



# Servicios Web

## Sesión 3: Orquestación de servicios: BPEL



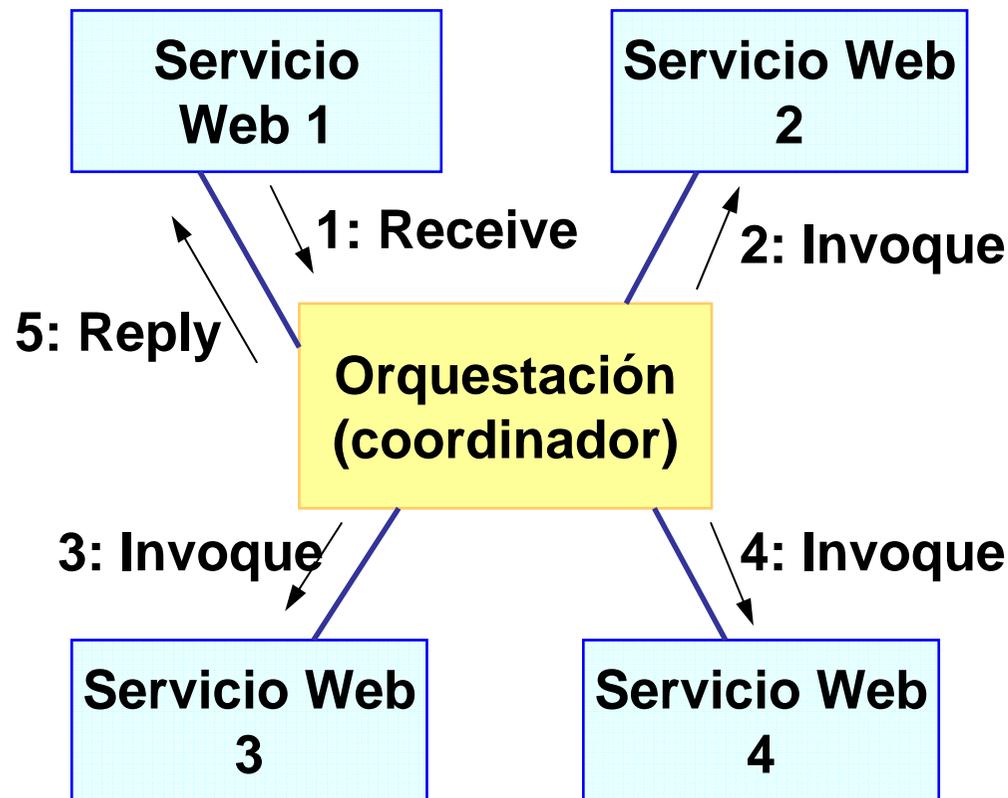
## Puntos a tratar

- Orquestación vs. Coreografía
- El lenguaje BPEL
- Estructura de un proceso BPEL
- Pasos para desarrollar un proceso BPEL
- Despliegue y pruebas de un proceso BPEL
- Creación y ejecución de casos de prueba

