**MA173 Configuración complementos de pago**

**Documentación técnica**

# **Control de documento**

### Registro de cambios

31

| **Autor** | **Puesto** | **Versión** | **Fecha Revisión** |
| --- | --- | --- | --- |
| Pedro Sanchez Martinez | Desarrollador SOA | 1.1 | 28 de diciembre de 2023 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

### Revisión de documento

| **NOMBRE** | **PUESTO** | **FECHA REVISIÓN** |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |

**CONTENIDO**

[**Control de documento** 2](#_Toc154625307)

[**Registro de cambios** 2](#_Toc154625308)

[**Revisión de documento** 2](#_Toc154625309)

[Introducción 4](#_Toc154625310)

[**Objetivo** 4](#_Toc154625311)

[**integraciones a afectar** 4](#_Toc154625312)

[Ajuste Técnico 5](#_Toc154625313)

[**Objeto Bpel RecordUnitMterics** 5](#_Toc154625314)

[**Objeto DVM dvmRecordUnitMetrics** 11](#_Toc154625315)

[**Instrucción de Instalación** 12](#_Toc154625316)

# Introducción

### Objetivo

El presente documento pretende facilitar la comprensión de las funciones incluidas en cada uno de los Web Services, favoreciendo el entendimiento e interacción de los diferentes procesos en cada sistema que consume cada servicio descrito en este documento.

Referencias

Este diseño técnico está formado a partir del documento de análisis aprobado por GER:

* TSD\_MA176 Modificación interfaz carga de kilometrajes.

Siendo el complemento del documento de diseño técnico de la integración proporcionado por GER:

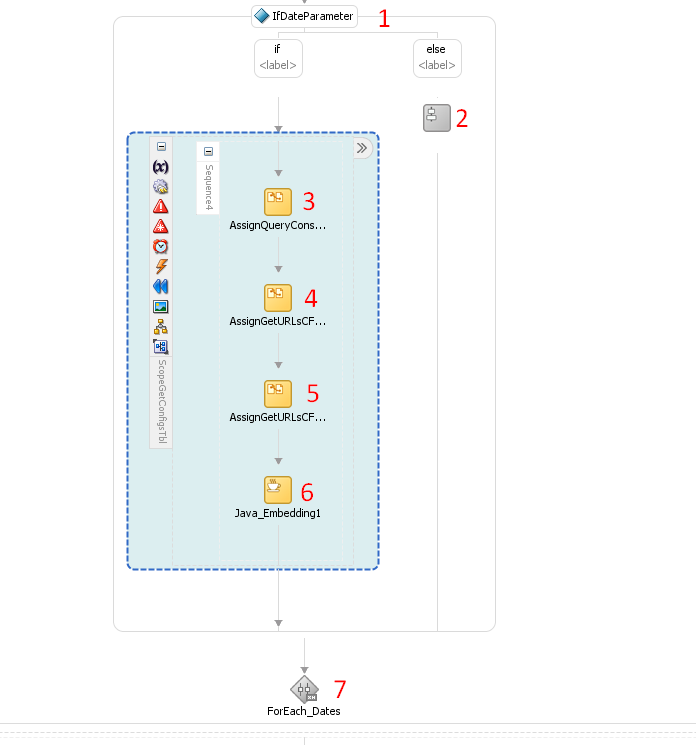
* TA.070-4 ER01\_Lectura de Kilometros\_V2.4

