Metodologías y Tecnologías de la Integración de Sistemas

Segunda entrega - 22/05/2018

Equipo

Iván Mora Maciá

Joaquín Vasalo Vicedo

Arancha Ferrero Ortiz de Zárate

Jaime Moreno Cantó

Álvaro Muñoz Delgado

Jaime Sarrión Sahuquillo

Índice

1. Escenario de integración

Nuestro escenario de integración será la famosa compañía Tesla, conocida por ser el fabricante de coches eléctricos estrella.

Tesla es una compañía estadounidense ubicada en Silicon Valley, California, que diseña, fabrica y vende coches eléctricos, como el Tesla Roadster, componentes para la propulsión de vehículos eléctricos y desde sistemas de almacenamiento a baterías.

En nuestro caso, nos centraremos en un apartado en concreto, la fabricación, programación y distribución de paneles energéticos solares.

En concreto, desarrollaremos 6 flujos de trabajo:

- ➔ En SOAP:
  - ◆ QA de Aplicaciones Software
  - ◆ Programación de paneles energéticos movibles
  - ◆ Subida de archivos de trabajo
- ➔ En REST:
  - ◆ Contratar personal
  - ◆ E-commerce de baterías y placas
  - ◆ Manufactura de paneles solares

Front End

Para la parte front-end de la aplicación, hemos utilizado HTML, CSS, Javascript y el lenguaje **PHP**, versión 7.1, a través del framework **Laravel**, en su versión 5.5.

El propio framework dispone de la herramienta Blade para gestionar las vistas. Disponemos de una plantilla (*layout)* principal que contiene el header, un *sidebar* y el *footer*, facilitando la edición de las mismas.

Back End

Hemos utilizado dos tecnologías diferentes para el desarrollo completo del back-end de la aplicación. Por un lado, para el desarrollo de las diferentes APIs de los flujos REST hemos utilizado **NodeJS** (versión 8.9.4), una librería de Javascript.

Por otro lado, para el desarrollo de los flujos SOAP hemos utilizado **Java Web Service**, que consume los contratos definidos para cada flujo previo a su orquestación con BPEL. La lógica de negocio se ha definido en el *skeleton* de cada servicio, añadiendo la capa de acceso a datos como clase java.

Bases de datos

Como sistema de gestión de bases de datos hemos utilizado MySQL, aprovechando las funcionalidades que ofrece el programa XAMPP como, por ejemplo, PhpMyAdmin para gestionar visualmente las bases de datos y comprobar la persistencia de los datos.

Hemos creado en total 6 bases de datos, una para cada flujo orquestado:

- SOAP - Subida de Ficheros → tesla_ficheros
- SOAP - Programación de paneles solares → mtispracticagrupo
- SOAP - QA Aplicaciones de Software → proceso_qa
- REST - Contratar personal → mtis_grupal_contratar_personal
- REST - Manufactura de paneles solares → paneles_solares
- REST - E-Commerce → almacentesla

2. Estilos de integración aplicados: BPEL, ESB

Dado que hemos desarrollado diferentes flujos SOAP y REST, hemos basado la decisión de si utilizar BPEL o ESB para manejar el comportamiento de cada flujo en la **automatización** y las **dependencias externas**.

SOAP

Todos los flujos SOAP se han orquestado mediante BPEL en Eclipse.

Programación de paneles energéticos móviles

El flujo al ser casi 100% automatizable y sin dependencias externas era uno de los candidatos favoritos para programarlo con BPEL utilizando la metodología SOAP dando a los servicios web que lo componen una orquestación que solo depende de la entrada en formato JSON que recibe el orquestados general en Mule. El cual le proporciona una hora en formato entero, los sensores adquieren de la base de datos los respectivos resultados de la hora señalada, posteriormente los datos obtenidos se procesan en el siguiente servicio web que calcula cuál es la posición más ventajosa para el panel solar. Estas posiciones(o la activación de defensas) son entregadas al tercer y último servicio web que se encarga de mover el panel a las coordenadas indicadas por el servicio web predecesor a este.

