Arquitectura de Software

Service Oriented Architectures

CSOF5204

SOA: Service Oriented Architecture

- Aproximación arquitectónica para solucionar problemas de EAI de última generación
- En SOA los recursos de software son **servicios** bien definidos, están disponibles y es posible encontrarlos
- SOA es una arquitectura diseñada para crear ambientes **organizados dinámicamente** de servicios interconectados que pueden ser **compuestos e interoperables**

SOA: Service Oriented Architecture

- Visión de la OMG: “[SOA] represents the best opportunity companies currently have to align their IT resources and business processes and make their systems more agile.”
- SOA busca alinear los procesos de negocio y los protocolos de servicios con los componentes subyacentes y las aplicaciones legado
- Para lograr éxito con SOA una empresa necesita entender como modelar procesos, servicios y componentes y como unir estos modelos de manera consistente

Servicios en SOA

- El bloque fundamental de una SOA son los servicios
- Cumplen un objetivo de negocio predeterminado
- Ejecutan unidades discretas de trabajo
- Son independientes
- No dependen del contexto o del estado de otros servicios
- Son compuestos de maneras específicas para crear aplicaciones

Ejemplo

- Una agencia es responsable de sacar papeles y licencias para carros y camiones (licenciamiento de vehículos)
- La agencia quiere ofrecer una aplicación a los vendedores de carros. Ésta debe permitir:
  - Licenciar un vehículo cuando es comprado
  - Calcular y pagar los impuestos de un vehículo
- Aproximación clásica
  - Los desarrolladores están a cargo de enlazar la lógica de negocio de administración de vendedores de carros, hacer el licenciamiento, y calcular y recolectar los impuestos en una sola aplicación
  - La aplicación debe acceder a datos de códigos de impuestos y vendedores de carros validos
    - Esta información está administrada por otras agencias y debe ser sincronizada con los datos locales regularmente

Ejemplo: Esquema de Desarrollo clásico

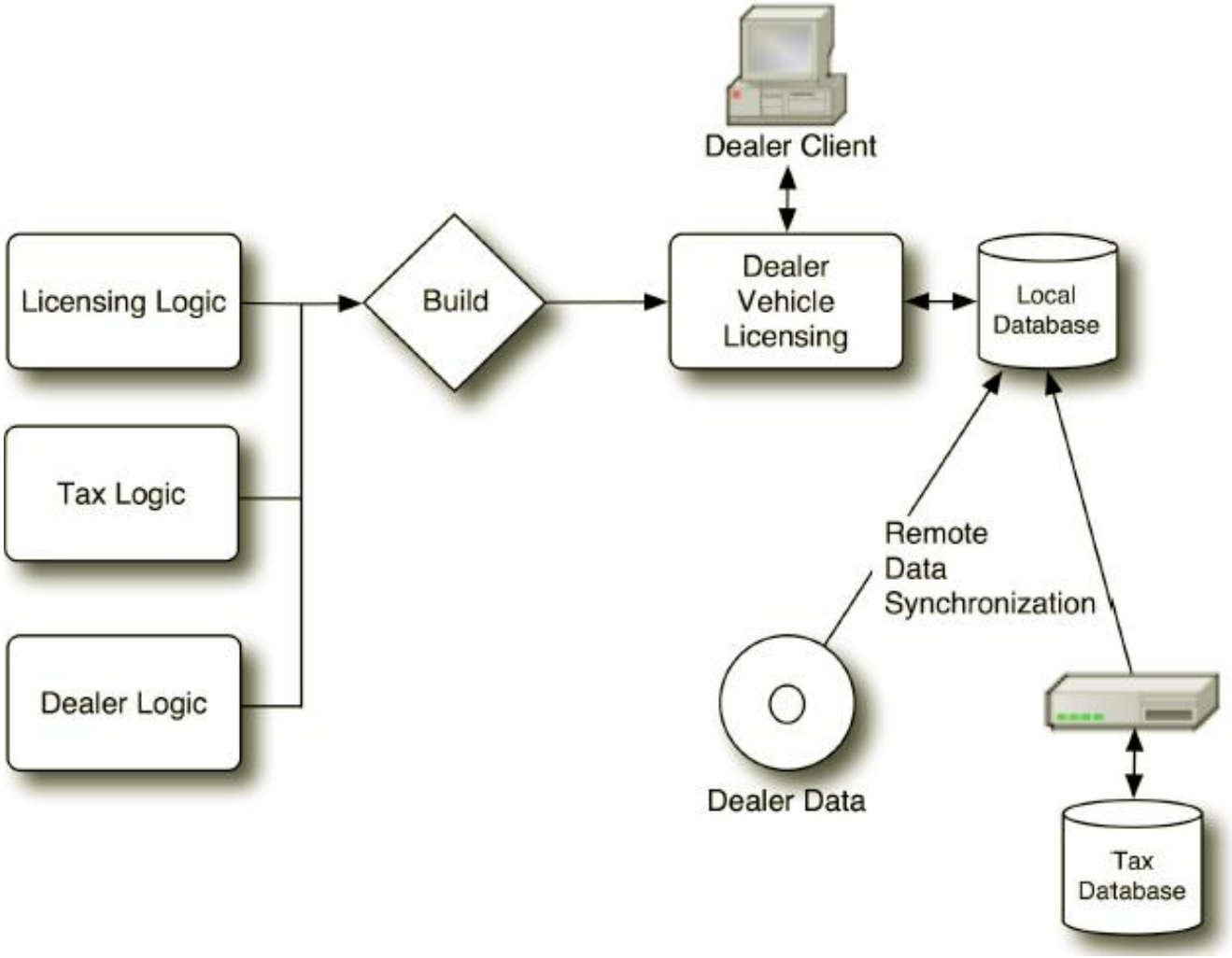


Imagen tomada de: [32] Service Oriented Architectures - Phillip J. Windley

Desventajas de esta aproximación

- En términos de lógica de negocio
  - La agencia que construye la aplicación está en el negocio de licenciar vehículos
  - No está en el negocio de regular los vendedores ni de recolectar impuestos
  - Los desarrolladores tendrán que incluir lógica de negocio asociada a estas tareas
- En términos de datos
  - Los datos que la agencia necesita para ejecutar su aplicación son administrados y mantenidos por otras empresas
  - La aplicación de licenciamiento requiere que se sincronicen los datos de los vendedores y las agencias de impuestos a menudo
    - Cada vez que cambien o cada cierto intervalo de tiempo?

Ejemplo: Aproximación SOA

- Aplicaciones como Integración de Servicios
- La empresa de licenciamiento se concentra en su negocio: licenciamiento
- Los datos y lógica de negocio asociados con los impuestos y vendedores son construidas y operadas por las empresas responsables de estas funciones
  - Son presentados a otras empresas como servicios que pueden ser accedidos cuando se necesiten
- Ventajas
  - Cada empresa es responsable de construir y operar los servicios asociados con sus procesos de negocio
  - Cada empresa mantiene los datos de sus entidades de negocio, no hay necesidad de sincronizar
  - Muchas empresas diferentes pueden usar los servicios de impuestos o de proveedores de carros

