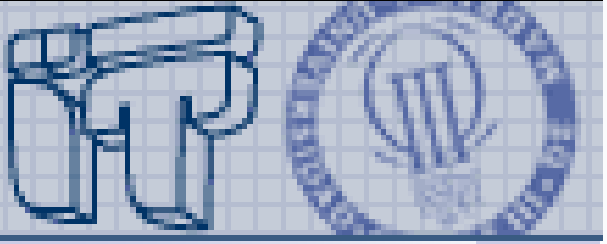


Sistemas de Información

Servicios Web

Agradecimientos: Mario Muñoz Organero <munozm>, Simon Pickin de IT/UCIIIM <spickin>

Índice

- Qué son
- Para qué sirven
- ¿Qué contienen?
- ¿Cómo se usan?
- Arquitectura
- Algunos estándares
  - Descripción del servicio: WSDL
  - Intercambio de mensajes: SOAP
  - Publicación y descubrimiento de serviciosUDDI
- Proceso de estandarización
- Ejemplos (próximo día)
- Conclusiones y bibliografía

Algunas cifras (*muy* aproximadas)

- La compañía de investigación de mercado IDC estima que la inversión de las empresas americanas en proyectos de servicios web ascenderá a los *25 mil millones de dolares* en el año 2008.

- La compañía de investigación de mercado Gartner pronostica que las empresas estadounidenses derrocharán *mil millones de dolares* en proyectos equivocados de servicios web antes del 2007.

CIO Magazine, edición del 1 de octubre del 2003
`http://www.cio.com/archive/100103/standards.html`

Servicios Web: ¿Qué son?
Definición simple

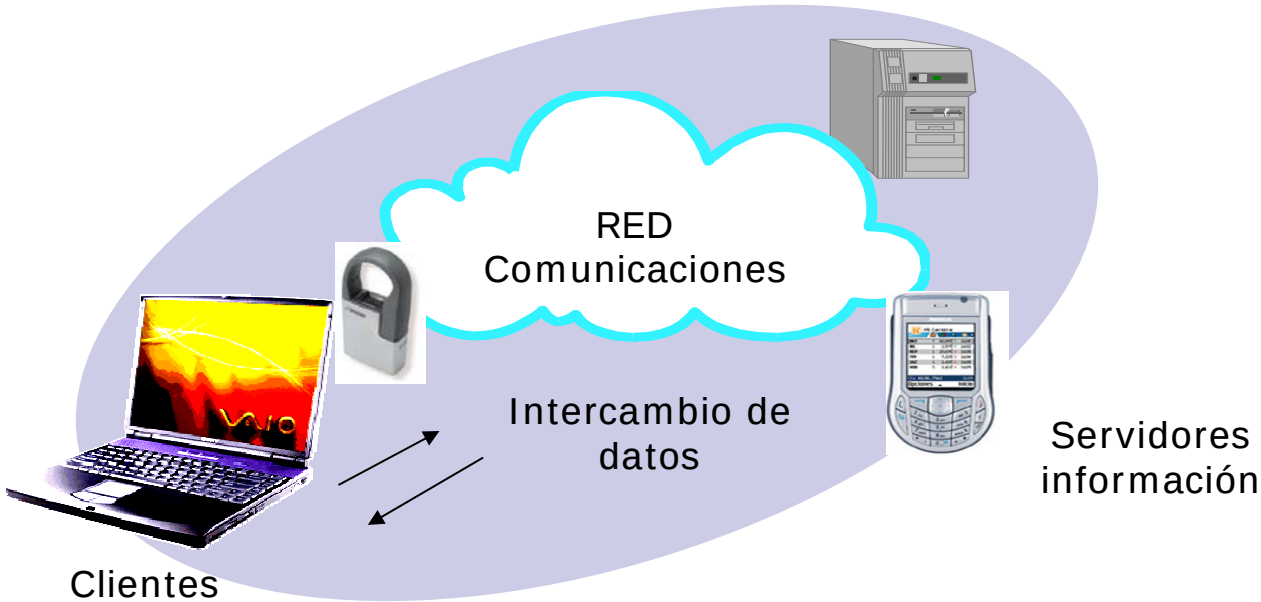
- Un *servicio web* es un componente programable que proporciona un servicio y es accesible por Internet.

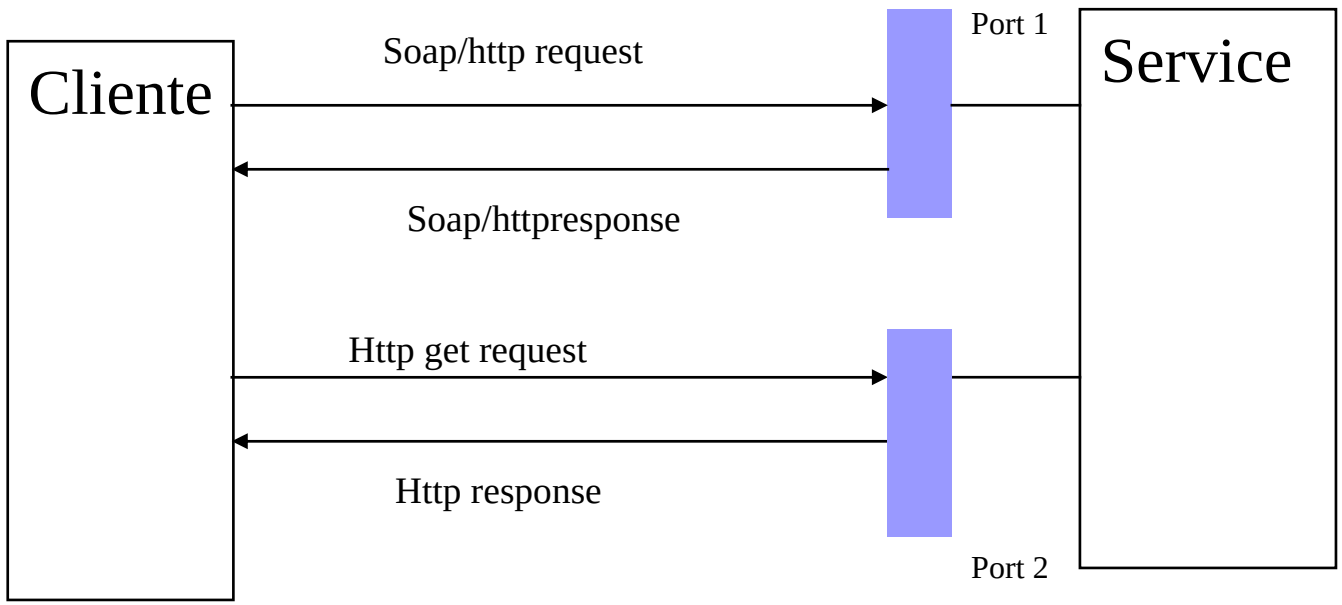
- Los servicios web pueden funcionar de manera independiente o pueden estar conectados entre sí para proporcionar una funcionalidad mayor.

> “Introduction to Web Services”, Embedded.com
> **`http://www.embedded.com/story/OEG20020125S0103`**

Servicios Web ¿Para qué sirven?

- Permiten que varias aplicaciones
  - Compartan información
  - Invoquen funciones de otras aplicaciones independientemente de:
    - Cómo hayan sido creadas (lenguaje de programación)
    - Cómo se ejecutan (sistema operativo y plataforma)
    - Dispositivos utilizados para acceder a ellas
- Crean una especie de WWW paralela de carácter cibernético
  - WWW humana (personas accediendo a pags web)
  - WWW cibernética (aplicaciones accediendo a servicios Web)


RED
Comunicaciones
Intercambio de
datos
Servidores
información
Clientes


Cliente
Soap/http request
Port 1
Service
Soap/httpresponse
Http get request
Http response
Port 2

Servicios Web: ¿Qué contienen?

- Un servicio Web consta de los siguientes 3 elementos
  - **el servicio**: software que puede procesar ciertos documentos XML bien definidos recibidos mediante alguna combinación de protocolos de transporte y de aplicación
  - **el documento XML**: contiene toda la información específica de la aplicación que el consumidor del servicio envía al servicio para procesar
  - **la dirección**: una dirección de red junto con un *binding* de protocolo que pueden utilizarse para acceder al servicio
- En la práctica, hace falta un cuarto elemento:
  - **la envoltura**: un protócolo de encapsulación de mensajes que separa el documento UML de la otra información que se quiere intercambiar

"Web Services are not distributed objects"
`http://weblogs.cs.cornell.edu/AllThingsDistributed/archives/000343.html`

Servicios Web: ¿Qué contienen?