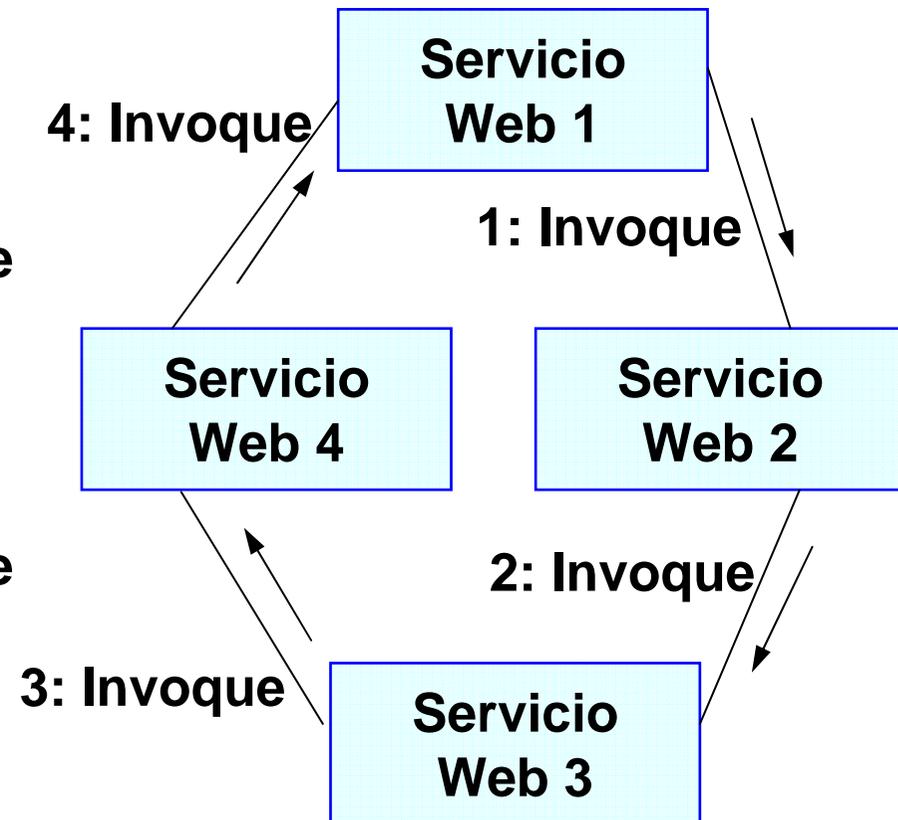


# Orquestación vs. Coreografía

- **ORQUESTACIÓN**



- **COREOGRAFÍA**





## ¿Por qué orquestar servicios Web?

- Los servicios Web como tecnología común para proporcionar puntos de integración entre las aplicaciones
  - Modelo de interfaces que permite integrar las aplicaciones independientemente de su origen
  - Descubrimiento de los servicios en tiempo de ejecución
  - Bajo acoplamiento
- Orquestación de servicios Web como aproximación abierta, basada en estándares para crear procesos de negocio de alto nivel



# ¿Por qué orquestar Servicios Web con BPEL?

- BPEL es un lenguaje estándar para la integración y automatización de procesos
- Menores costes de mantenimiento
- Menores costes de soporte
- Amplía el grupo de desarrolladores
- BPEL proporciona soporte para:
  - Elevados tiempos de ejecución
  - Compensación
  - Reacción ante eventos
  - Modelado de actividades concurrentes
  - Modelos con estado

