

# Taller Patrones de Arquitectura-ESB

Karen Paola Duran Vivas

En este taller se trató de hacer algo similar a la arquitectura de ServiceMix que consiste en un Enterprise Service Bus que se conecta a diferentes servicios o aplicaciones por medio de interfaces, las cuales se comunican a través de este. Entonces teníamos básicamente dos actores, un cliente productor y un cliente consumidor, el cliente productor es el encargado de crear y enviar mensajes a la cola de eventos, y el cliente consumidor los recibe y muestra los mensajes y archivos que se envían.

Para poder realizar este laboratorio es necesario, primero descargar e instalar ServiceMix como se indica en las instrucciones de este. Y arrancamos el servicio para verificar que quede correctamente instalado. Luego de esto, vamos a crear un proyecto con Maven como el que se encuentra en el repositorio, en donde también se encontrara una carpeta llamada /deploy dentro del ServiceMix, con dos archivos blueprint.xml, en estos archivos se encontraran las reglas necesarias para que se envíen los archivos o mensajes y estos lleguen a una cola, y se redireccionen a otro cola de manera que puedan ser recibidos por el cliente consumidor.

A continuación, vemos el código junto con las configuraciones necesarias de la clase consumidor:

```
public class Consumer implements Runnable, ExceptionListener {
    public static void main(String[] args)
    {
        Consumer ex = new Consumer();
        ex.run();
    }
    public void run() {
        try {

            // Create a ConnectionFactory
            ActiveMQConnectionFactory connectionFactory = new ActiveMQConnectionFactory("tcp://localhost:61616?jms.useAsyncSend=true");

            // Create a Connection
            Connection connection = connectionFactory.createConnection("smx", "smx");
            connection.start();

            connection.setExceptionListener(this);

            // Create a Session
            Session session = connection.createSession(false, Session.AUTO_ACKNOWLEDGE);

            // Create the destination (Topic or Queue)
            Destination destination = session.createQueue("salidas");

            // Create a MessageConsumer from the Session to the Topic or Queue
            MessageConsumer consumer = session.createConsumer(destination);

            // Wait for a message
            Message message = consumer.receive(1000);

            if (message instanceof TextMessage) {
                TextMessage textMessage = (TextMessage) message;
                String text = textMessage.getText();
                System.out.println("Received: " + text);
            } else {
                System.out.println("Received: " + message);
            }

            consumer.close();
            session.close();
            connection.close();
        } catch (Exception e) {
            System.out.println("Caught: " + e);
            e.printStackTrace();
        }
    }

    public synchronized void onException(JMSException ex) {
        System.out.println("JMS Exception occurred. Shutting down client.");
    }
}
```

Y esta seria nuestro cliente productor:

```
public class Example {

    public static void main(String[] args)
    {
        Example ex = new Example();
        ex.run();
    }

    public void run(){
        CamelContext context = new DefaultCamelContext();
        //String brokerURL = args[0];
        ConnectionFactory connectionFactory = new ActiveMQConnectionFactory("tcp://localhost:61616?jms.useAsyncSend=true");
        try {

            // Create a Connection
            Connection connection = connectionFactory.createConnection("smx","smx");
            connection.start();

            // Create a Session
            Session session = connection.createSession(false, Session.AUTO_ACKNOWLEDGE);

            // Create the destination (Topic or Queue)
            Destination destination = session.createQueue("events");

            // Create a MessageProducer from the Session to the Topic or Queue
            MessageProducer producer = session.createProducer(destination);
            producer.setDeliveryMode(DeliveryMode.NON_PERSISTENT);

            // Create a messages
            String text = "Hello world! From: " + Thread.currentThread().getName() + " : " + this.hashCode();
            TextMessage message = session.createTextMessage(text);

            // Tell the producer to send the message
            System.out.println("Sent message: "+ message.hashCode() + " : " + Thread.currentThread().getName());
            producer.send(message);

            // Clean up
            session.close();
            connection.close();
        }

        catch (Exception e) {
            System.out.println("Caught: " + e);
            e.printStackTrace();
        }
    }
}
```

El usuario y la contraseña, los dos son “smx”, se deben tener en cuenta al configurar el cliente productor, y también la URL debe cambiarse ya que así se permite el acceso y se conecta al ServiceMix.

