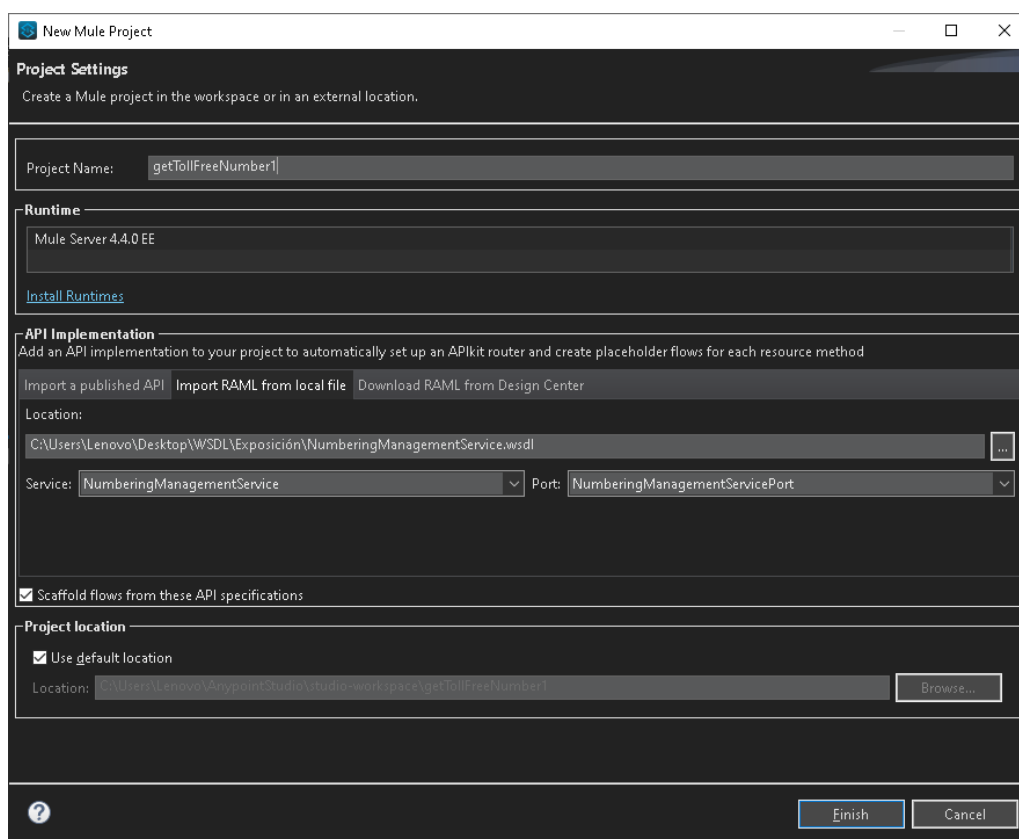


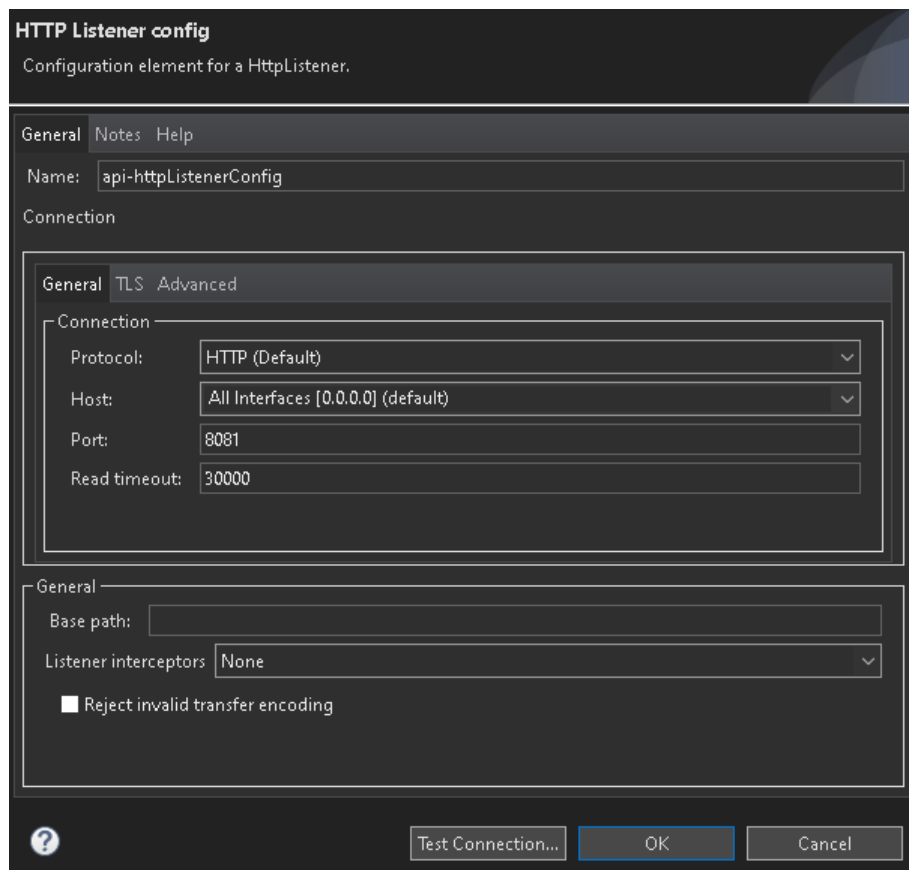
Manual de migración de OSB 12 a Mulesoft

Creación del proyecto y configuración.

