

[Inicio](#)[Quiénes somos](#)[Formación](#)[Publicaciones](#)[Tutoriales.](#)[adictosaltrabajo](#) / [Tutoriales](#) / [Tutorial de BPEL con OpenESB \(I\)](#)

Iván García Puebla

Consultor tecnológico de desarrollo de proyectos informáticos.



## Tutorial de BPEL con OpenESB (I)

### Uso de cookies

Este sitio web utiliza cookies para que usted tenga la mejor experiencia de usuario. Si continúa navegando está dando su consentimiento para la aceptación de las mencionadas cookies y la aceptación de nuestra política de cookies, pinche el enlace para mayor información

[ACEPTAR](#)

# Tutorial de BPEL con OpenESB (I)

## Indice

### 1. Tutorial de BPEL con OpenESB (I)

1. Introducción
2. El proceso de negocio
3. Identificar los servicios web requeridos
4. Modelar el proceso de negocio con BPEL
  1. Generar el descriptor del proceso BPEL
  2. Importar descriptores WSDL de los servicios involucrados
  3. Diseñar el proceso BPEL gráficamente

### 2. Tutorial de BPEL con OpenESB (II)

1. Desplegar el módulo BPEL en el ESB
2. Probar el proceso de compra
3. Conclusión

## Introducción

En este tutorial vamos a aprender a crear procesos [BPEL](#) practicando con un ejemplo: un proceso de negocio de venta online de libros. La empresa *TuLibroOnline* decidió adoptar SOA para sus infraestructuras IT y actualmente se encuentra en la fase de composición de servicios. Así pues, le llega el turno al proceso de venta de libros y se decide implementarlo con un proceso BPEL orquestando servicios web de las áreas implicadas en el negocio.

Para desarrollar el ejemplo es necesario tener instalado [OpenESB 2.1](#) tal como se muestra en el tutorial [OpenESB 2.1. Instalación e introducción al entorno](#), y tener unos conocimientos básicos de NetBeans y GlassFish. El código fuente puede descargarse aquí: [BPEL-openESB\\_adictosaltrabajo.zip](#) y el proyecto de pruebas de soapUI aquí: [BPEL-openESB\\_adictosaltrabajo.zip](#).

Hemos dividido el tutorial en dos partes debido a su envergadura. La segunda parte del tutorial se encuentra en



### Los más visitados de la semana

[Primeros pasos con Gulp](#)

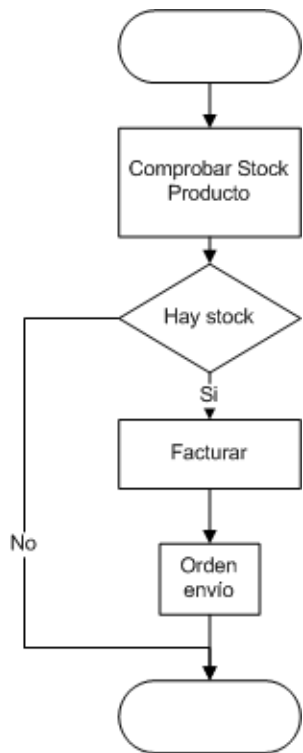
[Validación en lado cliente con HTML5](#)

[Un entorno de programación en la nube: un vistazo a Cloud 9 ...](#)

## Tutorial de BPEL con OpenESB (II).

### El proceso de negocio

Antes de programar nada, tenemos que comprender muy bien el proceso de negocio que pretendemos implementar con alguna tecnología. Para ello nos reunimos nuevamente con un responsable de negocio del área de ventas y tras comentarle los avances de la implantación tecnológica y lo que le permite ya hacer a la empresa, le pedimos valide el proceso de venta de libros. Finalmente se alcanza la propuesta:



Proceso de negocio de venta de libros

Los detalles del proceso son los siguientes:

- La orden de venta de inicia aportando el ID de cliente registrado que realiza la compra, el ISBN del libro que adquiere, el número de unidades y el precio por unidad.

Ejemplo de uso con JSF 2.0,  
Primefaces e Hibernate  
Tutorial básico de bases de datos  
en Java mediante JDBC

### Tweets por @adictosaltrabaj



**adictosaltrabajo**  
@adictosaltrabaj

TUTORIAL | Un entorno de programación en la nube. Un vistazo a #Cloud9 IDE, por @Paniadri  
[adictosaltrabajo.com/tutoriales/un-...](http://adictosaltrabajo.com/tutoriales/un-...)



Insertar

Ver en Twitter

### Archivos

Archivos

Elegir mes

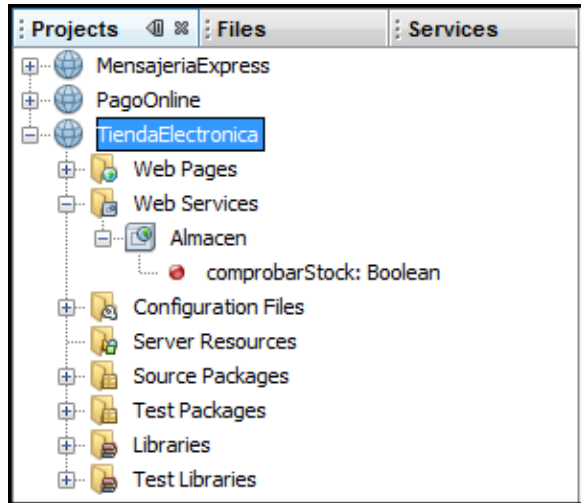
- Se comprueba en almacén que haya stock suficiente para el pedido. En su defecto el proceso finaliza y lo notifica, para así ofrecer alternativas al cliente y no perder la oportunidad de venta.
- Si hay unidades suficientes, hemos de invocar al sistema de pago online que tenemos contratado con nuestra entidad financiera. La entidad ya dispone de los datos de los clientes registrados, por lo que se encarga de hacer las tramitaciones necesarias y asegurar el pago.
- Finalmente emitimos la orden de envío a una empresa de mensajería. Facilitaremos un identificador de pedido para que lo recoja en almacén y lo reparta.
- El resultado del proceso será una confirmación de la venta realizada.

## Identificar los servicios web requeridos

Como siguiente paso tenemos que identificar los servicios web que realizan las funcionalidades concretas de cada paso del proceso. En nuestro caso son:

- Servicio web del almacén:
  - comprobarStock, que espera el ISBN del libro y las unidades de compra, y responde si existe o no stock suficiente
- Servicio web del sistema de pago online:
  - facturar, con un idCliente y una cuantía. Devuelve un código de factura
- Servicio web de la empresa de mensajería:
  - enviarProducto, con nuestro identificador de la empresa y el pedido a servir. Nos responde con el tiempo en días estimado de reparto.

