Arquitectura de Servicios Web y Marco de Trabajo

Rodrigo Frez

20 Julio de 2009

Temario

- Introducción
- WSDL
- SOAP
- UDDI
- Seguridad
- Aplicación Directa (Enterprise)
  - BPEL
    - Introducción SOA-BPM-BPMS
  - Grid Services
- Aplicación Directa (Development)
  - Ajax
- Buenas Prácticas

Introducción

- La Web está diseñada para interacciones entre aplicaciones y personas
- Sirve muy bien a este propósito:
  - Información compartida: Es una librería de contenido distribuida.
  - Permite realizar comercio electrónico.
  - Pueden exisitr transacciones no automáticas entre negocios.
- ¿Por que pudo esto suceder?
  - Está construida sobre pocos estándares: http + html
  - Modelo de interacción construido realizand pocas suposiciones sobre la plataforma computacional.
  - Ubicuidad: Estar en la Web y poder ser encontrado.

“Servicios Web” es un esfuerzo para construir una plataforma de computación distribuida para la Web.

Características de los Servicios Web

- Objetivos
  - Permitir interoperabilidad universal.
  - Adopción masiva, ubicuidad, rendimiento.
  - Soportar una arquitectura orientado a Servicio (ServiceOrientedArchitecture).
  - Soportar eficientemente ambientes abiertos (Web) y cerrados (corporativos).

Características de los Servicios Web

- Objetivos derivados
  - Conocer con precisión cuales son los Servicios ofrecidos por un proveedor, junto con la descripción de su interfaz
  - Utilizar un estándar que permita comunicar aplicaciones independiente de la plataforma

Características de los Servicios Web

- Objetivos (+ técnico)
  - Tener un protocolo universal para la comunicación entre aplicaciones
  - Basar la comunicación en los protocolos de internet, en especial HTTP
  - Estandarizar los mensajes a enviar, ya sea un documento a procesar remotamente o la invocación de un comando

Características de los Servicios Web

- Requerimientos
  - Basado en estándares.
  - Muy buen soporte es crítico.
  - Se asume una muy pequeña infraestructura.
    - Sólo un pequeño número de estándares debe ser implementado.
  - Se enfoca en MENSAJES y DOCUMENTOS, no en componentes de Software.

Características de los Servicios Web

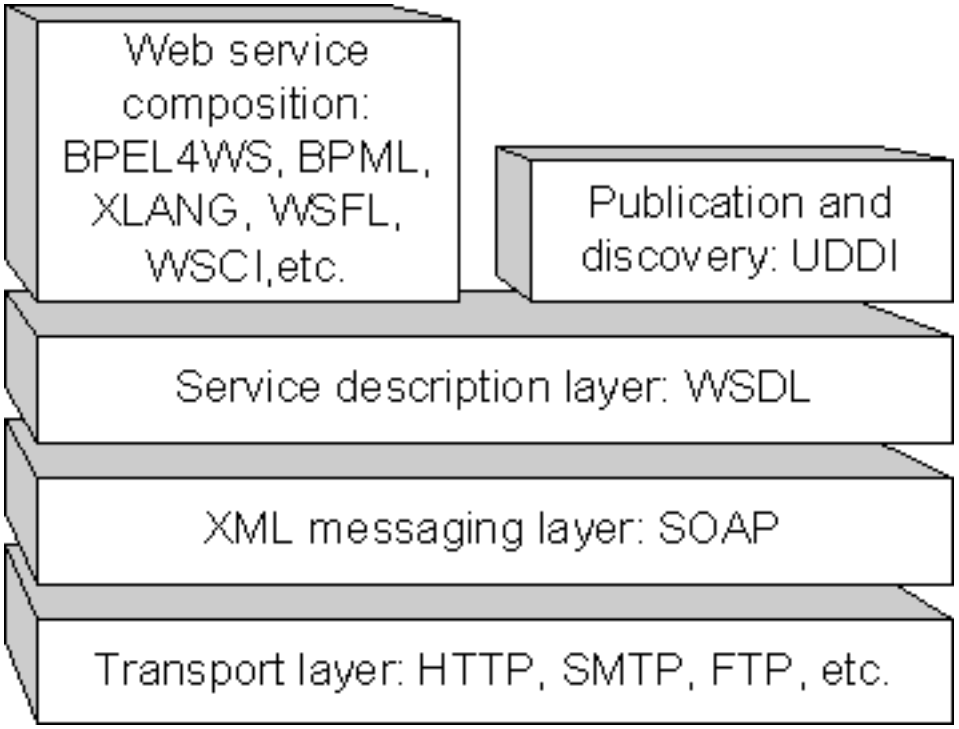
- Servicio Web:
- Es una arquitectura de computación distribuida
- Usa sus propias interfaces, protocolos y servicios de registro, para que distintas aplicaciones de distintas plataformas puedan usar sus procedimientos
- En constante evolución

Las aplicaciones de Servicios Web son encapsuladas, componentes Web bajamente acoplados que se pueden vincular dinámicamente entre sí

Framework de Servicios Web

- Se describe en términos de:
  - Lo que va en el “fierro” o en los “cables”: Formatos y protocolos.
  - Qué describe lo que va en el “cable”: Lenguajes de Descripción.
  - Qué nos permite encontrar estas descripiones: Descubrimiento de Servicios.

Arquitectura de Servicios Web



Que provee la Arquitectura?

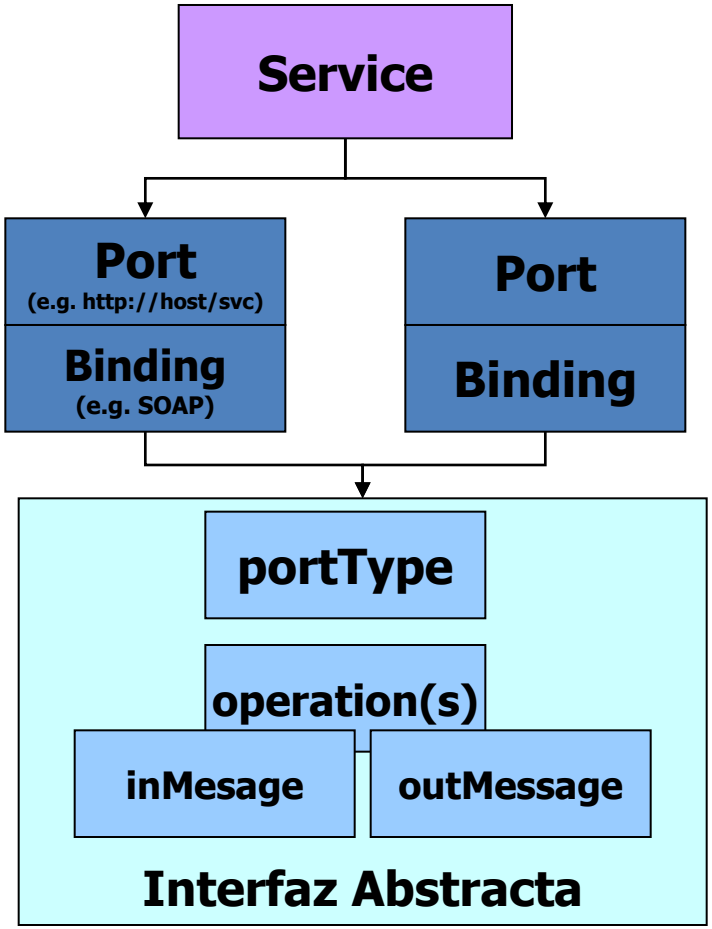
- Encapsulamiento de funcionalidades
- Sistemas débilmente acoplados
- Componentes reutilizables
- Acceso mediante programación
- Montado “sobre” la Web
- Comunicación sobre XML

Arquitectura de Servicios Web

WSDL

- Web Services Description Language
- Es un documento XML que introduce una gramática para describir puntos de emisión de mensajes (endpoints).
- Los elementos que permiten describir “endpoints” son:
  - Messages: Referencias a XML Schema que definen las diferente partes de un Mensaje.
  - Operations: Lista de Mensajes involucrados en un flujo de mensaje. Por ejemplo una operación de request-response contiene dos mensajes.
  - PortType: Es el set de flujos de mensajes (Operations), que se encuentran en un “endpoint”, sin detallar el transporte ni la codificación.
  - Binding: Medio de transporte y codificación particular de un PortType.
  - Port: Dirección en la red de un “endpoint” y el binding que lo caracteriza.
  - Service: Colección de “endpoints” relacionados.

WSDL


Service
Port
(e.g. http://host/svc)
Binding
(e.g. SOAP)
Port
Binding
portType
operation(s)
inMesage
outMessage
Interfaz Abstracta

Introducción : WSDL

```
<?xml version="1.0" ?>
- <definitions name="TranslatorService" targetNamespace="http://www.mindswap.org/2002/services/Translator.wsdl"
    xmlns:ns1="http://www.mindswap.org/2002/services/Translator.wsdl" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns="http://schemas.xmlsoap.org/wsdl/">
  - <message name="TranslatorRequest">
      <part name="inputString" type="xsd:string" />
      <part name="inputLanguage" type="xsd:string" />
      <part name="outputLanguage" type="xsd:string" />
    </message>
  - <message name="TranslatorResponse">
      <part name="return" type="xsd:string" />
    </message>
  - <portType name="TranslatorPortType">
    - <operation name="getTranslation">
        <input message="ns1:TranslatorRequest" />
        <output message="ns1:TranslatorResponse" />
      </operation>
    </portType>
  - <binding name="TranslatorBinding" type="ns1:TranslatorPortType">
      <soap:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http" />
    - <operation name="getTranslation">
        <soap:operation soapAction="" />
      - <input>
          <soap:body use="encoded" namespace="urn:org:mindswap:services:Translator" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
        </input>
      - <output>
          <soap:body use="encoded" namespace="urn:org:mindswap:services:Translator" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
        </output>
      </operation>
    </binding>
  - <service name="TranslatorService">
      <documentation>Fidn ISBN number of a book name</documentation>
    - <port name="TranslatorPort" binding="ns1:TranslatorBinding">
        <soap:address location="http://www.mindswap.org:8080/soap/servlet/rpcrouter" />
      </port>
    </service>
  </definitions>
```

