|  |
| --- |
|  |

OUM

TA.070-1 DISEÑO TÉCNICO DE EXTENSIONES E

INTERFACES

Un dibujo de una cara feliz

Descripción generada automáticamente con confianza baja

ER23\_AP\_Subir Complentos de Pagos

|  |  |  |
| --- | --- | --- |
| Autor: | Forte Innovation | |
| Fecha de creación: | 28 de octubre de 2021 | |
| Última actualización: | 02 de febrero de 2022 | |
| Código de referencia OUM: | BAN\_IDEINI\_V1.5\_F000n\_Rev.1.4  \_RequerimientosFuncionalesCXP  2052021-firmado | |
| Versión:  **Aprobadores:** | 1.1 | |
| Juan Gonzalez | |  |

Gregorio Flores

|  |  |
| --- | --- |
| Wendi Holguin |  |

Mirian Lavin



**Contenido**

1 Control de Documento 2

1.1 Bitácora de Cambios 2

1.2 Revisores 2

2 Resumen técnico 3

3 Diagrama de la integración 4

4 Lógica de implementación 5

4.1 Calendarización y Ejecución 5

4.2 ValidationPaymentsAPEnt 7

4.3 ValidationPaymentsAPBiz 14

4.4 DigitalStampGERTec 35

5 Diseño de Datos 42

5.1 Origen de Datos 42

5.2 Lógica de Validación 43

6 Diseño SQL 45

6.1 Sentencias SQL 45

7 Reglas de Negocio 46

7.1 Diseño del Servicio 47

8 Consideraciones de Rendimiento 48

8.1 Estrategia de Reinicio 48

8.2 Seguridad 48

8.3 Personalización 48

8.4 Catálogo de Errores 50

9 Consideraciones de Instalación 51

10 URL de Acceso y seguridad 52

11 Temas abiertos y cerrados 53

11.1 Temas Abiertos 53

11.2 Temas Cerrados 53

# Control de Documento

## Bitácora de Cambios

3

|  |  |  |  |
| --- | --- | --- | --- |
| **Fecha** | **Autor** | **Versión** | **Referencia del cambio** |
| 02-02-2022 | Pedro Sanchez | 1.1 | Lógica complementos con DR en USD pago MXN. |
| 04-09-2023 | Pedro Sanchez | 1.2 | MA173 Configuración complementos de pago |
|  |  |  |  |
|  |  |  |  |

## Revisores

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

# Resumen técnico

Esta especificación documenta el diseño para el timbrado de transacciones, cuyo caso de uso se describió en el documento **BAN\_IDEINI\_V1.5\_F000n\_Rev.1.4\_RequerimientosFuncionalesCXP 2052021-firmado**.

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.

# Diagrama de la integración



# Lógica de implementación

Como mencionamos anteriormente, se cuenta con tres capas en la integración de complementos de pagos, en esta sección describiremos la lógica que se utilizó en cada una de ellas.

## Calendarización y Ejecución

A continuación, se describe la programación de esta capa en el ESS, consta de 1 elemento, un job que se ejecuta 1 vez al día, encargado ejecutar la integración para enviar las Notificaciones de forma automática:

|  |  |  |
| --- | --- | --- |
| Componente | Propiedad | Valor |
| Trabajo | Nombre | ValidationPaymentsAPEnt-Notifications |
| Nombre mostrado | ValidationPaymentsAPEnt-Notifications |
| Paquete | /oracle/apps/ess/custom/soa |
| Descripción | Envio de notificaciones validacion de Pagos AP |
| Tipo de trabajo | SyncWebserviceJobType |
| Operación | SendPaymentsAP |
| Solicitud |  |
| Planificación | Nombre | SchedulerValidationPaymentsAPEnt-Notifications |
| Nombre mostrado | SchedulerValidationPaymentsAPEnt-Notifications |
| Paquete | /oracle/apps/ess/custom/soa |
| Descripción | Temporizador de ejecucion de envoi de notificaciones de AP validacion de complementos |
| Frecuencia | Una vez al dia |
|  | Zona horaria | UTC-06:00 Mexico City |

A continuación, se describe la programación de esta capa en el ESS, consta de 1 elemento, un job que se ejecuta 1 vez al día, encargado ejecutar la integración para Generar las carpetas de forma automática:

|  |  |  |
| --- | --- | --- |
| Componente | Propiedad | Valor |
| Trabajo | Nombre | VelidationPaymentsBiz-GenerateFolders |
| Nombre mostrado | JobGenerateFolders |
| Paquete | /oracle/apps/ess/custom/soa |
| Descripción | Generacion automatica de carpetas Comprobantes de Pago AP |
| Tipo de trabajo | SyncWebserviceJobType |
| Operación |  |
| Solicitud |  |
| Planificación | Nombre | SchedulerGenerateFolders |
| Nombre mostrado | SchedulerGenerateFolders |
| Paquete | /oracle/apps/ess/custom |
| Descripción | Candelarizacion de ejecucion de procesdo de generacion de carpetas automatico de pagos de AP |
| Frecuencia | Una vez al dia |
|  | Zona horaria | UTC-06:00 Mexico City |

Se cuenta con servicio de recepción de correo que procesa los correros de respuesta de los proveedores ;

## ValidationPaymentsAPEnt

En la capa ValidationPaymentsAPEnt se realiza una serie de pasos concentrados en un bpel como se muestra en la siguiente imagen:

Diagrama

Descripción generada automáticamente

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.