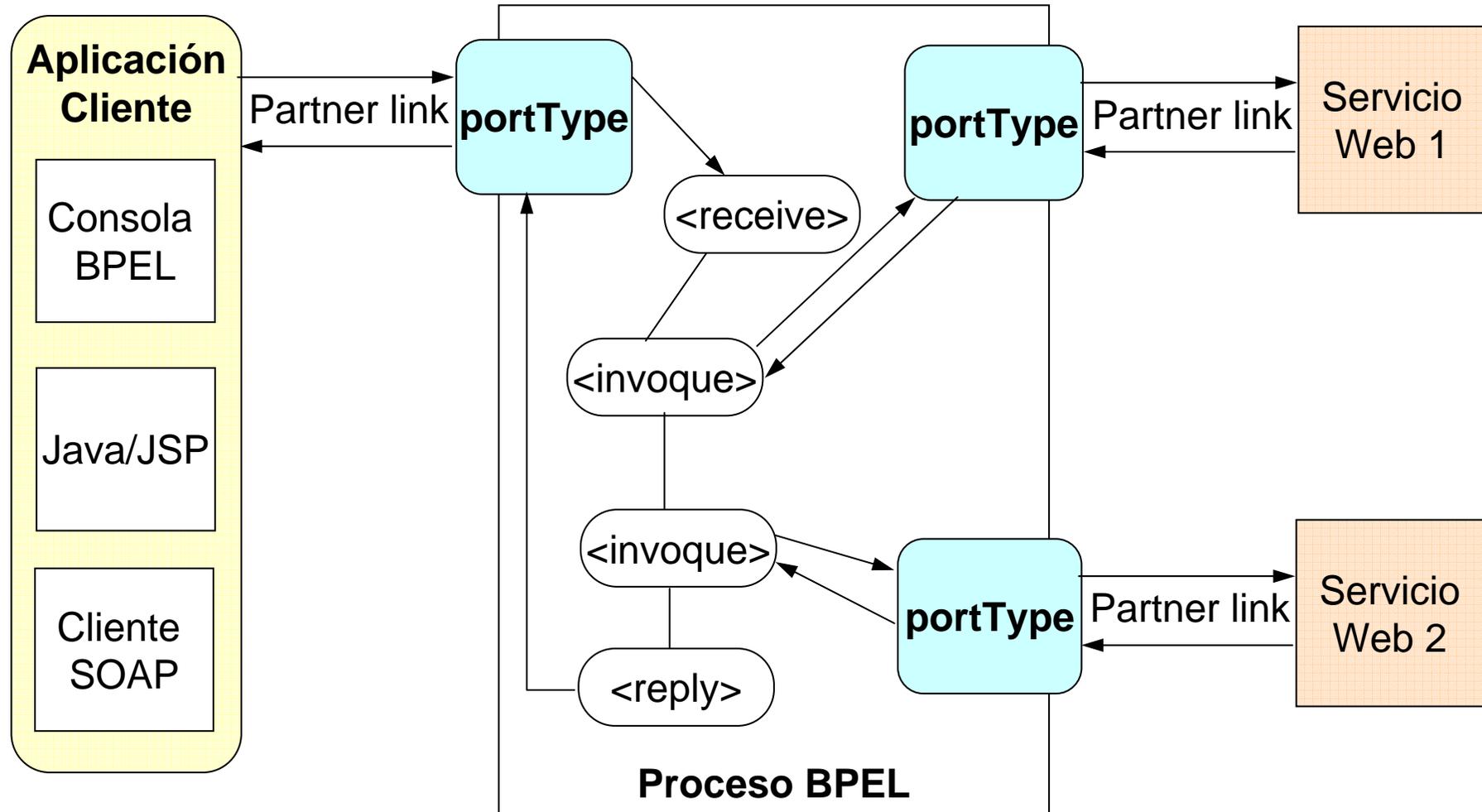


# El lenguaje BPEL

- BPEL: **B**usiness **P**rocess **E**xecution **L**anguage
- Lenguaje basado en XML que soporta la tecnología de servicios Web
- Diseñado para definir procesos de negocio
- BPEL es la convergencia entre WSFL (IBM) y XLANG (Microsoft) → BPEL4WS
- En 2003, BPEL4WS → WS-BPEL (estándar)
- BPEL puede utilizarse dentro de una empresa y entre empresas