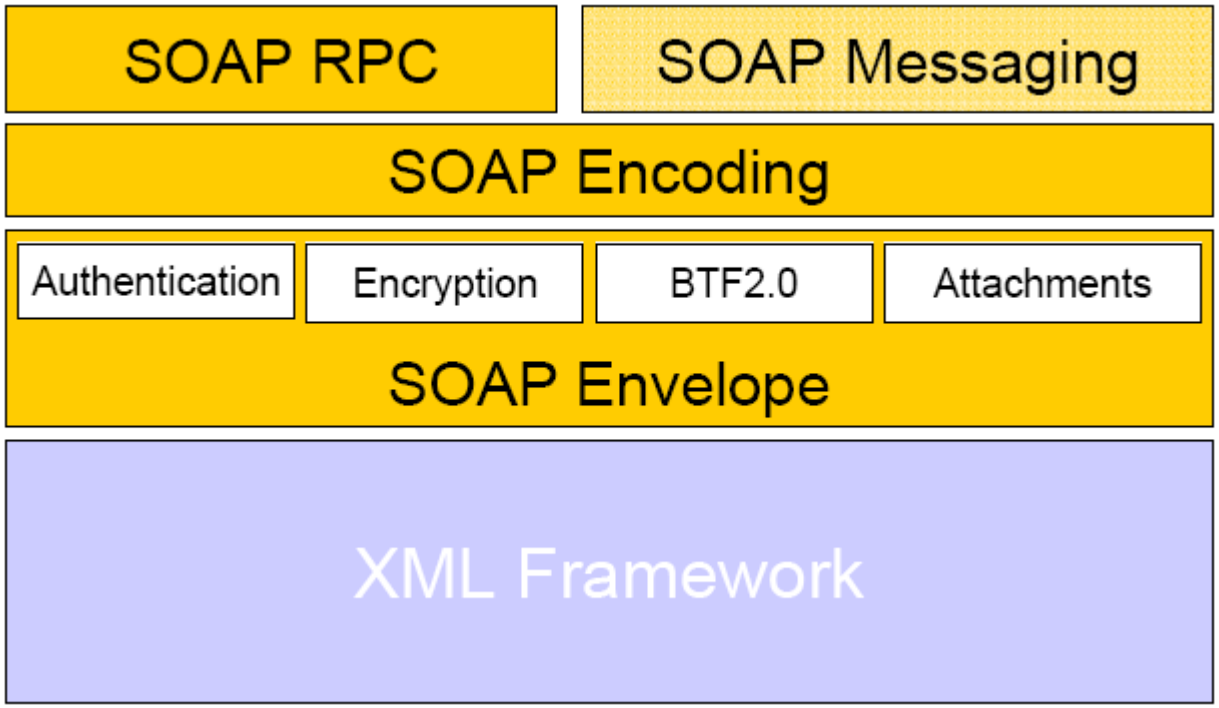
Introducción : SOAP

- Simple Object Access Protocol
- Es un protocolo liviano para el intercambio de información en ambientes distribuídos y descentralizados.
- Estructura Lógica:
  - Envelope: Es un marco que permite describir, que es lo que contiene el mensaje y como debe procesarse
  - Set de Encoding Rules: Reglas que permiten expresar instancias de tipos de datos
  - Convention: Convención para representar RPC (?) y respuestas.

Introducción : SOAP

- No hace todo lo que hace CORBA, RMI o alguna de las arquitecturas de Computación Distribuida del pasado
- Pero es simple…
- Estructura Física:
  - Envelope: Contenedor del mensaje
  - Header: Permite agregar nuevas funcionalidad dependientes de la plataforma
  - Body: Contenedor para la información a transmitir
  - Fault: Contenedor de Mensajes de error.

SOAP


SOAP RPC
SOAP Messaging
SOAP Encoding
Authentication
Encryption
BTF2.0
Attachments
SOAP Envelope
XML Framework

SOAP

Servicio Web	Orientado a RPC	Orientado a documento
Modelo de interacción	RPC	RPC + XML attachment
Modelo de procesado	Centrado en objetos de negocio	Centrado en Documento
Tipo de interacción.	Básicamente Síncrono	Básicamente Asíncrono

Introducción : SOAP

```
- <SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/" xmlns:m="urn:org:mindswap:services:Translator" xmlns:SOAP-
    ENC="http://schemas.xmlsoap.org/soap/encoding/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema" SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  - <SOAP-ENV:Body>
    - <getTranslation xmlns="urn:org:mindswap:services:Translator">
        <inputString xsi:type="xsd:string">Hola</inputString>
        <inputLanguage xsi:type="xsd:string">Spanish</inputLanguage>
        <outputLanguage xsi:type="xsd:string">English</outputLanguage>
      </getTranslation>
    </SOAP-ENV:Body>
  </SOAP-ENV:Envelope>
```

```
  <?xml version="1.0" encoding="UTF-8" ?>
- <SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  - <SOAP-ENV:Body>
    - <ns1:getTranslationResponse xmlns:ns1="urn:org:mindswap:services:Translator" SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
        <return xsi:type="xsd:string">Hello</return>
      </ns1:getTranslationResponse>
    </SOAP-ENV:Body>
  </SOAP-ENV:Envelope>
```

UDDI

- Universal Description, Discovery and Integration
- Es un documento XML que describe un negocio y los servicios que ofrece
- La idea es buscar la compañía que ofrece un servicio requerido, ver sus características y posiblemente contactar al encargado.
- Páginas Amarillas de SW, pero con páginas blancas, amarillas y verdes

UDDI

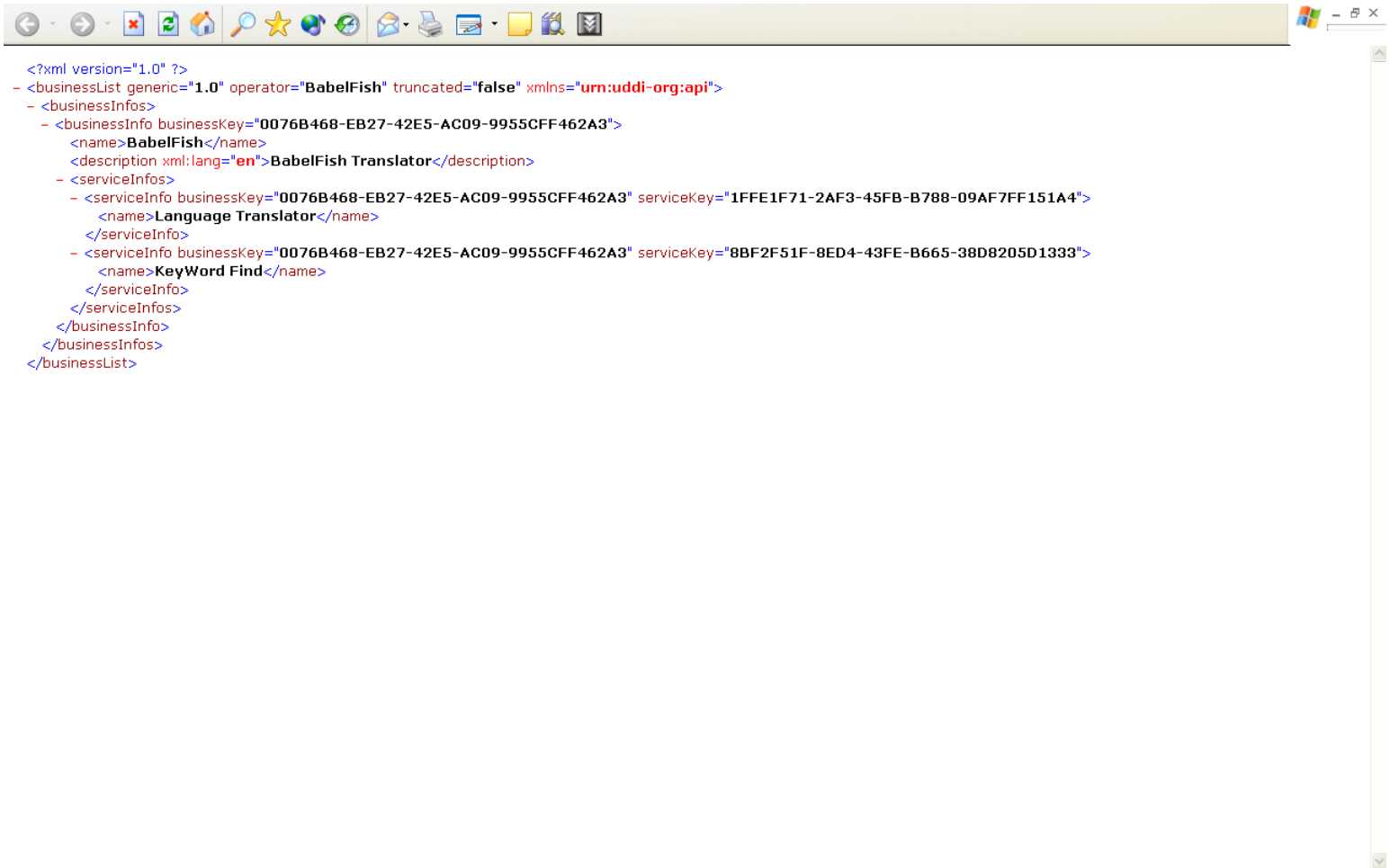
- Estándar de facto para la publicación y descubrimiento de Servicios Web
- Nace de la necesidad de compartir SW internos
- Pretende “divulgar y publicitar” los SW en la Internet
- Dos especificaciones:
  - API (herramientas para publicar y consultar)
  - Data Structure (definiciones y codificaciones)

UDDI

- Codificación:
  - businessEntity: Datos del proveedor del servicio
  - businessService: Categorización basada en negocios o tipos de servicio
  - tModel: Detalles técnicos (gralmente WSDL), pero la idea es independizarse de la implementación.
- Identificados con UUID, “identificador único universal”

UDDI

- Consulta UDDI Ejemplo



```
<?xml version="1.0" ?>
- <businessList generic="1.0" operator="BabelFish" truncated="false" xmlns="urn:uddi-org:api">
  - <businessInfos>
    - <businessInfo businessKey="0076B468-EB27-42E5-AC09-9955CFF462A3">
        <name>BabelFish</name>
        <description xml:lang="en">BabelFish Translator</description>
      - <serviceInfos>
        - <serviceInfo businessKey="0076B468-EB27-42E5-AC09-9955CFF462A3" serviceKey="1FFE1F71-2AF3-45FB-B788-09AF7FF151A4">
            <name>Language Translator</name>
          </serviceInfo>
        - <serviceInfo businessKey="0076B468-EB27-42E5-AC09-9955CFF462A3" serviceKey="8BF2F51F-8ED4-43FE-B665-38D8205D1333">
            <name>KeyWord Find</name>
          </serviceInfo>
        </serviceInfos>
      </businessInfo>
    </businessInfos>
  </businessList>
```

Introducción : UDDI

- Documento UDDI Ejemplo

```
<?xml version="1.0" ?>
- <serviceDetail generic="1.0" operator="BabelFish" truncated="false" xmlns="urn:uddi-org:api">
  - <businessService businessKey="0076B468-EB27-42E5-AC09-9955CFF462A3" serviceKey="1FFE1F71-2AF3-45FB-B788-09AF7FF151A">
      <name>Language Translator</name>
      <description xml:lang="en">Language translator Binding</description>
    - <bindingTemplates>
      - <bindingTemplate bindingKey="313C2BF0-021D-405C-8149-25FD969F7F0B" serviceKey="1FFE1F71-2AF3-45FB-B788-09AF7FF151A">
          <description xml:lang="en">Translation Engine</description>
          <accessPoint URLType="https">https://www.mindswap.org:8080/soap/servlet/rpcrouter</accessPoint>
        - <tModelInstanceDetails>
          - <tModelInstanceInfo tModelKey="uuid:64C756D1-3374-4E00-AE83-EE12E38FAE63">
              <description xml:lang="en">BabelFish SOAP</description>
            </tModelInstanceInfo>
          </tModelInstanceDetails>
        </bindingTemplate>
      </bindingTemplates>
    - <categoryBag>
        <keyedReference keyName="KEYWORD" keyValue="API" tModelKey="uuid:A035A07C-F362-44DD-8F95-E2B134BF43B4" />
        <keyedReference keyName="KEYWORD" keyValue="SOAP" tModelKey="uuid:A035A07C-F362-44DD-8F95-E2B134BF43B4" />
        <keyedReference keyName="KEYWORD" keyValue="XML" tModelKey="uuid:A035A07C-F362-44DD-8F95-E2B134BF43B4" />
      </categoryBag>
    </businessService>
  </serviceDetail>
```

Problemas UDDI

- Los estándares que componen la arquitectura de Servicios Web están diseñados para proveer descripciones del mecanismo de transporte (SOAP), y las interfaces utilizadas (WSDL). Sin embargo estas características no permiten lograr una automática localización de Servicios Web en base a las capacidades que poseen y las funciones que cumplen.