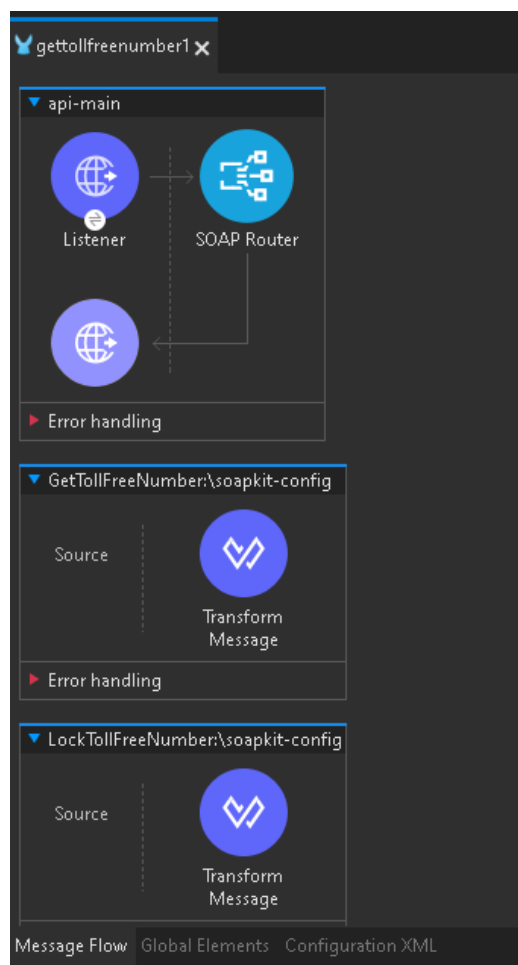
Creamos un mule Project, le ponemos un nombre y en el campo API Implementation seleccionamos el campo Import RAML from local file y buscamos el WSDL que vamos a usar. Los campos Service y Port se rellenan automáticamente al momento de seleccionar el WSDL.



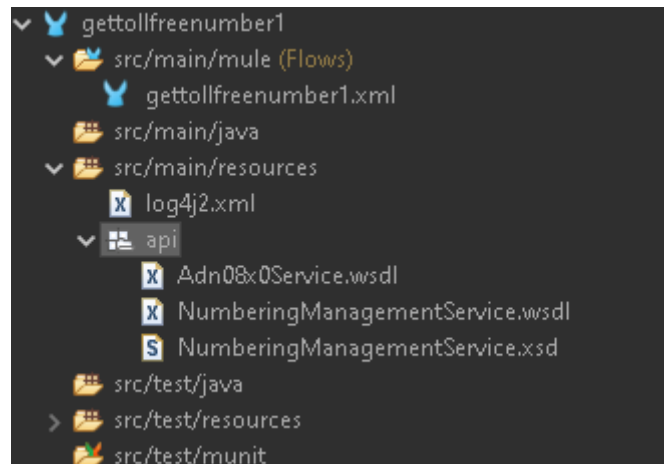
Una vez que se cargue el WSDL se generará una aplicación con un Listener, un componente Soap Router que identificará la operación solicitada en el request cada vez que se llama a la aplicación y se creará un subflujo para cada operación automáticamente. Al Listener hay que configurarle un host, un puerto y un path desde donde se va a poder llamar a este flujo. Para realizar esto debemos hacer click en el componente Listener, en la configuración que apareciera debajo, nos aparece un signo + de color verde en el apartado de Basic Settings, debemos hacer click en el y configurarlo de la siguiente manera.



Finalmente nos quedará creado el siguiente flujo:

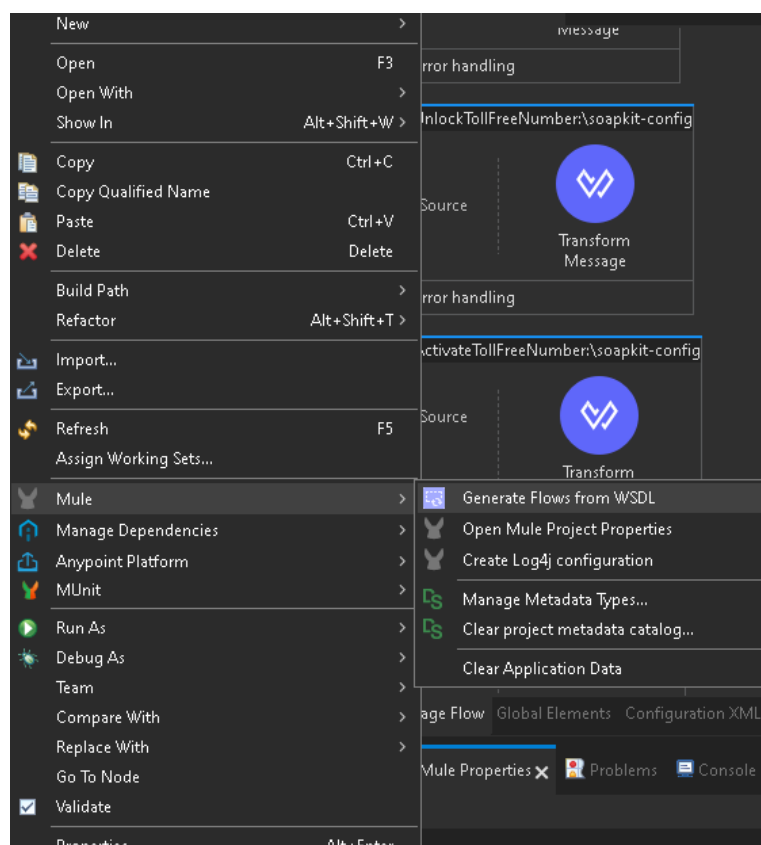


Si se necesitan utilizar más WSDL , arrastrarlos a la carpeta api dentro de src/main/resources en el apartado Package Explorer.

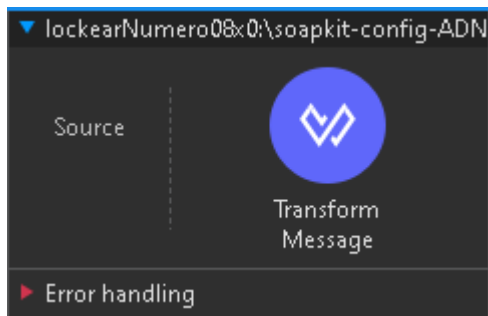


Si se necesitan crear flujos para otro WSDL se tiene que seguir los siguientes pasos:

- Seleccionar el WSDL a utilizar.
- Hacer click derecho , seleccionar la opción Mule y seleccionar Generate Flows from WSDL.



