificados por el protocolo de binding del trasporte considerado.

El siguiente DTD muestra la estructura predefinida de una sección de WfTransport:

```
<!ELEMENT WfTransport (Dialog?, CorrelationData?, Exception?)>

<!ELEMENT Dialog ((Acknowledgement, Key) | (ReplyToKey, Key?) | Key)?>

<!ATTLIST Dialog Type (synch | asynch) ”synch”

                 Mode (individual | batch) ”individual”
                 MessageID CDATA #IMPLIED>

<!ELEMENT Acknowledgement EMPTY>

<!ATTLIST Acknowledgement ReceivedAt CDATA #REQUIRED>

<!ELEMENT Key (#PCDATA)>

<!ELEMENT ReplyToKey (#PCDATA)>

<!ELEMENT CorrelationData (#PCDATA)>
```

Estos elementos y atributos pueden ser combinados de forma diferentes para soportar una variedad de modelos de mensajes (individual/sincrónico, individual/asincrónico, batch/sincrónico o batch/asincrónico). Cada uno de ellos es opcional como el elemento WfTransport, es opcional para permitir binding de transporte específicos e implementaciones. Si toda el elemento WfTransport es omitido, se refiere a un mensaje individual/sincrónico.

La estructura semántica es la siguiente:

Dialog: Contiene la información relevante para el tipo de dialogo que se establece entre los servicios interoperantes, como cuando las respuestas deben ser sincrónicas o asincrónicas y cuando el mensaje contiene una interacción múltiple o simple. Características específicas del procesamiento es requerido para soportar este mensaje que describe por los elementos y atributos siguientes. Si este elemento es omitido se trata de un mensaje individual/sincrónico (opcional).

Type: Es usado para indicar cuando el mensaje es tratado como sincrónico o asincrónico. Si el atributo tiene valor “synch” el mensaje es sincrónico, en este caso no ocurrirá más comunicación hasta

que el procesamiento requerido sea finalizado. Luego de la finalización una respuesta debe ser retornada al servicio solicitante inmediatamente.
Si el atributo tiene valor "asynch" el mensaje es asincrónico. El servicio que lo recibe debe retornar un acknowledgement al servicio iniciador como recibo de la solicitud.
Después de la finalización del procesamiento solicitado una respuesta es enviada al servicio iniciante. Si la respuesta es retornada, el servicio iniciante debe enviar un acknowledgement al servicio activante como acuse de recibo de la respuesta.
Si el atributo es omitido se trata del tipo de mensaje "synch" (opcional).

Mode: El valor de este atributo indica la clase de procesamiento que se requiere para este mensaje para el recurso que lo recibe. Si el valor del atributo es "individual" el mensaje debe ser procesado como una interacción simple. Si el atributo es "batch" la información adicional para su procesamiento se especificará en el mensaje. Si el atributo es omitido, el modo del mensaje es "individual". (opcional)

MensajeID: El atributo debe estar presente cuando el procesamiento es asincrónico o batch. Contiene un identificador único usado para correlacionar los acknowledgement de los mensajes, y/o para identificar el mensaje batch en operaciones de control. En un acknowledgement, el valor del atributo debe corresponder con el valor del mismo atributo del mensaje en el cual relaciona el acknowledgement. El valor de este atributo debe ser del tipo UUID. (opcional)

Acknowledgement: La presencia de este elemento indica que es un acknowledgement, usado en el procesamiento asincrónico. Luego este elemento no debe estar presente si el valor del atributo Type en el elemento Dialog esta seteado como "synch". Si el elemento esta presente en el mensaje no debe contener un cabezal o cuerpo de mensaje. Luego un acknowledgement solo debe ser dar acuso de recibo de un mensaje simple y debe ser procesado individualmente.
Consecuentemente, el atributo Mode en el elemento Dialog debe ser seteado a "individual" cuando este elemento esté presente, porque un mensaje de acknowledgement no debe requerir el procesamiento de información batch. El acuso de recibo de este mensaje indica que el mensaje correspondiente, identificado por el valor atributo MessageID , ha sido recibido. (opcional)

ReceivedAt: El valor de este atributo indica que el tiempo en el que el mensaje del acknowledgement ha sido recibido por el receptor. El valor de este atributo debe ser del tipo Date.

Key: Este elemento es usado dentro de la sección de transporte del mensaje cuando el procesamiento es batch y/o asincrónico. Complementa al elemento Key en el cabezal del mensaje en dos formas:

Cuando el procesamiento es batch el mensaje contiene múltiples cabezales, haciendo imposible el uso del elemento Key en el cabezal para rutear los mensajes. Luego este elemento provee un identificador del recurso a donde el mensaje batch es enviado.

Cuando el procesamiento es asincrónico no hay forma de incluir el elemento Key en un mensaje de acknowledgement, dado que no contiene cabezal. Luego este elemento provee un identificador del recurso al cuál el acknowledgement es enviado. El contenido de este elemento debe ser del tipo URI. (opcional)

ReplyToKey: Este elemento debe estar presente en los mensajes que contienen Request o Response en el procesamiento asincrónico. Contiene el identificador del recurso a donde el acknowledgement o respuesta debe ser enviada. Este elemento debe estar presente en mensajes que contienen un acknowledgement. El contenido de este elemento debe ser del tipo URI. (opcional)

CorrelationData: Este elemento contiene los datos específicos de la implementación y o estructuras necesarias para correlacionar el tráfico del mensaje entre recursos interoperantes. Dado que estos datos serán particulares a la interacción entre dos recursos interoperantes, los detalles específicos de sus estructuras y formatos deben ser acordados el contrato de interoperabilidad. Este elemento debe mantener en esta versión de la especificación para compatibilidad hacia atrás. (opcional)

Exception: El elemento que aparece aquí es usado para describir un error que debe ser encontrado relativo a la sección de transporte. El contenido de este elemento es descrito en sección de manejo de errores. Cuando el elemento es usado en la sección de transporte del mensaje, la excepción código 800 "WF_OTHER" debe ser usado y extendido como necesario para transmitir el error.

Los siguientes ejemplos muestran las posibles formas que la estructura de la sección WfTransport debe soportar varios modelos de procesamiento.

Ejemplo: Muestra una sección de transporte para un intercambio sincrónico de un mensaje individual.

```
<?xml version="1.0"?>
<WfMessage xmlns="http://www.wfmc.org/standards/docs/Wf-XML" Version="1.1">
        <WfTransport>
                <Dialog Type="synch" Mode="individual"/>
        </WfTransport>
                …
</WfMessage>
```

Se debe notar que dado este modelo de procesamiento, alguno o todos los atributos Type, atributos Mode y elemento Dialog o el elemento WfTransport debe ser omitido sin impactar la semántica de procesamiento de mensaje.

Ejemplo: Muestra una sección de transporte de un intercambio asincrónico de una solicitud o respuesta individual.

```
<?xml version="1.0"?>
<WfMessage xmlns="http://www.wfmc.org/standards/docs/Wf-XML" Version="1.1">
        <WfTransport>
                <Dialog Type="asynch" Mode="individual" MessageID="5a98d32e-7854-c751-
                5491-7d4a0c4e7102">
                        <ReplyToKey>http://www.myco.com/purchasing/orders</ReplyToKey>
                </Dialog>
        </WfTransport>
                …
</WfMessage>
```

Ejemplo: Muestra una sección de transporte para un intercambio asincrónico de un mensaje batch.

```
<?xml version="1.0"?>
<WfMessage xmlns="http://www.wfmc.org/standards/docs/Wf-XML" Version="1.1">
        <WfTransport>
                <Dialog Type="asynch" Mode="batch" MessageID="5a98d32e-7854-c751-5491-
                8f55e8a210ba">
                        <ReplyToKey>http://www.myco.com/purchasing/orders</ReplyToKey>
                        <Key>http://www.exampleco.com/processes</Key>
                </Dialog>
        </WfTransport>
                …
</WfMessage>
```

Ejemplo: muestra un acknowledgement de un mensaje batch del ejemplo anterior. El valor del atributo Mode en el elemento Dialog ha sido seteado a "individual" a pesar de que es un acknowledgement de un mensaje batch, esto es porque el atributo Mode indica el modo de procesamiento requerido para manejar este mensaje. El mensaje que es acknowledgement es asociado vía el MessageID.

```
<?xml version="1.0"?>
<WfMessage xmlns="http://www.wfmc.org/standards/docs/Wf-XML" Version="1.1">
        <WfTransport>
                <Dialog Type="asynch" Mode="individual" MessageID="5a98d32e-7854-c751-
                5491-8f55e8a210ba">
                        <Acknowledgement ReceivedAt="2001-09-24T16:31:05Z">
                        <Key>http://www.myco.com/purchasing/orders</Key>
                </Dialog>
        </WfTransport>
</WfMessage>
```

Definición del cabezal de un mensaje.

El cabezal de un mensaje contiene información que es genérica usada para todas las interacciones, como identificadores de recursos, nombres de operaciones, etc. La separación de está información del cuerpo de mensaje posibilita el pre-procesamiento de mensajes Wf-XML sin tener que parsear la información específica de la operación.

El cabezal se define como:

```
<!ENTITY % ISOLangs
"(aa|ab|af|am|ar|as|ay|az|ba|be|bg|bh|bi|bn|bo|br|ca|co|cs|cy|da|de|dz|el|en|eo|es|et|eu|fa|fi|fj|fo|fr|fy|ga|gd|gl|gn|gu|h
a|hi|hr|hu|hy|ia|ie|ik|in|is|it|iw|ja|ji|jw|ka|kk|kl|km|kn|ko|ks|ku|ky|la|ln|lo|lt|lv|mg|mi|mk|ml|mn|mo|mr|ms|mt|my|n
a|ne|nl|no|oc|om|or|pa|pl|ps|pt|qu|rm|rn|ro|ru|rw|sa|sd|sg|sh|si|sk|sl|sm|sn|so|sq|sr|ss|st|su|sv|sw|ta|te|tg|th|ti|tk|tl|tn|t
o|tr|ts|tt|tw|uk|ur|uz|vi|vo|wo|xh|yo|zh|zu)">

<!ELEMENT WfMessageHeader ((Request | Response), Key)>

<!ELEMENT Request EMPTY>

<!ATTLIST Request ResponseRequired (Yes | No | IfError) #REQUIRED

                        ResponseLang %ISOLangs; #IMPLIED
                        RequestID CDATA #IMPLIED>

<!ELEMENT Response EMPTY>

<!ATTLIST Response RequestID CDATA #IMPLIED>
```

La estructura semántica es la siguiente:

Request: Indica que la interacción es una solicitud. Si es utilizado debe contener el atributo ResponseRequired. Es opcional especificar el atributo ResponseLang. (opcional)

ResponseRequired: Debe contener el valor Si (se requiere una respuesta para todos los casos y debe ser procesado por el recurso solicitante), no (puede existir una respuesta pero no es necesaria para esta solicitud, si existe puede ser ignorada por el recurso solicitante) o IfError (solo se necesita una respuesta a la solicitud en caso que ocurra un error al procesarla, y el recurso solicitante debe procesar la respuesta).

ResponseLang: Indica el lenguaje usado en los elementos de datos como el Subject o la Description en la información es retornada. Este valor se elige de la lista de identificadores definidos en

el estándar ISO 639 para identificadores de lenguajes. Si no se utiliza no se puede realizar ninguna afirmación sobre el lenguaje utilizado. (opcional)

RequestID: Es usado en procesamiento asincrónico y batch para identificar únicamente una solicitud, para que ser más tarde pueda ser relacionada con su respuesta. Debe ser de tipo UUID. (opcional)

Response: La presencia de este elemento indica que es una respuesta. Es un acuse de recibo de que la interacción indicada que la solicitud correspondiente ha sido cumplida. (opcional)

RequestID: Este atributo del elemento Response es usado para el procesamiento asincrónico y batch para relacionar la respuesta a la solicitud. Debe corresponderse con el valor de un RequestID de una solicitud previamente enviada. Su tipo debe ser UUID (opcional)

Key: Contiene el identificador del recurso a quien va dirigida la solicitud o la respuesta. En el caso de procesamiento de un mensaje batch, el ruteo del mensaje es indicado por el elemento Key de la sección de transporte, y el contenido es usado por el procesador de Wf-XML para identificar un recurso en particular al cuál aplica esta operación. El tipo debe ser URI.

Un mensaje de solicitud individual asincrónico sería:

```
<?xml version="1.0"?>
<WfMessage xmlns="http://www.wfmc.org/standards/docs/Wf-XML" Version="1.1">
    <WfTransport>
        <Dialog Type="asynch" MessageID="4308d23b-e78c-2390-6271-
        743891d60a52">
            <ReplyToKey>
                http://www.myco.com/purchasing/orders/4089259
            </ReplYToKey>
        </Dialog>
    </WfTransport>
    <WfMessageHeader>
        <Request ResponseRequired="Yes" RequestID="4308d23b-675d-3b47-0931-
        768c4a0528b3"/>
        <Key>http://www.exampleco.com/processes/86947325</Key>
    </WfMessageHeader>
    <WfMessageBody>
        …
    </WfMessageBody>
</WfMessage>
```

Un acuse de recibo a este mensaje de solicitud sería:

```
<?xml version="1.0"?>
<WfMessage xmlns="http://www.wfmc.org/standards/docs/Wf-XML" Version="1.1">
    <WfTransport>
        <Dialog Type="asynch" MessageID="4308d23b-e78c-2390-6271-
        743891d60a52">
            <Acknowledgement ReceivedAt="2001-09-24T16:48:30Z"/>
            <Key>http://www.myco.com/purchasing/orders/4089259</Key>
        </Dialog>
    </WfTransport>
</WfMessage>
```

Una respuesta a esta solicitud sería:

```
<?xml version="1.0"?>
<WfMessage xmlns="http://www.wfmc.org/standards/docs/Wf-XML" Version="1.1">
```

```
<WfTransport>
    <Dialog Type="asynch" MessageID="4308d23b-430b-29e0-a867-
    32087b581afe">
        <ReplyToKey>http://www.exampleco.com/processes/86947325</ReplyT
        oKey>
    </Dialog>
</WfTransport>
<WfMessageHeader>
    <Response RequestID="4308d23b-675d-3b47-0931-768c4a0528b3"/>
    <Key>http://www.myco.com/purchasing/orders/4089259</Key>
</WfMessageHeader>
<WfMessageBody>
    …
</WfMessageBody>
</WfMessage>
```

Si utilizamos un escenario asincrónico, la respuesta será un acuse de recibo (acknowledgement) como el siguiente:

```
<?xml version="1.0"?>
<WfMessage xmlns="http://www.wfmc.org/standards/docs/Wf-XML" Version="1.1">
    <WfTransport>
        <Dialog Type="asynch" MessageID="4308d23b-430b-29e0-a867-
        32087b581afe">
            <Acknowledgement ReceivedAt="2001-09-24T16:54:14Z"/>
            <Key>http://www.exampleco.com/processes/86947325</Key>
        </Dialog>
    </WfTransport>
</WfMessage>
```

Definición del cuerpo de un mensaje.

El cuerpo de un mensaje contiene información específica de las operaciones.
Para una operación solicitud (request), contiene un elemento <OperationName.Request>, que contiene los parámetros del requerimiento. Para una operación de respuesta (response), contiene un elemento <OperationName.Response> conteniendo los parámetros de resultado de la operación.

El contenido de un elemento WfMessageBody es definido como:

```
<!ELEMENT WfMessageBody (%OperationRequest; | %OperationResponse;)>
```

Las entidades referenciadas aquí contienen declaraciones de elementos para operaciones específicas que serán especificadas en la sección Operaciones.

El modelo tendrá la siguiente estructura para interacciones de solicitud asincrónicas batch. Nótese que deben ser combinadas múltiples tipos de interacciones (solicitudes y respuestas) en un mensaje batch. El ejemplo muestra solo uno de los posibles escenarios para ilustrar el ciclo de procesamiento asincrónico.

```
<?xml version="1.0"?>
<WfMessage xmlns="http://www.wfmc.org/standards/docs/Wf-XML" Version="1.1">
    <WfTransport>
        <Dialog Type="asynch" Mode="batch" MessageID="e85327bc-dc9e-2878-4b52-
        947852107643">
            <ReplyToKey>http://www.myco.com/purchasing/orders</ReplyToKey>
            <Key>http://www.exampleco.com/processes</Key>
        </Dialog>
    </WfTransport>
```

```
<WfMessageHeader>
    <Request ResponseRequired="Yes" RequestID="d4253789-ce42-9165-3bed-
    825731d8d941"/>
    <Key>http://www.exampleco.com/processes/86947325</Key>
</WfMessageHeader>
<WfMessageBody>
    <OperationName.Request>
        …
    </OperationName.Request>
</WfMessageBody>
<WfMessageHeader>
    <Request ResponseRequired="Yes" RequestID="54879aac-ffe3-8d92-cd74-
    7983547bac21"/>
        <Key>http://www.exampleco.com/processes/79843209</Key>
</WfMessageHeader>
<WfMessageBody>
    <OperationName.Request>
        …
    </OperationName.Request>
</WfMessageBody>
<WfMessageHeader>
    <Request ResponseRequired="Yes" RequestID="25b76322-8ac6-e509-baca-
    172483dabcf3"/>
    <Key>http://www.exampleco.com/processes/30817842</Key>
</WfMessageHeader>
<WfMessageBody>
    <OperationName.Request>
        …
    </OperationName.Request>
</WfMessageBody>
</WfMessage>
```

El siguiente es un acuse de recibo (acknowledgement) de este mensaje:

```
<?xml version="1.0"?>
<WfMessage xmlns=http://www.wfmc.org/standards/docs/Wf-XML Version="1.1">
    <WfTransport>
        <Dialog Type="asynch" Mode="individual" MessageID="e85327bc-dc9e-2878-
        4b52-947852107643">
            <Acknowledgement ReceivedAt="2001-08-23T17:12:42Z"/>
            <Key>http://www.myco.com/purchasing/orders</Key>
        </Dialog>
    </WfTransport>
</WfMessage>
```

Una respuesta de esta solicitud tendría la siguiente forma:

```
<?xml version="1.0"?>
<WfMessage xmlns="http://www.wfmc.org/standards/docs/Wf-XML" Version="1.1">
    <WfTransport>
        <Dialog Type="asynch" Mode="batch" MessageID="832ba946-8754-4237-e8f0-
        924678a8e031">
            <ReplyToKey>http://www.exampleco.com/processes/order-handler.
            jsp</ReplyToKey>
            <Key>http://www.myco.com/purchasing/orders</Key>
        </Dialog>
    </WfTransport>
    <WfMessageHeader>
        <Response RequestID="d4253789-ce42-9165-3bed-825731d8d941"/>
        <Key>http://www.exampleco.com/processes/86947325</Key>
    </WfMessageHeader>
```

```
        <WfMessageBody>
            <OperationName.Response>
            …
            </OperationName.Response>
        </WfMessageBody>
        <WfMessageHeader>
            <Response RequestID="54879aac-ffe3-8d92-cd74-7983547bac21"/>
            <Key>http://www.exampleco.com/processes/79843209</Key>
        </WfMessageHeader>
        <WfMessageBody>
            <OperationName.Response>
                …
            </OperationName.Response>
        </WfMessageBody>
        <WfMessageHeader>
            <Response RequestID="25b76322-8ac6-e509-baca-172483dabcf3"/>
            <Key>http://www.exampleco.com/processes/30817842</Key>
        </WfMessageHeader>
        <WfMessageBody>
            <OperationName.Response>
                …
            </OperationName.Response>
        </WfMessageBody>
</WfMessage>
```

Finalmente un acuse de recibo (acknowledgement) de la respuesta tendría la siguiente forma:

```
<?xml version="1.0"?>
<WfMessage xmlns="http://www.wfmc.org/standards/docs/Wf-XML" Version="1.1">
    <WfTransport>
        <Dialog Type="asynch" Mode="batch" MessageID="832ba946-8754-4237-e8f0-
        924678a8e031">
            <Acknowledgement ReceivedAt="2001-09-24T17:32:10Z"/>
            <Key>http://www.exampleco.com/processes/order-handler.jsp</Key>
        </Dialog>
    </WfTransport>
</WfMessage>
```

Representación de datos del contexto del proceso y del resultado.

Típicamente un proceso es asociado con un número de ítems de datos específico de un proceso. Estos datos representan las propiedades de la Instancia de proceso (Datos de control y relevantes del Workflow), y/o datos relativos a las aplicaciones invocadas durante la activación del proceso (datos de la aplicación). Este conjunto de ítems de datos es llamado "contexto del proceso" cuando el proceso es instanciado, y el "resultado" del proceso cuando el proceso ha sido completado/terminado.
Cuando el proceso es activado, estos ítems de datos deben ser especificados y accesibles. Para este propósito, esta especificación provee un lugar para identificar estos ítems de datos en forma de elementos llamados ContextData y ResultData. Cuando una definición de proceso es instanciada, el contexto es inicializado con el contenido de los elementos del ContextData. Cuando una instancia de proceso es completado, los datos resultantes son intercambiados como el contenido del elemento ResultData.

Los datos de estos elementos pueden tomar varias formas, dependiendo de los tipos de datos que son intercambiados y los requerimientos particulares de una implementación. La estructura de estos datos pueden variar de una definición de proceso a otra. Es por ello que se recomienda que la descripción de los datos de contexto y resultado sean acordados a través del contrato de interoperabilidad de los servicios.