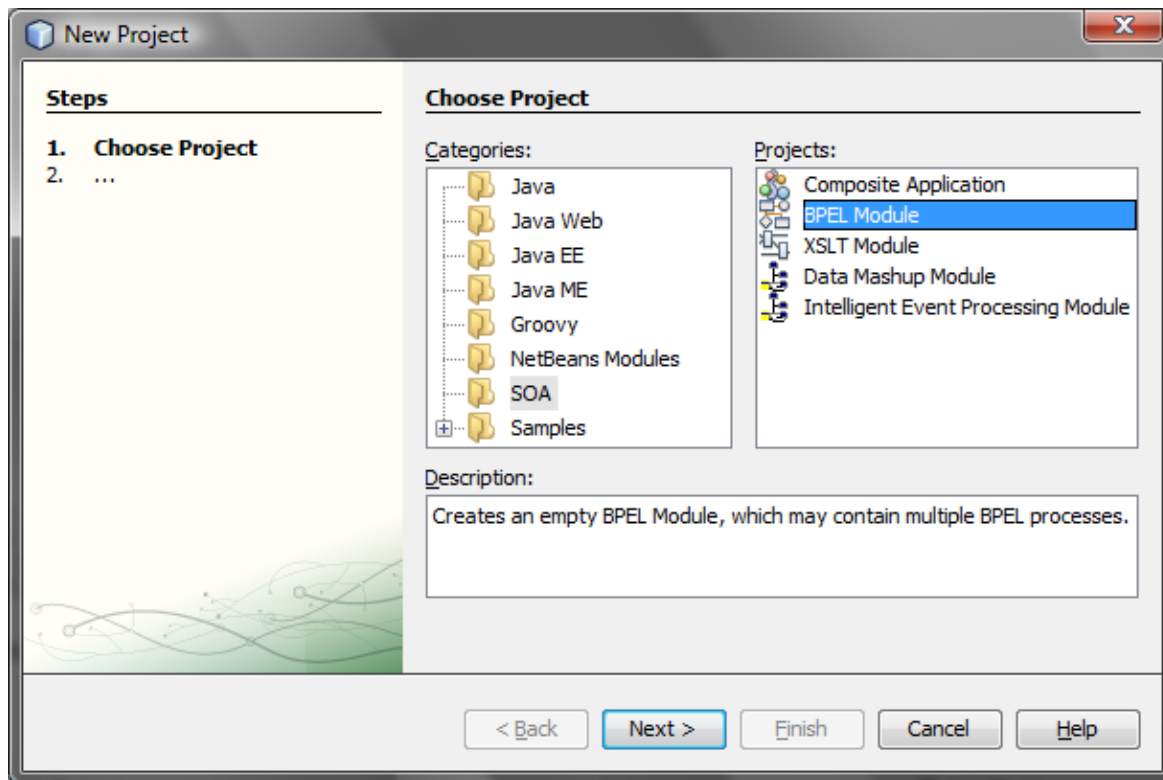
Estos servicios están implementados en los proyectos MensajeríaExpress, PagoOnline y TiendaElectronica del código fuente del tutorial. Deben ser abiertos como proyectos de NetBeans (menú File | Open Project):



Servicios Web del proceso de negocio en NetBeans

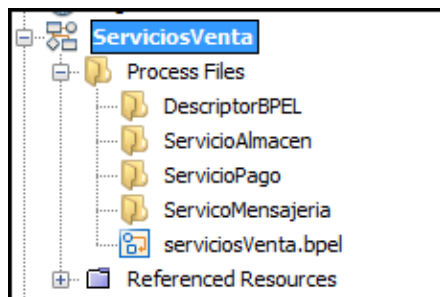
## Modelar el proceso de negocio con BPEL

Creamos un nuevo proyecto BPEL en NetBeans: File | New Project | SOA | BPEL Module:



### Crear un Módulo BPEL en OpenESB

Le asignamos el nombre ServiciosVenta. Sobre la estructura inicial del proyecto creamos las carpetas: DescriptorBPEL, ServicioAlmacen, ServicioPago, ServicioMensajeria (botón derecho sobre Process Files | New | Other | Other | Folder e introducimos el nombre en el campo Folder Name):



## Estructura inicial del módulo BPEL

### Generar el descriptor del proceso BPEL

Un proceso BPEL se expone al resto como si de un servicio web se tratara. Por ello debe tener su propio descriptor WSDL. Lo creamos (botón derecho sobre DescriptorBPEL | New | WSDL Document) e introducimos los valores de la imagen:

**New WSDL Document**

**Steps**

1. Choose File Type
- 2. Name and Location**
3. Abstract Configuration
4. Concrete Configuration

**Name and Location**

File Name: VentaLibrosWSDL

Project: ServiciosVenta

Folder: src\DescriptorBPEL Browse...

Created File: Ivan\Documents\NetBeansProjects\ServiciosVenta\src\DescriptorBPEL\Ven

Target Namespace: http://j2ee.netbeans.org/wsdl/ServiciosVenta/VentaLibrosWSDL

WSDL Type: ☐ Abstract WSDL Document ☒ Concrete WSDL Document

Binding: SOAP

Type: RPC Literal

< Back **Next >** Finish Cancel Help

### Crear descriptor WSDL del proceso BPEL

En el siguiente paso del asistente introducimos los siguientes valores:

**New WSDL Document**

**Steps**

1. Choose File Type
2. Name and Location
3. **Abstract Configuration**
4. Concrete Configuration

**Abstract Configuration**

Port Type Name: VentaLibrosWSDLPortType

Operation Name: VentaLibrosWSDLOperation

Operation Type: Request-Response Operation

**Input:**

Message Part Name	Element Or Type
ISBN	xsd:string
idCliente	xsd:string
unidades	xsd:int
precioUnidad	xsd:int

Add Remove

**Output:**

Message Part Name	Element Or Type
resultado	xsd:string

Add Remove

**Fault:**

Message Part Name	Element Or Type
-------------------	-----------------

Add Remove

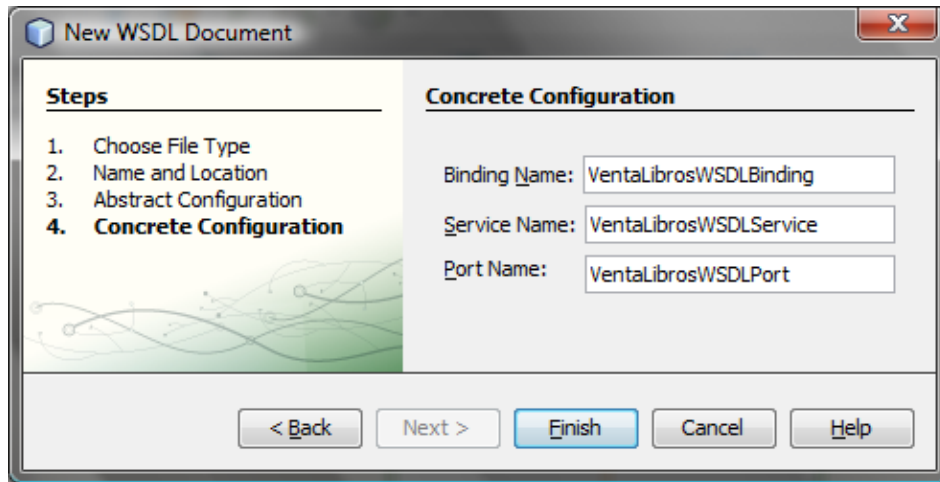
☒ Generate partnerlinktype automatically.

< Back Next > Finish Cancel Help

Definición de la operación del servicio y sus argumentos

En el último paso aceptamos los valores propuestos:





Configuración concreta del descriptor

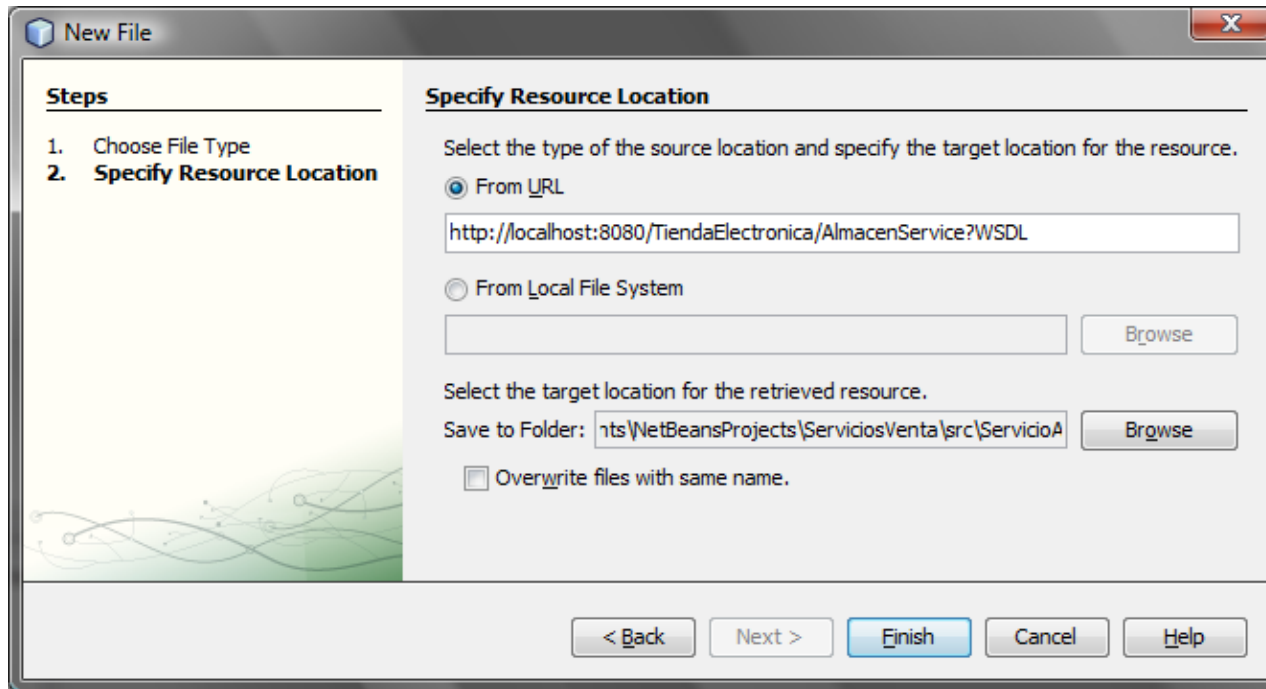
Finalizamos el asistente y ya tenemos definida la interfaz del proceso BPEL.