Ejemplo: Esquema de Desarrollo SOA

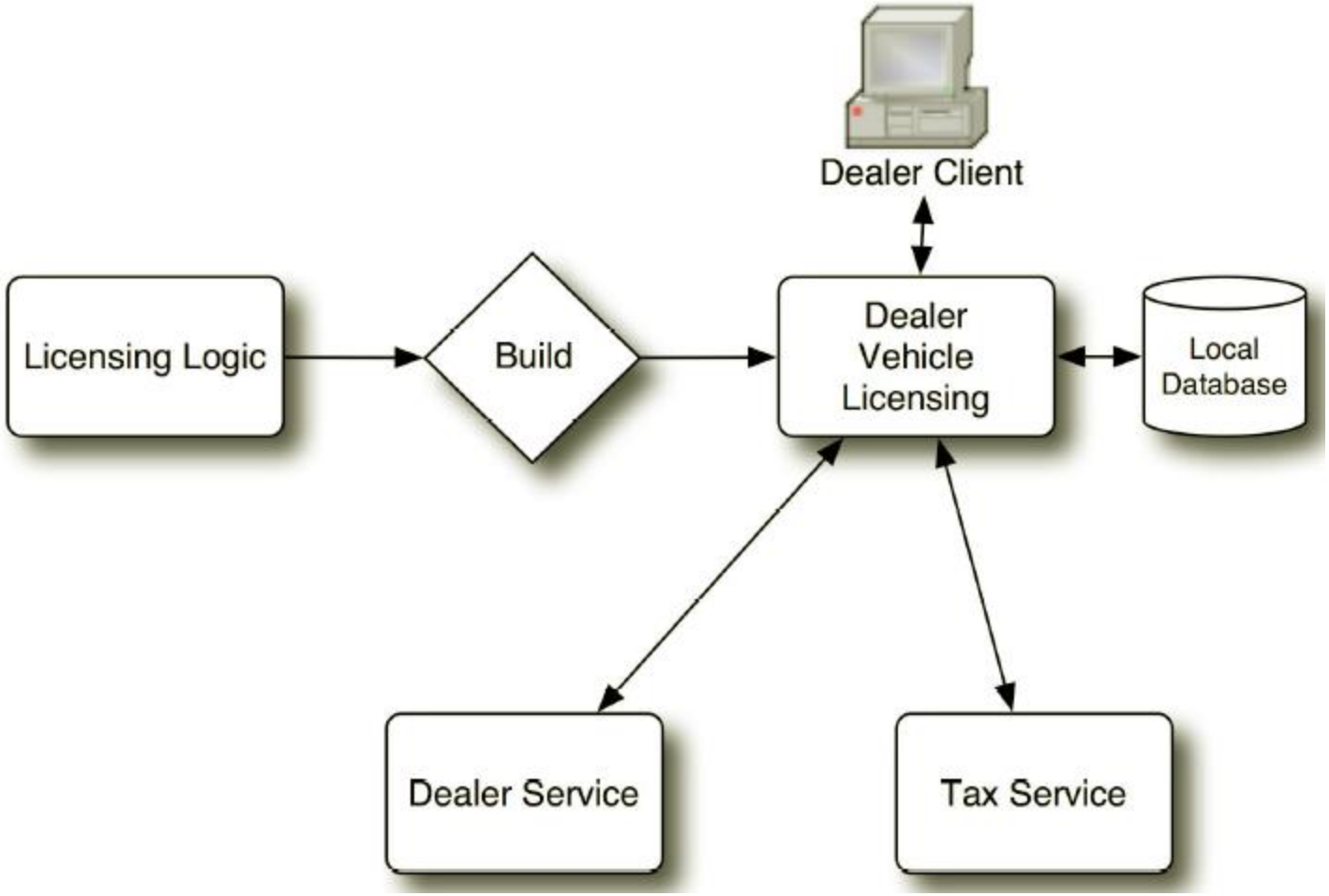


Imagen tomada de: [32] Service Oriented Architectures - Phillip J. Windley

Vista Arquitectónica SOA

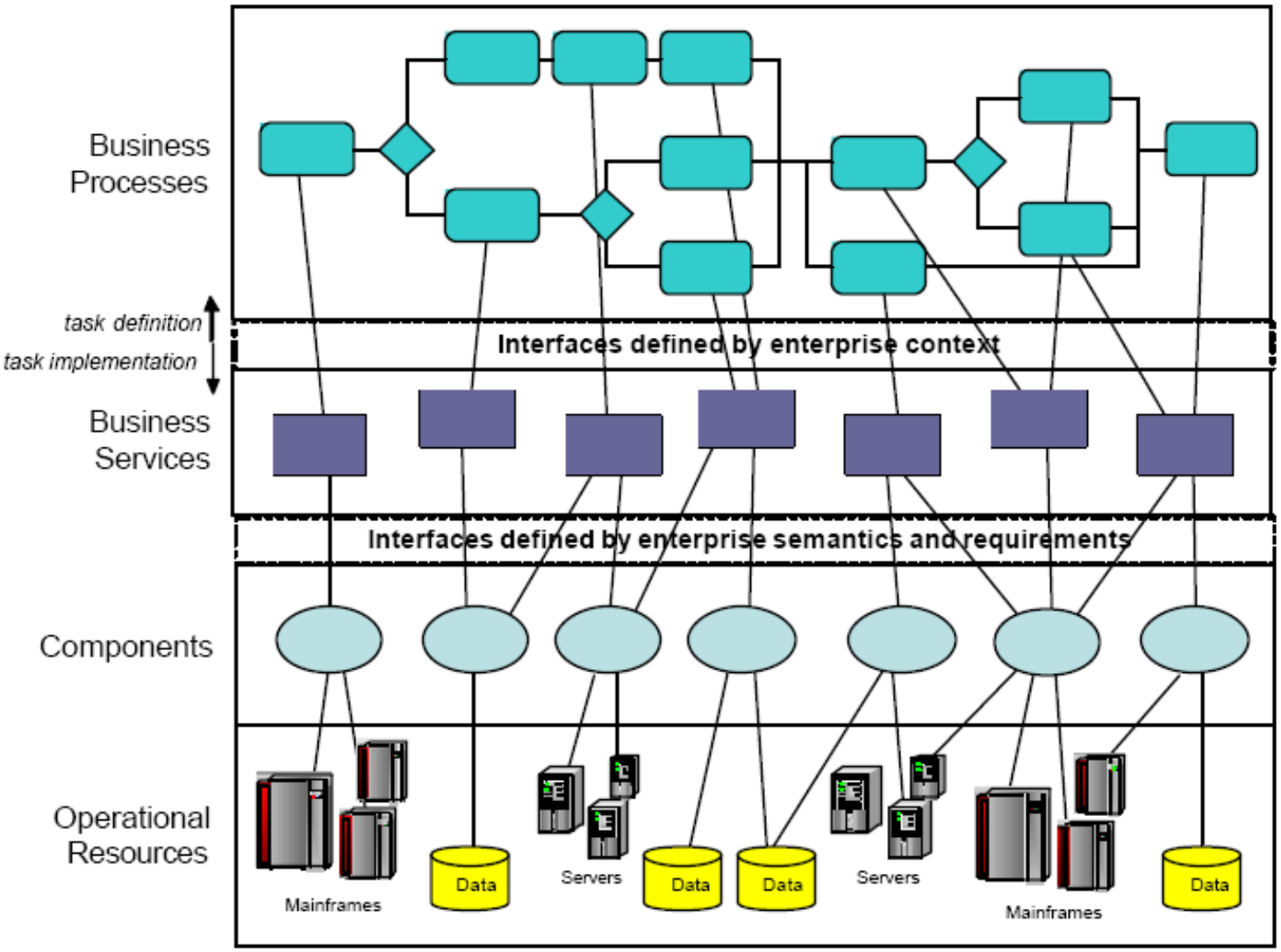
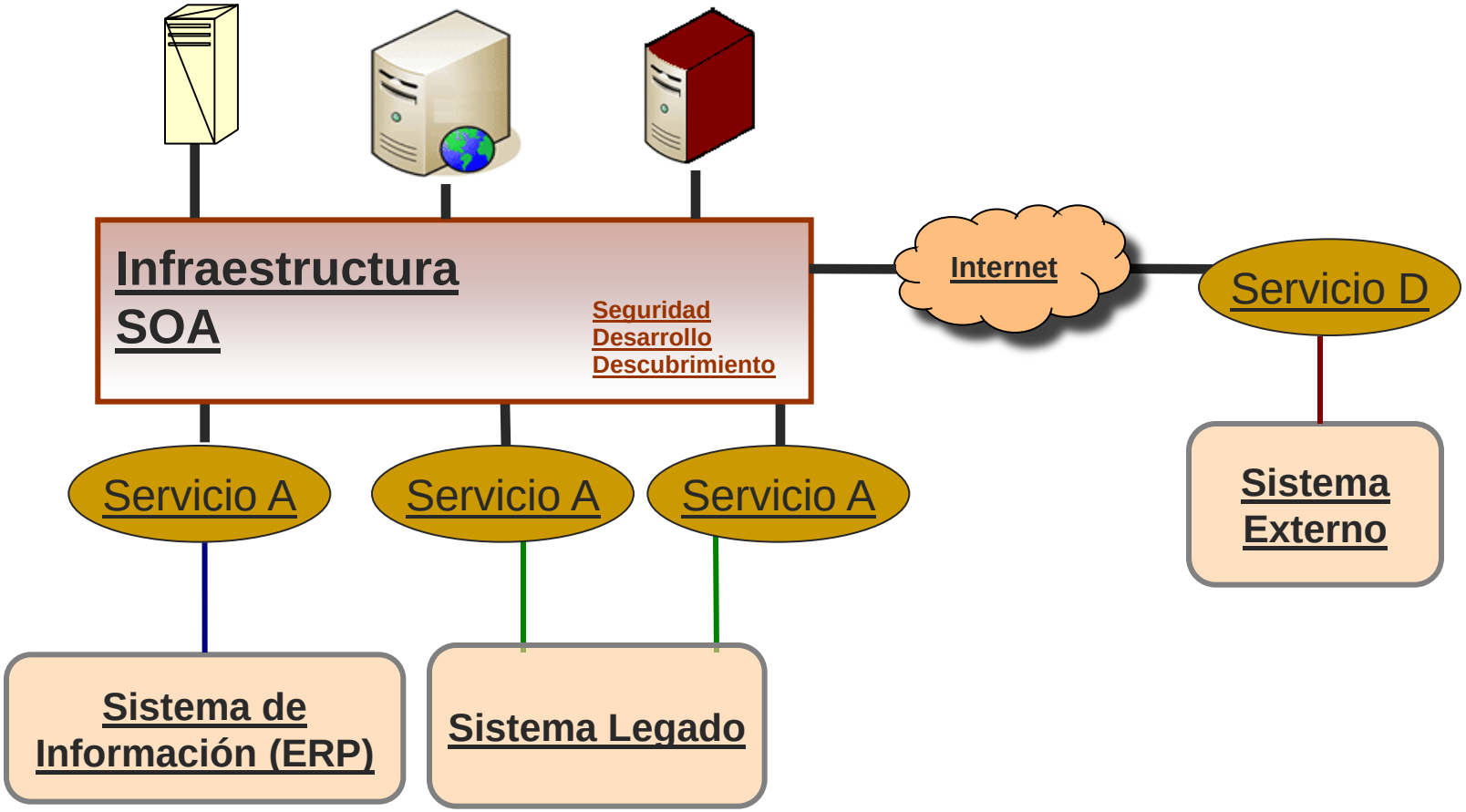