- El software que implementa el servicio
  - puede o no ser OO
  - puede o no operar como parte de un servidor web
  - puede o no ser un *front-end* de una aplicación grande
- Los documentos XML intercambiados
  - tienen que poder validarse e interpretarse por los actores
  - tienen que ser conformes a una especificación
- La dirección
  - identifica donde se puede encontrar el servicio con un protocolo particular (p.e. TCP, HTTP, SMTP,...)
- La envoltura
  - puede utilizarse por un intermediario para añadir información tal como información de encaminamiento o de seguridad

Servicios Web: ¿Qué son?
Definición completa (1/3)

- Los servicios Web son un mecanismo de comunicación distribuida que permiten que las aplicaciones compartan información y que además invoquen funciones de otras aplicaciones independientemente de cómo se hayan creado las aplicaciones, cuál sea el sistema operativo o la plataforma en que se ejecutan y cuáles los dispositivos utilizados para obtener acceso a ellas.

Servicios Web: ¿Qué son?
Definición completa (2/3)

- Los servicios Web son un sistema Software diseñado para soportar la interacción máquina a máquina sobre la red.
- Definen un formato procesable por ordenador de la descripción de las interfaces (WSDL).
- La interacción se realiza mediante el intercambio de mensajes a través de la red usando formatos y protocolos como SOAP, HTTP o MIME.

Servicios Web: ¿Qué son?
Definición completa (3/3)

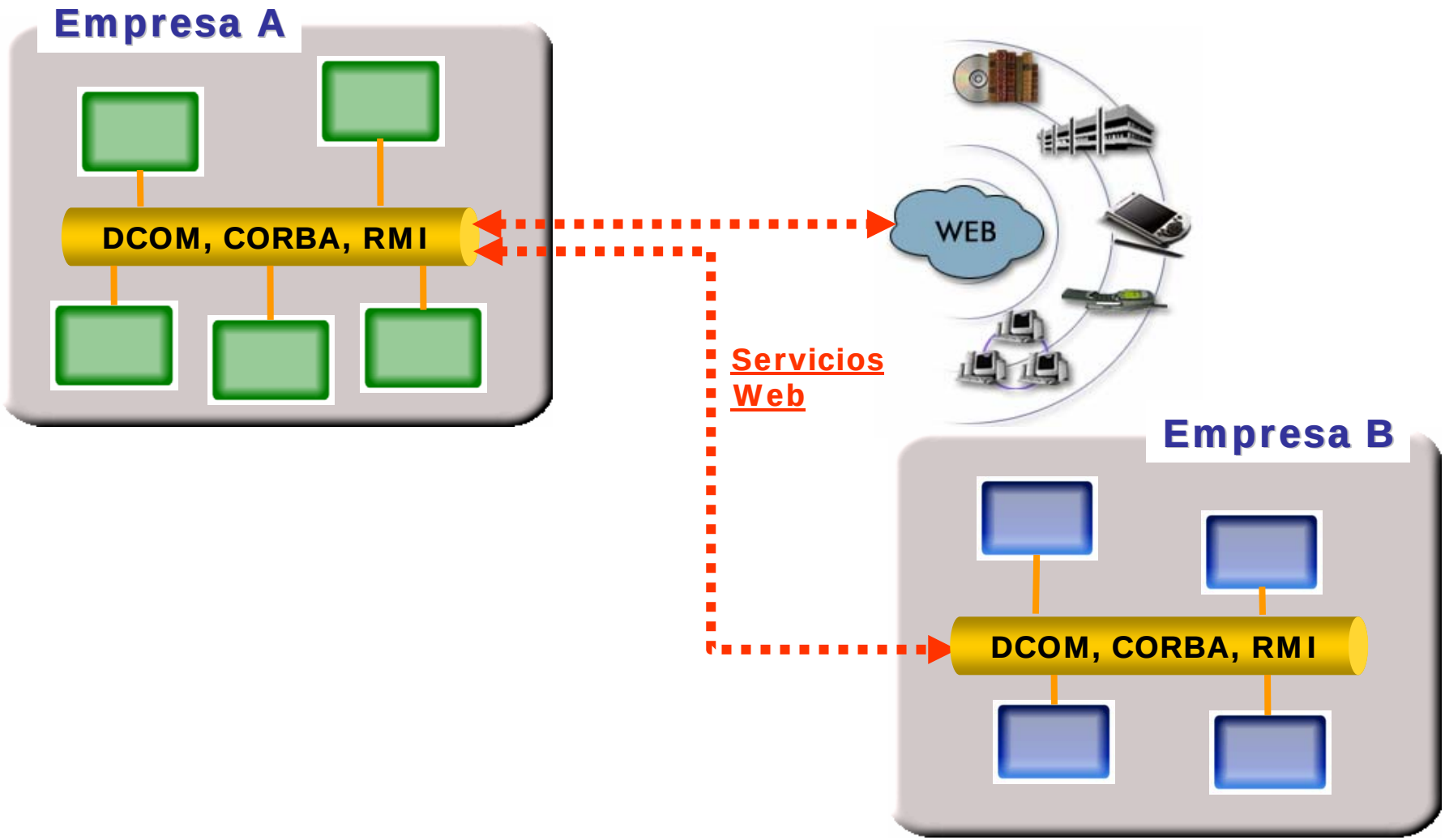
- Componentes que ejecutan procesos o funciones de negocio significativas.
- con una interfaz claramente definida y accesible a través de Internet.
- basada en el intercambio de documentos electrónicos en formato XML.
- y que pueden ser combinados entre sí.

Servicios Web: ¿Cómo se usan?