Como la naturaleza y definición de los contextos de datos no pueden ser conocidos (y son particulares a cada definición de proceso), es difícil de identificar marcas especiales para ello.
El contenido de los elementos ContextData y ResultData deben ser especificados caso a caso. Se utilizará un carácter comodín "ANY" que representa un modelo de contenido por defecto para estos elementos.
Los atributos reservados xml:space y xml:lang se utilizan aquí para permitir que éstas características sean sobrescritas en relación a las del elemento raíz, para los datos del contexto específicos.
Las especificaciones apropiadas del contexto de estos elementos deben hacerse en el contrato de interoperabilidad entre los servicios, extendiendo así esta especificación para lograr alcanzar necesidades específicas.

```
<!ELEMENT ContextData ANY>

<!ATTLIST ContextData xml:space (default | preserve) #IMPLIED
                      xml:lang NMTOKEN #IMPLIED>

<!ELEMENT ResultData ANY>

<!ATTLIST ResultData xml:space (default | preserve) #IMPLIED
                     xml:lang NMTOKEN #IMPLIED>
```

Mientras la flexibilidad provista por este modelo de contenido es esencial, un modelado más estructurado de los datos permitirá una mejor interoperabilidad; proveyendo del significado por el cuál un procesador Wf-XML pueda distinguir los campos que contienen la sección de datos específicos de contexto. Estos campos pueden luego, ser tratados por separado para ser procesados apropiadamente, como determinará la implementación. En orden de adoptar mayores niveles de interoperabilidad, esta versión de la especificación Wf-XML provee las siguientes marcas para ser usadas para especificar parámetros específicos en los elementos de ContextData y ResultData.

```
<!ELEMENT Parameter (Name, Value+)>

<!ELEMENT Name (#PCDATA)>

<!ELEMENT Value (#PCDATA)>
```

Los elementos tienen la siguiente semántica:

Parameter: Este elemento provee un mecanismo limitado usado para indicar que su contenido constituye un parámetro único del proceso en el que la operación ha sido ejecutado. Posee dos propiedades Name y Value. De todas maneras otras restricciones semánticas y sintácticas de este parámetro serán determinadas por los acuerdos del contrato de interoperabilidad. Este elemento puede aparecer cero o más veces en los elementos ContextData y/o ResultData.

Name: Este elemento provee un identificador de un parámetro. Las restricciones semánticas y sintácticas de este elemento serán determinadas por los acuerdos del contrato de interoperabilidad.

Value: Este elemento constituye el valor asignado al parámetro. Pueden incluirse múltiples valores especificando múltiples elementos Value de un parámetro. Las restricciones semánticas y sintácticas de este elemento serán determinadas por los acuerdos del contrato de interoperabilidad.

El siguiente ejemplo muestra como deben ser intercambiados los datos específicos del contexto asumiendo que se realizaron los acuerdos apropiados en el contrato de interoperabilidad con respecto al contenido del elemento Value de el parámetro "VehDesc".

```
<?xml version="1.0"?>
<WfMessage xmlns="http://www.wfmc.org/standards/docs/Wf-XML" Version="1.1">
    <WfMessageHeader>
        …
    </WfMessageHeader>
    <WfMessageBody>
        …
        <ContextData>
            <Parameter>
                <Name>POID</Name>
                <Value>3878547</Value>
            </Parameter>
            <Parameter>
                <Name>OrderConf</Name>
                <Value>http://www.exampleco.com/orders/3878547</Value>
            </Parameter>
            <Parameter>
                <Name>VehDesc</Name>
                <Value>
                    <Vehicle>
                        <VehicleType>Car</VehicleType>
                        <Specification>
                            <Manufacturer>Audi</Manufacturer>
                            <Model>A4</Model>
                        </Specification>
                    </Vehicle>
                </Value>
            </Parameter>
        </ContextData>…
    </WfMessageBody>
</WfMessage>
```

Estados

Otro aspecto importante de un proceso es el estado del mismo en un momento dado del tiempo. En general un proceso puede estar activo o inactivo por un número de razones.
Los estados están organizados en varios niveles de granularidad. El nivel más alto de estado debe ser soportado, una implementación puede elegir omitir los estados opcionales y agregar algún estado adicional a los definidos.

```
<!ENTITY % states "open.notrunning | open.notrunning.suspended | open.running | closed.completed |
closed.abnormalCompleted | closed.abnormalCompleted.terminated | closed.abnormalCompleted.aborted">

<!ELEMENT open.notrunning EMPTY>

<!ELEMENT open.notrunning.suspended EMPTY>

<!ELEMENT open.running EMPTY>

<!ELEMENT closed.completed EMPTY>

<!ELEMENT closed.abnormalCompleted EMPTY>

<!ELEMENT closed.abnormalCompleted.terminated EMPTY>

<!ELEMENT closed.abnormalCompleted.aborted EMPTY>
```

Estas estructuras tienen las siguiente restricciones semántica y significados:

- **Abierto - no ejecutando (open.notrunning)** – Un recurso está en este estado cuando ha sido instanciado, pero no está actualmente participado en la activación de un proceso.

- **Abierto - no ejecutando - suspendido (open.notrunning.suspended)** – Un recurso está en este estado cuando ha iniciado su participación en una activación de un proceso, pero ha sido temporalmente suspendido. (opcional)

- **Abierto – ejecutando (open.running)** – Un recurso está en un estado donde ha realizado parte de su ejecución normal dentro de un proceso.

- **Cerrado – finalizado (closed.completed)** – Un recurso está en este estado cuando ha finalizado sus tareas en el proceso global. Todos los recursos contenidos se asumen que han finalizado.

- **Cerrado – anormalmente finalizado (closed.abnormalCompleted)** – Un recurso está en este estado cuando ha finalizado anormalmente. Son retornados los resultados de las tareas finalizadas.

- **Cerrado - anormalmente finalizado - Terminado closed.abnormalCompleted. terminated)** Un recurso está en este estado cuando ha sido finalizado por el recurso solicitante antes de finalizar su proceso de trabajo. En este punto todos los recursos contendidos en él se asumen que o han finalizado o terminado. (opcional)

- **Cerrado - anormalmente finalizado - Abortado (closed.abnormalCompleted. aborted)** – Un recurso está en este estado cuando su ejecución ha sido anormalmente finalizado antes que completara su trabajo en el proceso. No hay suposiciones sobre el estado de los recursos contenidos. (opcional)

Estos estados son también utilizados cuando se realiza procesamiento batch para indicar el estado general del mensaje batch. Este estado no debe ser confundido con el estado del proceso iniciado por estos mensajes, es el estado del procesamiento del mensaje, este estado refleja el progreso hecho del procesamiento de cada operación individual que comprende el mensaje batch. El mensaje batch no es considerado finalizado hasta que cada operación individual esta finalizada. En el contexto del los mensajes batch las estructuras tienen la siguiente restricciones semánticas y significados:

- **Abierto - no ejecutando (open.notrunning)** – Un mensaje está en este estado cuando ha sido recibido, pero está actualmente siendo procesado, por ejemplo está en algún tipo de cola interna esperando por ser procesado.

- **Abierto - no ejecutando - suspendido (open.notrunning.suspended)** – Un mensaje está en este estado cuando ha sido recibido y su procesamiento ha sido iniciado, pero no está actualmente siendo procesado. Este estado puede ser el resultado de un requerimiento explícito de cambio de estado o demora normal de procesamiento interno. (opcional)

- **Abierto – ejecutando (open.running)** – Un mensaje está en este estado cuando está siendo procesado. Algunas operaciones en el batch pueden haber finalizado, otras estar en ejecución y otras aún no iniciadas.

- **Cerrado – finalizado (closed.completed)** – Un mensaje está en este estado cuando ha finalizado su procesamiento. Todos las operaciones han sido procesadas exitosamente. Las respuestas a las operaciones de requerimiento del batch serán (o han sido) enviadas al solicitante.

- **Cerrado – anormalmente finalizado (closed.abnormalCompleted)** – Un mensaje está en este estado cuando su procesamiento ha sido finalizado anormalmente. Esto significa que una o más operaciones en el batch no han sido procesadas exitosamente, algunas operaciones lo pueden haberlo sido exitosas. Las respuestas a las operaciones de requerimiento del batch serán (o han sido) enviadas al solicitante.

- **Cerrado - anormalmente finalizado - Terminado closed.abnormalCompleted. terminated)** Un mensaje está en este estado cuando su procesamiento ha sido cancelado por el iniciador. Algunas de las operaciones del batch pueden haber sido finalizadas. Las respuestas de cualquiera de las operaciones requeridas en el batch que pueden haber sido finalizado son enviadas al solicitante. Las operaciones que no han finalizado cuando el mensaje ha finalizado son ignoradas. (opcional).

- **Cerrado - anormalmente finalizado - Abortado (closed.abnormalCompleted. aborted)** – Un mensaje está en este estado cuando su ejecución ha sido anormalmente finalizado como resultado de un error interno de procesamiento. Algunas de las operaciones del batch pueden haber finalizado. Las respuestas a algunos requerimiento de operaciones que hayan finalizado serán enviadas al solicitante. Las operaciones que no hayan sido finalizadas son ignoradas. (opcional)

Manejo de errores

Cuando ocurre una excepción durante la ejecución de una operación de Wf-XML, se debe retornar información relativa a la excepción al llamador. Varios tipos de excepción pueden ser anticipados, incluyendo tipos de errores temporales y fatales.
Un elemento llamado "Exception" ha sido definido para contener esta información.

```
<!ELEMENT Exception (MainCode, SubCode?, Type, Subject, Description?)>

<!ELEMENT MainCode (#PCDATA)>

<!ELEMENT SubCode (#PCDATA)>

<!ELEMENT Type (#PCDATA)>

<!ELEMENT Subject (#PCDATA)>

<!ELEMENT Description (#PCDATA)>
```

Este elemento de excepción debe ser retornado como el contendido del elemento <OperationName.Response>, en lugar de los datos de respuesta normal, para todas las operaciones en las cuales una excepción puede ocurrir.
Las estructuras tienen las siguientes restricciones semánticas y significados:

MainCode: Código principal, entero positivo de tres dígitos que definido en la especificación de la operación, e indica cual fue el problema. Los programas pueden usarlo para saber cuando ocurrió el error.

SubCode: Entero positivo de tres dígitos, el código desglosa el código principal. (Opcional)

Type: Es el tipo de error ocurrido, puede ser "F" error fatal o "T" error temporal.

Subject: Es una descripción en línea de la excepción.

Description: Son varias líneas de texto que describen la excepción, que detalla el subject.(opcional)

Ejemplo:

```
<?xml version="1.0"?>
<WfMessage xmlns="http://www.wfmc.org/standards/docs/Wf-XML" Version="1.1">
        <WfMessageHeader>
                <Response/>
                <Key>http://www.exampleco.com/processes/86947325</Key>
        </WfMessageHeader>
        <WfMessageBody>
                <CreateProcessInstance.Response>
                        <Exception>
                                <MainCode>502</MainCode>
                                <Type>F</Type>
                                <Subject>Invalid process definition</Subject>
                                <Description>Cannot create instance</Description>
                        </Exception>
                </CreateProcessInstance.Response>
        </WfMessageBody>
</WfMessage>
```

Códigos de excepción

WfMessageHeader	**100 Series**
Excepciones relativas a parámetros inválidos o faltantes en el cabezal	
WF_PARSING_ERROR	100
WF_ELEMENT_MISSING	101
WF_INVALID_VERSION	102
WF_INVALID_RESPONSE_REQUIRED_VALUE	103
WF_INVALID_KEY	104
WF_INVALID_OPERATION_SPECIFICATION	105
WF_INVALID_REQUEST_ID	106

Data	**200 Series**
Excepciones de contexto o de datos de resultados incorrectos.	
WF_INVALID_CONTEXT_DATA	201
WF_INVALID_RESULT_DATA	202
WF_INVALID_RESULT_DATA_SET	203

Authorization	**300 Series**
Excepciones causadas porque un usuario no ha sido autorizado para realizar la operación sobre un recurso particular, por ejemplo no puede crear una instancia de un proceso para una definición de proceso dada.	
WF_NO_AUTHORIZATION	300

Internal	**400 Series**
La operación no puede ser finalizada debido a algún error interno temporal en el motor de workflow.	
WF_OPERATION_FAILED	400

Resource Access	**500 Series**
Una clave inválida ha sido usada, sin embargo esta operación no puede actualmente ser invocada en el recurso específico.	
WF_NO_ACCESS_TO_RESOURCE	500
WF_INVALID_PROCESS_DEFINITION	502
WF_MISSING_PROCESS_INSTANCE_KEY	503
WF_INVALID_PROCESS_INSTANCE_KEY	504

Operation-specific	**600 Series**
Estas son las excepciones específicas de las operaciones.	
WF_INVALID_STATE_TRANSITION	600
WF_INVALID_OBSERVER_FOR_RESOURCE	601
WF_MISSING_NOTIFICATION_NAME	602
WF_INVALID_NOTIFICATION_NAME	603

Extensibility	**800 Series**
Especifica otras excepciones	
WF_OTHER	800

Las siguientes excepciones son generales a todas las operaciones, 100 Series (100 - 106), 300, 400, 500, 800. Otras excepciones relevantes a operaciones específicas están definidas junto con las operaciones en la siguiente sección.

10.1.5 Definición de operaciones

El alcance de esta especificación está limitado a las operaciones mostradas en la tabla.
Se definirán las colecciones de operaciones utilizadas por los grupos de Control, ProcessDefinition, ProcessInstance y Observer.
Para claridad los ejemplos serán para el procesamiento individual sincrónico.

	Control	Process Definition	Process Instance	Observer
GetBatchMessageState	**X**			
ChangeBatchMessageState	**X**			
CreateProcessInstance		**X**		
GetProcessInstanceData			**X**	
ChangeProcessInstanceState			**X**	
ProcessInstanceStateChanged				**X**
Notify				**X**

La lista de los elementos de las operaciones válidos son definidos por las siguientes dos entidades, una para las operaciones Request y otro para las operaciones Response.

```
<!ENTITY % OperationRequest "(GetBatchMessageState.Request | ChangeBatchMessageState.Request |
CreateProcessInstance.Request | GetProcessInstanceData.Request | ChangeProcessInstanceState.Request |
ProcessInstanceStateChanged.Request | Notify.Request)">

<!ENTITY % OperationResponse "(GetBatchMessageState.Response | ChangeBatchMessageState.Response |
CreateProcessInstance.Response | GetProcessInstanceData.Response | ChangeProcessInstanceState.Response |
ProcessInstanceStateChanged.Response | Notify.Response)">
```

Operaciones de control

Este grupo de operaciones es usado para realizar interacciones administrativas entre servicios interoperantes. Estas interacciones son típicamente desligadas de procesos específicos.
Contienen las operaciones de GetBatchMessageState y ChangeBatchMessageState.

GetBatchMessageState
Se utiliza para devolver el estado de un mensaje batch previamente enviado a un recurso, utiliza los estados para mensajes batch ya definidos.

```
<!ELEMENT GetBatchMessageState.Request (MessageID)>
<!ELEMENT MessageID (#PCDATA)>
<!ELEMENT GetBatchMessageState.Response (State | Exception)>
<!ELEMENT State (%states;)?>
```

Parámetros para la solicitud:

MessageID: Identificador único del mensaje batch cuyo estado es recuperado. Los datos deben ser del tipo UUID y debe coincidir con el MessageID del mensaje batch previamente enviado al recurso que recibió la solicitud.

Ejemplo:

```
<?xml version="1.0"?>
<WfMessage xmlns="http://www.wfmc.org/standards/docs/Wf-XML" Version="1.1">
        <WfMessageHeader>
                <Request ResponseRequired="Yes"/>
                <Key>http://www.exampleco.com/processes</Key>
        </WfMessageHeader>
        <WfMessageBody>
                <GetBatchMessageState.Request>
                        <MessageID>e85327bc-dc9e-2878-4b52-947852107643</MessageID>
                </GetBatchMessageState.Request>
        </WfMessageBody>
</WfMessage>
```

Parámetros para la respuesta.

State: Estado actual del mensaje batch.

Excepciones:

Se aplican las excepciones generales.

Ejemplo:

```
<?xml version="1.0"?>
<WfMessage xmlns="http://www.wfmc.org/standards/docs/Wf-XML" Version="1.1">
        <WfMessageHeader>
                <Response/>
                <Key>http://www.exampleco.com/processes</Key>
        </WfMessageHeader>
        <WfMessageBody>
                <GetBatchMessageState.Response>
                        <State>
                                <open.running/>
                        </State>
                </GetBatchMessageState.Response>
        </WfMessageBody>
</WfMessage>
```

ChangeBatchMessageState
Es utilizado para cambiar de estado un mensaje batch previamente enviado a un recurso dado.

```
<!ELEMENT ChangeBatchMessageState.Request (MessageID, State)>
<!ELEMENT MessageID (#PCDATA)>
<!ELEMENT State (%states;)?>
<!ELEMENT ChangeBatchMessageState.Response (State | Exception)>
```

Parámetros para la solicitud:

MessageID: Identificador único del mensaje batch cuyo estado quiere ser cambiado. Los datos deben ser del tipo UUID y debe coincidir con el MessageID del mensaje batch previamente enviado al recurso que recibió la solicitud.

State: El nuevo estado del mensaje batch.

Ejemplo:

```
<?xml version="1.0"?>
<WfMessage xmlns="http://www.wfmc.org/standards/docs/Wf-XML" Version="1.1">
    <WfMessageHeader>
        <Request ResponseRequired="Yes"/>
        <Key>http://www.exampleco.com/processes</Key>
    </WfMessageHeader>
    <WfMessageBody>
        <ChangeBatchMessageState.Request>
            <MessageID>e85327bc-dc9e-2878-4b52-947852107643</MessageID>
            <State>
                <closed.abnormalCompleted.terminated/>
            </State>
        </ChangeBatchMessageState.Request>
    </WfMessageBody>
</WfMessage>
```

Parámetros para la respuesta.

State: El nuevo estado al que ha sido cambiado el mensaje batch. Si el requerimiento ha sido procesado exitosamente, el contenido de este elemento debe ser igual al de la solicitud.

Excepciones:
Se aplican las excepciones generales, más la siguiente: WF_INVALID_STATE_TRANSITION

Ejemplo:

```
<?xml version="1.0"?>
<WfMessage xmlns="http://www.wfmc.org/standards/docs/Wf-XML" Version="1.1">
    <WfMessageHeader>
        <Response/>
        <Key>http://www.exampleco.com/processes</Key>
    </WfMessageHeader>
    <WfMessageBody>
        <ChangeBatchMessageState.Response>
            <State>
                <closed.abnormalCompleted.terminated/>
            </State>
        </ChangeBatchMessageState.Response>
    </WfMessageBody>
</WfMessage>
```

Operaciones de Definición de proceso

Este grupo de operaciones es usado para realizar acciones sobre la definición de proceso, como crear instancias de procesos basadas en estas definiciones. El conjunto de definiciones de procesos soportadas por un sistema de activación debe ser predefinido. Actualmente este grupo solo contienen la operación CreateProcessInstance.

CreateProcessInstance
Es utilizada para instanciar una definición de proceso conocida. La instancia debe ser creada con los datos de contexto específico de acuerdo a los datos de entrada y automáticamente iniciado.

```
<!ELEMENT CreateProcessInstance.Request (ObserverKey?, Name?, Subject?, Description?, ContextData)>
<!ATTLIST CreateProcessInstance.Request StartImmediately (true | false) #FIXED "true">
<!ELEMENT ObserverKey (#PCDATA)>
<!ELEMENT Name (#PCDATA)>
<!ELEMENT Subject (#PCDATA)>
<!ELEMENT Description (#PCDATA)>
<!ELEMENT ContextData ANY>
<!ELEMENT CreateProcessInstance.Response ((ProcessInstanceKey, Name?) | Exception)>
<!ELEMENT ProcessInstanceKey (#PCDATA)>
```

Parámetros para la solicitud:

StartImmediately: Un booleano que indica cuando una nueva instancia creada debe ser inmediatamente iniciada después de su creación. El valor es actualmente siempre "true".

ObserverKey: URI del recurso que actúa como observer de la instancia que se creada por la operación. El recurso debe ser el que solicita la operación. Este observer debe ser notificado de los eventos que impactan la ejecución de la instancia de proceso, como el cambio de estado, y la finalización de la instancia. Con el elemento observerKey seteado, queda implícito que el modelo de interoperabilidad es de subproceso anidado o paralelamente sincronizado, de otra forma el modelo es un proceso encadenado. (opcional)

Name: Un nombre que se asigna a la nueva instancia del proceso. Si este nombre no es único, debe ser modificado para serlo, o cambiado. El uso de este nombre no puede ser garantido. Si no se usa, el nombre asignado debe ser retornado en el mensaje CreateProcessInstance.Response para informar al iniciador del nuevo nombre (opcional)

Subject: Una descripción corta del propósito de la nueva instancia del proceso (opcional)

Description: Una descripción larga del propósito de la nueva instancia del proceso (opcional)

ContextData: Datos del contexto específico requeridos para crear la instancia del proceso. Esta información debe estar codificada según lo acordado en el contrato de interoperabilidad.

Ejemplo:

```
<?xml version="1.0"?>
<WfMessage xmlns="http://www.wfmc.org/standards/docs/Wf-XML" Version="1.1">
    <WfMessageHeader>
        <Request ResponseRequired="Yes"/>
        <Key>http://www.exampleco.com/processes/86947325</Key>
    </WfMessageHeader>
    <WfMessageBody>
        <CreateProcessInstance.Request StartImmediately="true">
            <ObserverKey>http://www.myco.com/purchasing/orders/4089259</Obse
            rverKey>
            <ContextData>
                <Parameter>
                    <Name>VehDesc</Name>
                    <Value>
                        <Vehicle>
                            <VehicleType>Car</VehicleType>
                            <Specification>
                                <Manufacturer>Audi</Manufacturer>
                                <Model>A4</Model>
                            </Specification>
                        </Vehicle>
                    </Value>
                </Parameter>
                <Parameter>
                    <Name>Customer</Name>
                    <Value>John Doe</Value>
                </Parameter>
            </ContextData>
        </CreateProcessInstance.Request>
    </WfMessageBody>
</WfMessage>
```

Parámetros para la respuesta.