## Importar descriptores WSDL de los servicios involucrados

Las aplicaciones web que hemos importado en NetBeans deberán estar desplegadas sobre el servidor GlassFish previamente arrancado. Así, podemos acceder a los contratos de los web services en las URL:

- <http://localhost:8080/MensajeriaExpress/EnvioPaquetesService?WSDL>
- <http://localhost:8080/PagoOnline/PasarelaPagoService?WSDL>
- <http://localhost:8080/TiendaElectronica/AlmacenService?WSDL>

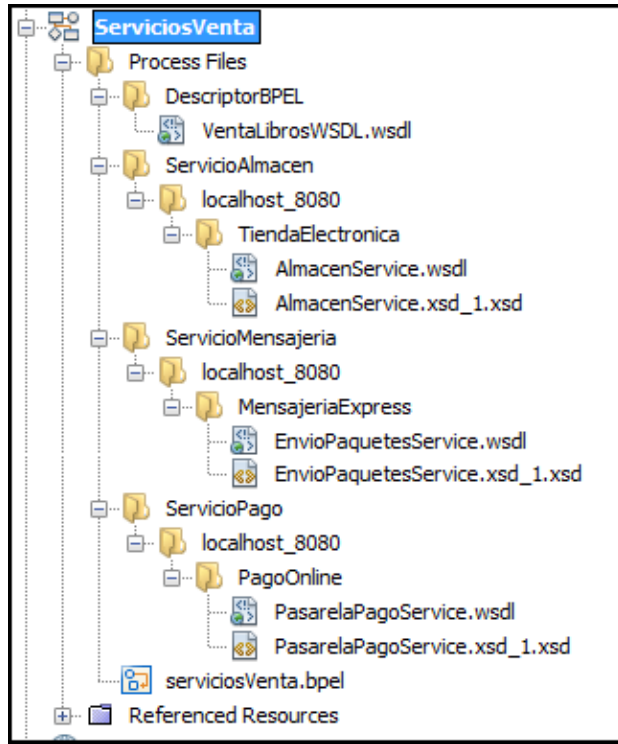
Todo proceso BPEL necesitará los contratos WSDL de los servicios que utilice, para poder invocarlos. Para importar los WSDL, hacemos botón derecho sobre la carpeta ServicioAlmacen | New | Other | XML | External WSDL Document(s) e introducimos la URL del descriptor del servicio de almacen:



Importamos los WSDL de los servicios utilizados

Repetimos la operación con las carpetas ServicioPago, ServicioMensajería y sus correspondientes descriptores.

El proyecto quedará por tanto:

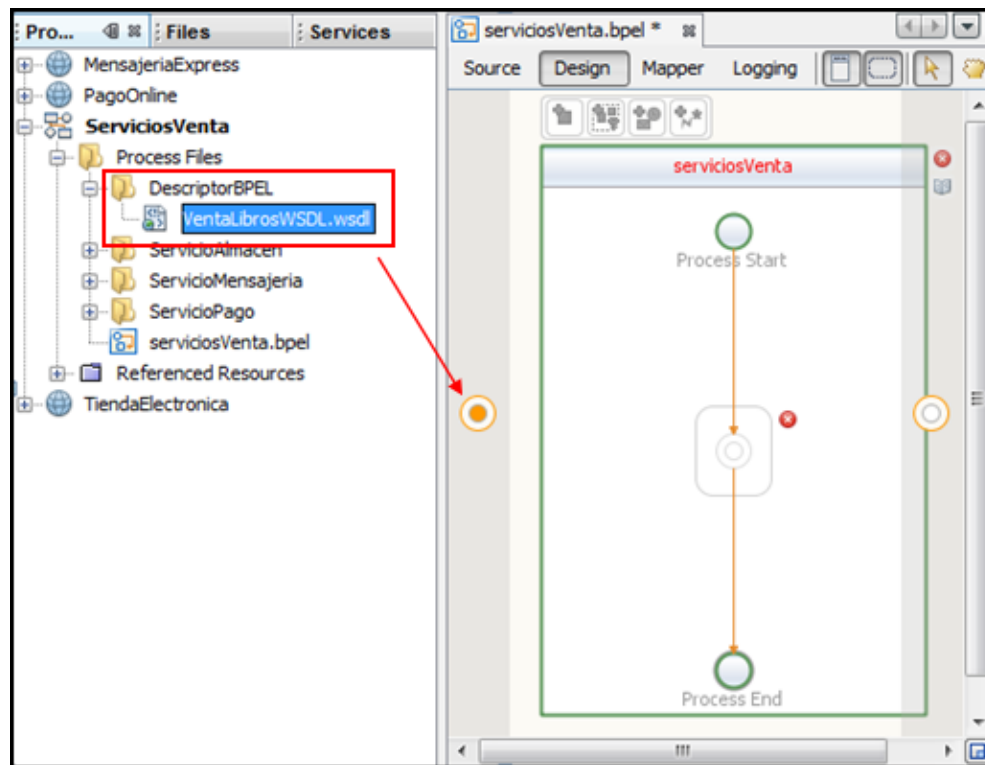


Importamos los descriptores de los webservices utilizados

Cabe decir que si disponemos los WDSL en disco en vez de resolverlos de la web, también pueden ser importados de manera similar. Incluso tendríamos una estructura de carpetas más limpia que la generada por NetBeans.

## Diseñar el proceso BPEL gráficamente

Ahora viene lo más divertido ;-). Hacemos doble click sobre serviciosVenta.bpel y accedemos al editor visual. Es intuitivo y se basa en arrastrar y soltar elementos en las zonas permitidas. En primer lugar insertamos el descriptor del proceso BPEL a la izquierda del diagrama:

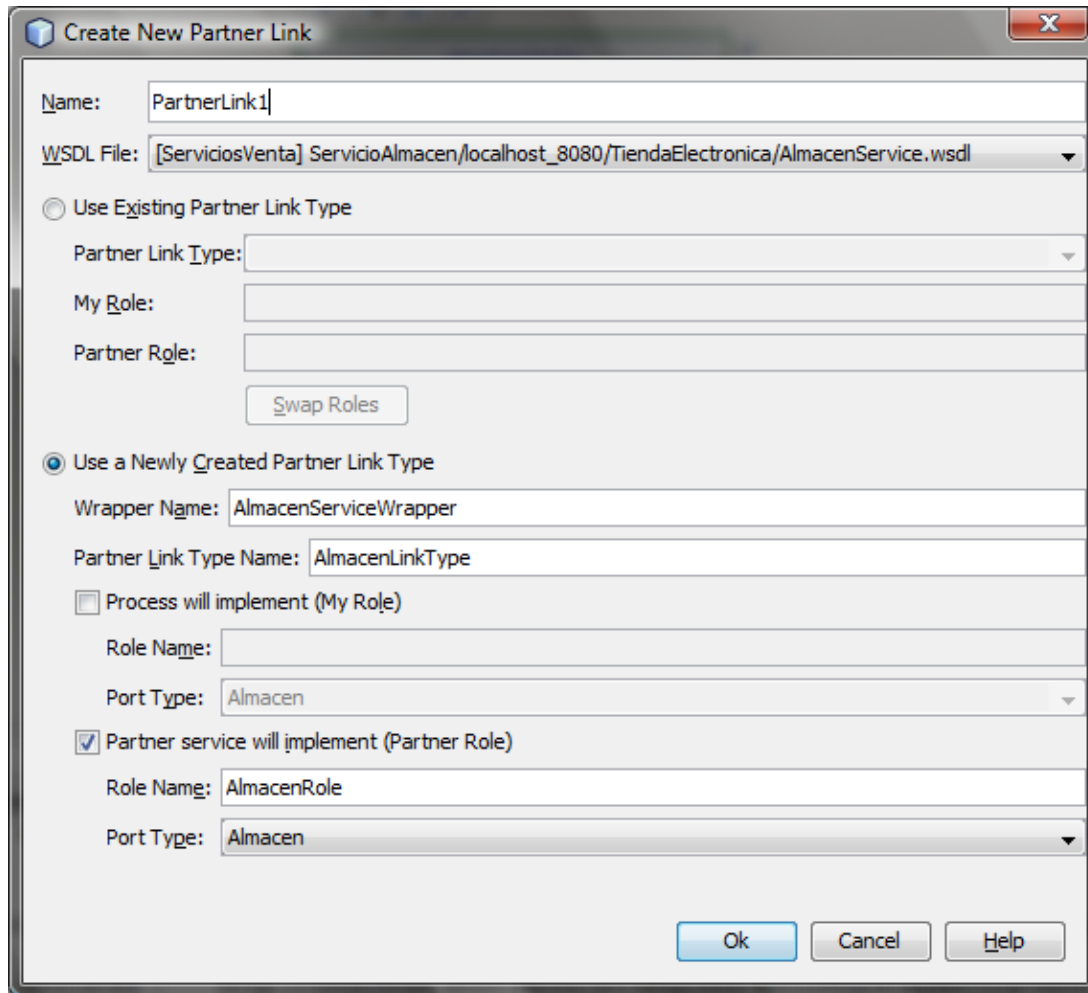


Insertar la interfaz WSDL del proceso BPEL

Pulsando sobre la representación del WSDL recién introducido y en su icono



, accedemos a las propiedades. Cambiamos el nombre por defecto PartnerLink1 por VentaLibros y aceptamos. De igual manera introducimos los descriptores de los webservices utilizados, pero esta vez a la derecha del diagrama, como proveedores de servicios. Al arrastrar y soltar, aparecerá el diálogo de creación de un nuevo enlace a un proveedor de servicio (en el cual aprovecharemos para cambiar el nombre PartnerLink1 por el del servicio que corresponda, e.g. ServicioAlmacen):



The screenshot shows a 'Create New Partner Link' dialog box. The 'Name' field is 'PartnerLink1'. The 'WSDL File' dropdown shows '[ServiciosVenta] ServicioAlmacen/localhost\_8080/TiendaElectronica/AlmacenService.wsdl'. The 'Use Existing Partner Link Type' section is disabled. The 'Use a Newly Created Partner Link Type' section is active. The 'Wrapper Name' is 'AlmacenServiceWrapper'. The 'Partner Link Type Name' is 'AlmacenLinkType'. The 'Process will implement (My Role)' checkbox is unchecked. The 'Partner service will implement (Partner Role)' checkbox is checked. The 'Role Name' for the partner role is 'AlmacenRole'. The 'Port Type' for the partner role is 'Almacen'. At the bottom are 'Ok', 'Cancel', and 'Help' buttons.

**Create New Partner Link**

Name: PartnerLink1

WSDL File: [ServiciosVenta] ServicioAlmacen/localhost\_8080/TiendaElectronica/AlmacenService.wsdl

☐ Use Existing Partner Link Type

Partner Link Type:

My Role:

Partner Role:

Swap Roles

☒ Use a Newly Created Partner Link Type

Wrapper Name: AlmacenServiceWrapper

Partner Link Type Name: AlmacenLinkType

☐ Process will implement (My Role)

Role Name:

Port Type: Almacen

☒ Partner service will implement (Partner Role)

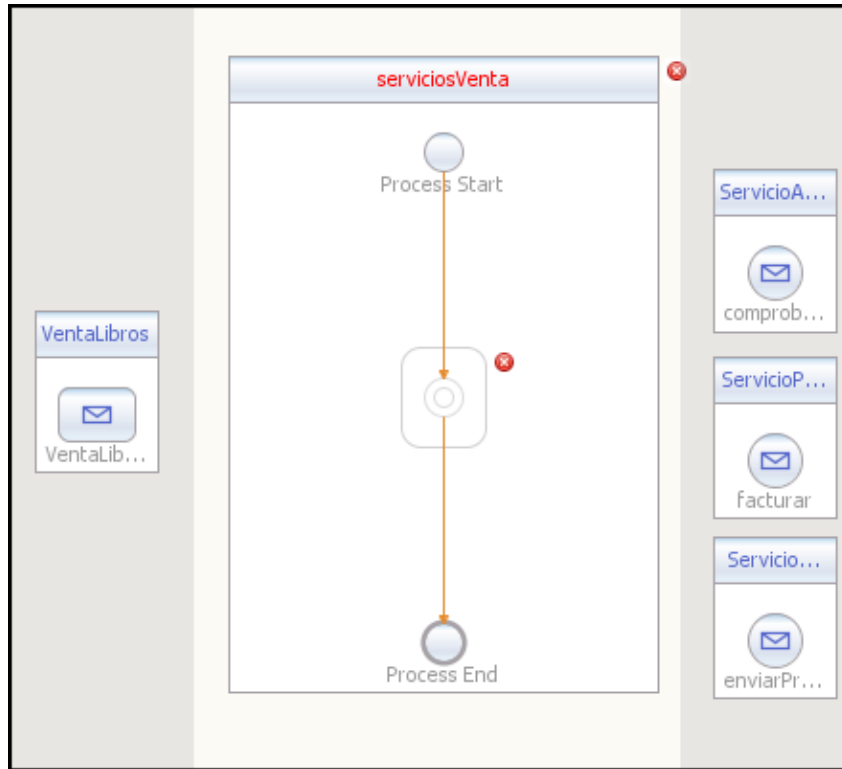
Role Name: AlmacenRole

Port Type: Almacen

Ok Cancel Help

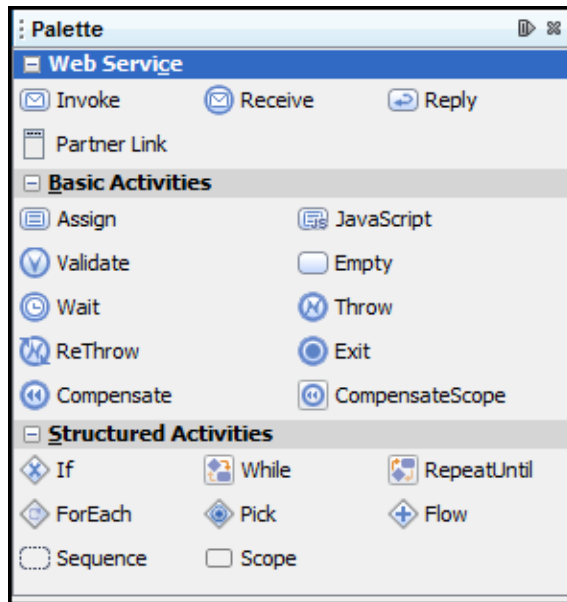
Crear enlace a servicio proveedor

El proceso quedará de la forma:



BPEL con los servicios proveedores (*PartnerLinks*)

A continuación utilizamos la paleta de actividades del editor BPEL de NetBeans:

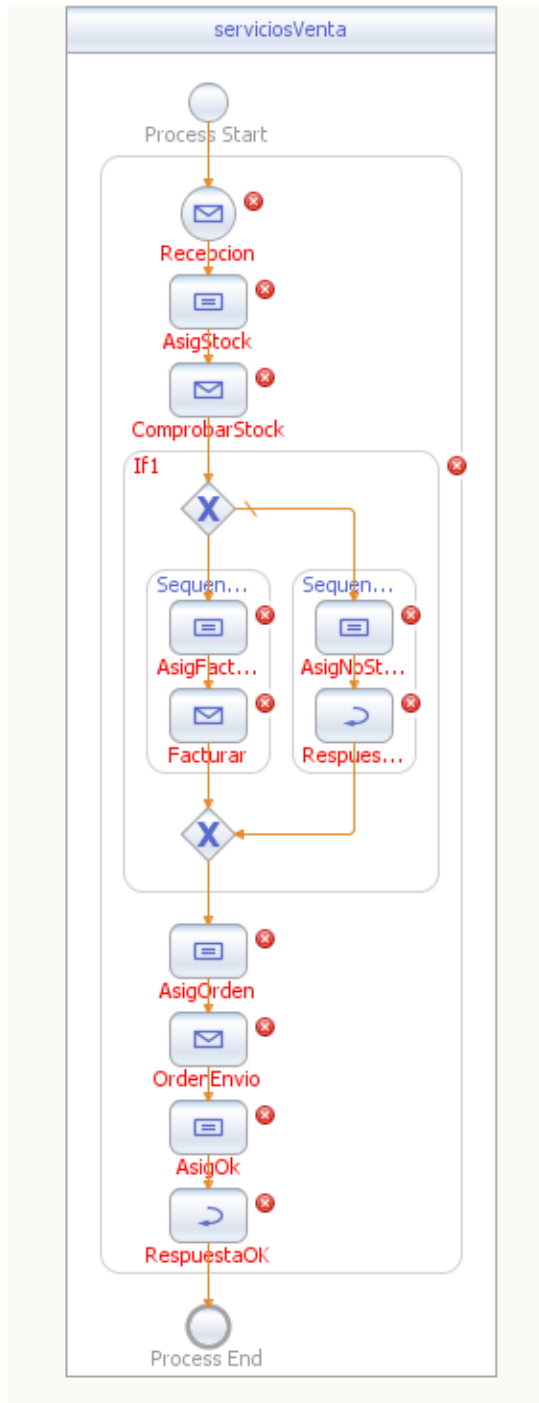


Paleta de actividades del editor de BPEL

y componemos en unos minutos un proceso como el de la imagen siguiente (para cambiar los nombres de las actividades utilizamos su icono de edición



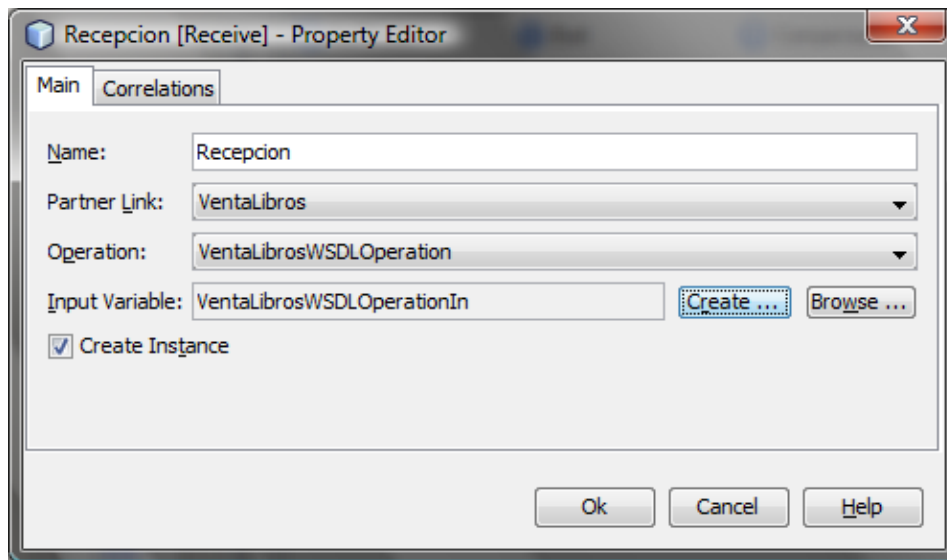
):





## BPEL inicial con sus actividades

Comenzamos asociando las actividades de recepción, invocación y respuesta a los servicios. Hacemos doble click sobre la actividad de recepción e introducimos los valores que muestra la siguiente imagen. Para obtener la propiedad Input Variable, pulsamos sobre el botón Create... y aceptamos los valores por defecto, pues son significativos:



## Editando la actividad de recepción

Tras aceptar, vemos en el BPEL que ha aparecido una flecha que une VentaLibros con la actividad de recepción. De igual manera hacemos con las actividades de respuesta. Editamos la última respuesta del proceso, RespuestaOk:

RespuestaOK [Reply] - Property Editor

Main Correlations

Name: RespuestaOK

Partner Link: VentaLibros

Operation: VentaLibrosWSDLOperation

☒ Normal Response

Output Variable: VentaLibrosWSDLOperationOut Create ... Browse ...

☐ Fault Response

Fault Name: Choose ...

Fault Variable: Create ... Browse ...

Ok Cancel Help

Editando la actividad de respuesta

Hacemos lo mismo con la actividad de respuesta contenida dentro de la estructura de decisión (RespuestaNoOk):

RespuestaNoOk [Reply] - Property Editor

Main Correlations

Name: RespuestaNoOk

Partner Link: VentaLibros

Operation: VentaLibrosWSDLOperation

☒ Normal Response

Output Variable: VentaLibrosWSDLOperationOut1 Create ... Browse ...

☐ Fault Response

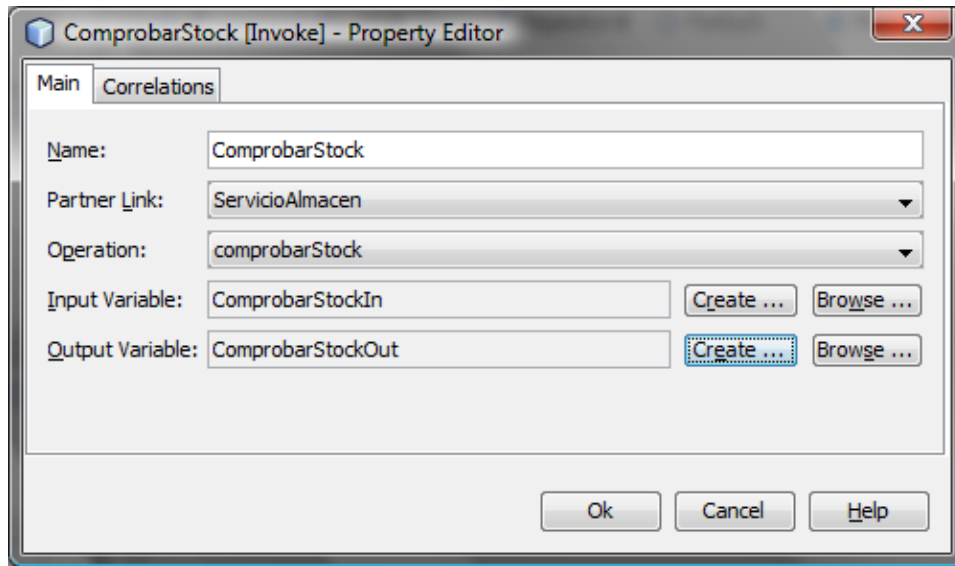
Fault Name: Choose ...

Fault Variable: Create ... Browse ...