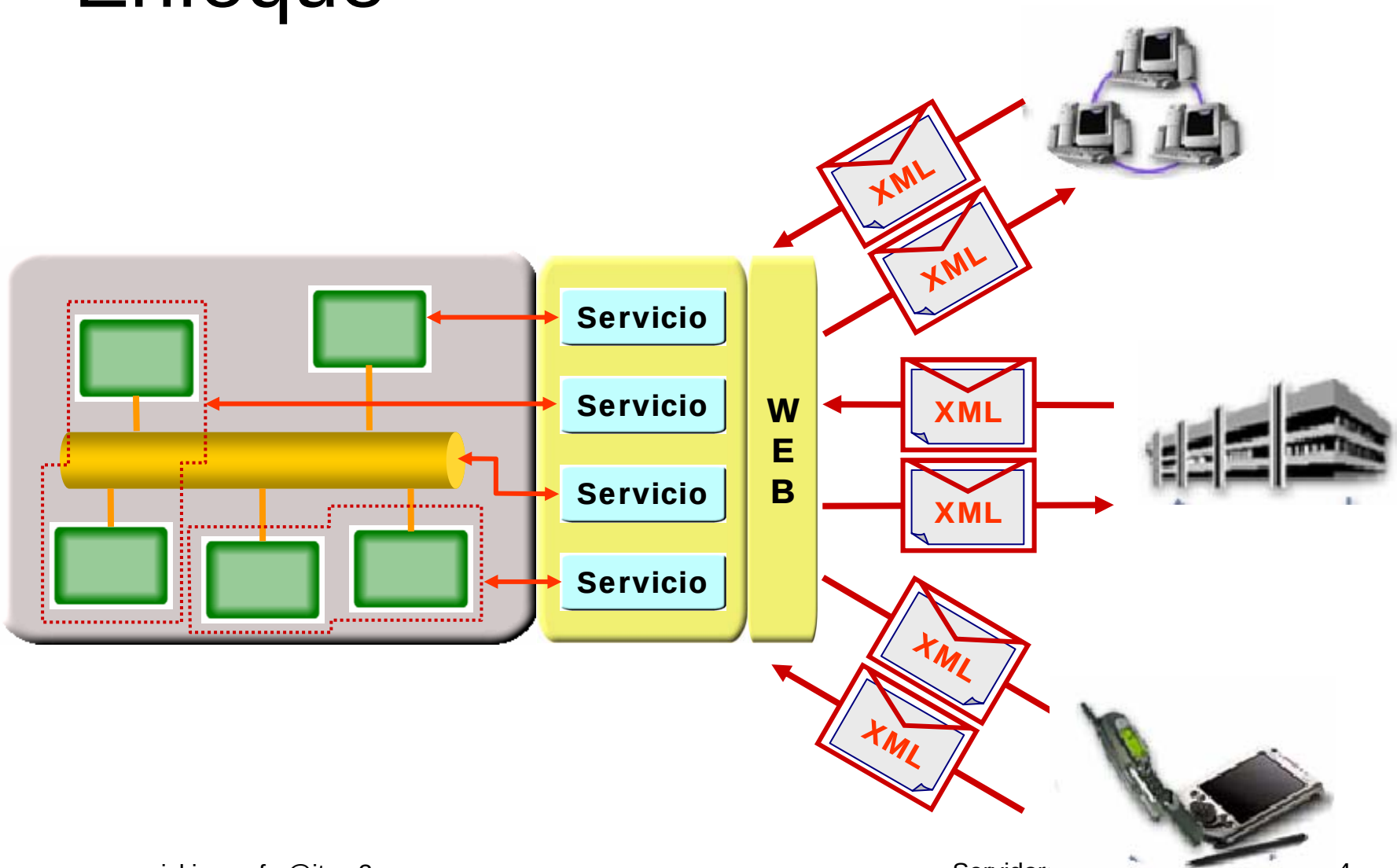
Fases para crear Servicio Web

- **Crear la lógica del negocio del Servicio Web.** Diseñar e implementar la aplicación que contenga la lógica de negocio del servicio Web (cualquier lenguaje, plataforma y sistema operativo)
- **Desplegar el servicio en un servidor**
  - Instalar y configurar el servidor
  - Ubicar la aplicación en el lugar adecuado
  - Convertir la aplicación en un servicio Web proporcionando descripción **WSDL**.
  - Registrar el servicio web en un directorio (opcional) usando **UDDI**
- **Desarrollar la aplicación cliente que accede al servicio Web**.
  - El cliente usa protocolo de mensajería XML para acceder al WS
  - 2 tendencias (**RPC** vs **SOAP**) incompatibles entre sí y de distinta complejidad.
  - Discusión sobre cual usar : http://bulmalug.net/body.phtml?nIdNoticia=503
    - RPC + sencillo, fácil de aprender y unix friendly
    - SOAP + potente y completo. Soporta varios conjuntos de caracteres (US-ASCII, UTF-8 y UTF-16)

Enfoque


Empresa A
DCOM, CORBA, RMI
WEB
Servicios
Web
Empresa B
DCOM, CORBA, RMI

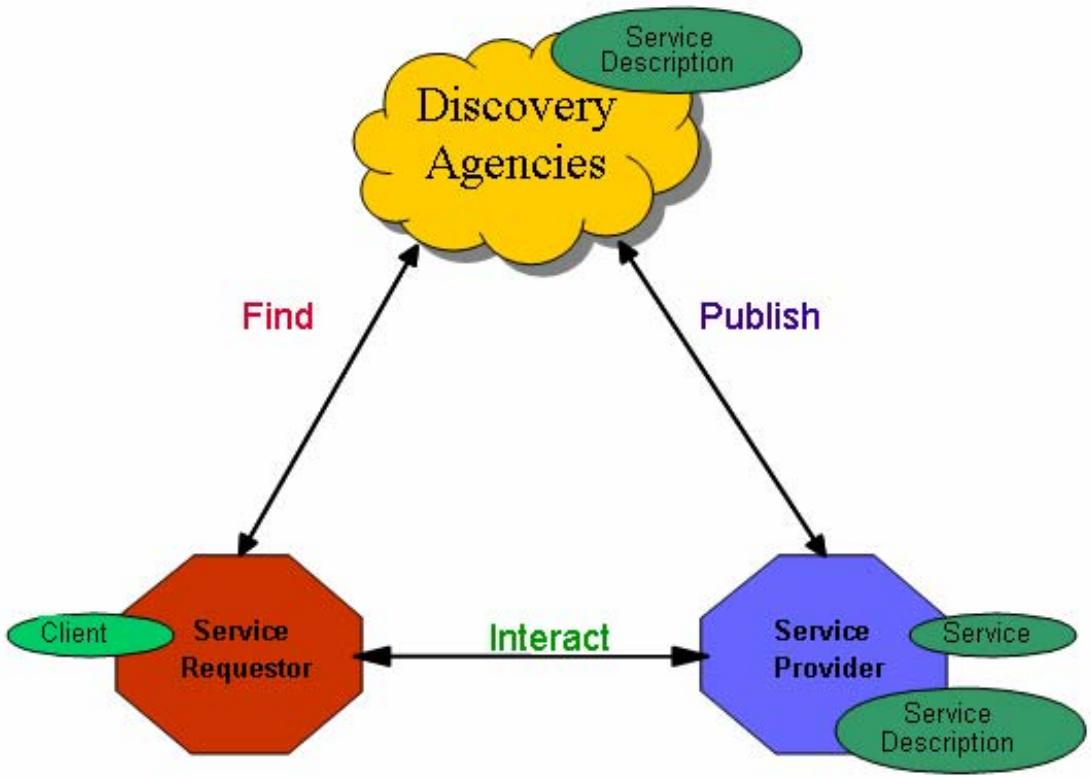
Enfoque


Servicio
Servicio
Servicio
Servicio
W
E
B
XML
XML
XML
XML
XML
XML

Service Oriented Architecture (SOA)

- Roles:
  - **proveedor del servicio**: ofrece un servicio y publica su definición en un registro junto con una descripción
  - **consumidor del servicio**: localiza e invoca el servicio
  - **registro o agencia de descubrimiento**: ofrece un servicio de información sobre la definición y descripción de servicios disponibles
- Características de SOA:
  - sistema es una colección de servicios débilmente acoplados
  - comunicación con el servicio por protocolos estándares
  - mecanismo común para la representación e intercambio de datos
  - lenguaje de meta-datos para describir los servicios ofrecidos
  - mecanismo para registrar y localizar los servicios
- Mismo servicio, distintas calidades de servicio (QoS)
  - distinto proveedor, descripción habla de distintos requisitos técnicos
    - disponibilidad, prestaciones, escalabilidad, seguridad,…
- Visión del consumidor:
  - concierne únicamente la funcionalidad y la QoS

Service Oriented Architecture (SOA)

