

OUM

TA.070-6 DISEÑO TÉCNICO DE EXTENSIONES E

INTERFACES



# ER12 GL Importación de Póliza

|  |  |
| --- | --- |
| Autor: | Forte Innovation |
| Fecha de creación: | Enero 15 2018 |
| Última actualización: | Febrero 09 2021 |
| Código de referencia OUM: | DS.140 Design Specification |
| Versión:  **Aprobadores:** | 1.1 |
| Ivan Muñoz |  |

Ángel Flores



Contenido

1. **Control de Documento ............................................................................................... 2**
   1. Bitácora de Cambios .................................................................................................... 2
   2. Revisores ...................................................................................................................... 2
2. **Resumen técnico ........................................................................................................ 1**
3. **Diagrama de la integración ....................................................................................... 3**
4. **Lógica de implementación ........................................................................................ 4**
   1. Calendarización y Ejecución..................................................................................... 4
   2. Implementación de servicio web JournalENT .............................................................. 5
   3. Implementación de servicio web JournalBiz ................................................................ 9
   4. Implementación de servicio web JournalBiz ................................................................ 9
   5. Implementación de servicio web FinancialsTec ......................................................... 13
   6. Diseño de Datos ......................................................................................................... 17
5. **Origen de Datos ........................................................................................................ 19**
6. **Lógica de Validación ................................................................................................ 20**
7. **Diseño SQL ............................................................................................................... 21**
   1. Sentencias SQL .......................................................................................................... 21
8. **Reglas de Negocio ................................................................................................... 26**
   1. Diseño del Servicio ..................................................................................................... 26
9. **Consideraciones de Rendimiento .......................................................................... 27**
   1. Estrategia de Reinicio................................................................................................. 27
   2. Seguridad ................................................................................................................... 27

9.3 Personalización .......................................................................................................... 27

1. **Catálogo de Errores ................................................................................................. 28**
2. **Consideraciones de Instalación ............................................................................. 29**
3. **URL de Acceso y seguridad .................................................................................... 30**
4. **Temas abiertos y cerrados ...................................................................................... 31**
   1. Temas Abiertos .......................................................................................................... 31
   2. Temas Cerrados ......................................................................................................... 31

## 1 Control de Documento

### 1.1 Bitácora de Cambios

3

|  |  |  |  |
| --- | --- | --- | --- |
| **Fecha** | **Autor** | **Versión** | **Referencia del cambio** |
| 02-feb-2018 | Efrain Arellanes | 1.0 | No Previous Document |
| 09-feb-2021 | Pedro Sanchez | 1.1 | Actualizacion del Documenrtacion. |
|  |  |  |  |
|  |  |  |  |

### 1.2 Revisores

|  |  |
| --- | --- |
| **Nombre** | **Posición** |
|  |  |
|  |  |
|  |  |
|  |  |



## 2 Resumen técnico

Este documento presenta el complemento para la integración que conforma la importación de pólizas contables hacia el ERP Cloud. Esta integración es referida por el documento funcional TA020-3\_ER12\_GL\_Importación de Pólizas Contables\_131217.docx.

El contenido del documento es el detalle para la representan la integración, de esta manera, su objetivo principal es identificar el flujo de la integración que está dividida en 3 capas.

* Enterprise (Ent)
* Business (Biz)
* Technical (Tec)

La capa **Ent** permite exponer aquellos servicios u operaciones de Oracle Service Oriented Architecture (SOA) mediante Oracle Service Bus (OSB), esto con la finalidad de tener una primera línea de seguridad sin exponer la arquitectura de los servidores internos de la empresa.

Por otro lado esta capa se encarga de administra las capas **Biz** utilizadas para esta integración, del mismo modo nos permite calendarizar en el servidor Enterprise Scheduler Service (EES) aquellas operaciones de SOA que se requieran ejecutar cada cierto tiempo.

La capa **Biz** tiene la finalidad de llevar acabo la lógica de negocio correspondiente a dicha integración, es decir, se encarga de orquestar aquellas capas técnicas y manejar aquellas reglas para el negocio.

La capa **Tec** nos permite ofrecer una línea de seguridad en cuestión a servicios de terceros mediante Oracle Service Bus (OSB); por otro lado esta capa nos permite conectar con aquellos orígenes de datos(Base de datos, Servicios Internos, EJB, etc.) permitiendo realizar diferentes operaciones para el guardado o manipulación de información.

Existen dos tipos de integraciones:

* Inbound: En este tipo de integraciones, las aplicaciones legadas llamarán a servicios web SOAP expuestos de lado de los servidores internos de la empresa, enviando la información hacia el ERP Cloud, es decir, son aquellas integraciones que nos permiten gestionar información referente a Clientes, Recibos, Transacciones, etc.
* **Outbound:** Estas integraciones nos permiten extraer información del ERP Cloud por medio de reportes a través de BI Publisher, posteriormente es enviada a diferentes sistemas legados según corresponda, algunas de las integraciones que aplican son: Timbrado de Facturas AR, validación de facturas AP, Validación de Complementos de Pago, etc.

Existen 4 fases importantes para este tipo de integraciones:

* 1. Consiste en la extracción de la información teniendo como fuente el ERP Cloud, donde se encuentran ciertas condiciones que permiten identificar qué información es apta para procesar.
  2. En esta fase consiste en el enriquecimiento del mensaje, es decir, implica realizar el complemento de información del lado de la capa de negocios en SOA para poder enviar al sistema externo (Capa Técnica), controlando cualquier tipo de error durante el flujo.
  3. En este punto se recupera la información devuelta por los servicios externos y se consolida la información de acuerdo a los resultados obtenidos.
  4. Por último, se envía una notificación con la información de todos los registros procesados, número de éxitos y errores con su detalle.

Estas fases están distribuidas en las 3 capas mencionadas anteriormente (Ent, Biz, Tec) las cuales se explicarán a detalle en los siguientes puntos.

Resumen técnico

## 3 Diagrama de la integración

Diagrama de la integración

## 4 Lógica de implementación

El desarrollo contiene dos flujos de ejecución TMS y Nomina de empleados los datos Usados por esta integración son colocados por TMS y Nomina de empleados en las tablas de SOA XXER\_GL\_INTERFACE, posteriormente tomados estos datos Creamos los registros tomados en el ERP.