### Lista de Objetos

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

|  |  |  |
| --- | --- | --- |
| **Elemento** | **Tipo** | **Descripción** |
| SendPaymentsAP | Web  Service | Servicio Expuesto |
| MediatorValidationPaymentsAPBiz | Mediator | Componente de SOA utilizado para integrar y enrutar los mensajes entre los servicios web |
| BpelSendPaymentsAP | Bpel | Componente de Orquestación |
| WsValidationPaymentsAPBiz | Web  Service | Servicio externo que tiene la lógica de negocio del proceso |
| wsErpIntegrationTec | Web    Service | Servicio externo que Reliza operaciones con el ERP importacion de templates y consulta de reportes |
| WsMessageTec | Web  Service | Servicio que envia notifiaciones. |

### BpelSendPaymentsAP

Como servicio expuesto de lado izquierdo tenemos la operación **SendPaymentsAP** que es la operación programada en el ESS, posterior utilizamos un mediator para dirigir a **BpelSendPaymentsAP** donde se concentra la lógica para hacer el llamado a las capas Biz correspondientes (ValidationPaymentsAPBiz)

El bpel realiza las siguientes acciones:

1. Se crea la variable de Response a enviar.
2. Asigna datos de la petición del servicio WsErpIntegrationTec que consulta un modelo de datos.
3. Se invoca el servicio de WsErpIntegrationTec – RunDataModel retona modelo de datos en B64.
4. Java que decodifica b64 recibido del modelo de datos
5. Asignamos el modelo de datos a un objeto XML mediante parseo.
6. Asignamos response con error general.
7. Restornamos el response con el error asignado.

Diagrama

Descripción generada automáticamente

1. Condición si se tienen pagos en la respuesta del modelo de datos.
2. Transformación de mapeo de datos para generar petición de Servicio WsValidationPaymentsAPBiz- SendPaymentsAP.
3. Se invoca el servicio de WsValidationPaymentsAPBiz- SendPaymentsAP encargado de enviar y asignar msj de notificacion.
4. Proceso vacio (no hace nada)
5. Retorna response Success.

Diagrama

Descripción generada automáticamente

### Lista de objetos

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

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Nombre** | **Tipo** | **Descripción** |
| 1 | assignRs | assign | Asignación de valores. |
| 2 | assignParams | assign | Asignación de valores. |
| 3 | WsErpIntegrationTec | invoke | Invocación de Servicio. |
| 4 | JavaDecoded | Java Enbebido | Ejecuta código java |
| 5 | AssignParseStringXML | assign | Asignación de valores. |
| 6 | assignCatchAll | assign | Asignación de valores. |
| 7 | replyOutput | Reply | Retorna Respuesta |
| 8 | IfExistPagos | IF | Condicional |
| 9 | Transformation1 | transformation | Transformación de datos. |
| 10 | WsValidationPaymentsAPBiz | invoke | Invocación de Servicio. |
| 11 | Empty1 | empty | No hace nada |
| 12 | replyOutput | Reply | Retorna Respuesta |

### 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/PROD/ValidationPaymentsAPBiz/ValidationPaymentsBiz?WSDL>

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

**Request:** No se necesitan parámetros de entrada para la ejecución de esta operación ya que es el dado de la fecha no es obligatorio:

Texto

Descripción generada automáticamente

Response: El servicio no retorna nada.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

## ValidationPaymentsAPBiz

En la capa **ValidationPaymentsAPBiz** se tiene una mayor cantidad de pasos a realizar debido a que se gestiona la integracion. Se parte de la operación expuesta ValidationPaymentsBiz y ReceiveEmail, posterior se tiene un mediador que direcciona múltiples bpel dependiendo de la operacion estos gestionan una serie de pasos que invoca a el BpelSendNotifications, a continuación, se explicará a detalle cada uno de ellos. El proceso general se muestra en la siguiente imagen:



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 |  |
| SubProccess |  |
| UMS soa |  |
| SFTP |  |

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.

### Lista de Objetos

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

|  |  |  |
| --- | --- | --- |
| **Elemento** | **Tipo** | **Descripción** |
| ValidationPaymentsBiz | Web  Service | Servicio Expuesto |
| BpelValidateAttachmentsAP | Bpel | Componente de Orquestación |
| BpelSendNotifications | Bpel | Componente de Orquestación |
| BpelSendPaymentsAP | Bpel | Componente de Orquestación |
| BpelGenerateFolders | Bpel | Componente de Orquestación |
| MediatorValidationPaymentsBiz | 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 |
| SubSendInvoiceToReport | Web  Service | Componente de Orquestación sencillo |
| SBpelValidateHeader | SubBpel | Componente de Orquestación sencillo |
| WsERPIntegrationServiceTec | Web  Service | Servicio externo |
| wsMessageTec | Web  Service | Servicio externo |
| WsERPIntegrationBiz | Web  Service | Servicio externo |
| SendNotificationService | Web  Service | Servicio externo |
| wsAuditControlTec | Web  Service | Servicio externo |
| WsValidacionAutomaticTec | Web  Service | Servicio externo |
| FTP\_ER\_SOA\_GET | Servidor SFTP | Servidor de archivos externo |
| FTP\_ER\_SOA | Servidor SFTP | Servidor de archivos externo |