Esto generará otra aplicación con un Listener que debe ser distinto. Las configuraciones HTTP listener config y Apikit for SOAP Configuration deben también tener nombres distintos en cada flujo que tenga un WSDL distinto. y hay que verificar que en cada el final del nombre de los subflujos de cada operación coincida con el nombre del Apikit for SOAP Configuration.



Configuración de Numbering_Flow

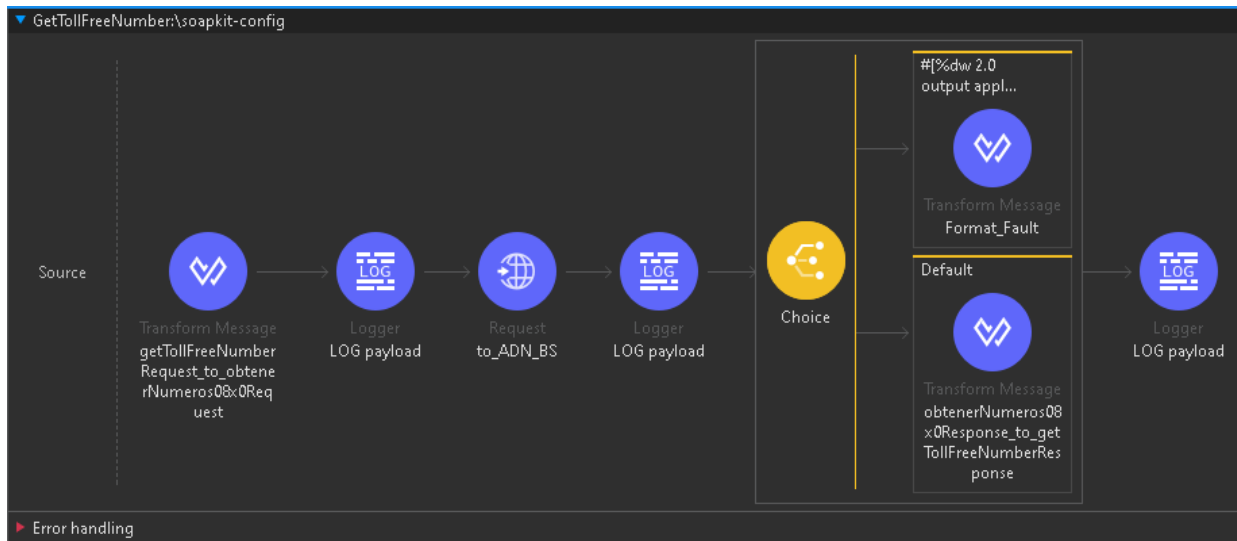
Global Configuration Elements			
Type	Name	Description	Create
HTTP Listener config (Configuration)	api-httpListenerConfig		Edit
APIKit for SOAP Configuration (Configuration)	soapkit-config		Delete
HTTP Request configuration (Configuration)	HTTP_Request_configura		

Configuración de ADN_BS_Flow

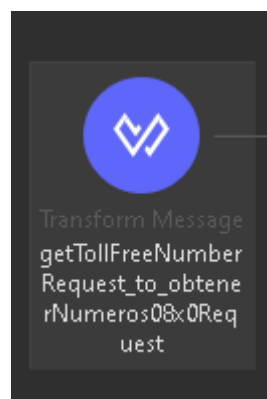
Global Configuration Elements			
Type	Name	Descrip	Create
HTTP Listener config (Configuration)	api-httpListenerConfig-ADN		Edit
APIKit for SOAP Configuration (Configuration)	soapkit-config-ADN		Delete
HTTP Request configuration (Configuration)	HTTP_Request_configuration1		

Desarrollo

Una vez terminada la configuración, nos dirigimos al subflujo de la operación *GetTollFreeNumber:\soapkit-config* y comenzamos con la primera transformación.



Utilizamos el componente Transform Message.

