ANEXO VII

Arquitectura de Software
Plan Informático

Contenido

1 Arquitectura de Software Servicio Nacional de Aduanas

A continuación se describe la arquitectura de software propuesta por el Servicio Nacional de Aduanas (SNA) de acuerdo a los lineamientos de desarrollo de software considerados como básicos y a la plataforma tecnológica que posee. La arquitectura se presenta como un modelo de n-capas en las cuales se identifican básicamente las capas: Presentación, Negocios y Datos. Complementario a esta estructura básica se visualizan las capas de Servicios e integración con sistemas internos o externos. Finalmente se distinguen 2 capas transversales a todos los sistemas: Seguridad y Auditoría. (*Ver Ilustraciones 1 y 2*)

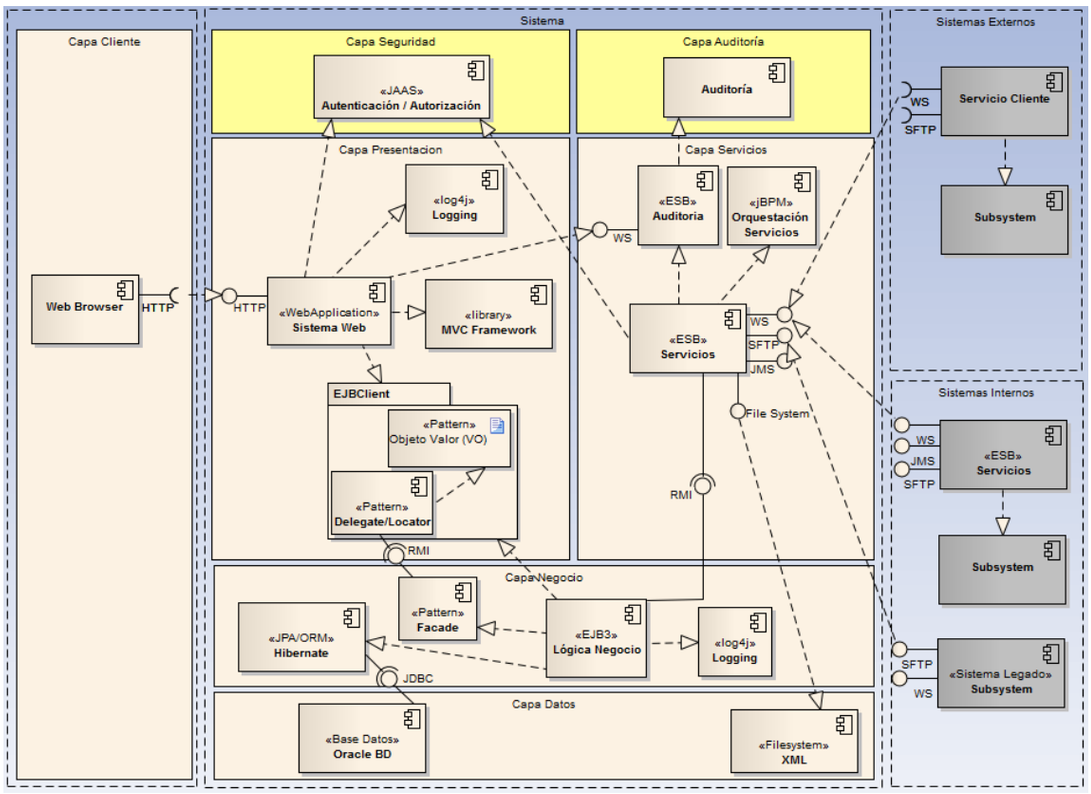

Ilustración 1: Diagrama de Arquitectura de Software SNA

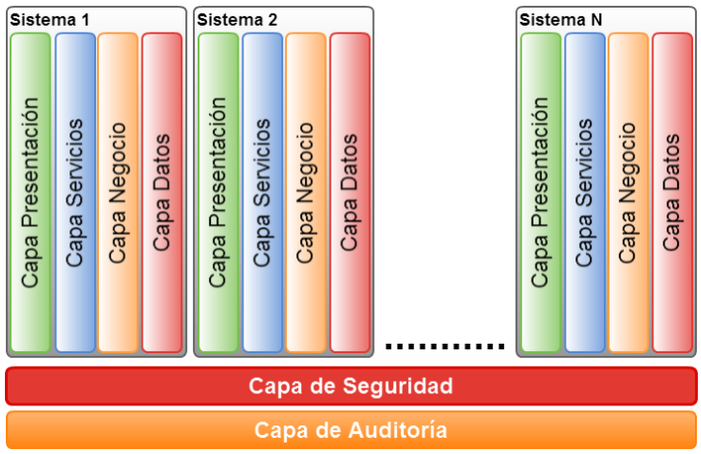

Ilustración 2: Diagrama de Capas

2 Descripción de Capas

2.1 Capa de Cliente

La capa cliente identifica el punto de acceso del usuario final a un sistema de información. En este caso el medio en que se expone el sistema es una página web, por consiguiente, el usuario para acceder al sistema debe utilizar un "Navegador Web" (Web Browser). Esta capa interactúa directamente con la capa de presentación.

2.1.1 Navegador Web (Web Browser)

Navegador Web o Web browser es una aplicación que opera a través de internet, es capaz de interpretar la información de archivos y sitios web para que estos puedan ser leídos. Dentro de un sistema de información que opera en la web, el navegador web provee al usuario de una interfaz para interactuar con los datos.

2.2 Capa de Presentación

La capa de presentación presenta el sistema al usuario, le comunica la información y captura la información del usuario en un mínimo de proceso (realiza un filtrado previo para comprobar que no hay errores de formato). También es conocida como interfaz gráfica y debe tener la característica de ser "amigable" (entendible y fácil de usar) para el usuario. Esta capa se comunica con las capas de Negocio y Servicios.

2.2.1 Sistema Web (Web Application)