- UDDI intenta describir Servicios Web, mediante un directorio jerárquico de negocios, que a través de una serie de atributos, permite navegar por una jerarquía clasificada de negocios que poseen servicios. Pero tampoco es capaz de representar las capacidades de un Servicio Web, por lo tanto no entrega herramientas adecuadas para buscar servicios en base a lo que proveen. Aún más, la búsqueda sólo permite como entrada palabras clave o *keywords,* lo que lleva a recibir resultados amplios y con una muy baja o nula relación.

Problemas UDDI

- Un Servicio Web puede existir en distintos repositorios UDDI, con distintos identificadores. No existen protocolos de comunicación estandarizados entre directorios, que permitan manejar una sincronización de servicios publicados. Además, un mismo servicio puede ser categorizado con atributos completamente distintos en dos directorios UDDI.

- Cabe recordar que la experiencia práctica en el uso de directorios UDDI públicos (un directorio privado es siempre más riguroso), muestra que existe un gran número de descripciones de Servicios Web incompletas, erróneas o mal clasificadas. Un caso típico es encontrar Servicios Web publicados en un directorio, que ni siquiera cuentan con atributo que indique donde encontrar la descripción WSDL del servicio.

En Resumen: ¿Qué va en el cable hasta ahora?

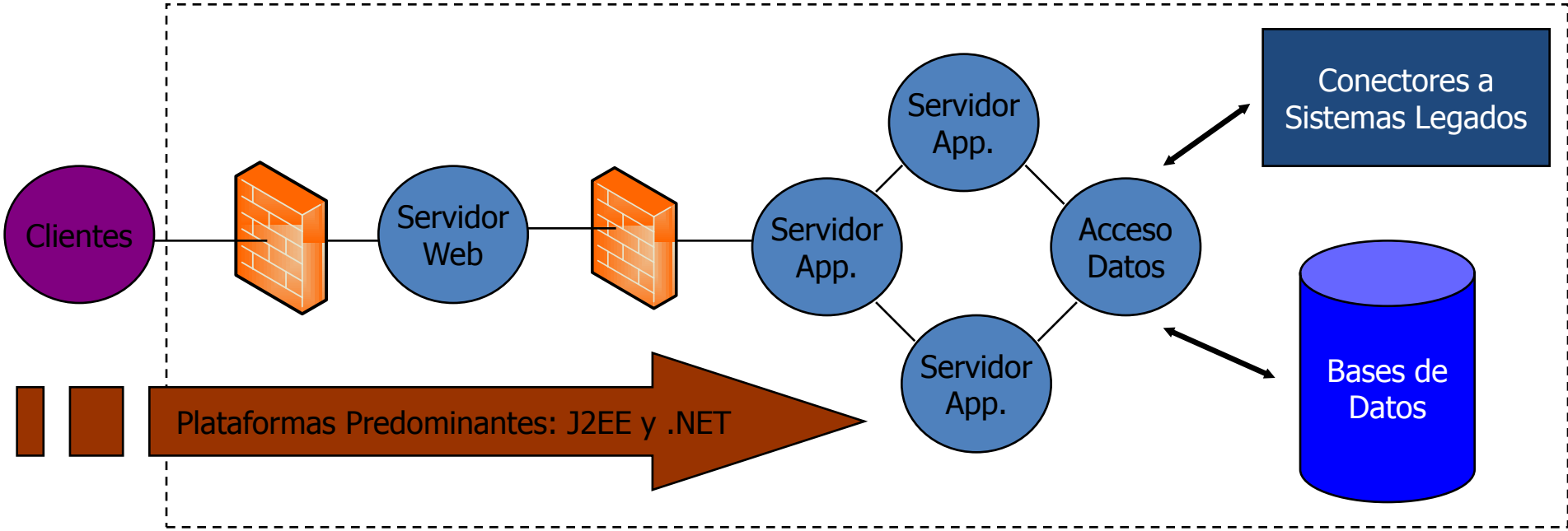
- La integración en Internet necesita un lenguaje
  - XML messaging protocol sobre HTTP: SOAP
- Dentro de organizaciones la integración debe permitir alternar entre:
  - CORBA, RMI
  - Messaging
  - Llamado de métodos en memoria

Piezas Perdidas

- Security (WS-Security)
  - Logeo, credenciales
- Transactions (WS-Transaction)
- Calidad de Servicio (WS-ReliableMessaging)
  - Garantías de la puntualidad
- Operaciones asincrónicas
  - Coordinaciones, Workflow

Introducción a la seguridad en Servicios Web

Arquitectura General Aplicaciones Empresariales



Seguridad Perímetro

- Firewalls/VPNs
- Criptografía
- Seguridad Servidores Web
- Detección de Intrusión
- etc.

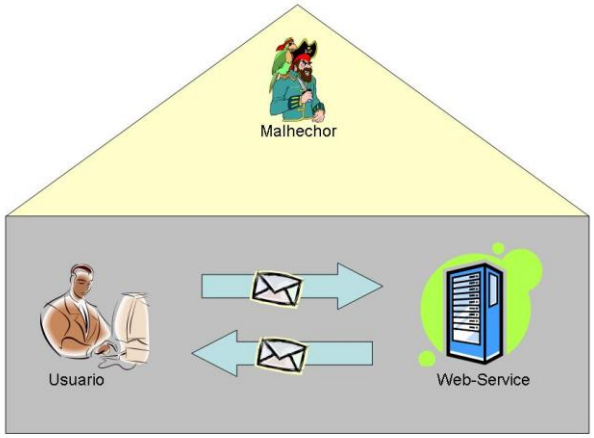
Seguridad Middleware

- Roles
- Seguridad Componentes
- Criptografía
- etc.

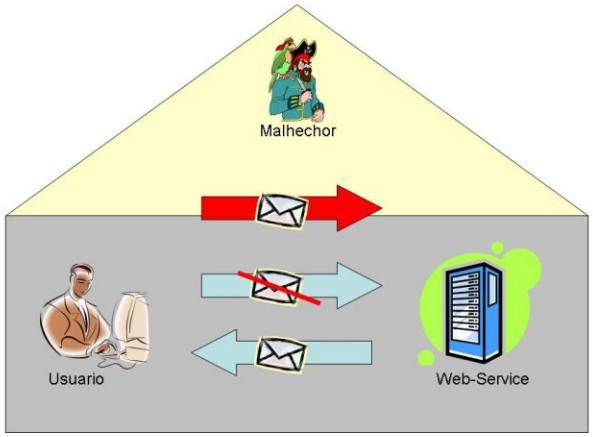
Seguridad Back-Office

- Seguridad Mainframe
- Seguridad RDBMS
- etc.

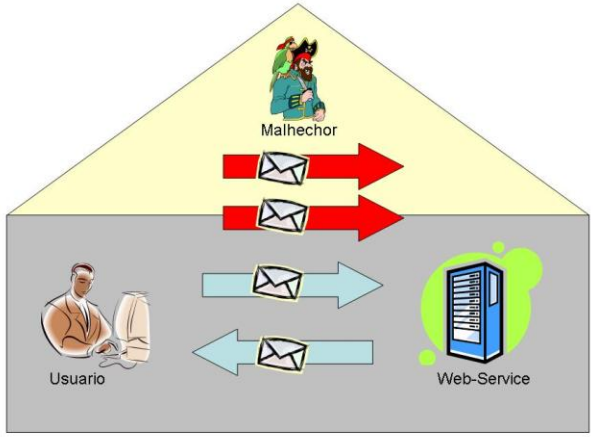
¿De qué debemos proteger nuestras aplicaciones?



Violación de Confidencialidad

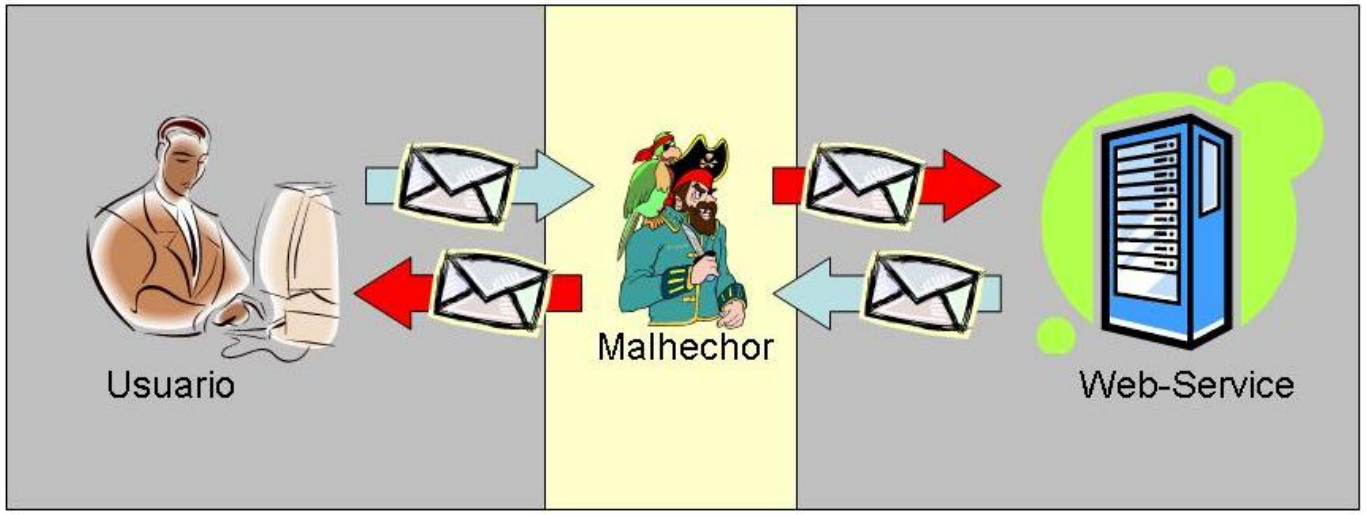


Violación de Integridad



Ataque de Repetición

Ataques Aplicaciones



Ataque del Intermediario *(conocido como "Man in the Middle")*

Requerimientos de Seguridad

- Autenticación
- Autorización
- Confidencialidad
- Integridad
- No repudiación
- Alta Disponibilidad

Desafíos de Seguridad Servicios Web

- Seguridad basada en el usuario final
- Mantener la seguridad al pasar por múltiples Servicios (“nodos”)
- Abstracción de la seguridad del transporte subyacente

WS-Security

- Esfuerzo conjunto de IBM, Microsoft y VeriSign
  - En Abril de 2002 publican “Security in a Web Services World: A Proposed Architecture and Roadmap”
- Hoy mantenida por OASIS
- Su objetivo es Proveer seguridad a SOAP
- Se enfoca en la correcta y efectiva aplicación de tecnologías como
  - XML Signature
  - XML Encryption
  - SAML
- Provee un contenedor para artefactos de seguridad

Opciones de Seguridad Capa de Mensajería

Servicio de Seguridad	Tecnologías	
Integridad	XML Signature	
	S/MIME	
	PKCS#7	
Confidencialidad	XML Encryption	
Autenticación del Emisor SOAP (Cliente)	XML Encryption	username & [password|digest]
	username & [password|digest]	
	Certificado X.509	
	Token de Seguridad	Kerberos
		SAML
		REL
		Etc.