### 4.1 Calendarización y Ejecución

### Se ejecuta diariamente para las Pólizas de TMS.

|  |  |  |
| --- | --- | --- |
| Componente | Propiedad | Valor |
| Trabajo | Nombre | Carga poliza viajado TMS |
| Nombre mostrado | Carga poliza viajado TMS |
| Paquete | /oracle/apps/ess/custom |
| Descripción | Carga poliza viajado TMS |
| Tipo de trabajo | SyncWebserviceJobType |
| Operación | SendFileJournal |
| Solicitud | <ns1:SendFileJournalRq xmlns:ns1="http://soa.estrellaroja.com.mx/JournalEnt"> <ns1:Source>TMS</ns1:Source> <ns1:Status>NEW</ns1:Status></ns1:SendFileJournalRq> |
| Planificación | Nombre | JourTMSSch |
| Nombre mostrado | JourTMSSch |
| Paquete | /oracle/apps/ess/custom |
| Descripción | Carga poliza viajado TMS |
| Frecuencia | Diario a 10:15:00 a. m. |
|  | Zona horaria | UTC-06:00 Mexico City |

### El Servicio es ejecutado Bajo demanda desde pantalla de ápex para la nomina de empleados.

URL: <https://129.150.96.227/ords/ersepdb1/f?p=126:LOGIN_DESKTOP:5453200106079>:::::

User / pass: ORA\_INTEGRACION  / A123gfhjs2

Peticion WS:

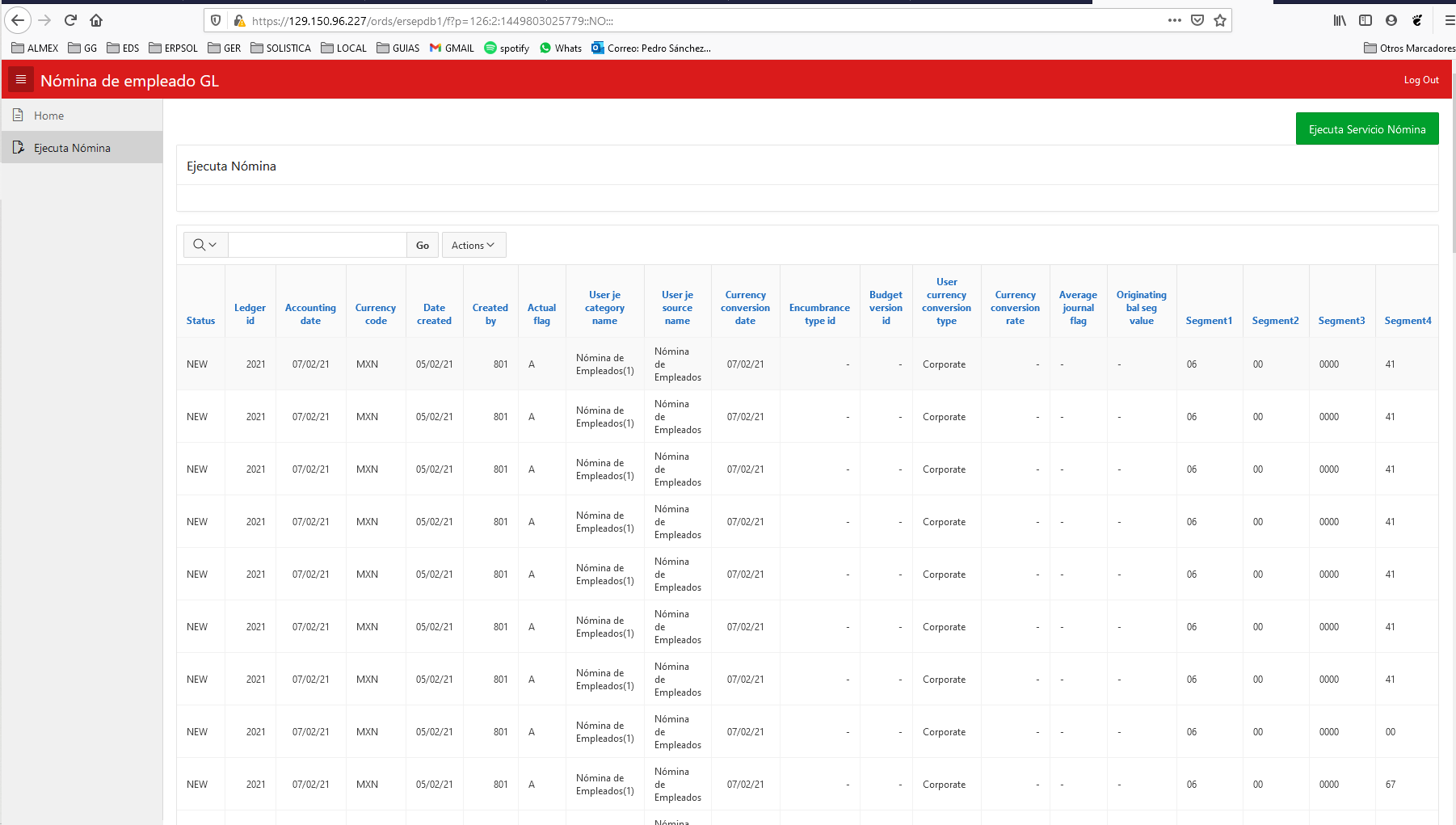
<ns1:SendFileJournalRq

xmlns:ns1="http://soa.estrellaroja.com.mx/JournalEnt">

<ns1:Source>Nómina de Empleados</ns1:Source>

<ns1:Status>NEW</ns1:Status>

</ns1:SendFileJournalRq>



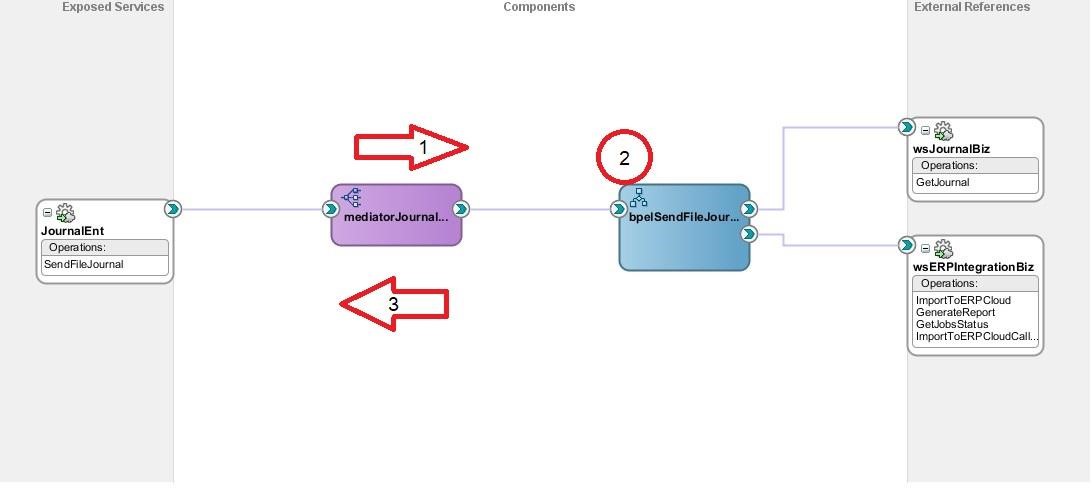
### 4.2 Implementación de servicio web JournalENT