El sistema web es un sistema de información que se ejecuta en un servidor web al cual los usuarios pueden acceder a través de internet utilizando un navegador web.

2.2.2 Trazabilidad (Logging)

El módulo de trazabilidad permite registrar mensajes importantes de la aplicación para realizar tareas de depuración que permitan identificar fácilmente problemas durante las fases de desarrollo en distintos ambientes (desarrollo, test y producción).

2.2.3 MVC Framework

El MVC Framework corresponde a alguna biblioteca que implemente MVC. MVC (Modelo Vista Controlador) es un patrón o modelo de abstracción de desarrollo de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de negocio en tres componentes distintos.

- **Modelo:** Es la representación específica de la información con la cual el sistema opera. El modelo se limita a lo relativo de la vista y su controlador facilitando

las presentaciones visuales complejas. El sistema también puede operar con más datos no relativos a la presentación, haciendo uso integrado de otras lógicas de negocio y de datos afines con el sistema modelado.

- **Vista:** Presenta el modelo en un formato adecuado para interactuar, usualmente la interfaz de usuario.
- **Controlador:** Responde a eventos, usualmente acciones del usuario, e invoca peticiones al modelo y, probablemente, a la vista.

2.2.4 EJB Client

Este componente encapsula la lógica de acceso y llamada a la capa de negocio. Dentro de este componente existen 2 sub-componentes principales: "Delegate/Locator", para el acceso a capa de negocio y "Objeto Valor (VO)" para la transferencia de datos.

2.2.4.1 Delegate/Locator

Los patrones de diseño delegate y service locator son los encargados de implementar la comunicación entre las capas de presentación y negocio.

Patrón delegate: El patrón de diseño de delegación es una técnica en la que un objeto de cara al exterior expresa cierto comportamiento pero en realidad delega la responsabilidad de implementar dicho comportamiento a un objeto asociado en una relación inversa de responsabilidad.

Patrón Service Locator: Este patrón de diseño permite abstraer todo los usos de JNDI simplificando el código del cliente, creando un único punto de control y mejorando el performance de la aplicación.

2.2.4.2 Objeto Valor (VO: Value Object)

El patrón de diseño objeto valor (VO) permite transformar cualquier documento u otro objeto en un objeto genérico que pueda ser entendido por el sistema en cuestión, aumentando la generalidad del sistema y ocultando información que manejan otros componentes. En general se complementa con facade y se debe utilizar siempre que se implementen llamadas remotas a una capa de negocio basada en EJB3.

2.3 Capa de Servicios

La capa de servicios es una capa de abstracción que permite la integración entre la capa de negocios y el bus de servicios a través del cual otros clientes o sistemas pueden intercambiar información o dar continuidad a un proceso de negocio. Las implementaciones en esta capa van dando forma al "Catálogo de Servicios" el cual permite reutilización de servicios de manera transversal dentro de la plataforma.

2.3.1 Servicios (ESB)

Los ESB (Bus de Servicio Empresarial) representan un elemento de software que media entre las aplicaciones empresariales y permite la comunicación entre ellas. Es capaz de sustituir todo contacto directo con las aplicaciones que se comunican a través del Bus. El principal beneficio es el integrar aplicaciones existentes en la organización para hacer que funcionen juntas.

2.3.2 Orquestación Servicios (jBPM)

Generalmente es un proceso de negocio independiente (podría ser incluso otro servicio) que llama a otros servicios y coordina la ejecución de ellos. Esto lleva a la ejecución de un proceso de negocio de alto nivel. Sólo el orquestador conoce cuál es el objetivo del proceso de negocio de alto nivel.

2.3.3 Auditoría (ESB)

El servicio de auditoría es el encargado de recibir en forma centralizada y mediante mensajes asíncronos el detalle de las acciones realizadas en el sistema. (*Ver: Capa de Auditoría*)

2.4 Capa de Negocio

La capa de negocio es donde se reciben las peticiones del usuario y se envían las respuestas tras el proceso. Se denomina capa de negocio (e incluso de lógica del negocio) porque es aquí donde se establecen todas las reglas que deben cumplirse. Esta capa se comunica con la capa de presentación, para recibir las solicitudes y presentar los resultados, y con la capa de datos, para solicitar al gestor de base de datos almacenar o recuperar datos de él. Similar a la capa de servicios los desarrollos implementados en esta capa dan forma a un "Catálogo de Lógicas de Negocio" las cuales pueden ser reutilizadas de manera transversal por otros sistemas dentro de la plataforma.

2.4.1 Lógica de Negocio (EJB3)

La lógica de negocio debe centralizarse en EJB3. Los EJB son clases java del tipo POJO (Plain Old Java Object), clases simples sin dependencias de frameworks asociados. A estos POJO´s se les agregan anotaciones que permiten abstraer funcionalidades asociadas a la implementación.

Las principales características de la tecnología basada en EJB3 son:

- La implementación de interfaces locales para el uso de EJB dentro del mismo contexto.
- La implementación de interfaces remotas para usos externos utilizando el protocolo RMI.
- El manejo de transaccionalidad aumentando la confiabilidad de los sistemas.
- La inyección de dependencias que permite abstraer al desarrollador de utilizar clases del tipo context y métodos lookup para invocar un EJB de session.

Actualmente la especificación de EJB3 basada en JEE5 contiene tres tipos de clases EJB, las cuales son:

Session Beans: Reciben las peticiones desde las capas o superiores (Web/Servicios/ESB) u otros componentes y en base a estas peticiones ejecutan las acciones contenidas en los métodos invocados. En general se utilizan como patrón fachada (se describe en el punto 4,4).

Existen dos tipos de session bean, los cuales son:

- **Stateful**: Permiten almacenar estados al momentos de ser invocados obteniendo como resultado un flujo transaccional durante la conexión con el componente o capa superior que lo está utilizando. En general son usados cuando el componente o capa superior implementa un proceso que requiere guardar estados de forma momentánea, más un inicio y un fin.
- **Stateless**: Estos EJB no almacenan estados por lo que su ciclo de vida es levantare cuando alguien lo invoca a través de una señal (método que contiene) y morir una vez realizada la acción. Se recomienda su uso cuando los sistemas tienen una alta concurrencia.

Entity Bean: EJB que representa una abstracción concreta de cada tabla perteneciente a un modelo de datos específico. Esta abstracción se logra implementado la API JPA (Java Persistence API) que provee la especificación de JEE5. JPA provee de una suite de anotaciones y clases que permiten gestionar bases de datos relacionales con un enfoque orientado a objetos. Para que JPA pueda realizar el mapeo entidad/objeto se debe utilizar un ORM (Object Relational Mapper), que se describe en el punto 2.4.2. JPA implementa su propio lenguaje SQL el cual es bastante similar al tradicional, pero se debe considerar que cada tabla ahora es un objeto en estudio.

Message Driven Bean: Conocido también como JMS es un componente del tipo asíncrono que permite implementar una cola de mensajes (queue) para la comunicación entre dos componentes debidamente suscritos a la cola. Se recomienda su uso en sistemas de alta transaccionalidad en el que el aseguramiento en la calidad de la entrega debe ser óptimo y asíncrono.

2.4.2 Persistencia de Datos (Hibernate)

La persistencia de datos es un enfoque en el cual se utiliza un componente llamado ORM (Object-Relational Mapping) que cumple la función de levantar una base de datos persistente y consistente con el modelo de base de datos y el acceso a ellos, para que JPA lo consuma como un componente más basado en objetos. Existen varios productos que proveen este enfoque, como por ejemplo: TopLink, EclipseLink, OpenJPA y Kodo (entre los más usados), pero por motivos de la solución que implementa la plataforma adoptada y de preservación de la arquitectura actual se debe implementar Hibernate 3.2.4 y JPA 2.0, que corresponden a versiones certificadas por la plataforma.

2.4.3 Trazabilidad (Logging)

El módulo de trazabilidad permite registrar mensajes importantes de la aplicación para realizar tareas de depuración que permitan identificar fácilmente problemas durante las fases de desarrollo en distintos ambientes (desarrollo, test y producción).

2.4.4 Patrón de Diseño "Facade"

Facade o fachada es un patrón de diseño que permite poner una clase que reciba peticiones y las transforme en objetos que puedan ser entendidos por lo componentes que están tras ella como por ejemplo el modelo (datos) y otros servicios que necesitan de más ocultamiento de información, esto permite que las capas superiores sólo se comuniquen con facade para solicitar servicios de componentes de capas inferiores.

2.5 Capa de Datos

La capa de datos es donde residen los datos y es la encargada de acceder a los mismos. Está formada por uno o más gestores de bases de datos que realizan todo el almacenamiento de datos, reciben solicitudes de almacenamiento o recuperación de información desde la capa de negocio.

2.5.1 Base de Datos

La base de datos o sistema gestor de base de datos es un software que permite almacenar y posteriormente acceder a los datos de forma rápida y estructurada. El gestor de base de datos abstrae al programador de toda la problemática de implementación del almacenamiento y acceso a los datos.

2.5.2 Procedimientos Almacenados

Los motores de bases de datos proveen un lenguaje de programación a través del cual se pueden encapsular secuencias de tareas, a estas piezas de software se les llaman "Procedimientos Almacenados". Los procedimientos almacenados deberían destinarse principalmente a resolver sentencias SQL complejas las cuales podrían tener un alto

costo de implementación en la capa de negocios. Los procedimientos almacenados pueden contener lógicas de negocios siempre y cuando el escenario de implementación lo justifique.

2.5.3 Sistema de Archivos (Filesystem)

El sistema de archivos es el espacio físico donde se almacenan archivos, como por ejemplo: discos de estado sólido, discos duros, cintas, etc. El sistema de archivos de un servidor también es considerado un almacén de datos que puede ser utilizado por un sistema de información. De hecho, el manejo de archivos binarios es recomendable a nivel de filesystem y no como un registro en una tabla de base de datos.

2.6 Capa de Seguridad

La capa de seguridad es una capa transversal a todos los sistemas en donde se centralizan las definiciones de los dominios de seguridad que pueden ser utilizados por los sistemas. El propósito de esta capa es proveer de los servicios de Autenticación y Autorización.

2.6.1 Autenticación y Autorización

Autenticación y Autorización forman parte del proceso de intento de acceso de un usuario a un sistema.

- **Autenticación:** Es el proceso de intento de verificar la identidad digital de un usuario.
- **Autorización:** Es el proceso por el cual el sistema autoriza al usuario identificado a acceder a determinados recursos.

Dentro de la arquitectura se considera que la Autenticación y Autorización sean centralizadas y configurables.

2.7 Capa de Auditoría

La capa de auditoría es una capa transversal a todos los sistemas, en ella se dispone de un método único y genérico de registro de actividades detallado para cada sistema. Esta capa cumple con el propósito de abstraer el medio en que se almacenan los datos de auditoría de sistemas.

2.7.1 Auditoría

El módulo de auditoría es el encargado de ejecutar el registro detallado de los eventos que ocurren en una aplicación. Los eventos básicos a ser registrados tienen relación con las acciones de Crear, Modificar y Eliminar (O alguna otra acción en caso de ser necesario). Además de identificar las acciones se debe registrar el usuario que realizó dichas acciones.

2.8 Sistemas Internos

Los sistemas internos corresponden a aplicaciones que residen dentro de los servidores del SNA. Estos sistemas pueden ser de desarrollo interno o desarrollos externos entregados al SNA.

2.8.1 Servicios (ESB)

Los ESB (Bus de Servicio Empresarial) representan un elemento de software que media entre las aplicaciones empresariales y permite la comunicación entre ellas. Es capaz de sustituir todo contacto directo con las aplicaciones que se comunican a través del Bus. El principal beneficio es el integrar aplicaciones existentes en la organización para hacer que funcionen juntas.

2.8.2 Subsistema (Subsystem)

Un subsistema se refiere a un sistema que puede intercambiar información con el sistema principal desde el punto de vista del diseño de esta arquitectura.

2.8.3 Sistema Heredado (legado/legacy)

Los sistemas heredados son sistemas que permanecen en funcionamiento dentro de la organización. Con el paso del tiempo estos sistemas pasan a ser anticuados. La permanencia de estos sistemas está dada por la complejidad de reemplazar o actualizar el sistema. Los software de tipo middleware como por ejemplo las implementaciones de buses de servicios, permiten comunicar estos sistemas heredados con el resto de los sistemas de la organización sin complicaciones.

2.9 Sistemas Externos

Los sistemas externos son software de aplicación que se encuentran fuera de las fronteras de los sistemas de información internos los cuales pueden desear intercambiar información (Interoperabilidad). La forma más frecuente de interoperabilidad de datos son los servicios web (Web Services).

2.9.1 Servicio Cliente

Los servicios clientes son implementaciones de módulos de interoperabilidad, los que frecuentemente utilizan una interfaz de web service para el intercambio de información. Otro medio en intercambio de información puede ser, por ejemplo, un servicio FTP (SFTP).

2.9.2 Subsistema (Subsystem)

Un subsistema se refiere a un sistema que puede intercambiar información con el sistema principal desde el punto de vista del diseño de esta arquitectura.

3 Diccionario de Acrónimos

Acrónimo	Descripción
API	Application Programming Interface
BPEL	Business Process Execution Language
EJB	Enterprise JavaBeans
ESB	Enterprise Service Bus
FTP	File Transfer Protocol
JAAS	Java Authentication Autorization Service
jBPM	Java Business Process Management
JDBC	Java Data Base
JEE	Java Enterprise Edition
JPA	Java Persistence Api
MVC	Model View Controller
ORM	Object-Relational Mapping
POJO	Plain Old Java Object
RAC	Real Application Clusters (Oracle)
SFTP	Secure File Transfer Protocol
SNA	Servicio Nacional de Aduanas
SOA	Service Oriented Architecture