ProcessInstanceKey: URI del la instancia de proceso creada

Name: El nombre actual asignado a la instancia de proceso creada por el recurso activante. (opcional)

Excepciones:

Se aplican las excepciones generales, más la siguiente:

WF_MISSING_PROCESS_INSTANCE_KEY
WF_INVALID_PROCESS_INSTANCE_KEY
WF_INVALID_PROCESS_DEFINITION
WF_INVALID_OBSERVER_FOR_RESOURCE

Ejemplo:

```
<?xml version="1.0"?>
<WfMessage xmlns="http://www.wfmc.org/standards/docs/Wf-XML" Version="1.1">
    <WfMessageHeader>
        <Response/>
        <Key>http://www.exampleco.com/processes/86947325</Key>
    </WfMessageHeader>
    <WfMessageBody>
```

```
            <CreateProcessInstance.Response>
                <ProcessInstanceKey>http://www.exampleco.com/orders/86947325-
                32914
                </ProcessInstanceKey>
            </CreateProcessInstance.Response>
        </WfMessageBody>
    </WfMessage>
```

Operaciones de Instanciación de proceso

Este grupo de operaciones es usado para comunicarse con una instancia particular de una definición del proceso (o la activación de un servicio), adquiriendo información sobre la instancia y controlándola. Cómo una instancia dada debe continuar ejecutando por una cantidad de tiempo, las operaciones deben ser llamadas sobre una instancia mientras ésta se ejecuta. Estas operaciones deben obtener información sobre el estado u obtener resultados primarios (aunque los resultados de una instancia de proceso no son finales hasta que la instancia haya sido completada). Este grupo contiene las operaciones de GetProcessInstanceData y ChangeProcessInstanceState.

GetProcessInstanceData
Es utilizada para devolver los valores de las propiedades definidas por una instancia de proceso.

```
<!ENTITY % ProcessInstanceData “Name | Subject | Description | State | ValidStates | ObserverKey |
ResultData | ProcessDefinitionKey | Priority | LastModified”>

<!ENTITY % states “open.notrunning | open.notrunning.suspended | open.running | closed.completed |
closed.abnormalCompleted.terminated | closed.abnormalCompleted.aborted”>

<!ELEMENT GetProcessInstanceData.Request (ResultDataSet?)>
<!ELEMENT ResultDataSet (%ProcessInstanceData;)+>
<!ELEMENT GetProcessInstanceData.Response ((%ProcessInstanceData;)+ | Exception)>
<!ELEMENT Name (#PCDATA)>
<!ELEMENT Subject (#PCDATA)>
<!ELEMENT Description (#PCDATA)>
<!ELEMENT State (%states;)?>
<!ELEMENT ValidStates (%states;)*>
<!ELEMENT ObserverKey (#PCDATA)>
<!ELEMENT ProcessDefinitionKey (#PCDATA)>
<!ELEMENT Priority (#PCDATA)>
<!ELEMENT LastModified (#PCDATA)>
```

Parámetros para la solicitud:

ResultDataSet: Este parámetro contienen un conjunto de propiedades a ser retornados, donde este conjunto pueden ser todas las propiedades o algún subconjunto. (opcional)

El siguiente ejemplo solicita todas las propiedades de una ProcessInstance en particular:

```
<?xml version="1.0"?>
<WfMessage xmlns=”http://www.wfmc.org/standards/docs/Wf-XML” Version=”1.1”>
    <WfMessageHeader>
        <Request ResponseRequired=”Yes”/>
        <Key>http://www.exampleco.com/orders/86947325-32914</Key>
    </WfMessageHeader>
    <WfMessageBody>
        <GetProcessInstanceData.Request/>
    </WfMessageBody>
</WfMessage>
```

El siguiente ejemplo solicita solo las propiedades Name y Priority de una ProcessInstance en particular:

```
<?xml version="1.0"?>
<WfMessage xmlns="http://www.wfmc.org/standards/docs/Wf-XML" Version="1.1">
        <WfMessageHeader>
                <Request ResponseRequired="Yes"/>
                <Key>http://www.exampleco.com/orders/86947325-32914</Key>
        </WfMessageHeader>
        <WfMessageBody>
                <GetProcessInstanceData.Request>
                        <ResultDataSet>
                        <Name/>
                        <Priority/>
                        </ResultDataSet>
                </GetProcessInstanceData.Request>
        </WfMessageBody>
</WfMessage>
```

Parámetros para la respuesta.

Name: Un identificador del recurso. (Opcional)

Subject: Una descripción corta de la instancia del proceso. (opcional)

Descripción: Una descripción larga del recurso de la instancia del proceso. (opcional)

State: el estado actual del recurso (opcional)

ValidStates: Una lista de estados válidos permitidas por este recurso. Es la lista de estados por los cuales la instancia actual puede realizar transacciones (opcional).

ProcessDefinitionKey: URI del recurso de definición de proceso por el que la instancia fue creada (opcional).

ObserverKey: URI del observador registrado para esta instancia del proceso, si es que existe (opcional)

ResultData: Datos específicos del contexto (especificados en el contrato de interoperabilidad) que representan los valores actuales del resultado de la ejecución del proceso. Si el resultado aún no está disponible el elemento aparece vacío (opcional)

Priority: Un indicador de la importancia relativa de la instancia del proceso. Este valor será un entero de 1 a 5, 1 es la prioridad mayor. El valor por defecto es 3. (opcional)

LastModified: El día de la última modificación de la instancia, si es que está disponible (opcional).

Excepciones:

Se aplican las excepciones generales, más las siguientes:

WF_INVALID_RESULT_DATA
WF_INVALID_RESULT_DATA_SET
WF_INVALID_OBSERVER_FOR_RESOURCE

El siguiente ejemplo de respuesta para la operación GetProcessInstanceData:

```
<?xml version="1.0"?>
<WfMessage xmlns="http://www.wfmc.org/standards/docs/Wf-XML" Version="1.1">
        <WfMessageHeader>
                <Response/>
```

```
            <Key>http://www.exampleco.com/orders/86947325-32914</Key>
        </WfMessageHeader>
        <WfMessageBody>
            <GetProcessInstanceData.Response>
                <Name>Order32914</Name>
                <Priority>3</Priority>
            </GetProcessInstanceData.Response>
        </WfMessageBody>
</WfMessage>
```

ChangeProcessInstanceState

Es utilizada para modificar el estado de la instancia del proceso, por ejemplo de open.running a open.notrunning.suspended.

```
<!ELEMENT ChangeProcessInstanceState.Request (State)>
<!ELEMENT ChangeProcessInstanceState.Response (State | Exception)>
<!ELEMENT State (%states;)?>
```

Parámetros para la solicitud:

State: El nuevo estado de la instancia del proceso.

Ejemplo:

```
<?xml version="1.0"?>
<WfMessage xmlns="http://www.wfmc.org/standards/docs/Wf-XML" Version="1.1">
        <WfMessageHeader>
            <Request ResponseRequired="Yes"/>
            <Key>http://www.exampleco.com/orders/86947325-32914</Key>
        </WfMessageHeader>
        <WfMessageBody>
            <ChangeProcessInstanceState.Request>
                <State>
                    <open.notrunning.suspended/>
                </State>
            </ChangeProcessInstanceState.Request>
        </WfMessageBody>
</WfMessage>
```

Parámetros para la respuesta:

State: El nuevo estado resultado de la operación.

Excepciones:

Se aplican las excepciones generales, más las siguientes:

WF_INVALID_STATE_TRANSITION
WF_INVALID_OBSERVER_FOR_RESOURCE

Ejemplo:

```
<?xml version="1.0"?>
<WfMessage xmlns="http://www.wfmc.org/standards/docs/Wf-XML" Version="1.1">
        <WfMessageHeader>
                <Response/>
                <Key>http://www.exampleco.com/orders/86947325-32914</Key>
        </WfMessageHeader>
        <WfMessageBody>
                <ChangeProcessInstanceState.Response>
                        <State>
                                <open.notrunning.suspended/>
                        </State>
                </ChangeProcessInstanceState.Response>
        </WfMessageBody>
</WfMessage>
```

Operaciones de observador

Este grupo de operaciones permite al solicitante de trabajo (el observador de una instancia de proceso) ser notificado de los eventos y de los cambios de estado que impactan la ejecución de una instancia de proceso. Este grupo contiene las operaciones de ProcessInstanceStateChanged y Notify.

ProcessInstanceStateChanged

Es utilizada para soportar los cambios de estados a closed.completed y closed.abnormalCompleted. El atributo ResponseRequired será típicamente seteado a false para esta operación que es solamente utilizado para notificación a un observer que un evento de cambio de estado a ocurrido.

```
<!ELEMENT ProcessInstanceStateChanged.Request (ProcessInstanceKey, State, ResultData?,
LastModified?)>
<!ELEMENT ProcessInstanceKey (#PCDATA)>
<!ELEMENT State (%states;)?>
<!ELEMENT ResultData ANY>
<!ELEMENT LastModified (#PCDATA)>
<!ELEMENT ProcessInstanceStateChanged.Response (Exception?)>
```

Parámetros para la solicitud:

ProcessInstanceKey: URI del recurso instancia del proceso que ha cambiado

State: El nuevo estado del recurso.

ResultData: Datos específicos del contexto que representan el valor actual del resultado. Si el resultado no está disponible, el elemento es retornado vacío. (opcional)

LastModified: El día de la última modificación de la instancia. (opcional).

Ejemplo:

```
<?xml version="1.0"?>
<WfMessage xmlns="http://www.wfmc.org/standards/docs/Wf-XML" Version="1.1">
    <WfMessageHeader>
        <Request ResponseRequired="No"/>
        <Key>http://www.myco.com/purchasing/orders/4089259</Key>
    </WfMessageHeader>
    <WfMessageBody>
        <ProcessInstanceStateChanged.Request>
            <ProcessInstanceKey>http://www.exampleco.com/orders/86947325-
            32914
            </ProcessInstanceKey>
            <State>
                <closed.completed/>
            </State>
            <ResultData>
                <Parameter>
                    <Name>VehDesc</Name>
                    <Value>
                        <Vehicle>
                            <VehicleType>Car</VehicleType>
                            <Specification>
                                <Manufacturer>Audi</Manufacturer>
                                <Model>A4</Model>
                            </Specification>
                        </Vehicle>
                    </Value>
                </Parameter>
                <Parameter>
                    <Name>Customer</Name>
                    <Value>John Doe</Value>
                </Parameter>
            </ResultData>
        </ProcessInstanceStateChanged.Request>
    </WfMessageBody>
</WfMessage>
```

Parámetros para la respuesta:

Ninguno.

Excepciones:

Se aplican las excepciones generales, más las siguientes:

WF_INVALID_RESULT_DATA
WF_MISSING_PROCESS_INSTANCE_KEY
WF_INVALID_PROCESS_INSTANCE_KEY
WF_INVALID_STATE_TRANSITION
WF_INVALID_OBSERVER_FOR_RESOURCE

Si los atributos de ResponseRequired son seteados a "true" en el requerimiento de ProcessInstanceStateChanged una mínima respuesta es retornada. Esto puede ser usado para atrapar algún error que puede ocurrir durante la notificación del cambio del estado.

Ejemplo:

```
<?xml version="1.0"?>
<WfMessage xmlns="http://www.wfmc.org/standards/docs/Wf-XML" Version="1.1">
        <WfMessageHeader>
                <Response/>
                <Key>http://www.myco.com/purchasing/orders/4089259</Key>
        </WfMessageHeader>
        <WfMessageBody>
                <ProcessInstanceStateChanged.Response/>
        </WfMessageBody>
</WfMessage>
```

Notify

Esta operación provee del significado por el que dos instancias de proceso pueden comunicarse y sincronizar sus ejecuciones. Es usado para notificar un observer (que puede ser otra instancia de proceso) de la ocurrencia de un evento en una instancia de proceso que es relevante para futuras operaciones del observer. Esta operación no debe ser usada para informar a un recurso del cambio de estado en una instancia del proceso, la operación ProcessInstanceStateChanged debe usarse para este propósito. Esta operación es usada para notificar a un recurso de un evento específico a una aplicación.
La naturaleza de estos eventos y detalles deben ser acordados en el contrato de interoperabilidad para usar esta operación.
Típicamente, estas notificaciones tienen que ver con los cambios en los datos de la aplicación que pueden impactar la interoperación entre procesos. Luego esta operación debe indicar (con la ContextData) información relativa a los ítems de datos afectados además de la información del evento en si mismo.

```
<!ELEMENT Notify.Request (ProcessInstanceKey, NotificationName, ContextData)>
<!ELEMENT ProcessInstanceKey (#PCDATA)>
<!ELEMENT NotificationName (#PCDATA)>
<!ELEMENT ContextData ANY>
<!ELEMENT Notify.Response (Exception?)>
```

Parámetros para la solicitud:

ProcessInstanceKey: URI de la instancia del proceso que invoca la operación.

NotificationName: Nombre del mensaje de notificación. El contenido de este elemento está sujeto a los acuerdos realizados en el contrato de interoperabilidad, como los eventos son específicos a una aplicación particular y/o requerimiento de instancia de proceso.

ContextData: Datos específicos del contexto que representan los datos de la aplicación a ser enviados al observer.

Ejemplo:

```
<?xml version="1.0"?>
<WfMessage xmlns="http://www.wfmc.org/standards/docs/Wf-XML" Version="1.1">
    <WfMessageHeader>
        <Request ResponseRequired="No"/>
        <Key>http://www.myco.com/purchasing/orders/4089259</Key>
    </WfMessageHeader>
    <WfMessageBody>
        <Notify.Request>
            <ProcessInstanceKey>http://www.exampleco.com/orders/86947325-
            32914
            </ProcessInstanceKey>
            <NotificationName>OrderChanged</NotificationName>
            <ContextData>
                <Parameter>
                    <Name>VehDesc</Name>
                    <Value>
                        <Vehicle>
                            <VehicleType>Car</VehicleType>
                            <Specification>
                                <Manufacturer>Audi</Manufacturer>
                                <Model>A4</Model>
                            </Specification>
                        </Vehicle>
                    </Value>
                </Parameter>
            </ContextData>
        </Notify.Request>
    </WfMessageBody>
</WfMessage>
```

Parámetros para la respuesta:
Ninguno.

Excepciones:

Se aplican las excepciones generales, más las siguientes:

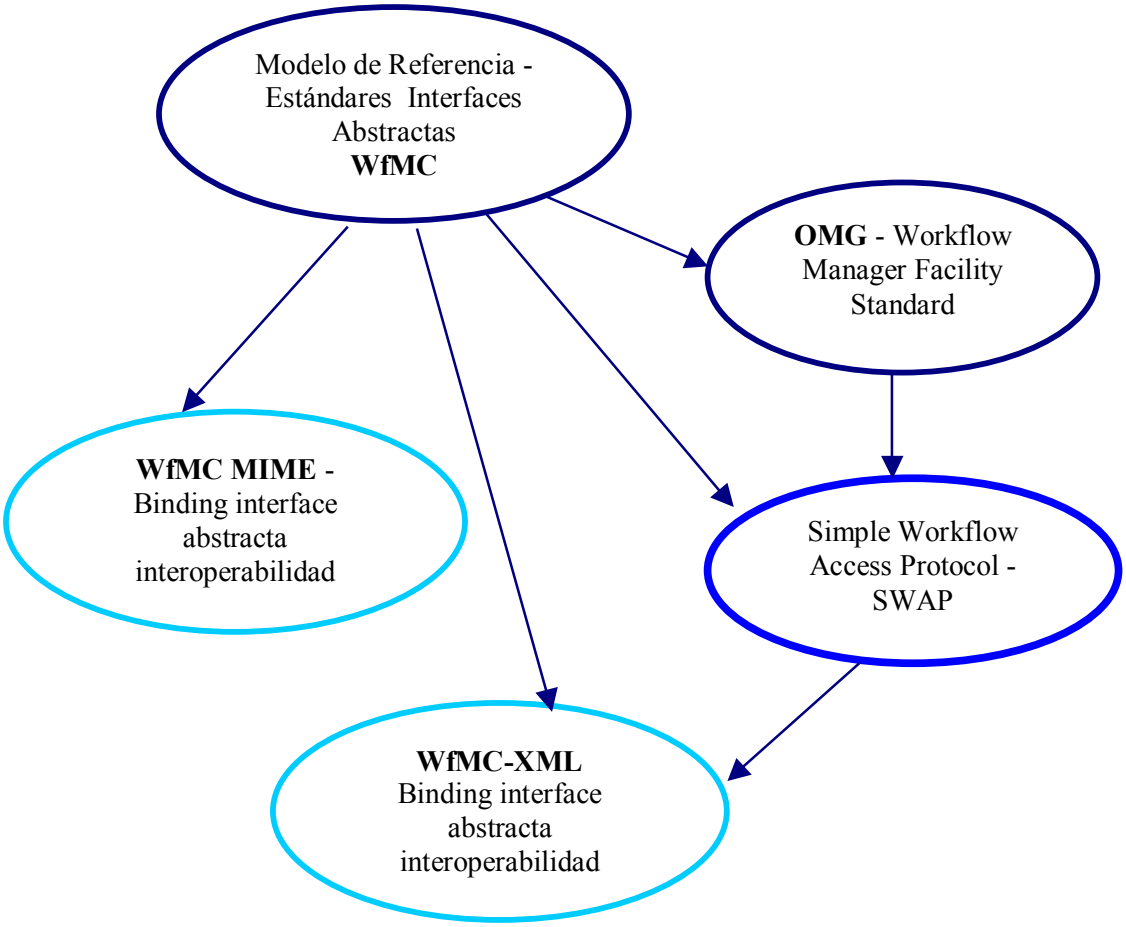
WF_INVALID_CONTEXT_DATA
WF_MISSING_PROCESS_INSTANCE_KEY
WF_INVALID_PROCESS_INSTANCE_KEY
WF_INVALID_OBSERVER_FOR_RESOURCE
WF_MISSING_ NOTIFICATION_NAME
WF_INVALID_ NOTIFICATION_NAME

Ejemplo:

```
<?xml version="1.0"?>
<WfMessage xmlns="http://www.wfmc.org/standards/docs/Wf-XML" Version="1.1">
    <WfMessageHeader>
        <Response/>
        <Key>http://www.myco.com/purchasing/orders/4089259</Key>
    </WfMessageHeader>
    <WfMessageBody>
        <Notify.Response/>
    </WfMessageBody>
</WfMessage>
```

10.2 Relaciones con otros estándares

10.2.1 Esquema de la evolución y relación de los estándares de workflow y sus bindings.



10.2.2 OMG Workflow Management Facility Standard (jointFlow)

Se describirá el mapeo de las interfaces definidas por el estándar del Object Management Group (OMG) Workflow Management Facility y los recursos y operaciones Wf-XML.
El estándar Wf-XML usa el modelo de objetos básico definido en la especificación de la OMG. Soporta un subconjunto de las entidades definidas en este modelo de objetos y combina operaciones que están separadas en las interfaces OMG Workflow Management Facility en una única operación, consecuentemente mejora la performance no requiriendo estás operaciones tan finamente granulares.

El tipo de recurso **ProcessDefinition** de la Wf-XML se corresponde con la interface **WfProcessMgr** de la OMG Workflow Management Facility. La operación **CreateProcessInstance** combina las operaciones de **create_process** sobre **WfProcessMgr**, la operación **start** y la operación **set_context** de **WfProcess**.

El tipo de recurso **ProcesssInstance** se corresponde a la interface **WfProcess** de la OMG Workflow Management Facility.
La operación **ChangeProcessInstanceState** se corresponde a la operación **change_state** de **WfProcess** (heredada del **WfExecutionElement**).
La operación **GetProcessInstanceData** de la Wf-XML se corresponde con la operación **get_result** de **WfProcess** en combinación con las funciones para variables de estado de la **WfProcess**.

El tipo de recurso **Observer** de la Wf-XML se corresponde con la interfaz **WfRequester** del OMG Workflow Management Facility.

La especificación de la OMG Workflow Management Facility define algunas interfaces que no son representadas por la Wf-XML hasta este momento: **WfActivity**, **WfResource**, **WfAssignment**.

10.3 Características de implementación

10.3.1 Contrato de interoperabilidad

Un contrato define claramente cada expectativa y requerimiento de cada involucrado en todas las áreas que puedan impedir la interoperabilidad.
Algunos tópicos que deberían incluir son enumerados a continuación:

- Requerimientos de datos: Datos específicos de las aplicaciones ha ser transferidos para utilizar funcionalidades básicas o extendidas.
- Restricciones sobre datos: Requerimientos de tipos de datos específicos de las aplicaciones, largo de campos, conjunto de caracteres codificados, tamaño total del mensaje, etc.
- Manejo de errores: Requerimientos específicos manejo de errores específicos de las aplicaciones como subcódigos, descripciones, acciones requeridas, etc.
- Especificaciones de protocolo de transporte: cabezal de datos requerido por el protocolo, valores de timeouts, tamaño de búferes, requerimientos de procesamiento batch o asincrónico.
- Consideraciones de seguridad: Métodos de encriptado, verificación de usuarios, requerimientos de configuración de firewalls, etc.
- Requerimientos de claves/ID: Detallar lo relativo a la administración de claves, formatos de identificadores (de objetos) específicos de una implementación, etc.
- Sincronización de procesos: Especificar lo relativo a los eventos a los cuales un proceso debe ser notificado para lograr sincronizarse.

10.4 Conformidad con el estándar

El estándar determina como se establecen la conformidad de un producto pueda tener con el mismo.
Hay cuatro grandes niveles de categorías para características de la especificación::

- Modelos de interoperabilidad: Encadenados, Anidados y Paralelamente sincronizados.
- Tipos de procesamientos de mensajes: Sincrónico, asincrónico, individual y batch.
- Constructores del protocolo: Header/Body, Transport/Header/Body, solo Transport, transport & Multiples Header/Body.
- Operaciones: GetBatchMessageState, ChangeBatchMessageState, CreateProcessInstance, GetProcessInstanceData, ChangeProcessInstanceState, ProcessInstanceStateChanged y Notify.

Estas características están clasificadas en siete perfiles de conformidad.

Los productos que satisfagan la especificación deben informar el perfil satisfecho, los mecanismos de transporte soportados, (para esta versión solo se soporta el mecanismo de transporte HTTP). Un ejemplo puede ser que un proveedor diga que satisface el perfil de interoperabilidad y asincronismo sobre http.