Imagen tomada de: The OMG and Service Oriented Architecture http://www.omg.org/attachments/pdf/OMG-and-the-SOA.pdf

Vista Arquitectónica SOA



Vista Arquitectónica SOA

- Aplicaciones que exponen servicios
- Sistemas de workflow que controlan el orden de la ejecución de los servicios
- Infraestructuras de soporte
  - Localización de servicios, paso de mensajes, RNF, etc.

Metáfora de documento

- Las SOA son caracterizadas por tener puntos de conexión (endpoints) simples y transportar datos ricos
- Modelos de interfaces simples que intercambian datos estructurados y ricos
- Las implementaciones actuales se basan en XML y pueden transportar datos complejos
- Documentos procesables por una máquina (p.e. XML) son usados para publicar interfaces de servicios
  - Estos documentos son enviados entre los endpoints para afectar el comportamiento general del sistema

Propiedades de los servicios

- Ubicables (Discoverable) y dinámicos
- Bajo acoplamiento (Loosely Coupled)
- De localización transparentes (Locationally Transparent)
- Interoperables
- Componibles
- Identificables en una red (Network-addressable)
- “Diversely Owned”
- Auto mantenibles (Self-healing)

Actores, Acciones y Artefactos en SOA

Detalles de la interface y su implementación:

- Tipos de datos
- Operaciones
- Información de enlace
- Localización en red

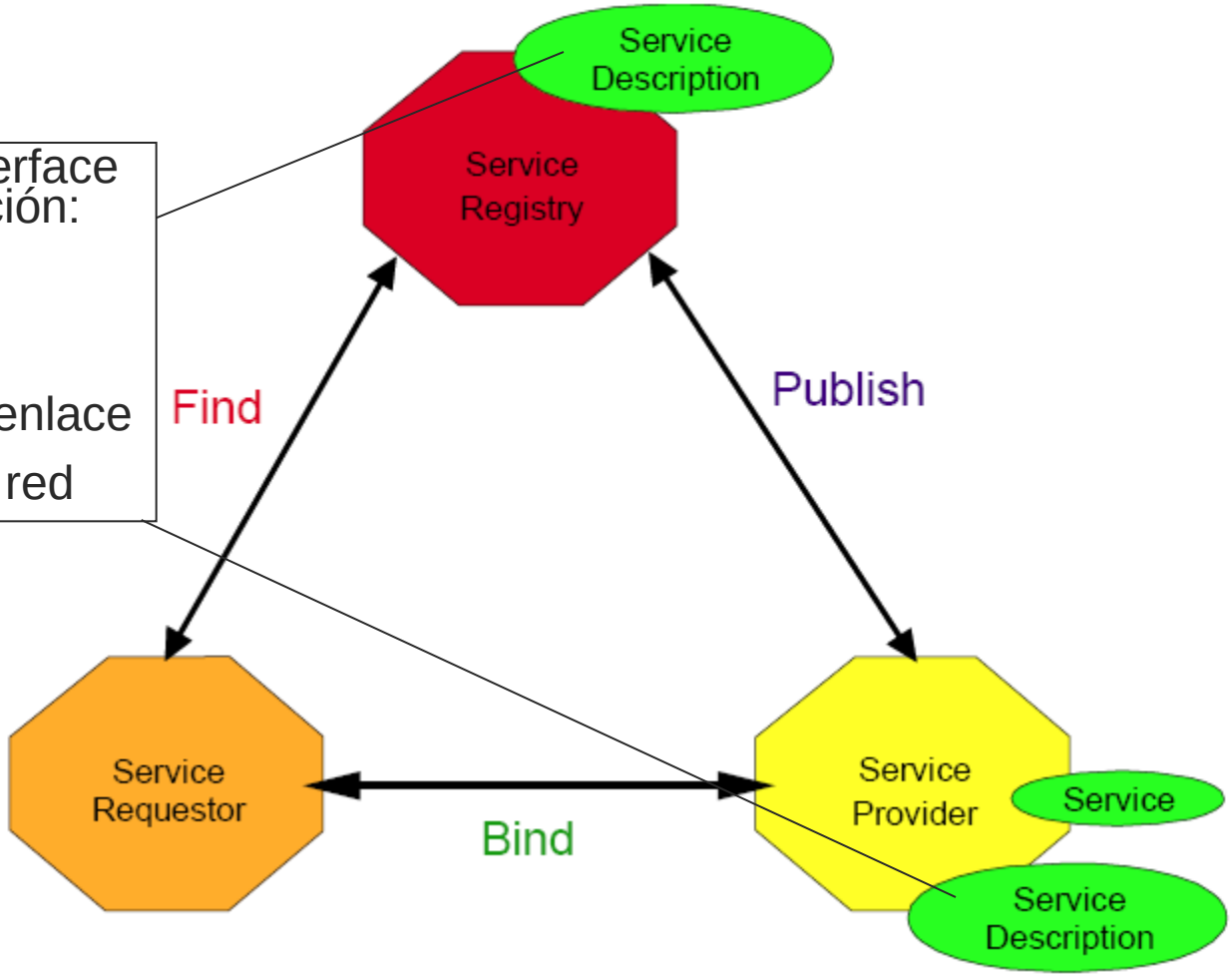


Imagen tomada de: [20] Heather KREGER. Web Services Conceptual Architecture. IBM Software Group. May 2001