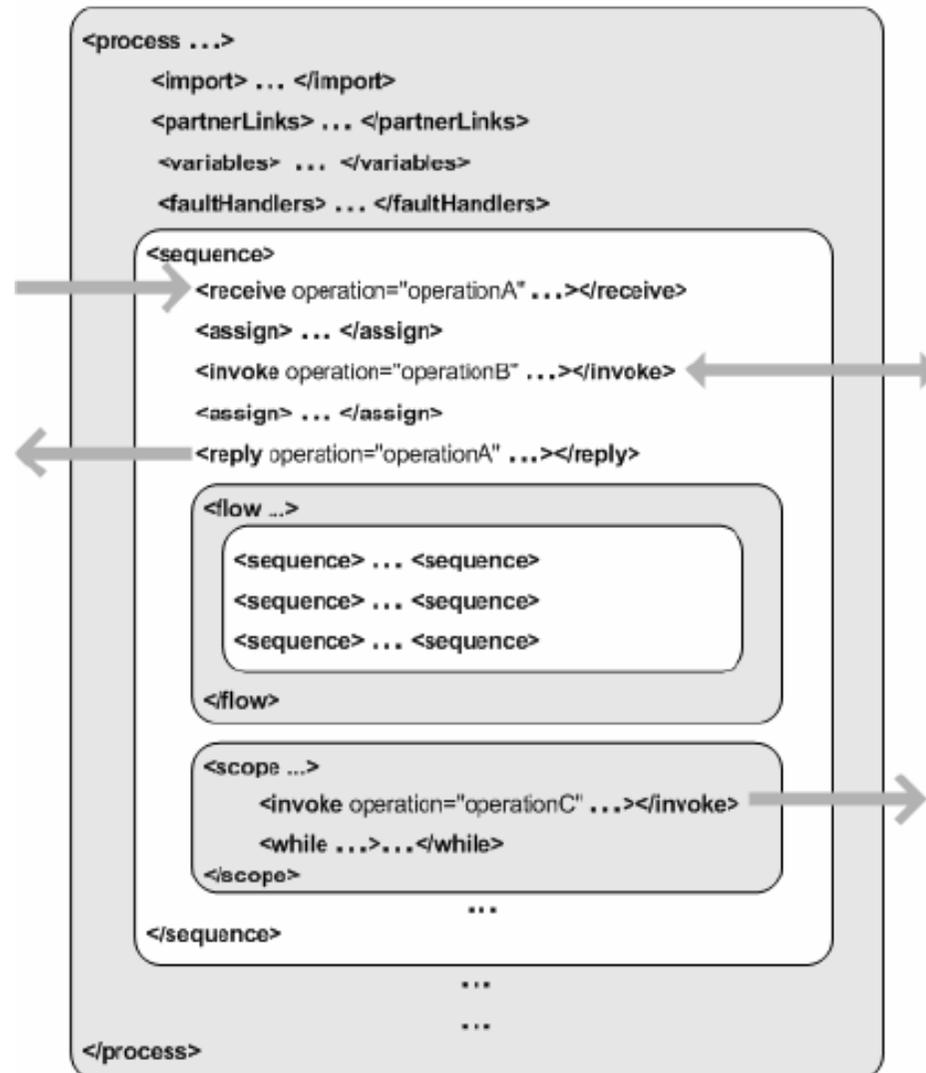


# Esquema de un proceso BPEL





# Estructura de un proceso BPEL





## Partner Links

- **Partner Links**: definen los enlaces con las diferentes "partes" que interactúan con el proceso BPEL
  - Se define en el proceso BPEL
  - ```
<partnerLink name=... partnerLinkType=... myrole=... partnerRole=.../>
```
- Los *roles* se definen en el WSDL de cada *partner*
- Las operaciones síncronas utilizan un único rol



