Aplicaciones con Enfoque Orientado a Servicios

Tema Nº9:Trabajando con Mule ESB - II

Indicador de logro Nº9:Implementa el control de excepciones en aplicaciones distribuidas que consumen servicios web

**TEMA 01 Teoría de los**

Imagen que contiene Icono

Descripción generada automáticamente

**TEMA Nº9:**

Trabajando con Mule ESB - II

**Subtema 9.1:**

Manejo de excepciones

Mule ofrece numerosas opciones para manejar errores. A estas fallas se les conocen como excepciones. Para administrar estas excepciones, se pueden configurar estrategias de excepción.

La estrategia de excepción predeterminada de Mule, que se aplica implícitamente a todas las aplicaciones, gestiona los errores (es decir, las excepciones lanzadas) en los flujos. Cuando sus flujos requieren una gestión de errores más sofisticada, puede implementar una o más estrategias de excepción para construir protocolos precisos y eficientes para manejar errores.

Desde una perspectiva de alto nivel, los errores se dividen en una de dos categorías:

* Excepciones del sistema

Se invoca una estrategia de excepción del sistema cuando se lanza una excepción a nivel del sistema (es decir, cuando no hay ningún mensaje involucrado, las excepciones se manejan mediante estrategias de excepción del sistema).

* Excepciones de mensajería

Se invoca una estrategia de excepción de mensajería cada vez que se lanza una excepción dentro de un flujo (es decir, cuando hay un mensaje involucrado, las excepciones se manejan mediante estrategias de excepción de mensajería).

Cuando un mensaje que se procesa a través de un flujo Mule genera una excepción, la ejecución normal del flujo se detiene y los procesos se transfieren a la secuencia del procesador de mensajes dentro de la estrategia de excepción.

**Ejemplos:**

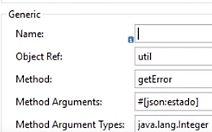
1. En *Anypoint Studio*, cree un nuevo archivo de configuración de Mule *(New Mule Configuration File)*.
2. En el flujo principal añada un componente *HTTP* que reciba un JSON en el método POST y un *Logger* que muestre el mensaje recibido.
3. Añada una clase Util.java en la carpeta de código fuente.
4. Dentro de la clase Util.java, agregue el siguiente método que servirá para forzar que el sistema genere una excepción.



1. En el flujo principal, añada al final un componente *Invoke* para utilizar el método de la clase Util.java



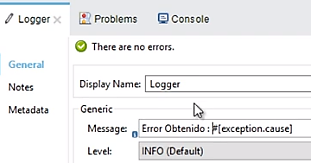
1. Configure el componente de la siguiente forma:



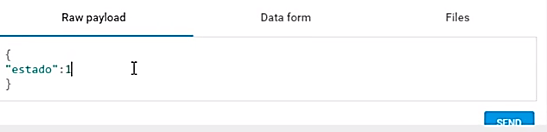
1. Luego, añada un componente *Catch Exception Strategy* en la sección Error handling.



1. A continuación, añada un componente *Logger* dentro del *Catch*. Configúrelo de la siguiente forma:



1. Ejecute su aplicación.
2. A través de Postman, envíe un mensaje en formato JSON



1. Verifique en la consola.



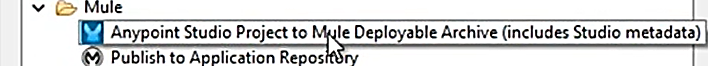
**Subtema 9.2:**

Despliegue en Mule StandAlone

Hasta ahora hemos visto como ejecutar nuestra aplicación desde *Anypoint Studio*. Esto es posible porque la aplicación tiene un servidor Mule integrado. Sin embargo, en producción necesitamos desplegar nuestra aplicación en un servidor Mule independiente.

**Ejemplos:**

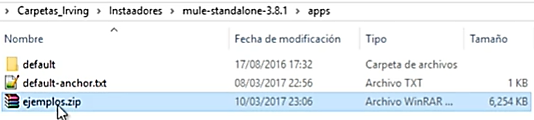
1. Seleccione el proyecto que desea desplegar, haga clic derecho y seleccione la opción *Export*.
2. En la nueva ventana, despliegue la opción *Mule* y seleccione la opción indicada:



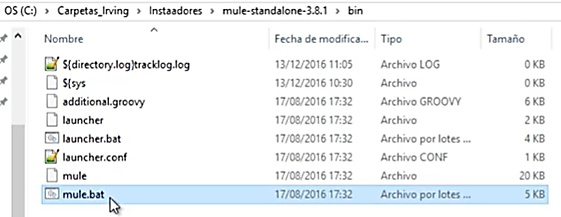
1. Indique la ruta del directorio donde exportará el proyecto (generará un archivo .zip) y haga clic en *Finish*.