El servicio web *JournalEnt* tiene dos flujos de ejecución carga de polizas de TMS y carga de polizas de Nomina empleados por medio de las operacion de:

1. SendFileJournal

Este servicio web utiliza una lógica de proceso de varios componentes de SOA que se encargan de enrutar las peticiones hacía los servicios de la capa de negocio. A continuación, se muestra su lógica de implementación:

1. Petición del cliente dependiendo de la operación SendFileJournal .
2. Invocación de BPEL bpelSendFileJournal.
3. Después de haber ejecutado una de las operaciones, el servicio responde la petición



La implementación se basa en la aplicación compuesta mostrada en la imagen anterior, se hace uso de los componentes:

|  |  |
| --- | --- |
| Componente | Icono del componente |
| Servicio Web |  |
| Mediador |  |
| BPEL |  |

El módulo principal se denomina mediador, este componente es el encargado de enrutar las peticiones desde el servicio de entrada hacia los demás componentes que se encargan de diferente funcionalidad y viceversa, es decir, las respuestas de cada componente son enrutadas y así entregadas a la petición que se generó.

### 4.2.1 Lista de Objetos

Los siguientes objetos fueron usados/creados para implementar la funcionalidad del servicio Web

|  |  |  |
| --- | --- | --- |
| **Elemento** | **Tipo** | **Descripción** |
| JournalEnt | Web  Service | Servicio Expuesto |
| mediatorJournalEnt | Mediator | Componente de SOA utilizado para integrar y enrutar los mensajes entre los servicios web |
| bpelSendFileJournal | Bpel | Componente de Orquestación |
| wsJournalBiz | Web  Service | Servicio externo que tiene la lógica de extracion de las polizas |
| wsERPIntegrationBiz | Web    Service | Servicio externo que Reliza la importación de los Datos al ERP |

### 4.2.2 Send File Journal

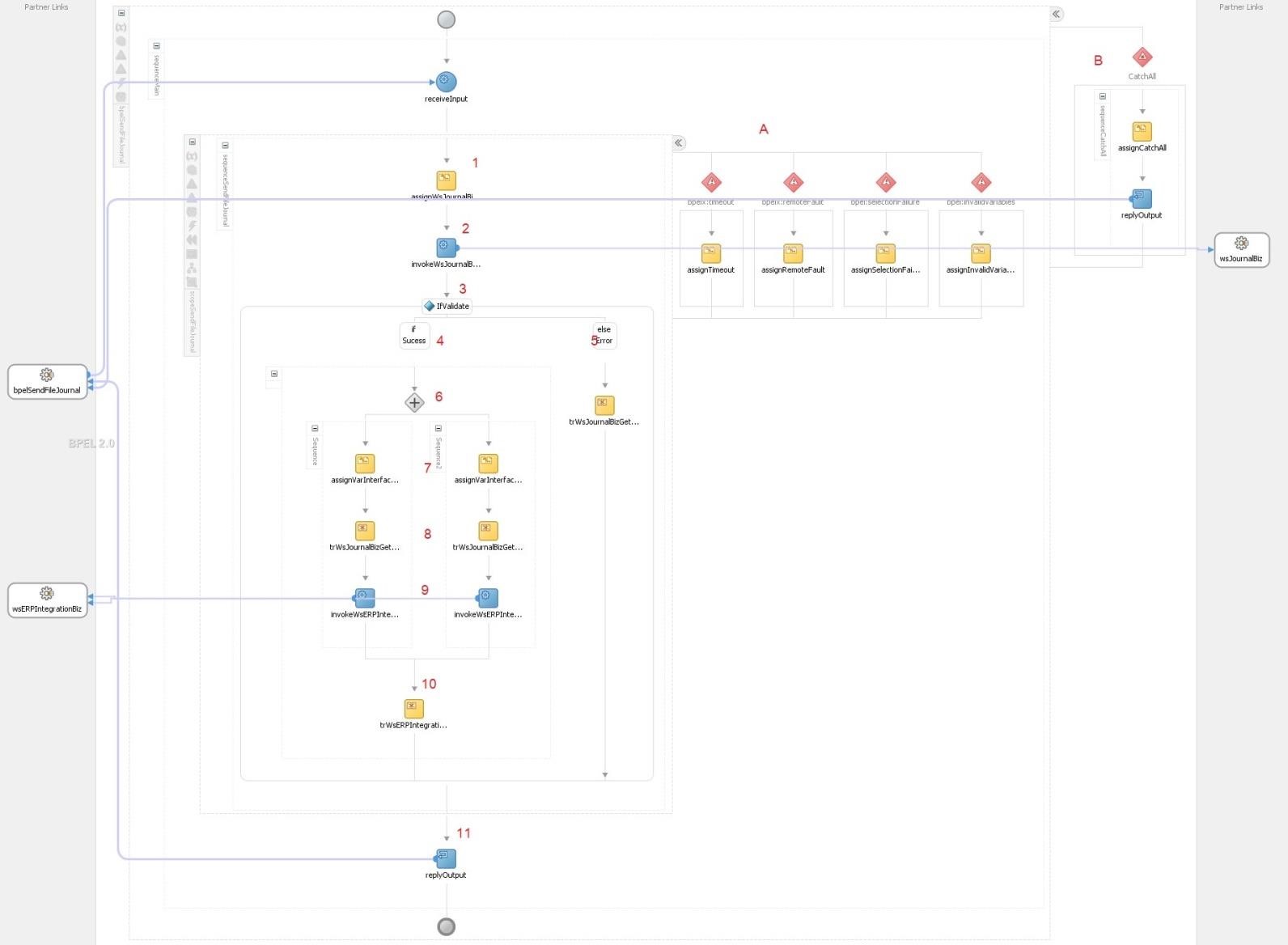
La operación *SendFileJournal* realiza una serie de pasos basados dentro de un bpel.

El proceso BPEL realiza las siguientes acciones:

1. Se procede a asignar los valores de entrada del bpel al servicio JournalBiz-SendFileJournal.
2. Se invoca al servicio JournalBiz-SendFileJournal.
3. Se valida si existen datos recuperados del servicio.
4. Sí existen datos, continua con el proceso de guardado de información.
5. Si no existen datos, se manda el error con datos no encontrados.
6. Se manda en paralelo para dos libros.
7. Se asigna la variable de la interface.
8. Se trasforma para la entrada del servicio ERPIntegrationBiz-ImportToERPCloud Trasforma aun formato CSV.
9. Se invoca al servicio ERPIntegrationBiz-ImportToERPCloud importa por medio de un template en CSV los datos al ERP.
10. Se da la respuesta del servicio

11. Salida y respuesta del bpel. 12. A y B Manejo de errores

13. Fin del Flujo.



### 4.2.3 Lista de objetos

La siguiente tabla contiene los objetos que fueron usados para la funcionalidad del bpel sendfileJournal.