Actores en SOA

- **Clientes (Service Requestors)**
  - Visión Negocio: Solicitan servicios, negocio que requiere un servicio
  - Vista Arq.: Aplicaciones o servicios que buscan e invocan servicios
- **Proveedores de Servicios (Service Providers)**
  - Visión Negocio: Publican servicios y los hacen disponibles a los clientes, dueños del servicio
  - Visión Arq.: Plataforma donde se ejecuta y da acceso al servicio
  - Aceptan solicitudes de clientes y las ejecutan
  - Son aplicaciones corriendo en una maquina accesible desde una red
- **Service Brokers**
  - Visión Negocio: Juntan a los proveedores y clientes. Ofrecen:
    - Registro de proveedores y sus servicios
    - Publicación de contratos de servicios (interfaces)
  - Conectan a los clientes con algún proveedor de servicio que ofrezca un contrato de servicio específico

Acciones en SOA

- **Publicar (publish)**
  - Para que un servicio sea usado, su descripción de servicio (Service Description) debe ser publicada
- **Descubrir / Encontrar (find)**
  - El cliente busca una descripción de servicio directamente, o hace una búsqueda de servicios
  - La operación se puede hacer en dos puntos del ciclo de vida del cliente: durante diseño o durante ejecución
- **Enlazar (bind)**
  - El cliente de servicio invoca o inicia la interacción con el servicio durante ejecución con los detalles en la descripción de servicio para localizar, contactar e invocar el servicio

Publicación y descubrimiento

- Publicar y descubrir son dos aspectos en los que SOA se diferencia de otras arquitecturas mas acopladas (como EJBs)
- Para usar un EJB es necesario conocer una interface para su uso en vez de adaptarse automáticamente como se puede con una interfaz auto descriptiva y publicada como las de SOA
- Interfaz publicada vs interfaz pública
  - Las interfaces públicas tienen un diseño público pero son solo conocidas por los clientes
    - Para cambiarlas es suficiente encontrar los clientes y cambiar la interfaz en cada cliente
  - Las interfaces publicadas pueden ser usadas por clientes sin que el operador del servicio tenga conocimiento o de permiso explícito
    - El contrato representado por la interface no es fácil de cambiar

Datos en SOA

- Cómo asegurar que las fuentes de datos serán útiles en una variedad de circunstancias?
- Los datos tienen importancia independientemente de una aplicación específica
- Cada recurso debe ser identificable de manera única en una red
- Las búsquedas sobre los datos deben hacerse de una manera transparente de localización
- Los datos retornados por una búsqueda deberían preservar la estructura de los datos originales
- Debe haber disponible en línea meta información sobre la estructura de los datos
- Deben existir mecanismos de traducción o transformación de datos de una estructura a otros tipos de estructuras y estilos de presentación
- Los datos deben retornarse en estructuras estándares

Ventajas de SOA

- Reutilización de código
  - Retorno de inversión
  - Correctitud
- "Productización"
  - Servicios como productos
- Seguridad
  - Servicios de autenticación generales
- Focalización de roles de desarrollo
  - Especialización por dominios o por capas
- Alineación con objetivos de negocio
  - Servicios enfocados en operaciones de negocio
- Desarrollo de aplicaciones centrado en configuración
  - Potenciando con workflows es posible crear aplicaciones completas ensamblando servicios
- Aumento de features
  - Proxies sobre servicios pueden aumentar características NF
- Mejor escalabilidad
  - Paralelismo posible por transparencia de localización
- Interoperabilidad tecnológica
  - Estándares proveen interoperabilidad

Retos en SOA

- Disponibilidad
  - Los brokers y mecanismos de comunicación son posibles cuellos de botella
- Sesiones / Transacciones
  - Introducen acoplamiento
  - Es necesario mantener el nivel de abstracción de los servicios por encima de las sesiones y transacciones
- Versionamiento
  - Los servicios deben estar versionados para asegurar actividades de control de cambios
- Documentación e instrumentación de desempeño
  - Determinar el desempeño de un servicio ayuda a los ensambladores de servicios a determinar niveles de servicios esperados
- Granularidad
  - Los servicios muy granulares pueden presentar cuellos de botella
- Ambientes
  - Configuraciones del ambiente estándar como el código de caracteres, parámetros de tiempo, etc.
- Auditoría / Debugging
  - Los servicios deberían hacer logging de su estado y sesiones para facilitar detección de errores

Web Services

CSOF5204

Algunas definiciones generales

"Un Web Service es como un tipo de sitio web que trata de remover el elemento humano. En lugar de una persona buscar una página específica . . . una pieza de software lo hace en su lugar"

WebServicesArchitects.com

"Son aplicaciones autodescriptivas que pueden descubrir y entablar contacto con otras aplicaciones web a través de la internet para completar tareas complejas"

Sun

"Componente de software que representa una función de negocio y puede ser accedido por otra aplicación (un cliente, un servidor u otro Servicio Web) sobre redes públicas utilizando protocolos y transportes generalmente disponibles y ubicuos"

Gartner Group

"El termino Web Service describe una manera estandarizada de integrar aplicaciones basadas en [tecnologias] web utilizando los estándares abiertos XML, SOAP, WSDL y UDDI . . . Permiten a las organizaciones comunicar datos sin un conocimiento detallado de los sistemas de TI detrás de sus firewalls. "

Webopedia

Web Services y SOA

- Web Services es un conjunto de estándares
- Los estándares son ampliamente usados para construir arquitecturas basadas en servicios
- **NO SON** la única implementación de SOA: no son equivalentes a SOA
- Una aplicación que exponga Web Services no es necesariamente parte de un sistema con una visión SOA