### BpelValidateAttachmentsAP

El **BpelValidateAttachmentsAP** contiene los siguientes pasos:

1. Java embebido encargado de realizar la descomposición del msj de correo recibido para obtener el ID del pago a validar.
2. Asignación de la variable de v\_payment\_id.
3. Condicional si no encuentra v\_payment\_id en el correo.
4. termino de ejecución y termina el proceso.
5. Catch de errores.
6. Empty no realiza ninguna acción .

Diagrama

Descripción generada automáticamente

1. Asignacion de WsErpIntegrationBizGenerateReportRq de servicio que consulta reporte para saber los datos de la factura a validar .
2. Invocación del servicio ErpIntegrationBiz-GenerateReport .
3. Asignación de objeto de respuesta v\_payment del resultado del reporte consultado.
4. Condicional que valida si existe información en la consulta realizada del pago.
5. Condicional que valida si el estatus es suspendido.
6. Asignación de estatus de pago suspendido .
7. Asignación de estatus de pago no existe.
8. Secuencia de validación de pago.

Diagrama

Descripción generada automáticamente

1. For que recorre las líneas para asignar la lista en el correo de notificación
2. Asignación de información de lista de facturas de notificación.
3. Asignación de datos de notificación y inserción de tabla a nivel header.
4. Condicional si el pago ya esta en un estatus validado.
5. Asignación de mensaje de complemento de pago ya validado.
6. For que recorre los attachments recibidos en el correo.
7. Asignamos valores del attachement en curso a variables.
8. Java embebido que muestra los valores de las variables previamente asignadas.
9. Condicional que evalúa que en tipo del archivo sea XML o PDF.

Diagrama

Descripción generada automáticamente con confianza baja

1. Condicional que evalúa si el archivo tiene un nombre definido o asignamos uno .
2. asignación de nombre genérico en caso de que no tenga.
3. Asignación del nombre encontrado.
4. En caso de no ser un tipo valido se borra del arreglo de adjuntos para que no se cargue en el ERP.
5. Asignación de $WsSOAUtilitiesTecDecodeRq del servicio que decodifica el adjunto.
6. Asigna al arreglo de adjuntos la información a subir $WsERPIntegrationServiceTecUploadAttachmentRq.
7. Invocación del servicio que decodifiac un b64 SOAUtilitiesTec-Decode.
8. Asigna documento y pasea a XML en la variable $v\_paymentXML.

Interfaz de usuario gráfica, Diagrama

Descripción generada automáticamente con confianza media

1. Condicional que valida el numero de xml obtenidos en caso de sea diferente de 1 manda notificación de error.
2. Condicional que evalúa que el tipo de pago sea P encaso contrario manda msj con el error.
3. Asignación de valores del header del XML.
4. Asignación de msj de notificación de error de tipo de pago no valido.
5. Asignación de msj de notificación de error número de XML no valido.
6. Sbpel de validación de header.
7. Condicional que evalúa si el sbpel retorno errores en el header.
8. Sbpel que realiza las validaciones de las líneas del sistema vs líneas del XML.
9. Asignación de mensajes de error de validaciones a nivel header.

Interfaz de usuario gráfica

Descripción generada automáticamente

1. Condicional que evalúa si el response de las líneas tiene errores.
2. asignación de mensajes de errores de líneas de facturas.
3. Invocación de servicio que sube los adjuntos al ERP ERPIntegrationServiceTec-UploadAttachment.
4. Asignación de mensajes estatus de ejecución correcta.

Diagrama

Descripción generada automáticamente

1. invocación de servicio AuditControlTec-MergeComplementsAP.
2. Invocación de servicio NotificationEmail.

Diagrama

Descripción generada automáticamente con confianza media

**SBpelValidateHeader**

1. Condicional si el RFC emisor del sistema es diferente al del XML, asigna notificación y estatus de error.
2. Condicional si el RFC receptor del sistema es diferente al del XML, asigna notificación y estatus de error.
3. Condicional si el fecha de pago del sistema es diferente al del XML, asigna notificación y estatus de error.
4. Condicional si el importe de pago del sistema es diferente al del XML.
5. asigna notificación especificando los montos y estatus de error.
6. Condicional si el numero de líneas del sistema es diferente al del n de líneas del XML.
7. asigna notificación de mas líneas y estatus de error.
8. asigna notificación de menos líneas y estatus de error.