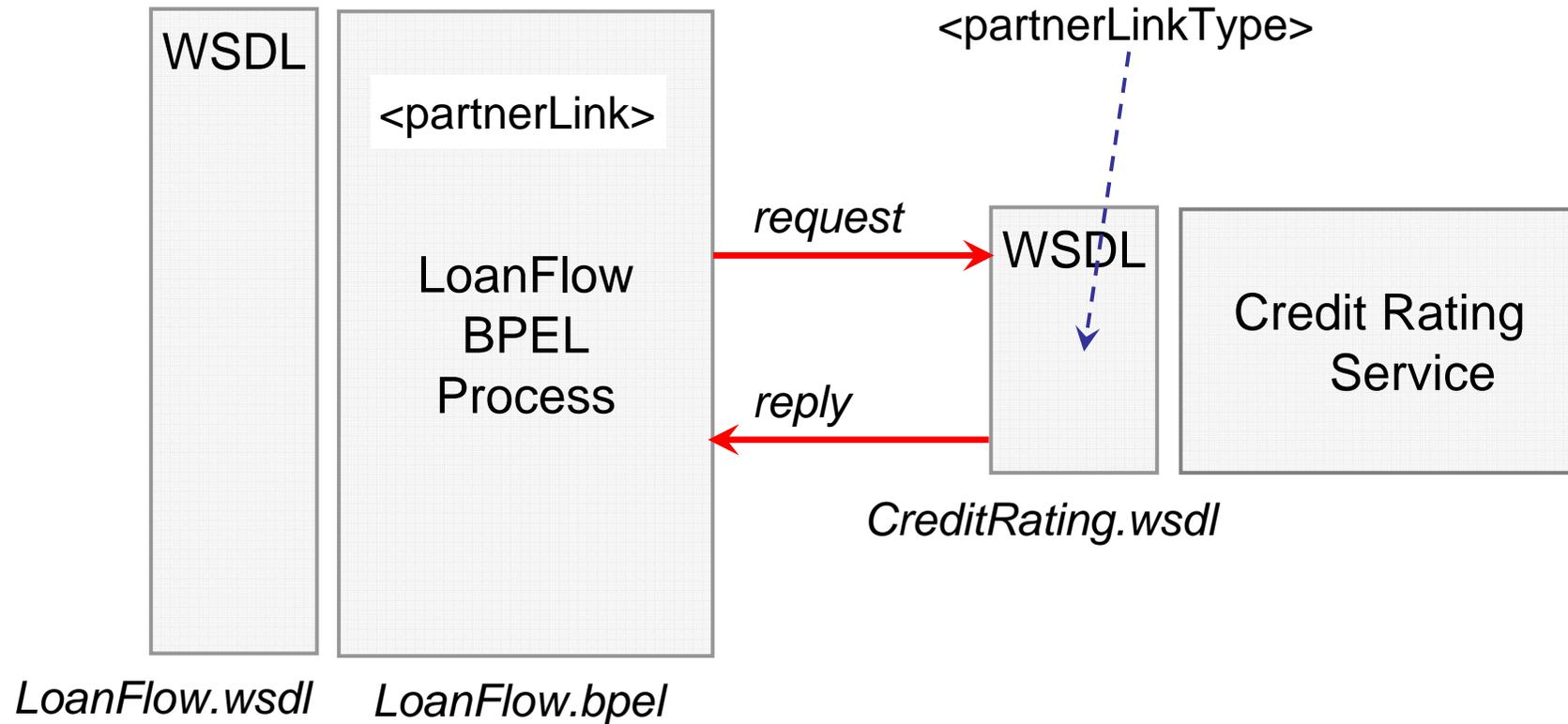
## *PartnerLinkType*

- *PartnerLinkType*: especifica la relación entre dos servicios, definiendo el rol que cada servicio implementa
  - Se define en el WSDL

```
<partnerLinkType name=... >  
  <role name=... portType=.../>  
</partnerLinkType
```



# PartnerLink Types y PartnerLinks. Ejemplo





# PartnerLink Types y PartnerLinks.Ejemplo

```
<plnk:partnerLinkType name="CreditRatingService">  
  <plnk:role name="CreditRatingServiceService">  
    <plnk:portType name="tns:CreditRatingService" />  
  </plnk:role>  
</plnk:partnerLinkType>
```

*Extracto de CreditRating.wsdl*

```
<partnerLinks>  
  <partnerLink name="creditRatingService"  
    partnerLinkType="crs:CreditRatingService"  
    partnerRole="CreditRatingServiceService"/>  
</partnerLinks>
```

*Extracto de LoanFlow.bpel*



# Variables

- Declaración

```
<variables>
  <variable name="nombreVar" messageType="qname"
            type="qname" element="qname" />
</variables>
```

- Asignación

```
<assign>
  <copy>
    <from variable="ncname" part = "ncname"/>
    <to variable="ncname" part = "ncname"/>
  </copy>
</ assign >
```



# Actividades (I)

- **Primitivas**: representan construcciones básicas
  - **<receive>**: bloquea al proceso que la invoca
  - **<reply>**: devuelve una respuesta
  - **<invoque>**: invoca a un servicio Web
  - **<assign>**: asigna un valor a una variable
  - **<wait>**: suspende al proceso un cierto tiempo
  - **<throw>**: para indicar fallos y excepciones



## Actividades (II)

- **Estructuradas**: permiten combinar las actividades primitivas
  - **<sequence>**: las actividades se invocan en forma de secuencia ordenada
  - **<flow>**: las actividades se ejecutarán en paralelo
  - **<switch>**: ramas de ejecución alternativas
  - **<while>**: definición de bucles
  - **<pick>**: selección de un camino de entre varios alternativos



# Pasos para crear un proceso de negocio con BPEL

- Conocer los servicios Web implicados
  - Familiarizarnos con los *port types* de los *Web partners*
- Definir el WSDL del proceso BPEL
  - Definir el *partner link type* del cliente
- Desarrollar el proceso BPEL
  - Definir los *partner links*
  - Declarar las variables
  - Escribir la definición de la lógica del proceso