SOA y Web Services

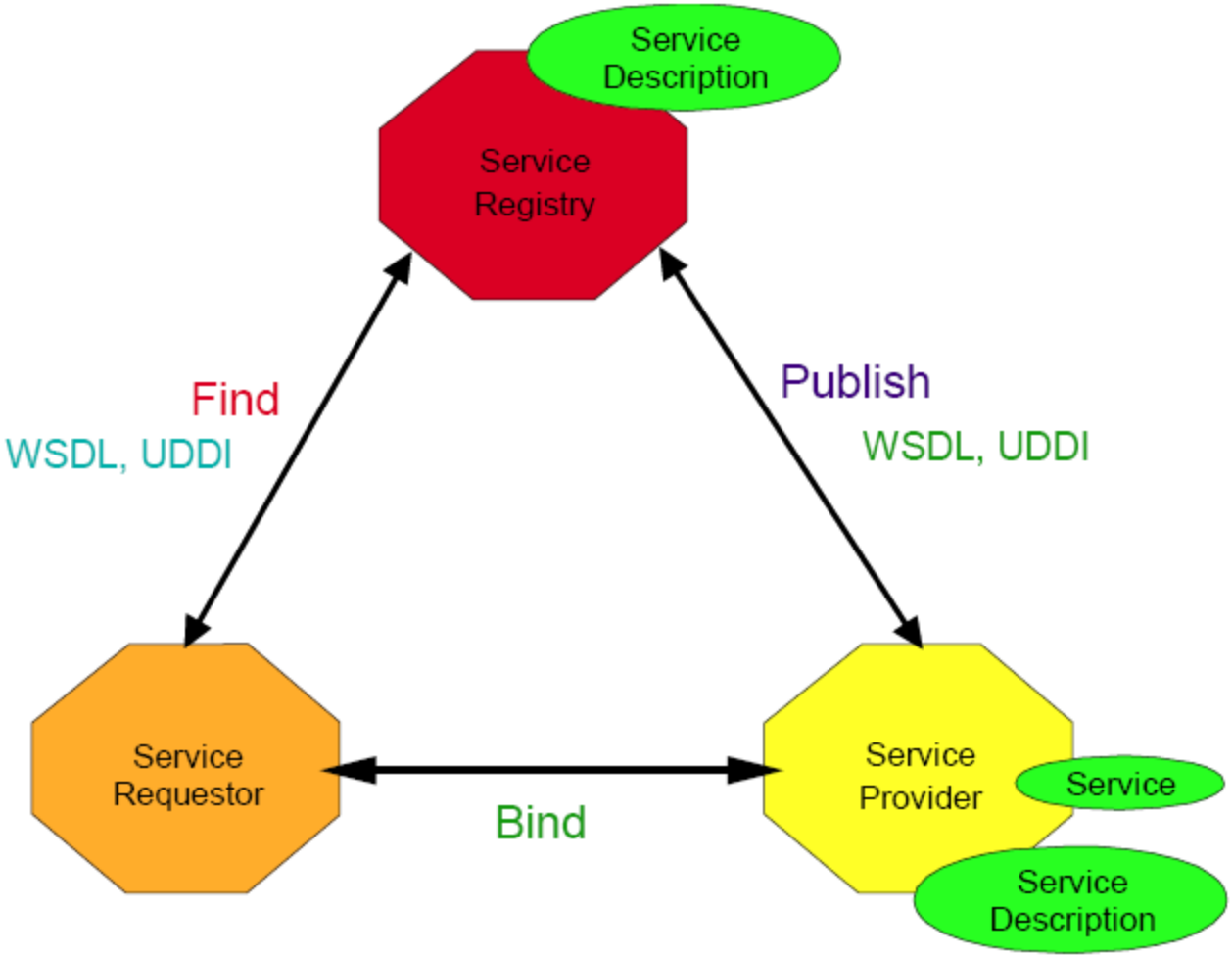
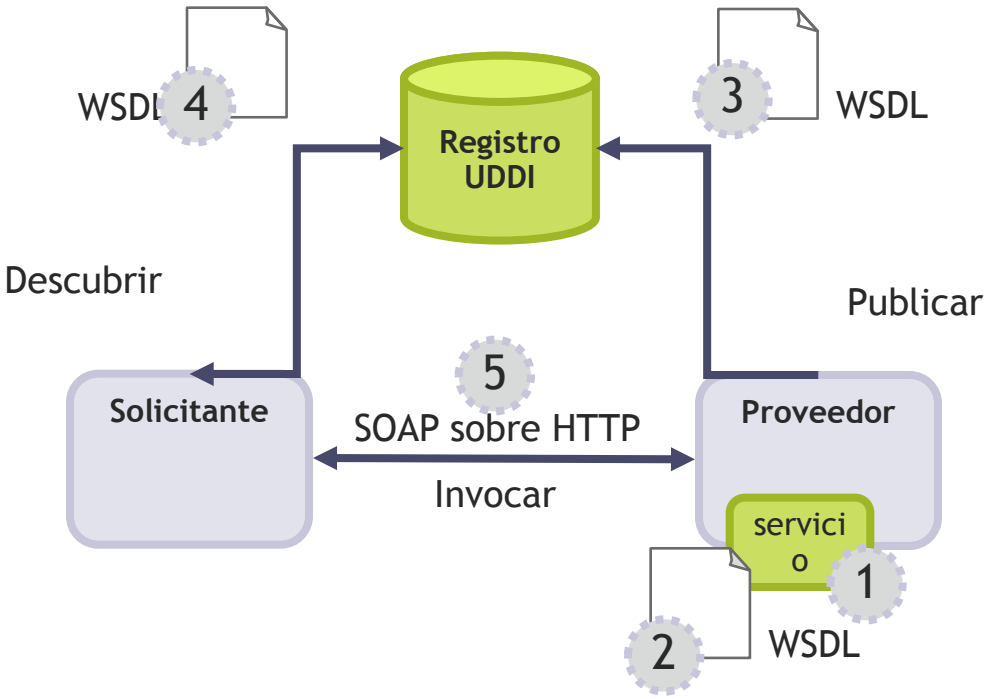


Imagen tomada de: [20] Heather KREGER. Web Services Conceptual Architecture. IBM Software Group. May 2001

Estándares básicos

Directorio (Publicar / Buscar Servicios)	UDDI
Descripción Formal de los Servicios	WSDL
Formato de Invocación de Servicios	SOAP
Formato de Datos Universal	XML
Comunicación Ubicua	HTTP

Modelo



1. El Proveedor crea el servicio.
2. El Proveedor describe el servicio en un archivo WSDL.
3. El Proveedor publica el servicio en el servidor de Registro UDDI.
4. El Solicitante descubre el servicio en el Registro.
5. El Solicitante invoca el servicio vía SOAP.

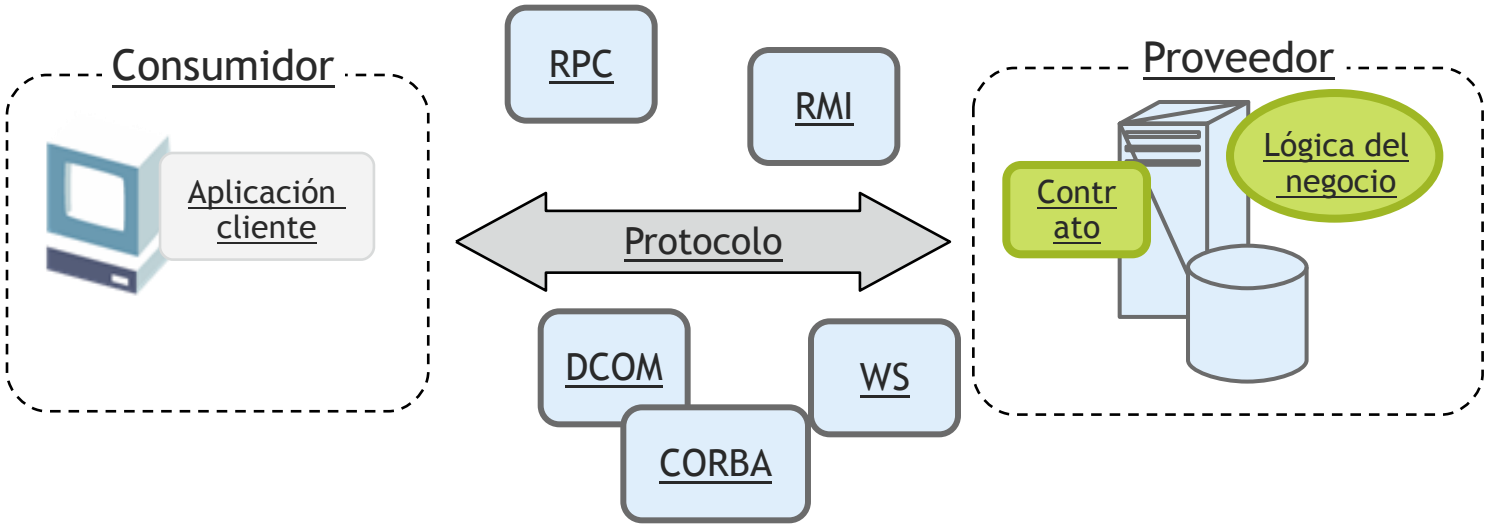
WebServices vs Otras Tecnologías Orientadas a Servicios