Interfaz de usuario gráfica, Diagrama

Descripción generada automáticamente

**SBpelValidateHeader**

1. For que recorre todas las líneas de la factura a validar .
2. Asigna los valores de la línea del sistema al merge de la tabla de datos.
3. Asigna la consulta a las tablas de la información de la factura.
4. Ejecuta servicio que consulta información de la factura ValidationAutomaticTec-GetReportValidation.
5. For para recorrer los pagos del XML.
6. For para recorrer los documentos relacionados del XML.
7. Condicional que valida si existe el uuid de la línea a validar.
8. asigna de valores de la línea del XML a validar.

Diagrama, Esquemático

Descripción generada automáticamente

1. Si el XML existe valida lo demás si no manda error .
2. Si la moneda de la línea es MXN asigna un monto si es otro asigna otro monto.
3. Asigna variable del monto a validar para MXN.
4. Asigna variable del monto a validar para otras monedas.
5. Asigna variables con las que se genera el mensaje de notificación.
6. Java embebido que genera el mensaje de notificación con las variables asignadas.
7. Asignación de estatus y mensaje de notificación .
8. Si el monto la comparación de los montos en menor a la tolerancia definida manda success si no acumula en error de la línea en curso.
9. Condicional que valida si existe el uuid de la línea a validar.
10. Java embebido que genera el mensaje de notificación
11. Asignación del mensaje y estatus de notificación.

Diagrama

Descripción generada automáticamente

### BpelSendPaymentsAP

El **BpelSendPaymentsAP** contiene la mayor lógica para el envió automático de notificaciones el proceso realizando los siguientes pasos:

1. Input de entrada del desarrollo recibe todos los pagos a notificar.
2. Asignación de valores de respuesta default.
3. Si el desarrollo limpia variables o esta en fase de pruebas se asigna variables verdadero o falso.
4. Asignación de valores verdadero o falso para pruebas y limpiar datos.
5. For que recorre los pagos recibidos en el input.
6. Proceso que genera los distintos tipos de notifiaciones.
7. Catch de excepciones.
8. Asignación de error general
9. Invoca servicio que genera html de mensaje de respuesta del desarrollo.
10. Asignación de mensaje de notificación y estatus del servicio.
11. Ejecución de servicio de notificación email.

Interfaz de usuario gráfica, Diagrama

Descripción generada automáticamente

En la siguiente imagen se describen los pasos contenidos en el scope de ForPayments:

1. Asigna los valores del pago para el mensaje de respuesta con los datos que ya se tienen.
2. Asigna los datos de la petición a enviar para la insersion del registro del pago en la tabla de complementos.
3. For que recorre las líneas para generación del listado de facturas a responder en el correo.
4. Bloque que asigna los valores de las líneas en el mensaje de notificación.
5. Si es el código es 1 es recordatorio de adjunto si es 2 es cancelación de adjunto si es 0 quiere decir que se debe valorar que tipo de notificación se va enviar.
6. Asignación de notificación de recordatorio de solicitud de adjuntos.
7. Asignación de notificación de suspensión de pago.
8. Bloque de loguica que define que tipo de notificación se envía según el pago.
9. Invoca servicio encargado de enviar la información del pago a tablas de complementos.
10. Invocación de servicio encargado del envió de las notificaciones.

