What is Holodeck Gateway?

Holodeck Gateway is a component inside Holodeck (deployed as an Axis2 Service called "msh") that is responsible for dispatching received messages to consumers. When a message arrives, it is generally processed by the different modules inside Holodeck (starting with the Security module which will verify signatures and decrypt whatever is encrypted, then the Reliability Module which processes reliability headers, then the ebMS3 Module, and only when the message has passed all the modules, then the Gateway component of Holodeck would handle the message by figuring out who might be the consumer of the message.

For example, in the context of webservices, each webservice has its own address, and therefore the webservice stack can easily dispatch the message to the right webservice which is the final consumer of the message. In the case of B2B messaging with Holodeck, you can have 20 or more consumers all using the same addresse (which happens to be the address of the Holodeck system itself). Each of these consumers may be interested to consume only certain type of messages. For example, you may have a consumer interested in consuming all messages that have a certain value in their SOAP header. The goal of Holodeck Gateway component is to figure out which consumer should the message be given to (webservices can be among these consumers).

How to create a consumer?

Holodeck comes shipped with some built-in consumers that would fit the need to many users. However, if you don't find the type of consumer you want, and you would like to write your own consumer, this can be done by simply implementing the Java interface *org.holodeck.ebms3.consumers.EbConsumer*. This interface contains three methods:

- (a) **push** method: you implement this method if you want to consume User Messages (that is Messages with payloas in them)
- (b) **pull** method: you implement this method if you want to consume a PullRquest (this is if you want to control how messages are pulled from your side. A message-to-be-pulled might not be present in the database and may be constructed dynamically by calling other entities such as an ESB, a workflow application or other applications).
- (c) **setParameters** method: this method provides you with parameter configuration. If you don't need any parameters to configure your task, then you don't need this method. For example, a consumer that invokes a webservice would need to know the address of the webservice (such an address would be a parameter that will be passed to you in the setParameters() method).

How to register a consumer with Holodeck Gateway?

You declare a consumer by adding an entry **<consumption>...</consumption>** in the file "**holodeck-1.0/config/gateway.xml**". Each consumer may have a filter that specifies the criteria that the received message should meet in order to be consumed.

The Gateway component, when trying to figure out who is the consumer the received message should be given to, tries to evaluate the filters declared in the file "**holodeck-1.0/config/gateway.xml**" in the order they are declared, and whoever consumer meets the criteria (his filter is satisfied), then the received message will be given to that consumer. If a consumer does not have a filter, that mean such a consumer is interested in consuming "All" received messages.

If you open the file "**holodeck-1.0/config/gateway.xml**", you will find there the first consumer without any filter declared (meaning that such a consumer will consume all messages). That consumer is only saving the attachments data under the folder "holodeck/store/receive/" folder. If you want to do something else with the received message (instead of saving its attachments to a folder, you may want to do something else), then you can create your own "Consumer" class and then declare your consumer in the "gateway.xml" file.

Let's examine the gateway configuration file where consumers are registered. The following shows the contents of this configuration file **holodeck/config/gateway.xml**

```
        //------------------------------------------------------------------
```

```
            a parameter to indicate to the consumer application "ConsumerTest"
            where to save the attachment in the request. The value could be
            either an absolute path or relative to the "Received_Messages_Folder"
            (directory where the MSH saves attachments of the received messages)
        //---------------------------------------------------------------------------
        Messages







        HOLODECK
        tcp://localhost:61616





      //-------------------------------------------------------------------------------
      // The filter element is optional and all its children are optional. The child
      // of the filter are grouped by the logical "AND" connector. This means that i
      // for a filter to be evaluated to true, all the values of the existing childr
      // of the filter must match those values in the SOAP header of the received me
      //-------------------------------------------------------------------------------

        shipment
        ......

          ...
          ...

        ...

          ...
          ...

        ...
        ...
        ...
        ...
        ...
        ...

          ...
          ...
```

Each consumer object is declared by an xml entry **<consumption>**. The definition of a consumer contains an optional element called **<filter>** which basically tells the "Gateway" service what kind of messages it is interested in consuming. If the element **<filter>** is absent, that means the consumer is interested in all received messages and therefore it will consume all messages. All the children elements of **<filter>** are optional. For example, if a consumer wants to consume only messages that have SOAP header **<eb:Service>** with a value equals to "Shipment", the filter would have only one one element and it would look like the following:

```
  <filter>
    <service>Shipment</service>
  </filter>
```

If another consumer wants to consume messages in which the value of the SOAP header "**eb:Messaging/eb:UserMessage/eb:CollaborationInfo/eb:Action**" is "NewPurchaseOrder" AND the value of the SOAP header **eb:Messaging/eb:UserMessage/eb:PartyInfo/eb:From /eb:PartyId** is "http//:oracle.com", then the filter would defined as the following:

```
<filter>
  <fromParties>
    <party type="">http//:oracle.com</party>
  </fromParties>
  <action>NewPurchaseOrder</service>
</filter>
```

Remember that consumers (and their filters inside of them) are evaluated one after the other in the order they are declared in the file "**gateway.xml**". The first consumer to evaluate to true (its filter is evaluated to true, that is matches the values in the SOAP header), then that consumer will be the winner and the message will be given to it. That means the evaluation of the consumers after it will not happen. As soon as a consumer is evaluated to true, the other consumers after it are not even evaluated. This is because the message will only be given to one consumer only and to many consumers at the same time which is not possible because the data cannot be synchronized for multiple access (plus the data is usually streamed and given as is to consumer, so it is not possible to be given to multiple consumers at the same time. If multiple consumers need to consume the same message, it is better to send the message to a JMS topic and let the subscribers be notified by the JMS provider).