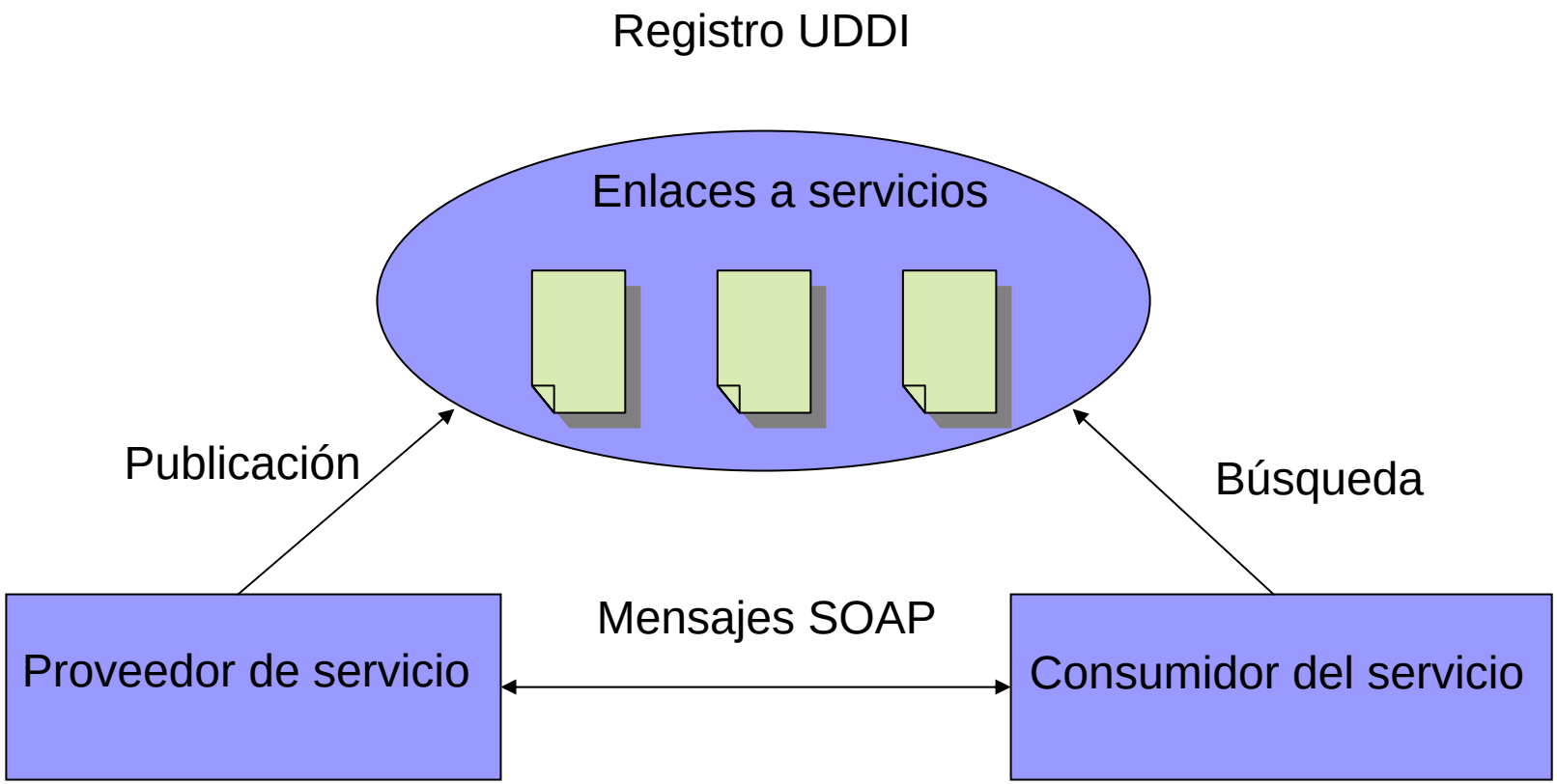


Partes de la arquitectura

- Descripción de los servicios: WSDL
- Registro y búsqueda de servicios: UDDI
- Uso de los servicios: SOAP, HTTP, MIME

Directorio: Publicar & encontrar Servicios:	**UDDI**
Inspeccion: Encontrar servicios en servidor:	**DISCO**
Descripcion: Descripciones formales:	**WSDL**
Bits a transmitir: Interacciones de servicio:	**SOAP**
Formato universal de datos:	**XML**
Comunicaciones:	**Internet**

Partes de la arquitectura



Tecnologías básicas

- XML
  - de uso obligatorio
- SOAP
  - ampliamente utilizada
- WSDL
  - suele utilizarse
- UDDI
  - empieza a utilizarse

XML: DTDs

- Algunos problemas con el uso de DTDs:
  - no se escriben ellos mismos con XML
  - no se puede poner restricciones sobre datos de tipo carácter
  - modelo de valores de atributos (enumeraciones) demasiado simple
  - no soporta *XML Namespace*
  - soporte limitado para modularidad y reuso (via entidades)
  - no soporta evolución, extensión y herencia de declaraciones
  - declaraciones de atributos y contenidos no pueden depender de contexto de atributos o de elementos (¡esquemas tampoco!)
  - no hay valores por defecto para elementos (en esquemas, sólo pueden ser datos de tipo carácter)
  - no se puede especificar “cualquier elemento” o “cualquier atributo”
  - valores por defecto no se pueden especificar separadas de las declaraciones (¡esquemas tampoco!)
  - …

XML: Esquemas

- XML Schema:
  - define una clase de documentos XML
  - **documento instancia**: un documento XML conforme con una esquema XML (*XML Schema*) particular
    - una instancia no tiene por qué referenciar una esquema
- Construcciones principales de una esquema:
  - **declaración de elemento**: asocia un nombre de elemento a un tipo
  - **definición de tipo complejo**: define requisitos para atributos, sub-elementos y datos de tipo carácter (*character data*) en elementos de este tipo
    - **declaraciones de atributos**: describen los atributos que pueden o deben aparecer.
    - **referencias de elementos**: describen los sub-elementos que pueden o deben aparecer, así como cuantos y en qué orden.
  - **definición de tipo sencillo**: define un conjunto de cadenas que se pueden utilizar como valores de atributos o datos de tipo carácter

XML: el estándar XML Schema

- Especificación:
  - *Primer* (~ tutorial oficial):
    `http://www.w3.org/TR/xmlschema-0/`
  - Estructuras:
    `http://www.w3.org/TR/xmlschema-1/`
  - Tipos de datos:
    `http://www.w3.org/TR/xmlschema-2/`

Descripción del servicio

WSDL: presentación básica

- WSDL (*Web Services Description Language)*
  - es una especificación que define cómo describir servicios Web usando una gramática XML.
  - utiliza XML y XML Schema
- El documento WSDL representa un contrato entre el proveedor y el usuario de un servicio.
  - Consumidor y proveedor del servicio quieren validar y interpretar los documentos que intercambian
  - los dos necesitan acceso a una descripción del servicio
- WSDL se usa:
  - En la fase de despliegue para crear los interfaces de servicio.
  - Algunas implementaciones SOAP, también usan WSDL durante la ejecución para soportar comunicaciones dinámicas
- Actualmente, la mayoría de las herramientas usan WSDL 1.1

Descripción del servicio

WSDL: presentación básica

- Es una gramática XML, orientada a describir en forma estructurada, la funcionalidad de un Web Service y la forma en que esa funcionalidad se hace disponible.
- Describe un servicio, como una colección de "**comunication endpoints**" (puertos) capaces de intercambiar mensajes.
- Cada port tiene un definición abstracta (**port type**) y una definición concreta (**binding**).
- Permite describir en forma abstracta operaciones y mensajes, prescindiendo de las especificaciones de protocolo y tipos de datos.
- Vincula las descripciones abstractas a una implementación concreta de protocolos y tipos de datos, permitiendo la reutilización de las definiciones abstractas.
- Es extensible tanto en lo que respecta a tipos de datos (XSD) como a protocolos y formatos de mensajes.
- Proporciona documentación sobre el servicio que describe

Descripción del servicio

WSDL: Componentes del servicio

- El fichero WSDL tiene 4 partes principales que indican **qué** hace el servicio, qué **tipo** de datos utiliza en sus mensajes, **cómo** se comunica y **donde** reside
  - **QUÉ**: Proporciona información sobre la interfaz exportada. Define el mensaje.Consta de elementos **message**, y **portType**,
  - **TIPO**: Define los tipos de datos soportados por los mensajes intercambiados (peticiones, respuestas, errores, etc.). Consta de elementos **types**
  - **CÓMO**: Describe los detalles de la implementación técnica de nuestro Servicio Web. Indica cómo transportar los mensajes en la comunicación. Consta de elementos **binding**, Un binding une un portType a un protocolo de comunicación (Ej. SOAP sobre HTTP).
  - **DONDE**:Indica la localización del servicio. Consta del elemento **service**, y coloca juntos el **porttype**, el **binding**, y la **localización** real (una URI) del Servicio Web