Comparación Seguridad Según Capa

Seguridad de Transporte	Seguridad de Mensajería
Punto a Punto	Destino a Destino
Madura, su implementación es relativamente directa	Nueva, relativamente compleja con muchas opciones de seguridad
No granular, enfoque del todo o nada	Muy granular, puede aplicar selectivamente a trozos de mensajes y solamente a los requerimientos o respuestas
Dependiente del Transporte	La misma estrategia puede aplicarse a distintas tecnologías de transporte

Seguridad Capa de Mensajería

- Construida sobre modelos maduros
- Gran cantidad de especificaciones
  - Muchas de ellas muy inmaduras
- Mayor Fortaleza
  - **Flexibilidad**
- Mayor Debilidad
  - **Flexibilidad**

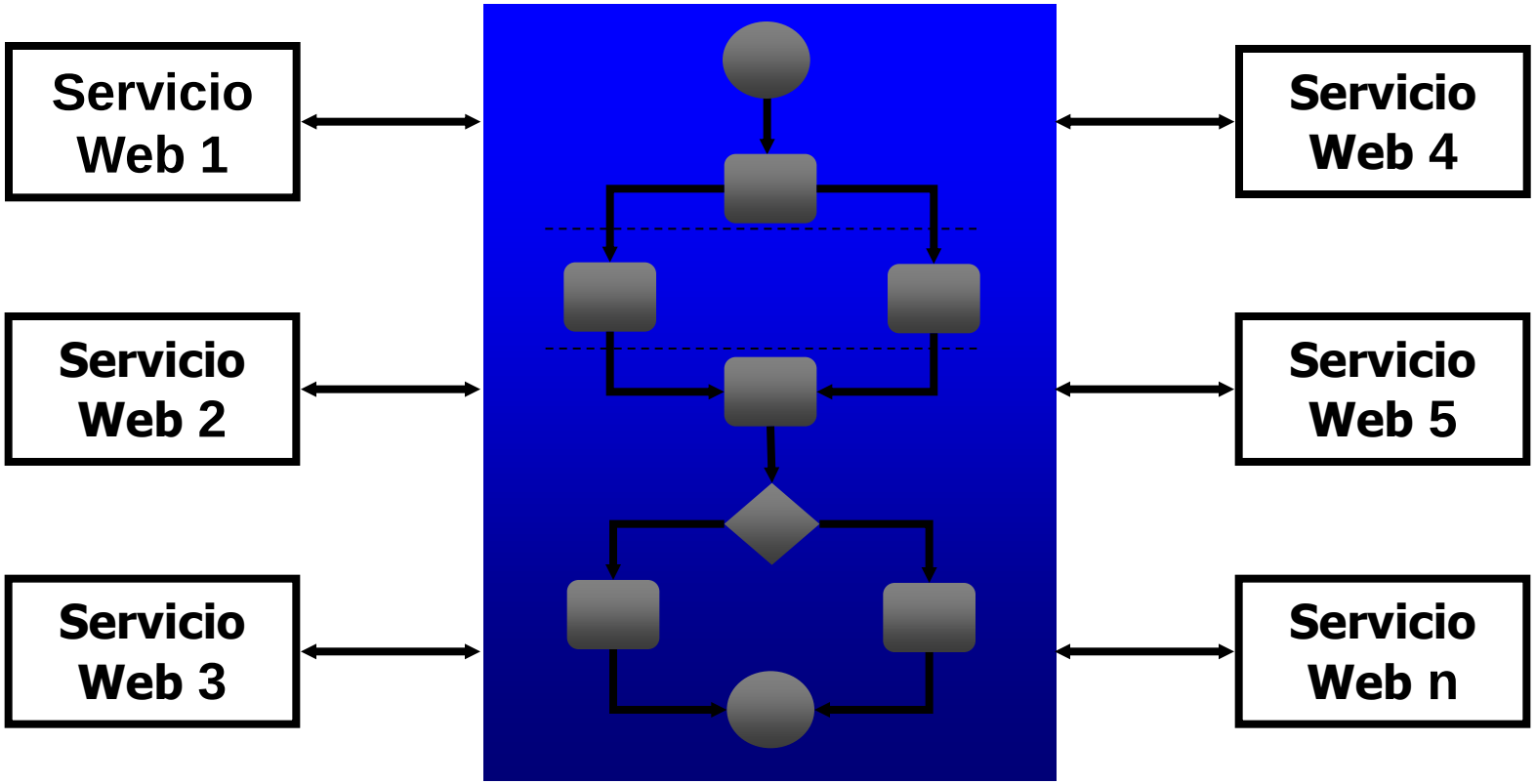
Aplicaciones Directas “Enterprise”

Composición, Orquestación y Coreografía de Servicios Web

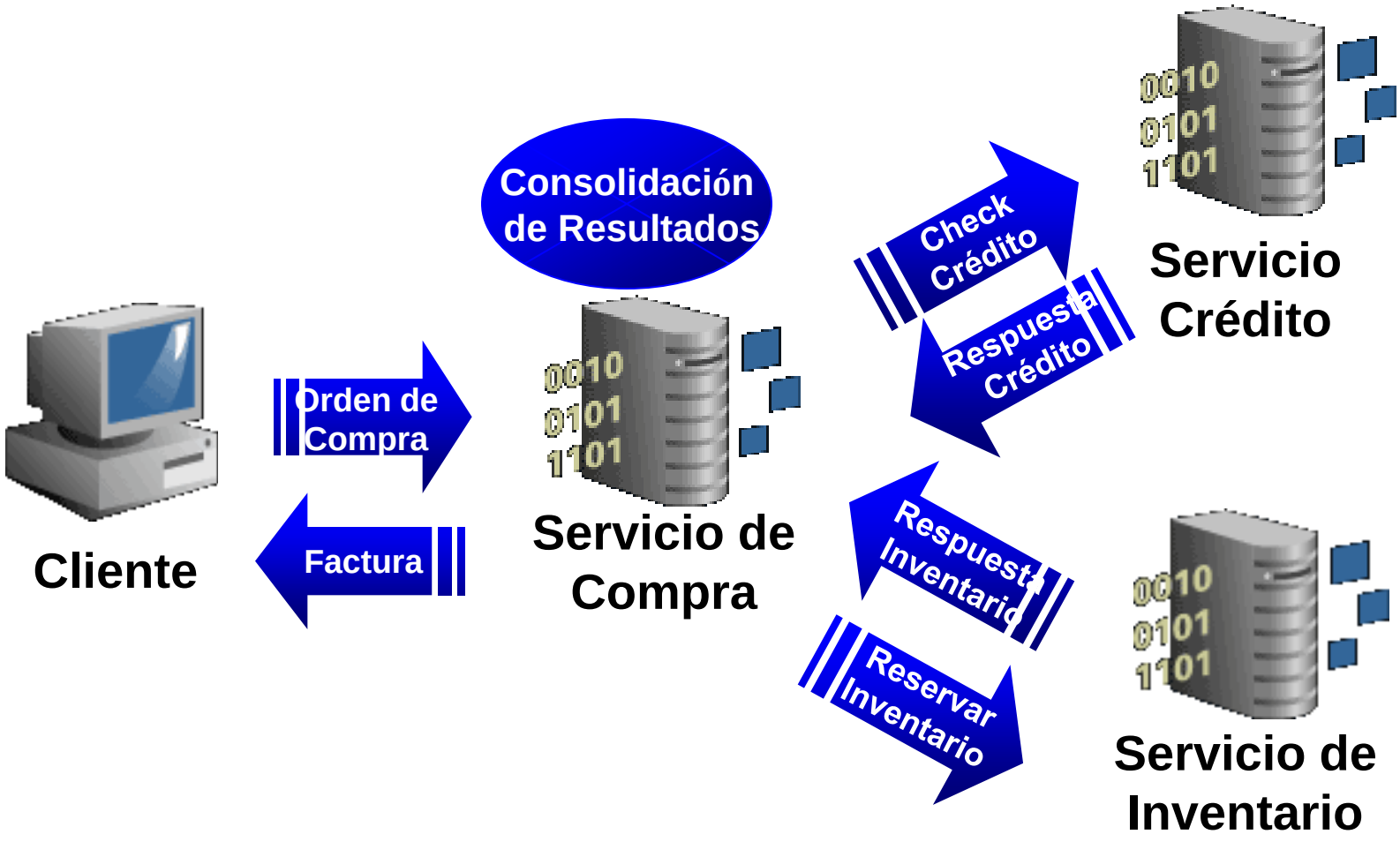
Composición

- Paradigma emergente
- Multiples intentos de estandarización de lenguajes (WSFL,XLANG,BPML)
- Se pueden caracterizar en función de patrones de diseño
- Comparable a diseñar un workflow

Servicios Web como procesos de negocio



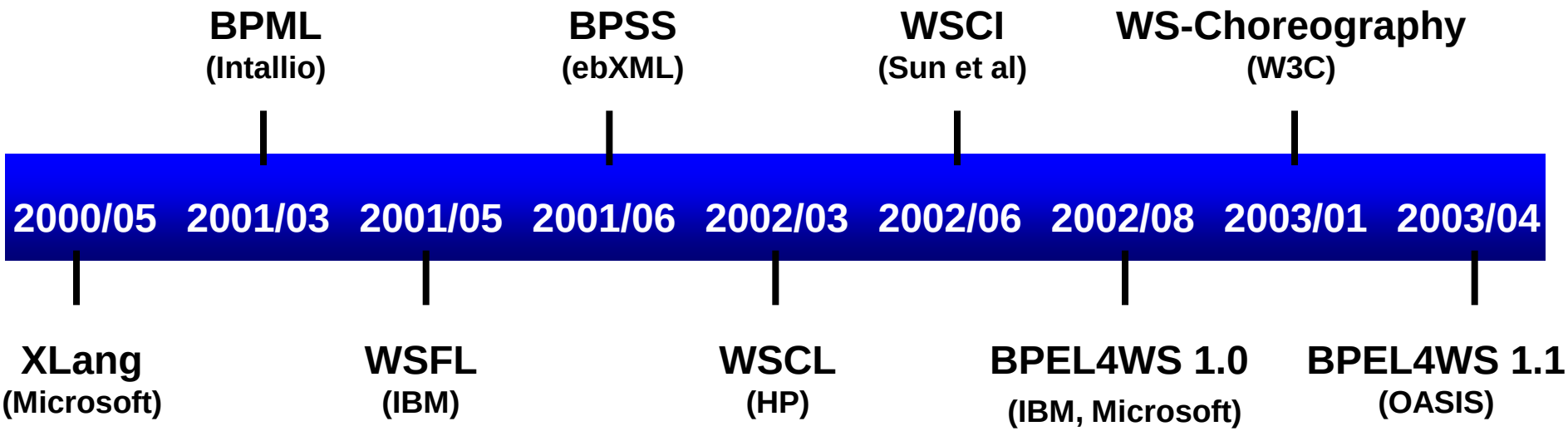
Problema Tipo



Desafío de los Procesos de Negocio

- Comunicación asincrónica coordinada entre los servicios
- Intercambios de mensaje correlacionados entre los participantes
- Implementar el procesamiento paralelo de actividades
- Manipular y transformar data entre participantes de la interacción
- Soporte para las transacciones y las actividades duraderas de negocio
- Proporcionar manejo de excepciones consistente