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Nombre** | **Tipo** | **Descripción** |
| 1 | assignWsJournalBizGetJournalRq | assign | Asignación de valores. |
| 2 | invokeWsJournalBizGetJournal | invoke | Invocación de Servicio. |
| 3 | IfValidate | if | Validación de datos. |
| 4 | assignVarInterfaceL1 | assign | Asignación de valores. |
| 5 | trWsJournalBizGetJournalRsToBpelRs | transformation | Transformación de datos. |
| 6 | invokeWsERPIntegrationBizImportToERPCloud | invoke | Invocación de Servicio. |
| 7 | assignVarInterfaceL2 | assign | Asignación de valores. |
| 8 | trWsJournalBizGetJournalRsToBpelRs | transformation | Transformación de datos. |
| 9 | invokeWsERPIntegrationBizImportToERPCloud | invoke | Invocación de Servicio. |

**4.2.4 Pruebas**

Las pruebas se ejecutarán en el ambiente de desarrollo tomando la dirección del siguiente WSDL:

**http://ersoaprd-soa-0.snproduccin.vcner.oraclevcn.com:9073/soa-infra/services/default/JournalEnt/JournalEnt?WSDL**

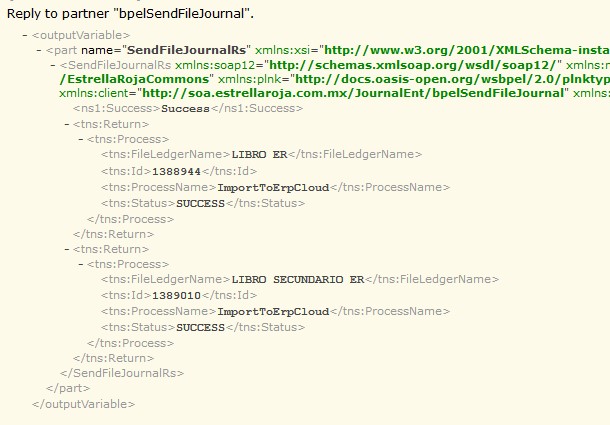
A continuación, se muestran los parámetros de entrada y la respuesta que se presentan en esta operación:

**TMS**

**Request:** No se necesitan parámetros de entrada para la ejecución de esta operación ya que es calendarizada, van adjuntos en la misma, para Nomina de empleados o viajado:



Response: El servicio retorna un id, con el cual se puede consultar la operación JobStatus para checar el estatus de los jobs del flujo.

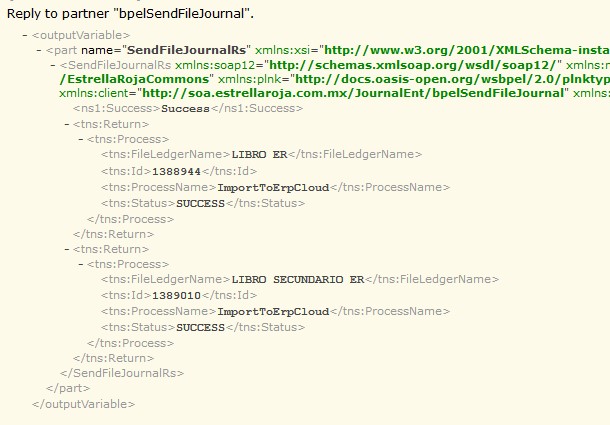


**Nómina de Empleados**

**Request:** para la ejecución de esta operación ya que es ejecutada desde Apex por medio de un botón que manda esta información para ejecutar este flujo:



Response: El servicio retorna un id, con el cual se puede consultar la operación JobStatus para checar el estatus de los jobs del flujo.

:

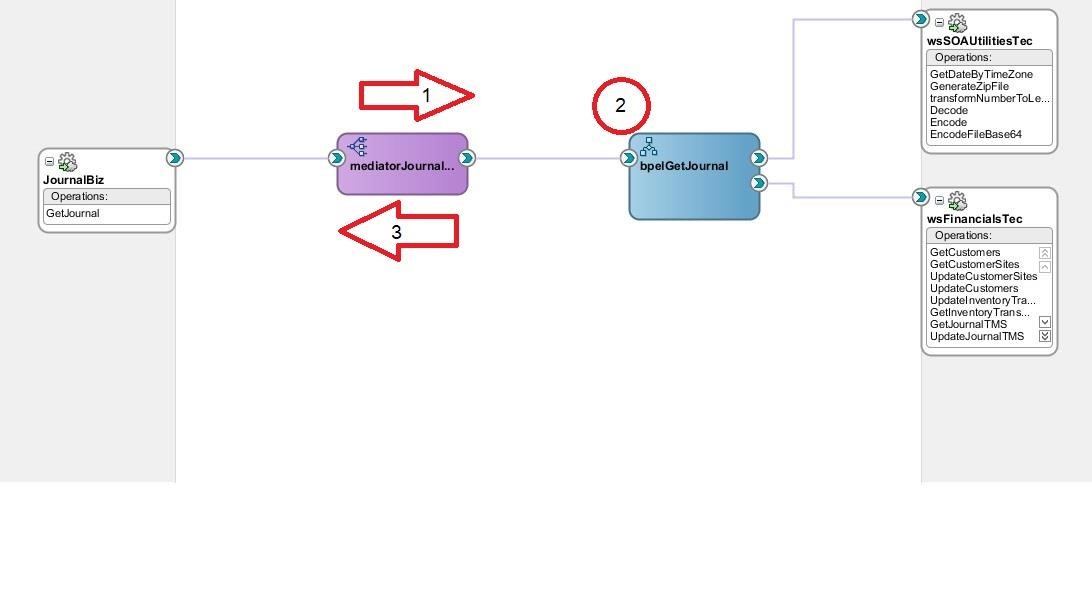
### 4.3 Implementación de servicio web JournalBiz

El servicio web *JournalBiz* tiene las operaciones de:

1. GetJournal

Este servicio web utiliza una lógica de proceso de varios componentes de SOA que se encargan de enrutar las peticiones hacía los servicios de la capa de negocio. A continuación, se muestra su lógica de implementación:

1. Petición del cliente dependiendo de la operación que desee ejecutar.
2. Invocación de BPEL bpelGetJournal.
3. Después de haber ejecutado una de las operaciones, el servicio responde la petición



La implementación se basa en la aplicación compuesta mostrada en la imagen anterior, se hace uso de los componentes:

|  |  |
| --- | --- |
| Componente | Icono del componente |
| Servicio Web |  |
| Mediador |  |
| BPEL |  |

El módulo principal se denomina mediador, este componente es el encargado de enrutar las peticiones desde el servicio de entrada hacia los demás componentes que se encargan de diferente funcionalidad y viceversa, es decir, las respuestas de cada componente son enrutadas y así entregadas a la petición que se generó.