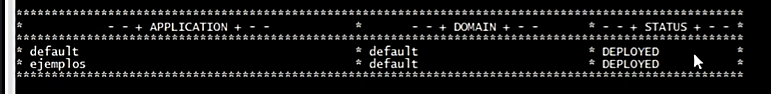
1. En el directorio indicado, verifique la creación del archivo .zip



1. Inicie el servidor *Mule StandAlone* haciendo doble clic al archivo **mule.bat** que se encuentra ubicado en la carpeta bin de la carpeta del servidor descargado.



1. Se abrirá la consola del servidor y verifique que se haya desplegado el proyecto (tendrá el mismo nombre del archivo \*.zip generado)



1. Sin correr la aplicación desde *AnypointStudio*, pruebe su aplicativo (en el ejemplo se envía un JSON) y verifique el resultado esperado.



**Subtema 9.3:**

Consumo de servicios Soap y REST

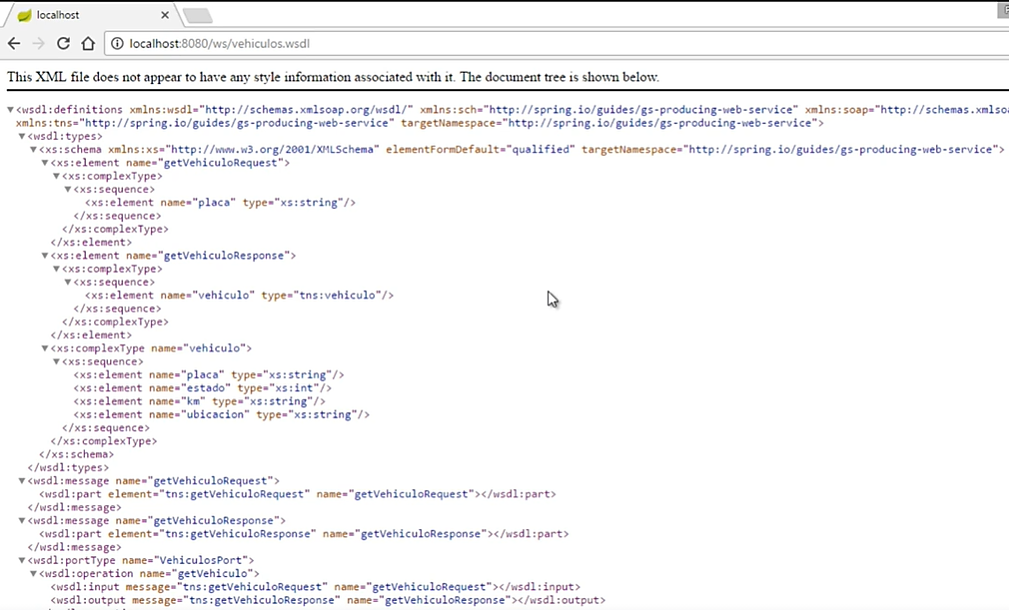
Mule ESB se basa en el concepto de arquitectura orientada a servicios (SOA), por lo que ahora es el turno de integrar en nuestros flujos el consumo de servicios web, tanto Soap, como los que usan arquitectura REST.

**Ejemplos:**

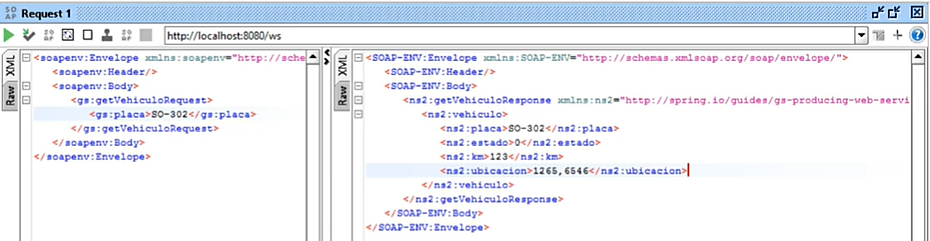
**Consumo de servicio Soap**

Para este ejemplo, se ha construido previamente un servicio Soap que realiza la búsqueda de un vehículo según su placa. El vehículo es una entidad con atributos placa, estado, km y ubicación).

El servicio ha sido desplegado desde Spring Tool Suite en el puerto 8080 del equipo local. A continuación, se muestra su archivo WSDL, donde se observa que el método es *getVehiculo*, y que en el *request* o petición recibe un parámetro placa y retorna como respuesta o *response* un vehiculo.

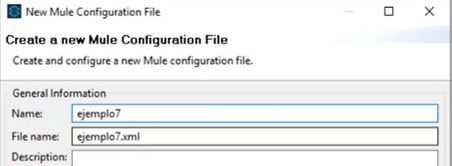


En SoapUI se prueba el servicio para comprobar su funcionalidad.