# Pasos para crear un proceso de negocio con BPEL y *Netbeans*

- Crear un proyecto BPEL
  - Proceso BPEL (\*.bpel)
  - Fichero WSDL (\*.wsdl)
  - Fichero de esquema (opcional) (\*.xsd)
- Crear un proyecto *Composite Application*
- Añadir el módulo BPEL como un módulo JBI en la *Composite Application*
- Arrancar el servidor de aplicaciones
- Desplegar el proyecto *Composite Application* en la *BPEL service engine*
- Crear y ejecutar los casos de prueba



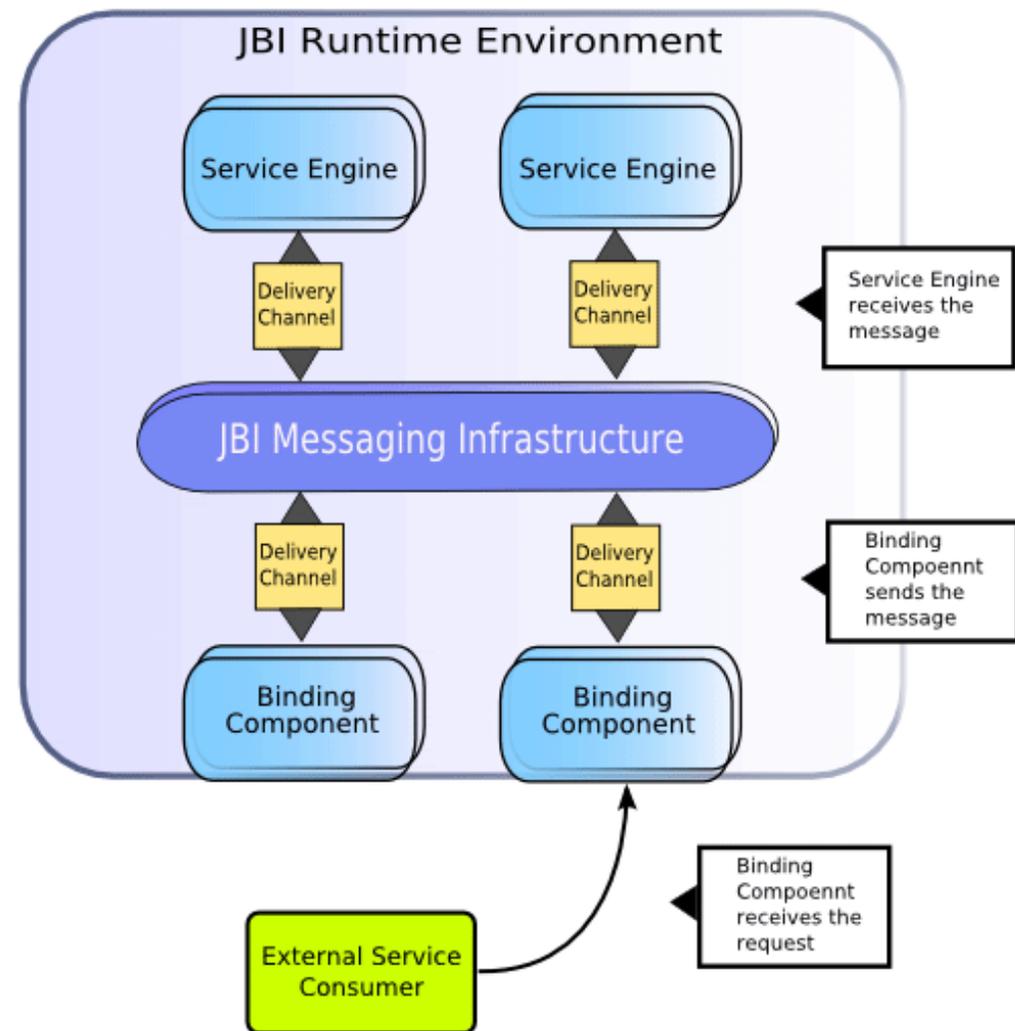
# Composite Application Project

- Se utiliza para crear un ensamblado de servicios (***Service Assembly***) que puede desplegarse en el servidor de aplicaciones como un componente JBI.
- Un proyecto BPEL no es directamente desplegable. Primero debemos añadir dicho proyecto BPEL, como un módulo JBI, en un proyecto ***Composite Application***.
- A continuación podremos desplegar el proyecto *Composite Application* en la máquina de servicios BPEL.