### integraciones a afectar

1. Integración de negocio de Lectura de Kilómetros.

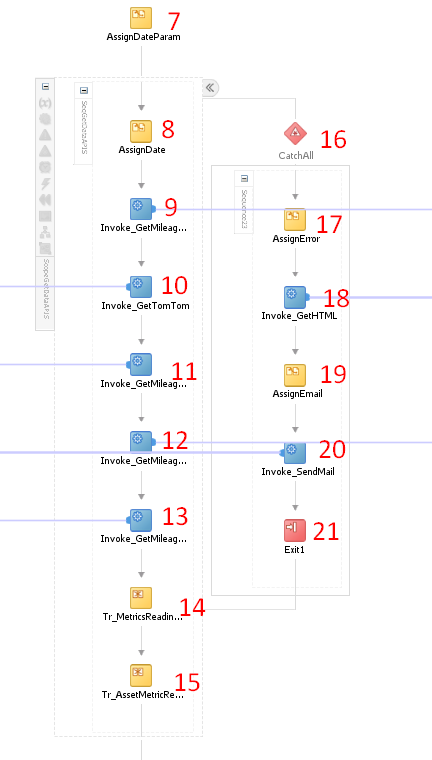
# Ajuste Técnico

### Objeto Bpel RecordUnitMterics



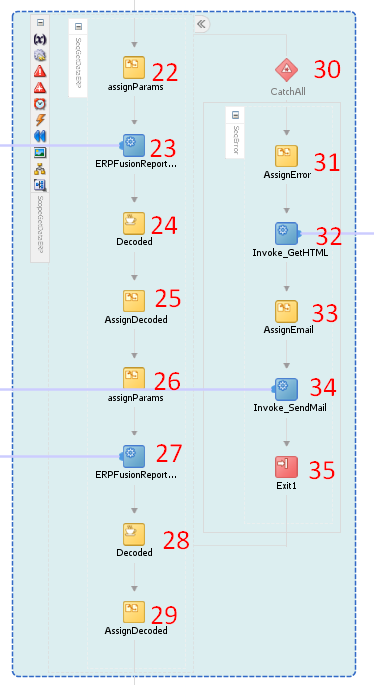
**Descripción**

1. Condición if que evalúa existe el rango de fecha en los valores de entrada de la integracion.
2. Asignación de sysdate por default.
3. Asignación de consulta para obtener listado del rango de fechas recibido.
4. Ejecución de consulta y replace de namespace asignación a variable string.
5. Asignación de variable string a variable node mediante función parse.
6. Muestra de datos mediante java.
7. For para recorrer listado de rango de días.



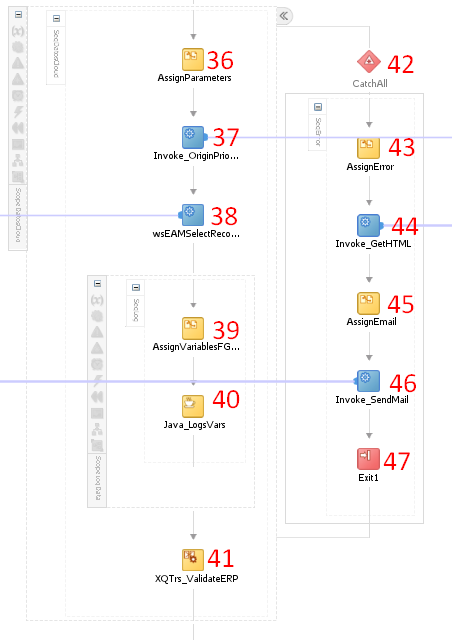
**Descripción**

1. Asignación de datos de entrada de cada API a consultar.
2. Ejecución de API Controlnet.
3. Ejecución de API TomTom.
4. Ejecución de API Omnitracs.
5. Ejecución de API Sine.
6. Ejecución de API Excel.
7. Transformación para Agrupar el resultado de todas las APIS.
8. Transformación para Reagrupar resultado de todas las APIS por unidad y ordenar por fecha más actual.
9. Catch all de errores.
10. Asignamos el error en la variable GetHtmlControlRq.
11. Ejecutamos Invoke\_GetHTML que se encarga de generar el texto de salida en HTML para la notificación.
12. Asignamos el mensaje de notificación en variable In\_SendMail.MessageRq.
13. Ejecutamos Invoke\_SendMail que se encarga enviar la notificación de correo electrónico.
14. Exit del proceso.



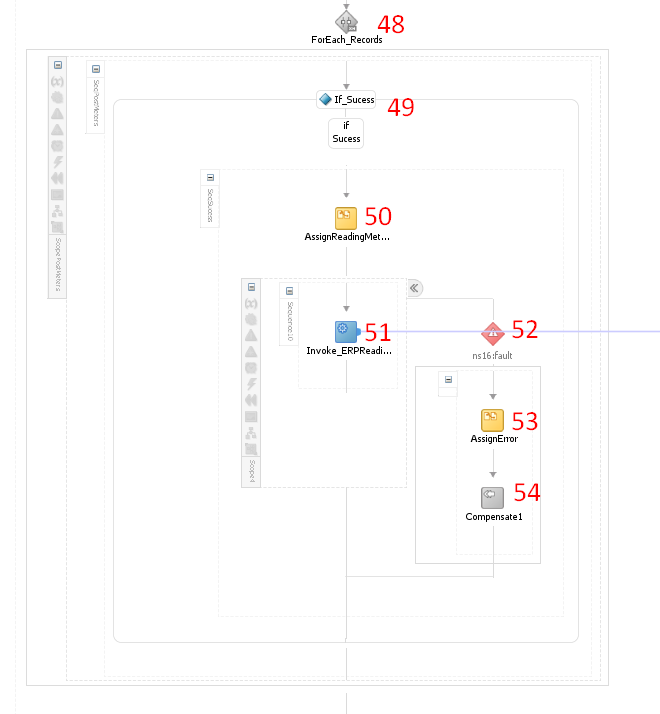
**Descripción**

1. Asignación de datos de variable RunDataModelRq para Consultar reporte ASSETS.
2. Ejecutamos Invoke ERPFusionReportASSETS que se encarga de generar Consultar reporte otbi de ASSETS.
3. Java para decodificar base 64 devuelto por el Invoke ERPFusionReportASSETS.
4. Asignación de string decodificado a variable nodo.
5. Asignación de datos de variable RunDataModelRq para Consultar reporte Readings.
6. Ejecutamos Invoke ERPFusionReportReadings que se encarga de generar Consultar reporte otbi de Readings.
7. Java para decodificar base 64 devuelto por el Invoke ERPFusionReportReadings.
8. Asignación de string decodificado a variable nodo.
9. Catch all de errores.
10. Asignamos el error en la variable GetHtmlControlRq.
11. Ejecutamos Invoke\_GetHTML que se encarga de generar el texto de salida en HTML para la notificación.
12. Asignamos el mensaje de notificación en variable In\_SendMail.MessageRq.
13. Ejecutamos Invoke\_SendMail que se encarga enviar la notificación de correo electrónico.
14. Exit del proceso.



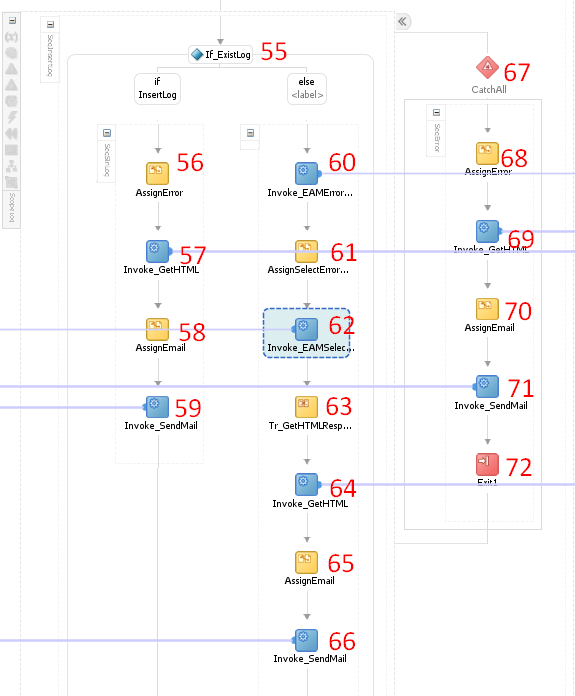
**Descripción**

1. Asignación de datos de variables de entrada para el servicio Invoke\_OriginPriority y wsEAMSelectRecordService.
2. Ejecutamos Invoke Invoke\_OriginPriority nos retorna el listado de prioridad de inserción de origen por unidad.
3. Ejecutamos Invoke wsEAMSelectRecordService nos retorna el ultimo registro success por unidad.
4. Asignación de las variables en string para visualizar en log.
5. Java que codifica string para mostrar en log.
6. Transformación con lógica para validar y calcular kilómetros a partir de todas las consultas previas de información, retorna la salida de errores eh inserciones correctas.
7. Catch all de errores.
8. Asignamos el error en la variable GetHtmlControlRq.
9. Ejecutamos Invoke\_GetHTML que se encarga de generar el texto de salida en HTML para la notificación.
10. Asignamos el mensaje de notificación en variable In\_SendMail.MessageRq.
11. Ejecutamos Invoke\_SendMail que se encarga enviar la notificación de correo electrónico.
12. Exit del proceso.



**Descripción**

1. For que recorre los resultados obtenidos por cada unidad.
2. Condición IF si el estatus es success se inserta la medición.
3. Asignacion de variable de entrada In\_ERPReadingMetrics.
4. Ejecutamos Invoke Invoke\_ERPReadingMetrics servicio que ingresa las mediciones a ERP.
5. Catch all de errores.
6. Asignamos el error en la variable In\_EAMInsertError variable de log de errores.
7. Compásate para seguir iterando.

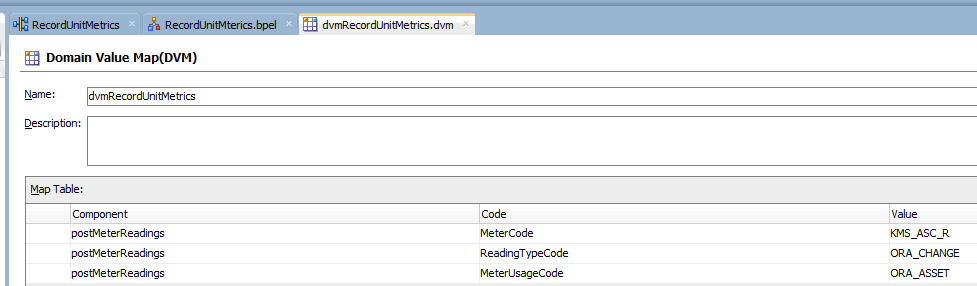


**Descripción**

1. Condición IF valida si Existen Datos procesados.
2. Asignamos el error de API EMPTY en la variable GetHtmlControlRq.
3. Ejecutamos Invoke\_GetHTML que se encarga de generar el texto de salida en HTML para la notificación.
4. Asignamos el mensaje de notificación en variable In\_SendMail.MessageRq.
5. Ejecutamos Invoke\_SendMail que se encarga enviar la notificación de correo electrónico.
6. Ejecutamos Invoke\_EAMErrorRecord que se encarga de enviar los registros procesados a la tabla de log.
7. Asignamos la variable de entrada In\_EAMSelectErrorRecord.
8. Ejecutamos Invoke\_EAMSelectErrorRecord de consultar el listado de errores por rango de fechas.
9. Transformacion que recibe listado de errores consultado para retornar la variable de entrada GetHtmlControlRq.
10. Ejecutamos Invoke\_GetHTML que se encarga de generar el texto de salida en HTML para la notificación.
11. Asignamos el mensaje de notificación en variable In\_SendMail.MessageRq.
12. Ejecutamos Invoke\_SendMail que se encarga enviar la notificación de correo electrónico.
13. Catch all de errores.
14. Asignamos el error en la variable GetHtmlControlRq.
15. Ejecutamos Invoke\_GetHTML que se encarga de generar el texto de salida en HTML para la notificación.
16. Asignamos el mensaje de notificación en variable In\_SendMail.MessageRq.
17. Ejecutamos Invoke\_SendMail que se encarga enviar la notificación de correo electrónico.
18. Exit del proceso.

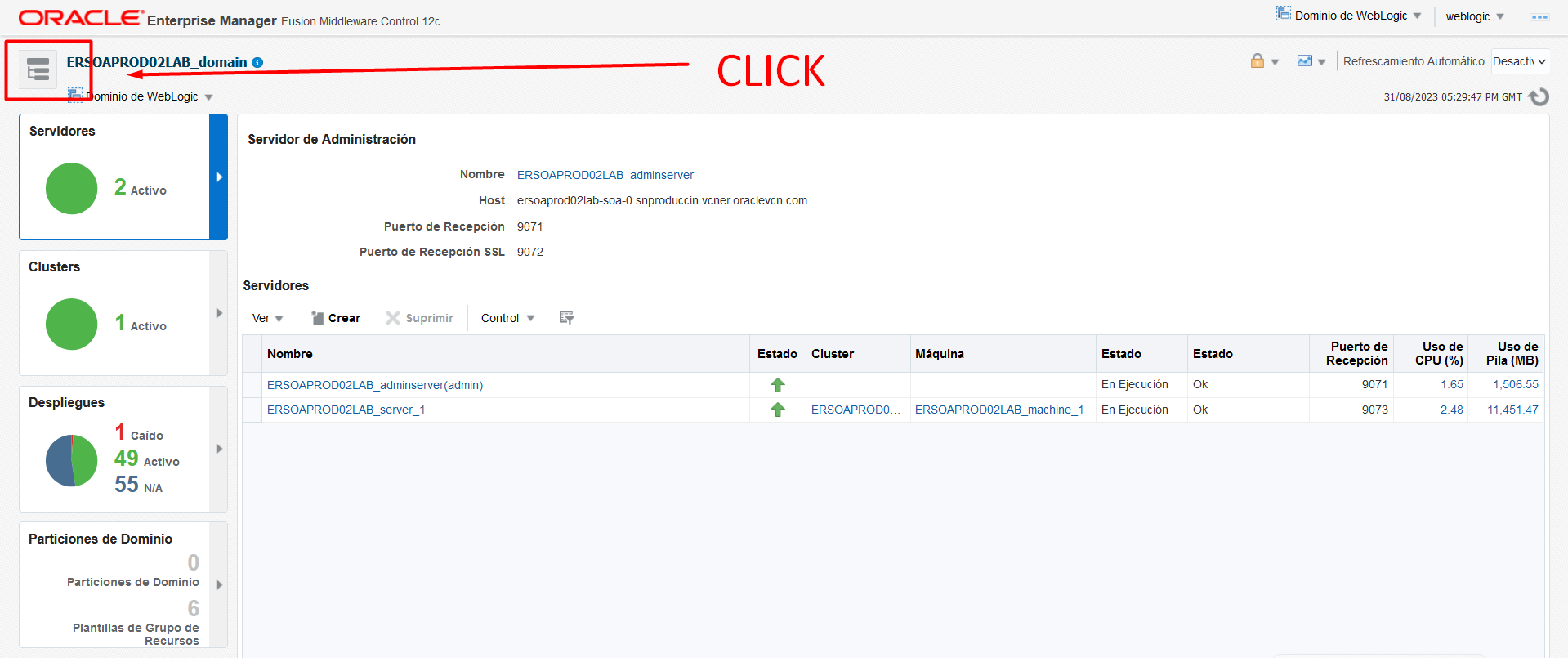
### Objeto DVM dvmRecordUnitMetrics

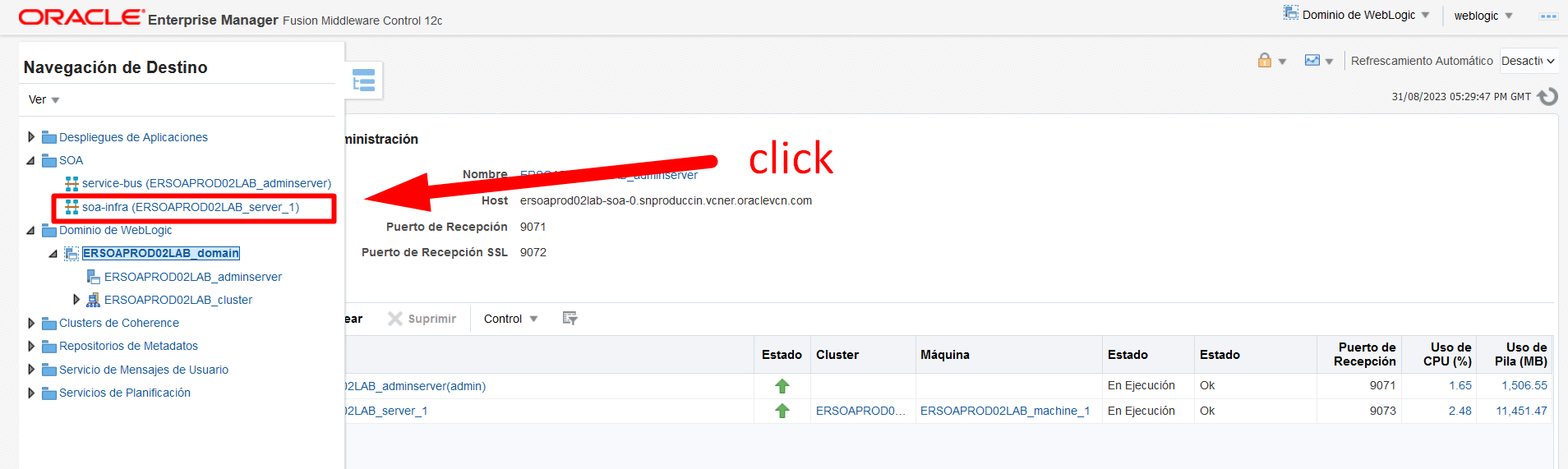
Se agrego DVM para no tener en duro los valores de configuración para el envió de los kilómetros al ERP.

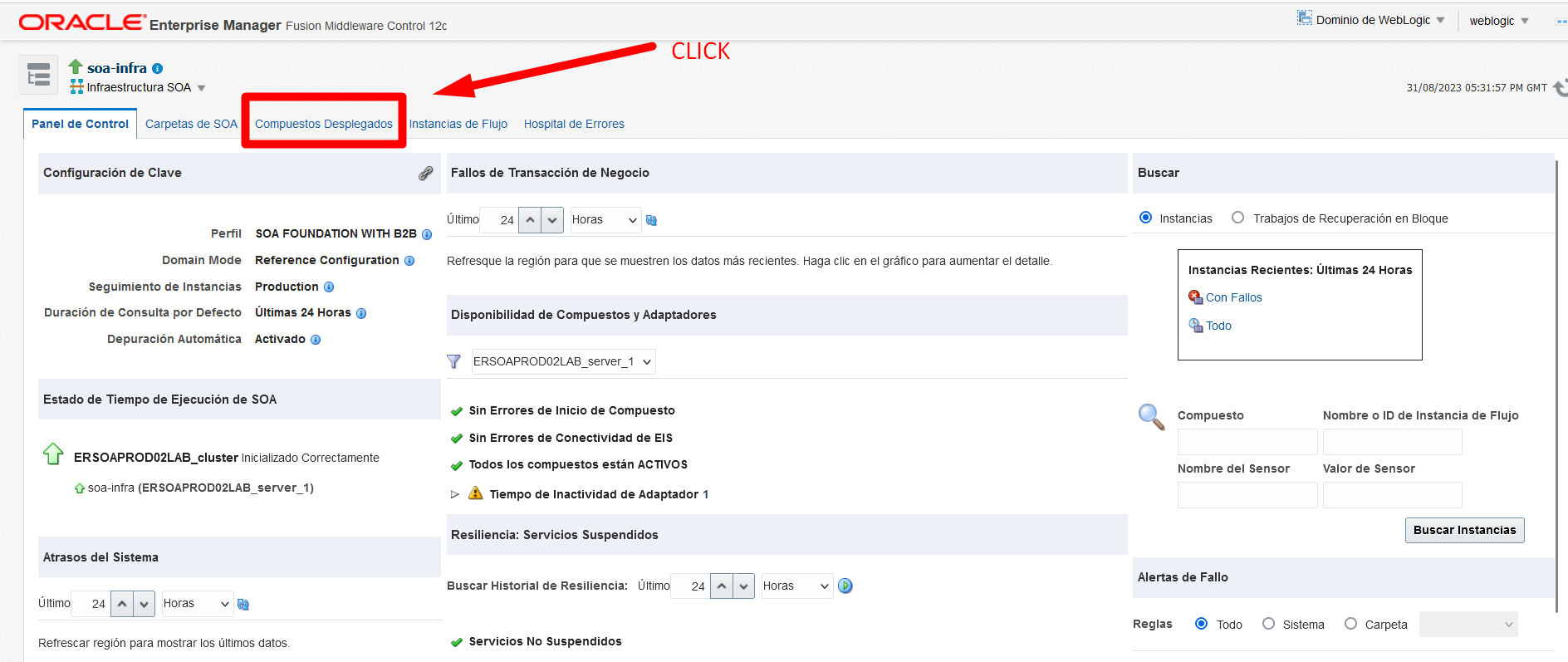


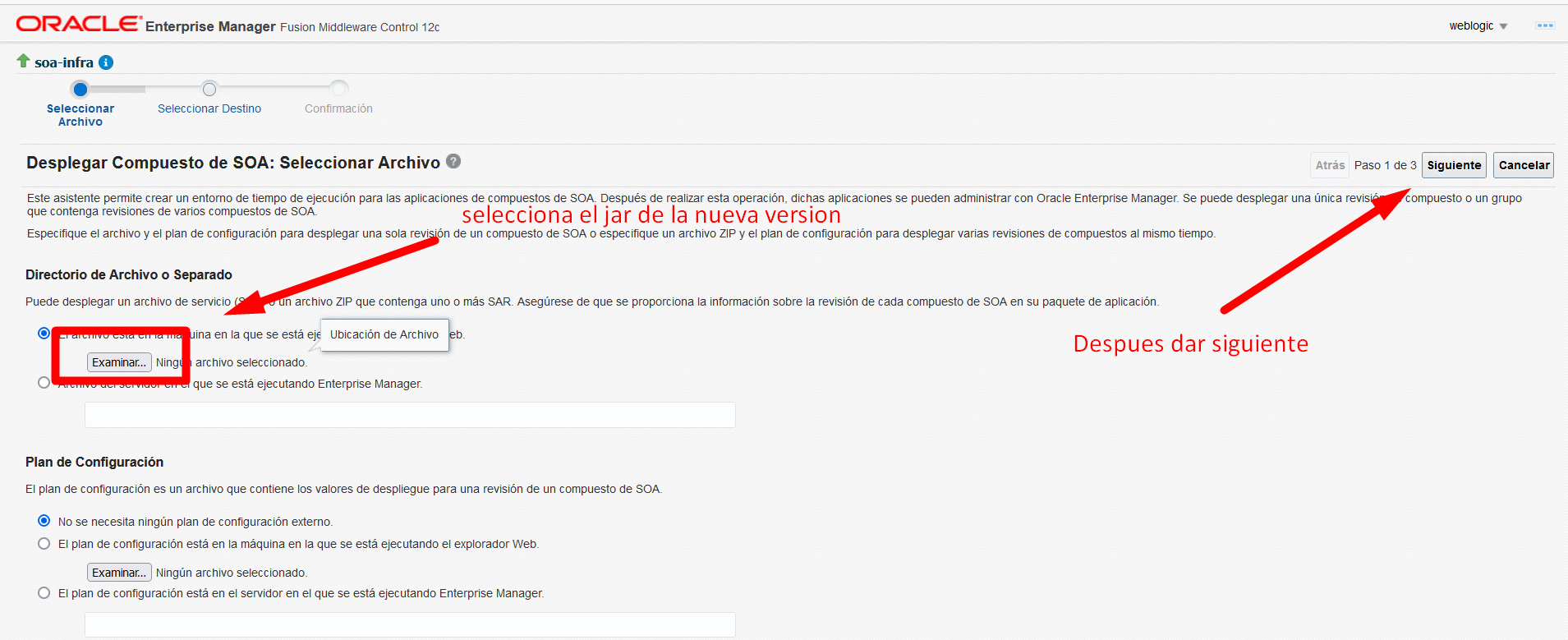
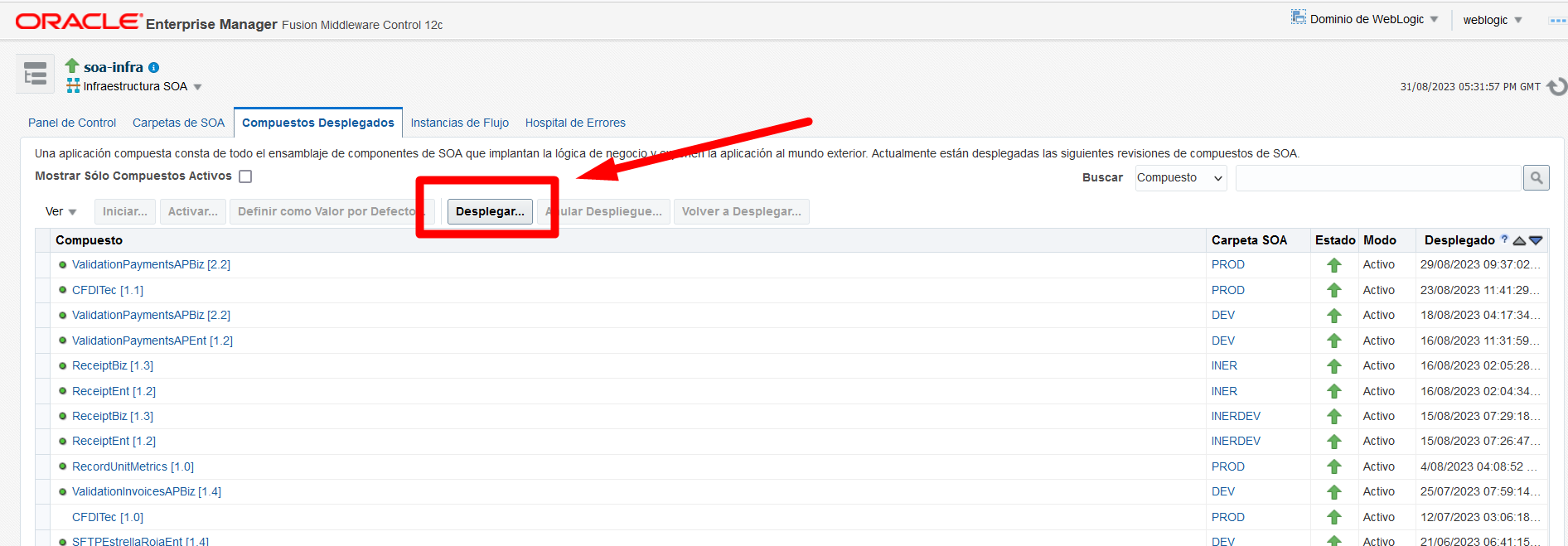
# **Instrucción de Instalación**

Solo se debe desplegar el jar de la integración como un desarrollo nuevo ya que este es la versión nueva.











MA173 Configuración complementos de pago

Versión 1.1 Diciembre de 2023

Elaborado por: Sanchez Martinez Pedro