QA de aplicaciones software

Dado que se trata de un flujo que podría llegar a tener alguna dependencia externa, hemos optado por la solución que más se adapta a las necesidades del proyecto, por lo que se ha automatizado y eliminado toda dependencia externa que impida su ejecución individual. De esta manera, se ha orquestado el flujo en BPEL, incluida la persistencia de datos de los servicios “Ejecutar tests” y “Generar informe”.

Subida de archivos de trabajo

La subida de archivos de trabajo se trata de un proceso inherente al trabajador, por lo cual es 100% automático desde que el fichero se procesa hasta que se envía el destinatario, por lo que se ha decidido emplear la orquestación en BPEL del proceso.

La validación, subida, procesado y envío del fichero son acciones que el servidor realiza automáticamente, por lo que no es necesario la actuación de un trabajador para el proceso.

REST

Todos los flujos REST se han orquestado mediante Mule ESB en AnyPoint Studio.

Contratar personal

El objetivo del flujo consiste en automatizar lo máximo posible el proceso de contratar personal en una empresa. Al tener bastantes "cortes" (necesidades de decisión humana no automatizable) no era candidato para SOAP, sino para REST.

Por lo que, el flujo que sigue sería: primero, crear una oferta, publicarla, (esperar, aunque nosotros tenemos datos de prueba ya introducidos) recibir solicitudes de diversos perfiles al puesto, valorar las solicitudes candidatas y, por último, elegir entre los candidatos al trabajador/es.

E-commerce de baterías y placas

Este flujo es el que se encarga de dar soporte al cliente que quiere comprar los productos que estamos ofertando. Mediante este flujo, el cliente puede ver los productos que tenemos en el catálogo, en la sección ‘e-commerce’ del menú lateral, una vez tiene en mente el producto que quiere poseer, debe darle a comprar producto. Una vez ahí, introducirá su id, y el id del producto que quiere comprar, se realizarán una serie de comprobaciones muy sencillas, y si todo ha salido bien, se mostrará un mensaje de color verde y la factura será añadida a la base de datos; en caso de que no haya salido bien, por que se necesitan más productos, la aplicación mostrará un mensaje diciendo el número de productos que hay que pedir al proveedor y hará una llamada a dicho proveedor, en este caso el flujo de manufactura de paneles solares, y le hará un pedido.

Manufactura de paneles solares

Mediante este flujo vamos a poder controlar el estado de los pedidos de fabricación de paneles solares. Cuando creamos un pedido se pondrá en estado “Nuevo”, lo que significa que está pendiente de fabricar. Podemos ir avanzando en el estado del pedido con un solo click. En concreto si el estado es “Nuevo”, se realizarán una serie de comprobaciones internas para comprobar si se dispone de la cantidad de material suficiente para fabricar la cantidad de paneles solares encargada en el pedido: en caso negativo, el pedido pasará a espera de la llegada de materiales y se creará un pedido por cada material con pocas existencias para enviarlo a los proveedores, además de incrementar la cantidad de materiales; en caso positivo el pedido pasará a construcción y se restará la cantidad de materiales que se usarán en la fabricación.

Integración de los procesos de negocios

Cómo se han integrado los procesos de negocios

Una vez desarrollados los flujos SOAP y REST mediante las tecnologías correspondientes, hemos ido integrando los flujos 1 a 1 para ir detectando posibles problemas y no extender los errores al resto de flujos.

Para maximizar el desacople entre procesos, hemos establecido una ruta para cada uno de ellos manejados por el conector “Choice”. Además se han separado las llamadas a la API en componentes APIKit separados del XML principal del proyecto.