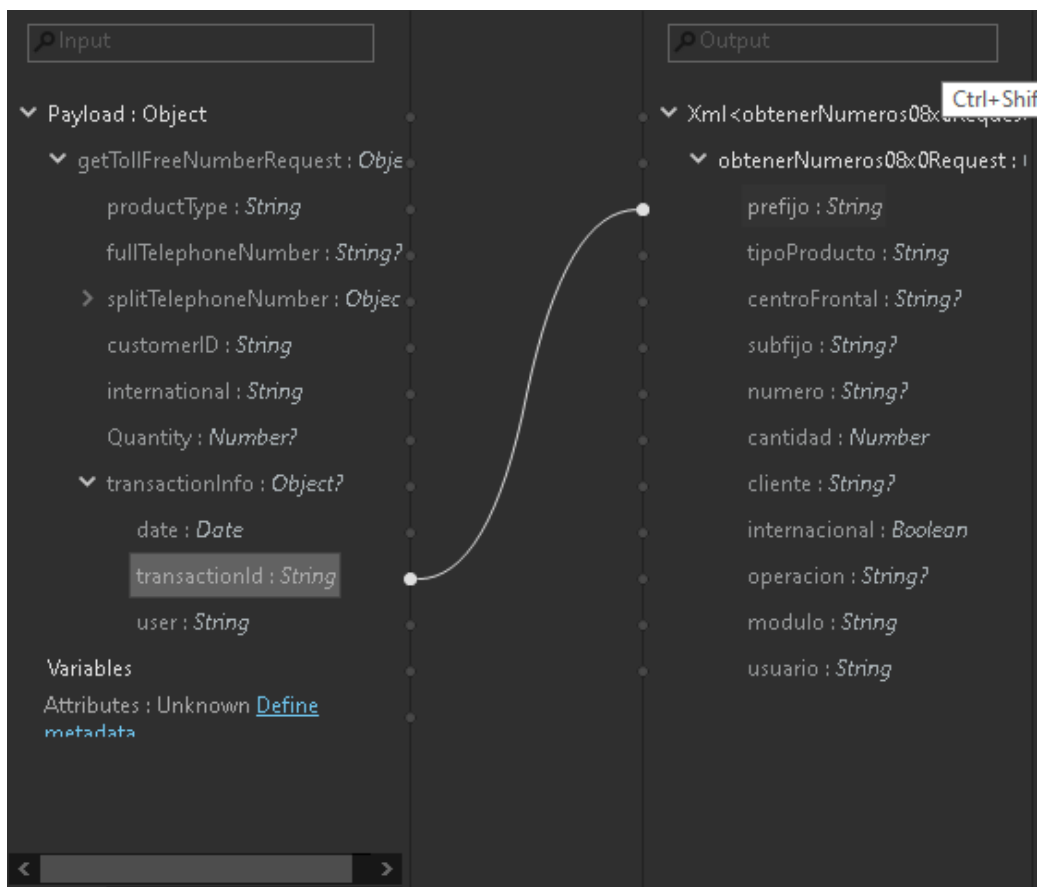


Transformamos el *getTollFreeNumberRequest* a *obtenerNumeros08X0Request* realizando el mapeo del dato *transactionId* con prefijo.

Además se le agrega manualmente en el código los tag *Envelope*, *Header* y *Body* con su correspondiente namespace. Esto se hace porque es necesario para que el request pueda viajar a la aplicación *ADN_BS_Flow*.

```
Output Payload
1 <?xml version="1.0"
2 output application/xml
3 ns soapenv http://schemas.xmlsoap.org/soap/envelope/
4 ns2 http://www.movistar.com.ar/ws/schema/ACN/types/v0100
5 ns ns01 http://www.movistar.com.ar/ws/schema/amdocs/NumberingManagementService
6 ---
7
8 {
9   soapenv#Envelope: {
10     soapenv#Header: {},
11     soapenv#Body: {
12       ns2#obtenerNumeros08x0Request: {
13         ns2#prefijo: payload.Body.ns01#getTollFreeNumberRequest.ns01#transactionInfo.ns01#transactionId
14       }
15     }
16   }
17 }
```


Sugerencia: si se necesita realizar el mapeo manual de los valores recibidos, y modificar el código resultante por alguna razón, aconsejamos hacer el mapeo en el siguiente apartado y luego realizar los cambios en el código ya que luego se perderá la imagen de los valores mapeados, pero el código seguirá funcionando.



Si en Input o Output no aparece la estructura deseada a la que queremos mapear los valores se deben seguir los siguientes pasos.

Hacer click derecho en el primer valor que aparece debajo del buscador, elegir Set Metadata, hacer click al signo + verde que aparece en la parte superior, darle un nombre al archivo a crear, elegir en Type el formato, debajo podemos seleccionar que tipo de estructura tendrá el archivo que vamos a importar para traernos la estructura, en nuestro caso Schema y luego en Root Element Name elegiremos la estructura deseada.

Select metadata type

Choose metadata type from tree and click Select

+ Add ✕ Delete



▼ User Defined

Fault : *Object*
getTollFreeNumberRequest_estructura : *Object*
getTollFreeNumberResponse_estructura : *Object*
obtenerNumero08x0Request_estructura : *Object*
obtenerNumero08x0Response_estructura : *Object*
payloadString : *String*

Nothing selected to display.

☐ Wrap element in a collection

Select

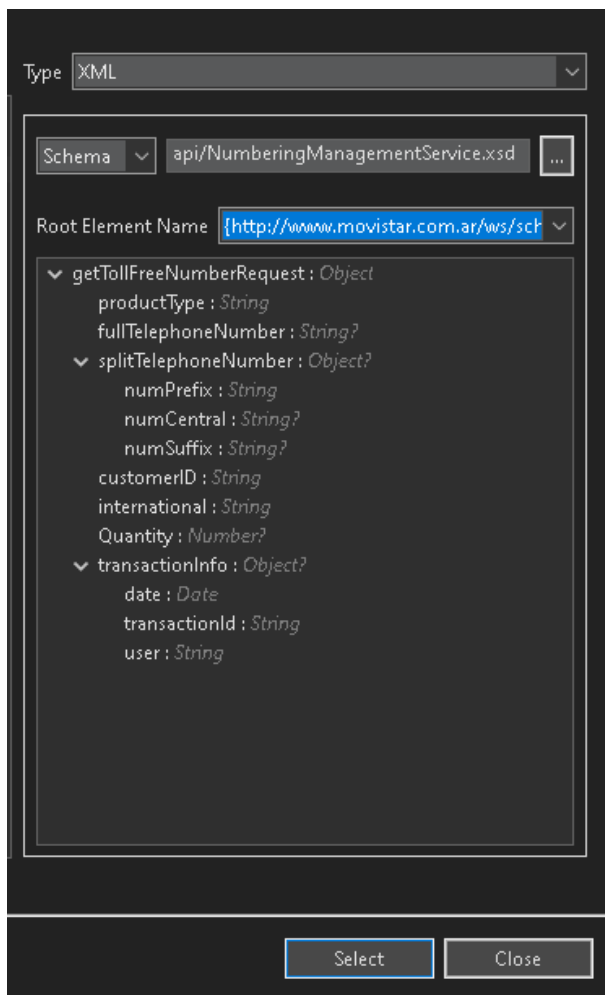
Close

Create new type

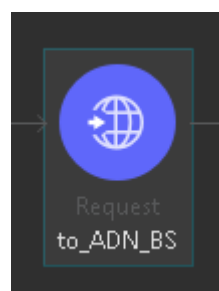
Type id:

Create type

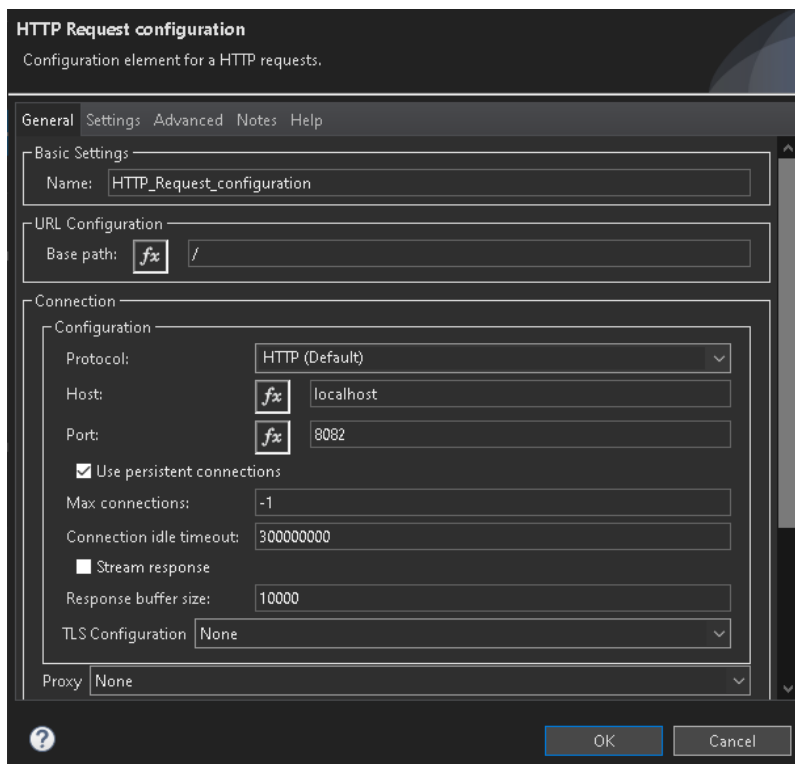
Cancel



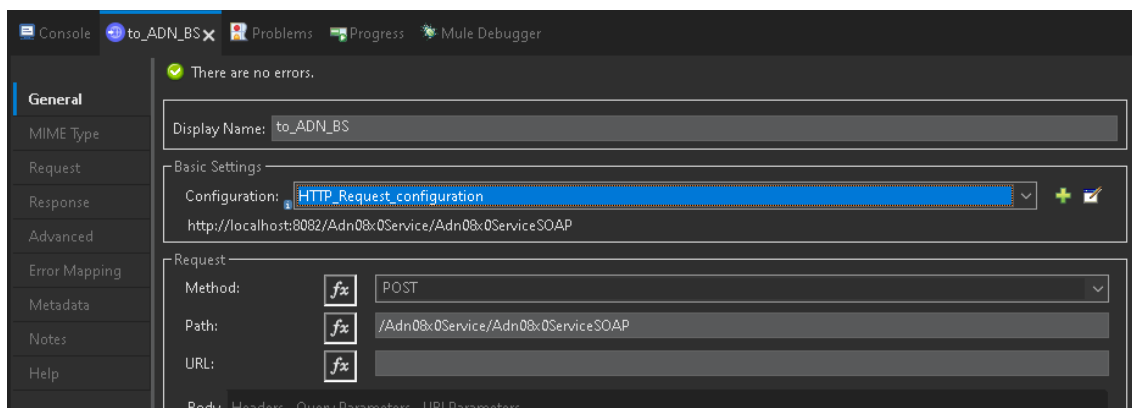
Para enviar el request transformado hacia el ADN_BS_Flow se utiliza el componente request. Con este componente se va a poder realizar la comunicación con la aplicación ADN_BS_Flow, este componente cumple una función similar al componente routing en OSB 12.



Para configurar este componente debemos dirigirnos a la sección Basic Settings y hacer click en el símbolo + verde para añadir una nueva configuración que debe ser la siguiente:



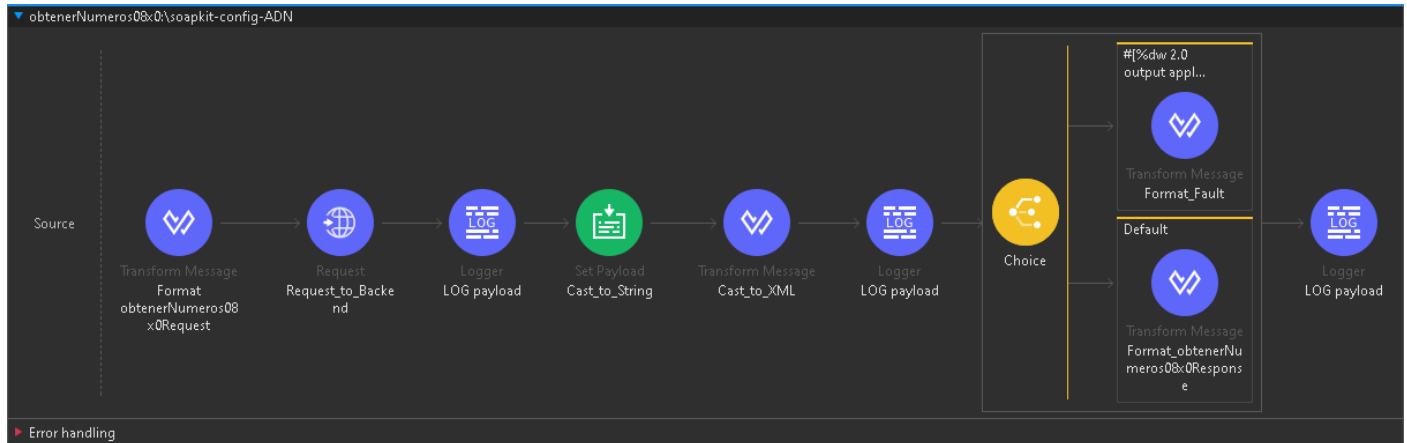
Luego de seleccionar OK volveremos a la ventana anterior y en la sección Request seleccionar como método POST y en PATH escribir la parte final de la URI sin el host y el puerto de la aplicación a la que queremos conectar.