10.4.1 Perfiles de Conformidad:

Perfil Interoperabilidad (obligatorio):

Debe soportar los modelos de interoperabilidad encadenados y anidados. Debe soportar la mensajería sincrónica e individual, los constructores básicos Header/Body, y puede soportar el constructor Transport/Header/Body opcionalmente. Además debe soportar las siguientes operaciones: CreateProcessInstance, GetProcessInstanceData, ChangeProcessInstanceState, ProcessInstanceStateChanged y Notify.

Perfil Asincrónico (opcional):

Debe soportar el tipo de mensaje asincrónico y debe soportar los constructores Transport/Header/Body y solo Transport.

Perfil Batch (opcional):

Debe soportar el tipo de mensaje batch y debe soportar los constructores Transport & Multiples Header/Body. Debe soportar las operaciones de GetBatchMessageState y ChangeBatchMessageState.

10.4.2 Validado vs. Bien formado

Todos las instancias del documentos XML (en particular los mensajes Wf-XML) deben poseer uno de los estados de validez. Pueden ser "inválidos" por errores sintácticos en sus marcas, pueden ser "bien formados" (well formed) que significa que son sintácticamente correctos con respecto a la especificación de XML. Pueden ser "válidos" que no solo son válidos sintácticamente pero además son compatibles con un DTD o una descripción de un esquema XML (XSD).
La especificación solo pide que el documento Wf-XML sean bien formados. Es responsabilidad de la aplicación de la implementación de la especificación para asegurar que las restricciones semánticas de la misma no sean violadas.

De todas maneras, hay un indicador de integridad de datos provisto por el validar la instancia del documento vía un parser XML. Se puede basar en el DTD provisto por la especificación para este propósito. Aunque se debe recordar que hay ciertas restricciones semánticas de los datos que no se pueden expresar mediante DTD, éstas deben ser tratadas por la aplicación.

10.4.3 Conformidad vs. Extensibilidad

Otro factor que puede impactar en la conformidad es la extensibilidad. Un vendedor de un producto puede realizar acuerdos de funcionalidades y formatos de mensajes fuera de la especificación o utilizar marcas indefinidas que serán ignoradas por el par en la interoperabilidad, deben poder hacerlo pero de forma de mantener la conformidad con el estándar.

Se recomienda la utilización de namespaces para facilitar el intercambio de datos específicos de la aplicación dentro de los mensajes wf-xml. Una implementación puede utilizarlos para diferencias los datos relacionados con el proceso de los datos de la aplicación, tanto como de los datos del protocolo Wf-xml. La utilización de namespaces adecuados que califiquen los datos específicos del contexto los protegerán de cambios en los datos del protocolo Wf-xml de las nuevas versiones.

10.5 Binding para la capa de transporte

Los mensajes Wf-XML pueden ser comunicados entre los sistemas de workflow interoperantes usando diferentes mecanismos o protocolos de transporte. Estos protocolos deben soportan el requerimiento fundamental de lograr interoperabilidad basada en el intercambio de mensajes.
Los comportamientos y acciones especificadas por el Wf-XML debe ser soportado por estos protocolos subyacentes. Se definirán las relaciones e interacciones entre el Wf-XML y su mecanismo de transporte subyacente.

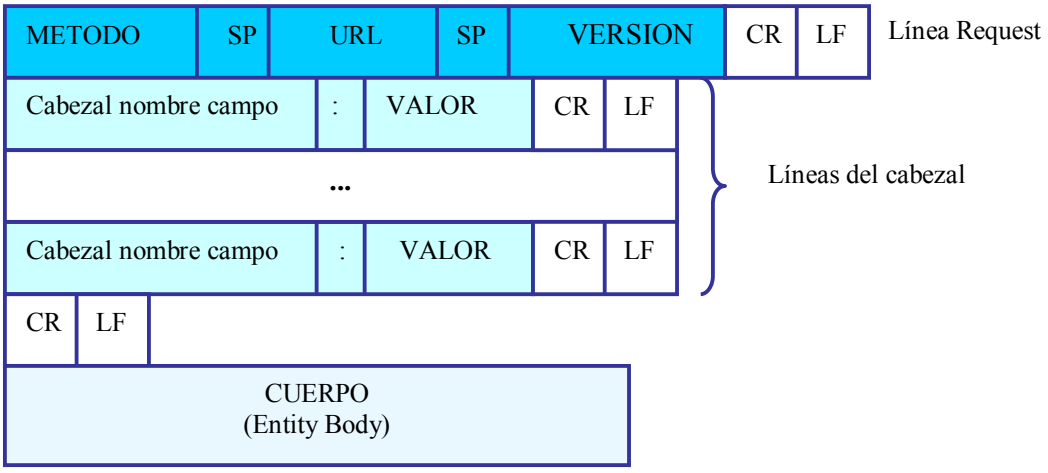
Se dará una especificación para el binding para Hypertext Transfer Protocol (HTTP), dado que se considera el mecanismo de transporte más comúnmente utilizado para comunicar mensajes Wf-XML entre servicios de activación interoperantes.
El soporte de este u otro binding de capa de transporte no es un requerimiento para que una implementación posea conformidad con la especificación Wf-XML.

A continuación se recordarán muy brevemente conceptos básicos del protocolo http, luego se describirá el binding.

10.5.1 HTTP - HyperText Transfer Protocol

Es un protocolo de capa de aplicación, que se ejecuta sobre TCP/IP, es el usado por la World Wide Web.
Básicamente el protocolo http se implementa como dos programas un cliente y un servidor que ejecutan en distintos sistemas intercambiando mensajes http.
Http define un formato de mensajes, como son transmitidos y que acciones toman los servidores y clientes Web en respuesta de varios comandos.
Existen dos tipo de mensajes http: Request y Response. (de requerimiento y respuesta).
Se describirá brevemente la estructura genérica de estos tipos de mensaje.

El siguiente es el formato de un mensajes http Request:



La línea de Request tiene los siguientes campos:

Métodos:

- GET: se utiliza cuando se requiere que el servidor retorne un objeto, que se identifica por el campo URL.
- POST: para enviar algo al servidor, por ejemplo un formulario.
- HEAD. Es similar al método POST pero cuando el servidor recibe un mensaje HEAD responde con un mensaje HTTP sin enviar el objeto requerido.

URL
Versión HTTP

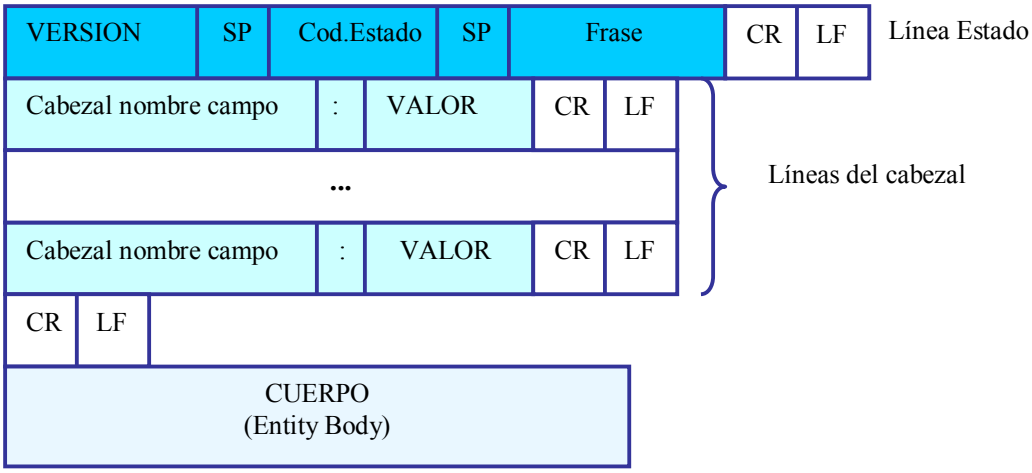
Las líneas de cabezal: Existen varios tipos de líneas de cabezal que incluyen conjunto de seteos, por ejemplo, Connection: Close significa que no requiere conexiones persistentes, Accept notifica al servidor de que tipo de objetos puede aceptar etc.

Cuerpo: Cuerpo del mensaje
En el caso de un método GET no se utiliza un cuerpo, se utiliza en el método POST

Un ejemplo de un mensaje Request que pide página Pagina.html a un servidor http:

GET //algunadirección/pagina.html HTTP/1.1
Connection: Close
User-agent: Mozila/4.0 //el tipo de browser que envía el requerimiento.
Accept: text/html, image/gif, image/jpeg //tipos de objetos que acepta
Accept-language:sp //prefiere si es que existe la versión español del objeto requerido
…

El siguiente es el formato de un mensajes http Response:



La línea de Estado contiene:

Versión HTTP.
Código de estado:
Frase: Mensaje del estado

Líneas de cabezal: similares al mensaje Response, por ejemplo la Content-type indica que tipo de objeto es el contenido en el cuerpo del mensaje.

Cuerpo:
Contiene el objeto devuelto por el mensaje.

Un ejemplo de un mensaje Response:

HTTP/1.1 200 OK //Código de éxito, requerimiento satisfecho, envío información requerida
Connection: Close //Servidor cierra conexión
Date: Thu, 06 Aug 2002 12:00:15 GMT //Día hora de la respuesta
Server: Apache/1.3.0 (Unix) //Tipo servidor que generó la respuesta
Last-Modified: Mon, 22 Jun 2002 09:23:24 GMT //Día y hora modificación objeto enviado
Content-Length: 6849 //número de bytes de tamaño del objeto enviado como respuesta
Content-Type: text/html //tipo del objeto
Datos, datos, datos… // objeto enviado como respuesta.

10.5.2 HTTP binding para Wf-xml

Para http los servicios de activación comunicantes son considerados servidores http (los servicios deben comunicarse directamente vía http, o deben ser combinados con otros programas que los habiliten a enviar y recibir métodos http). Los mensajes Wf-XML para todas las operaciones son integradas como datos de entrada y salida con respecto a las interacciones http.

En más detalle, una operación es codificada con el método http POST. POST está dirigida a alguna URI y puede tener MIME (Multipurpose Internet Mail Extension) body parts[3] para entrada y salida. Para Wf-XML, se usa exactamente un MIME body part para entrada y exactamente un MIME body part para salida.

La dirección URI a la cuál es dirigido el método POST es la clave del recurso del mensaje Wf-XML. Esta clave puede ser encontrada en uno o dos lugares dentro del mensaje. La localización primaria es el elemento "Key" dentro del elemento WfMessageHeader. De todos modos si el mensaje es un Acknowledgement (usado en procesamiento asincrónico) o mensajes batch (que contiene múltiples cabezales), el elemento "Key" dentro del elemento Dialog de la sección WfTransport sirve como el identificador del recurso con el cual el método POST es dirigido.

Como alternativa al direccionamiento absoluto, una implementación debe elegir mantener una base URI por un recurso internamente y combinarla con una URI relativa en el cabezal del mensaje para formular una URI absoluta. Si una implementación desea utilizar URIs relativas de esta forma, detalles adicionales deben ser acordados en el contrato de interoperabilidad. En todo caso, estos datos son específicos de una de las partes y debe ser conocido de antemano (por ejemplo, en el caso de una key de definición de proceso) o obtenido como el resultado de un parámetro de respuesta retornado por una operación previa (por ejemplo "ProcessInstanceKey"). Por el propósito de este binding, todas las URIs finales serán resueltas vía http, por ejemplo deben ser de la forma: " http://...".

En un procesamiento sincrónico, el mensaje Request Wf-XML es un MIME body part para entrada y el mensaje de Response Wf-XML un MIME body part para salida. Más aún, estos mensajes deben ser incluidos con sus respectivos request/response http. Esto quiere decir, que un mensaje de Request Wf-XML debe ser enviado con un request http y un mensaje Response Wf-XML debe ser enviado como un response http.

En un procesamiento asincrónico, un mensaje Wf-XML (consistente en Request y/o Responses) son un MIME body part para entrada y el mensaje Acknowledgement Wf-XML es un MIME body part para salida.

Todas los Wf-XML MIME body parts deben usar MIME content type "Content-type:text/xml" (tipo de del objeto enviado en el cuerpo del mensaje http) en el cabezal http-method , y la content-length (número de byte del tamaño del objeto enviado en el cuerpo del mensaje) debe ser acordada en el largo del mensaje Wf-XML request o response respectivamente.

Todos los mensajes Wf-XML procesados son sujeto de la ejecución exitosa del procesamiento de los métodos http. Por lo tanto, todos los códigos de estados http deben ser interpretados independientemente por esta especificación, y todos los procesamientos Wf-XML, asumen la completitud exitosa de los procedimientos http previamente a la ejecución. Para la autenticación, el mecanismo usual debería ser utilizado. Este incluye el uso de las respectivos campos de cabezales http.

[3] MIME body part significa una secuencia de bytes con un tipo de contexto y una codificación de transferencia del contexto, por defecto el tipo de contexto en un MIME body part es text/plain (una secuencia de líneas de texto) Un multipart/mixed es un tipo que encapsula varios body parts en uno.
No se debe confundir con el concepto de mensaje MIME que es un caso especial de un MIME body part.

10.6 Document Type Definition (DTD)

Ver apéndice A.

10.7 Terminología

Términos, acrónimos y abreviaturas usadas:

- DTD – Document Type Definition., un conjunto de declaraciones de marcas que provee una gramática para una clase de documento.
- Element – Un componente de un documento XML que consiste en una marca y el texto contenido entre esta marca.
- Root Element – El elemento más externo de una instancia de documento, tal que el elemento no aparece en el contexto de otro elemento en la instancia.
- Attribute – un componente de un documento XML usado para asociar propiedades con un elemento.
- Entity – Una unidad de almacenamiento en la cuál la unidad de almacenamiento está asociado a un nombre.
- Document Instance – Una instancia de tipo documento o clase documentos.

10.8 Futuras direcciones (no normativas)

Se describen un conjunto de características de la especificación Wf-XML las cuales se ha considerado realizar mejoras en futuras versiones. Aunque son sugerencias preliminares pueden dar directivas a desarrolladores de los temas que se están considerando.

10.8.1 Protocolo de mensaje.

La especificación puede ser considerada que consiste en dos componentes mayores: el protocolo de mensajes y las funciones de interoperabilidad.
El protocolo de mensajes es esencial para comprender la estructura global de los mensajes, un mecanismo de manejo de excepciones, el mecanismo de identificación y direccionamiento, y los bindings de capa de transporte. La WfMC creo el protocolo de mensaje Wf-XML dado que en ese momento no habían protocolos adecuados disponibles.

Actualmente la industria y los estándares se han comenzado a usar mensajería basada en XML. Han surgido protocolos como **XML-RPC[4], Blocks (BXXP), SOAP[5], XP y ebXML TR&P (ebXML transport, routing and packaging)** (ver el punto ebxml)**.**

Actualmente, la coalición está considerando estos protocolos como alternativa de protocolo de mensajes Wf-XML. Esto puede derivar en el reemplazo del mismo por uno de estos estándares emergentes o resolver permanecer independiente de ellos.

10.8.2 Especificación de un meta-lenguaje

La especificación utiliza sintaxis Document Type Definition (DTD) para definir el vocabulario y la gramática de Wf-XML.

[4] XML-RPC desarrollado en 1998 es un mecanismo simple RPC que utiliza XML sobre HTTP,
[5] SOAP (Simple Access Object Protocol) es un protocolo liviano para ambientes distribuidos basado en XML, utilizado en los Web Services.

De todas maneras se reconoce que dado el uso actual de XML existen numerosas alternativas como esquema de definición de lenguaje. Entre ellos se incluye **W3C XML schema definition language (XSDL), Schematron[6], RELAX, TREX y otros.**

Es muy probable que la próxima versión considere a W3C XML schema definition language (XSDL) dado su enriquecimiento semántico y la validación de tipos de datos. (se pueden ver las características y ventajas sobre DTD en punto XML Schema de Marco teórico – eXtensible Markup Language)

Puede existir la posibilidad que se opte por una especificación de esquema neutral y se permita la creación de bindings a lenguajes de esquemas .

10.8.3 Operaciones

Las operaciones serán reservadas para futuros usos. Se describen en alto nivel ya que los detalles de las funcionalidades no están aún determinados. La siguiente tabla muestra las operaciones adicionales:

	Control	Process Definition	Process Instance	Observer
WfQueryInterface	X			
ListInstance		X		
SetData			X	
Subscribe			X	
Unsubscribe			X	
GetHistory			X	

Operaciones de control

- **WfQueryInterface:** Se usa para consultar una implementación por varias capacidades genéricas. En particular puede ser usada para determinar los requerimientos de seguridad, conforme a esta especificación o características de procesamiento de XML.

Operaciones de definición de proceso

- **ListInstance:** Se usa para obtener la lista de instancias de una definición de proceso dada. Se retorna una lista conteniendo la clave, el nombre y la prioridad.

Operaciones instancias de procesos

- **SetData:** Se usa para asignar valores a un número de propiedades de un recurso de instancia de proceso. Permite el seteo de parámetros, se pueden setear todos o algunos de ellos. Retorna los valores de todas las propiedades del recurso.

[6] Schematron es distinto a los demás esquemas dado que valida esquemas usando patrones en vez de definiciones de esquemas. Posee una definición simple y provee una poderosa especificación de restricciones a través de Xpath. Soporta namespaces, no soporta tipos de datos.

- **Subscribe:** Se usa para registrar un recurso en otro, como parte interesada en el cambio de estados y eventos que ocurran. Si este recurso no soporta otros observadores se produce una excepción.

- **Unsubscribe:** Se usa para remover un recurso de la lista de observadores de otro recurso.

- **GetHistory:** Se utiliza para recibir la lista de todos los eventos ocurridos en un recurso. Si el servicio que implementa el recurso no registra el log de transacción no se podrá tener la historia disponible.

10.8.4 Mecanismos auxiliares soportados

Contrato de interoperabilidad

Mientras se realizan recomendaciones de realizar un contrato de interoperabilidad entre los sistemas de activación de workflow, no existen sintaxis bien definida o estructuras para ese contrato.
Como el uso de los web services continúa su propagación a través de varias industrias, se han producido varios avances en el área de la interoperabilidad dinámica. Uno de los avances es la especificación de **trading partner agreement Markup Language (tpaML)** la que se recomienda en la iniciativa ebXML. (Ver el punto tpaML)

Implementación de un marco de trabajo de referencia

Una herramienta muy útil para los implementadores de nuevas especificaciones es un marco de trabajo de referencia donde se puedan basar. Se considera el desarrollo del mismo en un ambiente "open source".

Test de conformidad

Una de las metas de la WfMC es construir test de conformidad, se están realizando pasos para facilitar el testeo de la especificación, también se está estudiando un mecanismo de certificación sobre el marco de trabajo de implementación discutido en el punto anterior.

11. Estado de la industria con respecto a los estándares de la WfMC

Periódicamente, son realizadas demostraciones, patrocinadas por la WfMC, para validar y promover la realimentación con respecto a las especificaciones de interoperabilidad, entre los proveedores de workflow.

Los proveedores para demostrar que poseen un producto en conformidad a las especificaciones deben:

1. Demostrar que los productos pueden completar los testeos descriptos en el marco de trabajo de interoperabilidad interoperando con un producto de otro vendedor.
2. Publicar una matriz de conformidad donde muestra los resultados de los testeos.
3. Establecer públicamente una fecha de cuando se liberará el producto al mercado.

La tabla[7] siguiente, publicada por la WfMC, muestra el estado de la conformidad de proveedores de sistemas de workflow, que interfaces se ha probado que verifican y como ha sido lograda (prototipo, desarrollo, etc.).
Sirve como guía cuando se trate de incorporar un sistema de workflow ya que podemos verificar si el producto verifica las interfaces de la WfMC. Por ejemplo, es altamente deseable, que los productos cumplan la interface 4 de interoperabilidad, ya sea para que el sistema de workflow pueda ser integrado con otros existentes o para futuras integraciones. En nuestro caso nos interesa la especificación del binding de la interface 4, Wf-XML.

Octubre 2002.

Proveedor	**Producto /Versión**	**Versión de fecha**	**WfMC Interface Soportada**	**Implementación**
Action Technologies Inc	ActionWorks Metro 5.0	27/4/98	IF2/3 IF4	Prototipo demostrado
BancTec/Plexus Software	FloWare V 5.0	9/98	IF1 IF2/3 IF4	Sí Prot. demostrado Planeado
Blockade Systems Corp.	ManageID™4.2	2/2003	IF1 IF4	Sí
BOC GmbH	ADONIS V2.0/V3.0	1998	IF1 IF5	Prototipo Desarrollo Trial
Concentus	KI Shell 2.0/3.0	25/6/97	IF1 IF2/3 IF4 IF5	No Sí Sí No
CSE Systems	CSE/WorkFlow Gold Edition	9/97	IF1,IF2/3,IF4	Sí
	Cibon	31/3/99	IF1,IF2/3 IF4	Sí Soporta JointFlow
Drala Software	Drala Workflow	TBA	**Wf-XML v1.0**	Planeado

[7] Se puede acceder a la tabla en la dirección: http://www.wfmc.org/standards/conformance

Inc.	Engine			
FileNET	Visual WorkFlo 3.0 Integrated WorkFlo 1.22	9/98	IF4	Sí
HandySoft Corp	Handy*Solution V5.0	30/9/98	IF2/3 IF4	Planeado
Hitachi Ltd	Groupmax Workflow V5	5/99	IF4	Sí
	WorkCoordinator	11/98	IF2/3 IF4	Sí Planeado
IBM	MQ Series Workflow	2002	IF1 IF2/4/5 & OMG JointFlow	Implementado Planeado
Identitech, Inc.	FYI® Flowoks	2002	**WF-XML** and XPDL	Planeado
IDS Prof. Scheer GmbH	ARIS Toolset 3.2 Modelling and BPR Tool	1997	IF1	Prototipo
Image Integration Systems	DocuSphere Workflow V5.0	2002	IF1, IF2/3 IF4	Planeado Prototipo
INSIEL Sp.A	Office241 OfficeFlow V2.6.9a	6/98	IF2/3 IF4 IF5	Sí Prototipo Planeado
ivyTeam	ivyGrid Version 2.1	06/2002	IF 1/2/3 IF 4/5	Impl.en parte Planeado
KAISHA-Tec	Modeler Pro V 1.2	5/97	IF1	Sí
Ley GmbH	COSA Workflow 2.0	31/03/97	IF1,IF2/3,IF4,IF5	No
Meta Software Corporation	WorkFlow Analyzer V 4.0	6/98	IF1	WPDL 7.04 Beta Implementado 1.0
PROMATIS	INCOME Workflow V 1.0	1/4/98	IF1,IF2/3 IF4 IF5	Sí No Sí
SAP	SAP WebFlow/ SAP Business Workflow	1997 R/3. Rel. 3.1 2000 Technology Release 4.6C	IF2/3 **Wf-XML**	Implementado Implemented
Staffware	Staffware Process Suite 1.0	December 2001		
	Staffware Process Suite 1.5	October 2002	IF 1 If 2/3 If 4 MIME **IF 4 WfXML 1.0** IF 5	Desarrollo Trial Prototipo Demost. Implementado Implementado Desarrollo Trial

Technology Deployment International	WebDeploy: WorkFlow 1.2	6/98	IF2	Sí
TIBCO	TIB/InConcert	2000 1996 1999 planeado	IF 1 IF 2/3 IF 4 IF 5	Prototipo Demost. Prototipo Demost. Protipo Demost. TIB/InConcert soporta la especificación abstracta de Auditoria de Datos
Versata Inc	Versata Interaction Server (VIS)	TBA	**Wf-XML v1.0**	Implementado

11.1 SAP WebFlow

Se incluye un artículo publicado en el sitio http://www.sap.info que describe como el producto SAP, WebFlow incorpora interface Wf-XML y los beneficios que encuentra.