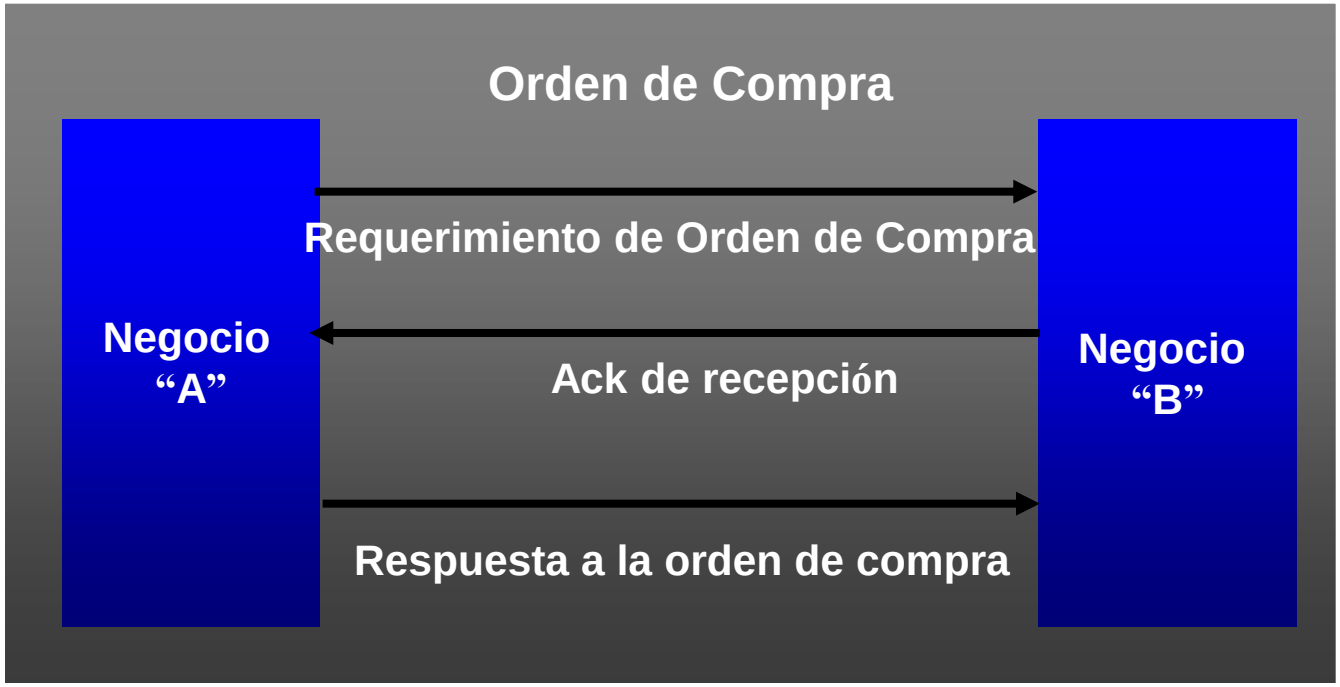
Historia de los estándares de Procesos de Negocio



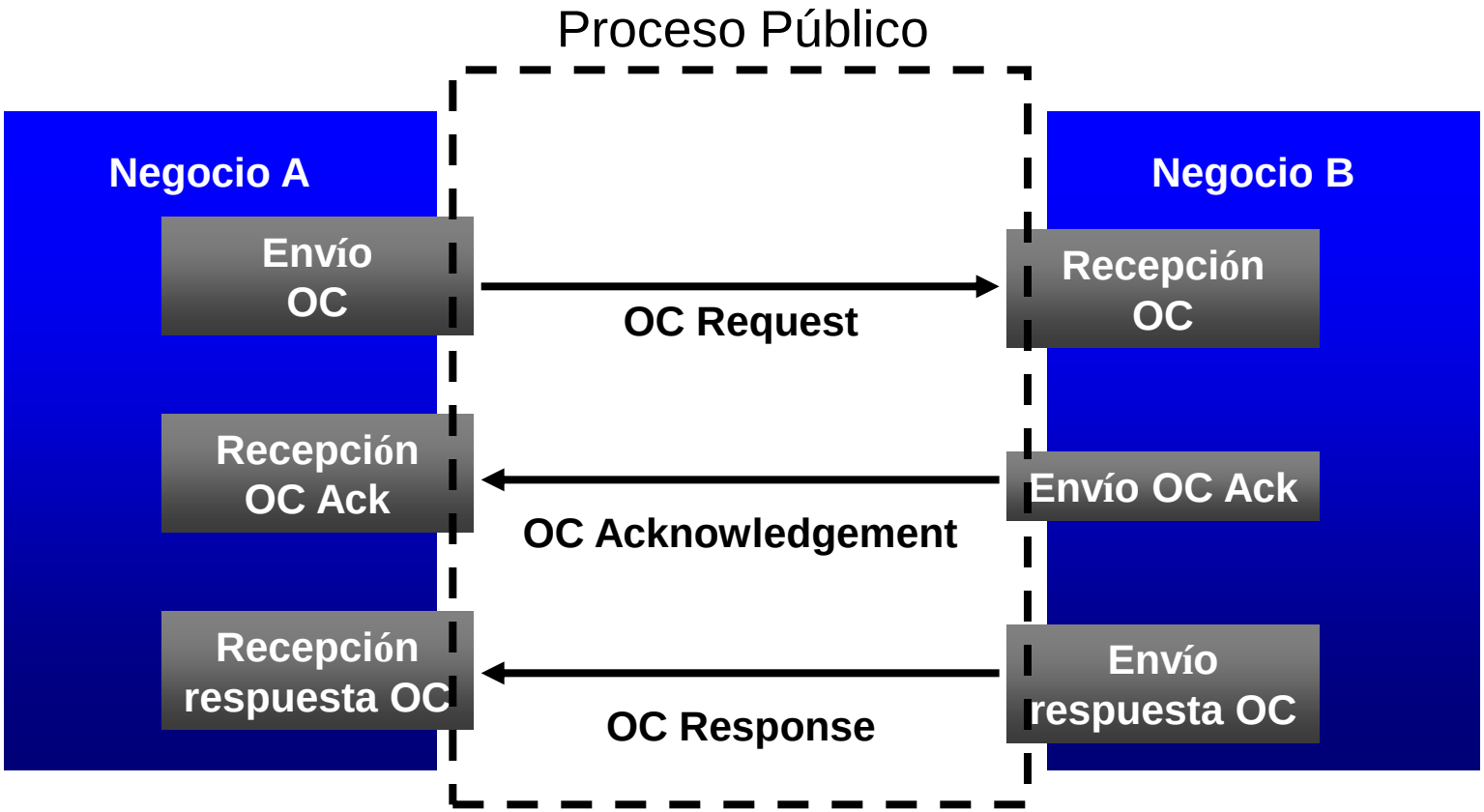
Orquestación vs Coreografía

- Orquestación
  - Es un proceso de negocio ejecutable que describe un flujo desde la perspectiva y control de un sólo punto de emisión (endpoint) (comúnmente: Workflow)
- Coreografía
  - Son los visibles y públicos intercambios de mensajes, reglas de interacción y acuerdos entre dos o más puntos de emisión de procesos de negocio