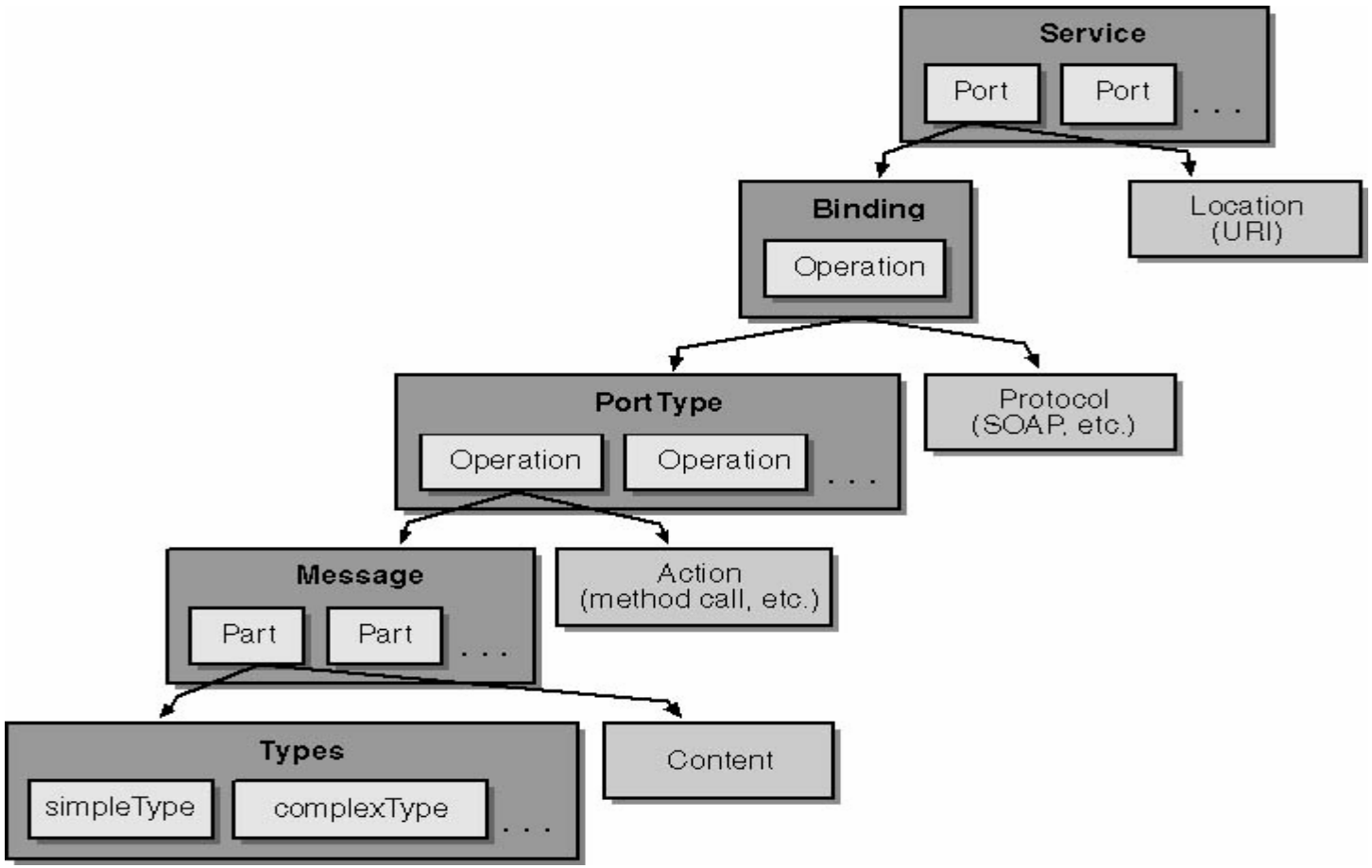
Descripción del servicio

WSDL: Componentes del servicio

- **definitions** = elemento raíz del documento WSDL. Se usa para declarar los espacios de nombres.
- **types** = se usa para describir tipos de datos para los mensajes intercambiados.
- **message** = descripción de los datos que van a ser transmitidos. Son una colección de “data values” de un tipo particular (utilizando XML Schema como mecanismo de tipación).
- **portType** = Colección de “operations” o “signatures” de los métodos que definen el intercambio ordenado de los mensajes.
- **bindings** = especifica los protocolos usa cada “port” y el “encoding”.
- **port** = “Port type” + “Binding”. Es un descripción de una acción soportada por el servicio. Cada operación se corresponde a un mensaje de input o de output
- **service** = Conjunto de “ports” relacionados que implementan el servicio.

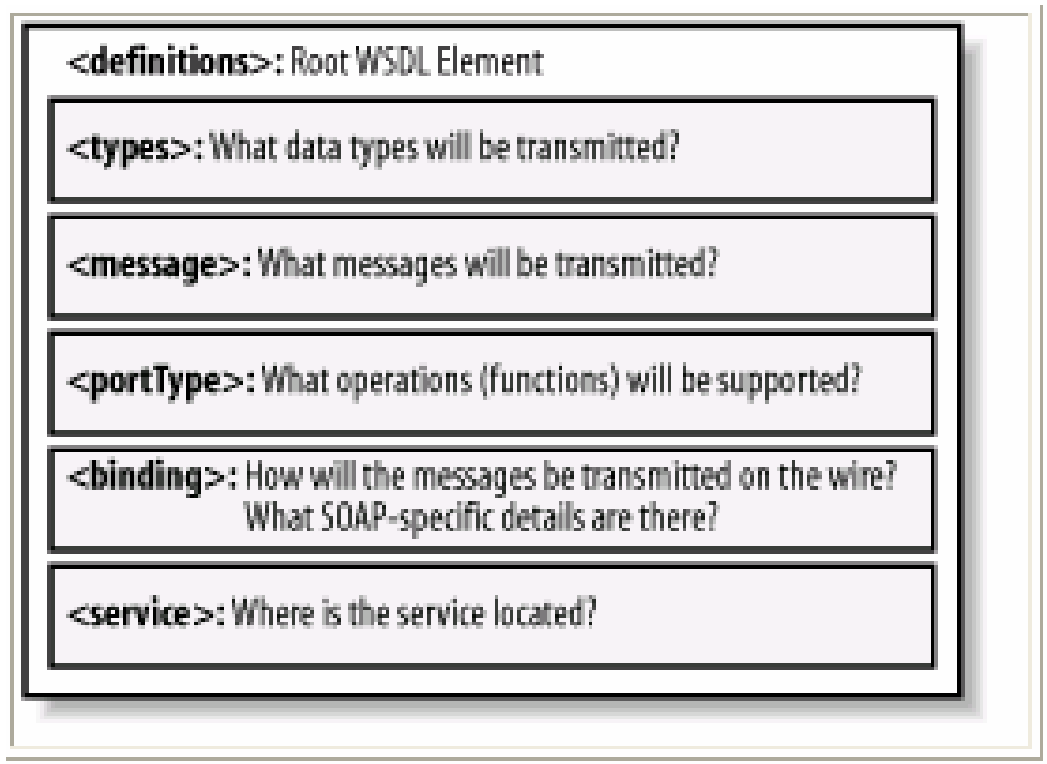
Descripción del servicio

WSDL: Componentes del servicio



Descripción del servicio

WSDL: Componentes del servicio



Descripción del servicio
WSDL: el estándar

- Especificación:
  - *Primer* (~ tutorial oficial):
    `http://www.w3.org/TR/wsdl20-primer/`
  - Nucleo del lenguaje:
    `http://www.w3.org/TR/wsdl20/`
  - Extensiones predefinidas:
    `http://www.w3.org/TR/wsdl20-extensions/`
  - *Bindings*:
    `http://www.w3.org/TR/wsdl20-bindings/`

Intercambio de mensajes

Protocolos en servicios web

- Los servicios Web se invocan mediante protocolos basados en XML
- 2 tendencias (RPC vs SOAP)
  - ambos son lenguajes de mensajería basados en XML
  - Son incompatibles entre sí se diferencian en complejidad
  - Discusión: http://bulmalug.net/body.phtml?nIdNoticia=503
- RPC diseñado para sencillez
  - Curva de aprendizaje rápida
  - Unix friendly
- SOAP diseñado para potencia (+ completo y minucioso)
  - Curva de aprendizaje lenta
  - Soporta varios conjuntos de caracteres: US-ASCII, UTF-8 y UTF-16
- El proyecto Apache tiene un módulo SOAP, que se puede usar:
  - como una librería cliente para invocar servicios SOAP disponibles en cualquier lugar
  - como una herramienta del lado del servidor para implementar servicios SOAP accesibles
  - Es lo que vamos a utilizar en prácticas

SOAP

- .
- .
- .

Intercambio de mensajes
SOAP: Presentación básica

- Es un protocolo de mensajería XML extensible. Proporciona un mecanismo para que una aplicación pueda enviar mensajes XML a otra
  - originalmente: *Simple Object Access Protocol*
  - SOAP 1.2: ya no es acrónimo: nada que ver con objetos
    - algunos leen *SOA protocol*; SOA = *Service Oriented Architecture*
- Es un protocolo de alto nivel define:
  - la estuctura del mensaje
  - algunas reglas para su procesamiento
- Es independiente del protocolo de transporte (**HTTP**, FTP, TCP, SMTP, POP3, JMS, MQSeries)
- HTTP es el protocolo de transporte más utilizado en la actualidad

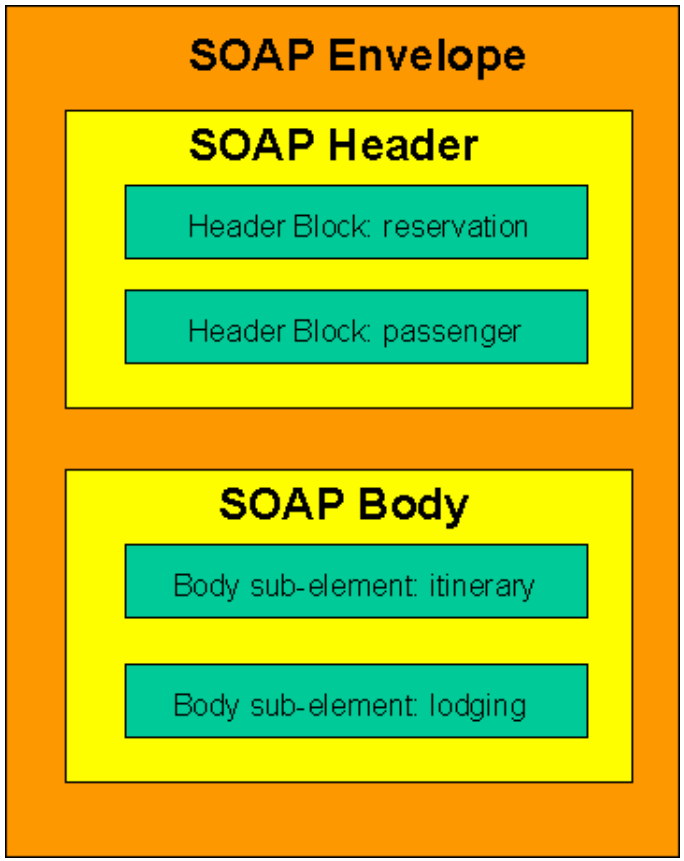
Intercambio de mensajes
SOAP: Presentación básica