Los archivos dentro de la carpeta /deploy, son activeMQ.xml y textfile.xml, a continuación, veremos las configuraciones que hay en ambos.

La siguiente imagen corresponde al archivo activeMqQ.xml, en este archivo estamos indicando desde cuál cola se reciben los mensajes y hacia cuál cola se dirigen.

```
<?xml version="1.0" encoding="UTF-8"?>
<blueprint
  xmlns="http://www.osgi.org/xmlns/blueprint/v1.0.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    http://www.osgi.org/xmlns/blueprint/v1.0.0
    http://www.osgi.org/xmlns/blueprint/v1.0.0/blueprint.xsd">

  <camelContext xmlns="http://camel.apache.org/schema/blueprint">
    <route>
      <from uri="activemq://events"/>
      <to uri="log://salidas"/>
    </route>
  </camelContext>
</blueprint>
```

---

Y en el archivo textFile.xml se encuentra la siguiente configuración:

```
<?xml version="1.0" encoding="UTF-8"?>
<blueprint
  xmlns="http://www.osgi.org/xmlns/blueprint/v1.0.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    http://www.osgi.org/xmlns/blueprint/v1.0.0
    http://www.osgi.org/xmlns/blueprint/v1.0.0/blueprint.xsd">

  <camelContext xmlns="http://camel.apache.org/schema/blueprint">
    <route>
      <from uri="file:camel/input"/>
      <log message="Moving ${file:name} to the output directory"/>
      <to uri="file:camel/output"/>
    </route>
  </camelContext>

</blueprint>
```

---

Si queremos confirmar que el proceso de enviar y recibir un mensaje se este realizando correctamente, lo podemos hacer en la parte de los logs, para esto escribimos el comando **log:display**, y nos deberá aparecer algo como esto:

```
File Edit View Search Terminal Help
2.16.5 | Runtime endpoint registry is in extended mode gathering usage statistics of all incoming and outgoing endpoints (c
ache limit: 1000)
2019-11-08 11:21:03,333 | INFO | mix-7.0.1/deploy | ActiveMQComponentResolver | 132 - org.apache.servicemix.activem
q.camel - 7.0.1 | Creating an instance of the ActiveMQComponent (activemq:)
2019-11-08 11:21:03,344 | INFO | mix-7.0.1/deploy | BlueprintCamelContext | 43 - org.apache.camel.camel-core -
2.16.5 | AllowUseOriginalMessage is enabled. If access to the original message is not needed, then its recommended to turn
this option off as it may improve performance.
2019-11-08 11:21:03,344 | INFO | mix-7.0.1/deploy | BlueprintCamelContext | 43 - org.apache.camel.camel-core -
2.16.5 | StreamCaching is not in use. If using streams then its recommended to enable stream caching. See more details at h
ttp://camel.apache.org/stream-caching.html
2019-11-08 11:21:03,543 | INFO | mix-7.0.1/deploy | TransportConnector | 25 - org.apache.activemq.activemq-o
sgt - 5.14.5 | Connector vm://amq-broker started
2019-11-08 11:21:03,575 | INFO | mix-7.0.1/deploy | BlueprintCamelContext | 43 - org.apache.camel.camel-core -
2.16.5 | Route: route2 started and consuming from: Endpoint[activemq://events]
2019-11-08 11:21:03,576 | INFO | mix-7.0.1/deploy | BlueprintCamelContext | 43 - org.apache.camel.camel-core -
2.16.5 | Total 1 routes, of which 1 is started.
2019-11-08 11:21:03,576 | INFO | mix-7.0.1/deploy | BlueprintCamelContext | 43 - org.apache.camel.camel-core -
2.16.5 | Apache Camel 2.16.5 (CamelContext: camel-2) started in 0.251 seconds
2019-11-08 11:21:03,582 | INFO | mix-7.0.1/deploy | fileinstall | 4 - org.apache.felix.fileinstall -
3.5.8 | Started bundle: blueprint:file:/home/estudiante/Downloads/apache-servicemix-7.0.1/deploy/blueprint2.xml
2019-11-08 11:47:25,363 | INFO | Consumer[events] | events | 43 - org.apache.camel.camel-core -
2.16.5 | Exchange[ExchangePattern: InOnly, BodyType: String, Body: Hello world! From: main : 1663166483]
```