### 4.3.1 Lista de Objetos

Los siguientes objetos fueron usados/creados para implementar la funcionalidad del servicio Web

|  |  |  |
| --- | --- | --- |
| **Elemento** | **Tipo** | **Descripción** |
| JournalBiz | Web  Service | Servicio Expuesto |
| bpelGetJournal | Bpel | Componente de Orquestación |
| mediatorJournalBiz | Mediator | Componente de SOA utilizado para integrar y enrutar los mensajes entre los servicios web |
| wsSOAUtilitiesTec | Web  Service | Servicio externo que Decodifica Archivo en este caso |
| wsFinancialsTec | Web  Service | Servicio externo consulta a BD SOA |

### 4.3.2 Send File Journal

La operación *GetJournal* realiza una serie de pasos basados dentro de un bpel.

El proceso BPEL realiza las siguientes acciones:

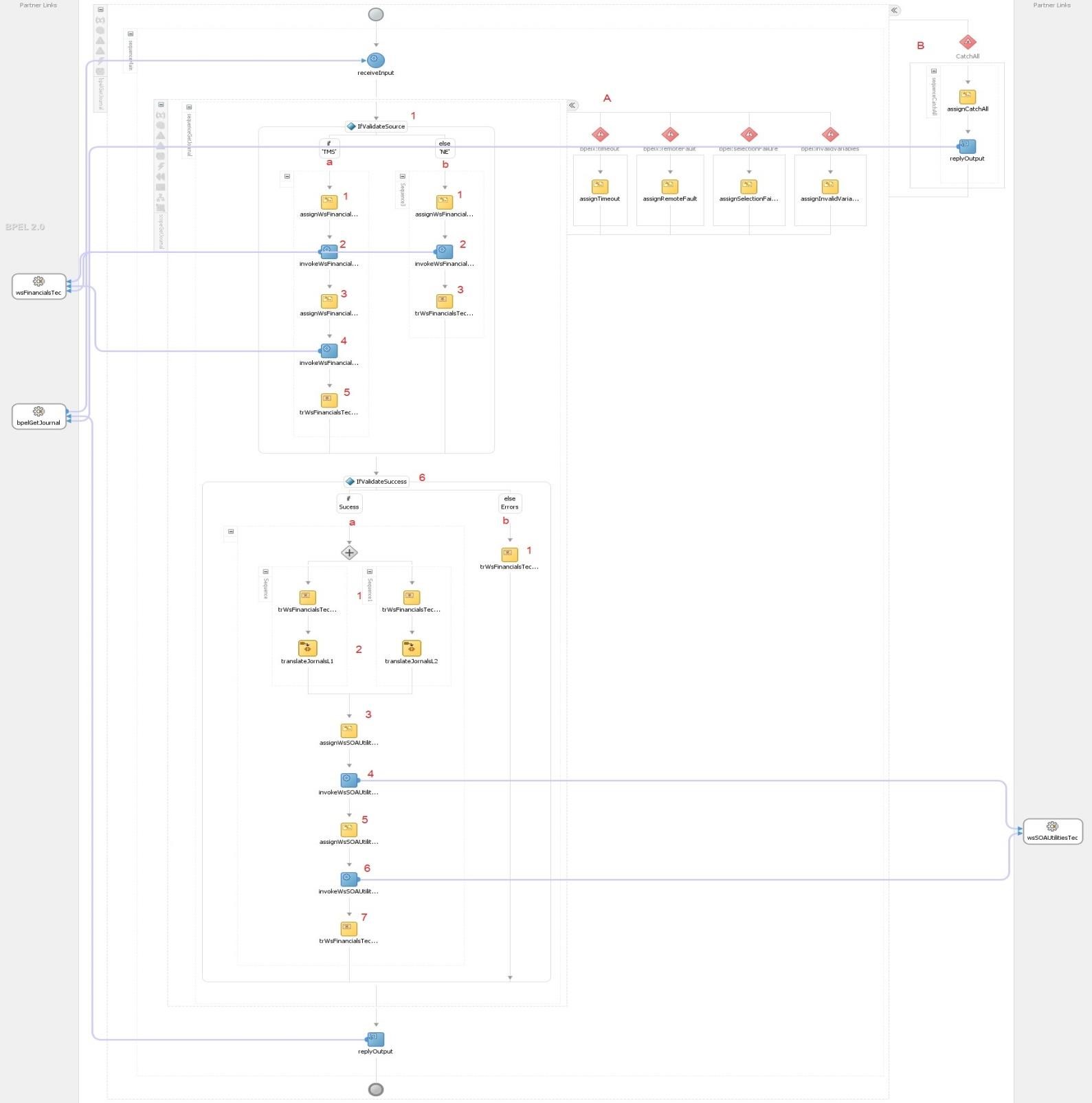
1. Valida source.

1. Viajado
   1. Se procede a asignar los valores de entrada del bpel al servicio FinancialsTecUpdateJournalTMS para actualizar bach\_id
   2. Se invoca al servicio FinancialsTec-UpdateJournalTMS
   3. Se procede a asignar los valores de entrada del bpel al servicio FinancialsTecGetJournalTMS.
   4. Se invoca al servicio FinancialsTec-GetJournalTMS 5. Se asigna la respuesta de salida.
2. Nómina
   1. Se procede a asignar los valores al servicio FinancialsTec-GetJournalNE
   2. Se invoca al servicio FinancialsTec-GetJournalNE.
   3. Se asigna la respuesta de salida.

1. Se valida éxito o error.
   1. Success
      1. Se procede a asignar los valores a las variables para entrada del servicio.
      2. Se hace un translate para armar el archivo en una cadena.
      3. Se procede a asignar los valores al servicio SOAUtilitiesTec-Encode
      4. Se invoca al servicio SOAUtilitiesTec-Encode
      5. Se procede a asignar los valores al servicio SOAUtilitiesTec-EncodeFileBase64.
      6. Se asigna la respuesta de salida.
   2. Error
      1. Se asigna salida con errores

1. A y B Manejo de errores

Fin del Flujo.



### 4.3.3 Lista de objetos

La siguiente tabla contiene los objetos que fueron usados para la funcionalidad del bpel sendfileJournal.