- Un mensaje SOAP es una transmisión de una-via desde un emisor SOAP a un receptor SOAP.
- Cualquier aplicación puede participar en este intercambio como emisor o receptor
- SOAP es fundamentalmente un protocolo sin estado y unidireccional.
  - Los mensajes unidireccionales de SOAP se pueden combinar para soportar patrones de interacción más complejos ej:
    - solicitud/respuesta
    - solicitud/respuesta múltiple
    - mensajería asíncrona de una-vía
    - notificación.
  - Para hacerlo utiliza:
    - propiedades del protocolo subyacente
    - información específica de la aplicación

Intercambio de mensajes
SOAP: Componentes

- El contenido del mensaje está encerrado en el envelope
- Una envoltura (envelope) contiene las siguientes partes:
  - **Header (cabecera)**: contiene información de sistema (facultativo). almacena información de contexto (por ejemplo, contexto de transación, credienciales de seguridad, etc.).
  - **Body (cuerpo)**: Contiene el mensaje propiamente dicho, almacena el documento XML que el receptor final procesará. Hay varios estilos para esta sección
  - **Fault (fallo)**: Es un mecanismo adicional para manejo de errores. Se usa cuando se produce un error. Contiene información sobre la naturaleza del fallo.

Intercambio de mensajes
SOAP: Envoltura



Representación gráfico de un ejemplo de un mensaje SOAP

Intercambio de mensajes
SOAP: Protocolo subyacente

- Distintos protocolos subyacentes posibles
  - la segunda parte del estándar proporciona un ej. de *binding* para el protocolo HTTP
  - también puede utilizarse encima de SMTP, TCP, UDP,…

Intercambio de mensajes
SOAP: Otros aspectos importantes

- Tiene una noción de “fallos” (SOAP *faults*)
  - mecanismo de excepciones
- Representación XML de datos, dos enfoques:
  - *literal*: enfocado en XML; conforme con una esquema XML. No tiene reglas específicas salvo las de XML
  - *encoded*: enfocado en el lenguaje de programación; inferir el XML de los tipos de datos del lenguaje según las reglas proporcionado en la segunda parte del estándar
- Uso de RPC con SOAP, dos enfoques:
  - utilizar la representación por defecto como documentos
  - utilizar la representación SOAP-RPC con datos *encoded*.

Intercambio de mensajes
SOAP: Otros aspectos importantes

- La especificación SOAP recomienda (no obliga) al estilo codificado *(encoded).*
- Permite los siguientes tipos de datos:
  - Tipos primitivos (int, float, double, String)
  - Define reglas para construir tipos complejos (arrays, estructuras, etc.)

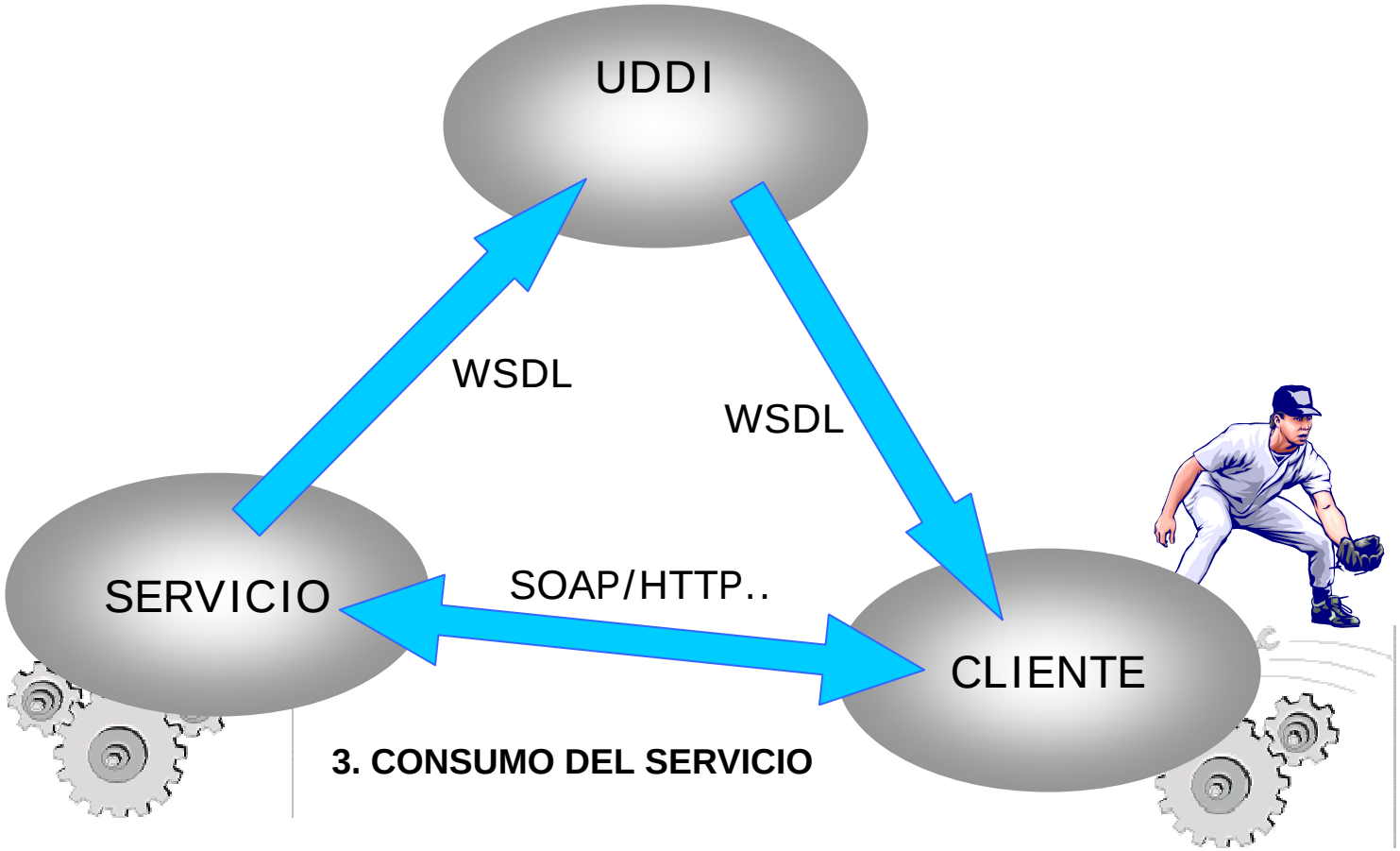
Intercambio de mensajes
SOAP: El estándar

- Especificación:
  - *Primer* (~ tutorial oficial):
    `http://www.w3.org/TR/soap12-part0/`
  - Estructura de mensajes:
    `http://www.w3.org/TR/2003/REC-soap12-part1-20030624/`
  - Adjuntos:
    `http://www.w3.org/TR/2003/REC-soap12-part2-20030624/`

Publicar y encontrar servicios
UDDI: presentación básica

- Para facilitar el uso de servicios web, hace falta un mecanismo
  - para publicar información sobre servicios web
  - para descubrir los servicios que están disponibles
- Descubrimiento, Descripción e Integración Universales
  - *Universal Discovery Description and Integration* (UDDI)
  - utiliza XML / XML Schema
  - comunicación por SOAP
  - uso de UDDI por un servicio Web es facultativo

Publicar y encontrar servicios
UDDI: arquitectura



UDDI

- Implementa la funcionalidad de “discovery” necesaria para poder encontrar la descripción WSDL del WebService que se necesita.
- UDDI tiene 2 partes:
  - Un directorio con los metadatos de todos los WebServices, incluyendo un puntero a la descripción WSDL de cada uno.
  - Las definiciones de “port types” WSDL para manipular y buscar en ese directorio.