# Entorno de ejecución JBI

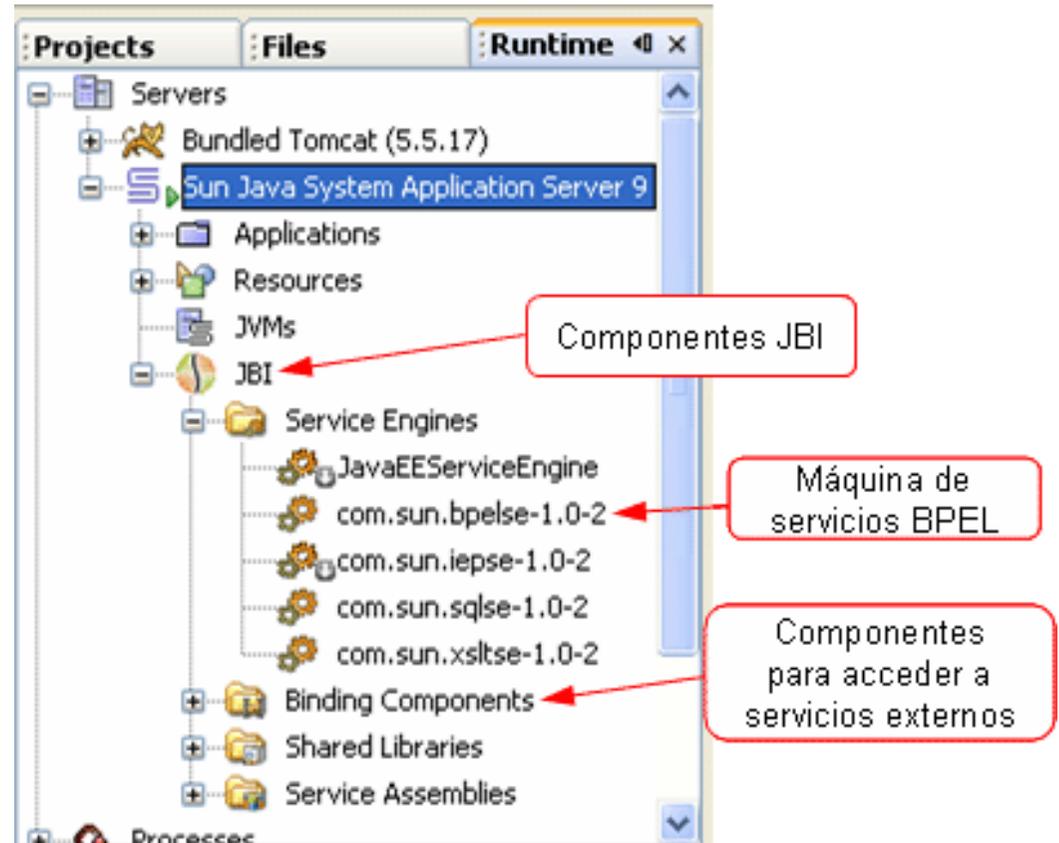
- Los componentes que suministran o consumen servicios dentro del entorno JBI son referenciados como máquinas de servicios (**Service Engines**)





# Máquina de servicios BPEL

- Es un componente JBI que proporciona servicios para ejecutar procesos de negocio desarrollados con WS-BPEL 2.0.
- La máquina de servicios BPEL arranca juntamente con el servidor de aplicaciones





# Creación y ejecución de Pruebas

- Añadimos un caso de prueba y lo enlazamos con una operación BPEL
- Determinamos las propiedades de la prueba
- Modificamos las entradas de las pruebas: fichero ***Input.xml***
- Ejecutamos la prueba: ésta queda registrada en el fichero ***Output.xml***



# ¿Preguntas...?