Diagrama

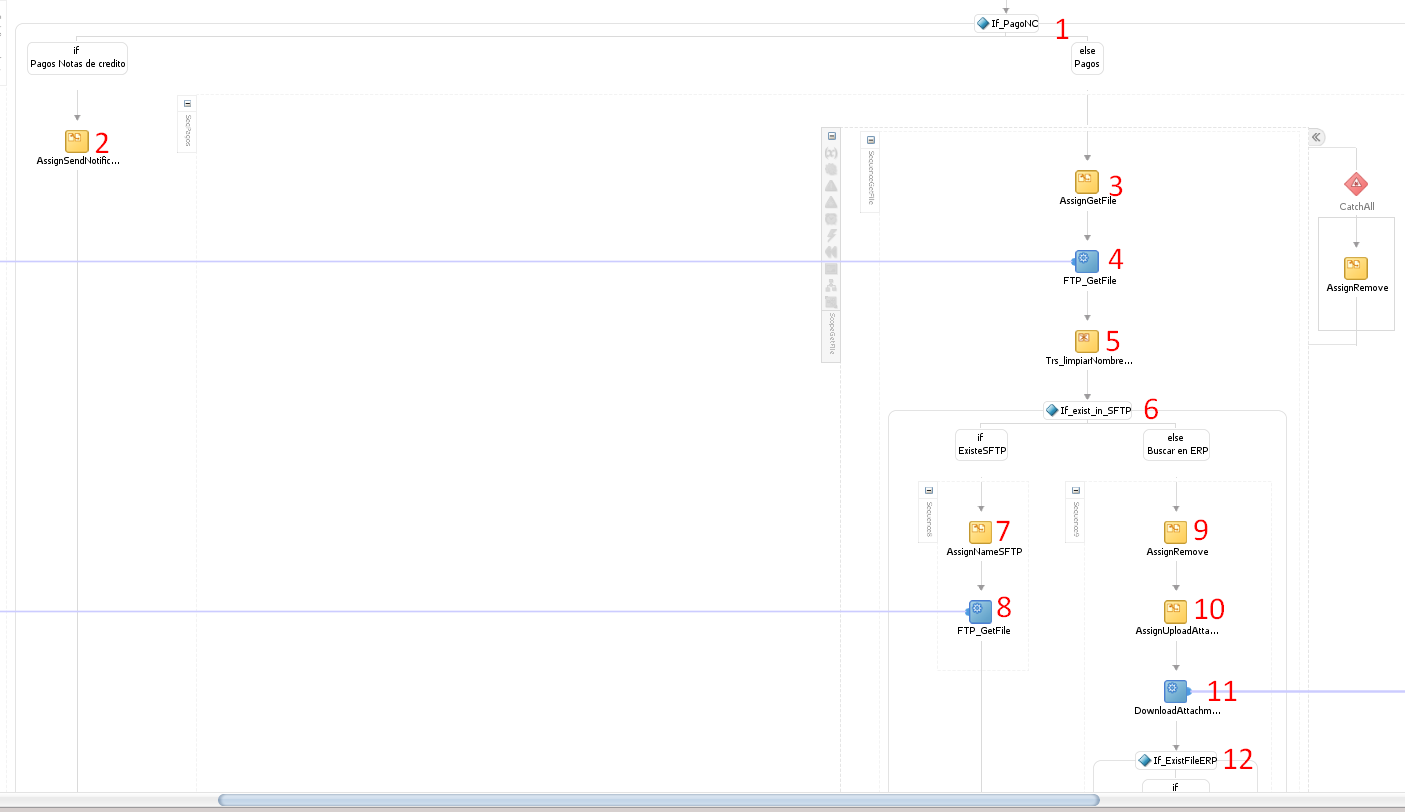
Descripción generada automáticamente

Diagrama

Descripción generada automáticamente con confianza media

En la siguiente imagen se describen los pasos contenidos en el seqEnvioComprobante:

1. Si el pago tienen un monto .01 es nota de crédito y se notifica si no sige el proceso
2. Asignación de mensaje y estatus de notificación de nota de crédito.
3. Asignamos valores para la consulta para listar lo adjuntos de la ruta del SFTP.
4. Invocamos servicio que consulta en el SFTP lista de adjuntos.
5. Ejecutamos transformación que limpia los nombres de los adjuntos e identifica si existe el adjunto en la carpeta.
6. Evaluamos si existe el adjunto
7. Asignamos el nombre encontrado del adjunto
8. Ejecutamos el servicio que extraer el Adjunto del SFTP
9. Limpiamos los valores de consulta de adjunto
10. Asignamos datos para consultar adjuntos en ERP
11. Ejecutamos servicio de consulta de adjuntos en ERP.
12. Si exiten adjuntos asignamos el primero a la variable de adjunto.
13. En caso de error en bloque del proceso de obtención de adjuntos se toma como adjunto vacío.
14. Si se encontró un comprobante en el SFTP el flujo sigue de lo contrario se notifica que no se encontró adjunto en el SFTP.
15. Proceso que define que notificación se va a enviar.
16. Asignación de mensaje y notificación de comprobante no encontrado en el SFTP.





En la siguiente imagen se describen los pasos contenidos en el ScopeNotificacion:

1. For que recorre las facturas del pago de entrada.
2. Validamos si es un prepayment para asignar que tiene una factura prepayment de lo contrario sigue el flujo.
3. Asignamos la bandera que tiene prepayment.
4. Asignamos la petición para la consulta de los datos de la factura.
5. Invocamos servicio de consulta de facturas.
6. Si existe la factura validamos si es tipo pue de lo contrario lanzamos el proceso que inserta la factura a la tabla.
7. Si la factura es tipo pue asignamos la bandera de pue como verdadera.
8. Asignamos variable de bandera pue como verdadero.
9. Scope de proceso que inserta los datos de la factura a la tabla donde se consultan los datos.

Interfaz de usuario gráfica

Descripción generada automáticamente

En la siguiente imagen se describen los pasos contenidos en el ScopeSendInvoiceToReport:

1. asignación de datos para ejecutar CallSendInvoiceToReport.
2. Llamada a **SBpelSendInvoiceToReport** proceso que se encarga de la extracción de información del XML de la factura y envió de a la tabla de información.
3. Si el proceso inserta la información de forma correcta se regresa a donde se realiza la consulta de lo contrario se manda error de no se tiene xml si hay un error en la generación de la información se manda ese error.
4. Replay que retorna el proceso al paso de la consulta de la factura.
5. Asignación de error de xml no encontrado en la factura.
6. Generación de excepción para termino del proceso.
7. Asignación de error de ejecución incorrecta de proceso de extracción de XML.
8. Generación de excepción para termino del proceso.