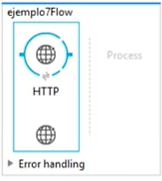


En Anypoint Studio:

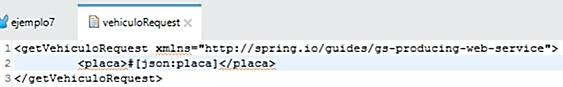
1. Cree un nuevo archivo de configuración de Mule *(New Mule Configuration File)*.



1. Añada un componente *HTTP* al flujo.



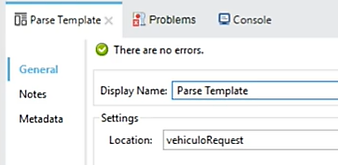
1. Para este ejemplo, utilizaremos una plantilla para esperar la petición. Por ello, en la carpeta *src/main/resources* añada un nuevo archivo llamado **vehiculoRequest** que tendrá una plantilla del request esperado.



1. Añada un componente *Parse Template* al flujo.



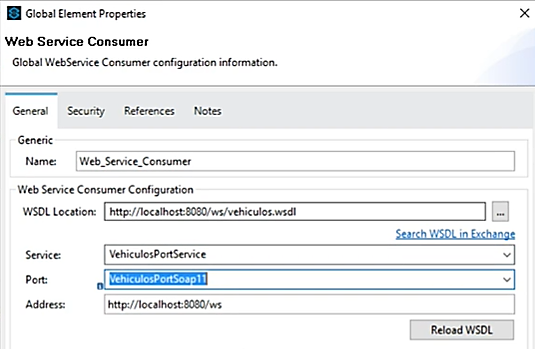
1. Configure el *template* indicando la plantilla del request: **vehiculoRequest**



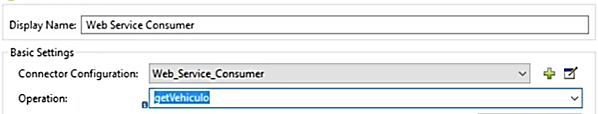
1. Añada un componente *Web Service Consumer* para consumir el servicio web SOAP.



1. Configure el componente haciendo clic en el botón **+** de la opción *Connector Configuration,* e ingrese la ruta del WSDL a consumir.



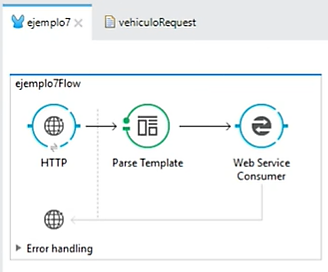
1. En la opción *Operation* seleccione el método a consumir. En este caso es *getVehiculo*.



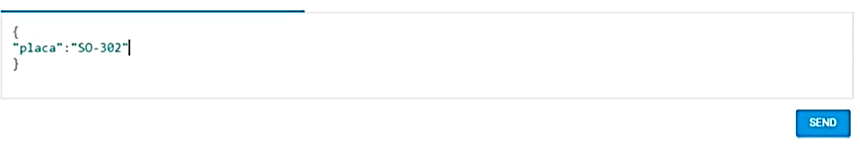
1. En el componente *HTTP* realice la siguiente configuración:



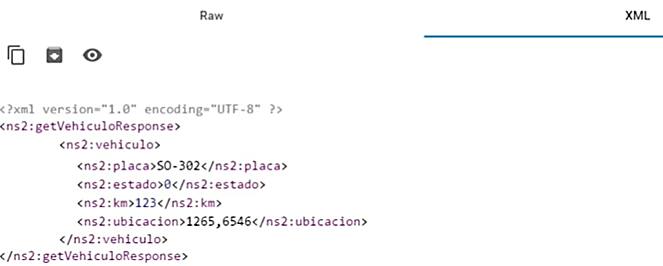
1. Verificamos nuestro flujo



1. Despliegue el proyecto en *Mule StandAlone*.
2. Pruebe su aplicativo enviando un JSON con la placa a



1. Verifique la respuesta obtenida.



**Consumo de servicio REST**

Para este ejemplo, se ha construido previamente un servicio REST que recibe a través de POST un bean Identificacion y retorna una entidad Vehiculo con data de prueba.

El servicio ha sido desplegado desde Spring Tool Suite en el puerto 8080 del equipo local. A continuación se muestra el JSON del request y el response obtenido.

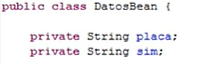


En Anypoint Studio:

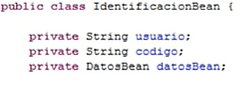
Creación de clases

Agregue las clases DatosBean.java e Identificacion.java para utilizarlas en el ingreso de datos.

