



Nombre del Proyecto

Consultoría de Migración de Aplicaciones

Cliente

Servicio Nacional
de Aduanas



Fecha

09/05/16

Versión

1.6

Tipo de documento

Documento de Estrategia de Migración

1	Introducción.....	6
2	Objetivo.....	7
3	Aplicaciones que serán objeto de estudio.....	8
3.1	Tabla Resumen Aplicaciones a Migrar.....	10
4	Formulación de la Estrategia de Migración de Aplicaciones.....	13
5	Escenarios de Migración.....	16
6	Estrategia de Migración de Aplicaciones.....	17
6.1.1	Diagrama de Estrategia de Migración.....	18
7	Evaluar el estado actual de las aplicaciones.....	19
7.1	Levantamiento.....	19
7.2	Análisis de Brecha.....	20
7.3	Riesgos específicos.....	20
8	Catálogo de Aplicaciones.....	28
9	Indicadores de las aplicaciones.....	29
9.1	Riesgo de Migración (RM).....	29
9.2	Tamaño de aplicación.....	30
9.3	Estudio Consolidado.....	33
9.3.1	Conclusiones a partir del Estudio Consolidado.....	35
9.4	Valoración del Esfuerzo de Migración en Horas Hombre (HH).....	38
10	Agrupación de aplicaciones en escenarios.....	51
11	Ejecución del plan de migración.....	56
11.1	Plan de Migración.....	56
11.2	Recodificación de una aplicación.....	58
12	Validación de la migración.....	60
12.1	Pruebas Funcionales.....	60
12.2	Pruebas de Rendimiento y Estrés.....	60
13	Mejora Continua y Mantenimiento.....	61
13.1	Calidad del Código.....	61

13.2 Pruebas Unitarias.....	61
13.3 Pruebas de Integración.....	61
13.4 Pruebas de Regresión.....	61
13.5 Pruebas de Accesibilidad.....	61
13.6 Pruebas de Vulnerabilidad y Seguridad.....	61
13.7 Otros Controles.....	62
14 Análisis 80/20.....	63
15 ANEXO 1: Visión de la Arquitectura de Software de las Aplicaciones..	66
15.1 Arquitectura Actual.....	66
15.2 Arquitectura Propuesta.....	66
15.2.1 Propuesta de mejoras sobre la arquitectura actual.....	66
15.2.2 Propuesta arquitectura alternativa. frontend – backend.....	67
16 ANEXO 2: Plan técnico de Migración.....	69
16.1 Grupo Weblogic.....	69
16.1.1 Puntos de Análisis.....	69
16.1.2 Configuración del Servidor.....	69
16.1.3 Descriptores de Despliegue.....	69
16.1.3.1 Mapeo de Archivos de Configuración.....	69
16.1.3.2 Reemplazar el Descriptor de Despliegue.....	70
16.1.3.3 Migrar Archivos de Configuración.....	71
16.1.4 Seguridad.....	73
16.1.5 WebLogic JNDI Lookups a Portable JNDI.....	74
16.1.6 ServletAuthentication.....	74
16.1.7 WebLogic Transaction Manager.....	74
16.1.8 Hibernate y JPA.....	74
16.1.9 Reemplazar tecnologías propietarias de Weblogic.....	75
16.1.9.1 WebLogic CommonJ Framework.....	75
16.1.9.2 WebLogic Virtual Directories.....	75
16.1.9.3 Anotaciones Propias de Weblogic.....	75
16.1.9.4 NetUI PageFlow.....	77

16.2 Grupo Web Services.....	78
16.2.1 Puntos de Análisis.....	78
16.2.2 Servicios Web Específicos de Weblogic.....	78
16.2.3 Servicios Web JAX-RPC.....	79
16.2.3.1 Ajuste del contexto raíz.....	79
16.2.3.2 Anotaciones reutilizables.....	79
16.2.3.3 Fichero WSDL.....	79
16.2.3.4 Uso de tipos JAXB.....	79
16.2.3.5 EJB 3.0.....	79
16.2.3.6 RPC vs SOAP.....	79
16.2.3.7 Manejadores SOAP.....	79
16.2.3.8 Los Clientes WS.....	79
16.3 Grupo Aplicaciones de Escritorio.....	80
16.3.1 Introducción.....	80
16.3.2 Estrategia por capas.....	80
16.3.2.1 Presentación.....	80
16.3.2.2 Negocio.....	80
16.3.2.3 Persistencia.....	81
16.3.3 Java Web Start.....	81
17 Documentos Anexos.....	82
17.1 Arquitectura Actual.....	82
17.2 Arquetipo Actual.....	82
17.3 Agrupación de Aplicaciones.....	82
17.4 Riesgo de migración.....	82
17.5 Indicadores de las aplicaciones.....	82
17.6 Valoración del Esfuerzo de Migración (HH).....	82
17.7 Prueba de Concepto.....	82
17.8 Otros.....	82

CONTROL DE DIFUSIÓN

Nombre y Apellidos
Antonio Gabriel González Casado
Rafael Vázquez Ballesteros
Pedro Lisana
Gerardo Olmedo Nova
Eduardo Godoy Llanca

1 Introducción

Actualmente el Servicio Nacional de Aduanas (**SNA**) posee un conjunto de aplicaciones soportadas sobre diversas plataformas, la mayoría de estos sistemas se han convertido, a pesar de soportar procesos de core de negocio, en sistemas legados dado que las herramientas con las cuales fueron construidas ya tienen una obsolescencia considerable.

La Migración de Aplicaciones del Servicio Nacional de Aduanas es un macro-proyecto planteado en fases el cual se visualiza en el siguiente diagrama:



Diagrama 1: Fases de la migración de aplicaciones

La “Consultoría de Migración de Aplicaciones” (**CMAA**) busca levantar y analizar la situación actual de las aplicaciones del core de negocio con el fin de generar documentación de alto valor que permita enfrentar de la mejor manera posible el proyecto de migración de aplicaciones que será abordado en los años siguientes.

El proceso base de la consultoría de migración de aplicaciones está compuesto por 3 actividades secuenciales:

- Levantamiento de Situación Actual.
- Análisis de Brecha.
- Estrategia de Migración de Aplicaciones.

El presente documento corresponde a la tercera de las 3 actividades anteriormente enunciadas. La “Estrategia de Migración de Aplicaciones” permitirá tener una visión clara de cómo llegar a la situación futura deseada.

2 Objetivo

El objetivo del presente documento es establecer una visión global de la estrategia a seguir para migrar las aplicaciones del core de negocio del Servicio Nacional de Aduanas desde su plataforma de origen a la plataforma de destino (JBoss EAP 6.4, Java 7, JEE 6) de manera efectiva.

Objetivos Específicos:

1. Plantear una estrategia a seguir para poder abordar la migración de aplicaciones.
2. Hacer un análisis de restricciones, prioridades y condiciones que permitan decidir la estrategia más ajustada a estas necesidades.
3. Hacer un análisis cuantitativo y cualitativo que permita hacer una toma de decisión para determinar la prioridad de la migración de aplicaciones.

Consideraciones generales:

- El alcanzar la “arquitectura de software ideal” (Ver: [Arquitectura Propuesta](#)) en la primera fase de la migración de aplicaciones no es un objetivo prioritario considerando que la restricción principal es salir de la condición de obsolescencia en el menor tiempo posible.
- La operación de las aplicaciones migradas sobre la plataforma de destino deberá mantener los actuales parámetros de: Seguridad, Rendimiento y Calidad.

3 Aplicaciones que serán objeto de estudio

Originalmente el SNA determinó que eran 95 las aplicaciones objeto a estudio. Durante la fase de levantamiento se fueron descartando algunas de las aplicaciones por no disponer de la información para analizar.

Framework/ Tecnología/ Dependencias	Versión	Lenguaje	IDE Desarrollo	Nº de aplicacio nes	Servidor de Despliegue
Apex	3.2.1	PL/SQL - HTML	Apex	7	Oracle 9 - Apex 3.2.1
BEA Workshop 8.1**	8.1.6	Java 1.4	Workshop 8.1	31	BEA Weblogic 8.1
Z3*	1	Java 1.4	Workshop 8.1	23	BEA Weblogic 8.1
Servlet	2.4	Java 1.5	Eclipse 3.7	28	Jboss 4.1
Seam	2	Java 1.5	Eclipse 3.7	1	Jboss 4.1
Struts 1.x / JBPM	1.x / 2	Java 1.4	Workshop 8.1	1	Jboss SOAP 5.3
JSF / Richface / JEE	1.2/3.3.3/5	Java 1.6	Eclipse 3.7	4	Jboss 5.2
TOTAL				95	

***Z3: Framework desarrollado por Empresa Tuxpan**

****BEA Workshop 8.1: Framework creado por BEA Systema que luego fue cedido a la Fundación Apache con el nombre de Apache Beehive**

Después, en la fase de Análisis de Brecha, se agruparon algunas de las aplicaciones por su contexto ya que formaban parte de un mismo despliegue en un servidor de aplicaciones, dando como resultado **56 aplicaciones a ser objeto de estudio**. En la siguiente tabla se muestran las equivalencias entre las aplicaciones y su contexto:

Aplicación	Conexto	ID's de aplicaciones originales
01-SIGES	%APEX%/f?p=132	1
02-OPVIG	%APEX%/f?p=149	2, 3
04-Veh_Integrado	veh_integrado	4
05-SWVehiculos	/WSVehiculos.ear/servicioWeb	5
06-VehiculosRACWeb	/Vehiculos.ear/	6, 7
08-SCVM	/SNA_SCVM	8
09-VehiculosFronteraWeb	/VehiculosFronteraWeb	9

Aplicación	Conexto	ID's de aplicaciones originales
10-Control_Mensajeria	ventanillaPrueba	10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86
21-DECARE	/DTTA	21
22-SICOMEXIN	/SicomexinWeb/	22
23-DIPS_COURIER	/WebServiceTutorialWeb/	23
24-F18_POSTAL	/soap/	24
26-Mensajeria_DIN	Monitor	26
27-DepositoFranco-Intranet	/erosprod/	27, 28
29-Consultas_DUS	DespachadoresWeb	29, 30, 31, 92
33-DIPSCF	/DipscfWeb/	33
34-DIPS_Viajeros	/KNA/webAduanas/	34
35-DTI	/DTI/	35
36-WebSeguridad	/WebSeguridad	36
37-Web_AdmPersonas	/WebAdmPersonas	37
38-WSAA_con_Argentina	/ServicioWebMicDta	38
39-WSAA_Chile	/wsaa/servicio/	39
40-INDIRA	/SWIndira/servicio/	40
42-Firmas_ALADI	/aduanaFirmasAladiWeb	41
43-TramitacionIVV	/eros/ivv/	43
44-SDA	%DESKTOP%/SmdtWS/	44
45-SVC	/Verificador/	45
46-SIDECO	%DESKTOP%/SmdtWS/	46
47_DepWebFiscalizacion	/WebFiscalizaciones	47
48_SIFER	%DESKTOP%/SmdtWS/	48
49_SIROFE	%APEX%/f?p=199	49
50-SIDEMAR	%DESKTOP%/S2mcWS/	50
51-Servicio_Web_Canje_Maritim o	/WebServiceCanjeMaritimo/	51

Aplicación	Conexto	ID's de aplicaciones originales
52-Aplicacion_Web_Canje_Maritim o	/WebCanjeMaritimo	52
54-WSAlmacen	/WebServiceAlmacenes/	54
55-MicWeb	/TransporteTerrestre	55
56_SIROTE	%APEX%/f?p=109	56
57-VisualSMS	%DESKTOP%/S2mcWS/	57
58-WSAlmacen_y_OT	/WebServicesIsidora/	58, 59
60-SMS_Mensajeria	%DESKTOP%/S2mcWS/	60
61-MensajeriaManifiestos	/S2mcWS/[Grupo Componentes]	61
62-ListaPasajeros	ListaPasajerosWeb	62
63-PortalRayosX	portalEscanerWeb	63
64-ParDusGuia	/ParDusGuiaWeb/	64, 70, 71, 72, 73, 91, 96
66-SAM	samWeb	66
67-SeguridadWeb	/SeguridadComexWeb/	67
68-Interoperabilidad_MIC_Argen tina	/ServicioWebMicDta/ /WebServiceEnvioMicDta/	68
69-GestionDocumental	/sgdoc/	69
74-SistemaSelectividad	/SelectividadWeb/	74
75-SRS	/SRS/	75
76-ConsultaMICTransBolivia	%APEX%/f?p=200	76
87-GCP-F09	/ControlF09Web/	87
88-Programacion_Naviera	/RecepcionNavieraWeb/ /RecepcionNavieraMobile/	88
90-Acopio_Almacenistas	/AlmacenWeb/	90
93-PortalExportadorWeb	/PortalExportadorWeb/	93
97-SmdtWS	/SmdtWS/	44, 46, 48

3.1 Tabla Resumen Aplicaciones a Migrar

El siguiente recuadro contiene un resumen de las aplicaciones a migrar agrupadas por Framework,

Tecnología o Dependencias.

Framework/ Tecnología/ Dependencias	Versión	Lenguaje	IDE Desarrollo	Nº de aplicacio nes	Servidor de Despliegue
Apex	3.2.1	PL/SQL - HTML	Apex	6	Oracle 9 - Apex 3.2.1
BEA Workshop 8.1**	8.1.6	Java 1.4	Workshop 8.1	39	BEA Weblogic 8.1
Z3*	1	Java 1.4	Workshop 8.1	3	BEA Weblogic 8.1
Servlet	2.4	Java 1.5	Eclipse 3.7	1	Jboss 4.1
Seam	2	Java 1.5	Eclipse 3.7	1	Jboss 4.1
Struts 1.x / JBPM	1.x / 2	Java 1.4	Workshop 8.1	1	Jboss SOAP 5.3
JSF / Richface / JEE	1.2/3.3.3/5	Java 1.6	Eclipse 3.7	4	Jboss 5.2

El siguiente recuadro contiene un resumen de las aplicaciones que serán objeto de análisis para la formulación de la estrategia de migración de aplicaciones, agrupadas por: servidor de despliegue, tipo de aplicación y framework.

Servidor de Despliegue	Lenguaje	IDE Desarrollo	Nº Apps	Tipo Apps.			Frameworks	
				App Web	App WS	App Dsktp	*Z3:	**Work shop
Oracle 11g - Apex 3.2.1	PL/SQL - HTML	Apex	5	6	0	0	0	0
BEA Weblogic 8.1.6	Java 1.4	Workshop 8.1	39	22	20	6	5	23
IBM WS 4.0.3	Java 1.3	Eclipse 3.7	2	2	0	0	0	0
Tomcat 5 (5.0.28 y 5.5.7)	Java 1.4	Eclipse 3.7	4	2	2	0	0	0
JBoss AS 4.4.2	Java 1.5	Eclipse 3.7	1	1	0	0	0	0
JBoss SOA-P 5.3.0	Java 1.6	Eclipse 3.7	1	1	0	0	0	0
JBoss EAP 5.2	Java 1.6	Eclipse 3.7	4	4	0	0	0	0

***Z3: Framework desarrollado por Empresa Tuxpan**

****BEA Workshop 8.1: Framework creado por BEA Systema que luego fue cedido a la Fundación Apache con el nombre de Apache Beehive**

A continuación se muestra un gráfico con la agrupación de las aplicaciones core de negocio de acuerdo al servidor de aplicaciones que soporta su operación.

Aplicaciones CORE SNA (56)

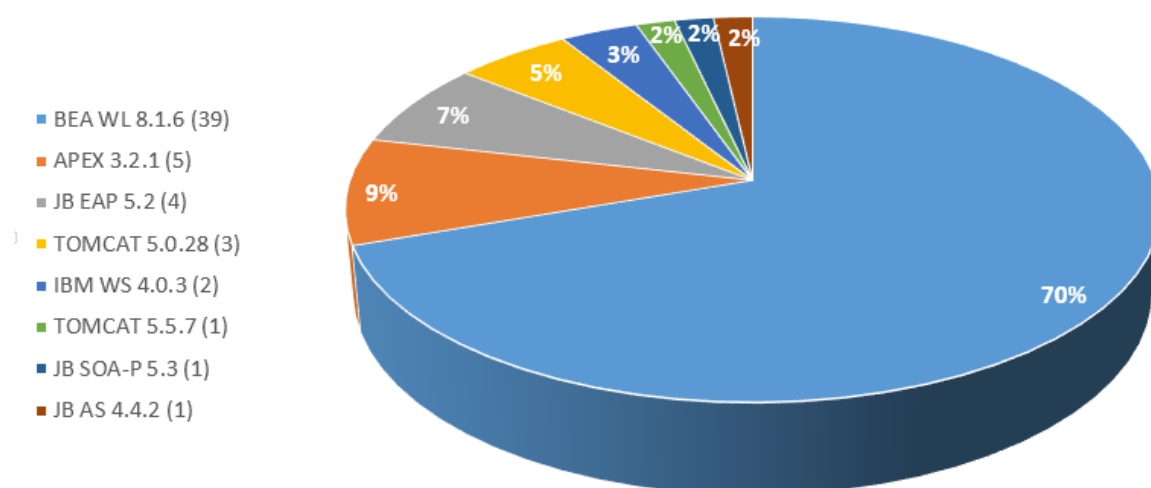


Diagrama 2: Aplicaciones CORE SNA

Aplicaciones / Servidor	# Aplicaciones
BEA WL 8.1.6	39
APEX 3.2.1	5
JB EAP 5.2	4
TOMCAT 5.0.28	3
IBM WS 4.0.3	2
TOMCAT 5.5.7	1
JB SOA-P 5.3	1
JB AS 4.4.2	1

4 Formulación de la Estrategia de Migración de Aplicaciones

La estrategia de migración de aplicaciones plantea un conjunto de acciones que son coherentes con el objetivo final de migrar todas las aplicaciones hacia la nueva plataforma de destino del SNA.

Estas acciones se enmarcarán dentro del contexto de los objetivos ya planteados, que en orden de ejecución son:

1. Hacer un análisis de restricciones, prioridades y condiciones que permitan decidir la estrategia más ajustada a estas necesidades. Este análisis se enfocará a establecer los criterios esenciales que guiarán la definición de la estrategia. (Ver: [Estrategia de Migración de Aplicaciones](#)). Hay que considerar las siguientes **restricciones**:
 1. **Abordar la condición de obsolescencia con mayor prioridad**: El salir de la actual obsolescencia es el factor más crítico a considerar.
 2. **Abordar la migración de aplicaciones para llegar a una Arquitectura de Software estándar**: Considerando la definición actual de la arquitectura de software se deben proponer mejoras para la definición de una arquitectura de software óptima para la operación de las aplicaciones en la plataforma de destino. (VER: [ANEXO 1: Visión de la Arquitectura de Software de las Aplicaciones](#))
2. Hacer un análisis cuantitativo y cualitativo que permita hacer una toma de decisión para determinar la prioridad de la migración de aplicaciones. Este análisis tiene como objetivo el generar indicadores a partir del levantamiento y análisis de brecha de las aplicaciones. (Ver: [Evaluar el estado actual de las aplicaciones](#))
 1. **Identificación general de aplicaciones**: A través de un catálogo se descubrirán los detalles de los recursos disponibles para cada aplicación. (Ver: [Catálogo de Aplicaciones](#))
 2. **Obtención de indicadores (Cuantitativos y Cualitativos)**: Producto del análisis de las aplicaciones se deben generar indicadores que permitan posteriormente ser parámetros clave para la toma de decisiones. (VER: [Estrategia de Migración de Aplicaciones](#) e [Indicadores de las aplicaciones](#))
3. Plantear una estrategia a seguir para poder abordar la migración de aplicaciones. Dado el volumen de aplicaciones a abordar, la diversidad de plataformas de origen y la criticidad del negocio se plantea como base, una planificación de ejecución de la estrategia en 2 fases:
 1. **Fase 1 “Migración de Aplicaciones de alta prioridad hacia la nueva plataforma”**: Esta fase consiste en ejecutar un proceso de migración de migraciones que apunte a migrar la mayor cantidad de aplicaciones prioritarias de manera coherente con los objetivos principales de la estrategia.
 2. **Fase 2 “Migración de Aplicaciones de baja prioridad y mejora de arquitectura de software de las aplicaciones existentes en la nueva plataforma”**: Esta fase consiste en migrar el resto de aplicaciones que no fueron migradas en la primera fase por razones de priorización y el aplicar mejora continua sobre las aplicaciones migradas para alcanzar la arquitectura de software ideal propuesta.



Diagrama 3: Formulación de estrategia de migración de aplicaciones

Considerando la diversidad de plataformas de origen de las aplicaciones que son objeto de este proceso de migración se espera que la estrategia genere como producto, patrones de acciones las cuales afecten a grupos de aplicaciones.

Los grupos de aplicaciones serán identificados de acuerdo a la siguiente estructura:

- **Plataforma Servidor Aplicaciones (Origen)**

- Weblogic
- JBoss AS/EAP
- JBoss SOA-P
- Tomcat
- IBM WebSphere
- APEX

- **Tipo Arquitectura Aplicación**

- Web
- Web Services
- Aplicaciones Escritorio
- Invocaciones a clases Java o Shell

- **Negocio de Aduana**

- Destinación Impo.
- Destinación Expo.
- Infraccionario
- Consultas Generales
- Franquicias
- Manifiesto

- Régimen Suspensivo
- Arancel Aduanero
- Administrativa y Gestión
- Gestión Configuración Sistemas
- Mensajería Interoperabilidad
- Fiscalización
- Vehículos
- Reintegro

Luego de identificar esta agrupación será necesario generar indicadores que permitan obtener una estimación en HH para el esfuerzo de migración de cada aplicación.

Los indicadores a generar son los siguientes:

- **Indicadores producto del levantamiento:**
 - Compleitud Documental: Cuanta documentación mínima requerida tiene la aplicación.
 - Complejidad de Migración: Cual es el grado de complejidad técnica de la aplicación.
- **Indicadores producto del Análisis de Brecha:**
 - Riesgo de Migración: De acuerdo a la completitud documental + complejidad técnica. Cuál es el factor de riesgo de migración de la aplicación (Porcentaje)
 - Estimación de esfuerzo de Migración en HH: De acuerdo a los indicadores anteriores + juicio experto se genera una estimación de HH para migrar la aplicación.

5 Escenarios de Migración

En esta etapa se identifican los distintos escenarios de migración en base al Ranking de Importancia de Migración de cada aplicación (ver [Análisis 80/20](#)):

Escenario 1: **20% de las aplicaciones priorizas** por el Ranking de Importancia de Migración que, por el Principio de Pareto, son las que darán mayor satisfacción al SNA. Corresponde a las de Ranking de Importancia de Migración menor 1-11 (inclusives).

Escenario 2: Aplicaciones **poco complejas y/o alta-media importancia para el negocio**. Ranking de importancia de migración entre 12-24 (inclusives).

Escenario 3: Aplicaciones de **baja-media complejidad y/o media-baja importancia para el negocio**. Ranking de importancia de migración entre 25-35 (inclusives).

Escenario 4: Aplicaciones de **media-alta complejidad y/o media-baja importancia para el negocio**. Ranking de importancia de migración entre 36-42 (inclusives).

Escenario 5: Aplicaciones de **alta complejidad y/o media-baja importancia para el negocio**. Ranking de importancia de migración entre 43-46 (inclusives).

Escenario 6: **Recodificación, complejidad alta y/o poca importancia para negocio**: Todas aquellas aplicaciones cuyo Ranking de Importancia de Migración esté entre 47-56 (inclusives).

Producto del análisis de las aplicaciones es posible que se puedan producir ajustes acerca del uso de los escenarios planteados.

6 Estrategia de Migración de Aplicaciones.

La estrategia de migración para las aplicaciones del Servicio Nacional de Aduanas está planteada en las siguientes fases:

- 1 Evaluar el estado actual de las aplicaciones e identificación de las aplicaciones (Plataforma, componentes)
- 2 Actualizar el catálogo de Aplicaciones para tener una vista de Negocio y Técnico.
- 3 Generar indicadores por cada aplicación
 - 3.1 Tamaño (Líneas de código)
 - 3.2 Completitud Documental
 - 3.3 Complejidad Técnica
 - 3.4 % Riesgo de Migración
 - 3.5 Estimación HH necesarias para Migración.
- 4 Identificar escenarios de migración
- 5 Agrupar las aplicaciones de acuerdo a los escenarios definidos.
- 6 Detallar un plan técnico de migración de aplicaciones para cada grupo de aplicaciones.
- 7 Ejecución del plan de migración para cada uno de los escenarios.
- 8 Validación de la migración de las aplicaciones.
- 9 Mejora Continua.

6.1.1 Diagrama de Estrategia de Migración

El siguiente diagrama representa el proceso de migración de las aplicaciones, desde la arquitectura y documentación disponible, pasando por las distintas fases para migrar, hasta obtener una aplicación migrada en la arquitectura JBoss EAP 6.4 y Java 7.

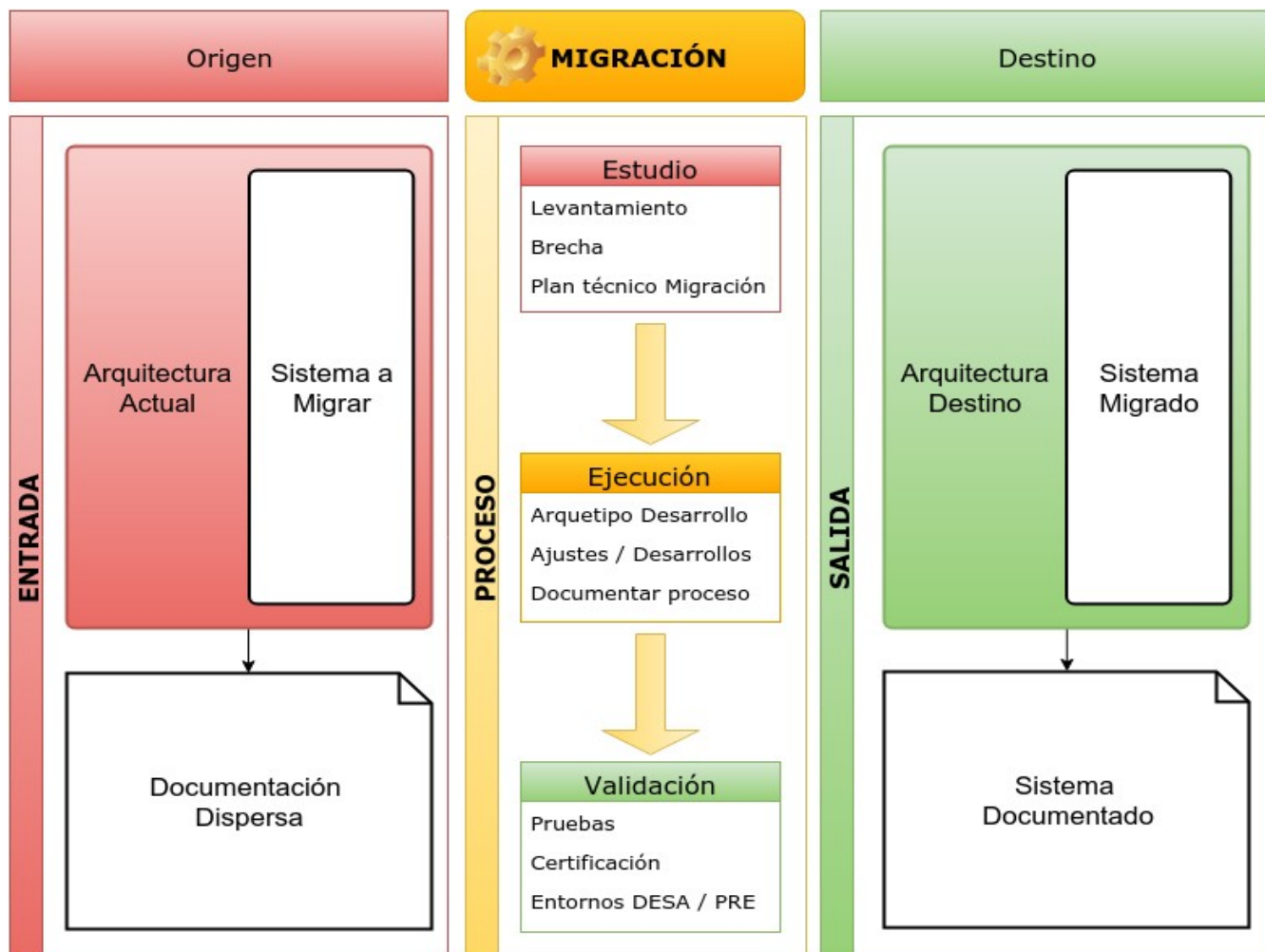


Diagrama 4: Proceso de Migración

7 Evaluar el estado actual de las aplicaciones

En esta fase se evalúan los sistemas a migrar, su arquitectura, documentación, recopilación de conocimientos sobre el sistema, riesgos, especificaciones tecnológicas de migración, etc. a partir de los documentos Levantamiento de Información y Análisis de Brecha que existen por cada aplicación.

7.1 Levantamiento

El origen del estudio es una **ficha de la aplicación elaborada por SNA**

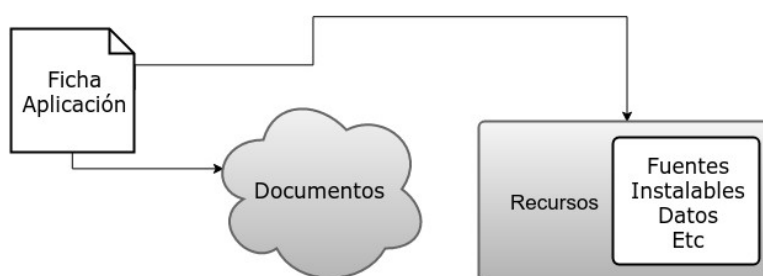


Diagrama 5: Levantamiento Situación Actual

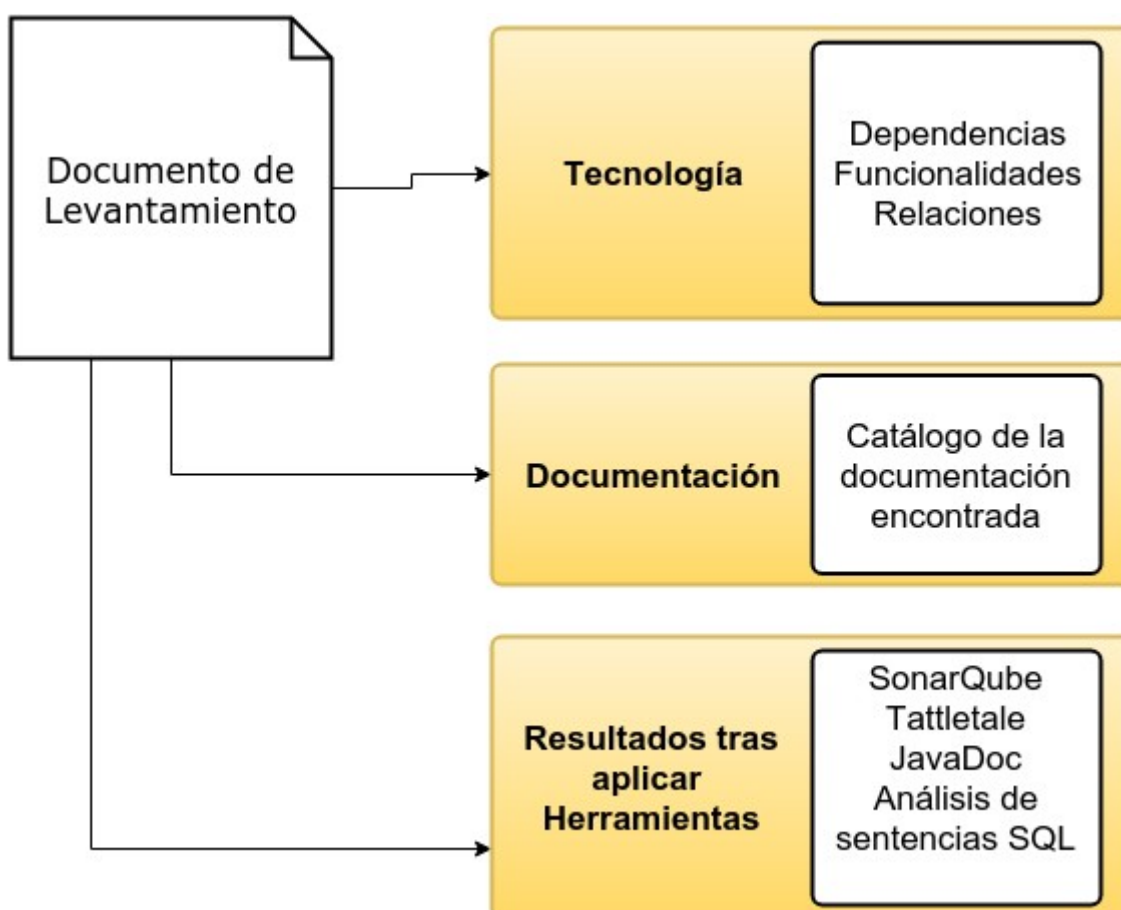


Diagrama 6: Levantamiento Situación Actual

La información es reorganizada y completada a más bajo nivel en un **documento de Levantamiento**.

7.2 Análisis de Brecha

El análisis de brecha permite detectar los riesgos a tener en cuenta en la migración del sistema, ya sea por falta de documentación, por dependencias tecnológicas arraigadas en la plataforma de origen, etc, además de efectuar una ponderación basada en los resultados del análisis.

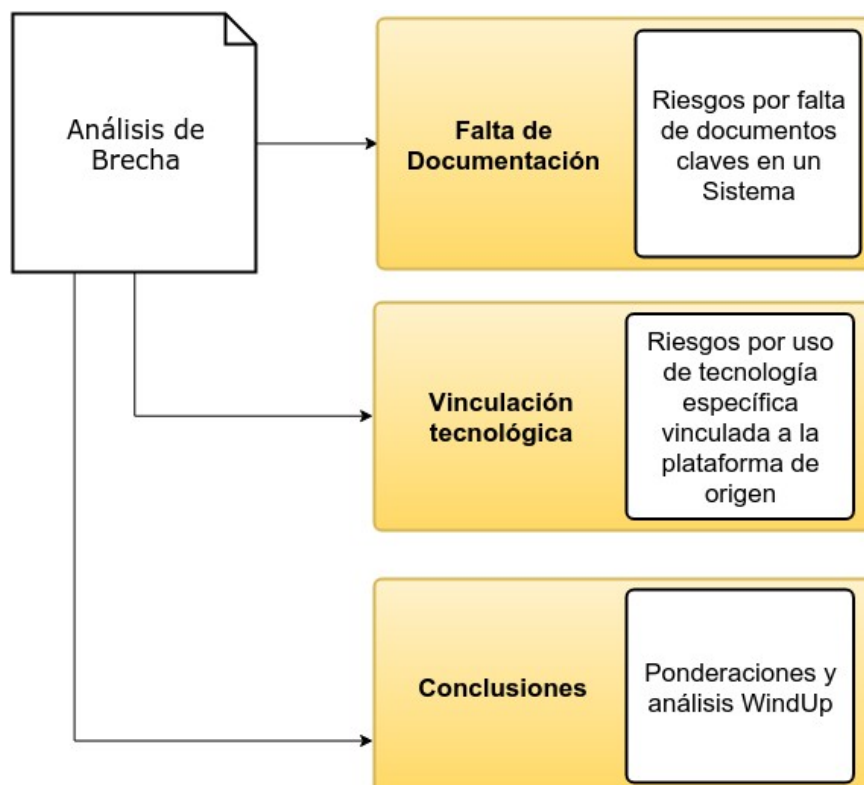


Diagrama 7: Análisis de Brecha

7.3 Riesgos específicos

A continuación se listan los riesgos más importantes encontrados en el análisis de brecha de cada aplicación:

Aplicación	Riesgos específicos
01-SIGES	<ol style="list-style-type: none"> 1. Las capas usan estándares básicos (HTML + PL/SQL + SQL) que corren sobre la propia base de datos Oracle, es decir, se podría seguir utilizando, mientras se migra, el sistema actual en la nueva plataforma ya que no se va a cambiar la tecnología de BBDD. 2. La complejidad viene determinada por el poco grado de reutilización posible, puesto que la lógica de negocio está encapsulada en PL/SQL soportados por Oracle que deberán ser traducidos en objetos soportados por la tecnología de destino (como objetos Domain y Daos, capas de persistencia, Beans de negocio, etc).

Aplicación	Riesgos específicos
02-OPVIG	<ol style="list-style-type: none"> 1. Las capas usan estándares básicos (HTML + PL/SQL + SQL) que corren sobre la propia base de datos Oracle, es decir, se podría seguir utilizando, mientras se migra, el sistema actual en la nueva plataforma ya que no se va a cambiar la tecnología de BBDD. 2. La complejidad viene determinada por el poco grado de reutilización posible, puesto que la lógica de negocio está encapsulada en PL/SQL soportados por Oracle que deberán ser traducidos en objetos soportados por la tecnología de destino (como objetos Domain y Daos, capas de persistencia, Beans de negocio, etc). 3. Se une la aplicación 3_Consultas Operadores Vigentes.
04-Veh_Integrado	<ol style="list-style-type: none"> 1. Las dependencias que requieren de clases BEA Systems deberán ser sustituidas por otras libres de dependencias BEA o eliminar estas dependencias BEA de ellas cuando sean librerías generadas por SNA. 2. JSP + Servlet es una tecnología portable a la arquitectura de destino. Sólo en caso de algunos paquetes utilizados por las JSP podría necesitarse algunos ajustes. 3. Al usar clientes SOAP hay que considerar que, si los Servicios consumidos son migrados a JAXWS, los clientes deberán ser regenerados para adaptarse a esta tecnología.
05-SWVehiculos	<ol style="list-style-type: none"> 1. Las dependencias que requieren de clases BEA Systems deberán ser sustituidas por otras libres de dependencias BEA o eliminar estas dependencias BEA de ellas cuando sean librerías generadas por SNA. 2. El uso de controles netui es una fuerte vinculación con BEA. 3. Al usar clientes SOAP hay que considerar que, si los Servicios consumidos son migrados a JAXWS, los clientes deberán ser regenerados para adaptarse a esta tecnología. 4. Los servicios provistos están construidos bajo la tecnología JWS que tiene una fuerte dependencia de Bea Systems y que se tendrá que eliminar por ejemplo, con el estándar JAX-WS.
06-VehiculosRACWeb	<ol style="list-style-type: none"> 1. Las dependencias que requieren de clases BEA Systems deberán ser sustituidas por otras libres de dependencias BEA o eliminar estas dependencias BEA de ellas cuando sean librerías generadas por SNA. 2. El uso de controles netui es una fuerte vinculación con BEA. 3. Al usar clientes SOAP hay que considerar que, si los Servicios consumidos son migrados a JAXWS, los clientes deberán ser regenerados para adaptarse a esta tecnología. 4. Se une la aplicación 7_Rent a Car
08-SCVM	<ol style="list-style-type: none"> 1. No existen dependencias de Bea Systems tanto en el código fuente como en dependencias de librerías externas. 2. Contiene librerías estándares que no presentan riesgo para la migración. 3. Al usar clientes SOAP hay que considerar que, si los Servicios consumidos son migrados a JAXWS, los clientes deberán ser regenerados para adaptarse a esta tecnología.
09-VehiculosFronteraWeb	<ol style="list-style-type: none"> 1. Las dependencias que requieren de clases BEA Systems deberán ser sustituidas por otras libres de dependencias BEA o eliminar estas dependencias BEA de ellas cuando sean librerías generadas por SNA. 2. El uso de controles netui es una fuerte vinculación con BEA. 3. Al usar clientes SOAP hay que considerar que, si los Servicios consumidos son migrados a JAXWS, los clientes deberán ser regenerados para adaptarse a esta tecnología.
10-Control_Mensajeria	<ol style="list-style-type: none"> 1. Eliminar dependencias de Bea Systems en las librerías que usa la aplicación. Utiliza 4 librerías del SNA que tienen dependencias de BEA Systems. 2. Al usar clientes SOAP hay que considerar que, si los Servicios consumidos son migrados a JAXWS, los clientes deberán ser regenerados para adaptarse a esta tecnología. 3. Es una aplicación que engloba a 21 módulos todos ellos con servicios provistos y consumidos (contexto ventanillaprueba)

Aplicación	Riesgos específicos
21-DECARE	<ol style="list-style-type: none"> 1. No se han detectado dependencias que requieran de clases BEA Systems, lo que facilitará el trabajo de migración. 2. Se recomienda actualizar la versión de Struts. 3. Aunque el motor jBPM está soportado en la plataforma de destino, pueden necesitarse ajustes relacionados con los mapeos de configuración y con las propiedades JPA, cambios que pueden ser complejos puesto que la especificación JPA 2.0 de la plataforma de destino contiene importantes cambios con respecto a sus versiones anteriores
22-SICOMEXIN	<ol style="list-style-type: none"> 1. Las dependencias que requieren de clases BEA Systems deberán ser sustituidas por otras libres de dependencias BEA o eliminar estas dependencias BEA de ellas cuando sean librerías generadas por SNA. 2. El uso de controles netui es una fuerte vinculación con BEA. 3. Al usar clientes SOAP hay que considerar que, si los Servicios consumidos son migrados a JAXWS, los clientes deberán ser regenerados para adaptarse a esta tecnología.
23-DIPS_COURIER	<ol style="list-style-type: none"> 1. Las dependencias que requieren de clases BEA Systems deberán ser sustituidas por otras libres de dependencias BEA o eliminar estas dependencias BEA de ellas cuando sean librerías generadas por SNA. Utiliza 7 librerías del SNA que tienen dependencias de BEA Systems. 2. El uso de controles netui es una fuerte vinculación con BEA. 3. Algunos de los Web Services Provistos están contruidos bajo la tecnología JWS de Bea Systems por lo que existe complejidad para reemplazarlos por algún estándar como JAX-WS.
24-F18_POSTAL	<ol style="list-style-type: none"> 1. No existen librerías incompatibles con la plataforma de destino. 2. La aplicación tiene 23 fuentes java por lo que su reducido tamaño facilita la migración. 3. Tanto en el levantamiento como en el análisis de brecha no se disponía de código fuente, se ha extraído el JAR mediante la herramienta JD-GUI 1.4.0. Esto podría ser una dificultad para efectuar la migración. 4. Los Web Services Provistos son expuestos a través del componente "Apache SOAP 2.3" [https://archive.apache.org/dist/ws/soap/version-2.3/]. Habrá que implementar la interfaz de exposición de estos servicios y plantear una estrategia de proxy a través de un ESB para mantener la compatibilidad con la forma de la mensajería actual.
26-Mensajeria_DIN	<ol style="list-style-type: none"> 1. No se encuentran librerías externas o del SNA incompatibles con la arquitectura de destino. 2. Al usar clientes SOAP hay que considerar que, si los Servicios consumidos son migrados a JAXWS, los clientes deberán ser regenerados para adaptarse a esta tecnología.
27-DepositoFranco-Intranet	<ol style="list-style-type: none"> 1. JSP + Servlet es una tecnología portable a la arquitectura de destino. Sólo en caso de algunos paquetes utilizados por las JSP podría necesitarse algunos ajustes. 2. El JDK utilizado para el desarrollo es muy antiguo y deberán realizarse ajustes para Java 1.7. 3. La falta de documentación y la no localización de las fuentes pueden ser, más que la complejidad del negocio en si, los mayores obstáculos a la hora de efectuar la migración. 4. Se une la aplicación 28_DepositoFrancoInternet
29-Consultas_DUS	<ol style="list-style-type: none"> 1. Las dependencias que requieren de clases BEA Systems deberán ser sustituidas por otras libres de dependencias BEA o eliminar estas dependencias BEA de ellas cuando sean librerías generadas por SNA. Utiliza 7 librerías del SNA que tienen dependencias de BEA Systems. 2. El uso de controles netui es una fuerte vinculación con BEA. 3. Al usar clientes SOAP hay que considerar que, si los Servicios consumidos son migrados a JAXWS, los clientes deberán ser regenerados para adaptarse a esta tecnología.

Aplicación	Riesgos específicos
33-DIPSCF	<ol style="list-style-type: none"> 1. Aunque existe poca documentación no se han encontrado dependencias de las librerías dependientes de Bea Systems en el módulo web que es al que pertenece el código fuente analizado. 2. No se encuentran librerías externas o de terceros que presenten incompatibilidad. 3. Se encuentran estas librerías PasswordEncryptCommon.jar y SecurityFramework.jar que no se consideran librerías estándar sino que parecen algún desarrollo propio y aunque en principio no parecen tener incompatibilidad habría que tener especial cuidado a la hora de la migración. 4. En el informe Tattleate se ha detectado que la librería ClienteTGRCID.jar (perteneciente a un desarrollo propio) contiene dependencias de Bea Systems por lo que habría que eliminar las dependencias de este cliente para que pueda ser usado dentro de la aplicación. Al usar clientes SOAP hay que considerar que, si los Servicios consumidos son migrados a JAX-WS, los clientes deberán ser regenerados para adaptarse a esta tecnología. 5. En el informe Tattleate no refleja dependencias de Bea Systems en el cliente DipscfEJB.jar y su código fuente no presenta problemas de migración.
34-DIPS_Viajeros	<ol style="list-style-type: none"> 1. Las dependencias que requieren de clases BEA Systems deberán ser sustituidas por otras libres de dependencias BEA o eliminar estas dependencias BEA de ellas cuando sean librerías generadas por SNA. 2. El uso de controles netui es una fuerte vinculación con BEA. 3. Al usar clientes SOAP hay que considerar que, si los Servicios consumidos son migrados a JAXWS, los clientes deberán ser regenerados para adaptarse a esta tecnología. 4. El uso de Flex y Flex Messaging (al ser una tecnología fuera de tendencia) puede hacer que la migración sea más compleja aunque no se considera una tecnología incompatible con la arquitectura. 5. Flex utiliza la API Java Message Service (JMS) para el envío de mensajes entre dos clientes, esto no se considera un riesgo ya que es compatible con la plataforma de destino.
35 DTI	Implica la migración de cliente WS
36 Esquema de Seguridad	
37-Web_AdmPersonas	<ol style="list-style-type: none"> 1. JSP es una tecnología portable a la arquitectura de destino. Sólo en caso de algunos paquetes utilizados por las JSP podría necesitarse algunos ajustes. 2. Este sistema cuenta con múltiples elementos (utilTuxpan.jar, Z3.jar, WebComponents.jar, etc) cuyo comportamiento en la plataforma de destino no puede ser evaluado de forma estándar, al ser componentes de desarrollos a medidas de terceras empresas. Estos componentes pueden suponer un escollo importante y se recomienda se enfoquen los esfuerzos a su sustitución más que su adaptación
38-WSAA_con_Argentina	<ol style="list-style-type: none"> 1. Las dependencias que requieren de clases BEA Systems deberán ser sustituidas por otras libres de dependencias BEA o eliminar estas dependencias BEA de ellas cuando sean librerías generadas por SNA. 2. El uso de controles netui es una fuerte vinculación con BEA. 3. Los servicios provistos por el sistema deberán ajustarse a JAX-WS, lo que conllevará también ajustes en los clientes que, actualmente, hagan uso de estos servicios.
39-WSAA_Chile	<ol style="list-style-type: none"> 1. Las dependencias que requieren de clases BEA Systems deberán ser sustituidas por otras libres de dependencias BEA o eliminar estas dependencias BEA de ellas cuando sean librerías generadas por SNA. 2. El uso de controles netui es una fuerte vinculación con BEA. 3. Los servicios provistos por el sistema deberán ajustarse a JAX-WS, lo que conllevará también ajustes en los clientes que, actualmente, hagan uso de estos servicios.

Aplicación	Riesgos específicos
40-INDIRA	<ol style="list-style-type: none"> 1. Las dependencias que requieren de clases BEA Systems deberán ser sustituidas por otras libres de dependencias BEA o eliminar estas dependencias BEA de ellas cuando sean librerías generadas por SNA. 2. Existen dos librerías del SNA que contienen dependencias de Bea Systems: ControlesINDIRA.jar y EsquemaINDIRA.jar. 3. El uso de controles netui es una fuerte vinculación con BEA. 4. Los servicios provistos están contruidos bajo la tecnología JWS que tiene una fuerte dependencia de Bea Systems y que se tendrá que eliminar por ejemplo, con el estándar JAX-WS. 5. Al usar clientes SOAP hay que considerar que, si los Servicios consumidos son migrados a JAXWS, los clientes deberán ser regenerados para adaptarse a esta tecnología.
42-Firmas_ALAD	<ol style="list-style-type: none"> 1. No se ha encontrado ninguna librería que sea incompatible con la arquitectura destino. 2. La versión de Java es 1.6 por lo que existe poca brecha para migrar a 1.7.
43-TramitacionIVV	<ol style="list-style-type: none"> 1. JSP + Servlet es una tecnología portable a la arquitectura de destino. 2. El JDK utilizado para el desarrollo es muy antiguo y deberán realizarse ajustes para Java 1.7. 3. La falta de documentación y la no localización de las fuentes pueden ser, más que la complejidad del negocio en sí, los mayores obstáculos a la hora de efectuar la migración.
44-SDA	<ol style="list-style-type: none"> 1. La ponderación se realiza sobre el supuesto de migrar el cliente SWT a Aplicación Web pero seguir utilizando el servicio SmdtWS como encapsulador de la lógica de negocio y manteniendo la no persistencia de datos más allá de los ficheros XML de intercambio que deberían almacenarse.
45-SVC	<ol style="list-style-type: none"> 1. Las dependencias que requieren de clases BEA Systems deberán ser sustituidas por otras libres de dependencias BEA o eliminar estas dependencias BEA de ellas cuando sean librerías generadas por SNA. 2. El uso de controles netui es una fuerte vinculación con BEA.
46-SIDECO	<ol style="list-style-type: none"> 1. La ponderación se realiza sobre el supuesto de migrar el cliente SWT a Aplicación Web pero seguir utilizando el servicio SmdtWS como encapsulador de la lógica de negocio y manteniendo la no persistencia de datos más allá de los ficheros XML de intercambio que deberían almacenarse.
47_DepWebFiscalizacion	<ol style="list-style-type: none"> 1. JSP es una tecnología portable a la arquitectura de destino. Sólo en caso de algunos paquetes utilizados por las JSP podría necesitarse algunos ajustes. 2. Este sistema cuenta con múltiples elementos (utilTuxpan.jar, Z3.jar, WebComponents.jar, etc) cuyo comportamiento en la plataforma de destino no puede ser evaluado de forma estándar, al ser componentes de desarrollos a medidas de terceras empresas. Estos componentes pueden suponer un escollo importante y se recomienda se enfoquen los esfuerzos a su sustitución más que su Adaptación.
48_SIFER	<ol style="list-style-type: none"> 1. La ponderación se realiza sobre el supuesto de migrar el cliente SWT a Aplicación Web pero seguir utilizando el servicio SmdtWS como encapsulador de la lógica de negocio y manteniendo la no persistencia de datos más allá de los ficheros XML de intercambio que deberían almacenarse.
49_SIROFE	<ol style="list-style-type: none"> 1. Las capas usan estándares básicos (HTML + PL/SQL + SQL) que corren sobre la propia base de datos Oracle, es decir, se podría seguir utilizando, mientras se migra, el sistema actual en la nueva plataforma ya que no se va a cambiar la tecnología de BBDD. 2. La complejidad viene determinada por el poco grado de reutilización posible, puesto que la lógica de negocio está encapsulada en PL/SQL soportados por Oracle que deberán ser traducidos en objetos soportados por la tecnología de destino (como objetos Domain y Daos, capas de persistencia, Beans de negocio, etc).
50 SIDEMAR	<p>Implica una codificación completa</p>

Aplicación	Riesgos específicos
51-Servicio_Web_Canje_Maritimo	<ol style="list-style-type: none"> 1. Las dependencias que requieren de clases BEA Systems deberán ser sustituidas por otras libres de dependencias BEA o eliminar estas dependencias BEA de ellas cuando sean librerías generadas por SNA. 2. El uso de controles netui es una fuerte vinculación con BEA. 3. Los servicios provistos por el sistema deberán ajustarse a JAX-WS, lo que conllevará también ajustes en los clientes que, actualmente, hagan uso de estos servicios.
52-Aplicacion_Web_Canje_Maritim o	<ol style="list-style-type: none"> 1. Con la información ofrecida en el informe de levantamiento la aplicación es fácil de migrar ya que no tiene dependencias de Bea Systems. 2. Requiere el framework Z3 por lo que depende de la migración de este framework para su correcto Funcionamiento.
54-WSAlmacen	<ol style="list-style-type: none"> 1. Las dependencias que requieren de clases BEA Systems deberán ser sustituidas por otras libres de dependencias BEA o eliminar estas dependencias BEA de ellas cuando sean librerías generadas por SNA. 2. El uso de controles netui es una fuerte vinculación con BEA. 3. Al usar clientes SOAP hay que considerar que, si los Servicios consumidos son migrados a JAXWS, los clientes deberán ser regenerados para adaptarse a esta tecnología.
55-MicWeb	<ol style="list-style-type: none"> 1. JSP + Servlet es una tecnología portable a la arquitectura de destino. Sólo en caso de algunos paquetes utilizados por las JSP podría necesitarse algunos ajustes. 2. Este sistema cuenta con múltiples elementos (utilTuxpan.jar, Z3.jar, WebComponents.jar, etc) cuyo comportamiento en la plataforma de destino no puede ser evaluado de forma estándar, al ser componentes de desarrollos a medidas de terceras empresas. Estos componentes pueden suponer un escoyo importante y se recomienda se enfoquen los esfuerzos a su sustitución más que su adaptación.
56_SIROTE	<ol style="list-style-type: none"> 1. Las capas usan estándares básicos (HTML + PL/SQL + SQL) que corren sobre la propia base de datos Oracle, es decir, se podría seguir utilizando, mientras se migra, el sistema actual en la nueva plataforma ya que no se va a cambiar la tecnología de BBDD. 2. La complejidad viene determinada por el poco grado de reutilización posible, puesto que la lógica de negocio está encapsulada en PL/SQL soportados por Oracle que deberán ser traducidos en objetos soportados por la tecnología de destino (como objetos Domain y Daos, capas de persistencia, Beans de negocio, etc).
57-VisualSMS	<ol style="list-style-type: none"> 1. La ponderación se realiza sobre el supuesto de migrar el cliente SWT a Aplicación Web pero seguir utilizando el servicio S2MC como encapsulador de la lógica de negocio y manteniendo la no persistencia de datos más allá de los ficheros XML de intercambio que deberían almacenarse.
58-WSAlmacen_y_OT	<ol style="list-style-type: none"> 1. Las dependencias que requieren de clases BEA Systems deberán ser sustituidas por otras libres de dependencias BEA o eliminar estas dependencias BEA de ellas cuando sean librerías generadas por SNA. 2. El uso de controles netui es una fuerte vinculación con BEA. No obstante, no aplica a la lógica de negocio en sí del WS sino a las páginas que lo consumen y que van en el propio EAR. 3. Si los Servicios consumidos son migrados a JAX-WS, los clientes deberán ser regenerados para adaptarse a esta tecnología. 4. Este sistema cuenta con múltiples elementos (utilTuxpan.jar, Z3.jar, etc) cuyo comportamiento en la plataforma de destino no puede ser evaluado de forma estándar, al ser componentes de desarrollos a medidas de terceras empresas. Estos componentes pueden suponer un escoyo importante y se recomienda se enfoquen los esfuerzos a su sustitución más que su adaptación. 5. Se incluye la aplicación 59 Servicio Web Documentos Tramitados.
60-SMS_Mensajería	<ol style="list-style-type: none"> 1. La ponderación se realiza sobre el supuesto de migrar el cliente SWT a Aplicación Web pero seguir utilizando el servicio S2MCWS como encapsulador de la lógica de negocio y manteniendo la no persistencia de datos más allá de los ficheros XML de intercambio que deberían almacenarse.
61 Sistema Mensajería Manifiestos Electrónicos	La mayor parte del código es reutilizable / No hay documentación

Aplicación	Riesgos específicos
62-ListaPasajeros	<ol style="list-style-type: none"> 1. No se ha encontrado ninguna librería que sea incompatible con la arquitectura destino. 2. La arquitectura de origen es JBoss EAP 5.2 por lo que la migración a JBoss EAP 6.4 implica pocos riesgos. 3. La versión de Java es 1.6 por lo que existe poca brecha para migrar a 1.7. 4. Al usar clientes SOAP hay que considerar que, si los Servicios consumidos son migrados a JAXWS, los clientes deberán ser regenerados para adaptarse a esta tecnología.
63-PortalRayosX	<ol style="list-style-type: none"> 1. No se ha encontrado ninguna librería que sea incompatible con la arquitectura destino. 2. La versión de Java es 1.6 por lo que existe poca brecha para migrar a 1.7.
64-ParDusGuia	<ol style="list-style-type: none"> 1. Las dependencias que requieren de clases BEA Systems deberán ser sustituidas por otras libres de dependencias BEA o eliminar estas dependencias BEA de ellas cuando sean librerías generadas por SNA. Utiliza 7 librerías del SNA que tienen dependencias de BEA Systems. 2. El uso de controles netui es una fuerte vinculación con BEA. 3. Al usar clientes SOAP hay que considerar que, si los Servicios consumidos son migrados a JAXWS, los clientes deberán ser regenerados para adaptarse a esta tecnología. 4. Los servicios provistos están contruidos bajo la tecnología JWS que tiene una fuerte dependencia de Bea Systems y que se tendrá que eliminar por ejemplo, con el estándar JAX-WS. 5. Existen 827 clases pero sin fuentes java identificados. Existen varias libs que no disponen de fuentes. Lo más probable es que los class que están desplegados se tengan que migrar decompilandolos. Esto añade más complejidad a la aplicación.
66-SAM	<ol style="list-style-type: none"> 1. No se ha encontrado ninguna librería que sea incompatible con la arquitectura destino. 2. La arquitectura de origen es JBoss EAP 5.2 por lo que la migración a JBoss EAP 6.4 implica pocos riesgos. 3. La versión de Java es 1.6 por lo que existe poca brecha para migrar a 1.7. 4. Al usar clientes SOAP hay que considerar que, si los Servicios consumidos son migrados a JAXWS, los clientes deberán ser regenerados para adaptarse a esta tecnología.
67-SeguridadWeb	<ol style="list-style-type: none"> 1. Las dependencias que requieren de clases BEA Systems deberán ser sustituidas por otras libres de dependencias BEA o eliminar estas dependencias BEA de ellas cuando sean librerías generadas por SNA. 2. El uso de controles netui es una fuerte vinculación con BEA. 3. La falta total de documentación supondrá un plus de complejidad.
68-Interoperabilidad_MIC_Argentina	<ol style="list-style-type: none"> 1. Las dependencias que requieren de clases BEA Systems deberán ser sustituidas por otras libres de dependencias BEA o eliminar estas dependencias BEA de ellas cuando sean librerías generadas por SNA. 2. El uso de controles netui es una fuerte vinculación con BEA. 3. Los servios provistos por el sistema deberán ajustarse a JAX-WS, lo que conllevará también ajustes en los clientes que, actualmente, hagan uso de estos servicios.
69-GestionDocumental	<ol style="list-style-type: none"> 1. No se ha encontrado ninguna librería que sea incompatible con la arquitectura destino. 2. La arquitectura de origen es JBoss EAP 5.2 por lo que la migración a JBoss EAP 6.4 implica pocos riesgos. 3. La versión de Java es 1.5 por lo que existe poca brecha para migrar a 1.7. 4. Al usar clientes SOAP hay que considerar que, si los Servicios consumidos son migrados a JAXWS, los clientes deberán ser regenerados para adaptarse a esta tecnología. 5. Aunque a priori puede parecer que no haya grandes dificultades para realizar la migración, la aplicación contiene muchas funcionalidades y 631 fuentes .java, por lo que puede ser laborioso testear todas las funcionalidades.
74-SistemaSelectividad	<ol style="list-style-type: none"> 1. No se han detectado dependencias que requieran de clases BEA Systems, lo que facilitará el trabajo de migración. 2. Se recomienda actualizar la versión de Struts. 3. El servicio que ofrece la aplicación así como los clientes que utiliza, deberán ajustarse a la especificación JAX-RPC.

Aplicación	Riesgos específicos
75-SRS	<ol style="list-style-type: none"> 1. Las dependencias que requieren de clases BEA Systems deberán ser sustituidas por otras libres de dependencias BEA o eliminar estas dependencias BEA de ellas cuando sean librerías generadas por SNA. 2. Se recomienda actualizar la versión de Struts. 3. La capa de persistencia iBatis puede requerir algunos reajustes en los ficheros de mapeos o Configuración.
76-ConsultaMICTransBolivia	<ol style="list-style-type: none"> 1. Las capas usan estándares básicos (HTML + PL/SQL + SQL) que corren sobre la propia base de datos Oracle, es decir, se podría seguir utilizando el sistema actual en la nueva plataforma ya que no se va a cambiar la tecnología de BBDD. 2. La complejidad viene determinada por el poco grado de reutilización posible, puesto que la lógica de negocio está encapsulada en PL/SQL soportados por Oracle que deberán ser traducidos en objetos soportados por la tecnología de destino (como objetos Domain y Daos, capas de persistencia, Beans de negocio, etc).
87-GCP-F09	<ol style="list-style-type: none"> 1. Las dependencias que requieren de clases BEA Systems deberán ser sustituidas por otras libres de dependencias BEA o eliminar estas dependencias BEA de ellas cuando sean librerías generadas por SNA. 2. El uso de controles netui es una fuerte vinculación con BEA. 3. Al usar clientes SOAP hay que considerar que, si los Servicios consumidos son migrados a JAXWS, los clientes deberán ser regenerados para adaptarse a esta tecnología.
88-Programacion_Naviera	<ol style="list-style-type: none"> 1. Las dependencias que requieren de clases BEA Systems deberán ser sustituidas por otras libres de dependencias BEA o eliminar estas dependencias BEA de ellas cuando sean librerías generadas por SNA. 2. El uso de controles netui es una fuerte vinculación con BEA.
90-Acopio_Almacenistas	<ol style="list-style-type: none"> 1. Las dependencias que requieren de clases BEA Systems deberán ser sustituidas por otras libres de dependencias BEA o eliminar estas dependencias BEA de ellas cuando sean librerías generadas por SNA. 2. El uso de controles netui es una fuerte vinculación con BEA. 3. El servicio que ofrece la aplicación deberá ajustarse a la especificación JAX-RPC.
93-PortalExportadorWeb	<ol style="list-style-type: none"> 1. Las dependencias que requieren de clases BEA Systems deberán ser sustituidas por otras libres de dependencias BEA o eliminar estas dependencias BEA de ellas cuando sean librerías generadas por SNA. 2. El uso de controles netui es una fuerte vinculación con BEA. 3. Existen 15 librerías propias del SNA dependientes de BEA Systems que deberán ser tratadas en primer lugar para que la aplicación funcione correctamente una vez migrada.
97-SmdtWS	<ol style="list-style-type: none"> 1. Las dependencias que requieren de clases BEA Systems deberán ser sustituidas por otras libres de dependencias BEA o eliminar estas dependencias BEA de ellas cuando sean librerías generadas por SNA. 2. El uso de controles netui es una fuerte vinculación con BEA. 3. El servicio que ofrece la aplicación así como los clientes que utiliza, deberán ajustarse a la especificación JAX-RPC. 4. Este sistema cuenta con elementos como Z3.jar, cuyo comportamiento en la plataforma de destino no puede ser evaluado de forma estándar, al ser componentes de desarrollos a medidas de terceras empresas. Estos componentes pueden suponer un escollo importante y se recomienda se enfoquen los esfuerzos a su sustitución más que su adaptación.

8 Catálogo de Aplicaciones

En esta fase se estudia el documento anexo **CMAA-Estrategia-APPS-NEGOCIO.xlsx**, donde se han identificado los grupos a los que corresponde cada aplicación (a nivel de negocio y a nivel tecnológico). Esto proporciona una visión general de las aplicaciones según la tecnología y el negocio al que pertenecen, que ayudará a la toma de decisiones para ejecutar los planes de migración de una línea concreta de negocio, y a la vez dentro de un grupo tecnológico.

9 Indicadores de las aplicaciones

En esta fase se hace un estudio de los indicadores que se han generado en cada cada aplicación. Esto proporciona una visión global del riesgo de migración asumido en cada aplicación y su esfuerzo en HH.

9.1 Riesgo de Migración (RM)

Los indicadores Completitud Documental (CD) y Complejidad de Migración (CM) son obtenidos del documento Análisis de Brecha de cada aplicación. El indicador RM se corresponde a dar un 30% de peso al indicador CD y un 70% al indicador CM para después acotarlo en un rango de 15 a 100, donde 15 es un riesgo bajo y 100 es un riesgo alto.

$$RM = ([CD]*0,3+[CM]*0,7)/6,5*100$$

	Riesgo Migración RM (15-100:Alto)	Completitud Documental CD (1:Poco-10:Todo)	Complejidad Migración CM (1:Alto-5:Poco)
Definición Indicador RM	15	10	5
Prueba Mínimo RM	100	1	1
Prueba Máximo RM	15	10	5
01-SIGES	53	4	1
02-OPVIG	77	2	1
04-Veh_Integrado	30	4	3
05-SWVehiculos	100	1	1
06-VehiculosRACWeb	100	1	1
08-SCVM	24	2	5
09-VehiculosFronteraWeb	100	1	1
10-Control_Mensajería	28	5	3
21-DECARE	22	8	3
22-SICOMEXIN	77	2	1
23-DIPS_COURIER	32	8	1
24-F18_POSTAL	42	1	3
26-Mensajería_DIN	100	1	1
27-DepositoFranco-Intranet	59	1	2
29-Consultas_DUS	50	2	2
33-DIPSCF	50	2	2
34-DIPS_Viajeros	24	7	3
35-DTI	24	7	3
36-WebSeguridad	23	3	5
37-Web_AdmPersonas	53	4	1
38-WSAA_con_Argentina	77	2	1
39-WSAA_Chile	38	4	2
40-INDIRA	50	2	2
42-Firmas_ALADI	20	5	5
43-TramitacionIVV	59	1	2
44-SDA	31	6	2
45-SVC	38	4	2
46-SIDECO	31	6	2
47_DepWebFiscalizacion	77	2	1
48-SIFER	31	6	2
49-SIROFE	77	2	1
50-SIDEMAR	23	5	4
51-Servicio_Web_Canje_Mari	77	2	1
52-Aplicacion_Web_Canje_Ma	26	1	5
54-WSAlmacen	77	2	1
55-MicWeb	45	5	1
56-SIROTE	53	4	1
57-VisualSMS	31	6	2
58-WSAlmacen_y_OT	40	6	1
60-SMS_Mensajería	31	6	2
61-MensajeríaManifiestos	59	1	2
62-ListaPasajeros	20	5	5
63-PortalRayosX	17	8	5
64-ParDusGuía	63	3	1
66-SAM	18	7	5
67-SeguridadWeb	59	1	2
68-Interoperabilidad_MIC_Ar	53	4	1
69-GestionDocumental	23	5	4
74-SistemaSelectividad	30	4	3
75-SRS	25	4	4
76-ConsultaMICTransBolivia	77	2	1
87-GCP-F09	77	2	1
88-Programacion_Naviera	59	1	2
90-Acopia_Almacenistas	77	2	1
93-PortalExportadorWeb	77	2	1
97-SmdtWS	100	1	1

La gama de colores expresa el riesgo de migrar la aplicación siendo:

Color	Significado
Rojo	Riesgo Alto
Amarillo	Riesgo Moderado
Verde	Riesgo Bajo

9.2 Tamaño de aplicación

El tamaño de la aplicación, **de todas las aplicaciones excepto las de tipología APEX**, ha sido calculado en función del Riesgo de migración, las líneas de código, cantidad de librerías, cantidad de sentencias SQL y una puntuación para penalizar la dificultad desde la plataforma origen:

$$\text{Tamaño de la Aplicación} = [\text{RM}] * 0,7 + [\text{LC}] * 0,1 + [\text{CL.}] * 0,1 + [\text{CS}] * 0,1 + [\text{ES}]$$

Acrónimo	Significado	Peso
TA	Tamaño de la aplicación	-
RM	Riesgo de Migración	70%
LC	Líneas de Código (Se ha penalizado a las aplicaciones APEX con 200.000 solo para indicar dificultad o no compatibilidad total con la plataforma de destino)	10%
CL	Cantidad de Librerías	10%
CS	Cantidad de sentencias SQL	10%
ES	Equiv. App. Serv. (Puntuación aplicada a la complejidad del servidor de aplicaciones)	-

Listado de aplicaciones ordenadas por el Tamaño de Aplicación:

Aplicación	Riesgo de Migración (16-100)	Líneas de Código	Cant. Libs	Cant. SQL	Equiv. App. Serv.	Tamaño de la aplicación
27-DepositoFranco-Intranet	59	162.197	0	545	500	16.816
43-TramitacionIVV	59	154.463	0	660	500	16.054
60-SMS_Mensajería	32	104.114	34	0	100	10.537
69-GestionDocumental	24	82.348	63	579	50	8.366
21-DECARE	23	74.389	8	376	10	7.503
57-VisualSMS	32	69.859	32	0	100	7.112

Aplicación	Riesgo de Migración (16-100)	Líneas de Código	Cant. Libs	Cant. SQL	Equiv. App. Serv.	Tamaño de la aplicación
50-SIDEMAR	24	67.973	24	0	100	6.917
35-DTI	24	62.750	79	359	100	6.436
10-Control_Mensajeria	28	57.353	26	283	50	5.836
75-SRS	25	56.206	41	7	100	5.743
48_SIFER	32	52.838	26	0	100	5.409
44-SDA	32	45.852	24	9	100	4.711
33-DIPSCF	50	41.563	7	175	100	4.310
46-SIDECO	32	41.852	24	0	100	4.310
37-Web_AdmPersonas	53	29.792	1	559	100	3.172
47_DepWebFiscalizacion	77	29.777	1	407	100	3.172
04-Veh_Integrado	31	26.465	23	4	50	2.721
74-SistemaSelectividad	31	24.499	45	506	100	2.627
66-SAM	18	22.405	44	54	10	2.273
08-SCVM	25	18.995	16	45	100	2.023
63-PortalRayosX	17	17.036	45	32	10	1.733
29-Consultas_DUS	50	14.463	38	4	100	1.586
36-WebSeguridad	23	14.177	1	230	100	1.557
55-MicWeb	46	14.105	8	36	100	1.547
62-ListaPasajeros	20	12.620	46	20	10	1.293
05-SWVehiculos	100	6.593	25	6	100	832
22-SICOMEXIN	77	3.319	47	75	100	498
34-DIPS_Viajeros	24	3.300	32	101	100	460
90-Acopio_Almacenistas	77	2.420	38	44	100	404
42-Firmas_ALADI	20	3.680	52	12	10	398
58-WSAlmacen_y_OT	40	2.507	21	21	100	383
38-WSAA_con_Argentina	77	1.940	29	22	100	353
26-Mensajeria_DIN	100	2.288	14	4	50	351

Aplicación	Riesgo de Migración (16-100)	Líneas de Código	Cant. Libs	Cant. SQL	Equiv. App. Serv.	Tamaño de la aplicación
06-VehiculosRACWeb	100	1.687	37	32	100	346
09-VehiculosFronteraWeb	100	1.687	26	32	100	345
68-Interoperabilidad_MIC_Argentina	53	1.940	27	40	100	338
52-Aplicacion_Web_Canje_Marítimo	27	1.640	1	5	100	284
64-ParDusGuia	63	750	45	428	100	266
87-GCP-F09	77	911	19	17	100	249
24-F18_POSTAL	42	1.472	0	17	50	228
88-Programacion_Naviera	59	784	16	14	100	223
23-DIPS_COURIER	33	629	25	178	100	206
51-Servicio_Web_Canje_Marítimo	77	324	28	6	100	190
54-WSAlmacen	77	331	28	5	100	190
67-SeguridadWeb	59	437	16	36	100	190
97-SmdtWS	100	63	19	53	100	184
93-PortalExportadorWeb	77	99	34	6	100	168
40-INDIRA	50	179	25	4	100	156
45-SVC	39	221	18	1	100	151
39-WSAA_Chile	39	170	19	10	100	147
61-MensajeriaManifiestos	59	79	19	67		58

El tamaño de la aplicación, **de las aplicaciones APEX**, ha sido obtenido en función del número de objetos de base de datos. A continuación se listan las aplicaciones APEX ordenadas por su Tamaño de Aplicación:

Aplicación	APEX OBJS. BD	APEX PAGES
56_SIROTE	79	79
01-SIGES	54	42

Aplicación	APEX OBJS. BD	APEX PAGES
02-OPVIG	47	51
49_SIROFE	28	23
76-ConsultaMICTransBolivia	6	3

9.3 Estudio Consolidado

El documento adjunto **CMAA - MIGAPPS- Ranking Aplicaciones_(FormatoExcel)_20160322.xlsx** es una matriz compuesta por de la información e indicadores obtenidos en las fases de Levantamiento de Información y Análisis de Brecha (columnas) para todas las aplicaciones (filas). Sus columnas son:

Columna	Descripción
App.Serv	Servidor de aplicaciones donde se ejecuta la aplicación.
Equiv.App.Serv	Equivalencia Servidor de Aplicaciones, la cual tiene una constante que califica a los servidores de aplicaciones de origen por su brecha con la plataforma de destino. <ul style="list-style-type: none"> APEX 3.2.1 1000 IBM WS 4.0.3 500 BEA WL 8.1.6 100 TOMCAT 5.5.7 50 TOMCAT 5.0.28 50 JB AS 4.4.2 50 JB SOA-P 5.3 10 JB EAP 5.2 10
Tipo	Clasificación por tipo de aplicaciones (App Web, AppWeb + WS, WS, WS + AppDesktop)
Contexto	Contexto de la aplicación.
ID_Apps./Contexto	Ids de aplicaciones analizadas a las cuales corresponde cada contexto.
Casos Uso	Si la aplicación tienen casos de uso o no.
Lineas	Cantidad de líneas de código. Se castiga las aplicaciones APEX con 200.000 solo para indicar dificultad o no compatibilidad total con la plataforma de destino.
Cant. Clases	Cantidad de archivos .java
Cant. Métodos	Cantidad de métodos contenidos en las clases java

Columna	Descripción
Cant. DataSrc	Cantidad de directorios src
Complej. Sonar	Indicador de complejidad ofrecido por SonarQube
Criticidad Negocio	Nivel de importancia de la aplicación para el negocio de SNA
Complej. Técnica	Nivel de complejidad técnica aportado por los técnicos o conocedores de la aplicación del SNA.
Compl. Doc. (1-10)	Complejidad documental calculado en el Análisis de Brecha.
Complej. Mig. (1-5)	Complejidad de la migración calculado en el Análisis de Brecha.
Riesgo Migración (16-100)	Riesgo de Migración obtenido a partir del peso de la complejidad documental (30%) y la complejidad de migración (70%)
Mapa de Calor	Indicador de color del riesgo de migración: $[Riesgo\ Migración]*0,7 + [Líneas\ de\ Código]*0,1 + [Cant.Libs.]*0,1 + [Cant.SQL]*0,1 + [Equiv.App.Serv.]$
Cant. Libs	Número de librerías (jar) de cada aplicación
Tipo Proyecto	IDE donde está construida la aplicación.
Workshop	Archivo .work del proyecto workshop (solo aplica a aplicaciones Weblogic)
Pond. BEA WL	$["Netui\ tags\ JSP"]*0.2 + [jcs]*0.2 + [jcx]*0.2 + [jpf]*0.2 + [jws]*0.2$
Pond. BEA WL (%)	Pond. BEA WL en porcentaje
Netui tags JSP	Tags de netui que han sido encontrados.
htm	Número de archivos htm
html	Número de archivos html
jcs	Número de archivos Java Control Source de Bea Systems
jcx	Número de archivos Java Control Extensions de Bea Systems
jpf	Número de archivos Java Pages Flow de Bea Systems
jsp	Número de archivos Java Servlet Page
jws	Número de archivos Java Web Services de Bea Systems
properties	Número de archivos properties.
swf	Número de archivos Shockwave Flash

Columna	Descripción
wsdl	Número de archivos wsdl
xhtml	Número de archivos xhtml
xml	Número de archivos xml
xsd	Número de archivos xsd
xsl/xslx	Número de archivos xsl o xslx
Cant. SQL	Cantidad de sentencias sql encontradas en las aplicaciones

9.3.1 Conclusiones a partir del Estudio Consolidado

Las aplicaciones con mayor **riesgo de migración** (Riesgo 100 – 77) son:

Aplicación	Descripción
05-SWVehiculos	Web Service de recepción y envíos de mensajería de rent a car con Argentina
06-VehiculosRACWeb	Módulo de ingreso de registros de las empresas de rent a car
09-VehiculosFronteraWeb	Sistema donde usuarios de aduana digitan solicitudes de salida temporal mediante módulos de autoconsulta (touch)
26-Mensajería_DIN	Proceso que controla el inicio y termino de la mensajera el cual genera un log de procesamiento.
97-SmdtWS	Servicio Web de las aplicaciones: 44-SDA, 46-SIDECO, 48_SIFER
02-OPVIG	Sistema encargado del manejo de los operadores vigentes, registro, actualización y finalización de las garantías.
22-SICOMEXIN	Aplicación que maneja la tramitación DIN para los efectos del usuario aduanero (digitadores, fiscalizadores, mesa de ayuda)
38-WSAA_con_Argentina	El WS de Autenticación y Autorización es un servicio B2B ("Business to Business") que permite que los computadores pertenecientes a la AFIP y Entes Externos a la AFIP intercambien información en forma directa sin intervención de operadores.
47_DepWebFiscalizacion	Sistema destinado a que el funcionario de Aduana pueda revisar las guías Courier y defina el tipo de fiscalización que corresponde hacer a cada una de ellas.
49_SIROFE	Sistema que permite registrar las operaciones ferroviarias, ingresar BTI y TIF, realizar consultas y obtener estadísticas asociadas el manifiesto ferroviario.

Aplicación	Descripción
51-Servicio_Web_Canje_Maritimo	El objetivo de este servicio es que las Agencias de Nave y los Freight Forwarders informen cuando se realiza el canje del BL.
54-WSAlmacen	Servicio Web orientado para que el almacenista pueda registrar los BL que han sido recepcionada y/o retirada su carga en los almacenes ya que con ello se tiene la seguridad que los BL indicados en un MFTO efectivamente llegaron y se entregaron.
76-ConsultaMICTransBolivia	Sistema para consultar los MIC de tipo tránsito desde y hacia Bolivia.
87-GCP-F09	Sistema de control y administración de formularios F09
90-Acopio_Almacenistas	Módulo para el control de las mercancías en Acopio en los Almacenes Autorizados.
93-PortalExportadorWeb	Módulo para la tramitación de Solicitudes de Acopio por parte de los Exportadores / Portal para los exportadores que incluye Acopio y consultas Dus.

Aplicaciones con **mayor importancia** (Alta) para el negocio:

Aplicación	Descripción
10-Control_Mensajería	Sistema de mensajería Ventanilla Única. Es un sistema para el envío y recepción de Información a Instituciones vía Servicio WEB mediante procesos y aplicaciones.
21-DECARE	Sistema Web para el ingreso de denuncias, cargos y reclamos, además, de permitir el registro de las acciones de los Tribunales Tributarios Aduanero (TTA).
23-DIPS_COURIER	Servicio Web que recepciona y responde DIPS enviadas y confeccionadas por las agencias de COURIER, las que son procesadas y validadas en el SICOMEXIN
24-F18_POSTAL	Recepción y validación vía WebServices de F18 POSTAL. Es una librería que se ocupa para enviar los Formularios F18 de la DIN.
26-Mensajería_DIN	Proceso que controla el inicio y termino de la mensajería el cual genera un log de procesamiento.
29-Consultas_DUS	El sistema de consultas DUS es una funcionalidad comprendida dentro de la aplicación DespachadoresWeb.
33-DIPSCF	Registrará y controlará las operaciones de ingreso de mercancías presentadas ante el Servicio Nacional de Aduanas
49_SIROFE	Sistema que permite registrar las operaciones ferroviarias, ingresar BTI y TIF, realizar consultas y obtener estadísticas asociadas al manifiesto ferroviario.

Aplicación	Descripción
55-MicWeb	Sistema web para el envío de documentación electrónica de Manifiesto Terrestre por parte de las empresas transportistas terrestres o de su representante legal. El sistema muestra un formulario web con los datos que se deben enviar el Servicio.
56_SIROTE	Sistema que permite registrar las operaciones terrestres, ingresar MIC y CRT, cerrarlos, realizar consultas y obtener estadísticas asociadas el manifiesto terrestres.
61-MensajeríaManifiestos	Sistema de procesamiento de documentos electrónicos XML. Este sistema recibe, valida y procesa Manifiestos Aéreos, Marítimos, Courier, Ferroviarios y Mic/DTA Terrestres.
64-ParDusGuia	WS para el ingreso y salida de mercancías desde el PTLA.
74-SistemaSelectividad	El Sistema de Selectividad es una herramienta que permite seleccionar operaciones para desarrollar acciones de fiscalización dirigidas.
90-Acopia_Almacenistas	Módulo para el control de las mercancías en Acopia en los Almacenes Autorizados.
93-PortalExportadorWeb	Módulo para la tramitación de Solicitudes de Acopia por parte de los Exportadores / Portal para los exportadores que incluye Acopia y consultas Dus.
97-SmdtWS	Servicio Web de las aplicaciones: 44-SDA, 46-SIDECO, 48_SIFER

Aplicaciones con **mayor complejidad técnica** (Alta) :

Aplicación	Descripción
10-Control_Mensajería	Sistema de mensajería Ventanilla Única. Es un sistema para el envío y recepción de Información a Instituciones vía Servicio WEB mediante procesos y aplicaciones.
21-DECARE	Sistema Web para el ingreso de denuncias, cargos y reclamos, además, de permitir el registro de las acciones de los Tribunales Tributarios Aduanero (TTA).
23-DIPS_COURIER	Servicio Web que recepciona y responde DIPS enviadas y confeccionadas por las agencias de COURIER, las que son procesas y validadas en el SICOMEXIN
24-F18_POSTAL	Recepción y validación vía WebServices de F18 POSTAL. Es una librería que se ocupa para enviar los Formularios F18 de la DIN.
26-Mensajería_DIN	Proceso que controla el inicio y termino de la mensajera el cual genera un log de procesamiento.
29-Consultas_DUS	El sistema de consultas DUS es una funcionalidad comprendida dentro de la aplicación DespachadoresWeb.

Aplicación	Descripción
33-DIPSCF	Registrará y controlará las operaciones de ingreso de mercancías presentadas ante el Servicio Nacional de Aduanas
49_SIROFE	Sistema que permite registrar las operaciones ferroviarias, ingresar BTI y TIF, realizar consultas y obtener estadísticas asociadas el manifiesto ferroviario.
55-MicWeb	Sistema web para el envío de documentación electrónica de Manifiesto Terrestre por parte de las empresas transportistas terrestres o de su representante legal. El sistema muestra un formulario web con los datos que se deben enviar el Servicio.
56_SIROTE	Sistema que permite registrar las operaciones terrestres, ingresar MIC y CRT, cerrarlos, realizar consultas y obtener estadísticas asociadas el manifiesto terrestres.
61-MensajeríaManifiestos	Sistema de procesamiento de documentos electrónicos XML. Este sistema recibe, valida y procesa Manifiestos Aéreos, Marítimos, Courier, Ferroviarios y Mic/DTA Terrestres.
64-ParDusGuía	WS para el ingreso y salida de mercancías desde el PTLA.
74-SistemaSelectividad	El Sistema de Selectividad es una herramienta que permite seleccionar operaciones para desarrollar acciones de fiscalización dirigidas.
90-Acopio_Almacenistas	Módulo para el control de las mercancías en Acopio en los Almacenes Autorizados.
93-PortalExportadorWeb	Módulo para la tramitación de Solicitudes de Acopio por parte de los Exportadores / Portal para los exportadores que incluye Acopio y consultas Dus.
97-SmdtWS	Servicio Web de las aplicaciones: 44-SDA, 46-SIDECO, 48_SIFER

9.4 Valoración del Esfuerzo de Migración en Horas Hombre (HH)

Para obtener una forma de medir el esfuerzo de Migración de cada una de las diferentes aplicaciones se ha seguido las siguientes reglas.

Este conjunto de Reglas nace de la valoración experta de Guadaltel basada en criterios recogidos de la experiencia de la propia empresa en trabajos realizados tanto en migración de aplicaciones a otros ambientes como en las desarrolladas desde cero:

1. Las tareas necesarias para llevar cabo la migración de una aplicación al nuevo entorno serán:

- **Adquisición**, en esta tarea se recogerá el esfuerzo necesario para que el nuevo equipo consiga el conocimiento necesario para llevar a cabo la migración. Se apoyará en la documentación existente en cada una de las aplicaciones.
- **Implementación de la solución**, en esta tarea se contabilizará el esfuerzo necesario en llevar a cabo la construcción del sistema.
- **Pruebas de verificación**, tarea donde se recogerá el esfuerzo necesario para la realización de las diferentes pruebas y corrección de defectos obtenidos de las mismas

2. Estimación porcentual del tiempo dedicada a cada una de las tareas

Tarea	% Dedicación
Aquisición	15%
Implementación	55%
Pruebas	30%

3. Tiempos base estimados para la migración en base a grupos de aplicaciones:

Tipo	Horas base
Desde 0 (APEX, Destok, Sin código fuente)	600
Servicio Web	180
Weblogic	270
Toncat	180
Jboss	120
Weblogic+Servicio Web	300

4. Adaptaciones a los tiempos base según características de la aplicación estudiada.

Es decir aquellos parámetros que implicarán un incremento en los tiempos base como pueden ser:

- Complejidad
- Falta de documentación o documentación incompleta.
- Integraciones necesarias.
- Otros factores específicos de cada aplicación.

Tras el estudio del análisis de brecha realizada para cada una de las aplicaciones se ponderó la complejidad de cada una de las aplicaciones dentro de 5 niveles, correspondiendo al nivel 1 aquellas aplicaciones de mayor complejidad y el nivel 5 a las de mas bajo nivel de complejidad.

Siguiendo esta valoración se define el siguiente factor de complejidad según los diferentes niveles:

Complejidad	Factor corrección
1	0,4
2	0,3
3	0,2
4	0,1
5	0

Según lo anterior, la valoración HH quedará:

Aplicación	Adq.	Imp.	Pru.	Total	Observaciones
01-SIGES	126	462	252	840	<p>1. Las capas usan estándares básicos (HTML + PL/SQL + SQL) que corren sobre la propia base de datos Oracle, es decir, se podría seguir utilizando, mientras se migra, el sistema actual en la nueva plataforma ya que no se va a cambiar la tecnología de BBDD.</p> <p>2. La complejidad viene determinada por el poco grado de reutilización posible, puesto que la lógica de negocio está encapsulada en PL/SQL soportados por Oracle que deberán ser traducidos en objetos soportados por la tecnología de destino (como objetos Domain y Daos, capas de persistencia, Beans de negocio, etc).</p>
02-OPVIG	189	693	378	1260	<p>1. Las capas usan estándares básicos (HTML + PL/SQL + SQL) que corren sobre la propia base de datos Oracle, es decir, se podría seguir utilizando, mientras se migra, el sistema actual en la nueva plataforma ya que no se va a cambiar la tecnología de BBDD.</p> <p>2. La complejidad viene determinada por el poco grado de reutilización posible, puesto que la lógica de negocio está encapsulada en PL/SQL soportados por Oracle que deberán ser traducidos en objetos soportados por la tecnología de destino (como objetos Domain y Daos, capas de persistencia, Beans de negocio, etc).</p> <p>3. Se une la aplicación 3_Consultas Operadores Vigentes.</p>
04-Veh_Integrado	32,4	118,8	64,8	216	<p>1. Las dependencias que requieren de clases BEA Systems deberán ser sustituidas por otras libres de dependencias BEA o eliminar estas dependencias BEA de ellas cuando sean librerías generadas por SNA.</p> <p>2. JSP + Servlet es una tecnología portable a la arquitectura de destino. Sólo en caso de algunos paquetes utilizados por las JSP podría necesitarse algunos ajustes.</p> <p>3. Al usar clientes SOAP hay que considerar que, si los Servicios consumidos son migrados a JAXWS, los clientes deberán ser regenerados para adaptarse a esta tecnología.</p>
05-SWVehiculos	37,8	138,6	75,6	252	<p>1. Las dependencias que requieren de clases BEA Systems deberán ser sustituidas por otras libres de dependencias BEA o eliminar estas dependencias BEA de ellas cuando sean librerías generadas por SNA.</p> <p>2. El uso de controles netui es una fuerte vinculación con BEA.</p> <p>3. Al usar clientes SOAP hay que considerar que, si los Servicios consumidos son migrados a JAXWS, los clientes deberán ser regenerados para adaptarse a esta tecnología.</p> <p>4. Los servicios provistos están contruidos bajo la tecnología JWS que tiene una fuerte dependencia de Bea Systems y que se tendrá que eliminar por ejemplo, con el estándar JAX-WS.</p>

Aplicación	Adq.	Imp.	Pru.	Total	Observaciones
06-VehiculosRACWeb	94,5	346,5	189	630	1. Las dependencias que requieren de clases BEA Systems deberán ser sustituidas por otras libres de dependencias BEA o eliminar estas dependencias BEA de ellas cuando sean librerías generadas por SNA. 2. El uso de controles netui es una fuerte vinculación con BEA. 3. Al usar clientes SOAP hay que considerar que, si los Servicios consumidos son migrados a JAXWS, los clientes deberán ser regenerados para adaptarse a esta tecnología. 4. Se une la aplicación 7_Rent a Car
08-SCVM	40,5	148,5	81	270	1. No existen dependencias de Bea Systems tanto en el código fuente como en dependencias de librerías externas. 2. Contiene librerías estándares que no presentan riesgo para la migración. 3. Al usar clientes SOAP hay que considerar que, si los Servicios consumidos son migrados a JAXWS, los clientes deberán ser regenerados para adaptarse a esta tecnología.
09-VehiculosFronteraWeb	56,7	207,9	113,4	378	1. Las dependencias que requieren de clases BEA Systems deberán ser sustituidas por otras libres de dependencias BEA o eliminar estas dependencias BEA de ellas cuando sean librerías generadas por SNA. 2. El uso de controles netui es una fuerte vinculación con BEA. 3. Al usar clientes SOAP hay que considerar que, si los Servicios consumidos son migrados a JAXWS, los clientes deberán ser regenerados para adaptarse a esta tecnología.
10-Control_Mensajeria	340,2	1351,35	737,1	2428,65	1. Eliminar dependencias de Bea Systems en las librerías que usa la aplicación. Utiliza 4 librerías del SNA que tienen dependencias de BEA Systems. 2. Al usar clientes SOAP hay que considerar que, si los Servicios consumidos son migrados a JAXWS, los clientes deberán ser regenerados para adaptarse a esta tecnología. 3. Es una aplicación que engloba a 21 módulos todos ellos con servicios provistos y consumidos (contexto ventanilla prueba)
21-DECARE	48,6	178,2	97,2	324	1. No se han detectado dependencias que requieran de clases BEA Systems, lo que facilitará el trabajo de migración. 2. Se recomienda actualizar la versión de Struts. 3. Aunque el motor jBPM está soportado en la plataforma de destino, pueden necesitarse ajustes relacionados con los mapeos de configuración y con las propiedades JPA, cambios que pueden ser complejos puesto que la especificación JPA 2.0 de la plataforma de destino contiene importantes cambios con respecto a sus versiones anteriores

Aplicación	Adq.	Imp.	Pru.	Total	Observaciones
22-SICOMEXIN	56,7	207,9	113,4	378	1. Las dependencias que requieren de clases BEA Systems deberán ser sustituidas por otras libres de dependencias BEA o eliminar estas dependencias BEA de ellas cuando sean librerías generadas por SNA. 2. El uso de controles netui es una fuerte vinculación con BEA. 3. Al usar clientes SOAP hay que considerar que, si los Servicios consumidos son migrados a JAXWS, los clientes deberán ser regenerados para adaptarse a esta tecnología.
23-DIPS_COURIER	37,8	138,6	75,6	252	1. Las dependencias que requieren de clases BEA Systems deberán ser sustituidas por otras libres de dependencias BEA o eliminar estas dependencias BEA de ellas cuando sean librerías generadas por SNA. Utiliza 7 librerías del SNA que tienen dependencias de BEA Systems. 2. El uso de controles netui es una fuerte vinculación con BEA. 3. Algunos de los Web Services Provistos están contruidos bajo la tecnología JWS de Bea Systems por lo que existe complejidad para reemplazarlos por algún estándar como JAX-WS.
24-F18_POSTAL	32,4	118,8	64,8	216	1. No existen librerías incompatibles con la plataforma de destino. 2. La aplicación tiene 23 fuentes java por lo que su reducido tamaño facilita la migración. 3. Tanto en el levantamiento como en el análisis de brecha no se disponía de código fuente, se ha extraído el JAR mediante la herramienta JD-GUI 1.4.0. Esto podría ser una dificultad para efectuar la migración. 4. Los Web Services Provistos son expuestos a través del componente "Apache SOAP 2.3" [https://archive.apache.org/dist/ws/soap/version-2.3/]. Habrá que implementar la interfaz de exposición de estos servicios y plantear una estrategia de proxy a través de un ESB para mantener la compatibilidad con la forma de la mensajería actual.
26-Mensajería_DIN	27	99	54	180	1. No se encuentran librerías externas o del SNA incompatibles con la arquitectura de destino. 2. Al usar clientes SOAP hay que considerar que, si los Servicios consumidos son migrados a JAXWS, los clientes deberán ser regenerados para adaptarse a esta tecnología.
27-DepositoFranco-Intranet	78,98	289,58	157,95	526,5	1. JSP + Servlet es una tecnología portable a la arquitectura de destino. Sólo en caso de algunos paquetes utilizados por las JSP podría necesitarse algunos ajustes. 2. El JDK utilizado para el desarrollo es muy antiguo y deberán realizarse ajustes para Java 1.7. 3. La falta de documentación y la no localización de las fuentes pueden ser, más que la complejidad del negocio en si, los mayores obstáculos a la hora de efectuar la migración. 4. Se une la aplicación 28_DepositoFrancoInternet

Aplicación	Adq.	Imp.	Pru.	Total	Observaciones
29-Consultas_DUS	52,65	193,05	105,3	351	<ol style="list-style-type: none"> 1. Las dependencias que requieren de clases BEA Systems deberán ser sustituidas por otras libres de dependencias BEA o eliminar estas dependencias BEA de ellas cuando sean librerías generadas por SNA. Utiliza 7 librerías del SNA que tienen dependencias de BEA Systems. 2. El uso de controles netui es una fuerte vinculación con BEA. 3. Al usar clientes SOAP hay que considerar que, si los Servicios consumidos son migrados a JAXWS, los clientes deberán ser regenerados para adaptarse a esta tecnología.
33-DIPSCF	52,65	193,05	105,3	351	<ol style="list-style-type: none"> 1. Aunque existe poca documentación no se han encontrado dependencias de las librerías dependientes de Bea Systems en el módulo web que es al que pertenece el código fuente analizado. 2. No se encuentran librerías externas o de terceros que presenten incompatibilidad. 3. Se encuentran estas librerías PasswordEncryptCommon.jar y SecurityFramework.jar que no se consideran librerías estándar sino que parecen algún desarrollo propio y aunque en principio no parecen tener incompatibilidad habría que tener especial cuidado a la hora de la migración. 4. En el informe Tattleate se ha detectado que la librería ClienteTGRCID.jar (perteneciente a un desarrollo propio) contiene dependencias de Bea Systems por lo que habría que eliminar las dependencias de este cliente para que pueda ser usado dentro de la aplicación. Al usar clientes SOAP hay que considerar que, si los Servicios consumidos son migrados a JAX-WS, los clientes deberán ser regenerados para adaptarse a esta tecnología. 5. En el informe Tattleate no refleja dependencias de Bea Systems en el cliente DipscfEJB.jar y su código fuente no presenta problemas de migración.
34-DIPS_Viajeros	48,6	178,2	97,2	324	<ol style="list-style-type: none"> 1. Las dependencias que requieren de clases BEA Systems deberán ser sustituidas por otras libres de dependencias BEA o eliminar estas dependencias BEA de ellas cuando sean librerías generadas por SNA. 2. El uso de controles netui es una fuerte vinculación con BEA. 3. Al usar clientes SOAP hay que considerar que, si los Servicios consumidos son migrados a JAXWS, los clientes deberán ser regenerados para adaptarse a esta tecnología. 4. El uso de Flex y Flex Messaging (al ser una tecnología fuera de tendencia) puede hacer que la migración sea más compleja aunque no se considera una tecnología incompatible con la arquitectura. 5. Flex utiliza la API Java Message Service (JMS) para el envío de mensajes entre dos clientes, esto no se considera un riesgo ya que es compatible con la plataforma de destino.
35 DTI	40	240	80	360	Implica la migración de cliente WS
36 Esquema de Seguridad	40	150	80	270	

Aplicación	Adq.	Imp.	Pru.	Total	Observaciones
37-Web_AdmPersonas	56,7	207,9	113,4	378	1. JSP es una tecnología portable a la arquitectura de destino. Sólo en caso de algunos paquetes utilizados por las JSP podría necesitarse algunos ajustes. 2. Este sistema cuenta con múltiples elementos (utilTuxpan.jar, Z3.jar, WebComponents.jar, etc) cuyo comportamiento en la plataforma de destino no puede ser evaluado de forma estándar, al ser componentes de desarrollos a medidas de terceras empresas. Estos componentes pueden suponer un escoyo importante y se recomienda se enfoquen los esfuerzos a su sustitución más que su adaptación
38-WSAA_con_Argentina	37,8	138,6	75,6	252	1. Las dependencias que requieren de clases BEA Systems deberán ser sustituidas por otras libres de dependencias BEA o eliminar estas dependencias BEA de ellas cuando sean librerías generadas por SNA. 2. El uso de controles netui es una fuerte vinculación con BEA. 3. Los servios provistos por el sistema deberán ajustarse a JAX-WS, lo que conllevará también ajustes en los clientes que, actualmente, hagan uso de estos servicios.
39-WSAA_Chile	35,1	128,7	70,2	234	1. Las dependencias que requieren de clases BEA Systems deberán ser sustituidas por otras libres de dependencias BEA o eliminar estas dependencias BEA de ellas cuando sean librerías generadas por SNA. 2. El uso de controles netui es una fuerte vinculación con BEA. 3. Los servios provistos por el sistema deberán ajustarse a JAX-WS, lo que conllevará también ajustes en los clientes que, actualmente, hagan uso de estos servicios.
40-INDIRA	35,1	128,7	70,2	234	1. Las dependencias que requieren de clases BEA Systems deberán ser sustituidas por otras libres de dependencias BEA o eliminar estas dependencias BEA de ellas cuando sean librerías generadas por SNA. 2. Existen dos librerías del SNA que contienen dependencias de Bea Systems: ControlesINDIRA.jar y EsquemaINDIRA.jar. 3. El uso de controles netui es una fuerte vinculación con BEA. 4. Los servicios provistos están construidos bajo la tecnología JWS que tiene una fuerte dependencia de Bea Systems y que se tendrá que eliminar por ejemplo, con el estándar JAX-WS. 5. Al usar clientes SOAP hay que considerar que, si los Servicios consumidos son migrados a JAXWS, los clientes deberán ser regenerados para adaptarse a esta tecnología.
42-Firmas_ALAD	18	66	36	120	1. No se ha encontrado ninguna librería que sea incompatible con la arquitectura destino. 2. La versión de Java es 1.6 por lo que existe poca brecha para migrar a 1.7.

Aplicación	Adq.	Imp.	Pru.	Total	Observaciones
43-TramitacionIVV	52,65	193,05	105,3	351	1. JSP + Servlet es una tecnología portable a la arquitectura de destino. 2. El JDK utilizado para el desarrollo es muy antiguo y deberán realizarse ajustes para Java 1.7. 3. La falta de documentación y la no localización de las fuentes pueden ser, más que la complejidad del negocio en si, los mayores obstáculos a la hora de efectuar la migración.
44-SDA	117	429	234	780	1. La ponderación se realiza sobre el supuesto de migrar el cliente SWT a Aplicación Web pero seguir utilizando el servicio SmdtWS como encapsulador de la lógica de negocio y manteniendo la no persistencia de datos más allá de los ficheros XML de intercambio que deberían almacenarse.
45-SVC	52,65	193,05	105,3	351	1. Las dependencias que requieren de clases BEA Systems deberán ser sustituidas por otras libres de dependencias BEA o eliminar estas dependencias BEA de ellas cuando sean librerías generadas por SNA. 2. El uso de controles netui es una fuerte vinculación con BEA.
46-SIDECO	117	429	234	780	1. La ponderación se realiza sobre el supuesto de migrar el cliente SWT a Aplicación Web pero seguir utilizando el servicio SmdtWS como encapsulador de la lógica de negocio y manteniendo la no persistencia de datos más allá de los ficheros XML de intercambio que deberían almacenarse.
47_DepWebFiscalizacion	56,7	207,9	113,4	378	1. JSP es una tecnología portable a la arquitectura de destino. Sólo en caso de algunos paquetes utilizados por las JSP podría necesitarse algunos ajustes. 2. Este sistema cuenta con múltiples elementos (utilTuxpan.jar, Z3.jar, WebComponents.jar, etc) cuyo comportamiento en la plataforma de destino no puede ser evaluado de forma estándar, al ser componentes de desarrollos a medidas de terceras empresas. Estos componentes pueden suponer un escollo importante y se recomienda se enfoquen los esfuerzos a su sustitución más que su Adaptación.
48_SIFER	117	429	234	780	1. La ponderación se realiza sobre el supuesto de migrar el cliente SWT a Aplicación Web pero seguir utilizando el servicio SmdtWS como encapsulador de la lógica de negocio y manteniendo la no persistencia de datos más allá de los ficheros XML de intercambio que deberían almacenarse.
49_SIROFE	126	462	252	840	1. Las capas usan estándares básicos (HTML + PL/SQL + SQL) que corren sobre la propia base de datos Oracle, es decir, se podría seguir utilizando, mientras se migra, el sistema actual en la nueva plataforma ya que no se va a cambiar la tecnología de BBDD. 2. La complejidad viene determinada por el poco grado de reutilización posible, puesto que la lógica de negocio está encapsulada en PL/SQL soportados por Oracle que deberán ser traducidos en objetos soportados por la tecnología de destino (como objetos Domain y Daos, capas de persistencia, Beans de negocio, etc).

Aplicación	Adq.	Imp.	Pru.	Total	Observaciones
50 SIDEMAR	90	330	180	600	Implica una codificación completa
51-Servicio_Web_Canje_Marítimo	37,8	138,6	75,6	252	1. Las dependencias que requieren de clases BEA Systems deberán ser sustituidas por otras libres de dependencias BEA o eliminar estas dependencias BEA de ellas cuando sean librerías generadas por SNA. 2. El uso de controles netui es una fuerte vinculación con BEA. 3. Los servios provistos por el sistema deberán ajustarse a JAX-WS, lo que conllevará también ajustes en los clientes que, actualmente, hagan uso de estos servicios.
52-Aplicacion_Web_Canje_Marítimo	40,5	148,5	81	270	1. Con la información ofrecida en el informe de levantamiento la aplicación es fácil de migrar ya que no tiene dependencias de Bea Systems. 2. Requiere el framework Z3 por lo que depende de la migración de este framework para su correcto Funcionamiento.
54-WSAlmacen	37,8	138,6	75,6	252	1. Las dependencias que requieren de clases BEA Systems deberán ser sustituidas por otras libres de dependencias BEA o eliminar estas dependencias BEA de ellas cuando sean librerías generadas por SNA. 2. El uso de controles netui es una fuerte vinculación con BEA. 3. Al usar clientes SOAP hay que considerar que, si los Servicios consumidos son migrados a JAXWS, los clientes deberán ser regenerados para adaptarse a esta tecnología.
55-MicWeb	56,7	207,9	113,4	378	1. JSP + Servlet es una tecnología portable a la arquitectura de destino. Sólo en caso de algunos paquetes utilizados por las JSP podría necesitarse algunos ajustes. 2. Este sistema cuenta con múltiples elementos (utilTuxpan.jar, Z3.jar, WebComponents.jar, etc) cuyo comportamiento en la plataforma de destino no puede ser evaluado de forma estándar, al ser componentes de desarrollos a medidas de terceras empresas. Estos componentes pueden suponer un escollo importante y se recomienda se enfoquen los esfuerzos a su sustitución más que su adaptación.
56_SIROTE	126	462	252	840	1. Las capas usan estándares básicos (HTML + PL/SQL + SQL) que corren sobre la propia base de datos Oracle, es decir, se podría seguir utilizando, mientras se migra, el sistema actual en la nueva plataforma ya que no se va a cambiar la tecnología de BBDD. 2. La complejidad viene determinada por el poco grado de reutilización posible, puesto que la lógica de negocio está encapsulada en PL/SQL soportados por Oracle que deberán ser traducidos en objetos soportados por la tecnología de destino (como objetos Domain y Daos, capas de persistencia, Beans de negocio, etc).

Aplicación	Adq.	Imp.	Pru.	Total	Observaciones
57-VisualSMS	117	429	234	780	1. La ponderación se realiza sobre el supuesto de migrar el cliente SWT a Aplicación Web pero seguir utilizando el servicio S2MC como encapsulador de la lógica de negocio y manteniendo la no persistencia de datos más allá de los ficheros XML de intercambio que deberían almacenarse.
58-WSAlmacen_y_OT	56,7	207,9	113,4	378	1. Las dependencias que requieren de clases BEA Systems deberán ser sustituidas por otras libres de dependencias BEA o eliminar estas dependencias BEA de ellas cuando sean librerías generadas por SNA. 2. El uso de controles netui es una fuerte vinculación con BEA. No obstante, no aplica a la lógica de negocio en sí del WS sino a las páginas que lo consumen y que van en el propio EAR. 3. Si los Servicios consumidos son migrados a JAX-WS, los clientes deberán ser regenerados para adaptarse a esta tecnología. 4. Este sistema cuenta con múltiples elementos (utilTuxpan.jar, Z3.jar, etc) cuyo comportamiento en la plataforma de destino no puede ser evaluado de forma estándar, al ser componentes de desarrollos a medidas de terceras empresas. Estos componentes pueden suponer un escollo importante y se recomienda se enfoquen los esfuerzos a su sustitución más que su adaptación. 5. Se incluye la aplicación 59 Servicio Web Documentos Tramitados.
60-SMS_Mensajería	117	429	234	780	1. La ponderación se realiza sobre el supuesto de migrar el cliente SWT a Aplicación Web pero seguir utilizando el servicio S2MCWS como encapsulador de la lógica de negocio y manteniendo la no persistencia de datos más allá de los ficheros XML de intercambio que deberían almacenarse.
61 Sistema Mensajería Manifiestos Electrónicos	54	99	54	207	La mayor parte del código es reutilizable / No hay documentación
62-ListaPasajeros	18	66	36	120	1. No se ha encontrado ninguna librería que sea incompatible con la arquitectura destino. 2. La arquitectura de origen es JBoss EAP 5.2 por lo que la migración a JBoss EAP 6.4 implica pocos riesgos. 3. La versión de Java es 1.6 por lo que existe poca brecha para migrar a 1.7. 4. Al usar clientes SOAP hay que considerar que, si los Servicios consumidos son migrados a JAXWS, los clientes deberán ser regenerados para adaptarse a esta tecnología.
63-PortalRayosX	45	165	90	300	1. No se ha encontrado ninguna librería que sea incompatible con la arquitectura destino. 2. La versión de Java es 1.6 por lo que existe poca brecha para migrar a 1.7.

Aplicación	Adq.	Imp.	Pru.	Total	Observaciones
64-ParDusGuia	63	231	126	420	1. Las dependencias que requieren de clases BEA Systems deberán ser sustituidas por otras libres de dependencias BEA o eliminar estas dependencias BEA de ellas cuando sean librerías generadas por SNA. Utiliza 7 librerías del SNA que tienen dependencias de BEA Systems. 2. El uso de controles netui es una fuerte vinculación con BEA. 3. Al usar clientes SOAP hay que considerar que, si los Servicios consumidos son migrados a JAXWS, los clientes deberán ser regenerados para adaptarse a esta tecnología. 4. Los servicios provistos están construidos bajo la tecnología JWS que tiene una fuerte dependencia de Bea Systems y que se tendrá que eliminar por ejemplo, con el estándar JAX-WS. 5. Existen 827 clases pero sin fuentes java identificados. Existen varias libs que no disponen de fuentes. Lo más probable es que los class que están desplegados se tengan que migrar decompilándolos. Esto añade más complejidad a la aplicación.
66-SAM	18	66	36	120	1. No se ha encontrado ninguna librería que sea incompatible con la arquitectura destino. 2. La arquitectura de origen es JBoss EAP 5.2 por lo que la migración a JBoss EAP 6.4 implica pocos riesgos. 3. La versión de Java es 1.6 por lo que existe poca brecha para migrar a 1.7. 4. Al usar clientes SOAP hay que considerar que, si los Servicios consumidos son migrados a JAXWS, los clientes deberán ser regenerados para adaptarse a esta tecnología.
67-SeguridadWeb	52,65	193,05	105,3	351	1. Las dependencias que requieren de clases BEA Systems deberán ser sustituidas por otras libres de dependencias BEA o eliminar estas dependencias BEA de ellas cuando sean librerías generadas por SNA. 2. El uso de controles netui es una fuerte vinculación con BEA. 3. La falta total de documentación supondrá un plus de complejidad.
68-Interoperabilidad_MIC_Argentina	37,8	138,6	75,6	252	1. Las dependencias que requieren de clases BEA Systems deberán ser sustituidas por otras libres de dependencias BEA o eliminar estas dependencias BEA de ellas cuando sean librerías generadas por SNA. 2. El uso de controles netui es una fuerte vinculación con BEA. 3. Los servios provistos por el sistema deberán ajustarse a JAX-WS, lo que conllevará también ajustes en los clientes que, actualmente, hagan uso de estos servicios.

Aplicación	Adq.	Imp.	Pru.	Total	Observaciones
69-GestionDocumental	19,8	72,6	39,6	132	1. No se ha encontrado ninguna librería que sea incompatible con la arquitectura destino. 2. La arquitectura de origen es JBoss EAP 5.2 por lo que la migración a JBoss EAP 6.4 implica pocos riesgos. 3. La versión de Java es 1.5 por lo que existe poca brecha para migrar a 1.7. 4. Al usar clientes SOAP hay que considerar que, si los Servicios consumidos son migrados a JAXWS, los clientes deberán ser regenerados para adaptarse a esta tecnología. 5. Aunque a priori puede parecer que no haya grandes dificultades para realizar la migración, la aplicación contiene muchas funcionalidades y 631 fuentes .java, por lo que puede ser laborioso testear todas las funcionalidades.
74-SistemaSelectividad	48,6	178,2	97,2	324	1. No se han detectado dependencias que requieran de clases BEA Systems, lo que facilitará el trabajo de migración. 2. Se recomienda actualizar la versión de Struts. 3. El servicio que ofrece la aplicación así como los clientes que utiliza, deberán ajustarse a la especificación JAX-RPC.
75-SRS	44,55	163,35	89,1	297	1. Las dependencias que requieren de clases BEA Systems deberán ser sustituidas por otras libres de dependencias BEA o eliminar estas dependencias BEA de ellas cuando sean librerías generadas por SNA. 2. Se recomienda actualizar la versión de Struts. 3. La capa de persistencia iBatis puede requerir algunos reajustes en los ficheros de mapeos o Configuración.
76-ConsultaMICTransBolivia	126	462	252	840	1. Las capas usan estándares básicos (HTML + PL/SQL + SQL) que corren sobre la propia base de datos Oracle, es decir, se podría seguir utilizando el sistema actual en la nueva plataforma ya que no se va a cambiar la tecnología de BBDD. 2. La complejidad viene determinada por el poco grado de reutilización posible, puesto que la lógica de negocio está encapsulada en PL/SQL soportados por Oracle que deberán ser traducidos en objetos soportados por la tecnología de destino (como objetos Domain y Daos, capas de persistencia, Beans de negocio, etc).
87-GCP-F09	56,7	207,9	113,4	378	1. Las dependencias que requieren de clases BEA Systems deberán ser sustituidas por otras libres de dependencias BEA o eliminar estas dependencias BEA de ellas cuando sean librerías generadas por SNA. 2. El uso de controles netui es una fuerte vinculación con BEA. 3. Al usar clientes SOAP hay que considerar que, si los Servicios consumidos son migrados a JAXWS, los clientes deberán ser regenerados para adaptarse a esta tecnología.

Aplicación	Adq.	Imp.	Pru.	Total	Observaciones
88-Programacion_Naviera	52,65	193,05	105,3	351	1. Las dependencias que requieren de clases BEA Systems deberán ser sustituidas por otras libres de dependencias BEA o eliminar estas dependencias BEA de ellas cuando sean librerías generadas por SNA. 2. El uso de controles netui es una fuerte vinculación con BEA.
90-Acopio_Almacenistas	37,8	138,6	75,6	252	1. Las dependencias que requieren de clases BEA Systems deberán ser sustituidas por otras libres de dependencias BEA o eliminar estas dependencias BEA de ellas cuando sean librerías generadas por SNA. 2. El uso de controles netui es una fuerte vinculación con BEA. 3. El servicio que ofrece la aplicación deberá ajustarse a la especificación JAX-RPC.
93-PortalExportadorWeb	56,7	207,9	113,4	378	1. Las dependencias que requieren de clases BEA Systems deberán ser sustituidas por otras libres de dependencias BEA o eliminar estas dependencias BEA de ellas cuando sean librerías generadas por SNA. 2. El uso de controles netui es una fuerte vinculación con BEA. 3. Existen 15 librerías propias del SNA dependientes de BEA Systems que deberán ser tratadas en primer lugar para que la aplicación funcione correctamente una vez migrada.
97-SmdtWS	27	99	54	180	

10 Agrupación de aplicaciones en escenarios

En esta etapa se agrupan las aplicaciones priorizadas en base a los escenarios definidos en el punto [Escenarios de Migración](#).

Escenario 1:

Aplicación	Descripción
61-MensajeríaManifiestos	Sistema de procesamiento de documentos electrónicos XML. Este sistema recibe, valida y procesa Manifiestos Aéreos, Marítimos, Courier, Ferroviarios y Mic/DTA Terrestres.
23-DIPS_COURIER	Servicio Web que recepciona y responde DIPS enviadas y confeccionadas por las agencias de COURIER, las que son procesas y validadas en el SICOMEXIN
24-F18_POSTAL	Recepción y validación vía WebServices de F18 POSTAL. Es una librería que se ocupa para enviar los Formularios F18 de la DIN.
64-ParDusGuia	WS para el ingreso y salida de mercancías desde el PTLA.
26-Mensajería_DIN	Proceso que controla el inicio y termino de la mensajera el cual genera un log de procesamiento.
55-MicWeb	Sistema web para el envío de documentación electrónica de Manifiesto Terrestre por parte de las empresas transportistas terrestres o de su representante legal. El sistema muestra un formulario web con los datos que se deben enviar el Servicio.
29-Consultas_DUS	El sistema de consultas DUS es una funcionalidad comprendida dentro de la aplicación DespachadoresWeb.
74-SistemaSelectividad	El Sistema de Selectividad es una herramienta que permite seleccionar operaciones para desarrollar acciones de fiscalización dirigidas.
33-DIPSCF	Registrará y controlará las operaciones de ingreso de mercancías presentadas ante el Servicio Nacional de Aduanas
10-Control_Mensajería	Sistema de mensajería Ventanilla Única. Es un sistema para el envío y recepción de Información a Instituciones vía Servicio WEB mediante procesos y aplicaciones.
21-DECARE	Sistema Web para el ingreso de denuncias, cargos y reclamos, además, de permitir el registro de las acciones de los Tribunales Tributarios Aduanero (TTA).

Escenario 2:

Aplicación	Descripción
39-WSAA_Chile	El WS de Autenticación y Autorización es un servicio B2B ("Business to Business") que permite que los computadores pertenecientes a la AFIP y Entes Externos a la AFIP intercambien información en forma directa sin intervención de operadores.

Aplicación	Descripción
51-Servicio_Web_Canje_Marítimo	El objetivo de este servicio es que las Agencias de Nave y los Freight Forwarders informen cuando se realiza el canje del BL.
54-WSAlmacen	Servicio Web orientado para que el almacenista pueda registrar los BL que han sido recepcionada y/o retirada su carga en los almacenes ya que con ello se tiene la seguridad que los BL indicados en un MFTO efectivamente llegaron y se entregaron.
68-Interoperabilidad_MIC_Argentina	Intercambio MIC-DTA argentina previa generación de token y firma vía sistema WSAA.
06-VehiculosRACWeb	Módulo de ingreso de registros de las empresas de rent a car
38-WSAA_con_Argentina	El WS de Autenticación y Autorización es un servicio B2B ("Business to Business") que permite que los computadores pertenecientes a la AFIP y Entes Externos a la AFIP intercambien información en forma directa sin intervención de operadores.
58-WSAlmacen_y_OT	Este servicio permite consultar los XML relacionados a un número de manifiesto aéreo o marítimo y responde con documentos BL o Guías Aéreas según el perfil de usuario que consulte: Almacenista u Operador de Terminal.
34-DIPS_Viajeros	Registra DIPS correspondientes a la importación de mercancías que porten los viajeros procedentes del extranjero o de una zona de tratamiento aduanero especial y que no se encuentren comprendidas en el concepto de "equipaje de viajeros".
22-SICOMEXIN	Aplicación que maneja la tramitación DIN para los efectos del usuario aduanero (digitadores, fiscalizadores, mesa de ayuda)
05-SWVehiculos	Web Service de recepción y envíos de mensajería de rent a car con Argentina
62-ListaPasajeros	Registro de salida de bus (viaje, pasajero tripulantes) y control en frontera.
36-WebSeguridad	Sistema para la asignación de claves y perfiles para los usuarios de los sistemas de manifiestos Electrónicos. Se administran claves tanto para usuarios internos como para usuarios externos.
08-SCVM	Sistema donde usuarios externos a la aduana elaboran solicitudes anticipadas de salida temporal, las que luego son procesadas en frontera.

Escenario 3:

Aplicación	Descripción
66-SAM	Es un sistema de alerta o aviso de parte de los particulares interesados en denunciar situaciones anómalas de las cuales posee información.
04-Veh_Integrado	Control de vehículos particulares que pasan por fronteras que no están integradas con Argentina.

Aplicación	Descripción
37-Web_AdmPersonas	Sistema para la administración de los usuarios de los sistemas de Manifiestos Electrónicos. Se administran tanto usuarios internos como usuarios externos.
47_DepWebFiscalizacion	Sistema destinado a que el funcionario de Aduana pueda revisar las guías Courier y defina el tipo de fiscalización que corresponde hacer a cada una de ellas.
46-SIDECO	Sistema que permite la transmisión electrónica (confección y envío de manifiestos courier, guías time courier) de la lista de carga courier para el ingreso de mercancías al país.
44-SDA	Sistema para el envío de documentación electrónica de Manifiesto Aéreo por parte de las Líneas Aéreas y de las empresas Freight Forwarders. Este sistema está orientado para el envío de un documento a la vez, para lo cual presenta un formulario con los datos que se deben enviar el Servicio.
48_SIFER	Sistema para el envío de documentación electrónica de Manifiesto Ferroviario, TIF/DTA y BTI, por parte de las Empresa Ferroviarias. Este sistema está orientado para el envío de un documento a la vez, para lo cual presenta un formulario con los datos que se deben enviar el Servicio.
75-SRS	Sistema que permite el control de las operaciones de Regimenes Suspensivos que son aquellas operaciones de ingreso o salida de mercancías del país, que gozan de un régimen especial que les permite permanecer a la espera de la destinación definitiva que se les otorgue.
35-DTI	Sistema destinado a controlar el tránsito interno de mercancías dentro del país.
50-SIDEMAR	Sistema para el envío de documentación electrónica de Manifiesto Marítimo por parte de las agencias de Naves, Cías. Marítimas y de las empresas Freight Forwarders. Este sistema está orientado para el envío de un documento a la vez, para lo cual presenta un formulario con los datos que se deben enviar al Servicio.
57-VisualSMS	Sistema para consultar y visualizar documentos electrónicos por parte de usuarios y funcionarios de acuerdo al perfil que tenga asignado. Permite consultar por manifiesto aéreo, marítimo, courier y ferroviario.

Escenario 4:

Aplicación	Descripción
69-GestionDocumental	Sistema destinado a controlar el tránsito interno de mercancías dentro del país.

Aplicación	Descripción
60-SMS_Mensajería	El sistema SMS fue diseñado con el fin de servir como herramienta de apoyo a los programadores para desarrollar sistemas similares, que se encarguen de gestionar mensajería vía SOAP con los WebServices de Aduana para el envío masivo de documentos de manifiestos electrónicos, ya sean Marítimos, Aéreo, Courier o Ferroviario. Este sistema es utilizado por los usuarios para el envío masivo de documentación de manifestación electrónica generados desde sus propios sistemas.
45-SVC	Sistema para validar que los conocimientos de embarque impresos desde el servidor de Aduanas son válidos.
40-INDIRA	WS de consulta de declaraciones DIN y DUS y por rango de fechas de las mismas, desde Argentina a Chile y viceversa.
67-SeguridadWeb	Sistema que permite otorgar permisos a usuarios internos y externos a distintas aplicaciones.
52-Aplicacion_Web_Canje_Marítimo	Aplicación Web desarrollada para que el usuario emisor de un conocimiento de embarque digite en un formulario web la información del canje.
93-PortalExportadorWeb	Módulo para la tramitación de Solicitudes de Acopio por parte de los Exportadores / Portal para los exportadores que incluye Acopio y consultas Dus.

Escenario 5:

Aplicación	Descripción
09-VehiculosFronteraWeb	Sistema donde usuarios de aduana digitan solicitudes de salida temporal mediante módulos de autoconsulta (touch)
42-Firmas_ALADI	WS de consulta de declaraciones DIN y DUS y por rango de fechas de las mismas, desde Argentina a Chile y viceversa.
88-Programacion_Naviera	Módulo de Consulta de Manifiestos e ingreso de Zarpes o Arribos de naves marítimas.
63-PortalRayosX	Visualización de los datos de escaneo de camiones y asociación de documentos aduaneros (DIN, DUS, MIC, DTI) con el registro del escáner.

Escenario 6:

Aplicación	Descripción
49_SIROFE	Sistema que permite registrar las operaciones ferroviarias, ingresar BTI y TIF, realizar consultas y obtener estadísticas asociadas al manifiesto ferroviario.

Aplicación	Descripción
56_SIROTE	Sistema que permite registrar las operaciones terrestres, ingresar MIC y CRT, cerrarlos, realizar consultas y obtener estadísticas asociadas el manifiesto terrestres.
90-Acopio_Almacenistas	Módulo para el control de las mercancías en Acopio en los Almacenes Autorizados.
97-SmdtWS	Servicio Web de las aplicaciones: 44-SDA, 46-SIDECO, 48_SIFER
87-GCP-F09	Sistema de control y administración de formularios F09
43-TramitacionIVV	Sistema para el control del Informe de Variación de Valor.
01-SIGES	Sistema encargado de entregar estadísticas de DIN y DUS.
02-OPVIG	Sistema encargado del manejo de los operadores vigentes, registro, actualización y finalización de las garantías.
27-DepositoFranco-Intranet	Sistema que lleva el control de las empresas de aeronavegación, nacionales o extranjeras, que operan servicios internacionales, traigan o reciban del exterior para el uso, empleo, consumo o venta a bordo de sus vuelos, como asimismo, para el mantenimiento y conservación de sus aeronaves
76-ConsultaMICTransBolivia	Sistema para consultar los MIC de tipo tránsito desde y hacia Bolivia.

11 Ejecución del plan de migración

En esta etapa, con el apoyo de la información recopilada, el plan técnico de migración propuesto y la visión de negocio, se ejecuta el plan de migración para los escenarios.

De acuerdo a los escenarios de migración identificados (VER: [Escenarios de Migración](#)) el plan de ejecución de migración debería estar dado por el siguiente orden:



Diagrama 8: Escenarios de Migración

Precisar que esta propuesta de ejecución basada en esfuerzo podría verse afectada por una decisión de negocio, la cual podría priorizar la migración de aplicaciones por el grupo de negocio que se necesite disponer con mayor prioridad.

Ejemplos:

- Se decide ejecutar un plan de aplicaciones de la línea de negocio de "Destinaciones Aduaneras" (DIN/DUS)
- Se decide ejecutar un plan de aplicaciones de la línea de negocio de "Manifiesto Electrónico"

11.1 Plan de Migración

Se pueden establecer los siguientes hitos en la ejecución del plan de migración para una aplicación que no requiera recodificarse desde 0:

- Planear, Diseñar, Implementar y ejecutar un plan de pruebas a la aplicación en el ambiente de testing antes de migrarla. Ver: [Validación de la migración](#)
- Preparación de un entorno de desarrollo.
- Adaptar los archivos de configuración y descriptores de despliegue a la nueva arquitectura.
- Sustituir las dependencias incompatibles con la arquitectura destino por otros frameworks o librerías compatibles.
- Ejecutar el plan de pruebas a la aplicación migrada en el ambiente de testing, y comparar los resultados de salida con los resultados obtenidos en la ejecución del plan de pruebas sobre la aplicación antes de migrarla. Ver: [Validación de la migración](#)

- Documentación: Sobre todo la orientada a la certificación del resultado de la migración
- Traspaso de conocimiento necesario al SNA para que su equipo de desarrollo interno pueda llevar a cabo la mantención de la aplicación.
- Paso a producción de la aplicación.
- [Mejora continua y Mantenimiento](#)

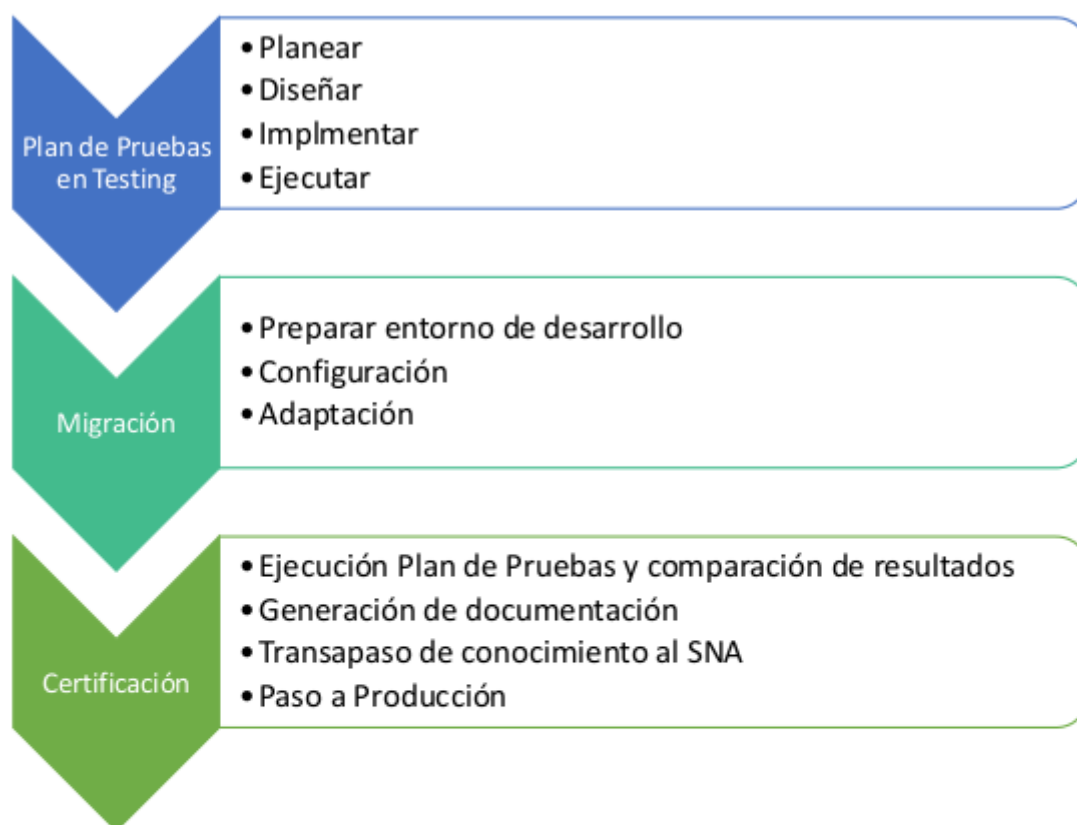


Diagrama 9: Plan de Migración

Ver [Plan técnico de Migración](#)

11.2 Recodificación de una aplicación

Se pueden establecer los siguientes hitos en la ejecución del plan de migración para una aplicación que deba recodificarse (que no deben confundirse con el ciclo de vida de un sistema o las fases de metodologías de desarrollo):

- Planear, Diseñar, Implementar y ejecutar un plan de pruebas a la aplicación en el ambiente de testing antes de recodificarla. Ver: [Validación de la migración](#)
- Elaboración de un Arquetipo de Desarrollo (solo aplica a las aplicaciones que requieran una codificación)
- Ajustes y reutilización de código / Abordar nuevos Desarrollos
- Ejecutar el plan de pruebas a la aplicación migrada en el ambiente de testing, y comparar los resultados de salida con los resultados obtenidos en la ejecución del plan de pruebas sobre la aplicación antes de migrarla. Ver: [Validación de la migración](#)
- Documentación: Sobre todo la orientada a la certificación del resultado de la migración.
- Traspaso de conocimiento necesario al SNA para que su equipo de desarrollo interno pueda llevar a cabo la mantención de la aplicación.
- Paso a producción de la aplicación.
- [Mejora continua y Mantenimiento](#)

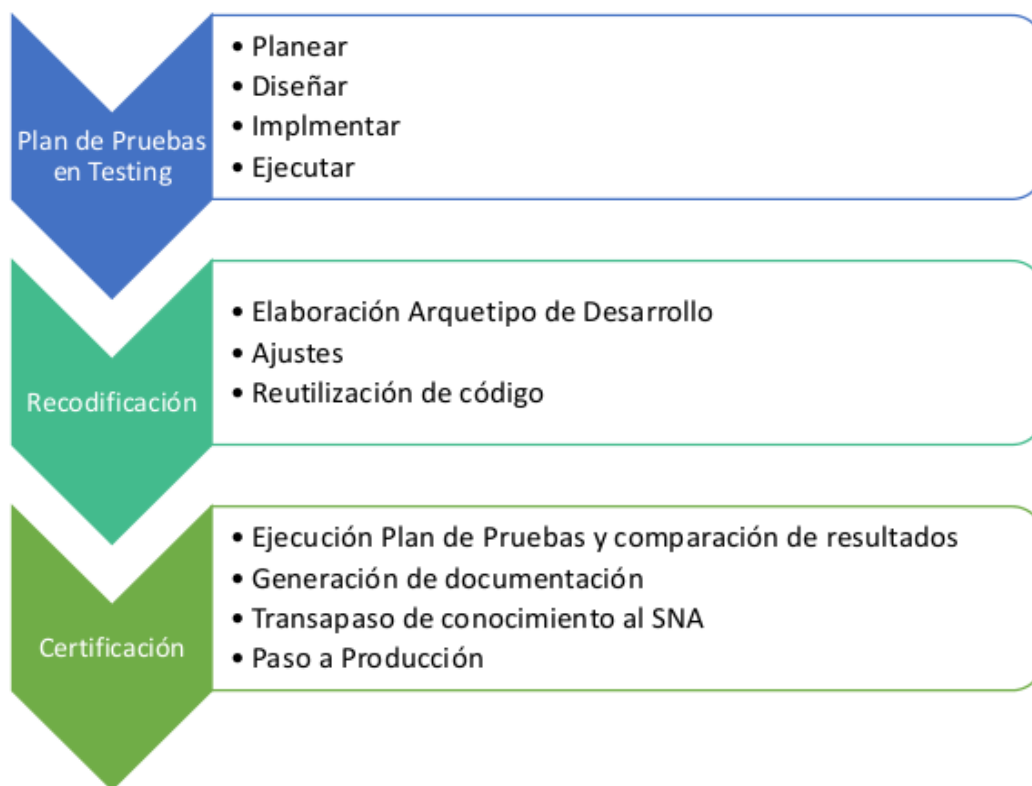


Diagrama 10: Recodificación

Ver [Plan técnico de Migración](#)

12 Validación de la migración

Cuando el producto está ajustado a la nueva plataforma, deberán establecerse ejecuciones de pruebas en ambientes de Desarrollo y Testeo antes de ejecutar el paso final al ámbito de explotación.

Un equipo de Calidad del SNA, será el encargado de llevar a cabo, todas las pruebas necesarias que garanticen que la migración de una aplicación, cumplen con los requisitos acordados. Por lo que toda entrega aplicación desplegada en el entorno testing, debe obtener la aprobación de dicho equipo. A continuación se enumeran las principales pruebas que todo software debería superar, junto con algunas de las herramientas a utilizar para facilitar la realización de estas pruebas.

El material generado en la fase Ejecución (Manuales de usuarios adaptados al nuevo sistema, Planes de pruebas que sirvan para certificar que el producto cubre, como mínimo, la funcionalidad que existía en la plataforma de origen, etc) será el sirva para determinar la validez del producto final antes de su paso a explotación. Puede que algunas de las aplicaciones no dispongan de los insumos necesarios para realizar la validación de la migración, por lo tanto, el equipo de Calidad del SNA, será el encargado de proporcionar la información necesaria para generar los insumos.

12.1 Pruebas Funcionales

Estas son las pruebas que se realizan con la aplicación desplegada y funcionando, probando cada una de las pantallas, comprobando su correcto funcionamiento. Estas pruebas son muy importantes, ya que éstas encuentran los errores que suelen ser detectados por los propios usuarios, durante el uso de la aplicación. En cada entrega se debe verificar el correcto funcionamiento de las distintas pantallas de la aplicación.

En primer lugar el proceso se realizará manualmente, pero utilizando distintas herramientas se podrá automatizar la ejecución de los distintos casos de prueba. Se debe realizar tantas pruebas como casos de prueba se detallan en la documentación, en todos los navegadores y versiones que se indiquen .

Para realizar estas pruebas se puede utilizar la herramientas Selenium, así como otros mecanismos que permitan emular diferentes entornos.

12.2 Pruebas de Rendimiento y Estrés

Se debe comprobar que la herramienta funciona correctamente bajo una carga de trabajo excepcional (muchos usuarios utilizando la aplicación simultáneamente). Para realizar estas pruebas se definen distintos casos de uso de la aplicación, que se lanzan simultáneamente para comprobar que la aplicación responde de manera satisfactoria.

También sirven para calcular el consumo mínimo por sesión por usuario, así como el consumo base del sistema, para de este modo poder proporcionar adecuadamente los entornos donde desplegar la aplicación.

Para realizar estas pruebas se recomienda utilizar la herramienta JMeter de Apache, WAPT, LoadRunner, etc...

13 Mejora Continua y Mantenimiento

La estrategia está orientada a una salida rápida de la obsolescencia de las aplicaciones del SNA por lo que esta etapa, **no entra dentro del alcance de la migración de aplicaciones**. En esta etapa se evaluarán las aplicaciones migradas con el objetivo de ser llevadas a la arquitectura propuesta o "ideal". (Ver: [ANEXO 1: Visión de la Arquitectura de Software de las Aplicaciones.](#))

La mejora continua incluye la Mantención de Aplicaciones por parte del equipo de desarrollo interno del SNA. A continuación se listan una serie de buenas prácticas a aplicar en la mejora continua:

13.1 Calidad del Código

Estas pruebas no se realizan directamente por parte del equipo de Calidad, sino que el entorno de desarrollo dispondrá una instalación de SonarQube que analizará el repositorio donde se aloja el código fuente, en la cual cada proyecto estará configurado según el libro de reglas que requiera o en su defecto el perfil de calidad marcado por defecto. También los propios desarrolladores pueden disponer de plugins en Eclipse que le permiten ir controlando estas cuestiones durante el desarrollo.

13.2 Pruebas Unitarias

Pruebas destinadas a verificar la funcionalidad y estructura de cada componente individualmente una vez que ha sido codificado. En caso de no existir, se tendrían que codificar pruebas unitarias para los componentes de las aplicaciones.

Estas pruebas no se ejecutan como tal por parte del equipo de Calidad sino que el proceso de integración continua de Jenkins, el cual se encarga de valorar periódicamente los test.

Las herramientas mas utilizadas para realizar estos test, son JUnit, Arquillian y TestNG.

13.3 Pruebas de Integración

Pruebas destinadas a verificar el correcto ensamblaje entre los distintos componentes una vez que han sido probados unitariamente con el fin de comprobar que interactúan correctamente a través de sus interfaces, tanto internas como externas, cubren la funcionalidad establecida y se ajustan a los requisitos no funcionales especificados en las verificaciones correspondientes.

Se utilizan las mismas herramientas comentadas en el apartado Pruebas Unitarias.

13.4 Pruebas de Regresión

Pruebas destinadas a comprobar que los cambios sobre un componente de un sistema de información, no introducen un comportamiento no deseado o errores adicionales en otros componentes no modificados.

El conjunto de pruebas que se planifiquen con este objetivo pueden estar formado tanto por pruebas automatizadas como por pruebas manuales.

13.5 Pruebas de Accesibilidad

La accesibilidad es el grado en el que todas las personas pueden utilizar un objeto, visitar un lugar o acceder a un servicio, independientemente de sus capacidades técnicas, cognitivas o físicas. La accesibilidad web se refiere a la capacidad de acceso a la Web y a sus contenidos por todas las personas independientemente de la discapacidad (física, intelectual o técnica) que presenten o de las que se deriven del contexto de uso (tecnológicas o ambientales).

Hay herramientas de evaluación que ayudan a realizar evaluaciones de accesibilidad, como por ejemplo Total Validator.

13.6 Pruebas de Vulnerabilidad y Seguridad

Gracias a estas pruebas, se puede identificar los puntos débiles de su seguridad lógica en aplicaciones e implementar las medidas de protección necesarias, además de evaluar y auditar su política de seguridad actual.

Para este tipo de pruebas se puede utiliza W3af.

13.7 Otros Controles

Existen otros controles que se pueden optimizar, como son la apariencia de la aplicación, la homogeneidad en los estilos, la existencia de ayuda contextual, la usabilidad de la aplicación. Estos son criterios subjetivos, que serán valorados por los integrantes del equipo de Calidad.

14 Análisis 80/20

El objetivo del análisis 80/20 es dotar de herramientas simples de decisión, de manera que al aplicar restricciones o criterios particulares (ej. Presupuesto, disponibilidad de recursos, nivel de riesgo aceptado, etc.), se pueda determinar el orden en que un conjunto de aplicaciones/escenarios/grupos deben ser abordados en la implementación de la estrategia de migración.

De esta forma los criterios usados deben apoyar la priorización de las aplicaciones cuya migración prioritaria aporten el mayor valor vs riesgos/costos al Servicio.

Partiendo de los [Indicadores de las aplicaciones](#) surgidos del levantamiento de información y el análisis de brecha, se ha calculado un factor de importancia para cada aplicación considerando la complejidad técnica y la importancia para el negocio el negocio:

$$\text{Factor de Importancia de Migración} = (\text{Negocio} * 0.5) + ([\text{Mapa Calor}][\%] * 0.5)$$

La criticidad en el negocio se ha considerado para contrarrestar la complejidad técnica, priorizando así aplicaciones que son importantes para el negocio y a su vez complejas de migrar técnicamente:

- Negocio = 10 (Importancia Alta)
- Negocio = 50 (Importancia Media)
- Negocio = 100 (Importancia Baja)

La ordenación ascendente de este ranking determina las aplicaciones más importantes de migrar desde un punto de vista técnico y de negocio.

Rankig de Importancia de Migración	Aplicación	Factor de Importancia de Migración	% Acumulado 80/20
1	61-MensajeríaManifiestos	5,14	1,8%
2	23-DIPS_COURIER	5,49	3,6%
3	24-F18_POSTAL	5,54	5,4%
4	64-ParDusGuia	5,64	7,1%
5	26-Mensajería_DIN	5,83	8,9%
6	55-MicWeb	8,67	10,7%
7	29-Consultas_DUS	8,77	12,5%
8	74-SistemaSelectividad	11,24	14,3%
9	33-DIPSCF	15,23	16,1%
10	10-Control_Mensajería	18,85	17,9%
11	21-DECARE	22,81	19,6%
12	39-WSAA_Chile	25,35	21,4%
13	51-Servicio_Web_Canje_Maritim o	25,45	23,2%

Rankig de Importancia de Migración	Aplicación	Factor de Importancia de Migración	% Acumulado 80/20
14	54-WSAlmacen	25,45	25,0%
15	68-Interoperabilidad_MIC_Argentina	25,8	26,8%
16	06-VehiculosRACWeb	25,82	28,6%
17	38-WSAA_con_Argentina	25,84	30,4%
18	58-WSAlmacen_y_OT	25,91	32,1%
19	34-DIPS_Viajeros	26,09	33,9%
20	22-SICOMEXIN	26,18	35,7%
21	05-SWVehiculos	26,98	37,5%
22	62-ListaPasajeros	28,07	39,3%
23	36-WebSeguridad	28,7	41,1%
24	08-SCVM	29,8	42,9%
25	66-SAM	30,4	44,6%
26	04-Veh_Integrado	31,46	46,4%
27	37-Web_AdmPersonas	32,53	48,2%
28	47_DepWebFiscalizacion	32,53	50,0%
29	46-SIDECO	35,23	51,8%
30	44-SDA	36,18	53,6%
31	48_SIFER	37,84	55,4%
32	75-SRS	38,64	57,1%
33	35-DTI	40,28	58,9%
34	50-SIDEMAR	41,42	60,7%
35	57-VisualSMS	41,89	62,5%
36	69-GestionDocumental	44,87	64,3%
37	60-SMS_Mensajeria	50,02	66,1%
38	45-SVC	50,36	67,9%
39	40-INDIRA	50,37	69,6%

Rankig de Importancia de Migración	Aplicación	Factor de Importancia de Migración	% Acumulado 80/20
40	67-SeguridadWeb	50,45	71,4%
41	52-Aplicacion_Web_Canje_Marítimo	50,68	73,2%
42	93-PortalExportadorWeb	50,72	75,0%
43	09-VehiculosFronteraWeb	50,82	76,8%
44	42-Firmas_ALADI	50,95	78,6%
45	88-Programacion_Naviera	52,56	80,4%
46	63-PortalRayosX	54,12	82,1%
47	49_SIROFE	54,97	83,9%
48	56_SIROTE	55	85,7%
49	90-Acopio_Almacenistas	55	87,5%
50	97-SmdtWS	55	89,3%
51	87-GCP-F09	55,76	91,1%
52	43-TramitacionIVV	63,1	92,9%
53	01-SIGES	74,94	94,6%
54	02-OPVIG	75	96,4%
55	27-DepositoFranco-Intranet	89,91	98,2%
56	76-ConsultaMICTransBolivia	100	100,0%

El Principio de Pareto o Análisis 80/20 determina que: Haciendo el esfuerzo de migrar el 20% de las aplicaciones, ordenadas por su Ranking de Importancia de Migración, se obtendrá el 80% de satisfacción para el SNA en su proceso de migración de aplicaciones. Estas aplicaciones se corresponden a las que se encuentran en formato **negrita** en la tabla anterior.

En el documento **CMAA - MIGAPPS- Ranking Aplicaciones_(FormatoExcel)_20160322.xlsx** se puede encontrar el detalle de los cálculos realizados.

15 ANEXO 1: Visión de la Arquitectura de Software de las Aplicaciones.

15.1 Arquitectura Actual

La arquitectura actual del SNA es un modelo de n-capas, donde se identifican básicamente las capas: Presentación, Negocios y Datos. Por otro lado, la arquitectura está compuesta por una capa de servicios e integración con sistemas externos e internos. Transversalmente a todos los sistemas se encuentran las capas de auditoría y seguridad.

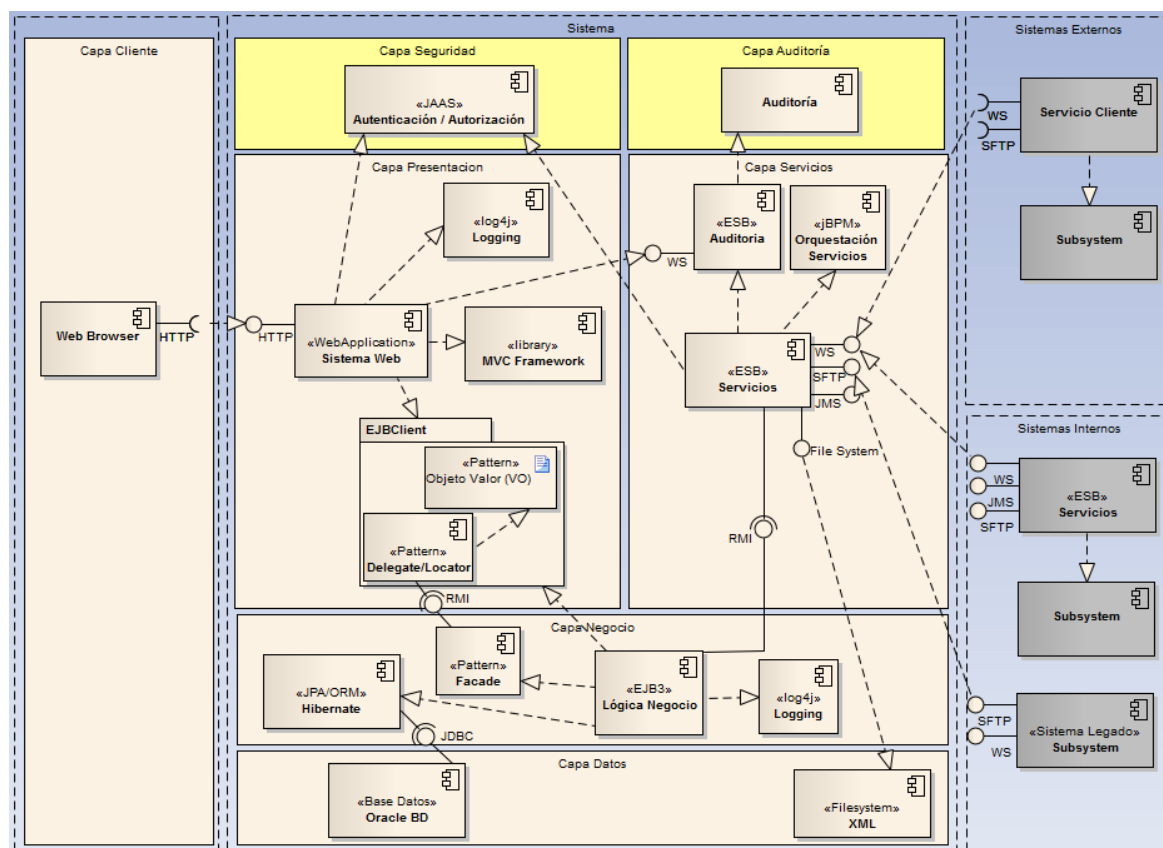


Diagrama 11: Arquitectura Actual

En el documento anexo **Arquitectura_de_software_SNA_v1.0.3.0.pdf** se encuentra la descripción detallada de cada capa.

15.2 Arquitectura Propuesta

15.2.1 Propuesta de mejoras sobre la arquitectura actual

Conforme a la arquitectura actual, se proponen una serie de mejoras para aumentar la facilidad de uso de la aplicación, haciendo uso en mayor medida de framework acordes con la arquitectura propuesta:

Como normas técnicas para la seguridad de las aplicaciones se propone:

- Uso del framework **Picketlink**¹. Picketlink es el framework de JBoss que provee a las aplicaciones (de la capa de presentación) la interfaz de autenticación con el servicio de identidad mediante JAAS y los mecanismos para autorizar la aplicación.
- La comunicación entre la capa de presentación y los servicios SOA debe ser cifrado mediante SSL bidireccional. Esto asegura la confidencialidad del mensaje entre origen y destino, asegurando que

1 <http://picketlink.org/>

el mensaje no ha sido modificado desde el origen o el destino. Asimismo se recomienda aplicar cifrado SSL en la comunicación entre el usuario y la capa de presentación, en este caso únicamente en el lado servidor, evitando que un atacante pueda capturar información privilegiada.

- Para auditar las modificaciones en la capa de datos se recomienda el uso de Hibernate Envers², donde en cada registro de base de datos auditado se incluya el identificador del usuario autenticado.

Mejoras en la lógica de negocio de la aplicación:

- Utilizar un proyecto maven padre para toda la suite de aplicaciones donde se establezcan las versiones de las librerías utilizadas (jsf, picketlink, etc.). De esta forma, ante la aparición de errores de seguridad en una librería/framework la actualización en los aplicativos se realiza en un único punto.
- Uso de CDI como especificación de inyección entre capas (conocido como “magic glue”). CDI nos aparta a la arquitectura independencia entre los servicios DAO de acceso a la capa de datos/SOA y los beans de la capa de presentación.

Para la capa de presentación, se recomienda el uso de:

- Hacer uso de PrettyFaces³ (Rewrite en su última versión) para configurar en los proyectos JSF el uso de URLs amigables. Estas urls amigables mejoran la integración del usuario con el proyecto puesto que se pueden añadir a favoritos, invocar enlaces directamente, etc.
- Uso del framework Bootstrap para aportar a la solución un framework responsivo según el dispositivo de entrada. Para los frameworks de librerías gráficas de JSF puede valorarse el uso de Primefaces PRO o similar que ya integran la salida de forma nativa en formato responsivo.

15.2.2 Propuesta arquitectura alternativa. frontend – backend

Una de las **tendencias** actuales en los desarrollos de aplicaciones web es el desacoplamiento de la capa de presentación con la capa de servicio. Mediante los frameworks tradicionales MVC de desarrollo de aplicaciones web, no existe un desacoplamiento claro entre la capa de negocio y la capa de presentación. Asimismo, existe una gran dependencia entre los componente visuales de JSF (richfaces, primefaces, etc.) con el HTML resultante, donde se pierde el control directo sobre el HTML generado.

La capa de frontend o cliente sería la aplicación cliente basada en HTML5 + CSS3 nativo, con control total sobre la vista, en la que se encuentra tanto la vista como parte del controlador, y mediante el uso de librerías javascript se invoca a la lógica de negocio con llamadas en javascript. Algunos frameworks conocidos para esta capa son AngularJS, BackboneJS, etc.

La capa de Backend o servidor serían un conjunto de servicios REST que son invocados desde el cliente para obtener/enviar los datos desde la capa de presentación. Estos servicios residirían en el servicio SOA de la arquitectura actual.

Esta arquitectura es utilizada por la mayoría de los proyectos actuales de software como pueden ser Facebook, Twitter, Google apis, etc.

Esta arquitectura presenta las siguientes **ventajas**:

- La capa de presentación no se genera en servidor, reduciendo el consumo de cpu y memoria que esto conlleva en los casos de JSF. Toda la lógica de presentación se sirve una única vez y se altera mediante javascript.
- Se aumenta el control sobre el resultado final de la solución, eliminando capas de html auto-generado.

Las principales **desventajas** de esta alternativa son:

- Los servicios de acceso a datos deben ser públicos desde la web, requiriendo de un control férreo

² <http://hibernate.org/orm/envers/>

³ <http://www.ocpssoft.org/prettyfaces/>

de la autenticación/autorización. En caso de utilizar esta arquitectura debe tratarse como un punto crítico.

- No son una tecnología estándar, cada framework trabaja según su arquitectura
- El tiempo de aprendizaje es mayor, así como el coste de mantenimiento.

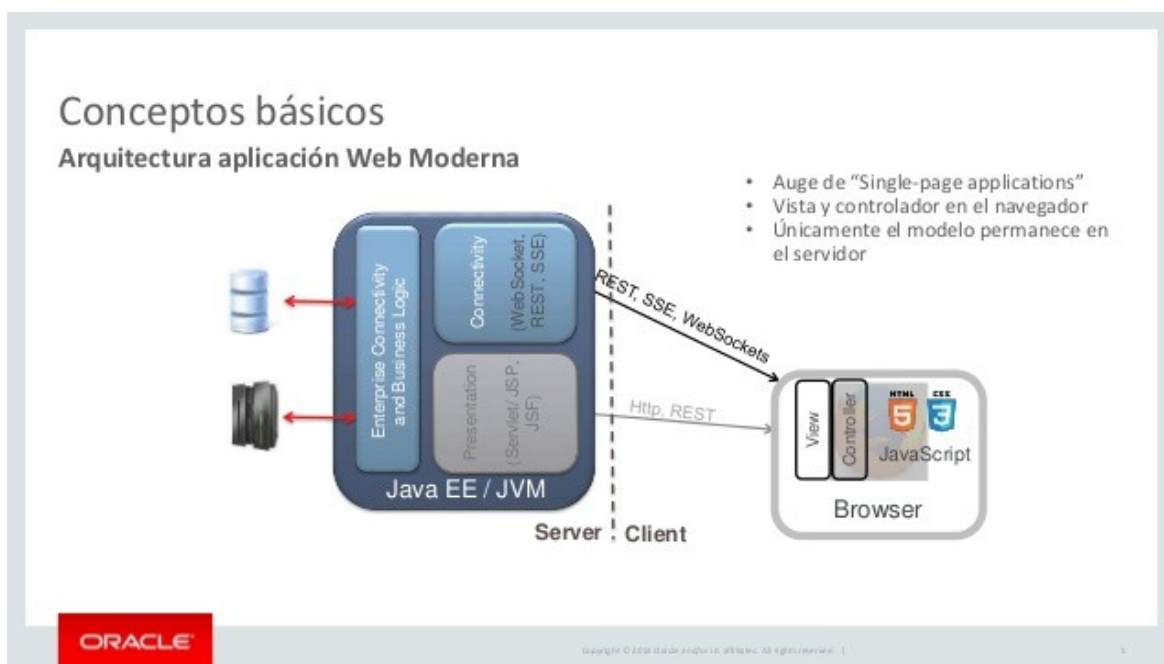


Diagrama 12: Arquitectura Web Moderna

16 ANEXO 2: Plan técnico de Migración

En esta etapa se definen los detalles técnicos y aspectos a tener en cuenta para la migración de las aplicaciones pertenecientes a un grupo.

16.1 Grupo Weblogic

16.1.1 Puntos de Análisis

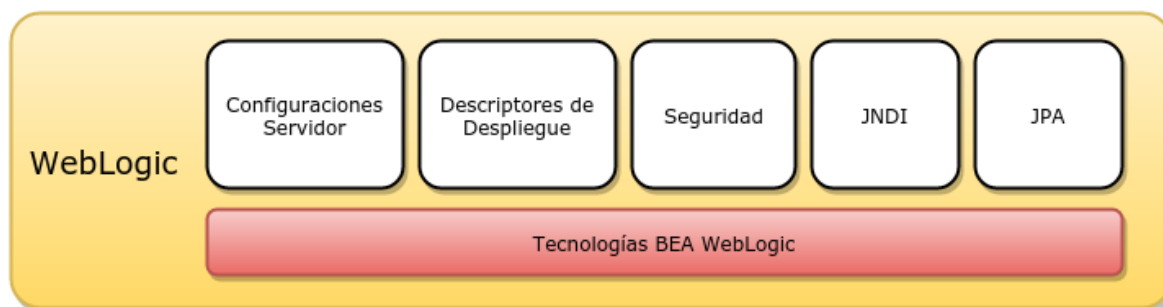


Diagrama 13: Puntos de Análisis WebLogic

16.1.2 Configuración del Servidor

En WebLogic la configuración del Servidor está en el archivo:

WEBLOGIC_HOME/domains/DOMAIN_NAME/config/config.xml

Contiene los parámetros de configuración de cada instancia, cluster, fuente y servicio dentro del dominio. En la mayoría de los casos la consola de administración o alguna herramienta de Weblogic para modificar el dominio, escribe sobre este archivo.

En JBoss EAP 6.4 el servidor puede ejecutarse como un standalone server o managed domain. En el caso de que se esté ejecutando como standalone server el fichero de configuración está en:

JBOSS_HOME/standalone/configuration/standalone.xml

En el caso de que el servidor esté ejecutándose como managed domain los ficheros de configuración están en:

JBOSS_HOME/domain/configuration/domain.xml

JBOSS_HOME/domain/configuration/host.xml

La Management Console o Management Command Line Interface (CLI) escriben directamente sobre estos archivos.

Se deberá explorar cada sección del fichero de configuración config.xml para encontrar su sección equivalente en standalone.xml, domain.xml o host.xml.

16.1.3 Descriptores de Despliegue

16.1.3.1 Mapeo de Archivos de Configuración

Es difícil encontrar una equivalencia entre los archivos de configuración de WebLogic con los de JBoss pero la siguiente tabla contiene el mapeo de algunos:

Archivo WebLogic	Equivalencia en JBoss	Descripción
ejb-jar.xml weblogic-ejb-jar.xml, weblogic-cmp-rdbms-jar.xml	jboss-ejb3.xml standalone.xml domain.xml	Especifica los elementos del EJB que son únicos en WebLogic Server

weblogic.xml	standalone.xml domain.xml	Especifica características que no están en la configuración estándar.
*-jdbc.xml	standalone.xml domain.xml	Especifica los datasources.

16.1.3.2 Reemplazar el Descriptor de Despliegue

El archivo **weblogic-ejb-jar.xml** especifica los elementos que son propios del Servidor WebLogic. El archivo equivalente en JBoss es **jboss-ejb3.xml**. A continuación se listan los mapeos para la conversión de los elementos entre ambos archivos.

weblogic-ejb-jar.xml XPath	jboss-ejb3.xml XPath
WEBLOGIC_PREFIX/ejb-name	JBOSS_PREFIX/ejb-name
WEBLOGIC_PREFIX/resource-description/res-ref-name	JBOSS_PREFIX/resource-ref/res-ref-name
WEBLOGIC_PREFIX/resource-env-description/res-env-ref-name	JBOSS_PREFIX/resource-env-ref/resource-env-ref-name
WEBLOGIC_PREFIX/ejb-reference-description/ejb-ref-name	JBOSS_PREFIX/ejb-ref/ejb-ref-name
WEBLOGIC_PREFIX/service-reference-description/service-ref-name	JBOSS_PREFIX/service-ref/service-ref-name
WEBLOGIC_PREFIX/service-reference-description/wsdl-file	JBOSS_PREFIX/service-ref/wsdl-file

Donde:

- WEBLOGIC_PREFIX corresponde a la siguiente ruta: /weblogic-ejb-jar/weblogic-enterprise-bean para todos los beans.
- JBOSS_PREFIX corresponde a una de las siguientes rutas dependiendo del tipo de bean:
 - Entity Bean: /ejb-jar/enterprise-beans/entity
 - Session Bean: /ejb-jar/enterprise-beans/session
 - Message-Driven Bean: /ejb-jar/enterprise-beans/message-driven

Existe algunas configuraciones dentro del archivo weblogic-ejb-jar.xml que deben ser trasladadas al archivo jboss-cmp-jdbc.xml:

delay-updates-until-end-of-tx

Por defecto es True y se utiliza para retrasar todas las actualizaciones de persistencia sobre la base de datos hasta el final de la transacción. Si está establecido a False entonces las actualizaciones son efectuadas

cuando la actualización es invocada, pero el commit no se realiza hasta el final de la transacción. Esto permite a otros procesos acceder a los datos actualizados mientras la transacción todavía está pendiente de ser completada.

```
<jbosscmp-jdbc>
...
<entity>
  <ejb-name>Student</ejb-name>
  <create-table>true</create-table>
  <remove-table>true</remove-table>
  <table-name>STUDENT</table-name>
  ...
  <sync-on-commit-only>true</sync-on-commit-only>
  <insert-after-ejb-post-create>true</insert-after-ejb-post-create>
  <call-ejb-store-on-clean>true</call-ejb-store-on-clean>
</entity>
</jbosscmp-jdbc>
```

16.1.3.3 Migrar Archivos de Configuración

El archivo de configuración **weblogic.xml** es usado para especificar características de las aplicaciones web que no están incluidas en la configuración estándar del servidor.

weblogic.xml	Equivalencia en JBoss
Contexto Raíz: <context-root> MyAppContextRoot </context-root>	Archivo jboss-web.xml: <jboss-web> <context-root>MyAppContextRoot</context-root> </jboss-web>
Mapeo de Roles: <security-role-assignment> <role-name>TestRole1</role-name> <principal-name>TestRole1</principal-name> </security-role-assignment>	En el archivo de configuración del servidor usar el módulo RoleMappingLoginModule : <security-domain name="FormBasedAuthWebAppPolicy"> <authentication> \\ <login-module code="org.jboss.security.auth.spi.DatabaseServerLoginModule" flag="required"> <module-option name="dsJndiName" value="java:/MySQLDS"/> <module-option name="principalsQuery" value="select password from PRINCIPLES where

```
principal_id=?"/>
```

```
    <module-option name="rolesQuery"
value="select user_role, 'Roles' from ROLES where
principal_id=?"/>
```

```
</login-module>
```

```
<!--- Following is the RoleMappingLoginModule
!-->
```

```
<login-module
code="org.jboss.security.auth.spi.RoleMappingLoginMo
dule" flag="optional">
```

```
    <module-option name="rolesProperties"
value="/home/userone/jboss-eap-6.0-
GA/standalone/configuration/test-roles.properties"/>
```

```
    <module-option name="replaceRole"
value="false"/>
```

```
</login-module>
```

```
</authentication>
```

```
</security-domain>
```

Después añadir en el **archivo jboss-web.xml** lo siguiente:

```
<jboss-web>
```

```
    <security-
domain>java:/jaas/FormBasedAuthWebAppPolicy</sec
urity-domain>
```

```
</jboss-web>
```

Especificar **Sesión Timeout**:

```
<session-descriptor>
```

```
    <session-param>
```

```
        <param-name>TimeoutSecs</param-
name>
```

```
        <param-value>3650</param-value>
```

```
    </session-param>
```

```
</session-descriptor>
```

Añadir lo siguiente en **WEB-INF/jboss-web.xml**

```
<session-config>
```

```
    <session-timeout>3650</session-timeout>\
```

```
</session-config>
```

Especificar **class loading precedence**:

```
<container-descriptor>
```

```
    <save-sessions-enabled>true</save-
sessions-enabled>
```

```
    <prefer-web-inf-classes>true</prefer-web-inf-
classes>
```

No hay especificación conocida dentro de JBoss, ya que el WAR tiene su propio ámbito de class loader y las clases empaquetadas en WEB-INF/lib y en WEB-INF/classes siempre serán cargadas en la aplicación web.

</container-descriptor>	
Configuración JSPs	Configuración JSPs en la sección subsystem del archivo de configuración del servidor.
<jsp-descriptor>	<subsystem xmlns="urn:jboss:domain:web:1.1" default-virtual-server="default-host" native="false">
<jsp-param>	<configuration>
<param-name>keepgenerated</param-name>	<jsp-configuration development="true" java-encoding="ISO8859_1" keep-generated="true"/>
<param-value>true</param-value>	</configuration>
</jsp-param>	<connector name="http" protocol="HTTP/1.1" scheme="http" socket-binding="http"/>
<jsp-param>	<virtual-server name="default-host" enable-welcome-root="true">
<param-name>encoding</param-name>	<alias name="localhost"/>
<param-value>ISO8859_1</param-value>	<alias name="example.com"/>
</jsp-param>	</virtual-server>
</jsp-descriptor>	</subsystem>

El archivo de configuración **weblogic-application.xml** es usado para especificar los archivos EAR de Weblogic. No existe mapeo directo de este descriptor pero muchas de las especificaciones pueden ser configuradas en archivos de la arquitectura JEE.

weblogic-application.xml	Equivalencia en JBoss
<application-param> element:	Maps to the web.xml <context-param> element:
<application-param>	<context-param>
<description>	<description>
Web application default encoding	Web application default encoding
</description>	</description>
<param-name>	<param-name>
webapp.encoding.default	webapp.encoding.default
</param-name>	</param-name>
<param-value>	<param-value>
UTF8	UTF8
</param-value>	</param-value>
</application-param>	</context-param>

16.1.4 Seguridad

En el ámbito de este documento, que corresponde al grupo de aplicaciones priorizadas, no se hace uso de componentes de seguridad propietarios de Oracle o WebLogic como por ejemplo: WebLogic Oracle Wallets o WebLogic Certicom-based SSL Implementation.

16.1.5 WebLogic JNDI Lookups a Portable JNDI

En Weblogic, se crea un Hashtable para contener la información del entorno con parámetros tales como:

- URL del servidor
- Context Factory
- Security principal (usuario)
- Security credential (password)

Búscar en el proyecto código semejante a este:

```
Hashtable<String, String> env = new Hashtable<String, String>();  
env.put( Context.PROVIDER_URL, "t3://localhost:7001" );  
env.put( Context.INITIAL_CONTEXT_FACTORY, "weblogic.jndi.WLInitialContextFactory" );  
env.put( Context.SECURITY_PRINCIPAL, "weblogic" );  
env.put( Context.SECURITY_CREDENTIALS, "weblogic" );  
Context context = new InitialContext( env );  
Service service = (Service)context.lookup( "sample.Service#" + Service.class.getName() );
```

Reemplazar por código semejante a este, usando reglas portables para JNDI lookup tales como: java:global, java:app, java:module.

```
Context context = new InitialContext();  
Service service = (Service) context.lookup( "java:app/service/" + ServiceImpl.class.getSimpleName() );
```

Donde:

- java:global se utiliza cuando será compartido en todas las aplicaciones desplegadas en la instancia del servidor de aplicaciones.
- java:module se utiliza cuando será compartido en todos los componentes en un modulo EJB o Web.
- java:app se utiliza cuando está compartido en todos los módulos de una aplicación.

El reporte de WindUp proporciona información de donde existen declaraciones JNDI y propone una alternativa de reemplazo válida para JBoss.

16.1.6 ServletAuthentication

En el ámbito de este documento, que se corresponde al grupo de aplicaciones priorizadas, no se hace uso de la autenticación propietaria de WebLogic.

16.1.7 WebLogic Transaction Manager

En el ámbito de este documento, que corresponde al grupo de aplicaciones priorizadas, no se hace uso del manejo de transacciones propietaria de WebLogic.

16.1.8 Hibernate y JPA

En el ámbito de este documento, que corresponde al grupo de aplicaciones priorizadas, no se hace uso de Hibernate ni de JPA.

16.1.9 Reemplazar tecnologías propietarias de Weblogic

16.1.9.1 WebLogic CommonJ Framework

En el ámbito de este documento, que corresponde al grupo de aplicaciones priorizadas, no hay referencias a WebLogic Work Manager API (a veces referenciado por WebLogic Work Manager API) que no está soportado por JBoss Application Server.

16.1.9.2 WebLogic Virtual Directories

En el ámbito de este documento, que corresponde al grupo de aplicaciones priorizadas, no hay referencias <virtual-directory-mapping>

16.1.9.3 Anotaciones Propias de Weblogic

Reemplazar las siguientes anotaciones:

WebLogic	Java EE 6	Java EE 6 Import
@WLServlet	@WebServlet	javax.servlet.annotation.WebServlet
@WLFilter	@WebFilter	javax.servlet.annotation.WebFilter
@WLInitParam	@WebInitParam	javax.servlet.annotation.WebParam

Atributos de la anotación @WLServlet

WebLogic	Java EE 6	Descripción
String displayName	String displayName	Nombre del Servlet para visualización
String description	String description	Descripción del Servlet
String icon	String smallicon String largeicon	Ruta al icono
String name	String name	Nombre del servlet
WLInitParam[] initParams	WebInitParam[] initParams	Parámetros de iniciación
int loadOnStartup	int loadOnStartup	Cargar el servlet cuando se inicia el servidor
String runAs	No hay atributo equivalente	Usar javax.annotation.security.RunAs en la clase Definir el elemento run-as para el servlet en web.xml Definir el elemento urn-as en un archivo web- fragments.xml
String[] mapping	String[] urlPatterns String[] value	Mapeo de url del servlet

Atributos de la anotación @WLFiter

WebLogic	Java EE 6	Descripción
String displayName	String displayName	Nombre del Servlet para visualización
String description	String description	Descripción del Servlet
String icon	String smallicon String largeicon	Ruta al icono
String name	String name	Nombre del servlet
WLInitParam[] initParams	WebInitParam[] initParams	Parámetros de iniciación
String[] mapping	String[] urlPatterns String[] value	Mapeo de url del servlet
No hay atributo equivalente	DispatcherTypes[] dispatcherTypes	Tipos de Dispatcher a los que aplica el filtro
No hay atributo equivalente	String[] servletNames	Nombre de los Servlet que aplica el filtro

Atributos de la anotación @WLInitParam

WebLogic	Java EE 6	Descripción
String name	String name	Nombre de iniciación del parámetro
String value	String value	Valor de iniciación del parámetro
No existe atributo equivalente	String description	Opcional descripción de iniciación del parámetro

Por otro lado se puede aprovechar una alto porcentaje del código reemplazando las anotaciones propias de WebLogic por anotaciones pertenecientes a estándares como JPA y CDI o creando algunas anotaciones propias en el caso de que los estándares no puedan cubrirlos. A continuación se identifican las anotaciones y el equivalente (dentro del ámbito de las 5 aplicaciones priorizadas):

Anotación	Teconolgia	Equivalente	Tecnología
@common:control	Weblogic controls	@Inject	WELD
@jc:sql statement="..."	Database controls	Anotación Propia	Se necesitará construir un Interceptor que sea capaz de ejecutar statement sobre la base

			de datos.
@jc:connection data-source-jndi-name="..."	Database controls	@Resource	javax.annotation
@editor-info:code-gen control-interface="true"	Java control	Generación de código automática mediante Java Annotation Processors	javax.annotation.processing.SupportedAnnotationTypes javax.annotation.processing.SupportedSourceVersion javax.annotation.processing.SupportedOptions

La generación de scripts que hagan sustituciones mediante el uso de expresiones regulares podría facilitar esta tarea.

16.1.9.4 NetUI PageFlow

Aunque está deprecado desde el 01/11/2010, Apache Beehive, es la forma más rápida de eliminar las dependencias de Bea System de una aplicación.

Se ha llevado a cabo una prueba de concepto (PoC) para migrar una aplicación (no toda, sino algunos de sus componentes) de Welogic 8.1 a JBoss EAP 6.4, el estudio detallado se encuentra en el documento anexo: **CMAA_Prueba_de_Concepto.pdf**.

De forma general la prueba de concepto ha consistido en los siguientes pasos:

1. Usar el asistente de Workshop 9.2 para importar el proyecto eliminando las dependencias de netui y pageflows las cuales son reemplazadas por Apache Beehive.
2. Usar el IDE actual del SNA (Eclipse 4.4 Luna) para poder desplegar un ear en el servidor de desarrollo JBoss EAP 6.4 y replicar proyectos de Workshop.
3. Instalar Apache Beehive en los proyectos que tengan dependencias de este framework.
4. Resolver manualmente errores de compilación derivados de las librerías faltantes en el IDE.
5. Ejecutar un ciclo web completo. (Presentación <-> Negocio <-> Acceso a Datos <-> BD)
6. Ejecutar un ciclo webservice completo. (Interfaz WebService <-> Negocio <-> Acceso a Datos.
7. Entrega de código fuente migrado y archivo ear generado corriendo en JBoss 6.4

Conclusiones:

Existe un asistente en el IDE Workshop 9.2 que elimina las dependencias de BeaSystems, y las reemplaza por el framework Apache Beehive. Este asistente realiza gran parte del trabajo de transformación del código fuente, pero no se consigue una aplicación totalmente independiente de Weblogic y hay que seguir transformando el código fuente (manualmente en el alcance de la PoC).

En los servicios web que una aplicación web provee se han encontrado problemas con la integración de Apache Beehive en Jboss 6.4 debido al classloader que implementa Jboss.

JBoss carga los webservices anotados con JSR181 @WebService (la misma anotación que utiliza Beehive) pero lo hace desde su propio módulo. Al cargarse desde el módulo de JBoss, los interceptores de Beehive no se encuentran en el cargador de clases del flujo de las peticiones, por lo que la inyección de dependencias no se realiza. Se ha intentado deshabilitar el módulo de jax-ws de JBoss para que se cargue desde el war donde está Beehive sin éxito.

Consecuencia de lo anterior se ha optado por reemplazar las anotaciones @Control por inyección CDI-WELD

y los interceptores @JdbcControl por interceptores de construcción propia para las operaciones con la base de datos.

Como información adicional: Existen algunas alternativas a Apache Beehive, como es Spring WebFlow, pero implicaría una recodificación de las aplicaciones (algo que no se espera en la salida de la obsolescencia).

16.2 Grupo Web Services

16.2.1 Puntos de Análisis

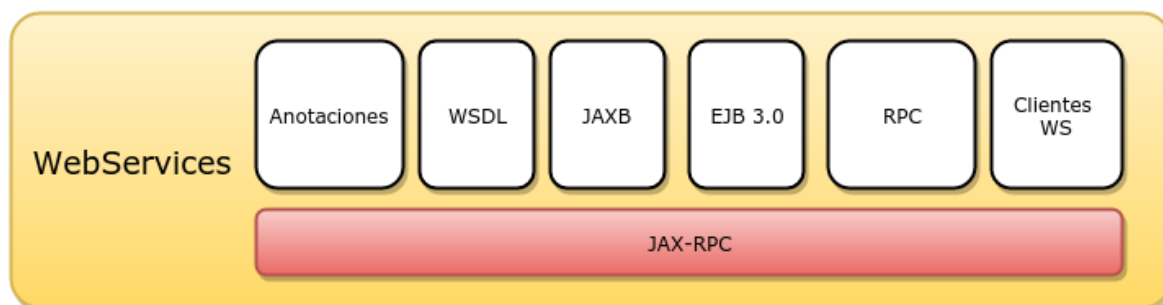


Diagrama 14: Puntos de Análisis Web Services

16.2.2 Servicios Web Específicos de Weblogic

Se puede aprovechar una alto porcentaje del código reemplazando las anotaciones propias de WebLogic por anotaciones pertenecientes a estándares como JAX-WS. A continuación se identifican las anotaciones y el equivalente (dentro del ámbito de las 5 aplicaciones priorizadas):

Anotación	Tecnología	Equivalente	Tecnología
@common:target-namespace namespace="..."	Anotaciones Específicas de WebLogic	@WebService	javax.jws.WebService Web Services Metadata Annotations (JSR-181)
@jws:protocol soap-style="rpc" http-xml="false" form-post="true" http-soap="true"	Anotaciones Específicas de WebLogic	@SOAPBinding	javax.jws.soap.SOAPBinding Web Services Metadata Annotations (JSR-181)
@common:target-namespace namespace="..."	Anotaciones Específicas de WebLogic	@WebService	javax.jws.WebService Web Services Metadata Annotations (JSR-181)
@jws:protocol soap-style="rpc" http-xml="false" form-post="true" http-soap="true"	Anotaciones Específicas de WebLogic	@SOAPBinding	javax.jws.soap.SOAPBinding Web Services Metadata Annotations (JSR-181)

La generación de scripts que hagan sustituciones mediante el uso de expresiones regulares podría facilitar esta tarea.

Los archivos de configuración y descriptores de despliegue deben ser sustituidos tal y como se especificó en el apartado [Descriptores de Despliegue](#) de WebLogic.

16.2.3 Servicios Web JAX-RPC

16.2.3.1 Ajuste del contexto raíz

En los servicios JAX-RPC, se puede utilizar las anotaciones `@WLXXXTransport` para especificar la raíz de contexto. Estas anotaciones no son válidas para los servicios Web JAX-WS y deberán ser sustituidas por fórmulas compatibles como el atributo `contextPath` del elemento `<module>`

16.2.3.2 Anotaciones reutilizables

Sólo se podrán reutilizar las siguientes anotaciones:

- `@Policy`
- `@Policies`
- `@SecurityPolicy`
- `@SecurityPolicies`
- `@WssConfiguration`

16.2.3.3 Fichero WSDL

Cuando se ejecuta el archivo `jwsc` en un servicio web JAX-RPC, un archivo WSDL es generado en el directorio de salida especificado. Para los servicios JAX-WS, el archivo WSDL es generado cuando se despliega el endpoint del servicio. Para generar un archivo WSDL en el directorio de salida, debe especificarse el atributo `wSDLOnly` del `jws`.

16.2.3.4 Uso de tipos JAXB

JAX-WS utiliza Java Architecture for XML Binding (JAXB). El uso actual de tipos personalizados ofrecidos por XMLBeans o Tylar tendrá que ser sustituido por los tipos ofrecidos por JAXB.

16.2.3.5 EJB 3.0

JAX-WS soporta EJB 3.0, mientras que JAX-RPC soporta sólo hasta EJB 2.1.

EJB 3.0 introdujo metadatos que permiten la generación automática de las clases e interfaces y del descriptor de despliegue que necesita la implementación del EJB.

16.2.3.6 RPC vs SOAP

El uso del `SOAPBinding.Style.RPC` está soportado en JAX-WS, sin embargo, se recomienda encarecidamente que sea sustituido por el uso de `SOAPBinding.Style.DOCUMENT`.

16.2.3.7 Manejadores SOAP

Aunque los controladores SOAP de JAX-RPC son bastante similares a los de JAX-WS, será necesario efectuar ajustes en ellos para ser compatibles con JAX-WS.

16.2.3.8 Los Clientes WS

Al igual que con los manejadores, hay que efectuar cambios en los clientes. Sin embargo, aquí las diferencias son tan sustanciales que exigirán una regeneración completa de los clientes adaptados a JAX-WS.

16.3 Grupo Aplicaciones de Escritorio

16.3.1 Introducción

Las aplicaciones de escritorio son, en general, poco reutilizables a la hora de ser migradas a una plataforma JEE7 como la planteada. La capa de presentación, en la que predominarán tecnologías específicas de este tipo de aplicaciones (SWT, Swing, etc) será el principal escollo a la hora de poder aprovechar el trabajo existente.

La capa de negocio y la capa de persistencia sí podrían (dependiendo de cada aplicación y la forma de implementar dichas capas) tener un mayor grado de reutilización a la hora de pasar a la nueva plataforma pero, aún cuando hayan podido ser implementadas con tecnologías a priori portables a la nueva plataforma, si sus especificaciones son anteriores puede resultar complejo su adaptación a JEE7.

16.3.2 Estrategia por capas

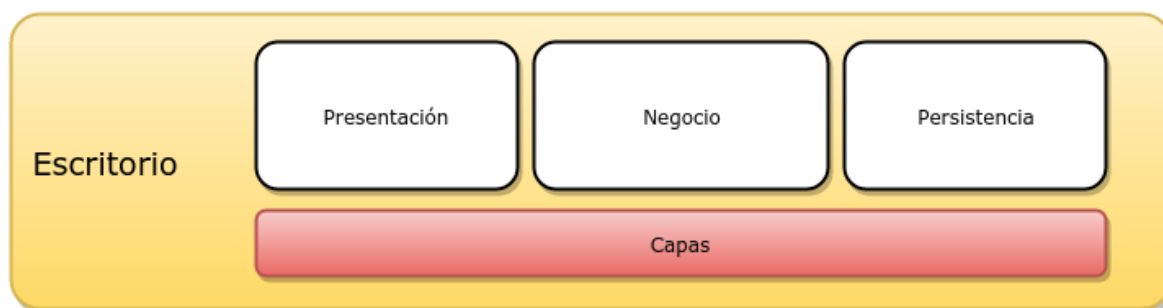


Diagrama 15: Puntos de Análisis Aplicaciones Escritorio

16.3.2.1 Presentación

Es la capa que más dificultades presenta a la hora de ser reutilizada.

En el mercado, existen herramientas que han intentado dar soporte a este tipo de migraciones, pero los intentos más fiables (como por ejemplo AjaxSwing: <http://www.creamtec.com/products/ajaxswing/> o Swingweb: <http://swingweb.sourceforge.net/swingweb/index.html>) siempre parten de, al menos, tecnología Swing, y no SWT, como sería el caso de la aplicación **50 SIDEMAR**.

Además, aunque siempre se puede realizar un primer intento con alguna de estas aplicaciones, incluso partiendo de base Swing, si el renderizado o controles es medianamente complejo, el resultado final de la conversión puede ser bastante pobre.

Por todo esto, una vez se disponga de un arquetipo de desarrollo consolidado, ajustado para la nueva plataforma de destino, y se tenga un equipo formado en su uso, el desarrollo de la capa de presentación partiendo desde el arquetipo (y teniendo en cuenta que se dispondría de la propia aplicación de escritorio como guía de diseño, con el considerable tiempo ahorrado en un aspecto tan importante) será la forma más eficiente de migrar a la nueva plataforma. Dicho esto, habrá que tener siempre presente que:

- Si bien la propia aplicación de escritorio debe servir como guía de diseño al desarrollador, no debe existir una obsesión por la réplica del aspecto en la nueva tecnología, ya que hay que ser consciente que los controles de escritorio pueden no tener siempre su equivalente en la tecnología web.
- Debe aprovecharse para implementar mejoras de diseño que se hayan detectado en el feedback de los usuarios o que salten a la vista como necesarias.

16.3.2.2 Negocio

Es una capa de la que se debe poder reutilizar mucho trabajo a la hora de migrar a la nueva plataforma:

- Por un lado, las clases asociadas a los elementos GUI, nos indicarán cómo responde la aplicación ante cada evento del usuario, mostrándonos por tanto, en líneas generales, cómo deben responder

nuestros bean en la nueva plataforma.

- Por otro lado, las librerías utilizadas para cubrir las funcionalidades en la aplicación de escritorio nos podrán servir, salvo en caso que sean librerías también vinculadas a la tecnología de escritorio, para cubrirlas en la aplicación web. En este punto si se aconseja una revisión de las versiones de estas librerías por si estuvieran muy obsoletas y pudiera aprovecharse para ser actualizadas (siempre y cuando guarden retrocompatibilidad o su impacto sea menor) en la nueva plataforma.

En la aplicación **50 SIDEMAR**, el cliente S2MC es el encargado de encapsular gran parte de la lógica de negocio y, como cliente WS, su uso podría ser reutilizado en la nueva plataforma, puesto que la lógica en sí está definida por el WS al que ataca. Eso sí, el consumo de clientes WebServices en un entorno JEE6 conllevará adaptaciones como establecer confianza del JBoss con el certificado que usa el WS si es accedido en modo SSL, etc. Pero pueden catalogarse como actuaciones menores.

Así mismo, **50 SIDEMAR** utiliza librerías específicas para elementos concretos de negocio, como son las librerías itext o Barbecue, que podrán reutilizarse en la nueva plataforma para cubrir las mismas necesidades para las que fueron utilizadas en la aplicación de escritorio.

16.3.2.3 Persistencia

Hibernate será la tecnología más susceptible de reaprovechamiento, sin embargo, hay que tener mucho cuidado con la implementación de origen, puesto que el paso de JPA de una versión a otra es uno de los elementos más complejos en una migración a JEE6. En cualquier caso, la existencia de mapeos Hibernate siempre ayudará al desarrollador, en el peor de los casos a entender mejor el diseño de persistencia.

En el caso de **50 SIDEMAR**, existe una *doble persistencia* a tener en cuenta:

- Por un lado, la gestionada por el WebService de destino que encapsula gran parte de la lógica de negocio y que, por tanto, estaría fuera del ámbito de migración de SIDEMAR en sí, ya que dependería de la operativa del WebService.
- Por otro lado, una persistencia local basada en una estructura de directorios del equipo cliente que debería ser trasladada al servidor, a algún gestor documental o a otro tipo de solución para mantener unificada pero por cada usuario esta persistencia local actual.

16.3.3 Java Web Start

Consideramos que el objetivo final es migrar aplicaciones de escritorio a aplicaciones 100% Web. No obstante, creemos necesario al menos mencionar aquí la existencia de esta posibilidad, a medio camino entre la conversión a Aplicación Web y el mantenimiento de aplicaciones de escritorio distribuidas cliente a cliente:

- Java Web Start es la implementación de referencia de la especificación Java Network Launching Protocol (JNLP), desarrollada por Sun Microsystems (actualmente Oracle), mediante la cual permite controlar la distribución de aplicaciones Java que están en un servidor web de aplicaciones.
- Es un enfoque más orientado a la distribución de aplicaciones (al control de versiones en los equipos clientes, actualizaciones, etc) que a la portabilidad que estamos tratando aquí, ya que la aplicación en sí no es migrada de tecnología.

17 Documentos Anexos

17.1 Arquitectura Actual

En el documento anexo **Arquitectura_de_software_SNA_v1.0.3.0.pdf** se encuentra la descripción detallada de cada capa.

17.2 Arquetipo Actual

En el documento anexo **Arquetipo_Manual_de_Uso_v2.0.1.0.pdf** se encuentra el arquetipo actual del SNA.

17.3 Agrupación de Aplicaciones

En el documento anexo **CMAA-Estrategia-APPS-NEGOCIO.xlsx** se encuentra la agrupación de aplicaciones por: plataforma, tipo de aplicación y negocio al que pertenece.

En el documento anexo **CMAA_Apps.Migrar.Consolidado_20160404.xlsx** se encuentra la tabla resumen de aplicaciones a migrar agrupadas por servidor de despliegue, tecnología y framework.

17.4 Riesgo de migración

En el documento anexo **CMAA_Estrategia_Indicador_RiesgoMigracion.ods** se encuentra un mapa de calor de las aplicaciones por su completitud documental, complejidad de migración y riesgo de migración.

17.5 Indicadores de las aplicaciones

En el documento anexo **CMAA - MIGAPPS- Ranking Aplicaciones_(FormatoExcel)_20160322.xlsx** se encuentra, en forma de matriz, todos los indicadores recopilados durante el levantamiento de las aplicaciones y el análisis de brecha.

17.6 Valoración del Esfuerzo de Migración (HH)

En el documento anexo **CMAA Ponderacion Esfuerzo Migracion.ods** se encuentra la valoración del esfuerzo de migración.

17.7 Prueba de Concepto

En el documento anexo **CMAA_Prueba_de_Concepto.pdf** se la prueba de concepto realizada para migrar una aplicación de Weblogic 8.1 a JBoss EAP 6.4.

17.8 Otros

Gráficos del Plan de Migración: **Graficos Plan Migracion.docx**