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Nombre** | **Tipo** | **Descripción** |
| 1 | assignWsJournalBizGetJournalRq | assign | Asignación de valores. |
| 2 | invokeWsJournalBizGetJournal | invoke | Invocación de Servicio. |
| 3 | IfValidate | if | Validación de datos. |
| 4 | assignVarInterfaceL1 | assign | Asignación de valores. |
| 5 | trWsJournalBizGetJournalRsToBpelRs | transformation | Transformación de datos. |
| 6 | invokeWsERPIntegrationBizImportToERPCloud | invoke | Invocación de Servicio. |
| 7 | assignVarInterfaceL2 | assign | Asignación de valores. |
| 8 | trWsJournalBizGetJournalRsToBpelRs | transformation | Transformación de datos. |
| 9 | invokeWsERPIntegrationBizImportToERPCloud | invoke | Invocación de Servicio. |

### 4.3.4 Pruebas

Las pruebas se ejecutarán en el ambiente de desarrollo tomando la dirección del siguiente WSDL:

##### http:// ersoaprd-soa-0.snproduccin.vcner.oraclevcn.com:9073/soa-infra/services/default/JournalBiz/JournalBiz?WSDL

A continuación, se muestran los parámetros de entrada y la respuesta que se presentan en esta operación:

**Request:** Status y source:



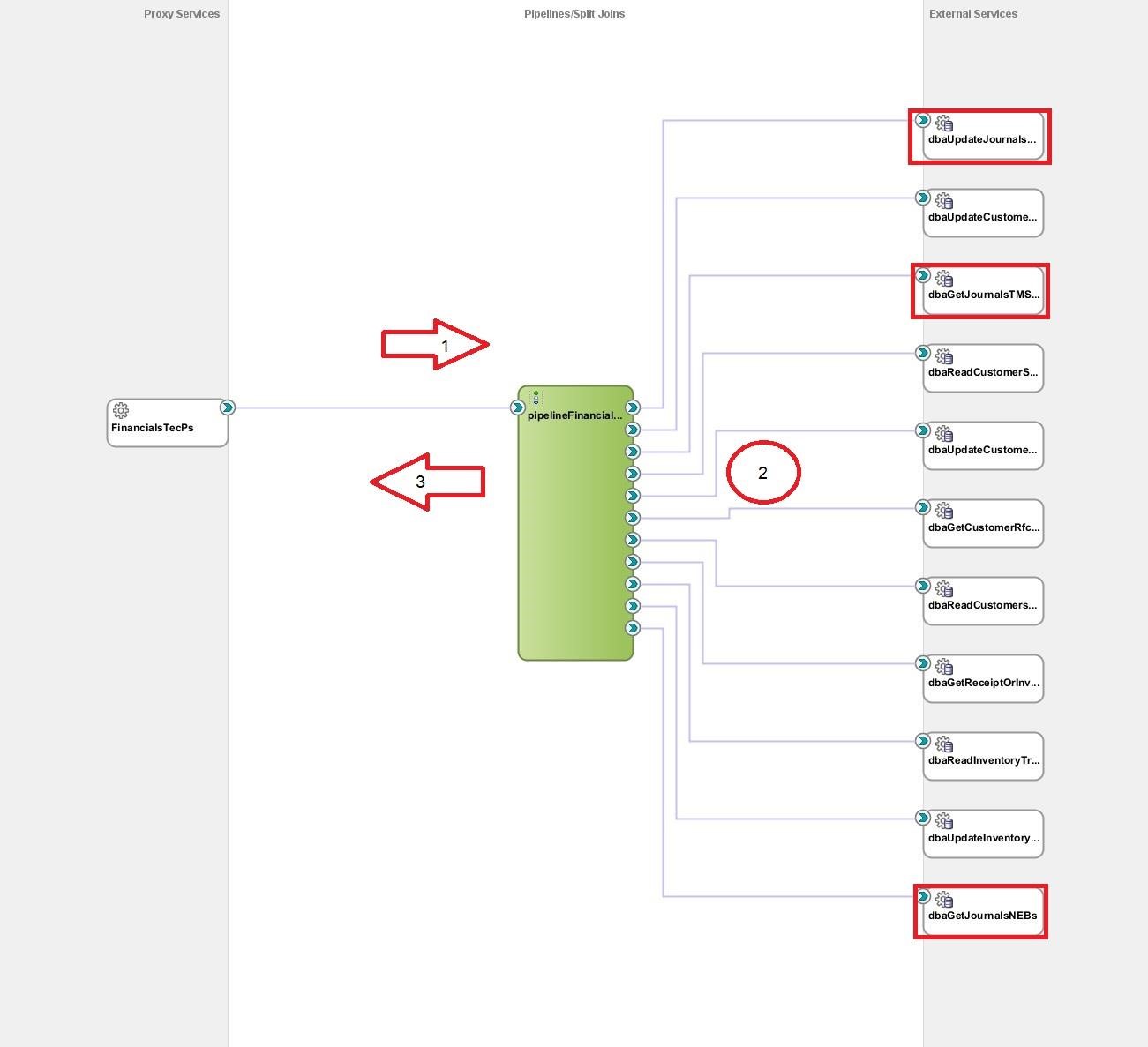
Response: El servicio retorna cadenas de base 64 con el contenido de la consulta de información de pólizas.



### 4.4 Implementación de servicio web FinancialsTec

Este servicio web utiliza una lógica de proceso de varios componentes de OSB que se encargan de enrutar las peticiones hacía los adaptadores de base. A continuación se muestra la lógica de implementación:

1. Petición del cliente dependiendo de la operación que desee ejecutar.
2. Invocación de pipeline de acuerdo con la petición.
3. Después de haber ejecutado una de las operaciones, el componente pipeline responde la petición.(se utilizan los adaptadores enmarcados)



La implementación OBS mostrada en la imagen anterior, se hace uso de los componentes:

|  |  |
| --- | --- |
| **Componente** | **Icono del componente** |
| Adaptador de base |  |
| Tubería |  |

El módulo principal se denomina pipeline , este componente es el encargado de enrutar las peticiones desde el servicio de entrada hacia los demás componentes que se encargan de diferente funcionalidad y viceversa, es decir, las respuestas de cada componente son enrutadas y así entregadas a la petición que se generó. A continuación, se muestra la interface del asistente que se utiliza en tiempo de implementación:

### 4.4.1 Lista de objetos

Los siguientes objetos fueron usados/creados para implementar la funcionalidad del servicio Web

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **#** | **Nombre** | **Tipo** | **Descripción** | **GIT** |
| 1 | FinancialsTecPs | Web  Service | Servicio Expuesto |  |
| 2 | dbaGetJournalsNEBs | Adaptador | Adaptador de base |  |
| 3 | dbaGetJournalsTMSBs | Adaptador | Adaptador de base |  |
| 4 | dbaUpdateJournalsNEBs | Adaptador | Adaptador de base |  |
| 5 | pipelineFinancialsTec | Tubería | Flujo de tubería |  |

### 4.4.2 GetJournalTMS

La operación **GetJournalTMS**realiza una serie de pasos basados dentro de un pipeline, a continuación, se enlistan los pasos a seguir para consumir dicha operación:

1. Se asigna la entrada del servicio a una variable “varGetJournalTMSRq”.
2. Se procede a enrutar hacia el servicio de negocios que deseamos consumir “dbaGetJournalsTMSBs”.
3. Se asignan los valores de la variable “varGetJournalTMSRq” al servicio de negocios “dbaGetJournalsTMSBs” y se consume dicho servicio.
4. Se recupera el resultado del servicio consumido, transformando y alojando estos dentro de una variable “GetJournalTMSRs”.
5. Se rutea al servicio saliente correspondiente a esta operación.
6. Se asignan los valores de la variable “varGetJournalTMSRs” hacía la salida del servicio.

