Integración de Aplicaciones aplicando Web Services:

Experiencias, Resumen y Perspectivas

Raul Ruggia
InCo - Facultad de Ingeniería - UDELAR

JIAP 2003

Temas

- **Motivaciones.**
- **Integrando J2EE y .NET con Web Services.**
- **Puntos fuertes de WS como integrador.**
- **Limitaciones (actuales) de WS.**
- **Futuro de los Web Services.**
- **Transacciones sobre Web Services.**

Motivación

- **Interesa integrar aplicaciones:**
  - Que se ejecutan en plataformas heterogéneas.
    - » Aplicaciones legadas y nuevas heterogéneas.
  - Con mecanismos sincrónicos y asincrónicos.
    - » Invocación a funciones y envío de mensajes.
  - A través de mecanismos lo más sistemáticos posibles (lo menos ad-hoc posible).
    - ➔ Estandarización, productividad, mantenibilidad.
  - Integrar aplicaciones a través de Internet.
    - ➔ Aplicaciones cooperativas en Internet.

Web Services en Integración de Aplicaciones

¿ Por qué ?

- Cumple mayoritariamente los puntos anteriores.
  - » Basado en estándares multiempresa.
  - » Implementado en plataformas dominantes (J2EE, .NET).
  - » Encapsula servicios con especificaciones declarativas (WSDL).
  - » Protocolo (SOAP) muy asociado a HTTP ➔ Internet.

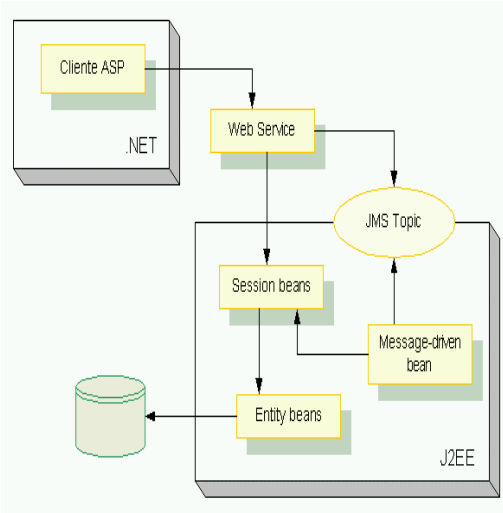
- **Otras opciones:**
  - Adapters (p.ej. JCA).
    - » Mecanismo de J2EE para generar gateways.
    - » Implementado mayoritariamente para acceso a ERPs.
  - Integration Brokers.
    - » Sistemas de porte medio y grande para integrar aplicaciones.
    - » Proveen fundamentalmente mecanismos asíncronos.

Integrando J2EE y .NET

- **Casos estudiados.**
  - Cliente J2EE → Servidor .NET.
  - Cliente .NET → Servidor J2EE.
    (Plataforma J2EE: Apache, Tomcat, JBoss)
- **Operaciones implementadas:**
  - Sincrónica de lectura.
  - Sincrónica de modificación de datos.
  - Asíncrona de lectura y modificación de datos.
- **Basado en:**
  - *Interoperabilidad entre Servidores de Aplicaciones*. J. Besil, C. Pais, D. Sande.
    - » Proyecto de Grado, InCo-Fing-UDELAR. 2003.
    - » Servicio Web para ensayos de interoperabilidad.

Cliente .NET, Servidor J2EE (1)

- **Lado Cliente.**
  - ASP invoca WS en J2EE.
    - » Se generan clases que actúan como proxy.
    - » Mapping de tipos transparente.
- **Lado Servidor.**
  - Exportó como Servicio Web una clase con métodos públicos.