Ok Cancel Help

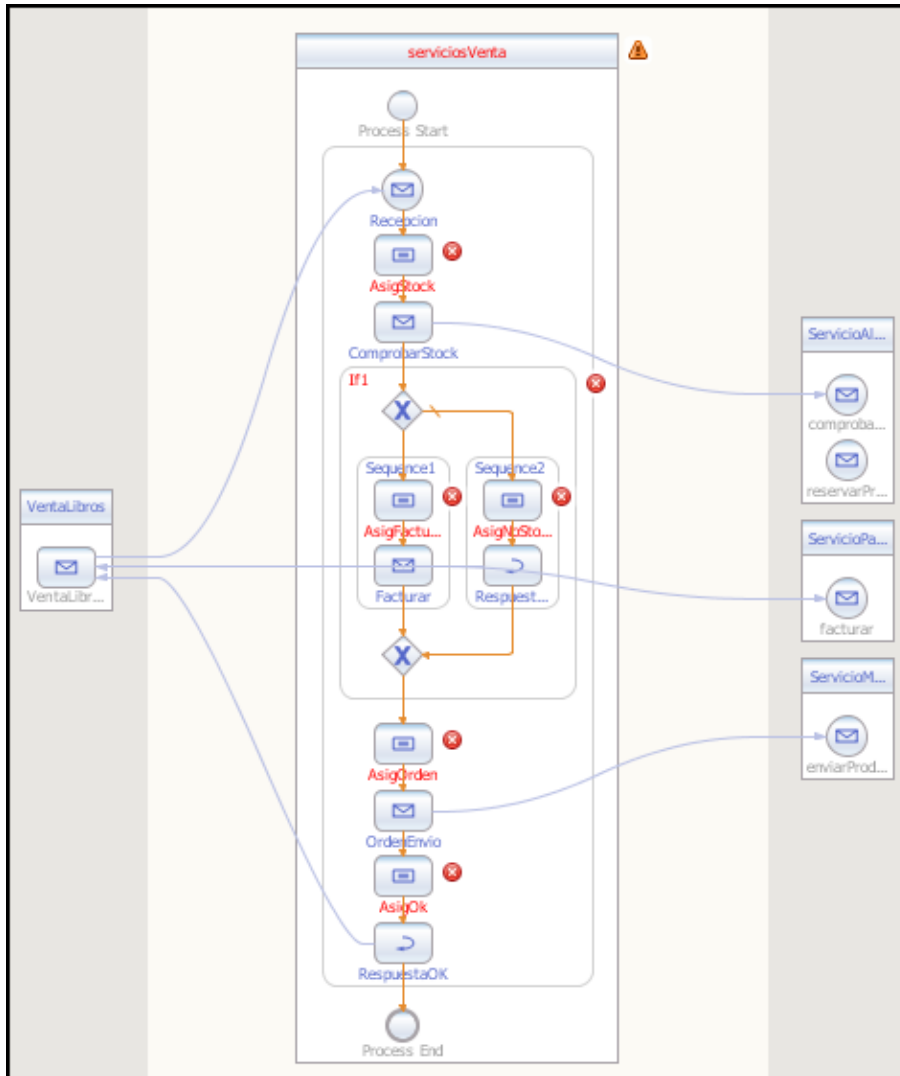
Editando la respuesta de falta de Stock

A continuación hacemos lo mismo para cada actividad de invocación. Comenzando por ComprobarStock:



Invocación del servicio de stock

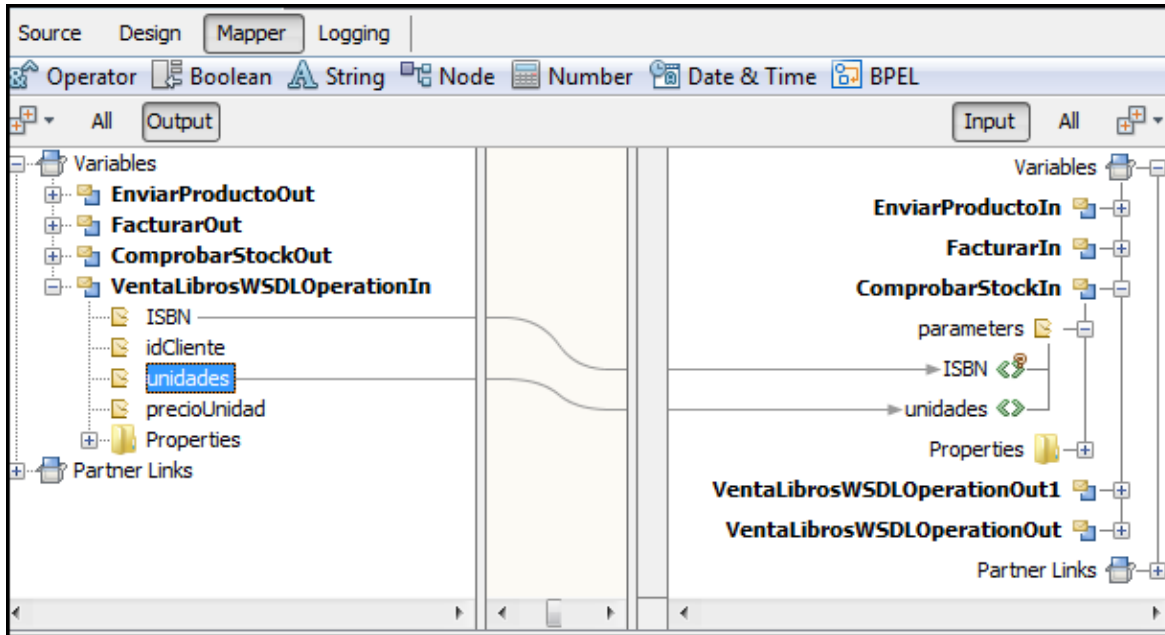
Repetimos la operación para el resto de invocaciones, y el proceso BPEL habrá tomado una forma más reconocible::



## BPEL con las invocaciones a los servicios

A continuación editamos las variables de asignación. Como su nombre indica, cada una asigna variables a la actividad que le sigue. Por ejemplo, a la actividad de invocación AsignarStock espera unos valores de entrada (los del web service al que invoca). Estos valores los tenemos que asignar de la actividad que los tenga, en este caso la de recepción. Para ello pulsamos doble click sobre la actividad AsigStock y se abre el siguiente mapeador de valores, al que hemos asociado relaciones entre partes de mensajes, pulsando en el nombre de variable origen y

arrastrando la fecha hasta el destino:



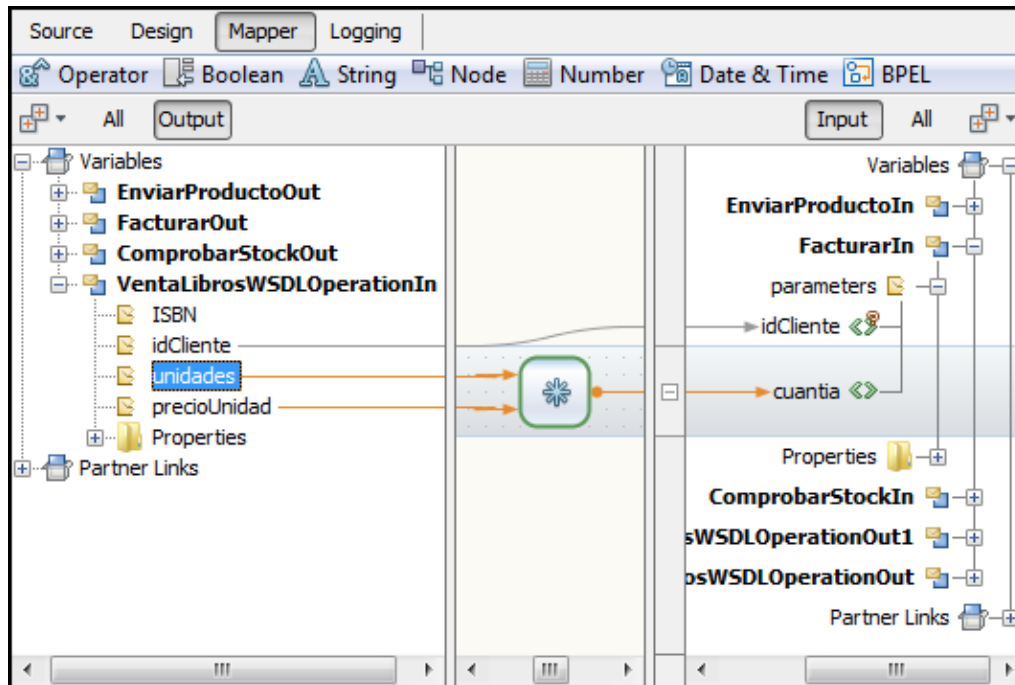
Asignación de valores entre partes de mensajes

A la izquierda hemos filtrado por valores de salida (Output) y a la derecha por variables receptoras (Input). En este caso estamos asignando los valores de la actividad de recepción (cuyo nombre de variable 'VentaLibrosWSDLOperationIn' fue definido en un paso anterior), a la variable de entrada de la invocación al web service de comprobación de Stock (ComprobarStockIn).