Desarrollo del flujo ADN_BS_Flow

Dado que Mulesoft no tiene la opción de crear un business service como la tiene OSB 12, que esta recibe un request y se comunica automáticamente con el backend y retorna el response correspondiente. El comportamiento de este business lo tuvimos que desarrollar como un flujo. Esta aplicación ADN_BS_Flow se creó con el WSDL Adn08x0Service. Este mismo es el WSDL que se utiliza en el business en la interfaz getTollFreeNumber de OSB 12. Tal como la anterior aplicación esta cuenta con un Listener y un Soap router. El host debe ser distinto al utilizado en la aplicación anterior.

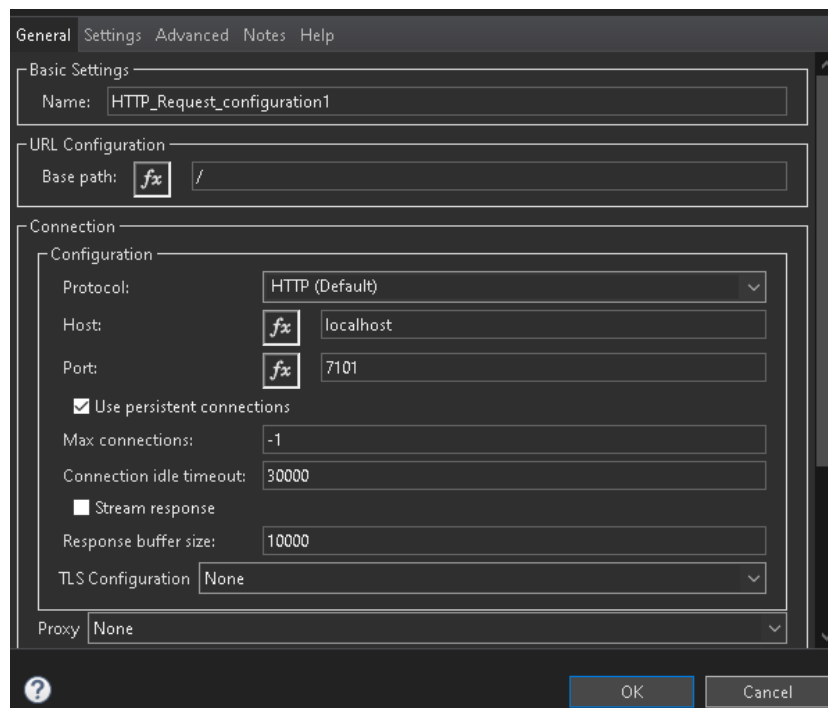
Al ingresar el request ya transformado en obtenerNumeros08X0Request este mismo se dirige al subflujo de la operación obtenerNumeros08X0. En el flujo de esta operación utilizamos estos componentes:



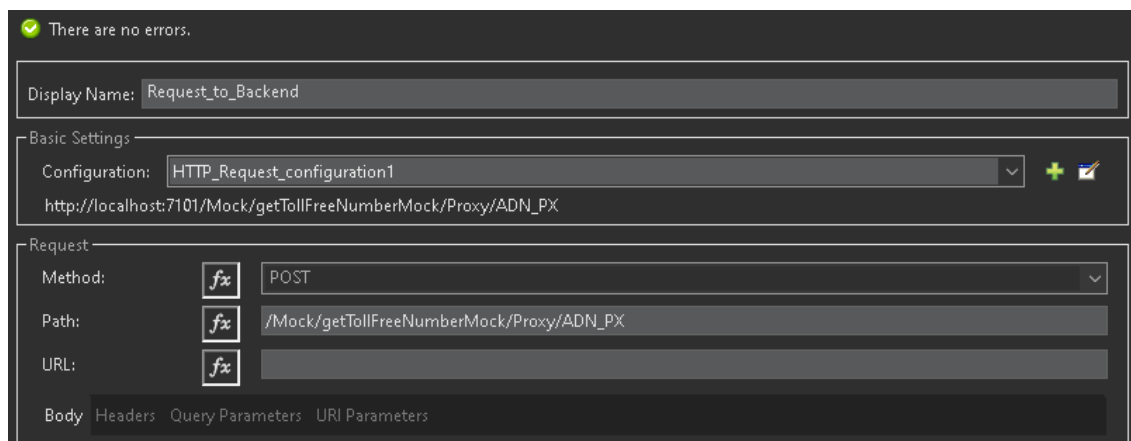
Primero transformamos el obtenerNumeros08X0 para agregarle un Envelope, un Header y un Body para que pueda viajar el request hacia el backend.

```
Output Payload
1 %dw 2.0
2 output application/xml
3 ns soapenv http://schemas.xmlsoap.org/soap/envelope/
4 ns ns2 http://www.movistar.com.ar/ws/schema/ACN/types/v0100
5 ns ns01 http://www.movistar.com.ar/ws/schema/amdocs/NumberingManagementService
6 ---
7 {
8   soapenv#Envelope: {
9     soapenv#Header: {},
10    soapenv#Body: {
11      ns2#obtenerNumeros08x0Request: {
12        ns2#prefijo: payload.body.ns2#obtenerNumeros08x0Request.ns2#prefijo default "",
13        ns2#tipoProducto: payload.body.ns2#obtenerNumeros08x0Request.ns2#tipoProducto default "",
14        ns2#centroFrontal: payload.body.ns2#obtenerNumeros08x0Request.ns2#centroFrontal,
15        ns2#subfijo: payload.body.ns2#obtenerNumeros08x0Request.ns2#subfijo,
16        ns2#numero: payload.body.ns2#obtenerNumeros08x0Request.ns2#numero,
17        ns2#cantidad: payload.body.ns2#obtenerNumeros08x0Request.ns2#cantidad default 0,
18        ns2#cliente: payload.body.ns2#obtenerNumeros08x0Request.ns2#cliente,
19        ns2#internacional: payload.body.ns2#obtenerNumeros08x0Request.ns2#internacional default true,
20        ns2#operacion: payload.body.ns2#obtenerNumeros08x0Request.ns2#operacion,
21        ns2#modulo: payload.body.ns2#obtenerNumeros08x0Request.ns2#modulo default "",
22        ns2#usuario: payload.body.ns2#obtenerNumeros08x0Request.ns2#usuario default ""
23      }
24    }
25  }
26 }
27 }
```

Volvemos a utilizar el componente Request para realizar el llamado al backend. En este caso tendría el mismo comportamiento que el componente Service Call Out de OSB 12. Con la siguiente configuración:

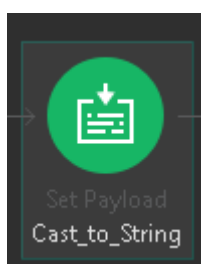


Esta configuración es para que se comunice con un mock desarrollado en OSB 12.