Ejemplo: Orden de compra

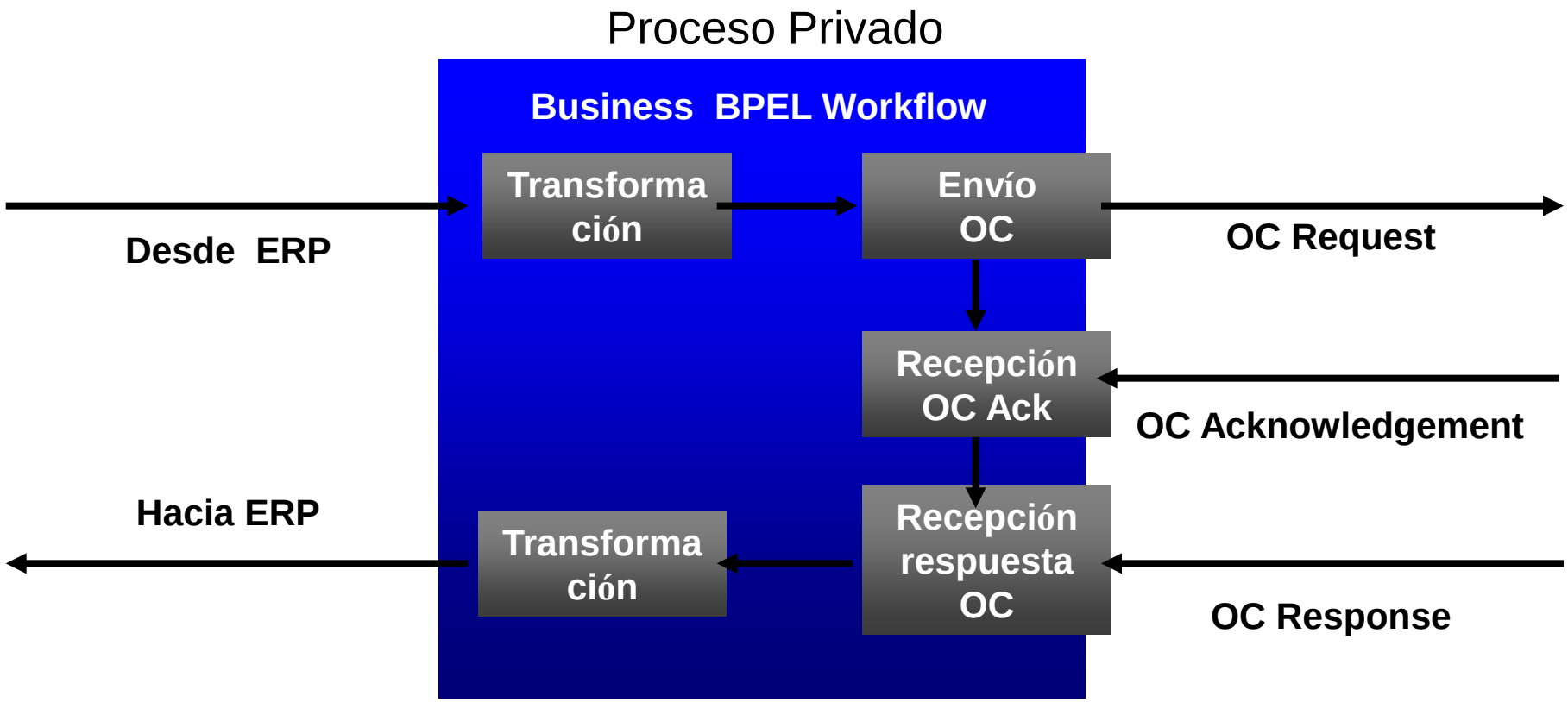


Visto como Coreografía



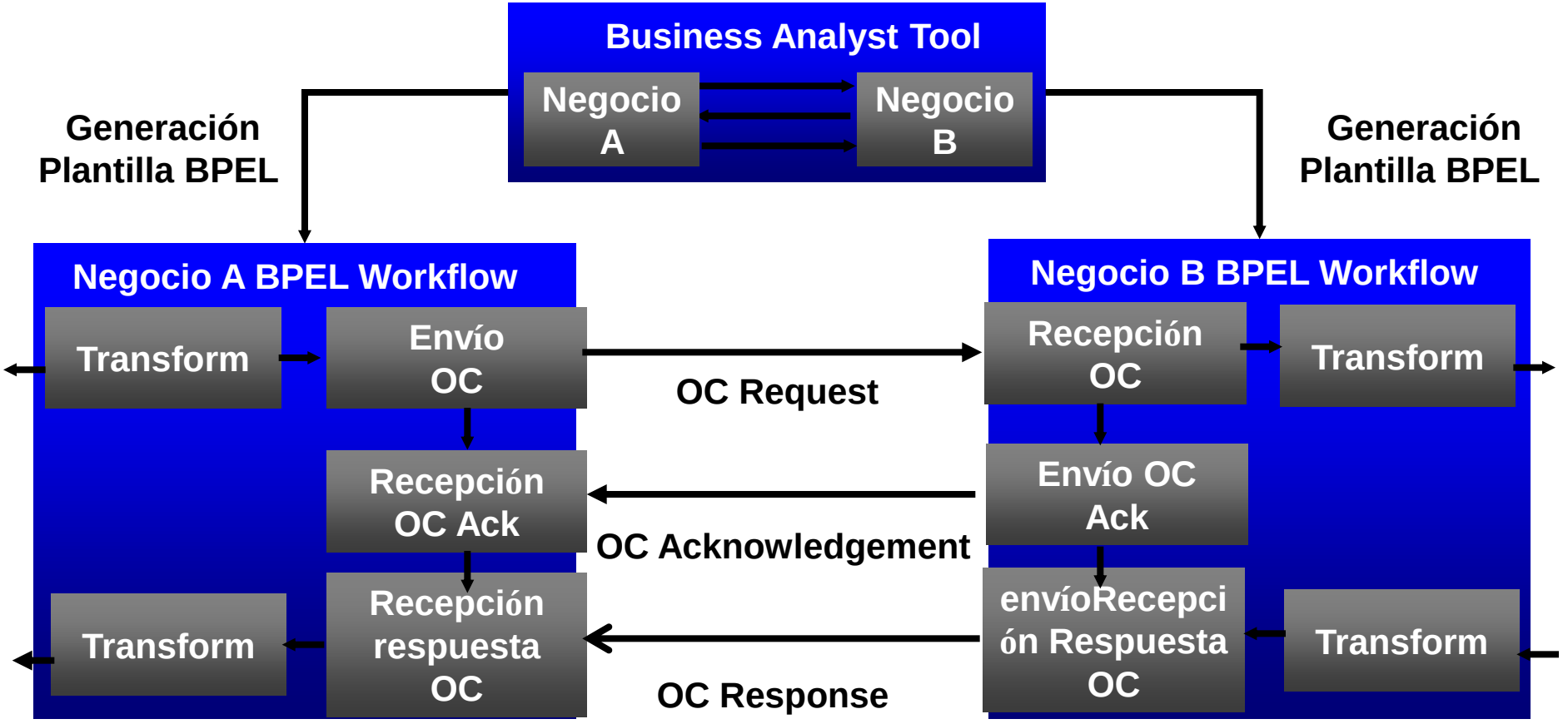
Coreografía – Las públicas y visibles interacciones de intercambio de mensajes

Visto como Orquestación



Orquestación – Es un proceso de negocio privado ejecutable

Orquestación y Coreografía juntos



Dos Workflow BPEL que muestran acuerdo entre negocios

Orquestación vs. Coreografía (Tener siempre presente)

Orchestration

Un único director en control

Choreography Define interacción. Desribe el público y visible intercambio de mensajes

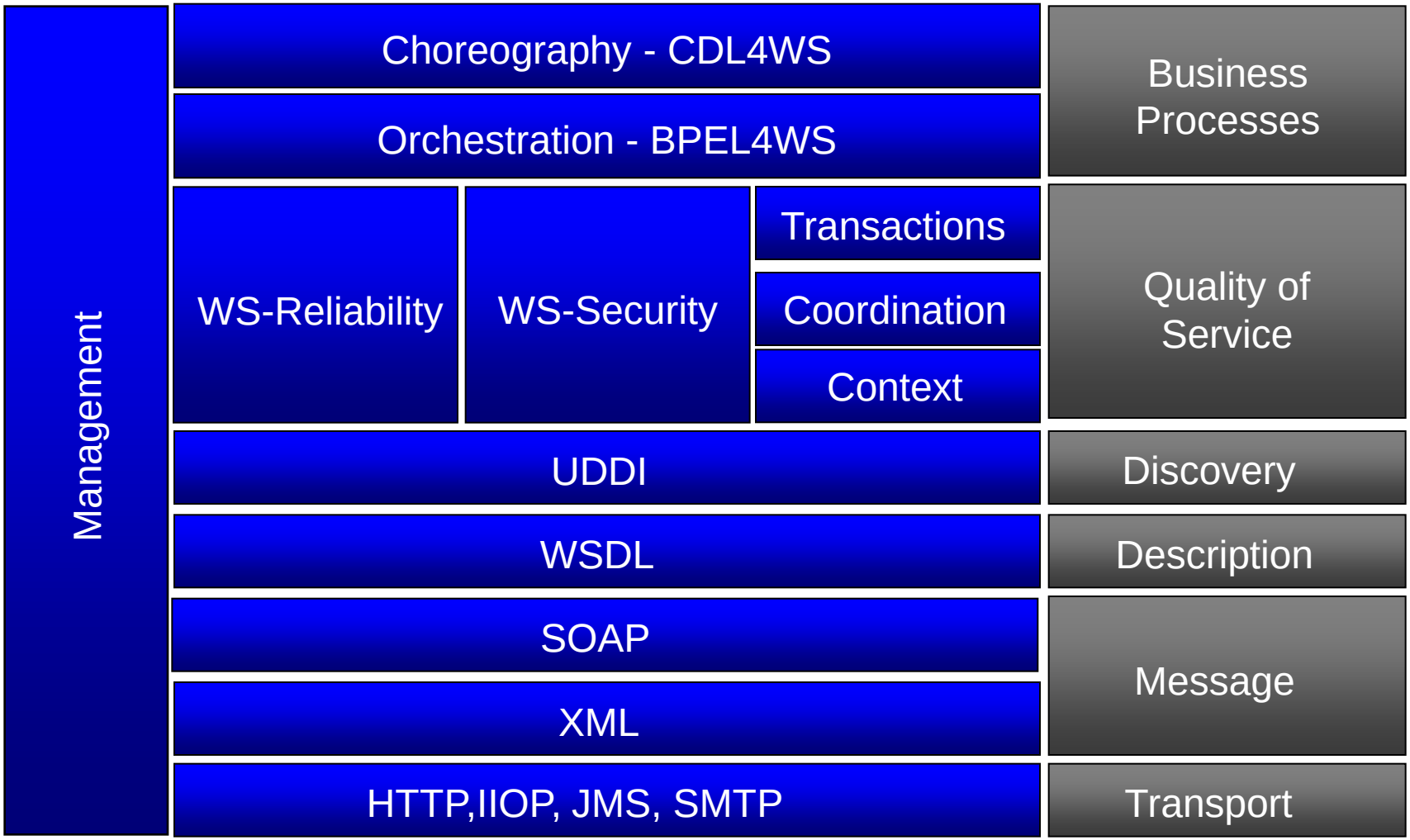
BPEL

- Business Process Execution Language
- Version 1.0 liberada por IBM, Microsoft y BEA agosto 2002
- Version 1.1 subida a OASIS Abril 2003
- Lenguaje XML para describir procesos de como Servicios Web
  - Convergencia de XLANG (Microsoft) y WSFL (IBM)
- Consenso corporativo
  - IBM, Microsoft, Oracle, Sun, BEA, SAP, Siebel …

Proposición de BPEL

- Procesos de negocio portables
  - Construido sobre la infraestructura interoperable de los Servicios Web
- Gran lenguaje empresarial para procesos de negocio
  - Set de habilidades y lenguaje común para desarrolladores
- Selección del motor de procesos
  - El estándard lleva a la competencia

Estándares que construyen BPEL



Actividades BPEL

Actividades

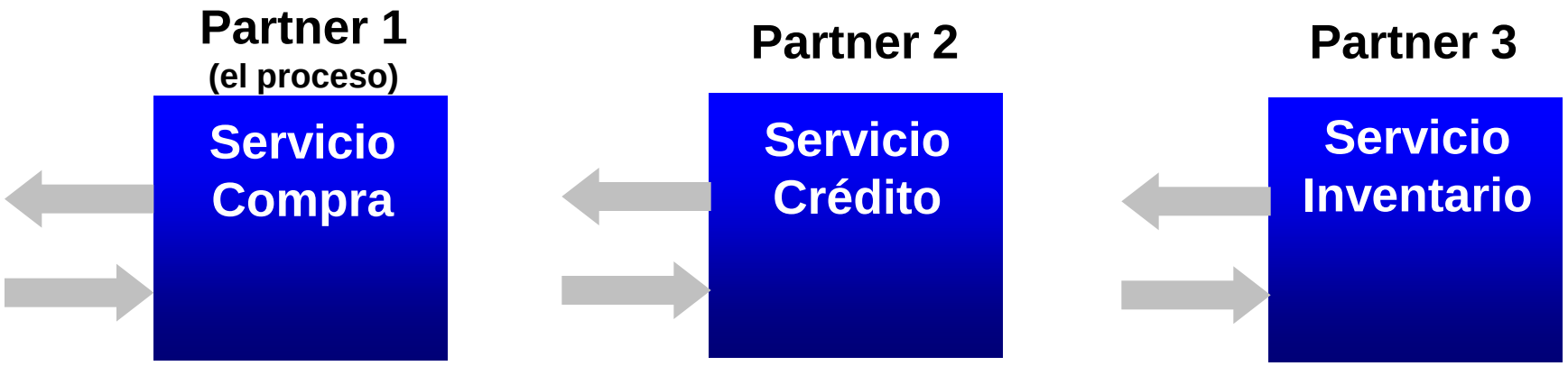
- **<invoke>**
- **<receive>**
- **<assign>**
- **<reply>**
- **<throw>**
- **<terminate>**
- **<wait>**

Actividades Estruturadas

- **<sequence>**
- **<switch>**
- **<pick>**
- **<flow>**
- **<link>**
- **<while>**
- **<scope>**