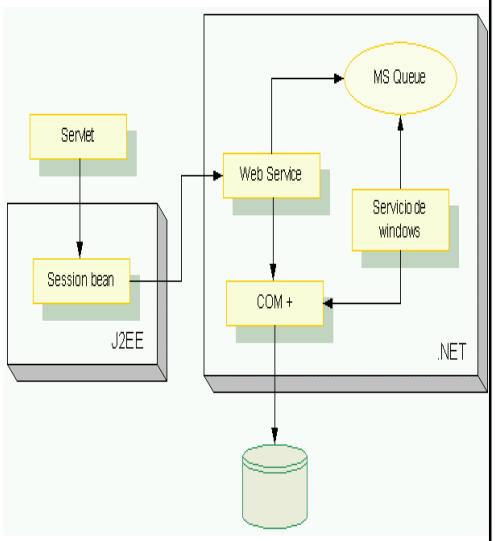


Cliente .NET, Servidor J2EE (2)

- **Invocaciones asíncronas.**
  - .NET maneja mecanismo en el cliente.
    - » Nuevos hilos de ejecución para cada invocación asincrónica realizada, y en este nuevo hilo se realiza una invocación sincrónica típica.
    - » Por ejemplo,
      - Para el método **`EnviarMensaje`** expuesto por el servicio Web, el proxy expondría los métodos **`EnviarMensaje`** (sincrónico) y **`BeginEnviarMensaje`**, **`EndEnviarMensaje`** (asincrónico).
  - Asincronismo en el cliente:
    - » **Ventaja:** independizade la plataforma Servidor.
    - » **Desventaja:** carga al Cliente con procesos.

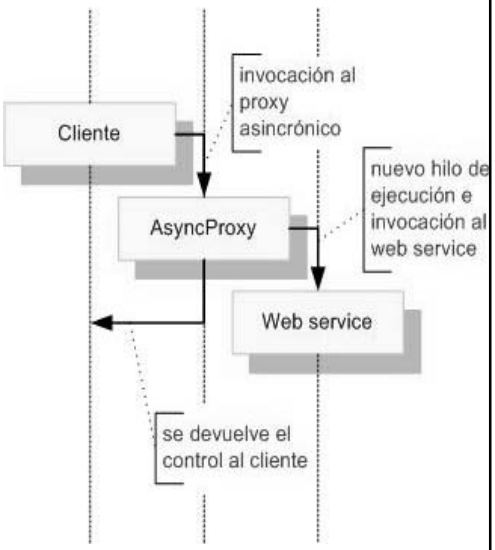
Cliente J2EE, Servidor .NET (1)

- **Lado Cliente, Session Bean invoca WS:**
  - **Sincrónica**, invoca métodos en el proxy normal.
  - **Asincrónica**, a través de los métodos de las clases en el paquete generado por *wsdlcompiler* (Sun).
- **Mapping de tipos:**
  - Transparente para básicos.
  - Problemas con importación de DataSet de .NET. Se implementó con array de objetos.



Cliente J2EE, Servidor .NET (2)

- **Asíncrono.**
  - J2EE no provee nativamente el modelo de .NET.
    - » Se uso WsdlCompiler (Sun).
      - Implementa asincronismo en el lado del cliente, creando un nuevo thread que invoca al Web Service.
    - » Por cada método del WS, WsdlCompiler crea dos métodos en el proxy.
    - » El cliente debe proveer una implementación a la interfaz IWebServiceCallback.
  - Limitación importante de la versión actual WsdlCompiler:
    - » No permite retorno de valores de tipos complejos.



Conclusiones : puntos fuertes

- **Funciona directamente en invoc. sincrónicas.**
  - Modelos similares entre J2EE y .NET.
- **Mappings de tipos (mayormente) transparentes.**
- **Herramientas de desarrollo facilitan MUCHO:**
  - Generan clases de lado cliente a partir de WSDL.
- **Mecanismo abierto:**
  - No depende de herramientas o plataformas específicas.
- **Soportado por serv. de aplicaciones J2EE y .NET.**
  - Buena conexión hacia los Sistemas de Información implementados en estos servidores.