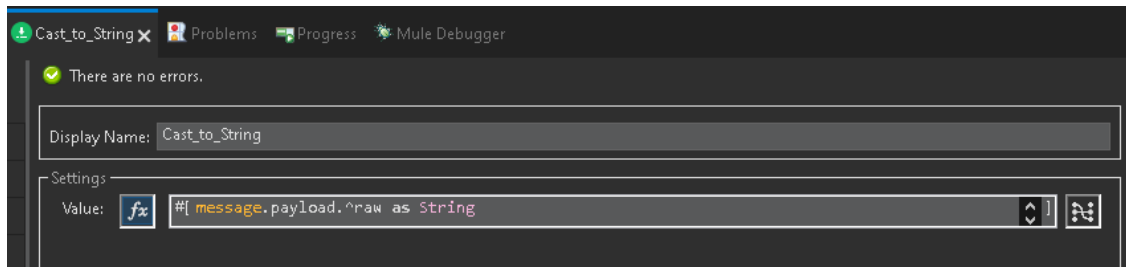


El response vendrá del backend con un formato de texto plano y este mismo no nos sirve para realizar el mapeo en la transformación a obtenerNumerosResponse08x0 ya que no reconoce los tags con sus valores en el formato de texto plano. Por eso tenemos que castearlo a String para luego poder castearlo a XML y poder transformarlo a obtenerNumerosResponse08x0.

Para ello utilizamos el componente Set Payload para castear el contenido del payload (que es el response que viene desde el backend) a String porque la función que lo convertirá finalmente a XML no reconoce el formato texto plano.



En el campo value se debe ingresar la siguiente linea de código:



Y luego deberemos castear de String a XML. Para ello utilizamos un Transform Message y agregamos el siguiente código:

```

Output Payload
1 %dw 2.0
2 output application/xml
3
4 ---
5 read(payload,"application/xml")
6

```

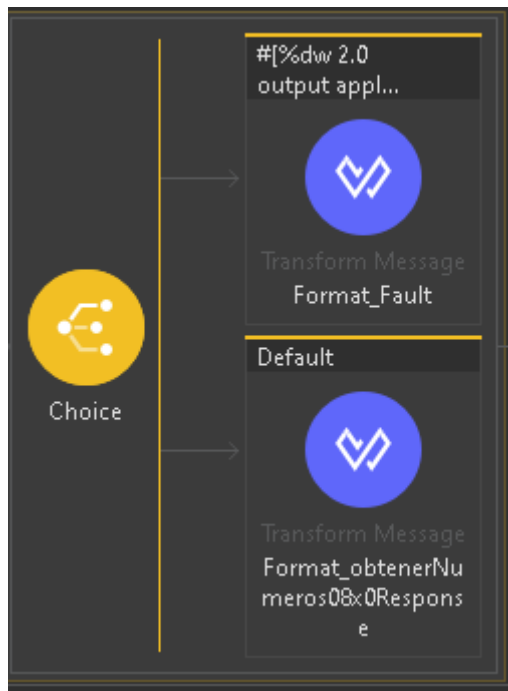
Para diferenciar si el response es una respuesta correcta o es un fault utilizamos el componente Choice con la siguiente condición:

```

1 %dw 2.0
2 output application/xml
3 ns ns0 http://www.movistar.com.ar/ws/schema/ACN/types/v0100
4 ---
5 ((keyOf(payload.Envelope.Body)).map($ as String)).contains "Fault"

```

Si el response contiene un nodo Fault el flujo continuará por la transformación Format Fault , si no contiene un Fault el flujo seguirá por el camino Default.



En caso de contener un Fault utilizaremos un Transform Message para darle formato y agregarle el envelope y el body porque el request que lo llamo espera que vuelva con ese formato.

```

1 %dw 2.0
2 output application/xml
3 ns soapenv http://schemas.xmlsoap.org/soap/envelope/
4 ns v01 http://www.movistar.com.ar/ws/schema/ACN/error/v0100
5 ---
6 {
7   soapenv#Fault: {
8     faultcode: payload.Envelope.Body.Fault.faultcode ,
9     faultstring : "USUARIO NO ENCONTRADO",
10    faultactor: payload.Envelope.Body.Fault.faultactor,
11
12    detail: {
13      v01#GeneralError: {
14        v01#mensaje: payload.Envelope.Body.Fault.detail.v01#GeneralError.v01#mensaje,
15        v01#descripcion: payload.Envelope.Body.Fault.detail.v01#GeneralError.v01#descripcion
16      }
17    }
18  }
19 }
  
```

Por otro lado si sigue el camino default usaremos también un Transform Message para darle el formato esperado por el request que lo llama.