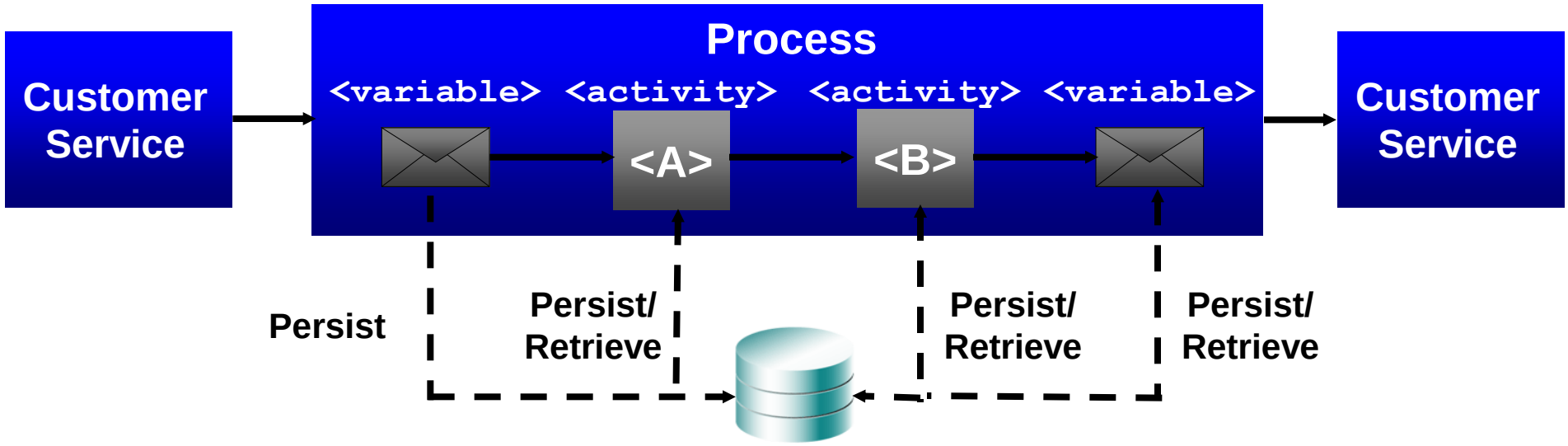
Partners/socios

- Aquí se declaran los Servicios Web y los roles utilizados por el proceso
- Ligado al WSDL del proceso en sí y a los Servicios Web participantes mediante tipos de enlaces de servicio



Variables

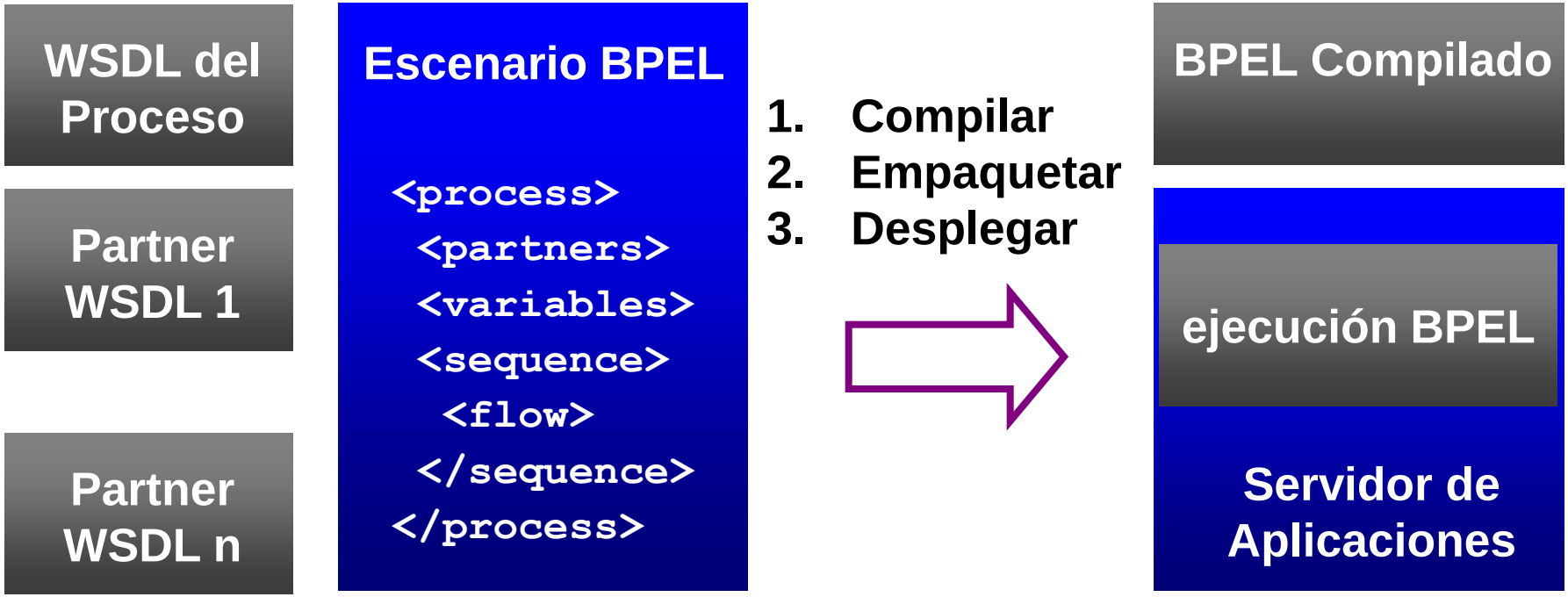
- Mensaje enviados y recibidos por partners
  - Persisten en largas interacciones
  - Definidas en los tipos y mensajes WSDL



Ejemplo documento BPEL

```
- <process xmlns="http://schemas.xmlsoap.org/ws/2003/03/business-process/" name="simple" targetNamespace="urn:news:NewsService"
    xmlns:tns="urn:news:NewsService">
  - <partners>
      <partner name="User" serviceLinkType="tns:UserSLT" />
      <partner name="EnglishNewsService" serviceLinkType="tns:EnglishNewsSLT" />
      <partner name="HTML2TextService" serviceLinkType="tns:HTML2TextSLT" />
      <partner name="BabelFishService" serviceLinkType="tns:BabelFishServiceSLT" />
    </partners>
  - <variables>
      <variable name="userRequest" messageType="tns:userRequest" />
      <variable name="newsRequest" xmlns:ns1="http://www.SoapClient.com/xml/SQLDataSoap.wsdl" messageType="ns1:ProcessSRL" />
      <variable name="newsResponse" xmlns:ns2="http://www.SoapClient.com/xml/SQLDataSoap.wsdl" messageType="ns2:ProcessSRLResponse" />
      <variable name="htmlNews" xmlns:ns3="http://mycompany.com/services/util/html2text" messageType="ns3:html" />
      <variable name="textNews" xmlns:ns4="http://mycompany.com/services/util/html2text" messageType="ns4:text" />
      <variable name="englishNews" xmlns:ns5="http://www.xmethods.net/sd/BabelFishService.wsdl" messageType="ns5:BabelFishRequest" />
      <variable name="translatedNews" xmlns:ns6="http://www.xmethods.net/sd/BabelFishService.wsdl" messageType="ns6:BabelFishResponse" />
      <variable name="userResponse" messageType="tns:userResponse" />
    </variables>
  - <sequence name="sequence">
      <receive name="receiveRequest" partner="User" portType="tns:NewsPT" operation="getNews" variable="userRequest" createInstance="yes" />
    - <assign name="CreateNewsInput">
      - <copy>
          <from expression="'/xml/NEWS.SRI'" />
          <to variable="newsRequest" part="SRLFile" />
        </copy>
      - <copy>
          <from expression="'yahoo'" />
          <to variable="newsRequest" part="RequestName" />
        </copy>
      - <copy>
          <from expression="''" />
          <to variable="newsRequest" part="key" />
        </copy>
      </assign>
      <invoke name="invokeNewsService" partner="EnglishNewsService" xmlns:ns7="http://www.SoapClient.com/xml/SQLDataSoap.wsdl"
        portType="ns7:SQLDataSoapPortType" operation="ProcessSRL" inputVariable="newsRequest" outputVariable="newsResponse" />
    - <assign name="mapNewsOutputToConversionInput">
      - <copy>
          <from variable="newsResponse" part="return" />
          <to variable="htmlNews" part="content" />
        </copy>
      </assign>
      <invoke name="invokeConvertor" partner="HTML2TextService" xmlns:ns8="http://mycompany.com/services/util/html2text" portType="ns8:Html2TextPortType"
        operation="convertHTML2Text" inputVariable="htmlNews" outputVariable="textNews" />
```

En Resumen


WSDL del Proceso
Partner WSDL 1
Partner WSDL n
Escenario BPEL
<process>
<partners>
<variables>
<sequence>
<flow>
</sequence>
</process>
1. Compilar
2. Empaquetar
3. Desplegar
BPEL Compilado
ejecución BPEL
Servidor de Aplicaciones

En que se usa BPEL?

SOA?

BPM-BPMS

Introducción a SOA

- SOA es la nueva palabra mágica en IT
  - Evolucionó del concepto API
  - La API expone una función de aplicación hacia el mundo externo hacia el mundo exterior a través de un modelo de componente establecido y un contrato de servicio (Ej.: interface definition language)
  - Nos referimos a estas API como interfaces de componentes.

Introducción a SOA

- SOA es una fusión de arquitecturas basada en componentes y Web Services.
  - En lugar del “estilo” RPC SOA utiliza SOAP sobre HTTP como medio de comunicación.
  - El Web service definition language remplazo al component interface definition languge

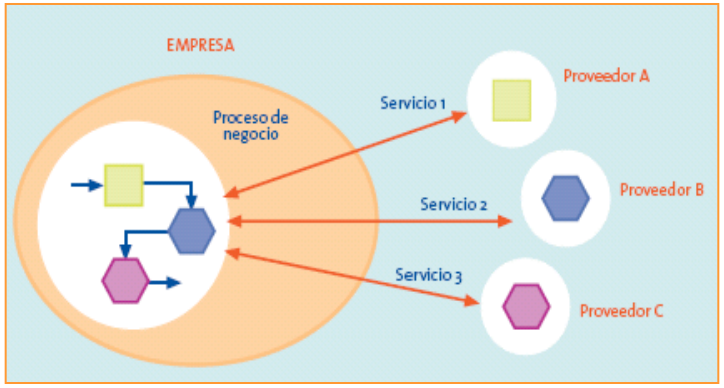
Introducción a SOA

- Diferencias importantes entre arquitecturas SOA y API:
  - SOA permite interacciones inter e intra empresas mientras que API normalmente sólo intra.
    - SOA utiliza comúnmente http y XML
    - La comunicación basada en API requiere invocar un cliente y recibir un destino para utilizar el mismo modelo (de componente) (CORBA, J2EE, .NET, etc.)

¿Qué es SOA?

SOA (**Service Oriented Architecture**) es un modelo de arquitectura del software por el cual las necesidades o procesos de negocio se plantean en base a servicios.