WebServices

- Estándares abiertos. Amigables con cualquier tecnología (XML)
- Se apoya en protocolos amigables con los mecanismos de seguridad y sistemas firewall
  - HTTP (Puerto 80)
  - SMTP (Puerto 25)
- No depende de tecnología propietaria alguna
- Modelo de desarrollo simple

RMI, CORBA, DCOM

- Se apoya en tecnologías cerradas y propietarias
- Requiere de puertos TCP no amigables (RMI-1099-TCP)
- Estándares complejos de operar (XDR, IDL)
- Modelo de desarrollo complejo (Requiere de extensos conceptos de programación)



Web Services y SOA

- Implementación basada en XML
  - WSDL y XML Schemas para definición de contratos
  - SOAP como protocolo de transporte
  - WSDL y UDDI para registro y descubrimiento
- HTTP como protocolo de comunicación
  - Infraestructura de bajo costo
- Mayor Interoperatividad
  - XML es neutral con respecto a la plataformas
  - HTTP está disponible de manera general (protocolo ubicuo)
- Flexibilidad
  - Facilita el manejo de versiones de contratos
  - Se facilita la realización de transformaciones de datos
  - Resulta sencillo realizar y adoptar cambios en los contratos
- Productividad
  - Disponibilidad de herramientas, modelo de programación sencillo

Limitantes de los WebServices

- **No ofrece ventajas significativas en el desarrollo de servicios que son de uso particular de una aplicación**
- **Estándares aun en evolución**
  - Existen problemas de interoperatividad
  - Las arquitecturas deben considerar cambios en los estándares
- **Tecnología no apropiada para integración a nivel de datos**
  - Alta latencia en el acceso a través de la red y sobrecarga en la invocación de cada servicio de manera individual.
  - No contempla manejo de lotes de servicios
  - Carece de mecanismo de manejo de transacciones del tipo “two face commit”
- **El modelo de programación orientado a servicios no transcientes**
  - No hay manejo de instancias de un servicio
  - Se requieren estándares para el manejo de estado y el ciclo de vida
- **Carece de mecanismos de garantía de servicio**
  - Definición de prioridades
  - Definición de atributos de confidencialidad a nivel de invocación de servicios
  - Control de uso de recursos (red, procesamiento de servidor) por servicio

Visión Web Services (x Capas)

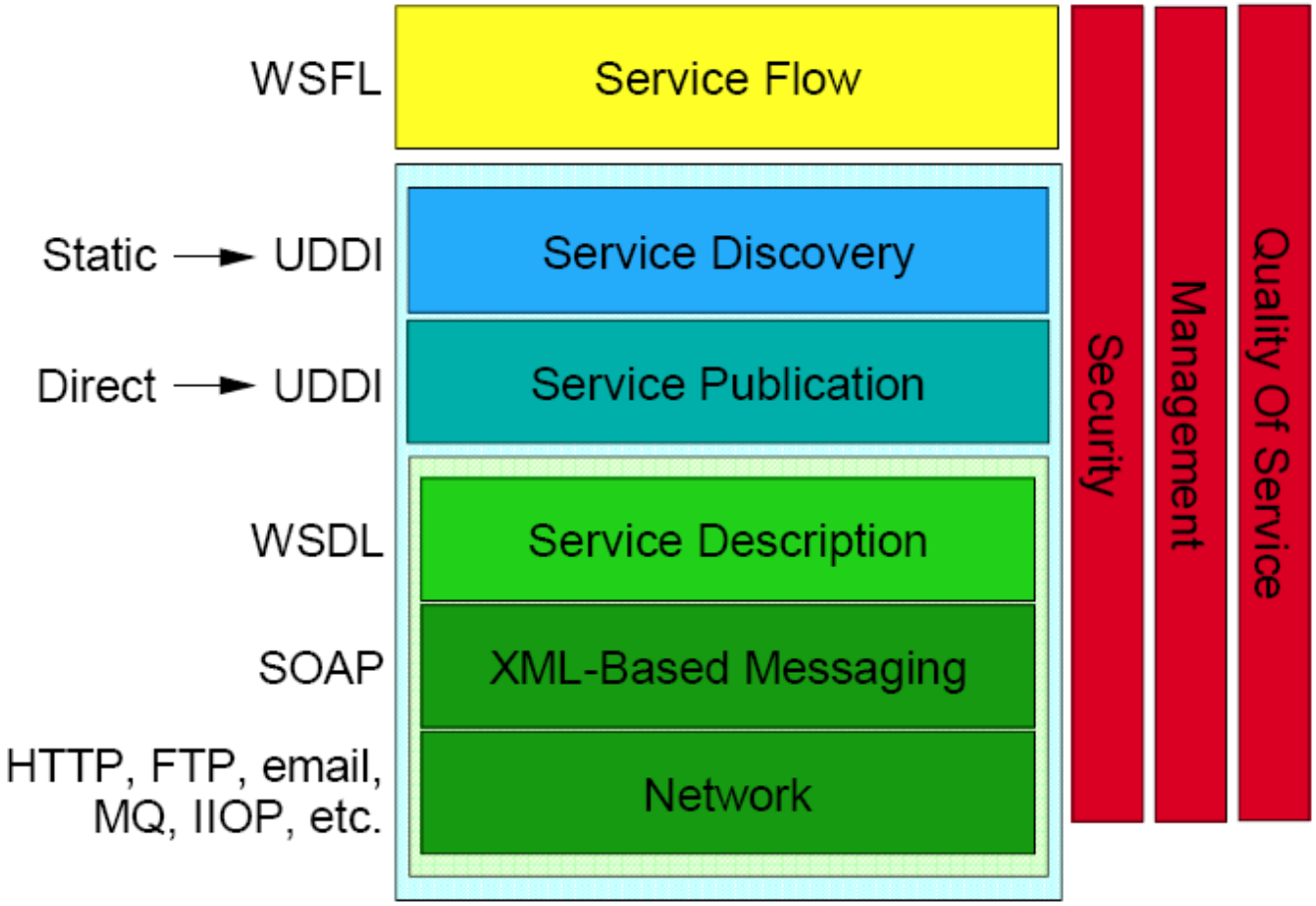


Imagen tomada de: [20] Heather KREGER. Web Services Conceptual Architecture. IBM Software Group. May 2001

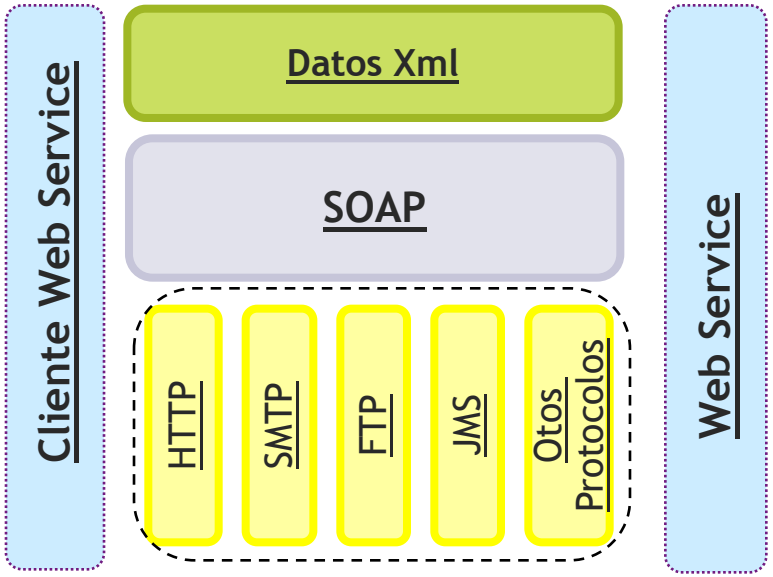
Estándares para Web Services

- **SOAP – Simple Object Access Protocol.**
  - Soporta la codificación de datos arbitrarios (XML) y su transferencia sobre HTTP
  - Permite a los solicitantes codificar parámetros y enviarlos a los proveedores así como la correspondiente respuesta
- **WSDL – Web Services Description Language.**
  - Lenguaje basado en XML que describe la funcionalidad de los Web Services
  - Describe las operaciones que un servicio puede soportar, así como los parámetros que cada operación acepta y retorna
- **UDDI – Universal Description Discovery and Integration.**
  - Directorio global de Web Services
  - Protocolo para publicarlos y descubrirlos
  - Los archivos WSDL que describen los Web Services son publicados en el directorio UDDI