Diagrama

Descripción generada automáticamente

1. Si es PPD asigna notificación de solicitud de adjuntos de lo contrario sigue el proceso de adelanto pue.
2. Asignación de mensaje y estatus de solicitud de adjuntos.
3. Si es adelanto pue y es adelanto o si es puro pue envia notificación para cada caso.
4. Asignación de mensaje y estatus notificación de pago adelanto pue.
5. Asignación de mensaje y estatus notificación de pago adelanto.
6. Asignación de mensaje y estatus notificación de pago pue.
7. Asignación de petición para subir archivos al ERP.
8. Invocación de servicio encargado de subir adjuntos al ERP:

Diagrama

Descripción generada automáticamente

1. Si hay error por notificar.
2. Invoca servicio que genera el cuerpo del email a enviar.
3. Asigna petición de servicio que envía email.
4. Invoca servicio de envió de notificaciones de email

Diagrama

Descripción generada automáticamente

**SubSendInvoiceToReport**

1. Asignación de petición para consulta de ajuntos de la factura en ERP.
2. Invocación del servicio que consulta los b64 de los adjuntos de la factura en el ERP.
3. For utilizado para recorrer los adjuntos encontrados .
4. Asignamos petición de ejecución de servicio que decodifica el B64.
5. Invocamos servicio que decodifica b64.
6. Asignamos el string a un objeto CFDI parseando los datos.
7. Asignamos datos para la petición de inserción de datos de la factura a la tabla.
8. Catch de errores en el proceso.
9. Empty no es un archivo xml.

Diagrama

Descripción generada automáticamente

1. invocación del servicio que inserta la factura el la tabla para su consulta.
2. Si es correcto se asigna success si no se asigna el error.
3. Asigna success.
4. Asigna error.
5. Si no existe xml.
6. Asigna estatus de que no existe XML.

Diagrama

Descripción generada automáticamente

### BpelSendNotifications

El **BpelSendNotifications** contiene la mayor lógica para el envió de notificaciones el proceso realizando los siguientes pasos:

1. Input de entrada del desarrollo recibe todos los pagos a notificar.
2. Asignación de valores de respuesta default.
3. Si el desarrollo limpia variables o esta en fase de pruebas se asigna variables verdadero o falso.
4. Asignación de valores verdadero o falso para pruebas y limpiar datos.
5. For que recorre los pagos recibidos en el input.
6. Proceso que genera los distintos tipos de notifiaciones.
7. Catch de excepciones.
8. Asignación de error general
9. Invoca servicio que genera html de mensaje de respuesta del desarrollo.
10. Asignación de mensaje de notificación y estatus del servicio.
11. Ejecución de servicio de notificación email.

Diagrama

Descripción generada automáticamente

### 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/PaymentComplementEnt/PaymentComplementbIz?WSDL>

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

**Request:** contrato del arreglo de complementos a timbrar:

Texto

Descripción generada automáticamente con confianza baja

Response: no retorna nada en caso de proceso correcto .

Texto

Descripción generada automáticamente

## DigitalStampGERTec

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. Diagrama

   Descripción generada automáticamenteDespués 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 Rest | Interfaz de usuario gráfica  Descripción generada automáticamente |
| 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:

### Lista de objetos

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

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **#** | **Nombre** | **Tipo** | **Descripción** | **GIT** |
| 1 | DigitalStampGERTecPS | Web  Service | Servicio Expuesto |  |
| 2 | DigitalStampGERApi | Adaptador | Adaptador de REST |  |
| 3 | DigitalStampGERTecPL | Tubería | Flujo de tubería |  |

### createTrxCFDI

La operación **createTrxCFDI** 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 “createTrxCFDIRq”.
2. Se toma el tipo de documento que se va a procesar de document\_type\_description y se coloca en v\_operation.
3. Se procede a mapeo de datos con la transformación DigitalStampRq\_to\_DigitalStampAPIRq.xsl para generar la petición del servicio Rest a consumir y se guarda en la variable varXMLJsonSchema.
4. Ejecutamos un traslate que convierte nuestro Shema XML a Json .
5. Se asigna nuestro json generado al cuerpo de la petición a enviar.
6. Se rutea al servicio saliente correspondiente a esta operación.
7. Dependiendo de la v\_operation ejecutamos la API correspondiente y enviamos la petición generada al Adaptador DigitalStampGERApi
8. Si la respuesta es correcta convertimos la respuesta de Json a XML con un traslate.
9. Después ejecutamos el mapeo de datos de DigitalStampGERApi a DigitalStampGERTecPS con la transformación DigitalStampAPIRs\_to\_DigitalStampRs.xsl.
10. Una ves Realizado esto remplazamos el resultado en el cuerpo de respuesta.

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

### Lista de objetos

La siguiente tabla contiene los objetos que fueron usados para la funcionalidad del DigitalStampGERTecPL

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Nombre** | **Tipo** | **Descripción** |
| 1 | BranchNode | BranchNode | Menú de operaciones del servicio |
| 2 | Pipeline Pair Node | Pipeline Pair Node | Flujo de petición y response del servicio |
| 3 | Assing | assign | Assingna un valor a una variable |
| 4 | nXSD Translate | nXSD Translate | Transforma datos de un esquema nativo a otros o viceversa |
| 5 | Error Handler | Error Handler | Cacha los errores ocurridos. |
| 6 | If Then | If Then | operación que condición si . |
| 7 | Java Callout | Java Callout | Ejecuta clase java. |
| 8 | Replace | Replace | Remplaza una variable por otra |
| 9 | Condicional Branch | Condicional Branch | Condiciona que servicios ejecutar. |
| 10 | insert | insertt | Inserta valores a la petición o salida. |

### Pruebas

Las pruebas se ejecutarán en el ambiente de desarrollo tomando la dirección de Direct Biding:

sb://ersoaprd-soa-0.snproduccin.vcner.oraclevcn.com:9073/DigitalStampGERTecPS

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

**Request:** Valores de entrada no requeridos

Texto

Descripción generada automáticamente

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

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

# Diseño de Datos

Hace referencia a las tablas custom empleadas en esta integración, como es que se consultan y se almacena la información correspondiente a Complementos de pago.