Un **servicio** es una pieza de software que cumple principalmente las siguientes características:



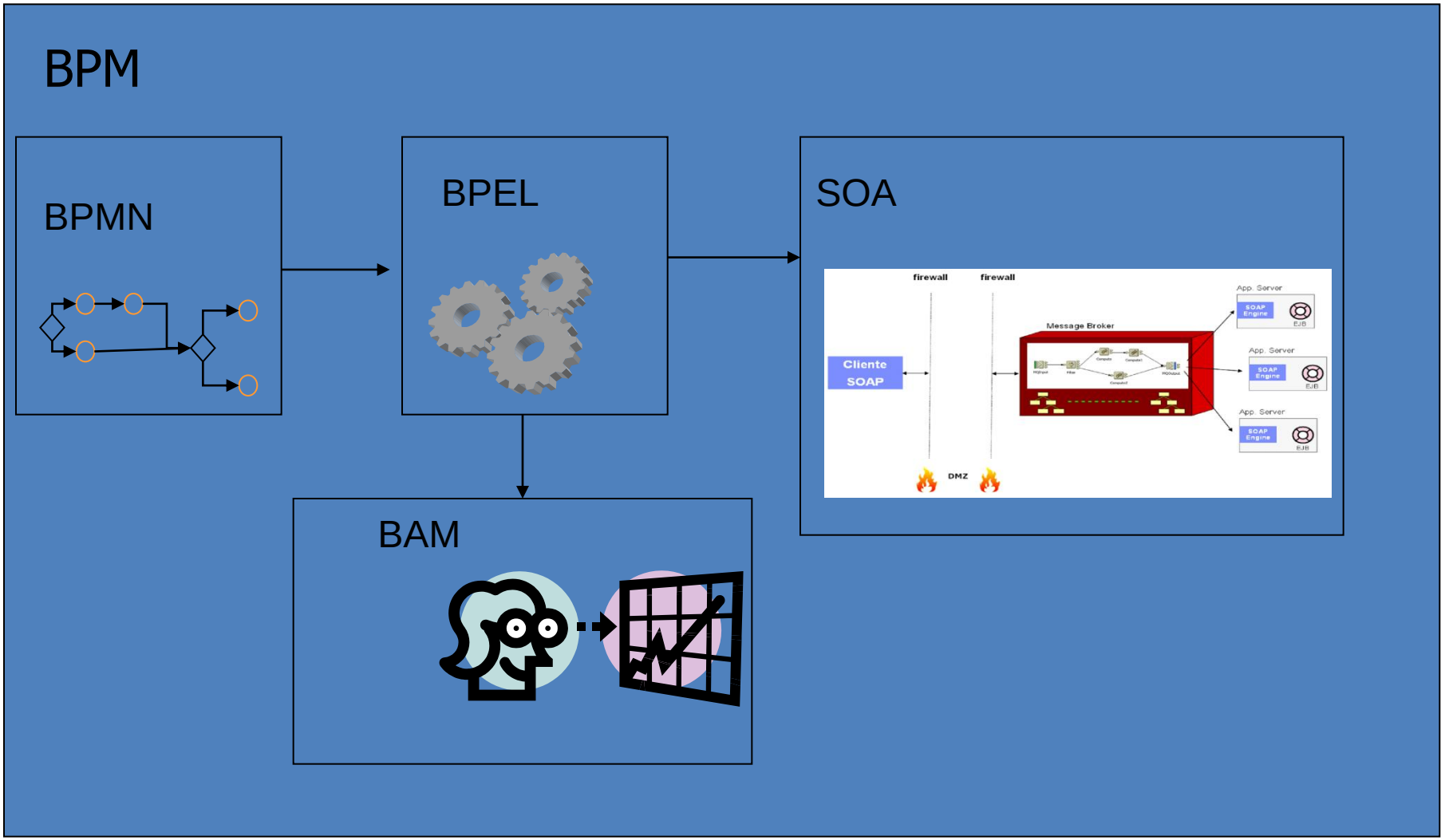
- Implementa una determinada función o *lógica de negocio reutilizable*.
- Es invocable por parte de *consumidores diversos* (otras piezas de software), de tecnologías diversas, y sin requerir un estado o información de contexto previa (autocontenido).
- Acoplamiento débil o *aislamiento* de los consumidores de un servicio respecto de la implementación del mismo (que puede alterarse sin afectarlos).

SOA define, por tanto, una *arquitectura de aplicación* cuyas funciones de negocio se definen como *servicios independientes* (entre sí, y respecto de sus llamadores), que en una determinada secuencia forman procesos de negocio

Dado esto..

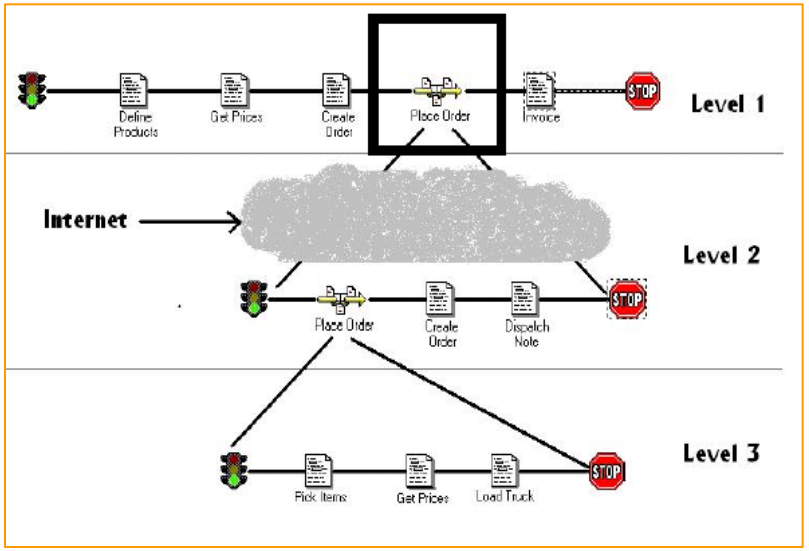
¿Qué es un BPMS y cómo construyo procesos BPM con un BPMS?

BPM, BPEL, BAM, BPMN, SOA, Web Services



Gestión de Procesos.

Ejemplo de Proceso implementado con un BPM



Construcción de un proceso con BPM.

1. **Definir el proceso**: El BPM incluye funcionalidades para el modelado del proceso mediante una herramienta gráfica que permite además la validación y depuración del mismo.
2. **Implementar el proceso**. El BPM incluye servicios de formularios, motores de reglas que facilitan la implementación del proceso.
3. **Despliegue del proceso**. El BPM incluye un motor sobre el que ejecutar workflows.
4. **Gestión del proceso**. El BPM permite al administrador del proceso controlarlo y monitorizarlo (reasignar tareas, cancelarlas, monitorear su estado, mantenimiento de diferentes versiones del proceso…).
5. **Medición del proceso**. Para esto el BPM incluye funciones de auditoria y reporting.

¿Qué es un BPMS?.

Un **BPMS** es la suma de las funcionalidades de un **Workflow**, más ciertas capacidades para **Integración de Aplicaciones**, más funcionalidades avanzadas para la **Gestión de Procesos**.

BPMS vs Workflow. La mayor parte de los productos tradicionales de Workflow han ido evolucionado hacia productos de BPM. Ahora, el mercado entiende ahora por productos de Workflow, aquellos que sirven para implementar exclusivamente **workflows manuales** con capacidades de gestión de procesos muy limitadas (Adobe, Teamtrack son ejemplos de productos de Workflows).

Resumen

Descripciones: Meta-data

- La integración requiere de descripciones interoperables entendibles por máquinas
- Permite el enlace tardío de componentes
- La extensión del los lenguajes provee soporte para diferentes niveles de integración entre aplicaciones.

Introducción a Grid Services

Grid Services

- La tecnología grid se ha vuelto muy popular
- En 2003 fue declarada por el MIT como una de las 10 tecnologías emergentes que cambiarán el mundo.
- La palabra ”grid” empieza a aparecer por todos lados, todo el mundo habla de ello
- Lo cual está introduciendo mucha confusión

Grid Services

- ¿Qué es el Grid?
  - “Mientras que la Web es un servicio para compartir información a través de Internet, el Grid es un servicio para compartir potencia de cálculo y capacidad de almacenamiento a través de la red”
- Esta forma de compartir se realiza abstrayendo/virtualizando los recursos que participan en una infraestructura grid, de manera que para el usuario final actúan como un único y potente “computador”
- Los teóricos del grid computing entienden que el objetivo final de la tecnología grid es crear una infraestructura cuyo ámbito sea todo Internet
- Integrando todos los heterogéneos recursos computacionales que existen alrededor del mundo

Grid Services

- Mitos y otras falsedades sobre grid computing
  - El Grid **NO** es una mejora/ampliación de Internet (no están al mismo nivel)
  - El Grid **NO** es un proyecto (es una tecnología)
  - El Grid **NO** es un cluster de servidores (en un grid puede haber integrados muchos o ningún cluster)

- El Grid toma el nombre de su analogía con la red eléctrica (inglés ”*power grid*”):

Grid Services: Grid Middleware

- El Grid es posible gracias al ”grid middleware”
  - El software que permite la integración de todos los distintos tipos de recursos que participan en él.
- ¿software especial? ¿*middleware*?
  - Definición de middleware (Wikipedia): ”En un entorno de computación distribuida, el middleware se define como la capa de software que se encuentra entre el sistema operativo y las aplicaciones en cada host/máquina que participa en el sistema”
- Ejemplos: RPC, RMI, Corba, EJBs…
- “grid middleware”: middleware que se usa en el Grid

Grid Services (Middleware)

- Finalidad: virtualización de los recursos de computación
- Funcionalidades:
  - Asignación eficiente de recursos
  - Ejecución de trabajos y posterior transferencia de resultados
  - Almacenamiento, registro y posterior localización y acceso a datos
  - Proveer mecanismos de seguridad: autenticación, autorización...
  - Monitorización

- El middleware es el **”cerebro”** del Grid

- El grid middleware se construye mediante servicios grid (”*grid services*”), que a su vez están basados en la tecnología de Servicios Web

Aplicación Directa
“Development”

Introducción

- La Web da un paso adelante cambiando el tradicional modelo de uso de recarga de páginas
  - Cada vez que existe nueva data ya sea de forma completa o en una porción de ella

- Las compañías experimentan con recargas dinámicas de porciones de páginas web, transmitiendo sólo una fracción de datos al browser cliente, lo que pretendía entregar una experiencia más rápida y mejor al usuario

Introducción AJAX

- Ejemplos:
  - Google Labs
    - Google Maps
    - Gmail
- Nace Ajax
  - Asynchronous Javascript + XML
  - Ver “Ajax: a New Approach to Web applications” (James Garret)

Arquitectura AJAX

- Servidor Web
- HTML/XML
- DOM
- XMLHttp/MSXML
- Javascript

Componentes

- XMLHttpRequest
  - Objeto nativo de browsers mozilla
- MSXML
  - Componente Activex Internet Explorer (hasta IE6)

Servicios Web y Ajax

- Utilización de SW como componente activo de software para los browsers.

- No estamos sólo requiriendo un dato o un documento, sino que se consume un servicio con parámetros de entrada y salida

Cierre

Aplicaciones/Frameworks

- Axis
- .NET
- SAP (mySAP) (*)
- Oracle (*)
- BEA (*)
- WebSphere - IBM

Conclusiones y Futuro

- Ventajas
  - Desarrolladores no se preocupan mas por la infraestructura
  - Reducción de costos en el ciclo de desarrollo de software por la alta reutilización de componentes
  - Proveedores de Servicios
  - Acceso personalizado en diversos medios y dispositivos
  - Interoperabilidad
  - Multiplataforma

Conclusiones y Futuro

- Posibilidades
- Actualmente Servicios Web ..
  - Necesitan de una codificación estática de su construcción, invocación y conexiones mutuas
  - Nula reacción a cambios: proveedor, interrupción, modificación de la interfaz
  - Mantenimiento es costoso
  - No permite optimizar, buscar el servicio que más se acomode a mi realidad y de menor costo