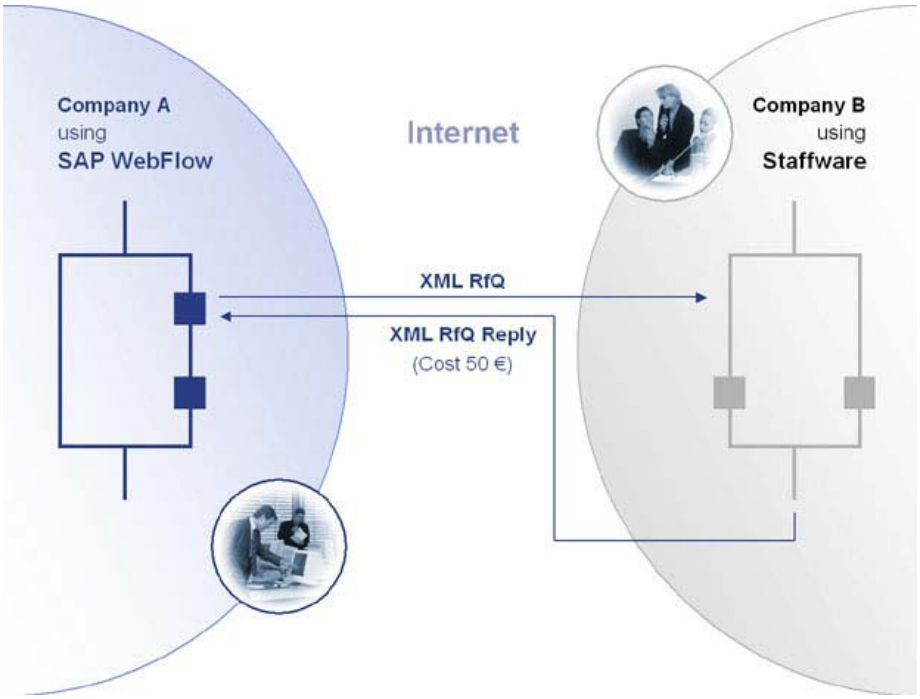
La solución de workflow de SAP llamada WebFlow usa la interface abierta Wf-XML, SAP ha elegido embeber directamente Wf-XML en el motor de WebFlow Release 4.6c dado como puede ayudar a modernizar los negocios.

Una simple experiencia llevada a cabo por SAP en el 2000 mostró la facilidad de uso de esta especificación. Se creó un workflow mediante, el componente de WebFlow para crear definiciones de workflow, Workflow builder para definir las variables del escenario. Se eligió un escenario de colaboración a través de un workflow anidado (RFQ Request For Quotation), se incluye un paso Web en la definición del proceso que tiene un campo con la URI apropiada, ésta representa la localización del sistema del socio de negocio que está involucrado en el workflow.

Con este paso el workflow queda listo para iniciar, el paso Web envía un mensaje XML que contiene la petición RFQ, a través del puerto de firewall. El mismo atraviesa la Internet hasta llegar al puerto del firewall del socio de negocio, donde es autenticado y dispara una instancia de workflow en su sistema de workflow.

En la empresa del socio de negocio se sucederán las pasos automáticos para obtener la información para que se genere un ítem de trabajo que llegará al inbox del sistema de workflow de un participante, se satisfará el trabajo, luego el sistema de workflow generará la respuesta XML que retornará al workflow de la compañía solicitante. El proceso solicitante retornará de su espera, y automáticamente desempaquetará los datos del mensaje XML y almacenará los datos en la base de datos del workflow antes de continuar en el siguiente paso de procesamiento (comparará el RFQ y se notificarán a los compradores).

Lo completamente distinto en este escenario es que el sistema del socio de negocio no es una solución SAP sino un sistema de workflow STAFFWARE (disponible a través de su Staffware Developer's Kit o SDK). Ambos sistemas WebFlow y Staffware han integrado la interface abierta Wf-XML en sus productos para posibilitar la integración de procesos de negocios sobre Internet de manera tan sencilla como realizar el "Drag and Drop" de una tarea Web.
El hecho que esta interface halla sido integrada en diferentes productos comerciales refleja el poder de la interface y la fuerza de los requerimientos de negocio para integrara sistemas de workflow en Internet y además internamente entre la compañía utilizando plataformas heterogéneas.



Que es exactamente Wf-XML para SAP?

Es solo una interface abierta, especialmente diseñada para la integración de procesos genéricos. Dado que Wf-XML es una interface genérica puede ser usada en todos los procesos de negocios. La interface define un número de llamados y respuestas que pueden ser usadas para posibilitar comunicarse a los procesos de workflow y sincronizarse entre ellos. Estos llamados son restringidos en número, para animar a diferentes vendedores de workflow de integrar esta interface en sus productos.
Típicamente los mensajes XML son distribuidos a través de método post de http, por lo que pueden ser usados para comunicarse sobre Internet desde cualquier plataforma, tomando ventaja de la seguridad ofrecida por la plataforma. La combinación de XML y métodos post de http son especialmente adecuados para la automatización de procesos de colaboración donde se requiere acuses de recibo inmediatos sincrónicos, porque los sistemas de workflow no pueden permanecer en espera mientras que el proceso remoto completa el procesamiento.

El acuse de recibo sincrónico es esencial en el mundo real donde los sistemas están muchas veces bajos e Internet no es suficientemente confiable para garantizar la entrega de cada mensaje. De todos modos especificando que las contestaciones son asincrónicas (en contraste a los acuses de recibo que son sincrónicos) esta interface permite al workflow central continuar procesando otras tareas mientras se espera por la respuesta del workflow remoto. Recuérdese que si muchos usuarios están involucrados en el sistema remoto, pueden transcurrir días antes que la respuesta XML retorne. El mensaje XML contienen la respuesta Wf-XML junto con la información de contexto del proceso.

Cómo esta interface es genérica se puede acordar libremente con los socios de negocio en que datos de contexto se requieren y como se empaquetan. Si se quiere automatizar un proceso estandarizado puede simplificar este paso optando por el estándar XML apropiado para este tipo de proceso

En el caso más simple, donde dos componentes SAP se comunican entre si, se puede confiar en el paso Web para empaquetar los datos desde el contenedor de workflow directamente al mensaje Wf-XML, sin crear un documento intermedio XML. En situaciones más complejas se puede definir métodos para empaquetar los objetos de negocio (SAP business object) en un documentos XML a ser incluido en la parte del contexto del mensaje XML. De la misma forma no hay que preocuparse sobre la estructura de

los mensajes Wf-XML. WebFlow se ocupa de ello, junto con la interpretación y el procesamiento de los mensajes Wf-XML que ingresan al sistema. Similarmente, la sincronización también es automática. Si el proceso central del workflow es cancelado, el proceso del socio de negocio es cancelado también, sin la necesidad de preocuparse de cómo esto es realizado. Lo que se ocupa de la automatización es la interface Wf-XML embebida en el motor de WebFlow.

Un caso de negocio

Cuando se automatiza los procesos de negocio se optimiza y mejora la calidad de resultados mientras se reducen los costos. Extendiendo el proceso de negocio, muchas veces alcanza los límites del sistema. Este es el fin del sistema físico (el alcance del sistema de workflow termina allí) o el límite corporativo donde la interacción entre socios se transforma en importante. Es precisamente aquí que la interface abierta de Wf-XML se utiliza.

En el mundo actual, el suceso de una compañía depende mucho de la calidad de la iteración entre ella y sus socios. No se podrá dictar exactamente como un socio puede llevar a cabo sus procesos de negocios o que herramientas usa pero se puede demandar que realice la parte del trabajo de forma correcta. Esto es el porque es importante cuando dos partes han comprometido a la automatización del proceso, la interacción entre los sistemas debe ser robusta y rápida como el proceso dentro de la compañía. Aquí es donde Wf-XML ingresa. Los sistemas de workflow que soportan Wf-XML proveen un conjunto de conexiones de rápido ajuste para hacer de una integración un trabajo fácil.

Siguientes pasos

Hay planes que la siguiente versión de Wf-XML, provea mayor acoplamiento y probablemente incorporará la infraestructura SOAP. WebFlow está comprometido con esta interface abierta y en posibilitar a los clientes SAP de integrar sus procesos de negocios sobre Internet con un mínimo de esfuerzo. A pesar de que la interface primariamente apunta a la administración de procesos de negocio de colaboración, se encontrará que es muy conveniente la forma de integrar aplicaciones de workflow que ejecutan en diferentes sistemas, aún si los mismos son de WebFlow. Haciendo que sean débilmente acoplados los procesos entre si, hay más áreas para optimizar los procesos individuales sin tenerse que preocupar por los efectos colaterales sobre el proceso global.
Para finalizar cuando se necesite que un proceso deba atravesar los límites de la compañía hacia otra donde otro motor de workflow de otro vendedor administre el segundo sistema, no significará problema si se tiene el soporte de Wf-XML. (Autor Alan Rickayzen)

12. Proyecto AFRICA, un ejemplo de utilización interoperabilidad de workflow basado en Wf-XML

(AFRICA - A Flexible Reliable Intelligent Communication Architecture)

El proyecto AFRICA fue creado para construir una plataforma confiable para workflow de negocio a negocio, usando mensajes codificados en Wf-XML. El foco estaba en incorporar modelos de procesos no secuenciales complejos que involucren muchos socios y que integrara un servicio de monitoreo global. El proyecto posee muchos sistemas de workflow comerciales con los cuales probar la interoperabilidad. El prototipo propuesto como extensión de esta propuesta como un componente (wrapper) que pueda agregar a los workflow existentes características para interoperar.
La primer versión del prototipo fue finalizado en marzo del 2000 y demostró el potencial de la comunicación de los procesos basados en XML.

Se utilizaron una serie de principios de diseño para que fuera un sistema usable en un gran número de contextos, reusable, seguro y que pueda soportar procesos que incluyen varias organizaciones.

Independencia de sistemas: debe posibilitar a las compañías a participar en el workflow interorganizacional son modificar los sistemas de workflow subyacente. Por eso es que el prototipo es un wrapper que se sitúa por encima de los sistemas de aplicación y encapsula la administración de los mensajes Wf-XML del sistema subyacente. Se utilizan las API 's provistas por los productos para acceder a los sistemas conservando la integridad de los mismos.

Reusabilidad: está diseñado para que se pueda ser reutilizado, para ello se separan en capas de transporte, capa de lógica de procesos y capa de abstracción.

Seguridad: La comunicación entre sistemas que puedan comunicarse a través de AFRICA debe ser de modo seguro y confiable, para lograrlo información adicional se ha insertado en la sección de transporte de los mensajes Wf-XML que son evaluados por la capa de transporte del wrapper.

El formato de mensajes utilizado es el de Wf-XML.

12.1 Arquitectura de la solución.

La capa de transporte administra la transferencia segura y confiable de los mensajes Wf-XML entre los sistemas que soportan AFRICA. Utiliza TCP/IP y SSL como mecanismo de seguridad.

La capa de lógica de procesos realiza el parsing de los mensajes XML, los comandos Wf-XML separan de los datos del contexto y es realizado un llamado a la función API de la capa de abstracción.
Después de recibir el mensaje de la capa de transporte, el componente de administración de mensajes determina si es una respuesta a una solicitud anterior o si es una solicitud. Este componente extrae los comandos Wf-XML de los mensajes y lo envía con los datos de contexto y la información de seguimiento de auditoría al administrador de instancias de proceso. Este componente lee el identificador del proceso y encuentra la instancia de proceso local que existe en el sistema de workflow local. Traslada el comando global en una instancia local del comando específico y se lo pasa a la nueva instancia del administrador de operaciones.
El administrador de operaciones transforma el contexto de datos en el formato requerido por el esquema del workflow local y le pasa el comando y los datos de contexto a través de un llamado de una API a la capa de abstracción.
Dependiendo del contenido del contrato de interoperabilidad la interpretación de la semántica de los datos de contexto llevada realizada en esta capa. El árbol XML de resultado de los datos de contexto puede ser transformado en varias formas, de acuerdo a los requerimientos del sistema de aplicación detrás de la capa de abstracción

La capa de abstracción encapsula los llamados a API's propietarias del sistema de administración de workflow de los vendedores específicos y exhibe una API estandarizada a la capa de lógica de proceso. Para el prototipo dos capas de abstracción fueron completadas, una para la interacción con un servidor web y otra para el componente de Workflow del sistema SAP R/3. En este escenario, el comando y los datos del contexto recibidos por la capa de lógica de proceso son trasladadas a la BAPI (Business Object API) específica de SAP.

Para incorporar distintos sistemas de workflow de distintos vendedores solo se debe cambiar la capa de abstracción, dejando la capa de proceso de negocio y transporte intactas.

13. XML – eXtensible Markup Language

Uno de los principales temas de este trabajo es la utilización del lenguaje XML desarrollado por la W3C (World Wide Web Consortium) como formato de los datos para la información del intercambio. Debido a que su surgimiento es reciente, se desea realizar una introducción sobre ésta donde, además, se definirán términos básicos para la comprensión de su posterior aplicación.

XML (eXtensible Markup Language) es un lenguaje de etiquetas: un subconjunto de SGML. XML es más sencillo comprender, leer y manipular una especificación XML tanto para una persona, como para una máquina, que una especificación SGML. Siendo un subconjunto de SGML, conserva las siguientes propiedades:

- Extensible: los autores pueden definir nuevas etiquetas y atributos para documentos, especificando su sintaxis y semántica.

- Estructurado: los documentos pueden ser contenedores de otros documentos, lo que permite construir documentos complejos a partir de documentos simples

- Auto-validable: todo documento puede hacer referencia a una descripción de su gramática de forma que las aplicaciones pueden validar la correcta formación del documento; además, este proceso puede ser automatizado.

XML se encuentra orientado a la estructura de los datos, a diferencia de HTML que trabaja sobre el formato de presentación de los mismos.

XML puede verse hoy en día desde diferentes enfoques: para estructuración de la Internet, búsqueda y recolección de datos más eficiente, personalización de los datos referentes a un usuario e intercambio de datos entre aplicaciones. Dependiendo del enfoque, es que surgen una serie de aplicaciones para las cuales XML resulta atractivo:

1. Aquellas que requieren que el cliente Web medie entre dos o más bases de datos heterogéneas.
2. Las que intentan distribuir una porción importante de la carga de procesamiento desde el servidor Web al cliente.
3. Aplicaciones que requieren que el cliente Web presente diferentes visualizaciones de los mismos datos a diferentes usuarios.
4. Aquellas en las que agentes Web inteligentes intentan adaptar el resultado de búsquedas de información a las necesidades de usuarios individuales.

Hoy en día, la W3C es quién centra el desarrollo entorno a XML: las tecnologías que esta implica, y las especificaciones de cada una de ellas.

14. Características de XML

Características de XML:

- Simple: es legible por lectores humanos y fácil de procesar por las computadoras.
- Abierto: es un estándar de la W3C, aprobado y soportado por líderes de la industria informática.
- Extensible: no hay un conjunto fijo de etiquetas; nuevas etiquetas pueden ser creadas cuando son necesarias.

- Auto-descriptivo: en bases de datos tradicionales, los registros de datos requieren un esquema soportado por el administrador de la base de datos. Los documentos XML pueden ser almacenados sin dicha definición porque contienen metadatos en la forma de etiquetas y atributos.
- Provee la base para identificador del autor y de versiones a nivel de elemento: una etiqueta XML posee ilimitado número de atributos como autor o versión.
- Información de contexto legible por las máquinas: etiquetas, atributos y estructuras de elementos proveen información de contexto que puede ser utilizada para interpretar el significado del contenido, abriendo nuevas posibilidades para alta eficiencia de máquinas de búsqueda, agentes, etc.
- Separación del contenido de la presentación: las etiquetas XML describen significado, no presentación. El maquillaje de los documentos XML puede ser controlado por hojas de estilo XSL, permitiendo modificar la vista del documento sin tocar el contenido del documento.
- Soporte de documentos multilenguaje y Unicode: importante para la globalización de aplicaciones.
- Facilita la comparación y agregación de datos: la estructura en árbol de XML habilita la comparación de documentos y permite agregar datos eficientemente elemento por elemento.
- Embebe múltiples tipos de datos: los documentos XML pueden contener cualquier tipo de datos, desde datos multimedia (imagen, sonido y video) a componentes activos (Java Applets y ActiveX).
- Embebe datos existentes: la correspondencia entre estructuras de datos (como un sistema de archivos o bases de datos relacionales) a XML es simple. XML cubre todas las estructuras de datos existentes y soporta múltiples formatos de datos.
- Vista de "un solo servidor" para datos distribuidos: documentos XML pueden consistir de elementos anidados distribuidos en múltiples servidores remotos.
- Gran aceptación en la industria: IBM, Sun, Microsoft, Netscape, DataChannel, SAP y muchas otras han anunciado soporte a XML. Microsoft utiliza XML como el formato de intercambio para su línea de productos Office, y como tecnología básica para Microsoft .NET y XML Web Services, mientras que los navegadores Internet Explorer y Netscape ya soportan XML.

14.1 Especificación del Lenguaje

Este es un breve resumen de la especificación realizada por la W3C.

14.1.1 Etiquetas

Etiquetas o "tags" (en la jerga informática) identifican los datos. A diferencia de HTML, donde las etiquetas son "comandos" que dan formato a los datos, en XML son utilizadas para identificar datos. Considérese la definición de la estructura de un correo electrónico:

```
<mensaje>
        <fecha>2002-11-12</fecha>
        <para>usuario1@adinet.com.uy</para>
        <de>usuario2@adinet.com.uy</de>
        <asunto>Tesis</asunto>
        <texto>Reunión viernes 10 a.m.</texto>
</mensaje>
```

Los datos entre la etiqueta de comienzo y la de fin definen un elemento de datos XML. Además, el contenido de la etiqueta <para> se encuentra dentro del alcance de la etiqueta <mensaje>…</mensaje>. Es esta característica de contener a otros la que permite a XML representar estructuras jerárquicas.

14.1.2 Etiquetas y atributos

Las etiquetas pueden contener atributos, información adicional incluida como parte de la misma , dentro de los símbolos "<" y ">" que lo delimitan. El ejemplo anterior del cuerpo de un correo electrónico puede escribirse de la siguiente manera utilizando atributos:

```
<mensaje fecha="2002-11-12" para=usuario2@adinet.com.uy
  de="usuario1@adinet.com.uy" asunto="Tesis">
        <texto>
              Reunión viernes 10 a.m.
        </texto>
</mensaje>
```

Como en HTML, el nombre de atributo es seguido por un símbolo "=" y el valor del atributo y múltiples atributos son separados por espacios entre los mismos. Pero, a diferencia de HTML, no puede utilizarse la coma para separar atributos: la utilización de la misma provoca un error en la sintaxis.

Considerando que una estructura puede ser diseñada utilizando etiquetas o atributos (como en los dos ejemplos del correo electrónico), un punto a tener en cuenta es la representación que se utilizará. Dependiendo del propósito de la representación se determinará cual es la mejor alternativa para ese caso.

14.1.3 Etiquetas vacías

Existen diversas reglas que determinan cuando un documento esta **bien formado**. Una de las más importantes es que cada etiqueta de apertura debe tener una etiqueta de cierre, otra característica que hace un documento bien formado es que las etiquetas son completamente anidadas (una estructura <mensaje> <para> … </mensaje> </para> no es válida).

Ahora, en el caso que la etiqueta no posea contenido, debido al tamaño que poseen los documentos XML, se ha definido el concepto de etiqueta vacía (como una forma de minimizar dicho tamaño). Una etiqueta vacía comienza de la misma forma que una etiqueta de apertura, con la diferencia que antes del cierre, el último carácter es un símbolo de división ("/", el mismo con el que comienza la etiqueta de cierre). De esta forma se le indica al analizador XML que no espere etiqueta de cierre para la etiqueta recién reconocida.