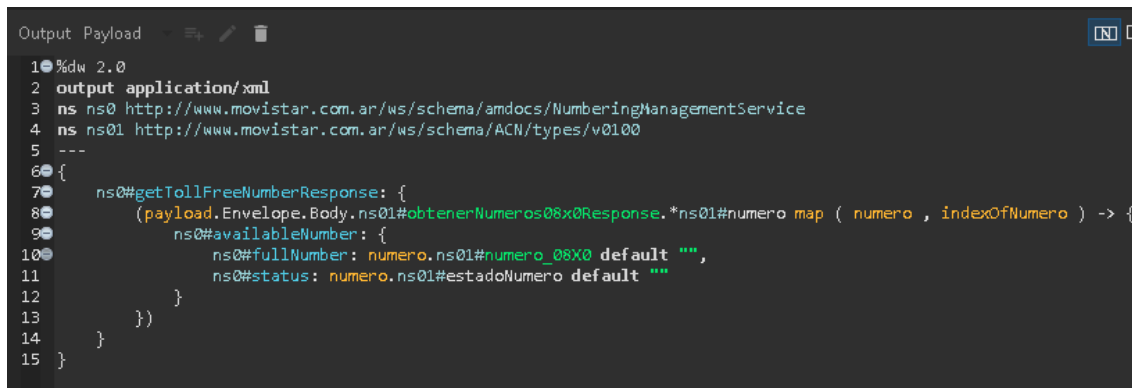
```

1 %dw 2.0
2 output application/xml
3 ns ns0 http://www.movistar.com.ar/ws/schema/ACN/types/v0100
4 ---
5 {
6   ns0#obtenerNumeros08x0Response: {
7     (payload.Envelope.Body.ns0#obtenerNumeros08x0Response.*ns0#numero map ( numero , indexOfNumero ) -> {
8       ns0#numero: {
9         ns0#numero_08X0: numero.ns0#numero_08X0,
10        ns0#estadoNumero: numero.ns0#estadoNumero
11      }
12    })
13  }
14 }
  
```

Luego de estas transformaciones el response retorna hacia el flujo que lo llamó en el flujo Numbering_Flow.

Volvemos a utilizar un choice para cuando retorne este flujo dependiendo del contenido de la respuesta vaya por una transformación o la otra.

Si contiene un fault se utiliza una transformación para darle el formato correcto y final, en cambio si no lo contiene convertimos el response de obtenerNumeros08X0Response a GetTollFreeNumberResponse y así concluye el flujo.



```
1 %dw 2.0
2 output application/xml
3 ns ns0 http://www.movistar.com.ar/ws/schema/amdocs/NumberingManagementService
4 ns ns01 http://www.movistar.com.ar/ws/schema/ACN/types/v0100
5 ---
6 {
7   ns0#getTollFreeNumberResponse: {
8     (payload.Envelope.Body.ns01#obtenerNumeros08x0Response.*ns01#numero map ( numero , indexOfNumero ) -> {
9       ns0#availableNumber: {
10        ns0#fullNumber: numero.ns01#numero_08X0 default "",
11        ns0#status: numero.ns01#estadoNumero default ""
12      }
13    })
14  }
15 }
```

Request para respuesta correcta:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:num="http://www.movistar.com.ar/ws/schema/amdocs/NumberingManagementService">
  <soapenv:Header/>
  <soapenv:Body>
    <num:getTollFreeNumberRequest>
      <num:productType>M</num:productType>
      <!--You have a CHOICE of the next 2 items at this level-->
      <num:fullTelephoneNumber>M</num:fullTelephoneNumber>
      <num:customerID>string</num:customerID>
      <num:international>N</num:international>
      <!--Optional:-->
      <num:Quantity>10</num:Quantity>
      <!--Optional:-->
```

```

    <num:transactionInfo>
      <num:date>2004-02-14</num:date>
      <num:transactionId>111</num:transactionId>
      <num:user>string</num:user>
    </num:transactionInfo>
  </num:getTollFreeNumberRequest>
</soapenv:Body>
</soapenv:Envelope>

```

Request para respuesta error con fault:

```

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:num="http://www.movistar.com.ar/ws/schema/amdocs/NumberingManagementService">
  <soapenv:Header/>
  <soapenv:Body>
    <num:getTollFreeNumberRequest>
      <num:productType>M</num:productType>
      <!--You have a CHOICE of the next 2 items at this level-->
      <num:fullTelephoneNumber>M</num:fullTelephoneNumber>
      <num:customerID>string</num:customerID>
      <num:international>N</num:international>
      <!--Optional:-->
      <num:Quantity>10</num:Quantity>
      <!--Optional:-->
      <num:transactionInfo>
        <num:date>2004-02-14</num:date>
        <num:transactionId>string</num:transactionId>
        <num:user>string</num:user>
      </num:transactionInfo>
    </num:getTollFreeNumberRequest>
  </soapenv:Body>
</soapenv:Envelope>

```

Response para respuesta correcta:


```

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"><soap:Body><ns0:getTollF
nagementService">
  <ns0:availableNumber>
    <ns0:fullNumber>08003216985</ns0:fullNumber>
    <ns0:status>D</ns0:status>
  </ns0:availableNumber>
  <ns0:availableNumber>
    <ns0:fullNumber>08003216986</ns0:fullNumber>
    <ns0:status>D</ns0:status>
  </ns0:availableNumber>
  <ns0:availableNumber>
    <ns0:fullNumber>08003216987</ns0:fullNumber>
    <ns0:status>D</ns0:status>
  </ns0:availableNumber>
  <ns0:availableNumber>
    <ns0:fullNumber>08003216988</ns0:fullNumber>
    <ns0:status>D</ns0:status>
  </ns0:availableNumber>
  <ns0:availableNumber>
    <ns0:fullNumber>08003216989</ns0:fullNumber>
    <ns0:status>D</ns0:status>
  </ns0:availableNumber>
</ns0:getTollFreeNumberResponse></soap:Body></soap:Envelope>

```

Response para respuesta error con fault:

```

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"><soap:Body><soapenv:Fault xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <faultcode>505</faultcode>
  <faultstring>USUARIO NO ENCONTRADO</faultstring>
  <faultactor>ERROR</faultactor>
  <detail>
    <v01:GeneralError xmlns:v01="http://www.movistar.com.ar/ws/schema/ACN/error/v0100">
      <v01:mensaje>ERROR</v01:mensaje>
      <v01:descripcion>ERROR</v01:descripcion>
    </v01:GeneralError>
  </detail>
</soapenv:Fault></soap:Body></soap:Envelope>

```