UDDI

- Define estándares para un registro distribuido de servicios Web
- Puede verse como la suma de:
  - **White pages** (información general): Contiene dirección e información de contacto del proveedor del servicio
  - **Yellow pages** (categorías de servicios): Permiten buscar servicios por categorías y clasificaciones
  - **Green pages** (reglas de negocio): Contienen suficiente infomación técnica sobre los servicios ofrecidos para permitir su invocación

Páginas blancas

- Información sobre un proveedor de servicios:
  - Nombre
  - Descripción en texto del proveedor
  - Información de contacto
    - Teléfono
    - Dirección postal
  - Otros nombres (ticket para el IBEX-35…).

Páginas amarillas

- Categorías de negocios.
- En UDDI V1 se definen 3 taxonomías:
  - NAICS (North American industry codes)
  - UN/SPSC (ECMA product/services codes)
  - Geographical information
- Estas se extienden en UDDI V2.

Páginas verdes

- Se describe la información necesaria de cómo interactuar con alguien:
  - Entidades de negocio
  - Descripciones de servicios
  - Información de binding
- Este directorio o registro es definido como una jerarquía de entidades “business”, “service” y “binding” cuyas descripciones son expresadas en XML.

Publicar y encontrar servicios
UDDI: modelo de información y APIs

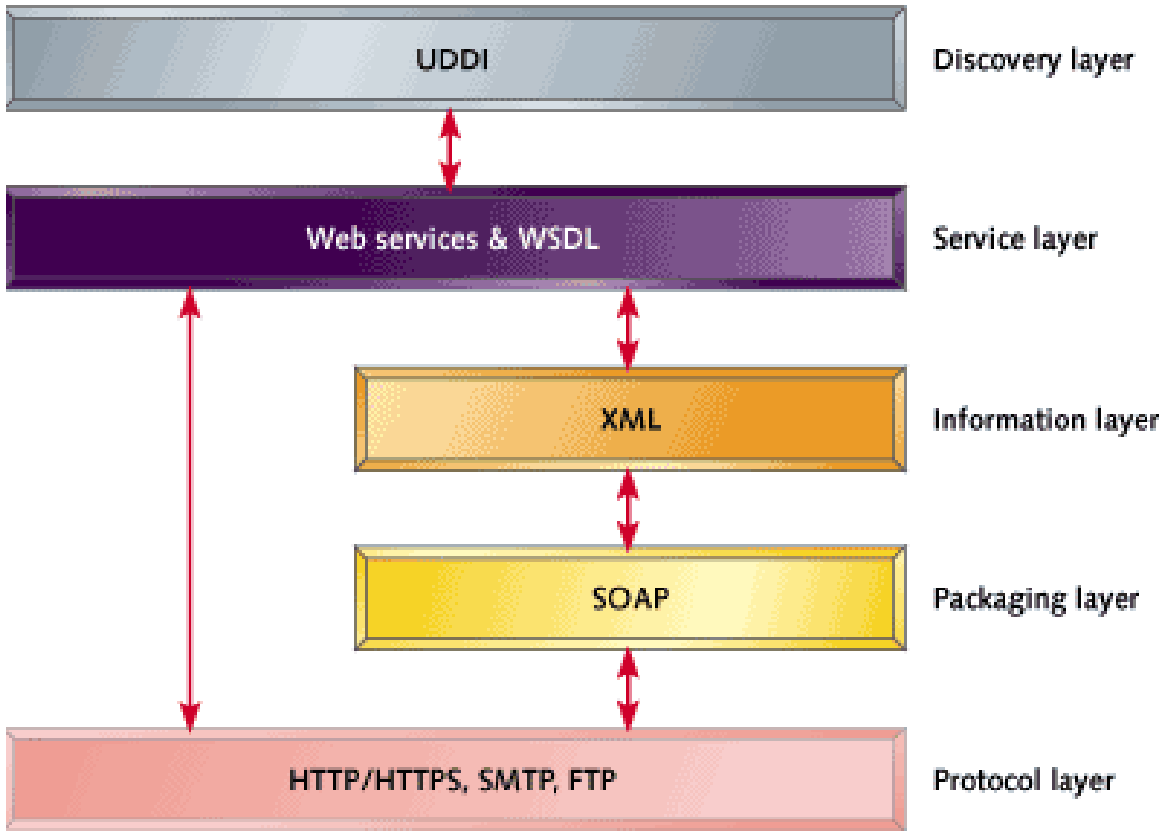
- El modelo de información UDDI contiene 4 elementos:
  - **información de negocio** (**`businessEntity`** element):
    describe la organización; incluye soporte para taxonomías de tipo páginas amarillas para facilitar busquedas.
  - **información de servicio** (**`businessService`** element):
    agrupa los servicios ofrecidos por una organización
  - **información de *binding*** (**`bindingTemplate`** element):
    describe cómo invocar un servicio
  - **información sobre especificaciones** (**`tModel`** element):
    representación abstracta de la especificación técnica de un servicio; un **`tModel`** tiene nombre, organización que lo publica y URIs que apuntan a las especificaciones (WSDL,…) de los servicios.
- Dos APIs:
  - para publicar
  - para buscar

Publicar y encontrar servicios
UDDI: el estándar

- Múltiples documentos accesibles en:

  `http://www.oasis-open.org/committees/uddi-spec/doc/tcspecs.htm`

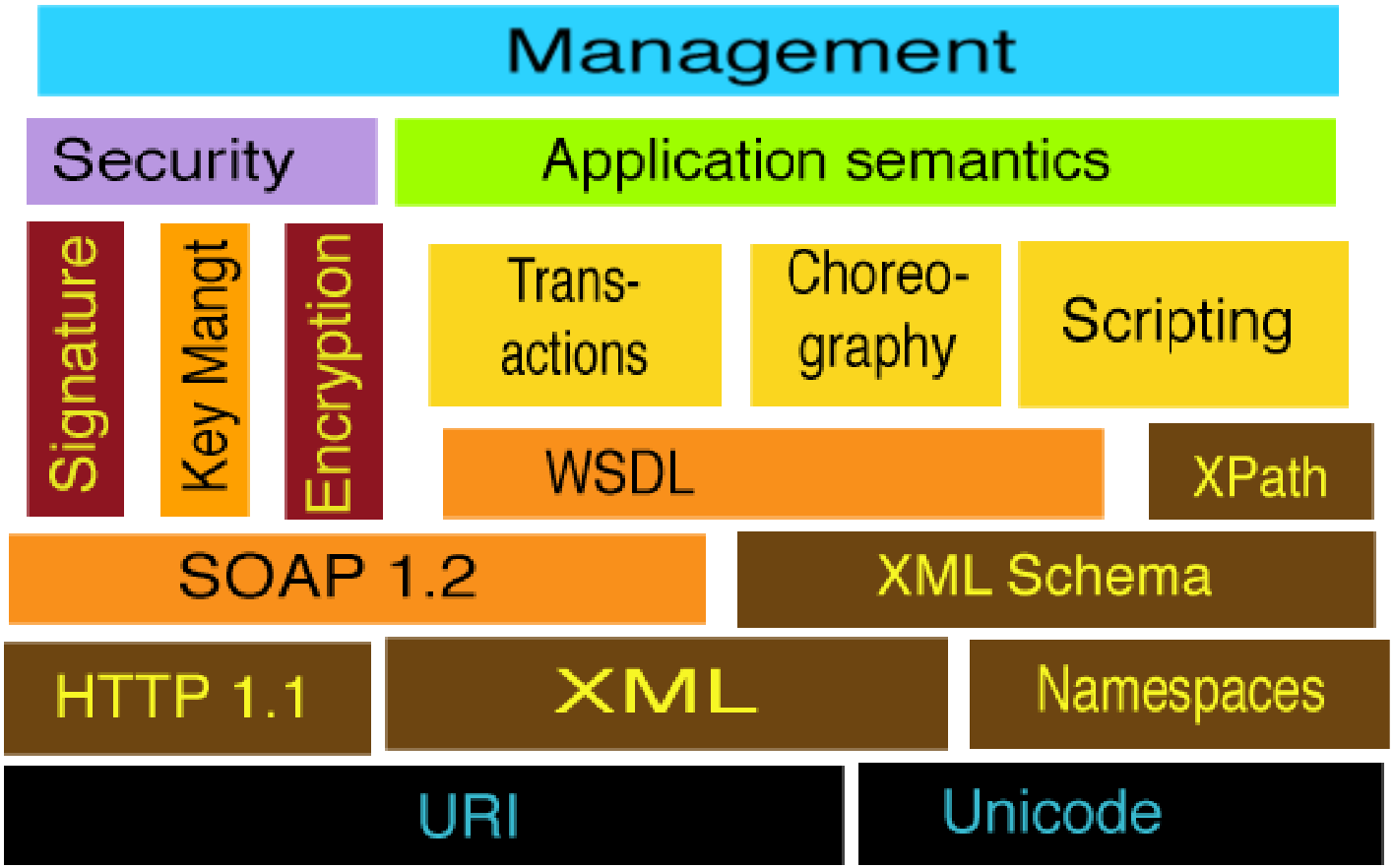
Tecnologías de los servicios web



Web services stack

según “Introduction to Web Services”, Embedded.com

Tecnologías de los servicios web



Web services stack extendido

según **`http://www.w3.org/DesignIssues/WebServices.html`**

Organismos de estandarización: W3C: `http://www.w3.org/`

- 1994: fundación del *World Wide Web Consortium*
  - por Tim Berners-Lee, inventor del Web
- Cometido: definición de estándares para la Web:
  - HTTP, HTML, CSS,…
- Ahora no solo estándares de infraestructura, también:
  - familia XML: XML, XSL, XLL, Xpath, Xpointer, XML Schema,...
  - Web Services: SOAP, WSDL,...
  - ...
- Web Services activity: **`http://www.w3.org/2002/ws/`**