SOAP: Simple Object Access Protocol

- "SOAP provee un mecanismo simple y liviano para intercambiar información estructurada entre participantes de un ambiente distribuidos y descentralizados, utilizando XML"
  W3C, SOAP 1.1 , Mayo 2000

- Ofrece como ventajas principales:
  - Simplicidad: facilita el desarrollo de proveedores y consumidores
  - Independencia
    - Plataformas y lenguajes de programación
    - Mecanismos de transporte (Aún cuando su uso sobre HTTP es el más generalizado)



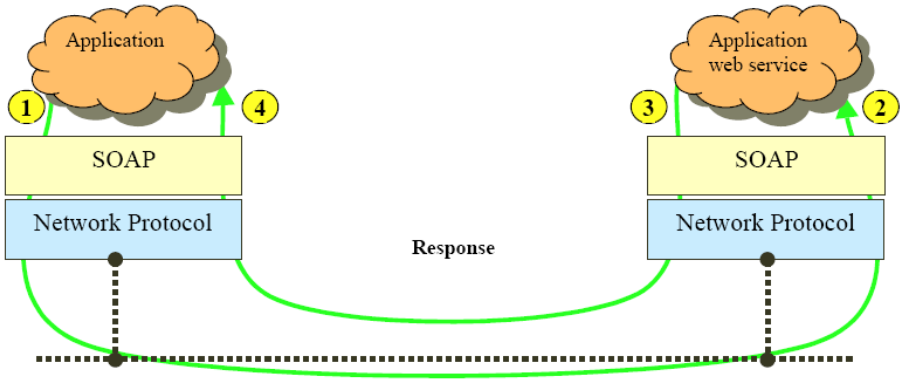
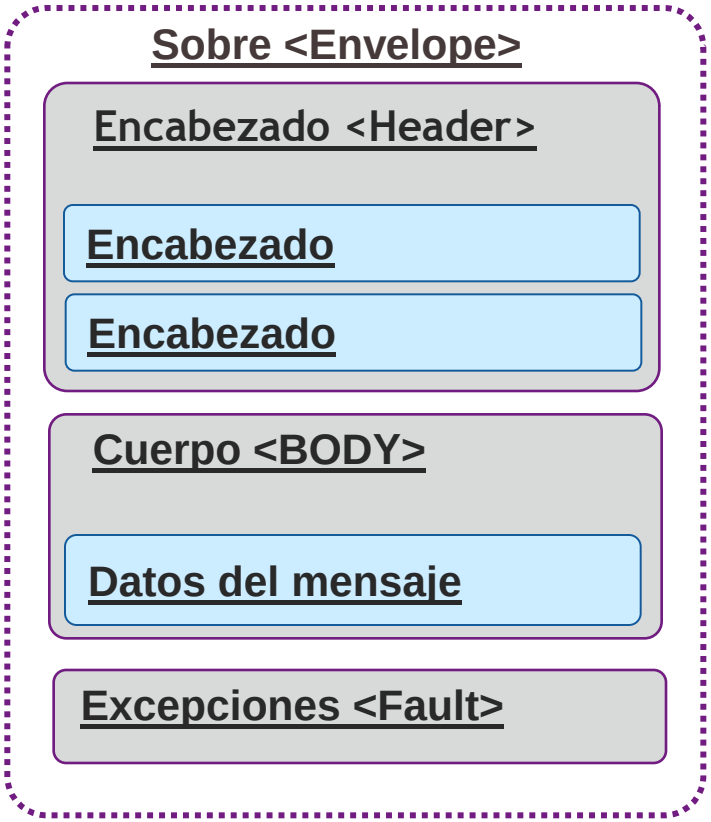


Imagen tomada de: [20] Heather KREGER. Web Services Conceptual Architecture. IBM Software Group. May 2001

Estructura de un mensaje SOAP



- **Envelope**
  - "Envuelve" al mensaje Xml que desea transmitirse
  - Debe estar presente
- **Encabezado**
  - Es opcional.
  - Su contenido no está definido en el estándar básico de SOAP
  - Puede contener información acerca de las características de procesamiento del servicio
  - Es utilizado por otros estándares como WS-Security, WS-Coordination
- **Cuerpo**
  - Debe estar presente
  - Contiene la información a ser transmitida
- **Excepciones**
  - Puede estar presente en la respuesta a un servicio
  - Contiene información acerca de excepciones en el procesamiento
  - Indica el código, mensaje, actor y descripción detallada de la excepción

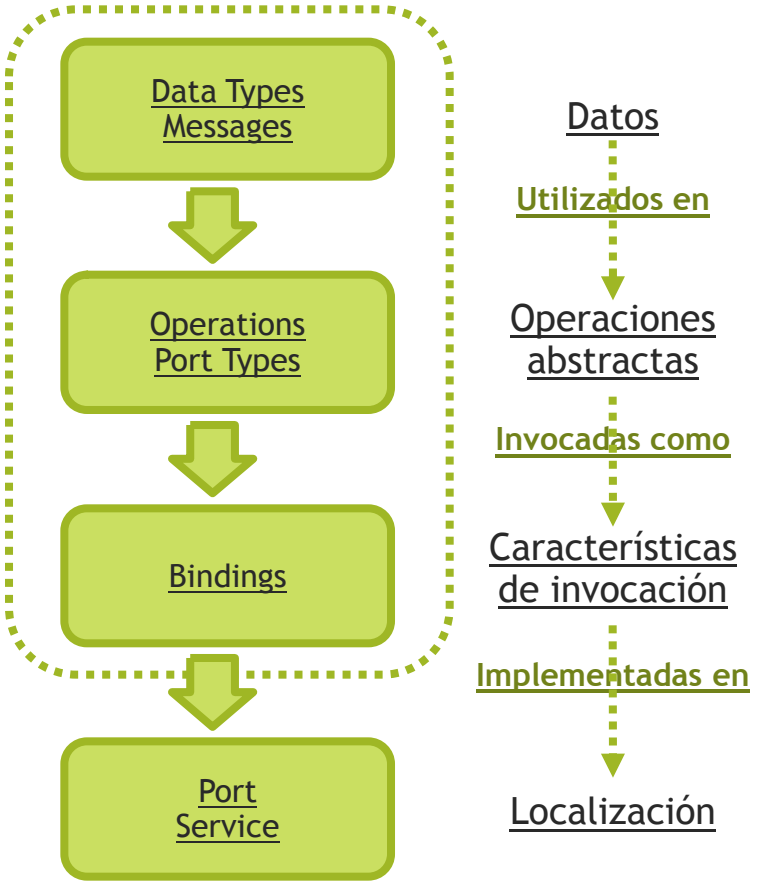
Ejemplo de solicitud SOAP sobre HTTP

```
POST /StockQuote HTTP/1.1
Host: www.stockquoteserver.com
Content-Type: text/xml;
charset="utf-8"
Content-Length: nnnn
SOAPAction: "www.stockquoteserver.com/services/getquote.htm"

<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
   <SOAP-ENV:Body>
       <m:GetLastTradePrice
   xmlns:m="www.stockquoteserver.com/services/getquote">
           <symbol>DIS</symbol>
       </m:GetLastTradePrice>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

WSDL

- **Describe información Operacional**
  - **Dónde el servicio está localizado** (definición de implementación de servicios)
  - **Qué es lo que el servicio hace** (definición de interfaz de servicio)
  - **Descripción del dato**, típicamente usando un o más esquemas XML, para que tanto el consumidor como el servidor entiendan el dato a ser intercambiado
- Similar en propósito al IDL, pero en formato XML
- Usualmente es generado de manera automática por las herramientas de desarrollo de servicios web



WSDL:Types

Service
Port
Binding
Port Types
Operations
Messages
Types

Definición de tipos de formato de sintáxis de los mensajes