## Origen de Datos

Hace referencia a las tablas custom empleadas en esta integración, dichas tablas son solo de control.

Tabla: **XXER\_PAYMENTS**

|  |  |  |
| --- | --- | --- |
| **Columna Tipo de dato** | | **Descripción** |
| PAYMENT\_ID | VARCHAR2(150 CHAR) | Identificador unico del Pago |
| CASH\_RECEIPT\_ID | VARCHAR2(3 CHAR) | Identificador del Recibo AR |
| RECEIPT\_NUMBER | VARCHAR2(150 CHAR) | Número de recibo |
| CUSTOMER\_NUMBER | NUMBER | Número de cliente |
| ACCOUNT\_NUMBER | NUMBER | Número de cuenta |
| SITE\_NUMBER | NUMBER | Número de Site |
| UUID | CLOB | UUID generado por el SAT |
| UUID\_DATE | VARCHAR2(150 CHAR) | Fecha de creación del UUID |
| STATUS | VARCHAR2(150 CHAR) | Estatus { 'STAMPED', 'ERRORED', 'NOT\_CONNECTED'} |
| PAYMENT\_TYPE | VARCHAR2(150 CHAR) | Type { 'INCOME', 'OUTCOME', 'COMPLEMENT'} |
| RECEIP\_METHOD | VARCHAR2(150 CHAR) | Metodo del recibo |
| CONTENT | TIMESTAMP(6) | Payload del registro |
| CREATED\_BY | VARCHAR2(150 CHAR) | Indica el usuario que creo el registro |
| CREATION\_DATE | TIMESTAMP(6) | Indica la fecha de creación |
| LAST\_UPDATED\_BY | VARCHAR2(150 CHAR) | Indica el último usuario que modificó el registro |
| LAST\_UPDATE\_DATE | TIMESTAMP(6) | Indica la última fecha de modificación |
| SOURCE\_LAST\_UPDATED\_BY | VARCHAR2(150 CHAR) | Indica el usuario que actualizó por última vez el recibo |
|  |  | en el momento del procesamiento |
| OBJECT\_VERSION\_NUMBER | NUMBER(9,0) | Número de modificaciones del registro |

Tabla: **XXER\_PAYMENT\_DETAILS**

|  |  |  |
| --- | --- | --- |
| **Columna Tipo de dato** | | **Descripción** |
| PAYMENT\_DETAIL\_ID | NUMBER(18,0) | Identificador unico del detalle de pago |
| PAYMENT\_ID | NUMBER(18,0) | ID del Payment |
| INVOICE\_NUMBER | VARCHAR2(150 CHAR) | Número de Factura |
| INSTALLMENT\_NUMBER | NUMBER(3,0) | Parcialidad |
| RECEIVABLE\_APPLICATION\_ID | NUMBER(18,0) | AR Application ID |
| TRX\_NUMBER | VARCHAR2(20 CHAR) | AR Referencia de Aplicación |
| BALANCE\_BEFORE | NUMBER | Saldo Previo |
| BALANCE\_AFTER | NUMBER | Saldo Pendiente |
| UUID | VARCHAR2(150 CHAR) | UUIDde la factura |
| STATUS | VARCHAR2(150 CHAR) | Estatus { ''STAMPED'', ''ERRORED'', ''NOT\_CONNECTED''} |
| CURRENCY\_CODE | VARCHAR2(3 CHAR) | Código de moneda |
| CREATED\_BY | VARCHAR2(150 CHAR) | Indica el usuario que creo el registro |
| CREATION\_DATE | TIMESTAMP(6) | Indica la fecha de creación |
| LAST\_UPDATED\_BY | VARCHAR2(150 CHAR) | Indica el último usuario que modificó el registro |
| LAST\_UPDATE\_DATE | TIMESTAMP(6) | Indica la última fecha de modificación |
| OBJECT\_VERSION\_NUMBER | NUMBER(9,0) | Número de modificaciones del registro |

## Lógica de Validación

Esta lógica hace referencia a las validaciones que se emplean en los origines para poder tener la información congruente y poder procesarla para finalmente timbrar el complemento de pago.

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

* El recibo debe estar listo para timbrar
* No debe ser un anticipo
* Debe ser una creación manual y no provenir de la superinterface.
* Sus aplicaciones debieron ser previamente timbradas
* Sus aplicaciones deben ser de tipo de pago 99-Por definir
* Solo nos interesan las aplicaciones de tipo factura

Las cuales se pueden observar en los queries del tema 5.1 Sentencias SQL, también se cuenta con las validaciones explicadas en el punto 3. Lógica de implementación

* No debe tener UUID.
* No debe tener Folio.
* Debe tener un estatus de Reprocesar, Pago con Anticipo o bien no debe tener un valor.
* El tipo de documento debe ser Factura o Nota de Crédito.