Nota: En caso de existir algún error al momento de invocar al servicio externo, este será devuelto como la salida del servicio.

### 4.4.3 UpdateJournalTMS

La operación **UpdateJournalTMS**realiza una serie de pasos basados dentro de un pipeline, a continuación, se enlistan los pasos a seguir para consumir dicha operación:

1. Se asigna la entrada del servicio a una variable “varUpdateJournalTMSRq”.
2. Se procede a enrutar hacia el servicio de negocios que deseamos consumir “dbaUpdateJournalsTMSBs”.
3. Se asignan los valores de la variable “varUpdateJournalTMSRq” al servicio de negocios “dbaUpdateJournalsTMSBs” y se consume dicho servicio.
4. Se recupera el resultado del servicio consumido, transformando y alojando estos dentro de una variable “UpdateJournalTMSRs”.
5. Se rutea al servicio saliente correspondiente a esta operación.
6. Se asignan los valores de la variable “varUpdateJournalTMSRs” hacía la salida del servicio.

Nota: En caso de existir algún error al momento de invocar al servicio externo, este será devuelto como la salida del servicio.

### 4.4.4 GetJournalNE

La operación **GetJournalNE**realiza una serie de pasos basados dentro de un pipeline, a continuación, se enlistan los pasos a seguir para consumir dicha operación:

1. Se asigna la entrada del servicio a una variable “varGetJournalNERq”.
2. Se procede a enrutar hacia el servicio de negocios que deseamos consumir “dbaGetJournalsNEBs”.
3. Se asignan los valores de la variable “varGetJournalNERq” al servicio de negocios “dbaGetJournalsNEBs” y se consume dicho servicio.
4. Se recupera el resultado del servicio consumido, transformando y alojando estos dentro de una variable “GetJournalNERs”.
5. Se rutea al servicio saliente correspondiente a esta operación.
6. Se asignan los valores de la variable “varGetJournalNERs” hacía la salida del servicio.

Nota: En caso de existir algún error al momento de invocar al servicio externo, este será devuelto como la salida del servicio.

### 4.4.5 Pruebas

Las pruebas se ejecutarán en el ambiente de desarrollo tomando la dirección del siguiente WSDL:

##### http:// ersoaprd-soa-0.snproduccin.vcner.oraclevcn.com:9073/soa-infra/services/default/JournalEnt/JournalEnt?WSDL

A continuación, se muestran los parámetros de entrada y la respuesta que se presentan en esta operación:

**Request:** Se require el source, status y batch\_id:



Response: El servicio retorna los datos que están contenidos en la base de datos de acuerdo a los parámetros seleccionados.

:

### 4.5 Diseño de Datos

A continuación, se listan los componentes SOA utlizados en las capas Ent y Biz.

### 4.5.1 Capa Empresarial

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Nombre** | **Tipo** | **Descripción** |
| 1 | assignWsJournalBizGetJournalRq | assign | Asignación de valores. |
| 2 | invokeWsJournalBizGetJournal | invoke | Invocación de Servicio. |
| 3 | IfValidate | if | Validación de datos. |
| 4 | assignVarInterfaceL1 | assign | Asignación de valores. |
| 5 | trWsJournalBizGetJournalRsToBpelRs | transformation | Transformación de datos. |
| 6 | invokeWsERPIntegrationBizImportToERPCloud | invoke | Invocación de Servicio. |
| 7 | assignVarInterfaceL2 | assign | Asignación de valores. |
| 8 | trWsJournalBizGetJournalRsToBpelRs | transformation | Transformación de datos. |
| 9 | invokeWsERPIntegrationBizImportToERPCloud | invoke | Invocación de Servicio. |

### 4.5.2 Capa de Negocio

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Nombre** | **Tipo** | **Descripción** |
| 1 | IfValidateSource | if | Validación de datos. |
| 2 | IfValidateSuccess | if | Validación de datos. |
| 3 | assignWsFinancialsTecGetJournalNERq | assign | Asignación de valores. |
| 4 | invokeWsFinancialsTecGetJournalNE | invoke | Invocación de Servicio. |
| 5 | trWsFinancialsTecGetJournalNERsToVarGetJournals | transformation | Transformación de datos. |
| 6 | trWsFinancialsTecGetJournalRsToVarJournalsL1 | transformation | Transformación de datos. |
| 7 | translateJornalsL1 | transformation | Transformación de datos. |
| 8 | trWsFinancialsTecGetJournalRsToVarJournalsL2 | transformation | Transformación de datos. |
| 9 | translateJornalsL2 | transformation | Transformación de datos. |

## 5 Origen de Datos

Los orígenes de datos se encuentran dentro del adaptador de base, por lo cual en este apartado se mencionará aquellos servicios web junto con las operaciones ocupadas para poder realizar dicha integración.

La siguiente tabla muestra los distintos orígenes de datos correspondientes a esta integración dentro de la capa de técnica.

|  |  |  |
| --- | --- | --- |
| **Servicio** | **Operación** | **Descripción** |
| FinancialsTec | GetJournalTMS | Extrae pólizas de viajado. |
| GetJournalNE | Extrae pólizas de Nómina de Empleados. |
| UpdateJournalTMS | Actualiza pólizas de viajado. |

## 6 Lógica de Validación

Esta lógica hace referencia a las validaciones que se emplean en los orígenes para poder tener la información congruente y poder procesarla para finalmente extraer pólizas, la forma de realizar este proceso lo podemos encontrar en el documento de capas técnicas.

Sin embargo, podemos mencionar que las principales validaciones de negocio son las siguientes:

 Las pólizas deberán tener un batch\_id null.

## 7 Diseño SQL

Este es mediante los adaptadores de base que se encuentran en el servicio FINANCIALSTEC el cual invoca procedimientos y/o funciones del paquete PXERINTUSER.XXER\_UTILITIES\_PKG y PXERINTNOM.XXER\_UTILITIES\_PKG.