Siguiendo con el ejemplo del correo electrónico, un campo que podría haberse encontrado en el mismo es el "con copia". Si fuera necesario que este apareciera en la estructura del mensaje (aunque no posea contenido), puede representarse:

```
<mensaje>
        <fecha>2002-11-12</fecha>
        <para>usuario1@adinet.com.uy</para>
        <de>usuario2@adinet.com.uy</de>
        <con_copia/>
        <asunto>Tesis</asunto>
        <texto>Reunión viernes 10 a.m.</texto>
</mensaje>
```

14.1.4 Comentarios

Igual que en HTML:

```
<mensaje fecha="2002-11-12" para=" usuario1@adinet.com.uy"
```

```
de=“ usuario2@adinet.com.uy” asunto=“ Tesis”>
        <!-- Esto es un comentario -->
        <texto> Reunión viernes 10 a.m texto>
</mensaje>
```

14.1.5 Prólogo

Un documento XML siempre comienza con un prólogo; el mínimo prólogo debe contener una **declaración** que identifica el documento como un documento XML:

```
<?xml version=“1.0”?>
```

Esta declaración puede contener información adicional:

```
<?xml version=“1.0” encoding=“ISO-8859-1” standalone=“yes” ?>
```

La declaración es esencial y está definida por <? … ?>, y puede contener los siguientes atributos:

- version: identifica la versión del lenguaje XML utilizado en los datos. Este atributo no es opcional.
- encoding: identifica el conjunto de caracteres utilizado para codificar los datos. La codificación por defecto es Unicode comprimido (UTF-8). El del ejemplo es la codificación “Latin-1”, conjunto de caracteres del inglés.
- standalone: indica cuando el documento referencia una entidad externa o una especificación de tipos de datos externa. Si no hay referencia externa, el valor de “yes” es el apropiado.

El prólogo puede contener definición de **entidades** (ítems que son insertados cuando se referencian desde el documento) y especificaciones que indican que etiquetas son válidas dentro del documento, ambos declarados en un DTD (Documento Type Definition).

15. Especificaciones complementarias

XML surge como una forma de estructurar los datos; debido a esta características, existen otras especificaciones que la complementan; algunos ejemplos son XSL (eXtensible Stylesheet Language), un lenguaje que aplica una transformación a un documento XML por ejemplo aplicar un formato de presentación, o XLink (una especificación para el direccionamiento de páginas XML o fragmentos de documentos).
Se presenta una breve introducción a XSL, XML Namespace, DTD (Document Type Definitions), XML-Schema, Xpath y XML-QL .

15.1 XSL – eXtensible Stylesheet Language

Una característica muy importante de XML es que puede transformarse en múltiples formatos.
El lenguaje de estilo XSL provee sintaxis simple para aplicar estilos a los documentos XML. Uno de los principales usos es definir como un documento XML será desplegado por ejemplo en un navegador, permite además la transformación de un documento XML de un formato a otro.

Para entender mejor este concepto se mostrará un ejemplo de la utilización.

Cuando se abre un documento XML en un navegador, se despliega la jerarquía de nodos de forma que se puede colapsar los distintos nodos como se ve en el siguiente ejemplo:

Archivo: “Mensaje sin estilo.xml”

```
<?xml version="1.0" ?>
```

```
- <lista>
  - <mensaje>
      <fecha>2002-11-12</fecha>
      <para>usuario1@adinet.com.uy</para>
      <de>usuario2@adinet.com.uy</de>
      <asunto>Tesis</asunto>
      <texto>Reunión viernes 10 a.m.</texto>
    </mensaje>
  - <mensaje>
      <fecha>2002-11-29</fecha>
      <para>usuario2@adinet.com.uy</para>
      <de>usuario1@adinet.com.uy</de>
      <con_copia>tutor@adinet.com.uy</con_copia>
      <asunto>Defensa de la Tesis</asunto>
      <texto>Reunión lunes 14 p.m.</texto>
    </mensaje>
  </lista>
```

Para que la información sea más presentable al usuario se puede utilizar una hoja de estilo para transformar el documento XML en elementos HTML y aplicar un formato determinado a los datos.

Ejemplo archivo XSL: Mensaje.xsl

```
- <xsl:stylesheet xmlns:xsl="http://www.w3.org/TR/WD-xsl">
  - <xsl:template match="/">
      Lista de Mensajes
      <xsl:apply-templates select="//mensaje" />
    </xsl:template>
  - <xsl:template match="mensaje">
    - <P>
        <HR />
        <xsl:apply-templates />
        <HR />
      </P>
    </xsl:template>
  - <xsl:template match="fecha">
      <FONT COLOR="green" />
      <BR />
      Fecha:
      <xsl:value-of />
      <BR />
    </xsl:template>
  - <xsl:template match="para">
      <FONT COLOR="red" />
    - <B>
        Para:
        <xsl:value-of />
      </B>
      <BR />
    </xsl:template>
  - <xsl:template match="de">
      <FONT COLOR="red" />
    - <B>
        De:
        <xsl:value-of />
      </B>
      <BR />
    </xsl:template>
  - <xsl:template match="asunto">
      <FONT COLOR="blue" />
```

```
        <xsl:value-of />
        <BR />
      </xsl:template>
    - <xsl:template match="texto">
        <FONT COLOR="black" />
      - <I>
          <xsl:value-of />
        </I>
      </xsl:template>
    </xsl:stylesheet>
```

Asociamos el estilo a nuestro documento XML haciendo referencia a la hoja de estilo mensajes.xsl

Ejemplo archivo mensaje.xml

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="mensajes.xsl"?>
<lista>
        <mensaje>
                <fecha>2002-11-12</fecha>
                <para>usuario1@adinet.com.uy</para>
                <de>usuario2@adinet.com.uy</de>
                <asunto>Tesis</asunto>
                <texto>Reunión viernes 10 a.m.</texto>
        </mensaje>
        <mensaje>
                <fecha>2002-11-29</fecha>
                <para>usuario2@adinet.com.uy</para>
                <de>usuario1@adinet.com.uy</de>
                <con_copia>tutor@adinet.com.uy</con_copia>
                <asunto>Defensa de la Tesis</asunto>
                <texto>Reunión lunes 14 p.m.</texto>
        </mensaje>
</lista>
```

Ahora en el navegador la información se mostrará de la siguiente manera.

Lista de Mensajes

Fecha: 2002-11-12
Para: usuario1@adinet.com.uy
De: usuario2@adinet.com.uy
Tesis
Reunión viernes 10 a.m.

Fecha: 2002-11-29
Para: usuario2@adinet.com.uy
De: usuario1@adinet.com.uy
Defensa de la Tesis
Reunión lunes 14 p.m.

15.2 XML Namespace

XML Namespace provee un método simple para calificar elementos y nombres de atributos usados en documentos XML mediante la asociación de ellos con identificadores de namespace (colección de nombres) mediante referencias URI.

Un documento simple puede contener elementos y atributos que son definidos y utilizados por múltiples módulos de software. Esos documentos contienen múltiples vocabularios de marcados, los cuales pueden tener problemas de reconocimiento y colisión. Los módulos de software deben ser capaces de reconocer las etiquetas y atributos, las cuales fueron diseñadas para ser procesadas, aún cuando haya colisiones con otros documentos, que usen el mismo tipo de elemento o nombre de atributo.

Estas consideraciones requieren que los constructores de documentos deban tener nombres universales, cuyo alcance se extienda más allá del contenido del documento.

Un XML Namespace es una colección de nombres, identificados por una referencia URI, la cual es utilizada en documentos XML como tipos de elementos y nombres de atributos.

Los nombres del XML Namespace aparecen como calificadores, los cuales contienen dos puntos que separan el prefijo de la colección y un nombre local. El prefijo, es emparejado a una referencia URI, selecciona un nombre. La combinación de un URI y el documento propietario produce identificadores que son universalmente únicos.

Un Namespace es declarado usando una familia de atributos reservados. Ese nombre de atributo debe ser xmlns o tener xmls: como prefijo. Estos atributos, como cualquier otro atributo XML, deben ser provistos directamente o por defecto.

NSAttName ::= PrefixedAttName | DefaultAttName
PrefixedAttName ::= **'xmlns**:' NCName
DefaultAttName ::= **'xmlns**:'
NCName ::=(Letter | '_') (NCNameChar)*
NCNameChar ::= Letter |Digit | '**.**' | '**-**' | '_' | CombiningChar | Extender

Un ejemplo de una declaración Namespace, el cual asocia el prefijo edi con el URI http://ecomerce.org/schema.

```
<x xmlns:edi='http://ecommerce.org/schema'>
<!-- el prefijo "edi" es limitado a http://ecommerce.org/schema para
los elementos "x" y el contenido -->
</x>
```

15.3 DTD

La especificación XML emplea la abreviatura DTD para referirse a definiciones del tipo de documento. La DTD define los tipos de elementos, atributos y entidades permitidas y puede expresar algunas limitaciones para combinarlos.

Los documentos que se ajustan a su DTD se denominan válidos. Al igual que una frase puede ser agramatical, un documento puede no ajustarse a su DTD y ser por lo tanto no válido. Pero esto no significa obligatoriamente que deje de ser un documento XML. La palabra válido no tiene su significado habitual. Un documento XML puede violar su DTD y seguir siendo un documento XML.

Al igual que las declaraciones del tipo de documento son opcionales, los documentos XML pueden no ajustarse para nada a ningún DTD. En ese caso no puede ser un documento válido, porque no puede demostrarse que se ajuste a una DTD. Pero tampoco es no válido por no ajustarse a una DTD.

XML no tiene una palabra para estos documentos que simplemente están bien formados.
Algunos los llaman "bien formados", pero no es lo bastante exacto. Si el documento no estuviera bien formado, no sería XML (por definición). Cuando se dice que un documento está bien formado no se dice nada de que se ajuste o no a ningún DTD.

La declaración del tipo de documento comienza en la primera línea con la expresión <!DOCTYPE a continuación de la que se encuentra el nombre del elemento que se define, y termina con "]>". Dentro de dicha declaración, se definen declaraciones de tipo de elemento (<!ELEMENT) las que representan etiquetas del documento y de tipo de atributo (<!ATTRIBUTE –un atributo simple- o <!ATTLIST –una lista de atributos) que identifican el o los atributos válidos para un cierto elemento.

El siguiente es un ejemplo de un documento XML con la declaración del tipo de documento.

```
<!DOCTYPE CorreoElectronico[
<!ELEMENT mensaje (fecha, para, de, con_copia, asunto, texto)
<!ATTRIBUTE mensaje prioridad CDATA>
<!ELEMENT fecha (#PCDATA)>
<!ELEMENT para (#PCDATA)>
<!ELEMENT de (#PCDATA)>
<!ELEMENT con_copia (#PCDATA)>
<!ELEMENT asunto (#PCDATA)>
<!ELEMENT texto (#PCDATA)>
]>

<mensaje prioridad=”urgente”>
        <fecha>2002-11-12</fecha>
        <para>usuario1@adinet.com.uy</para>
        <de>usuario2@adinet.com.uy</de>
        <asunto>Tesis</asunto>
        <texto> Reunión viernes 10 a.m </texto>
</mensaje>
```

En el ejemplo anterior todas las declaraciones DTD que definen la etiqueta DTD residen dentro de la entidad documento. Sin embargo la DTD puede haberse definido parcial o completamente en algún otro lugar. En ese caso, la declaración del tipo de documento contendrá una referencia a otra entidad que tenga dichas declaraciones.

En el siguiente ejemplo se muestra una declaración del tipo de documento con declaraciones DTD externas únicamente.

```
<?xml version = "1.0" ? >
<!DOCTYPE CORREO SYSTEM "http://www.dtds.com/correo.dtd>
<mensaje>
…
</mensaje>
```

La palabra clave SYSTEM indica al procesador que busque algún recurso que contenga la información externa.

15.3.1 Id, idref e idrefs

Son declaraciones de atributos. ID determina que un atributo particular define un identificador único para la entidad. De igual forma, IDREF significa que el valor del atributo es el identificador de algún otro elemento; IDREFS define que su valor es una lista de identificadores separados por espacios.

Siguiendo con el ejemplo del correo electrónico, se agrega un nuevo mensaje en respuesta al anterior y se guarda una referencia entre uno y otro como contestación del mismo. Se define para cada elemento un identificador.

```
<!DOCTYPE CorreoElectronico[
<!ELEMENT mensaje (fecha, para, de, con_copia, asunto, texto)
<!ATTRIBUTE mensaje mensajeID ID
        prioridad CDATA
        contestacion IDREF>
<!ELEMENT fecha (#PCDATA)>
<!ELEMENT para (#PCDATA)>
<!ELEMENT de (#PCDATA)>
<!ELEMENT con_copia (#PCDATA)>
<!ELEMENT asunto (#PCDATA)>
<!ELEMENT texto (#PCDATA)>
]>

<mensaje mensajeID=”2” prioridad=”normal”>
        <fecha>2002-11-12</fecha>
        <para> usuario1@adinet.com.uy </para>
        <de> usuario2@adinet.com.uy </de>
        <asunto>Tesis</asunto>
        <texto> Reunión viernes 10 a.m </texto>
</mensaje>

<mensaje mensajeID=”1” prioridad=”urgente” contestacion=”2”>
        <fecha>2002-11-12</fecha>
        <para>usuario1@adinet.com.uy</para>
        <de>usuario2@adinet.com.uy</de>
        <asunto>Tesis</asunto>
        <texto>Reunión viernes 10 a.m.</texto>
</mensaje>
```

15.4 XML SChema

El propósito de un esquema XML es definir una clase de documentos XML, y el término "instancia de un documento" es usado frecuentemente para describir un documento XML que conforma un esquema particular.

Consideremos la instancia de un documento, la cual describe un correo electrónico.

```
<?xml version="1.0"?>
<mensaje prioridad="Alta">
        <fecha>2002-11-12</fecha>
        <para> usuario1@adinet.com.uy </para>
        <de> usuario2@adinet.com.uy </de>
        <con_copia/>
        <asunto>Tesis</asunto>
        <texto>Reunión viernes 10 a.m.</texto>
```

```
</mensaje>
```

El correo electrónico consiste de un elemento principal mensaje y de subelementos para, de, con_copia, asunto y texto. Estos subelementos contienen a su vez otros subelementos.

Los elementos que contienen subelementos o contienen atributos son llamados de **Tipo complejo** (Complex types), mientras que los elementos que contienen números, caracteres y fechas, pero no contienen ningún subelemento son llamados de **Tipo simple** (Simple types). Algunos elementos tienen atributos, y los atributos siempre son de tipo simple.

15.4.1 Esquema del correo electrónico

```
<xsd:schema xmlns:xsd="http://www.tesis.com/XMLSchema">
        <xsd:annotation>
                <xsd:documentation>
                        Esquema de un correo electrónico tipo
                </xsd:documentation>
        </xsd:annotation>
        <xsd:element name="mensaje" type="mensajeTipo"/>
        <xsd:element name="con_copia" type="xsd:string"/>
        <xsd:complexType name="mensajeTipo">
                <xsd:element name="fecha" type="xsd:date"/>
                <xsd:element name="para" type="xsd:string"/>
                <xsd:element name="de" type="xsd:string"/>
                <xsd:element ref="con_copia" minOccurs="0"/>
                <xsd:element name="asunto" type="xsd:string"/>
                <xsd:element name="texto" type="xsd:string"/>
                <xsd:attribute name="prioridad" type="prioridadTipo" use="fixed"/>
        </xsd:complexType>
        <xsd:simpleType name="prioridadTipo" base="xsd:string">
                <xsd:enumeration value="Alta"/>
                <xsd:enumeration value="Normal"/>
                <xsd:enumeration value="Baja"/>
        </xsd:simpleType>
</xsd:schema>
```

El esquema del correo electrónico consiste de un elemento schema y una variedad de subelementos, los más importantes son element, complexType y simpleType, los cuales determinan la apariencia de los elementos y su contenido en las instancias de documentos.

Cada elemento en un esquema tiene un prefijo xsd: el cual es asociado con el esquema XML namespace.

15.4.2 Definición de tipos complejos, declaraciones Element y Attribute

Los nuevos tipos complejos son definidos usando el elemento complexType y esas definiciones típicamente contienen un conjunto de declaraciones de elementos, referencias de elementos y declaraciones de atributos. Los elementos son declarados utilizando el elemento element y los atributos son declarados utilizando el elemento attribute. Por ejemplo mensajeTipo es definido utilizando un tipo complejo, y con la definición de mensaje podemos ver seis declaraciones de elementos y una declaración de atributo:

```
<xsd:complexType name="mensajeTipo">
        <xsd:element name="fecha" type="xsd:date"/>
        <xsd:element name="para" type="xsd:string"/>
        <xsd:element name="de" type="xsd:string"/>
```

```
            <xsd:element ref="con_copia" minOccurs="0"/>
            <xsd:element name="asunto" type="xsd:string"/>
            <xsd:element name="texto" type="xsd:string"/>
            <xsd:attribute name="prioridad" type="prioridadTipo"
            use="fixed"/>
    </xsd:complexType>
```

Como consecuencia de esta definición cada elemento que aparezca en una instancia declarado del tipo mensaje debe contener seis elementos y un atributo. Estos elementos deben ser llamados fecha, para, de, con_copia, asunto y texto como fueron especificados por el valor de la declaración del atributo name. El primer elemento debe contener una fecha, los siguientes cinco elementos deben contener una cadena de caracteres.

La definición de mensajeTipo contiene la declaración de un atributo prioridad. De hecho, todas las declaraciones de atributos deben referenciar tipos de datos simples porque al contrario que las declaraciones de elementos, los atributos no pueden contener otros elementos u otros atributos.

En algunos casos es preferible utilizar un elemento existente en vez de declarar un elemento nuevo, por ejemplo:

```
<xsd:element ref="con_copia" minOccurs="0"/>
```

Esta declaración referencia a un elemento existente, con_copia, que ha sido declarado en el esquema del correo electrónico. En general, el valor del atributo ref debe referenciar a un elemento global, ejemplo, uno que ha sido declarado bajo el esquema más que como parte de un tipo de datos complejo. La consecuencia de esta declaración es que el elemento llamado con_copia debe aparecer en una instancia del documento, y su contenido debe ser consistente con el tipo del elemento, en este caso string.

El elemento con_copia es opcional dentro de mensaje porque el valor del atributo minOccurs en esta declaración es 0. Un elemento es requerido de aparecer cuando el valor de minOccurs es 1. El máximo número de veces que un elemento puede aparecer es determinado por el valor del atributo maxOccurs en su declaración. Este valor es un entero positivo o el término unbounded para indicar que no hay máximo número de ocurrencias. El valor por defecto de minOccurs es 1. Cuando un elemento es declarado sin el atributo maxOccurs, el máximo número de elementos ocurrentes es igual al valor del atributo minOccurs. Si este valor es también omitido el elemento debe aparecer exactamente una vez.

Los atributos pueden aparecer una vez o no aparecer, y la sintaxis para especificar las ocurrencias es diferente a la sintaxis de los elementos. El atributo use es usado en una declaración de atributos para indicar si el atributo es required u optional, y si es optional el valor del atributo es opcional.

15.4.3 Definición de tipos simples

El esquema del correo electrónico declara muchos elementos y atributos que tienen tipos simples (Simple types). Algunos de los tipos simples como ser string y date son definidos en el XML Schema, mientras que otros son derivados de su construcción. En el "Anexo C – XML Schema Tipos de Datos" se puede ver una lista de todos los tipos de datos simples construidos en el XML Schema.

Los nuevos tipos son definidos de la derivación de tipos simples existentes a través de la técnica llamada **restricción**. Un tipo nuevo debe tener un nombre diferente de un tipo existente y el nuevo tipo debe restringir el rango de los valores posibles. Se utiliza el elemento simpleType para definir un nuevo tipo simple, y podemos restringir sus valores aplicando una o más facetas.

Podemos ver en el ejemplo del correo electrónico la definición de un tipo simple llamado prioridadTipo que a través de la enumeración de facetas define el tipo simple. La enumeración de facetas limita un tipo simple a un conjunto de valores.

```
<xsd:simpleType name="prioridadTipo" base="xsd:string">
        <xsd:enumeration value="Alta"/>
        <xsd:enumeration value="Normal"/>
        <xsd:enumeration value="Baja"/>
</xsd:simpleType>
```

En este caso prioridadTipo será reemplazada por el tipo string usado en la declaración del elemento prioridad. Realizando el reemplazo, los valores del elemento prioridad estarán limitados a uno de los siguientes valores Alta, Normal, Baja. Podemos notar que la enumeración de valores especificados para este tipo particular debe ser único.

15.5 Diferencias entre XML Schemas y los DTD

Aunque ambos tienen propósitos similares, los Schemas XML ofrecen mayor precisión y flexibilidad. La siguiente lista sumariza las diferencias entre ambos:

- Modelo de contenido abierto
  Los Schema XML definen un modelo de contenido abierto. Esto posibilita que elementos y/o atributos adicionales puedan estar presentes en un elemento sin tener que declararlos en el esquema. En contraste DTD definen un modelo cerrado, lo que significa que un documento no puede contener contenido adicional excepto que sea explícitamente contenido en el DTD.

- Tipos de datos para cada elemento o atributo
  Los esquemas XML permiten especificar tipos de datos de los elementos o atributos. Los tipos de datos indican el formato de los datos, provistos para las validaciones de tipos por los parser XML, y posibilitan procesamiento específico para los tipos de datos en DOM. Los DTD no soportan tipos de datos.

- Extensible
  Los esquemas XML son extensibles, esto significa que se pueden construir esquemas personalizados a partir de esquemas estándares. Por ejemplo si hay una definición para el elemento <dirección> en un esquema, y esta definición cumple las necesidades par aun elemento <dirección> de mi esquema. A través de los namespaces ese esquema puede ser usado directamente con una referencia al esquema original, en vez de copiar su definición.
  Solo un documento DTD puede ser asociado a un documento XML.

- Aplicar a elementos individuales
  Los esquemas XML son aplicados a elementos específicos. Generalmente se querrá aplicar un esquema al elemento raíz de un documento XML para definir la gramática de todo el documento, de todos modos, se puede querer aplicar un esquema XML a un elemento para definir una gramática especial a una porción del documento XML. Esto difiere de DTD ya que solo se puede aplicar un DTD al documento en su totalidad.

- Sintaxis XML
  Los esquemas XML usan sintaxis regular en vez de la sintaxis no XML usada en los DTD. Una de las consecuencias importantes es que se puede acceder al esquema XML con DOM y acceder mediante un programa a las reglas gramaticales con un script.

15.6 XPATH - XML Path Language
W3C Recommendation 16 November 1999

Xpath es el resultado del esfuerzo para una semántica y sintaxis común para funcionalidades compartidas entre Transformaciones XSLT y Xpointer (lenguaje para ser usado como un identificador de un fragmento para una referencia URI que localiza un recurso, junto con Xpath soporta búsqueda en las estructuras internas de documentos XML, permite elegir partes internas basadas en varias propiedades).

El propósito primario es localizar partes de un documento XML. Además provee facilidades para manipular string, números y booleanos.

Xpath opera en una estructura abstracta y lógica del documento XML en vez de realizarlo en su sintaxis superficial. Xpath toma su nombre del uso de la notación path, como en la navegación de URL, que realiza a través de la estructura jerárquica de un documento XML.

Además está diseñado para que pueda ser usado para testear si un nodo se corresponde con su patrón, esta funcionalidad se describe en XSLT:

Xpath modela un documento como un árbol de nodos. Xpath opera sobre un documento XML como un árbol. Este árbol contiene nodos, existen siete tipos de nodos:

- Nodos raíz (root nodes)
- Nodos elementos
- Nodos de texto
- Nodos de atributos
- Nodos de namespaces
- Nodos de instrucción de procesamiento
- Nodos de comentarios.

Xpath define una forma de calcular un valor de string para cada tipo de nodo. Algunos nodos poseen nombres característicos. Soporta los XML namespaces, luego el nombre de un nodo es modelado como un par consistente en una parte local y un posible URI de un namespace (que puede ser Null)

La construcción primaria de Xpath es la expresión. Por cada tipo de nodo existe una forma de determinar el valor del string del tipo de nodo. Una expresión es evaluada para producir un objeto el cual puede ser de los siguientes tipos básicos:

- Conjunto de nodos (colección ordenada de nodos sin duplicados)
- Booleano (True, false)
- Número (de punto flotante)
- String (secuencia de caracteres UCS)

La evaluación ocurre con respecto a un contexto.
Una clase importante de expresión es el path de localización, el cuál selecciona un conjunto de nodos relativos a un contexto de un nodo. El resultado de evaluar una expresión es un conjunto de nodos que contienen los nodos seleccionados por el path de localización.

Ejemplos simples de expresiones Xpath sobre la lista de mensajes:

```
- <lista>
   - <mensaje prioridad="Media">
```

```
      <fecha>2002-11-12</fecha>
      <para>usuario1@adinet.com.uy</para>
      <de>usuario2@adinet.com.uy</de>
      <asunto>Tesis</asunto>
      <texto>Reunión viernes 10 a.m.</texto>
    </mensaje>
  - <mensaje prioridad="Alta">
      <fecha>2002-11-29</fecha>
      <para>usuario2@adinet.com.uy</para>
      <de>usuario1@adinet.com.uy</de>
      <con_copia>tutor@adinet.com.uy</con_copia>
      <asunto>Defensa de la Tesis</asunto>
      <texto>Reunión lunes 14 p.m.</texto>
    </mensaje>
  </lista>
```

a) Selección del destinatario del segundo mensaje del documento:

/lista/mensaje[2]/para

b) Selecciona todas los mensajes donde el atributo prioridad es “Alta” (los atributos se distinguen de los elementos por ser precedidos por un “@”) :

/lista/mensaje[@prioridad=’Alta’]

c) Selecciona todas los mensajes donde el emisor es “usuario1@adinet.com.uy”:

/lista/mensaje[de=’usuario1@adinet.com.uy’]

d) Utiliza la función string-length() para seleccionar aquellos mensajes que tienen menos de 21 caracteres de largo en el elemento texto
/lista/mensaje[string-length(texto)<20]

15.7 XML-QL

XML-QL es un lenguaje de consulta propuesto por la W3C, como una solución a la extracción de datos de un documento XML de gran volumen, para integrar diferentes datos XML provenientes de fuentes distintas y para transportar grandes cantidades de datos XML a un cliente o enviar una consulta a una fuente XML.

Este lenguaje cuenta con un constructor WHERE-CONSTRUCT, y toma características de lenguajes de consulta desarrollados recientemente para datos semiestructuradas.

XML-QL utiliza patrones de elementos para emparejar datos en un documento XML. Para dar una idea sobre como realizar una consulta en este lenguaje, considérese el siguiente ejemplo, extraído de la especificación de la W3C: el documento de entrada se encuentra en www.a.b.c/bib.xml, el que contiene entradas bibliográficas que conforman el siguiente DTD:

```
<!ELEMENT libro (autor+, titulo, editorial)>
<!ATTLIST libro anio CDATA>
<!ELEMENT articulo (autor+, titulo, anio?,
(versioncorta|versionlarga)>
<!ATTLIST articulo tipo CDATA>
<!ELEMENT editorial (nombre, direccion)>
<!ELEMENT autor (primernombre?, apellido)>
```

La siguiente consulta en XML-QL produce todos los autores de libros cuya editorial es Addison-Wesley en el documento www.a.b.c/bib.xml:

```
WHERE <libro>
                <editorial><nombre>Addison-Wesley</></>
                <titulo>$t</>
                <autor>$a</>
        </>
CONSTRUCT $a
```

Esta consulta empata cada elemento <libro> en el documento XML que se encuentra en www.a.b.c/bib.xml que tiene al menos un elemento <titulo>, un <autor> y una <editorial> cuyo elemento <nombre> sea igual a Addison-Wesley. Para ello se definen las variables $t y $a para cada par de <titulo> y <autor>. El resultado es la lista de autores.

15.7.1 Gramática de XML-QL

Para finalizar la definición de este lenguaje, se desea mostrar la gramática del mismo, extraída de la propia especificación.

```
XML-QL ::= (Function | Query) <EOF>
Function ::= 'FUNCTION' <FUN-ID> '(' (<VAR>(':' <DTD>)?)* ')' (':' <DTD>)?
        Query
        'END'
Query ::= Element | Literal | <VAR> | QueryBlock
Element ::= StartTag Query EndTag
StartTag ::= '<' (<ID> | <VAR>) SkolemID? Attribute* '>'
SkolemID ::= <ID> '=' ('"' <STRING> '"' | <VAR>)
EndTag ::= '<' /<ID>?'>'
Literal ::= <STRING>
QueryBlock ::= Where Construct ('{' QueryBlock '}')*
Where ::= 'WHERE' Condition (',' Condition)*
Construct ::= OrderedBy? 'CONSTRUCT' Query
Condition ::= Pattern BindingAs* 'IN' DataSource | Predicate
Pattern ::= StartTagPattern Pattern* EndTag
StartTagPattern ::= '<' RegularExpression Attribute* '>'
RegularExpression ::= RegularExpression '*' |
                      RegularExpression '+' |
                      RegularExpression '.' RegularExpression |
                      RegularExpression '|' RegularExpression |
                      <VAR> |
                      <ID>
BindingAs ::= 'ELEMENT_AS' <VAR> |<CONSTANT> <VAR>
Predicate ::= Expression OpRel Expression
Expression ::= <VAR> | <CONSTANT>
OpRel ::= '<' | '<=' | '>' | '>=' | '=' | '!='
OrderedBy ::= 'ORDERED-BY' <VAR>+
DataSource ::= <VAR> | <URI> | <FUN-ID>(DataSource (',' DataSource)*)
```

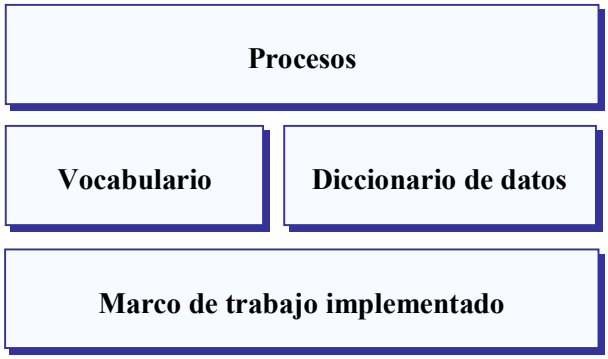
16. Estándares para la comunidad de comercio

La explosión del comercio electrónico y las nuevas tecnologías ocasionan que varias compañías e industrias hayan promovido iniciativas para definir como trabajar en conjunto formando redes o comunidades de comercio con socios de negocio.
El objetivo primario de las iniciativas es perfilar las interacciones entre socios de comercio eliminando redundancias y mejorando el flujo de información crítica. El primer paso hacia lograr este objetivo es establecer estándares para usar entre los socios en las redes de comercio para intercambiar información.

El alcance de los estándares de b2b muchas veces es subestimado por las empresas que buscan participar en las comunidades de comercio. Muchas creen que los estándares involucran solo la creación de definiciones de tipos de documentos XML (DTD o XML Schemas) para las transacciones relevantes. Pero solo el vocabulario XML es un aspecto de los estándares.

16.1 Componentes a estandarizar

La siguiente figura detalla los componentes que se requieren estandarizar para establecer redes de comercio eficientes.



El marco de trabajo implementado describe el protocolo (detalles técnicos de conectividad) para aplicaciones.
Vocabulario define la información de negocio que fluye entre las organizaciones.
Diccionario de datos definen los rangos de valores de datos compartidos para elementos enumerados.
Procesos estándares definen las interacciones entre socios de comercio.

Se detallan algunos puntos a tener en cuenta con respecto al alcance de los estándares:

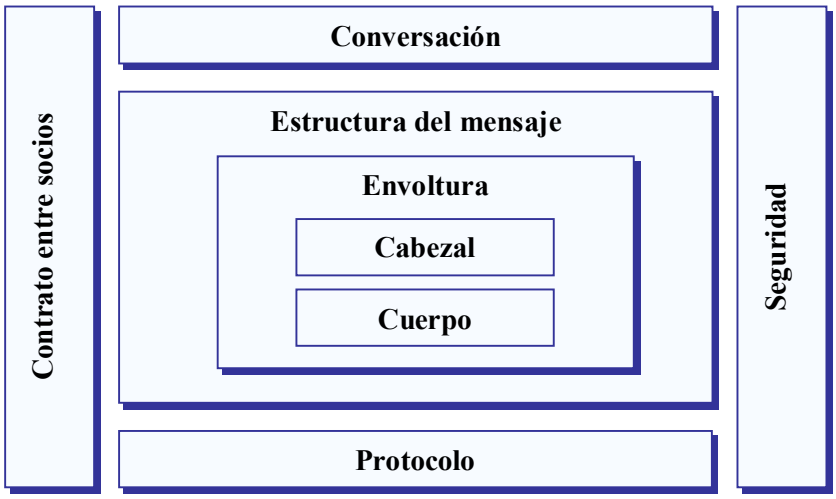
- Mientras que las relaciones entre organizaciones de pequeña escala pueden desarrollarse especificando estándares de vocabularios y diccionarios de datos, esto no alcanza cuando nos referimos a redes de comercio de mediando y gran tamaño. En estas últimas se necesitan definir procesos e implementar marcos de trabajo.

- Para el suceso de cualquier estándar, y luego de las redes de comercio, ser requiere una especificación bien detallada. La ambigüedad incrementa la complejidad y causa incompatibilidades entre socios si cada uno utiliza una implementación diferente referida a un elemento vagamente especificado.

- El vocabulario y el diccionario de datos generalmente no un elemento dificultoso o controversial de crear. Se puede crear un nuevo estándar en la base de ejemplos existentes como EDI y estándares basados en XML.

16.2 Componentes del marco de trabajo implementado

Especifica los detalles técnicos de cómo la información de negocio es pasada entre los socios de negocio. La siguiente figura muestra los componentes del marco de trabajo implementado.



Cada componente define un aspecto particular de la infraestructura tecnológica que permite que dos socios de negocio intercambie en forma segura y confiable, rutee, entienda y procese la información de negocio.
Se detallarán cada uno de ellos para ver la importancia que tiene para establecer la interoperabilidad.

16.2.1 Protocolo

Es el más bajo nivel del marco, especifica el contrato a nivel de redes. Ejemplos pueden ser HTTP, SMTP, etc. Varias implementaciones ofrecen múltiples opciones de protocolos.

16.2.2 Estructura del mensaje

Existen varios estándares para estructurar y empaquetar la información intercambiada. Separados en mensajes estos paquetes consisten en tres porciones.

Cuerpo: el portador de la información de negocio referida a la transacción.
Cabezal: Contenido del manifiesto del cuerpo del mensaje, contiene información de seguridad, instrucciones de procesamiento, y ruteo de información.
Envoltura (envelope): es el contenedor que vincula los componentes y contiene información de identificación.

16.2.3 Conversación

Es el componente que detalla como la especificación de cómo es software utiliza la información del cabezal del mensaje para asegurar la confiabilidad de la conectividad entre socios. Define la estructura y el uso de control crítico de mensajes, como la secuencia de mensajes y el acuse de recibo de un mensaje recibido y errores.
La conversación es confundida con el proceso de negocio, debe quedar claro que se refiere solo a características técnicas. Permite un diálogo técnico común para los socios de negocio de la red para todos los procesos de negocio.

16.2.4 Seguridad

La seguridad y integridad de la información en los mensajes es muy importante. El marco define exactamente las medidas de seguridad que están permitidas y como son aplicadas. Incluyen requerimientos para el encriptado de información, autenticación, no repudio y autorización.

16.2.5 Contratos entre socios

Definen la estructura de la información requerida para establecer el vínculo electrónico con los socios de negocios. La complejidad de mismo es conducida por la precisión en la definición del marco. EbXML y UDDI están actualmente desarrollando estándares para estos contratos. (trading partner agreement).

16.3 Vocabulario provee sintaxis

Es el componente más comúnmente discutido. Este imparte significado sintáctico al componente del cuerpo del mensaje. Define la información contenida en el mensaje, esto posibilita parsear los mensajes, validarlos y que sean comprendidos por los destinatarios.

Los vocabularios son expresados en dos formas, descripciones entendibles por humanos y esquemas legibles por aplicaciones. El primero provee una representación de alto nivel del contenido del mensaje y las representación de esquemas describen explícitamente la sintaxis de los mensajes.

Varios esquemas existen, si elegimos XML se pueden usar DTD o XML Schema.

16.4 Diccionarios de datos proveen semántica

Mientras los vocabularios definen la estructura de la información de negocio y proveen significado sintáctico, el diccionario de datos que provee semántica a los elementos, atributos y variables. Para los elementos de los esquemas donde los valores pueden ser enumerados es el diccionario de datos que incluye los posibles valores. Ejemplos pueden ser la carta de colores de productos. DUNS (Data Universal Numbering System) y GTINs (Global Trade Item Number), como un candidatos a incluir en un diccionario.
Es crítico el acuerdo sobre los diccionarios entre los socios de negocio tanto como el de los vocabularios para entender la información contenida en los mensajes.

16.5 Procesos posibilitan funciones de negocios

Los procesos estándares especifican las actividades, decisiones y roles para cada socio de negocio para cumplir un conjunto de funciones de negocio. Individualmente una función consiste en el intercambio de múltiples mensajes que resultan en múltiples transacciones.
A nivel de las aplicaciones se controlan los mensajes y acuses de recibo como parte de la definición de un proceso. Conceptos avanzados como meta modelos de procesos para asegurar que todas los participantes hablen el mismo vocabulario de proceso, y un repositorio de procesos para acceso dinámico a la última versión de los acuerdos sobre procesos, han comenzado a aparecer en estándares como ebXML.
Existen varios estándares que incluyen implícitamente en vez de explícitamente las definiciones de proceso. De todos los estándares existentes solo RosettaNet incluye una completa especificación de procesos a través de la definición de los Partner Interchange Process (Pips que veremos cuando describamos RosettaNet).

16.6 Evolución de los estándares

El primer estándar de comercio electrónico ampliamente adoptado fue EDI en los 80 (Electronic Data Interchange), estaba orientado a facilitar el intercambio de información entre empresas sin intervención

humana, siempre que los participantes se hayan puesto de acuerdo sobre el formato de los datos. Esto puede ocurrir entre dos empresas específicas o para un sector completo.

Desarrollar estándares para sectores específicos tiene la ventaja que se reduce sustancialmente la necesidad de desarrollo a medida, pero los estándares desarrollados de forma independiente evidentemente no son compatibles entre sí, por esa razón surgió el estándar EDIFACT (EDI for Administration, Commerce and Transport) que pretender establecer las reglas y un vocabulario en la que basar la descripción de los objetos de negocio para mantener así compatibilidad entre diferentes sectores. Aunque supone un paso adelante, queda un factor de coste muy importante: el coste de las comunicaciones de la red propietaria a la que todas las empresas necesitaban estar conectadas, con lo cual el uso de estas tecnologías ha sido un lujo solamente al alcance de las grandes compañías.

Con la introducción de XML se desarrolla a finales de los años noventa la fusión entre XML y EDI que convenientemente lleva el nombre de XML/EDI, un marco de trabajo compatible con las infraestructuras EDI existentes que sustituye los mensajes EDI por mensajes XML más una serie de elementos de infraestructura. Las ventajas se deben fundamentalmente al carácter autodescriptivo de XML, la estructuras del lenguaje y procesos de negocio ya existentes bajo EDI se aprovechan y al poder expresarlas en XML se añaden posibilidades derivadas de este hecho como poder buscar en estos documentos, introducir referencias, autodescripción de servicios, etc.

En este momento las iniciativas en el comercio electrónico se multiplican y nacen tanto líneas desvinculadas de fabricantes concretos como **ebXML** o **RosettaNet,** como propietarias empujadas por empresas como Comerse One **xCBL**[8] o Ariba **cXML**[9], centradas en productos de comercio electrónico, que poseen estándares propietarios con el fin de acelerar la adopción del comercio electrónico a gran escala y tomar una posición dominante en el mercado.

A continuación se detallan algunos de los estándares no propietarios más relevantes, información sobre organizaciones que apoyan el uso y las estandarizaciones de XML (W3C y OASIS), y la estandarizaciones XML de los web services.

[8] xCBL: Common Business Language es una especificación creada por la empresa Comerse One consiste en vocabularios y un marco de trabajo definido para intercambios entre socios de negocios que utilizan los productos Comerse One MarketSite.

[9] cXML: Creado por la empresa Ariba comprende un vocabulario y un contenedor de mensajes simple para el intercambio de documentos entre socios de negocio que utilicen los productos y redes Ariba

17. ebXML

En 1999 se inició el proyecto Electronic Business XML (ebXML) con el ambicioso objetivo de crear un mercado global electrónico único. Una iniciativa patrocinada por el United Nations Centre for Trading Facilitation en Electronic Business (UN/CEFACT, http://www.unece.org/cefact/) y la Organization for the Advancement of Structured Information Standards (OASIS, http://www.oasis-open.org) aportando el UN/CEFACT su experiencia sobre procesos de estandarización (son los autores del estándar B2B EDIFACT) y OASIS know-how sobre XML.

El objetivo de ebXML es:

"La visión de ebXML es la creación de un mercado electrónico global en el que puedan contactar empresas de cualquier tamaño y localización geográfica para llevar a cabo negocios mediante el intercambio de mensajes XML"

17.1 Elementos básicos para transacciones bajo ebxml

Veamos los mecanismos básicos para el caso más simple (dos empresas):

Para intercambiar información ambas utilizarán el **ebXML Messaging Service** como servicio que les resuelve la problemática de transporte de la información entre ellas. Los tipos de mensajes que han de intercambiarse y la forma de hacerlo, es decir, qué forma tienen los procesos de negocio estará contenida en el **Collaboration Protocol Agreement (CPA)** que es el convenio que las partes del negocio tienen que establecer para poder intercambiar datos. El CPA contiene la información para configurar correctamente el software que utiliza cada una de las partes para el intercambio electrónico de la información.

La condición para que esta negociación se pueda realizar de forma efectiva es la existencia de un elemento central en el que las diversas empresas afiliadas a un proveedor de servicios ebXML puede acudir para obtener la información de interés sobre potenciales socios de negocio. Este elemento central es el **ebXML Registry**, un repositorio en el cual se pueden depositar cualquier tipo de información ebXML para ser accesible a los posibles interesados.
El registry contiene, por ejemplo, descripciones de documentos como una factura en forma de DTDs o esquemas XML.

En este registro se publican los servicios que ofrece una empresa, en los términos de ebXML, el **Collaboration Protocol Profile (CPP).**

17.2 Un ejemplo práctico

Para dar forma a las ideas anteriores, se verá un ejemplo simple, una empresa dedicada a la fabricación de telas que quiere participar en un entorno ebXML:

- El primer paso que ha de dar es comprar el software necesario o adaptar sus sistemas existentes añadiendo los módulos correspondientes para la exportación / importación de información conforme a ebXML.

- Luego debe describir su perfil de negocio mediante una especificación **CPP**, supongamos para ello que básicamente especifica dos actividades: la compra de hilo y la venta de tela por metraje

- Con esta información de base debe completar la información para el CPP con la especificación del intercambio de mensajes (recordemos: en forma de esquemas XML o DTDs) relacionados con los procesos que soporta, es decir en el caso de la venta:

- Un cliente solicita una oferta según el formato definido a tal efecto (o bien uno general al que se adhiere nuestra empresa o un formato que ésta especifica explícitamente para este propósito), el formato estará definido en un documento XML del tipo DTD o XML Schema, por ejemplo: SolicitudOferta.xsd en el caso de utilizar XML Schema.

- Nuestra empresa envía su oferta (Oferta.xml) al cliente conforme al esquema Oferta.xsd

- El cliente está satisfecho y decide hacer un pedio que especifica en un mensaje con un documento XML conforme al esquema Pedido.xsd.

- Nuestra empresa entrega la mercancía y emite la factura Factura.xml conforme al esquema Factura.xsd

En resumen, el CPP contiene la descripción del perfil de actividad de nuestra empresa (compra de hilo y venta de tela) más la especificación del proceso de venta que contiene la información de qué mensajes intercambiar, su secuencia de intercambio y los formatos especificados en nuestro caso con esquemas XML.

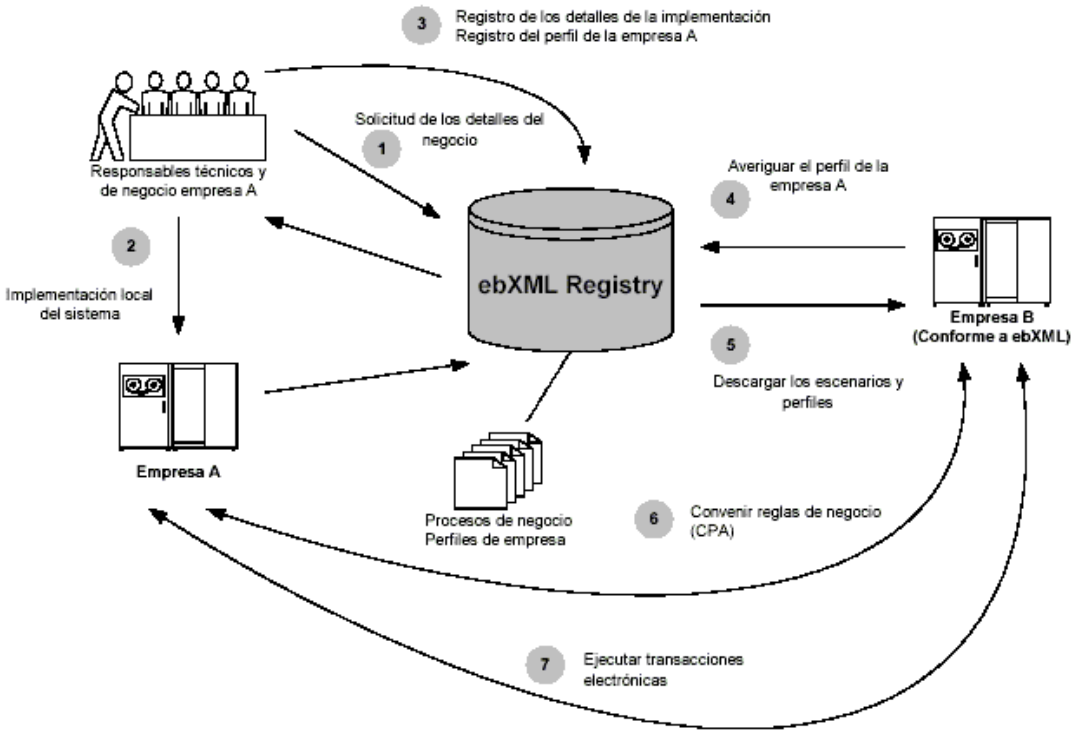
Es importante hacer resaltar el hecho que no necesita especificar sus propias definiciones, sino que puede utilizar las existentes. De hecho, la reutilización de estas es muy deseable con el fin de simplificar la participación en un entorno ebXML lo más posible y hacerla más asequible para empresas medianas y pequeñas que no pueden invertir en esfuerzos como los requeridos para la definición de estos documentos.

Ahora, nuestra empresa existe en el universo ebXML y ofrece públicamente la información necesaria para ejecutar negocios a nivel electrónico con ella. Con lo cual los posibles clientes o proveedores interesados en hacer negocios con ella pueden acudir al ebXML Registry y localizarla al buscar empresas con sus actividades.

Un posible interesado dispondrá de su propio CPP y querrá relacionarlo de alguna manera con el que ha especificado nuestra empresa. Es decir, ambas empresas tienen que asegurar que entienden lo mismo por conceptos como Pedido o Factura. En el caso más simple las dos CPP serán compatibles, si ambas empresas ha acudido ha utilizar descripciones estándar en vez de definir las suyas propias este caso puede ser relativamente probable, sino será muy difícil que a priori haya una compatibilidad.

Ante la ausencia de compatibilidad, empieza una negociación para convenir un subconjunto compatible o transformaciones necesarias. Si se llega a un acuerdo, éste se recoge en una CPA, que es el acuerdo de un protocolo de colaboración entre ambas empresas. Ahora sí pueden comenzar las transacciones electrónicas.

La siguiente figura muestra el escenario en el cual la empresa A quiere participar en un entorno ebXML, para ello debe primero dar los pasos para integrarse en el entorno. A partir de ahí otros como la empresa B serán capaces de localizarla y hacer negocios con ella.



17.2.1 Estructura general de un CPA

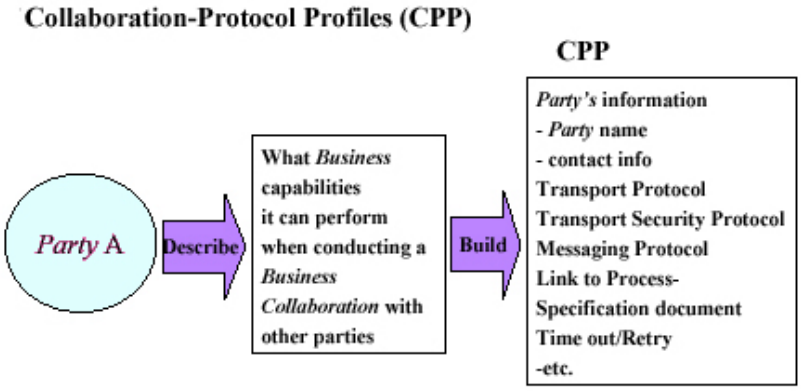
Para el intercambio de información entre dos organizaciones se requiere que cada una de ellas conozca que colaboraciones de negocio (Business Collaborations) soportan, el rol que cumple y los detalles tecnológicos de cómo envía y recibe mensajes.
La forma de que dos partes intercambien información sobre el contexto del negocio es el Collaboration Protocol Profile (CPP).

Los acuerdos entre ambas partes son expresados en el Collaboration Protocol Agreement (CPA)

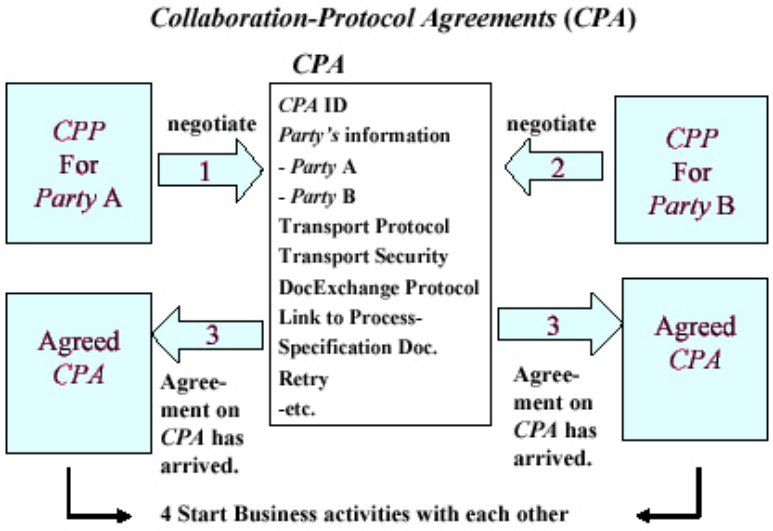
Los CPP se almacenarán en el Registro ebXML para que otras partes puedan buscarlos y realizar negociaciones con ellos.
El documento que define la interacciones entre dos partes es la especificación del proceso (Process Specification) que está de acuerdo a ebXML Business Process Specification Schema (ebBPSS)

El CPP y el CPA incluyen referencias a documentos de especificaciones de procesos (Process Specification).
La relación entre un CPP y los documentos de especificaciones de procesos (Process Specification), identifican que negocios (business collaboration) pueden realizar, y están contenidos en el elemento PartyInfo del CPP.
Un CPP describe las capacidades individuales de una organización (Party).
Un CPA describe las capacidades que dos partes han acordado a realizar para un negocio (Business Collaboration). Un CPA de fine las condiciones y términos de tecnología de información que posibilitan que sean intercambiado los documentos de negocio.

La siguiente figura muestra como se crea un CPP, la organización A (Party A) construye un CPP y lo ingresa la repositorio ebXML.



En la siguiente figura se muestran dos compañías (Party A y B) que se van a relacionar a través de un acuerdo de negocio, construyen un CPA calculando la intersección de la información de sus CPP. El CPA resultante define como ambas partes se comportarán cuando realicen sus negocios (Business Collaboration).



En especial nos interesa la composición del acuerdo de negocio, CPA, por lo que se va a describir las partes que componen este documento XML.

La siguiente es la estructura básica de un CPA:

```
<CollaborationProtocolAgreement
  xmlns:tp=http://www.oasis-open.org/committees/ebxml-cppa/schema/cpp-cpa-2_0.xsd
  xmlns:ds=http://www.w3.org/2000/09/xmldsig#  xmlns:xlink=http://www.w3.org/1999/xlink
  tp:cpaid="YoursAndMyCPA" tp:version="2.0a">
     <tp:Status tp:value="proposed"/>
     <tp:Start>1988-04-07T18:39:09</Start>
     <tp:End>1990-04-07T18:40:00</End>
     <!-- ConversationConstraints MAY appear 0 or 1 time -->
    <tp:ConversationConstraints tp:invocationLimit="10" tp:concurrentConversations="4"/>

     <tp:PartyInfo>
        ...
```

```
    </tp:PartyInfo>
    <tp:PartyInfo>
      ...
    </tp:PartyInfo>

    <tp:SimplePart tp:id="..."> <!-- one or more -->
      ...
    </tp:SimplePart>

    <tp:Packaging tp:id="..."> <!-- one or more -->
      ...
    </tp:Packaging>

    <tp:Signature> <!-- zero or one time -->
      ...
    </tp:Signature>
    <tp:Comment xml:lang="en-GB">any text</Comment> <!--zero or more -->

</tp:CollaborationProtocolAgreement>
```

El elemento raíz de un documento CPP es < CollaborationProtocolAgreement > contiene el atributo cpaid que lo identifica únicamente.

Elemento <Status>:

Es el estado del proceso de composición / negociación del CPA

Elementos <Start> y <End>:

Determinan el tiempo de vida del contrato.

Elemento <ConversationConstraints>:

Limita el número de conversaciones del CPA. Posee los atributos invocationLimit y concurrentConversations este último indica la cantidad de conversaciones que puede llevarse a cabo en el mismo tiempo.

Elemento <PartyInfo>:

Describe las partes involucradas en los términos del acuerdo de la colaboración de negocio definidos en el documento de especificación del proceso referenciado en el CPP. Contiene el atributo partyName que indica el nombre de la organización.

Incluye los siguientes elementos:

Partid: que provee una identificación lógica para la organización.

PartyRef: provee un puntero a mayor información sobre la organización

CollaborationRole: identifica el rol de la organización en el contexto y las especificaciones de procesos asociadas con el rol.

Certificate: identifica el certificado usado por la organización en las funciones de seguridad.

SecurityDetails: identifica las certificados y políticas de seguridad usadas.

DeliveryChannel: define las características utilizadas para el envió y recepción de mensajes que puede utilizar la organización. (por ejemplo HTTP) y el protocolo de mensajes (por ejemplo ebXML message service). Contiene tres elementos: Transport (que contiene la identificación del transporte), DocExchange (contiene el identificador del tipo de documentos de intercambio) y MessagingCharacteristics (contiene los atributos que definen el modo de respuesta, si requiere un acuse de recibo, si requiere que el acuse de recibo sea firmado, si elimina duplicados)

Transport: define las características del protocolo de transporte para enviar y recibir mensajes, por ejemplo, consideraciones de seguridad, protocolos de transporte, etc..

DocExchange: identifica las características de los mensajes intercambiados, como protocolos de firma y encriptación.

OverrideActionMshBinding: especifica el canal de entrega para la comunicación asincrónica de mensajes.

Elemento <SimplePart>:

Este elemento provee una lista repetitiva de partes que contiene identificadores de los valores de tipos de los contenidos MIME. Contiene un atributo id es utilizado para referenciar esta parte del mensaje, y el atributo mimetype especifica el valor del tipo de contenido del mensaje (por ejemplo, text/plain, application/xml, etc.)

Elemento <Packaging>

El subárbol del elemento Packaging provee información específica sobre como el cabezal del mensaje y el cuerpo son empaquetados para ser transmitidos sobre el transporte, incluye información de nivel de seguridad. Además contiene típicamente como se procesan las partes MIME, XML namespaces, parámetros de seguridad y estructura MIME de los datos intercambiados por las partes.

Elemento <Signature>

Este elemento posibilita que el CPA sea firmado digitalmente usando tecnología acorde a la especificación de firma digital XML [XMLDSIG].
Puede contener uno o más elementos Signature, la estructura de los mismos respeta la especificación de firma digital de XML.

Un documento CPA es firmado digitalmente por uno o más partes (parties) para significar que se asegure su integridad tanto como que lo que expresa el acuerdo.
Si firman dos partes una de ellas firma y la segunda firma sobre la primera firma.
Si la validación de alguna de las firmas falla se debe considerar al CPA como inválido.

Elemento <Comment>

Puede contener uno o más comentarios, que son notas de texto.

17.3 Otros elementos en la interacción entre entidades

Además de los componentes fundamentales de una infraestructura ebXML, el ebXML Registry, el CPP de cada empresa y el CPA entre ambas, existen otra serie de elementos importantes poder llevar a cabo transacciones correctamente.

17.3.1 Mensajes en ebXML

Una vez convenido cómo se debe estructurar la información intercambiada, hay analizar la problemática relacionada con el trasporte de la misma entre las entidades que participan en el negocio. Para ello existe el servicio de mensajería que se responsabiliza del transporte, enrutado y empaquetado de la información, un protocolo que representa una ampliación de SOAP 1.1, y por tanto es capaz de utilizar la infraestructura de Internet mediante la utilización de HTTP, SMTP, etc.

Este servicio también abarca las cuestiones relativas a la seguridad, los llamados Security Services abarcan la generación y verificación de firmas digitales, la autenticidad y autorización relativa a un mensaje. Además existe soporte para garantizar la fiabilidad del intercambio de los mensajes y el tratamiento de errores.

17.3.2 Core Components, componentes de negocio reutilizables

Los tipos de documentos intercambiados (los documentos de negocio, Business Documents) pueden ser documentos definidos previamente por otras entidades, aquí entran lógicamente sobre todo documentos relativamente simples y uniformes en sus formatos como lo puede ser una factura. Si algún tipo de documento no corresponde a las necesidades o bien han de modificarse los tipos existentes o se deben crear nuevos tipos válidos.

Para este fin, ebXML define una serie de componentes básicos (Core Components) con diferentes niveles de complejidad. En el nivel más básicos son comparables con el concepto informático de tipo de datos, por ejemplo un componente nombre que debe ser una cadena de caracteres alfanuméricos, éste será a su vez componente en estructuras más complejas como una dirección o una cuenta bancaria. Los componentes disponen de información de contexto que se refiere precisamente a esta cuestión: dónde se engloban y con qué papel, de esta manera se pueden agregar para crear objetos de información de mayor envergadura.

La idea fundamental de esta parte de ebXML es maximizar la reutilización de elementos predefinidos y llegar así a un compromiso óptimo entre las necesidades de personalización que requiere una relación comercial entre dos empresas y el esfuerzo requerido para definirla.
Resulta, además, más sencillo estandarizar tipos de documentos de mayor nivel si sus componentes básicos se encuentran estandarizados, es decir, es más fácil empezar ponerse de acuerdo cómo debe ser el formato de una fecha, determinadas medidas, métricas, etc. y luego basar la estructura de un pedido en ello que viceversa.

17.3.3 Procesos de negocio

Definidos los documentos de negocio con la ayuda de los Core Components han de definirse los procesos que los manejan, para ello, se utiliza la metodología UMM (UN/CEFACT Modeling Methodology) que a su vez se apoya UML (Unified Modeling Language), como lenguaje de modelado y la plasma luego en XML.

El ejemplo siguiente muestra un documento de especificación de procesos, se expone una parte de la especificación de proceso relacionada con respecto a un catálogo de productos y pedido. Al principio se referencian los documentos de negocio implicados, (Catalog Request, Catalog, Purchase Order, ...). Lo siguiente es la descripción de los participantes del proceso, el cliente (Customer) asume aquí dos papeles: solicitando de información (Requestor) y como comprador (Buyer), otros participantes son el vendedor, el proveedor de los productos y un proveedor para un crédito.
Las transacciones especificadas son la petición de catálogo y la creación del pedido, otras transacciones no mostradas en el extracto del código del ejemplo son la confirmación de envío de la mercancía, confirmación del pago, etc.

```
<!DOCTYPE ProcessSpecification SYSTEM "ebXMLProcessSpecification-v1.01.dtd">
<ProcessSpecification name="Simple" version="1.1" uuid="[1234-5678-901234]">
<!-- Business Documents -->
<BusinessDocument name="Catalog Request"/>
<BusinessDocument name="Catalog"/>
<BusinessDocument name="Purchase Order"/>
<BusinessDocument name="PO Acknowledgement"/>
<!-- ... -->
<Package name="Ordering">
<!-- First the overall MultiParty Collaboration -->
<MultiPartyCollaboration name="DropShip">
<BusinessPartnerRole name="Customer">
<Performs initiatingRole="requestor"/>
<Performs initiatingRole="buyer"/>
<Transition fromBusinessState="Catalog Request" toBusinessState="Create Order"/>
</BusinessPartnerRole>
<BusinessPartnerRole name="Retailer">
<Performs respondingRole="provider"/>
<Performs respondingRole="seller"/>
<Performs initiatingRole="Creditor"/>
<Performs initiatingRole="buyer"/>
<Performs initiatingRole="Payee"/>
<Performs respondingRole="Payor"/>
<!-- ... -->
</MultiPartyCollaboration>
<!-- Now the Binary Collaborations -->
<BinaryCollaboration name="Request Catalog">
<InitiatingRole name="requestor"/>
<RespondingRole name="provider"/>
<BusinessTransactionActivity name="Catalog Request" businessTransaction="Catalog Request"
fromAuthorizedRole="requestor" toAuthorizedRole="provider"/>
</BinaryCollaboration>
<BinaryCollaboration name="Firm Order" timeToPerform="P2D">
<Documentation>timeToPerform = Period: 2 days from start of transaction</Documentation>
<InitiatingRole name="buyer"/>
<RespondingRole name="seller"/>
<BusinessTransactionActivity name="Create Order" businessTransaction="Create Order"
fromAuthorizedRole="buyer" toAuthorizedRole="seller"/>
</BinaryCollaboration>
<!-- ... -->
<!-- Here are all the Business Transactions needed -->
<BusinessTransaction name="Catalog Request">
<RequestingBusinessActivity name="">
<DocumentEnvelope isPositiveResponse="true" businessDocument="Catalog Request"/>
</RequestingBusinessActivity>
<RespondingBusinessActivity name="">
<DocumentEnvelope isPositiveResponse="true" businessDocument="Catalog"/>
</RespondingBusinessActivity>
</BusinessTransaction>
<BusinessTransaction name="Create Order">
<RequestingBusinessActivity name="" isNonRepudiationRequired="true"
timeToAcknowledgeReceipt="P2D"
timeToAcknowledgeAcceptance="P3D">
<DocumentEnvelope isPositiveResponse="true" businessDocument="Purchase Order"/>
…
```

18. RosettaNet
(www.RosettaNet.org)

RosettaNet es un organización sin fines de lucro, creada en 1998, de más de 400 compañías integrantes de áreas de tecnología de información, componentes electrónicos, manufactura de semiconductores y proveedores de soluciones, que trabaja para crear, implementar y promover estándares abiertos de procesos de negocios electrónicos. Estos estándares forman un lenguaje común para negocios electrónicos.

Su misión es conducir un rápido desarrollo de estándares de negocio electrónico, crear un lenguaje común, procesos abiertos de negocio que provean beneficios mesurables, los cuáles son vitales para la evolución de las redes de comercio globales y de alta tecnología.

Su lema es ser la "Lingua franca for ebusiness". Su nombre proviene de la piedra Rosetta la cuál describe un mismo mensaje en tres lenguajes que data de 196. A.C. que permitió comprender jeroglíficos egipcios.

Los estándares de RosettaNet ofrecen una solución no propietaria, que integra: diccionarios de datos, un marco de trabajo para la implementación, esquemas de mensajes de negocio basados y especificaciones de procesos en XML. Estos estándares están accesibles sin costo en el sitio de RosettaNet.

Elementos fundamentales:

- Diccionarios:
  - Diccionario de negocios (Business Dictionary)
  - Diccionario técnicos (Technical Dictionary)
- Marco de trabajo de implementación (Implementation Framework (RNIF))
- Partner Interface Proceses (PIPs)

18.1 Diccionarios de RosettaNet

Para que sea posible realizar transacciones electrónicas autónomas, los sistemas deben utilizar el mismo lenguaje.
Así por ejemplo si dos sistemas pretenden intercambiar información de sus respectivas empresas, se podría dar el caso de que uno maneje el concepto de "Razón Social" mientras que el otro maneja el concepto de "Nombre".
La solución consiste en utilizar un diccionario de referencia en el cual se define qué términos han de usarse para qué conceptos, algo que implica evidentemente un trabajo de adaptación al lenguaje definido de los sistemas participantes.
En el ejemplo anterior el diccionario podría definir el término "Razón Social Empresa" de manera que ambos sistemas tendrían que limitarse a utilizar exclusivamente este término en sus transacciones vía RosettaNet, implementando este requisito típicamente con interfaces intermedias entre los sistemas corporativos y el software para RosettaNet en los que se relacionan los conceptos locales contra los términos estandarizados.
RosettaNet distingue entre dos tipos de diccionarios:

- **Diccionarios de negocio**, contiene el conjunto de términos relacionados con las transacciones en sí (por ejemplo, el concepto de cuenta bancaria, factura)

- **Diccionario técnico,** contiene el conjunto de términos para la descripción de los productos y servicios, objeto de las transacciones. (por ejemplo, Information Technology, Electronic components).