Pasos seguidos para la integración

Los pasos que hemos seguidos para la integración de los diferentes servicios SOAP son los siguientes:

Para los servicios SOAP:

1) Creación de la base de datos del servicio
   - Creada en el servidor web XAMPP
2) Servicios Web
   - Se han añadido los diferentes servicios web al servidor Tomcat
3) Bpel
   - Se han publicado los diferentes procesos en BPEL en el servidor Ode
4) Mule
   - Se han creado los flujos independientes de cada proceso
   - Para navegar por los diferentes flujos se ha establecido un conector “Choice” que evalúa la ruta por la que ha de navegar la petición del cliente
   - El Choice navega directamente al flujo del proceso correspondiente
   - Se devuelve el resultado en formato JSON

Para los servicios Rest:

1) Creación de la base de datos del servicio
   - Creada en el servidor web XAMPP
2) Raml
   - Se han hecho las definiciones de las diferentes APIs
3) Desarrollo de los endpoints de la API mediante Node.js
4) Mule

Pasos finales en común:

1) Organización de los flujos en Mule ESB
- El conector “Choice” evalúa la ruta del cliente y da paso al flujo correspondiente, ya sea BPEL o API REST

2) Creación del cliente
- Se ha incluído un controlador por cada proceso (BPEL/API) con una propiedad de clase que representa la URL del endpoint de la ruta del “Choice” correspondiente
- Se ha creado una vista de navegación complementada con una barra lateral con una sección por cada proceso

3) Pruebas
    - Pruebas con Postman
    - Pruebas con el debugger de Anypoint Studio

3. Problemas planteados y soluciones

En este apartado, vamos a recopilar los problemas y retos que hemos tenido que superar durante el desarrollo de la práctica, tanto en el desarrollo de los flujos como problemas generales del proceso y de la integración.

Problemas detectados en SOAP

La mayor parte de los problemas se han encontrado a la hora de hacer el BPEL por la aplicación más que por la tecnología. Eclipse se tenia que reiniciar varias veces tanto cuando elimina las asociaciones de las variables como a la hora de encender el tomcat donde daba fallo de puertos al iniciarlo y la unica solucion es reiniciar el eclipse.

Otro problema que hemos encontrado respecto al BPEL ha sido la necesidad de retocar el contrato de los servicios web de la subida de fichero, por lo que ha sido necesario volver a importar el WSDL al proyecto BPEL y volver a crear el partner link correspondiente, volviendo a configurar los diferentes invokes y assigns.

En el flujo de archivos de trabajo, el principal problema ha sido la subida del fichero en sí. Se ha solucionado pasando directamente el contenido del fichero y su extensión, para así convertirlo en el supuesto cliente del destinatario. Se podría haber solucionado con el aloje del fichero en sí en el servidor destinatario, pero no es el caso de uso de esta asignatura por lo que se ha optado por esta manera.

Problemas detectados en REST

El problema que nos ha surgido con este flujo ha sido básicamente el RAML debido a que lo realizamos en primer lugar antes de ponernos a desarrollar la API. Esto ha provocado que haya que realizar cambios en el RAML en varias ocasiones para ajustarlo al funcionamiento de los endpoints de la API, debido a que debe encajar a la perfección para que el ApiKit Router pueda redirigir correctamente hacia el endpoint correspondiente y acepte las entradas y salidas.

4. Conclusiones

Gracias al uso de las tecnologías estudiadas durante el transcurso de la asignatura, hemos sido capaces de desarrollar nuevas habilidades relacionadas con la creación y sobretodo orquestación de servicios web. Valorando en todo momento la dificultad y sobretodo el tiempo que requiere la orquestación de servicios, trabajo que actualmente es un perfil cada vez más demandado en el mundo profesional de la ingeniería informática para el desarrollo de software.

Además trabajando en equipo hemos sido capaces de desarrollar una aplicación completamente operativa sobre un escenario hipotético de integración en la compañía de Tesla. El cual nos a proporcionado una visión global sobre qué pasos se deben de seguir para la integración de un proyecto llevado a cabo por más de una persona.