```
<types>
  <schema
targetNamespace="http://example.com/stockquote.xsd"
    xmlns="http://www.w3.org/2000/10/XMLSchema">
<element name="TradePriceRequest">
                  <complexType>
               <element name="tickerSymbol"     type="string"/>
   </complexType>
</element>
<element name="TradePrice">
  <complexType>
               <element name="price" type="float"/>
   </complexType>
</element>
  </schema>
</types>
```

WSDL:Messages

- Service
- Port
- Binding
- Port Types
- Operations
- **Messages**
- Types

Datos asociados a las operaciones

```
<message name="GetLastTradePriceInput">
<part name="body" element="xsd1:TradePriceRequest"/>
</message>

<message name="GetLastTradePriceOutput">
<part name="body" element="xsd1:TradePrice"/>
</message>
```

WSDL:Operations

- Service
- Port
- Binding
- Port Types
- **Operations**
- Messages
- Types

Funciones disponibles en el servicio

```
<operation name="GetLastTradePrice">
<input message=“tns:GetLastTradePriceInput” />

<output message=“tns: GetLastTradePriceOutput” />
</operation>
```

WSDL: Port Types

- Service
- Port
- Binding
- **Port Types**
- Operations
- Messages
- Types

Conjunto de operaciones relacionadas

```
<portType name="StockQuotePortType">
  <operation name="GetLastTradePrice">
<input message="tns:GetLastTradePriceInput"/>
<output message="tns:GetLastTradePriceOutput"/>
  </operation>
</portType>
```

WSDL: Binding

- Service
- Port
- **Binding**
- Port Types
- Operations
- Messages
- Types

Definición de atributos de invocación

```
<binding name="StockQuoteSoapBinding“
type="tns:StockQuotePortType">
<soap:binding style="document“
        transport="http://schemas.xmlsoap.org/soap/http"/>
   <operation name="GetLastTradePrice">
        <soap:operation
          soapAction="http://example.com/GetLastTradePrice"/>
            <input>
                        <soap:body use="literal"/>
            </input>
            <output>
                        <soap:body use="literal"/>
            </output>
</operation>
</binding>
```

WSDL: Port

- Service
- Port
- Binding
- Port Types
- Operations
- Messages
- Types

Localización de Servicios

```
<port name="StockQuotePort" binding="tns:StockQuoteBinding">
<soap:address location="http://example.com/stockquote"/>
</port>
```

WSDL: Service

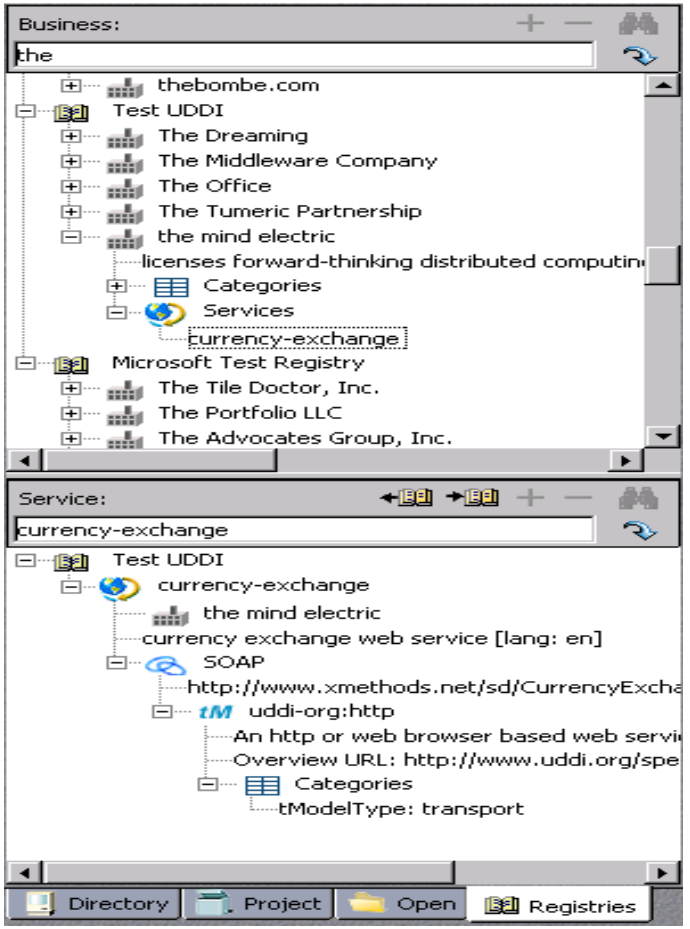
- Service
- Port
- Binding
- Port Types
- Operations
- Messages
- Types

Descripción de un servicio

```
<service name="StockQuoteService">
   <documentation>My first service</documentation>
   <port name="StockQuotePort"
binding="tns:StockQuoteBinding">
         <soap:address location="http://example.com/stockquote"/>
   </port>
</service>
```

Estándares - UDDI

- Estándar diseñado para proporcionar un directorio consultable de Web Services.
  - UDDI actúa como un DNS para Web Services.
- UDDI permite a los usuarios encontrar organizaciones, productos y servicios
- El registro (“Business Registry”) consiste de tres diferentes componentes:
  - Las "páginas blancas", permiten encontrar información acerca de los proveedores de servicios
  - Las "páginas amarillas“, permiten localizar una empresa que de servicios en alguna área concreta.
  - Las "páginas verdes“, permite eencontrar Información técnica acerca de Web Services.



Estándares de Web Services y RNF

- Seguridad
  - SSL, WS-Security, XML Encription, XML Signature
- Confiabilidad y enrutamiento de mensajes
  - WS-Addressing, WS-Reliability
- Integridad Transaccional
  - WS-transaction
- Administración y monitoreo
  - SNMP, Distributed Management
- Balanceo de carga y aprovisionamiento de servicios
  - WS-Addressing, SPML, WS-Provisioning
- Composición de procesos de negocio
  - BPELWS

Referencias

- The Ins and Outs, How EAI Differs By Jeff Pinkston http://www.eaijournal.com/PDF/Ins&OutsPinkston.pdf
- The OMG and Service Oriented Architecture http://www.omg.org/attachments/pdf/OMG-and-the-SOA.pdf
- Enterprise Integration EAI vs. SOA vs. ESB - Anurag Goel http://hosteddocs.ittoolbox.com/Enterprise%20Integration%20-%20SOA%20vs%20EAI%20vs%20ESB.pdf
- Service Oriented Architectures - Phillip J. Windley
- “Webservices: Conceptos, estándares, tecnología” – Conferencia: Principios y bases Conceptuales SOA. Capacitación Camara de Comercio. Mayo 2005, Bógota. Autor Jorge Arias. Derechos propietarios: Novell, Inc. Uso con permiso de Jorge Arias.
- Heather KREGER. Web Services Conceptual Architecture. IBM Software Group. May 2001