Conclusiones : limitaciones

- **No soporta (actualmente en el estándar):**
  - **Transacciones.**
    - » Propiedades ACID en invocaciones de Web Services.
  - **Robustez suficiente en comunicación asíncrona (mensajes).**
    - » P.ej., performance, simplicity, robustness and composability .
  - **Mecanismos para Seguridad** (estandarizados).
- **Mecanismo "débilmente acoplado".**
  - Basado en XML como lenguaje de especificación.
  - Correctitud de invocaciones controlada en desarrollo.
- **Modelos de integración no totalmente estandarizados.**
  - Modelo asíncrono es diferente en J2EE y .NET.
- **No implementados por sistemas importantes:**
  - CORBA, Monitores Transaccionales (Tuxedo, CICS).

Futuro de los Web Services

- **Visión general.**
  - Basado en "The Future of Web Services". Presentación de Felipe Cabrera, Microsoft Research (co-autor de WS-S, WS-T). Gentileza de Wilson Pais.

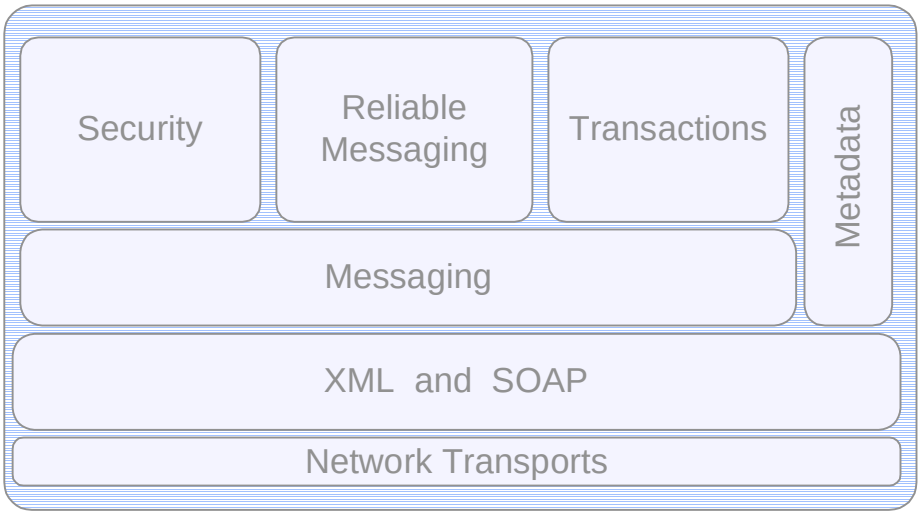
Interoperability

http://www.WS-i.org

- **An open industry effort**
  - Industry initiative focused on promoting Web Services interoperability formed by leaders
  - Open participation and membership (160 +)
- **Goal: Enable interoperability across platforms, applications, and programming languages**
- **Based on partnerships**
  - Symbiotic relationship with other standards organizations through integration of their outputs
  - Success will accelerate adoption and deployment of Web Services
- **Evidence of industry alignment around Web Services**

13
13

Foundation Protocols
"Secure, reliable, transacted messages"



14
14

WS-* Specifications Roadmap

- WS-Routing
- WS-Referral
- WS-Inspection
- WS-Security
  - » Addendum
  - » Profile for Tokens
- WS-Attachments
- WS-Coordination
- WS-Transaction
- WS-Trust
- WS-SecureConversation
- WS-SecurityPolicy
- WS-Policy
- WS-PolicyAttachment
- WS-PolicyAssertions
- WS-Addressing
- WS-ReliableMessaging

Transacciones en WS

- **Importancia:**
  - Fundamental para consistencia de operaciones distribuidas que modifiquen datos.
  - Carencia limita drasticamente el uso de los Web Services como mecanismo de interoperabilidad.
- **Situación actual:**
  - Actual especificación de Web Services no incluye la noción de transacción.
  - Trabajo actual en extensiones:
    - » WS-Transactions (BEA, IBM, y Microsoft).
- **En WS-T se diferencian los conceptos de:**
  - Transacciones atómicas, tradicionales ACID.
  - Transacciones del negocio, rollback por compensación.