Para salir del mapeador y regresar el diagrama BPEL, hay que pulsar sobre la palabra Design en la parte superior del mapeador.

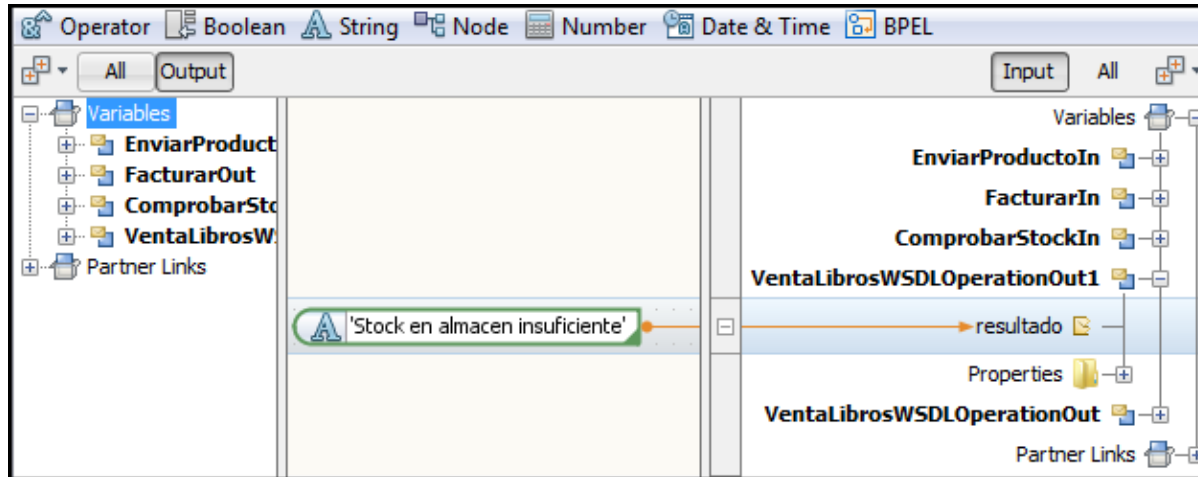
Editamos AsigFactura. En este caso El web service de facturar va a necesitar una variable cuantía, que es el resultado de multiplicar las unidades encargadas por el precio unitario. Para hacer esta transformación, seleccionamos en la parte derecha la variable cuantía. Cuando se ilumine la franja horizontal, desplegamos el menú Operator y seleccionamos Multiplication. La salida de la multiplicación la asignamos a cuantia, y los operadores de entrada son unidades y precioUnidad. Arrastramos sus nombres hacia el operador, y quedará como

en la siguiente imagen:



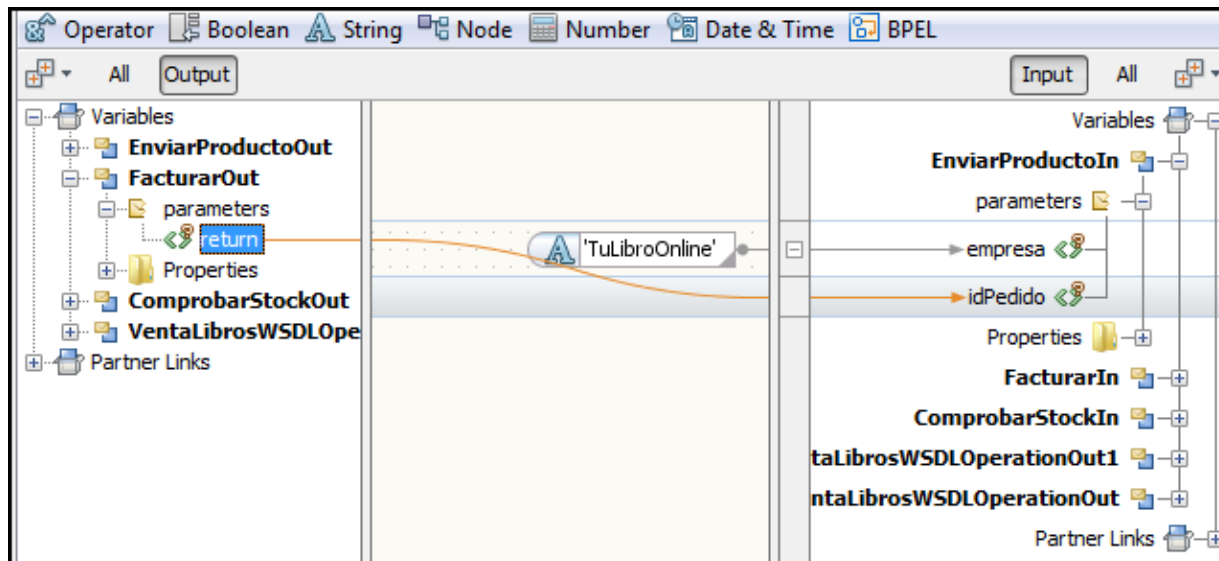
Asignar valores a la invocación del servicio de facturación

Editamos ahora AsigNoStock. Estamos en el camino del 'e/se', y tenemos que asignar a la variable de la actividad de respuesta un valor que exprese que no se ha podido completar el proceso debido a la falta de stock de producto. Dicha variable es VentaLibrosWSDLOperationOut1 y su mensaje es resultado. Seleccionamos esta variable, accedemos al menú String, seleccionamos String Literal, introducimos la cadena Stock en almacen insuficiente y lo asignamos a resultado:



Respuesta indicando la falta de stock del producto

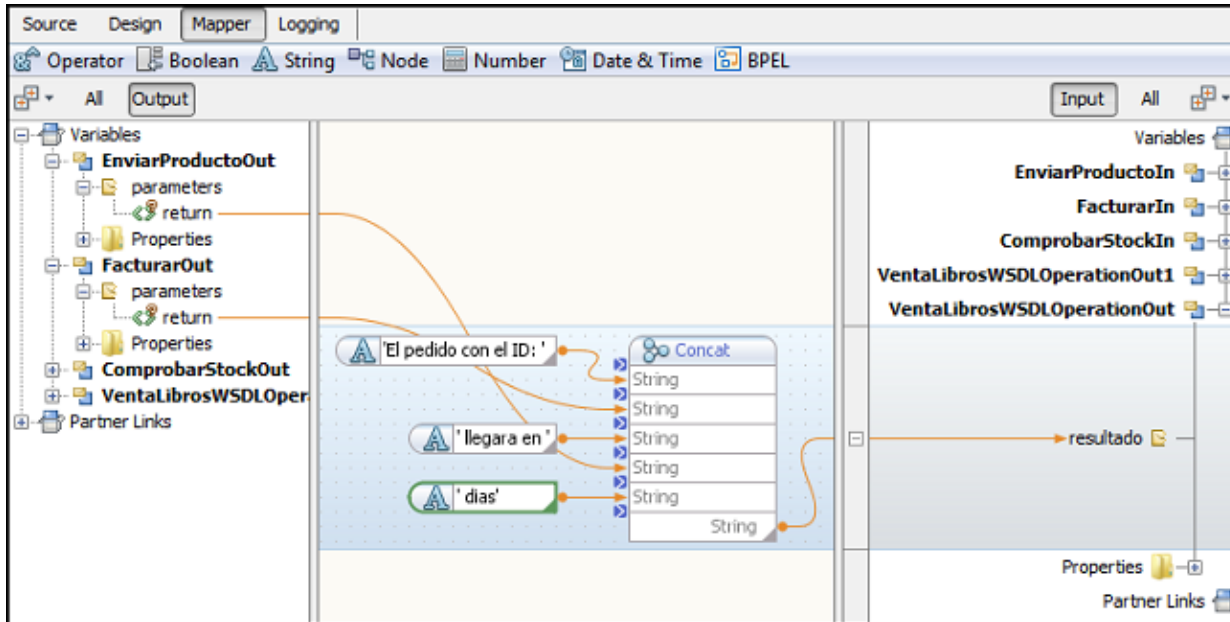
Tras la sentencia de selección nos encontramos con la actividad de asignación `AsigOrden`, que establecerá los valores de las variable `EnviarProductoIn` para la invocación al servicio de mensajería para que venga al almacén a recoger y repartir el pedido. `EnviarProductoIn` tiene dos parámetros en el mensaje: `empresa`, que será un literal, e `idPedido`, que será el resultado de la invocación al servicio de Facturación. Por tanto la asociación quedará de la forma:





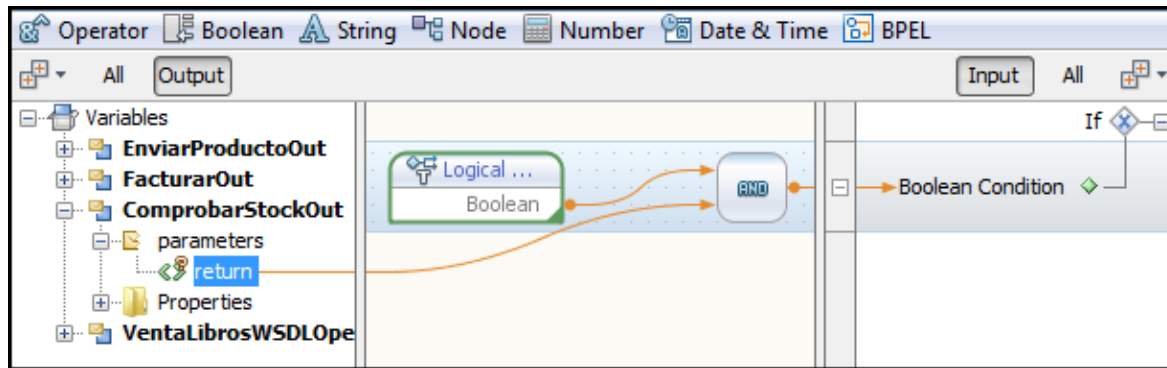
Valores para invocar al servicio de mensajería

Finalmente editamos AsigOk, que asignará a la actividad de respuesta final un mensaje de confirmación de venta indicando el id de albarán dado por el servicio de facturación y los dias que tardará en entregarse, indicado en la respuesta del servicio de mensajería. Concatenaremos en este caso varios literales con la respuestas de los servicios:



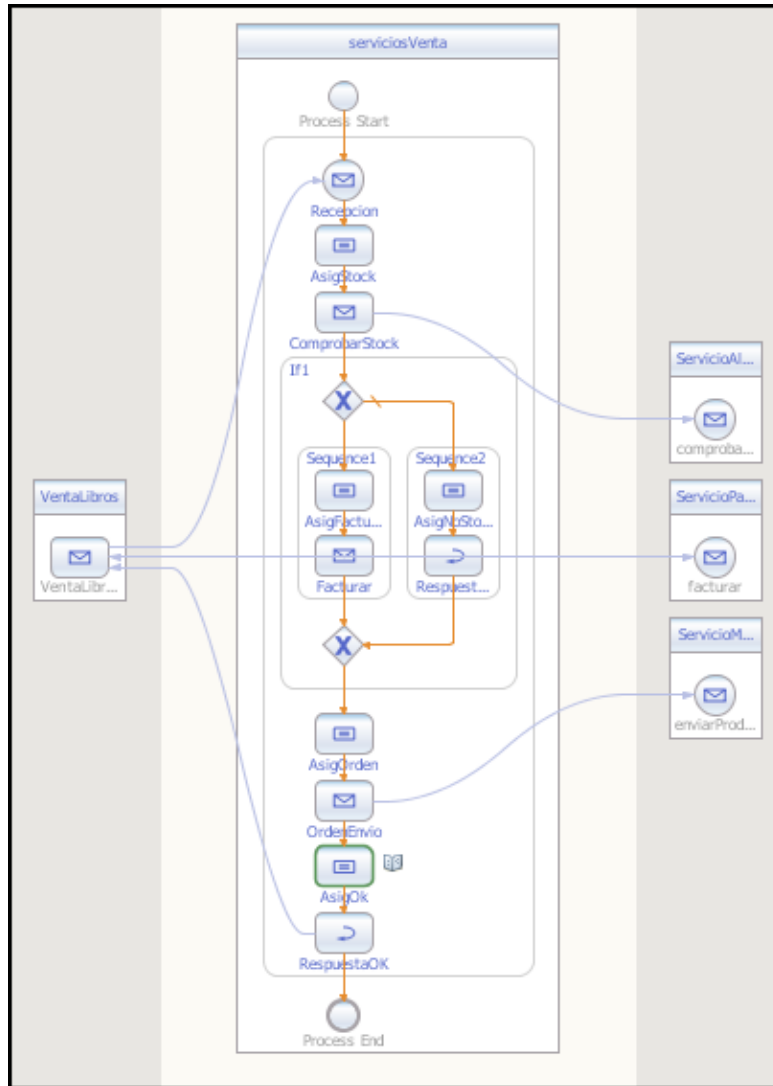
Asignacion de resultado a la actividad de respuesta

Por último tenemos que establecer la condición de la estructura de bifurcación. Pulsamos dos veces sobre la estructura del If, y utilizando el menú Boolean y los elementos Logical And y Logical True componemos la condición booleana en base a la respuesta del servicio de comprobación de stock:



Condición del proceso sobre la existencia de stock

En este punto el proceso BPEL estará finalizado, sin mensajes de error ni advertencias, listo para ser probado:



Aspecto final del proceso de negocio con BPEL

Continuación...

Este tutorial continua en [Tutorial de BPEL con OpenESB \(II\)](#)

---

Comparte este artículo!

---



Esta obra está licenciada bajo [licencia Creative Commons de Reconocimiento-No comercial-Sin obras derivadas 2.5](#)

## Respuestas (5)



**Andres Arconada**

octubre 30, 2014 at 11:44 pm · Responder

Muy buen tutorial. Funciona perfectamente en la nueva version de NetBeans 2.2



**Juan Carlos Madrazo San Jose**

octubre 30, 2014 at 11:44 pm · Responder

Muy bueno, quería ver en que consistía BPEL y este tutorial me ha ayudado a tener una visión de conjunto, explicado de una forma sencilla y agradable.



**Felix Ruiz**

octubre 30, 2014 at 11:45 pm · Responder

Muchas gracias Ivan excelente tutorial, explicas de forma clara el concepto y la tecnica para utilizar Netbeans en el tema de BPEL.

Gracias.

**Oscar José Antonio Pinto Salazar**

octubre 30, 2014 at 11:45 pm · Responder

Muy buen ejemplo, nada que envidiar a Oracle SOA Suite, excelente

**Jhonny**

enero 7, 2016 at 2:30 am · Responder

Hola, estoy aprendiendo acerca de BPEL y web service, segui los pasos del tutorial y la verdad no se como probarlos, podrias indicarme como o enviarme algun tutorial por favor ? muchas gracias de antemano.

Deja un comentario

---

#### Lo último

Pruebas multiplataforma con Selenium GRID

Un entorno de programación en la nube: un vistazo a Cloud 9 IDE

Primeros pasos con Gulp

Primeros pasos con Kubernetes

Instalación de Kubernetes en

Ubuntu con Ansible

---

#### Datos de contacto

Edificio BestPoint Avd. de Castilla,  
1, Planta 2, Oficina 21B (San  
Fernando de Henares)

**Phone:** 916 75 33 06

**E-Mail:** [info@adictosaltrabajo.com](mailto:info@adictosaltrabajo.com)

**Web:** <http://www.autentia.com>

---

#### Powered by



Copyright 2003-2016 © All Rights Reserved | Texto legal y condiciones de uso