Organismos de estandarización:

OASIS: `http://www.oasis-open.org/`

- 1993: fundado como *SGML-Open*
- Cometido original:
  - trabajar sobre la interoperabilidad entre productos SGML
- 1998: nuevo nombre, OASIS (*Organization for the Advancement of Structured Information Standards*)
  - para reflejar el nuevo alcance de su trabajo técnico, que incluye en particular, XML y estándares relacionados

Organismos de estandarización:

WS-I: `http://www.ws-i.org/`

- 2002: fundación del *Web Services Interoperability Organization*
  - por Microsoft, IBM y siete compañías más (Sun no incluido)
- Cometido:
  - crear, promocionar y dar soporte a protocolos genéricos para el intercambio interoperable de mensajes entre servicios Web.
  - proporcionar perfiles (guías para el uso conjunto de especificaciones de distinto índole para servicios Web), aplicaciones de demostración y herramientas de pruebas

Organismos de estandarización:
Liberty Alliance: `http://www.projectliberty.org/`

- 2001: fundado por Sun y otras compañias
- Cometido
  - desarrollar especificaciones para servicios Web para la gestión de identidad, utilizando el estándar SAML (*Security Assertion Markup Language*) de OASIS

Estandarización: historia

1998 *W3C* publica **XML 1.0**

Microsoft empieza el desarrollo de SOAP

Userland/Dave Winer publica XML-RPC basado en SOAP

1999 *Microsoft* publica **SOAP 1.0**

2000 *Microsoft* y *IBM* publica **SOAP 1.1** y la presenta al *W3C*

Microsoft, *IBM* y *Ariba* forman la *UDDI initiative* y publican las primeras especificaciones de **UDDI** y **WSDL**

2001 *W3C* publica **XML Schema 1.0**

Consorcio liderado por *Microsoft* e *IBM* publica **WSDL 1.1** y la presenta al *W3C*; *W3C* empieza a trabajar en WSDL 1.2

Estandarización: historia

2002 *UDDI initiative* (en la fecha compuesto por un gran número de compañías) publica **UDDI v2**

UDDI initiative se asocia a *OASIS*

UDDI initiative publica **UDDI v3**

2003 **UDDI v2** aprobado como estándar *OASIS*

W3C publica **SOAP 1.2**; hace uso importante de XML Schema

2005 WSDL 1.2, ahora renombrado **WSDL 2.0**, cerca de aprobación como estándar *W3C*

UDDI v3 cerca de aprobación como estándar *OASIS*

Estandarización: algunas dificultades

- **Feb 2002**: Microsoft y IBM y siete companías más fundan WS-I, excluyendo a Sun
  - Sun llama a la WS-I “un gobierno en la sombra para estándares”
- **Finales de 2002**: Microsoft e IBM llevan su especificación BEPL (*Business Process Execution Language*) a OASIS después de haberla presentado al W3C
  - aparentemente, estaban descontentos con la intención del W3C de considerarla junto con la especificación rival *Web Services Choreography Interface* de un consorcio liderado por Sun
- **Enero 2003**: Sun, Oracle and Sonic presenta la especificación "Web Services Reliability” a OASIS; Microsoft e IBM publica la especificación rival "WSReliability".
- **Presente**: Según Sun, la especificación WS-Security de Microsoft e IBM solapa con el trabajo de Liberty Alliance.

Estandarización: algunas dificultades

- La politica del W3C sobre patentes, royalties,... :
  - toda especificación que se convierte en estándar debe estar libre de pago y de otras restricciones
  - compañías que presentan especificaciones para consideración tienen que declarar sus intenciones con respecto a licencias y patentes antes del voto
- Preferencia de Microsoft e IBM por OASIS:
  - algunos dicen que se debe a la pólitica menos rigurosa de esta organización en lo que respecta a patentes y royalties
  - otros dicen que están desesperados con la lentitud del W3C en lo que respecta a la estandarización de WSDL

Concepciones erróneas comunes

- **Servicios Web son como objetos distribuidos** 

  - ninguna noción de objetos, ni de referencias de objeto, factorías, ciclos de vida, etc.
- **Servicios Web es RPC para internet** 

  - en el caso general, no se asocia ninguna semántica predefinida a los contenidos de los documentos XML enviados
  - envolver un objeto orientado a sesiones como un servicio Web puede tener problemas con respecto a controlar el ciclo de vida a través de interacciones con el cliente, el uso de URIs en vez de referencias de objeto, gestión de estado, etc., véase:

    http://www.iona.com/hyplan/vinoski/pdfs/IEEE-Web_Services_Interaction_Model_Part_1.pdf

"Cuando solo tienes un martillo, todo parece a un clavo"... Mucha gente implicada en servicios Web hoy en día solo tiene un martillo RPC en su caja de herramientas, y parece que son incapaces, o no quieren, considerar otras posibilidades.

http://www.iona.com/hyplan/vinoski/pdfs/IEEE-Web_Services_Interaction_Models_Part_2.pdf

Concepciones erróneas comunes

- servicios Web necesitan HTTP 

  - se suele utilizar HTTP pero se puede utilizar SMTP, TCP, UDP,...
- servicios Web necesitan servidores Web 

  - se puede aprovechar la funcionalidad de servidor de aplicaciones que tienen los servidores Web pero no es necesario
- servicios Web son fiables porque utilizan TCP 

  - documento puede llegar al primer nodo pero no al destino final
- Para más detalles consultar:

  `http://weblogs.cs.cornell.edu/AllThingsDistributed/archives/000343.html`

Web Services
Ejemplos

- Un ejemplo de un mensaje SOAP:

```
<SOAP-ENV:Envelope
 xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
   <SOAP-ENV:Body>
      <m:GetLastTradePrice xmlns:m= "http://wombat.ztrade.com">
        <symbol>SUNW</symbol>
      </m:GetLastTradePrice>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

- Para más ejemplos SOAP, véase el SOAP primer
  http://www.w3.org/TR/soap12-part0/
- Para ejemplos WSDL 1.1, véase el documento:
  http://www.w3.org/TR/wsdl/
- Para ejemplos WSDL 2.0, véase el WSDL primer
  http://www.w3.org/TR/wsdl20-primer/

Conclusiones

- Los servicios Web típicos utilizan:
  - elementos: XML, SOAP, WSDL y UDDI
  - HTTP
- Los servicios Web tienen buen futuro asegurado pero:
  - hay mucha información incompleta o incorrecta sobre ellos
  - la estandarización progresa lentamente por distintos razones
  - patentes / royalties etc. pueden ser un freno al desarrollo, puesto que se trata de buscar interoperabilidad y ubicuidad.
  - puede haber problemas de interoperabilidad debidos al uso de diferentes versiones de los estándares en distintas herramientas
  - falta un equivalente de muchos de los *object services* de CORBA

Bibliografía

- Libros
  - L/D 004.738.52 WEB, Web services: concepts, architectures and applications. Alonso, Gustavo
  - L/D 004.738.52 CER, Web services essentials. Cerami, Ethan
  - L/S 004.738.5.057.4 SNE, Programming Web services with SOAP. Snell, James
  - L/D 004.438 JAVA BUI, Building Web services with Java : making sense of XML, Soap, WSDL and UDDI. Graham, Steve
- Web (para practicar)
  - Introducción a los servicios Web en java:
    http://www.programacion.com/java/tutorial/servic_web/
  - Web Services: XML-RPC, SOAP, sobre PHP, Perl, y otros conceptos:
    http://www.programacion.com/tutorial/xmlrpcsoap/