Las clases deben contener métodos GET/SET

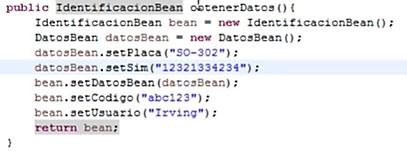


Clase DatosBean.java



Clase IdentificacionBean.java

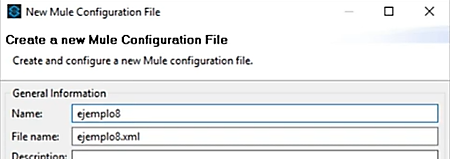
En la clase Util.java agregue el método:



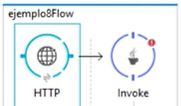
Clase Util.java

Creación de proyecto

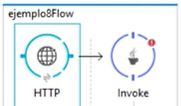
1. Cree un nuevo archivo de configuración de Mule *(New Mule Configuration File)*.



1. Añada un componente *HTTP* al flujo.



1. Añada un componente *Invoke* al flujo.



1. Configure el componente *HTTP*.



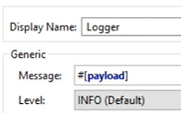
1. Configure el componente *Invoke* para utilizar la clase java Util.java.



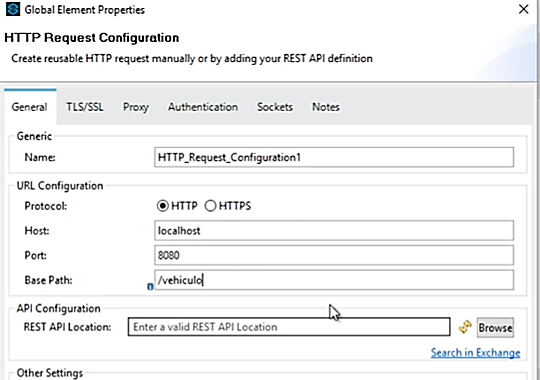
1. Para pasar la representación de un objeto a JSON (que espera el servicio REST), añada el componente *Object to JSON* al flujo.



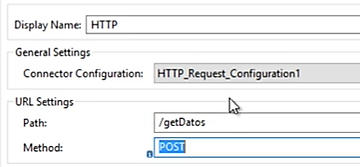
1. Añada un componente *Logger* al flujo para mostrar el *payload*.



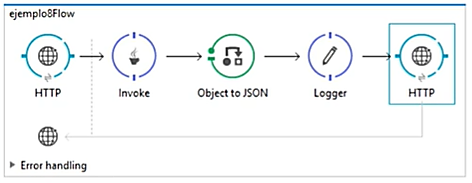
1. Añada un control *HTTP* para invocar al servicio REST y configure la petición



1. Configure el *path* y el método POST del componente HTTP.

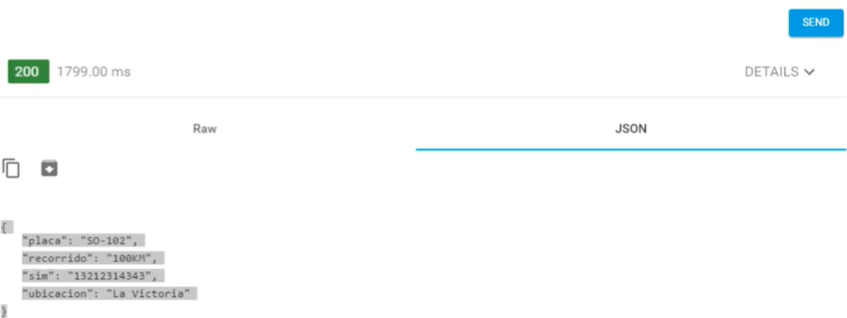


1. El flujo final debe ser:

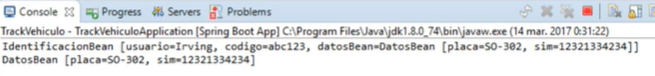


1. Despliegue su componente en *Mule StandAlone*.
2. Pruebe su proyecto.

Recuerde que la ruta es diferente a la del servicio REST y que el objeto JSON está siendo enviado desde el proyecto, por lo que no es necesario enviarlo desde Postman.



Respuesta en Postman

**

Vista de consola

**Actividad:**

1. A partir del servicio SOAP de calculadora, elabore un proyecto que reciba dos parámetros para las operaciones matemáticas del servicio y muestre en un control Logger el resultado de la operación

Además, implemente el control de las excepciones que podrían originarse (por ejemplo, la división por cero)

Servicio SOAP: <http://www.dneonline.com/calculator.asmx>

WSDL: <http://www.dneonline.com/calculator.asmx?WSDL>

El despliegue debe realizarse usando *Mule StandAlone*.

1. A partir del servicio REST de listado de universidades por país, elabore un proyecto que reciba el nombre del país y muestre únicamente la lista de los nombres de las universidades.

Servicio REST: <http://universities.hipolabs.com/search?country=United+States>

El despliegue debe realizarse usando *Mule StandAlone*.