Las cuales se pueden observar en los queries del tema 5.1 Sentencias SQL, también se cuenta con las validaciones explicadas en el punto 3. Lógica de implementación

# Diseño SQL

En esta integración existen principalmente dos consultas, una obtiene todos los recibos pertinentes que se enviarán a timbrar y la otra obtiene todas las aplicaciones del recibo.

## Sentencias SQL

### Query para obtener los recibos a timbrar.



### Query para obtener las aplicaciones del recibo.



# Reglas de Negocio

Las reglas de negocio se describen a continuación, para saber más detalle puedes observar el documento BAN\_IDEINI\_V1.5\_F000n\_Rev.1.4\_RequerimientosFuncionalesCXP 2052021-firmado donde se especifica todo el requerimiento.  Solo se timbrarán los recibos que son creados de forma manual dentro del ERP Cloud.



## Diseño del Servicio

Para llevar a cabo el inicio de la integración se tiene que ejecutar el compuesto de **ValidationPaymentsAPEnt** , el cual recibe 1 parámetros obligatorio , los cuales se describen a continuación.

|  |  |  |
| --- | --- | --- |
| **Argument Prompt** | | **Value Set** |
| p\_startDate | StartDate | dateTime | 01-01-2023 |

# 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.

## 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.

## Seguridad

* Se recomienda el monitoreo oportuno de la base de datos para asegurar el correcto espacio para su crecimiento.
* 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.

## Personalización

* En caso de requerir 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\_PT.12.2.1.4.0\_GENERIC\_190911.2248.S
* La integración de Complemento de Pagos cuenta con parámetros que pueden ser personalizables, los cuales se describen a continuación.

En la parte de configuración se tienen las siguientes DVMs



## Catálogo de Errores

Se cuenta con una lista de errores en el archivo CatalogosErroresEstrellaRoja.xlsx, de manera muy particular particular de esta aplicación, se cuenta con los siguiente:

|  |  |
| --- | --- |
| Código | Descripción |
| SOA-00021 | Fallo al agregar archivos adjuntos |
| SOA-00022 | El recibo ya está timbrado |
| SOA-00023 | El recibo no tiene aplicaciones |

Sin embargo, también se utilizan códigos generales compartidos con otras integraciones, ver el archivo mencionado para mayor detalle.



# Consideraciones de Instalación

Como Prerrequisitos se requiere un esquema de base de datos (DBCS) previamente configurado, con los data sources correspondientes, consultar el manual de configuración del ambiente.

En este esquema deben existir las siguientes tablas:

* XXER\_COMPLEMENTS\_AP\_TBL
* XXER\_COMPLEMENTS\_LINES\_AP\_TBL
* XXER\_INVOICES\_AP\_TBL

De lo contrario se requiere crearlas, el script de estas tablas se encuentra en el siguiente archivo, para ejecutarlo se debe contar con los permisos necesarios ya sea desde sqlplus o un entorno gráfico.



Instalación de la integración de complemento de pago tanto capa Ent como Biz, instalar como aplicación SOA:

* sca\_ValidationPaymentsAPBiz.jar
* sca\_ValidationPaymentsAPEnt.jar



# URL de Acceso y seguridad

A continuación, se definen los servicios utilizados en la integración cuyas URL están definidas con comodines que deben adaptarse dependiendo del ambiente del que se trate. Las dos primeras son propias de las integraciones de complemento de Pagos, las demás se incorporan para brindar funcionalidades especificas descritas anteriormente en el punto 3. Lógica de implementación

|  |  |
| --- | --- |
| Servicio | URL |
| **ValidationPaymentsAPEnt** | **http://<hostname>:<puerto>/soa-**  **infra/services/default/ValidationPaymentsAPEnt/ValidationPaymentsAPEnt?WSDL** |
| **ValidationPaymentsAPBiz** | **http://<hostname>:<puerto>/soa-**  **infra/services/default/ValidationPaymentsAPBiz/ValidationPaymentsAPBiz?WSDL** |
| CFDITec | http://<hostname>:<puerto>/soa-  infra/services/default/ CFDITec/CFDITec?WSDL |
| ERPIntegrationBiz | http://<hostname>:<puerto>/soa-  infra/services/default/ ERPIntegrationBiz/ ERPIntegrationBiz?WSDL |
| AuditControlTec | sb://<hostname>:<puerto>/AuditControlTecPs |
| SOAUtilitiesTec | http://<hostname>:<puerto>/soa-  infra/services/default/SOAUtilitiesTec/SOAUtilitiesTec?WSDL |
| MessageTec | http://<hostname>:<puerto>/soa-  infra/services/default/MessageTec/MessageTec?WSDL |
| ERPIntegrationServiceTec | sb://<hostname>:<puerto>/ ERPIntegrationServiceTec Ps |

# Temas abiertos y cerrados

## Temas Abiertos

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

## Temas Cerrados

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