### 7.1 Sentencias SQL

### 7.1.1 Nómina de Empleados

Procedimiento utilizado para obtener pólizas Nomina de empleados gl y actulizar batch\_id



### 7.1.1 TMS

Funcion utilizado para obtener pólizas viajado gl y actulizar batch\_id



Procedire utilizado para actulizar batch de gl



### 

## 8 Reglas de Negocio

### 8.1 Diseño del Servicio

A continuación, se describe la entrada y salida correspondiente del JournalEnt

Para la operación SendFileJournal, el mensaje de entrada se denomina SendFileJournalRq

|  |  |  |  |
| --- | --- | --- | --- |
| **Elemento** | **Sub-Elemento** | **Tipo** | **Requerido** |
| SendFileJournalRq |  | tns:SendFileJournalRqType | Sí |
| SendFileJournalRs |  | tns:SendFileJournalRsType | Sí |
|  | varInterfaceL1 | string | Sí |
|  | varInterfaceL2 | string | Sí |
| SendFileJournalRqType |  |  | Sí |
|  | Source | string | Sí |
|  | Status | string | Sí |
| SendFileJournalRsType |  |  | Sí |
| Return |  | tns:SendFileJournalReturnType | No |
| SendFileJournalReturnType |  |  | Sí |
| Process |  | tns:ProcessType | 1 o Más |
| ProcessType |  |  | Sí |
|  | FileLedgerName | string | Sí |
|  | Id | long | Sí |
|  | ProcessName | string | Sí |
|  | Status | string | Sí |

A continuación, se describe la entrada y salida correspondiente del JournalBiz para la operación GetJournal, el mensaje de entrada se denomina GetJournalRq

|  |  |  |  |
| --- | --- | --- | --- |
| **Elemento** | **Sub-Elemento** | **Tipo** | **Requerido** |
| GetJournalRq |  | tns:GetJournalRqType | Sí |
| GetJournalRs |  | tns:GetJournalRsType | Sí |
| GetJournalRqType |  |  | Sí |
|  | Status | string | Sí |
|  | Source | string | Sí |
| GetJournalRsType |  |  | Sí |
| Return |  | tns:GetJournalReturnType | No |
| GetJournalReturnType |  |  | Sí |
| Result |  | tns:JournalType | Opcional-Lista |
| JournalType |  |  | Sí |
|  | FileName | string | Sí |
|  | FileContent | base64Binary | Sí |

## 9 Consideraciones de Rendimiento

Este requerimiento ha sido probado con un conjunto de recibos pequeños, tener contemplado que el compuesto puede presentar demora con una cantidad de registros muy grande a procesar.

### 9.1 Estrategia de Reinicio

* Para llevar a cabo un reinicio de la aplicación no es necesario realizar movimiento en base de datos, la aplicación al ser reiniciada continuará con la ejecución que corresponde.
* Supervisar que al momento del reinicio no existan instancia del proceso en ejecución, de ser así, esperar a que estas terminen para asegurar la congruencia de datos.

### 9.2 Seguridad

* Se recomienda el monitoreo oportuno de la base de datos para asegurar el correcto espacio para su crecimiento.
* El proceso principal para las pólizas es JournalEnt el cual solo se encuentra expuesto en el ambiente interno de Estrella Roja.
* En caso de requerir exponerlo a un mayor nivel, se recomienda pasar por un servicio OSB para no poner en riesgo la infraestructura del dominio SOA.

.

### 9.3 Personalización

En caso de querer modificar el proyecto, tomar la versión más reciente del controlador de versiones con el que se cuente.

La versión del IDE de desarrollo de JDeveloper con la que se implementó dicha solución es JDEVADF\_12.2.1.2.0\_GENERIC\_161008.1648.S.

## 10 Catálogo de Errores

La lista de errores de SOA que se podrían suscitar, se encuentran dentro del archivo

“CatalogosErroresEstrellaRoja.xlsx”.

## 11 Consideraciones de Instalación

A continuación, se listan los pasos a seguir para la instalación de esta integración.

1. Creación de objetos de base de datos, estos deben haberse creado antes de instalar los compuestos de la capa técnica.

Objetos de base para Nómina de empleados (Base en Cloud):

Procedimientos

* PXERINTNOM.XXER\_UTILITIES\_PKG. MERGE\_JOURNALS\_NE
* PXERINTNOM .XXER\_UTILITIES\_PKG. MERGE\_JOURNALS

Tablas

* PXERINTNOM.XXER\_GL\_EQV\_TBL
* PXERINTNOM.GL\_INTERFACE

Tipos

* PXERINTNOM.XXER\_GL\_INTERFACE\_TYPE\_T
* PXERINTNOM.XXER\_GL\_INTERFACE\_TYPE

Vistas

* PXERINTNOM.XXER\_GL\_INTERFACE\_V



Objetos de base para viajado TMS (Base en Cloud):

Procedimientos

* XXER\_UTILITIES\_PKG.MERGE\_JOURNALS\_TMS
* XXER\_UTILITIES\_PKG.UPDATE\_JOURNALS\_TMS

Tablas

* "PXERINTUSER"."XXER\_GL\_EQV\_TBL"
* PXERINTUSER.XXER\_GL\_INTERFACE

Tipos

* XXER\_GL\_INTERFACE\_TYPE\_T
* XXER\_GL\_INTERFACE\_TYPE

Vistas

* XXER\_GL\_INTERFACE\_V



1. Los servicios de la capa técnica y de negocio deben de estar desplegados en SOA.
2. Instalar la aplicación compuesta de SOA JournalBiz (Este archivo se encuentra en los entregables, el cual tiene el mismo nombre con extensión Jar).
3. Instalación de la aplicación compuesta de SOA JournalEnt (Este archivo se encuentra en los entregables, el cual tiene el mismo nombre con extensión Jar).
4. Crear la calendarización de la operación “SendFileJournal” del compuesto “JournalEnt”, esta se muestra en el punto 3.1.

## 12 URL de Acceso y seguridad

A continuación se anexa el WSDL correspondiente al servicio JournalEnt, el cual dentro de la infraestructura de la empresa no requiere seguridad.

URL para el acceso al servicio web en el ambiente de desarrollo:

|  |  |
| --- | --- |
| Servicio | URL |
| **JournalEnt** | **http://<hostname>:<puerto>/soa-**  **infra/services/default/JournalEnt/JournalEnt?WSDL** |
| **JournalBiz** | **http://<hostname>:<puerto>/soa-infra/services/default/JournalBiz/JournalBiz?WSDL** |
|  |  |
|  |  |
|  |  |
| **FinancialsTec** | **sb://<hostname>:<puerto>/FinancialsTecPs** |

## 13 Temas abiertos y cerrados

### 13.1 Temas Abiertos

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ID** | **Tema** | **Solución** | **Responsabilidad** | **Fecha Objetivo** | **Fecha impacto** |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

### 13.2 Temas Cerrados

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ID** | **Tema** | **Solución** | **Responsabilidad** | **Fecha Objetivo** | **Fecha impacto** |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |