|  |
| --- |
| colored transparent rectangle  Sistema de Gestión de Cocheras (SGC)  Versión 1.1  Documento de Arquitectura del Software |

|  |
| --- |
| Tech Dev  Website: www.techdev.com |

|  |  |
| --- | --- |
|  |  |

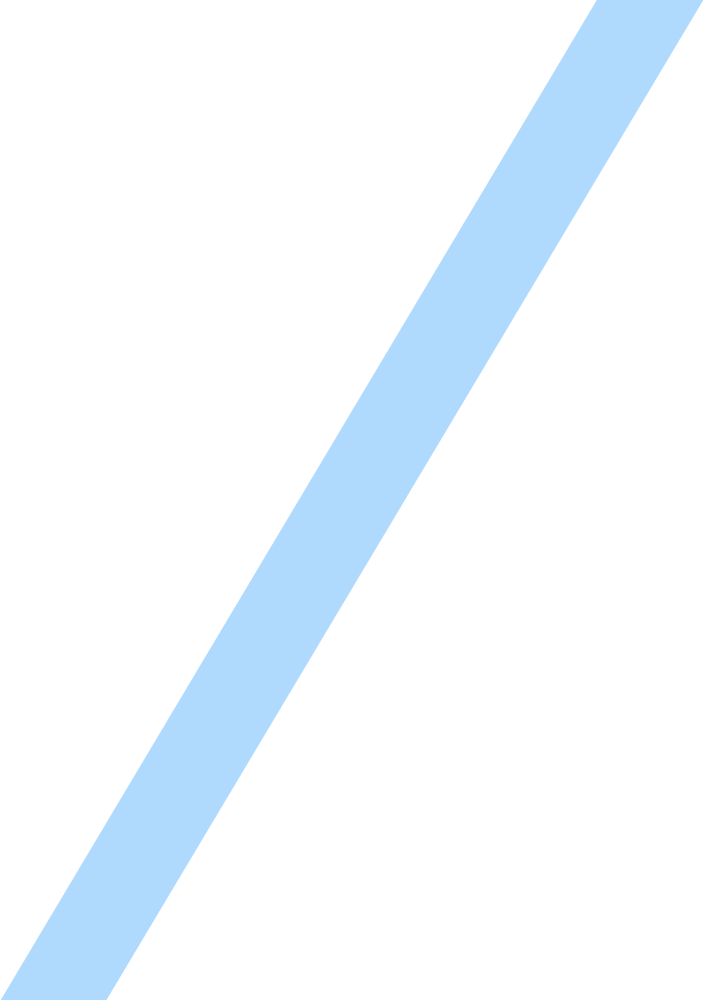
|

Tabla de Contenidos

1. Introducción 3

1.1 Propósito 3

1.2 Alcance 3

1.3 Definiciones, Acrónimos y abreviaturas 3

1.4 Referencias 3

1.5 Generalidades 4

2. Representación de la Arquitectura 4

3. Metas y Restricciones Arquitectónicas 6

3.1 Requerimientos no funcionales 6

➢ SIGE – DESEMP – 01 6

➢ SIGE – CONF – 01 6

➢ SIGE – FAC-SOP – 01 6

➢ SIGE – FAC-SOP – 02 6

3.2 Riesgos 7

3.3 Restricciones Especiales. 7

4. Vista de Casos de Uso 7

4.1 Diagrama de Caso de Uso 7

4.2 Casos de Uso Significativos de la Arquitectura 8

5. Vista Lógica 9

5.1 Generalidades 9

5.2 Paquetes de Diseño Arquitectónicamente Significativos 9

5.2.1 Controladora 9

5.2.2 Lógica del Negocio 9

5.3 Interpretaciones de los Casos de Uso 10

6. Vista de Procesos 11

7. Vista de Despliegue 11

7.1 Cliente web 11

7.2 Cliente móvil 11

7.3 Cliente escritorio 12

7.4 Servidor Web 12

8. Vista de Implementación 12

8.1 Generalidades 12

8.2 Capas 12

8.2.1 Capa de Presentación 13

8.2.2 Capa Controladora 13

8.2.3 Capa Lógica del Negocio 13

8.2.4 Capa de Persistencia 14

9. Vista de Datos 16

10. Tamaño y desempeño 16

11. Calidad 16

Documento de Arquitectura del Software (SAD)

# Introducción

El presente documento provee una visión general de la arquitectura del SIGE – Sistema de Gestión de Estacionamientos, usando diferentes vistas para apreciar los diferentes aspectos del Sistema, las cuales están basados en los estándares del Proceso Unificado de Desarrollo de Software (RUP) y utilizando el Lenguaje de Modelamiento Unificado (UML).

## Propósito

El presente documento tiene como objetivo.

* Plasmar mediante diagramas, modelos y artefactos del RUP, las consideraciones técnicas y tecnológicas (plataforma) en la que será implementada la solución.
* Esbozar los aspectos funcionales de la aplicación.
* Definir los mecanismos de despliegue y distribución del sistema.
* Esbozar el modelo entidad – relación de la arquitectura de datos a desarrollar.

## Alcance

Detalla la arquitectura propuesta por el equipo de desarrollo y contempla los modelos de dominio y datos, además de los diagramas de diseño necesarios para comprender el comportamiento de los componentes.

## Definiciones, Acrónimos y abreviaturas

Se refiere a toda definición, acrónimo y abreviatura requerida para entender este documento:

Chofer o Cliente Usuario del Aplicativo móvil.

Administrador Global Usuario del Aplicativo Web.

Encargado de cochera Empleado de una Empresa, que a la vez es Usuario del Aplicativo de Escritorio.

Empresa La Empresa administradora encargada de la gestión de sus estacionamientos.

Empleado de cochera Trabajador de la Empresa administradora de cocheras.

Placa Inscripción e identificador de un automóvil.

DNI Documento Nacional de Identidad de una persona

RUC Registro Único del Contribuyente

RENIEC Registro Nacional de Identificación Ciudadana

SIGE Sistema de Gestión de Estacionamientos

## Referencias

Los siguientes documentos referenciados han sido usados como base para elaborar el presente documento.

* *SIGE -* Especificaciones de Casos de Uso
* *SIGE-* Especificación de Requerimientos del Software

## Generalidades

Define las consideraciones de infraestructura para el sistema, así como los diagramas que permiten entender las dependencias entre artefactos y/o componentes en la arquitectura de la solución.

# Representación de la Arquitectura

A continuación se muestra la arquitectura del sistema SIGE – Gestión de Estacionamientos, está dividida en 4 capas:

* Capa de Presentación
* Capa de Control
* Capa de lógica-de-negocio
* Capa de Persistencia

Aplicativo Web:

Presentación

Angular - Templates

Controladora

Angular - Components

Lógica de Negocio

Angular - Services

REST API

LoopBack

Aplicativo Movil:

Presentación

Android - Vistas

Controladora

Android - Controladores

Lógica de Negocio

Android - Services

REST API

LoopBack

Aplicativo de Escritorio

Presentación

Java - JFrame

Controladora

Java - Controladores

Lógica de Negocio

Java - Services Classes

REST API

LoopBack

El documento se ha estructurado empleando la representación de la arquitectura de acuerdo con la arquitectura de 4 + “1” vistas propuestas por IBM Rational. La representación se realizará a fin de mostrar diferentes perspectivas del producto software, empleando las vistas siguientes:

1. Perspectiva Funcional – Vista de Casos de Usos. Presenta la arquitectura desde la perspectiva del usuario final. Esta vista se desarrolla a través del Modelo de Casos de Usos (usando Diagramas de Casos de Uso de UML).
2. Perspectiva Estructural – Vista Lógica. Presenta la arquitectura desde la perspectiva del desarrollador. Permite mostrar la organización de las piezas fundamentales de la arquitectura, organizando los elementos de diseño (clases, tablas, etc.).
3. Perspectiva de Construcción – Vista de Implementación. Presenta la arquitectura desde la perspectiva del programador, definiendo los componentes software a ser desarrollados, la distribución de las clases, tablas y demás.
4. Perspectiva Dinámica – Vista de los Procesos / Tareas. Presenta la arquitectura desde la perspectiva del desarrollador a fin de definir aspectos de concurrencia, comunicación interprocesos, sincronizaciones, etc.
5. Perspectiva de los Datos – Vista de Datos. Presenta la arquitectura de datos que soportará los requerimientos de información del sistema software. Se emplea el Modelo Entidad Relación.
6. Perspectiva del Despliegue – Vista de Despliegue. Presenta la arquitectura desde la perspectiva del implantador de la solución. Define como los componentes de la arquitectura serán desplegados sobre la infraestructura de TI definida.

# Metas y Restricciones Arquitectónicas

Se han identificado los requerimientos no funcionales y riesgos que impactan sobre la arquitectura del sistema.

## Requerimientos no funcionales

Los Requerimientos no Funcionales que tienen impacto sobre la arquitectura del sistema son los siguientes:

### SIGE – DESEMP – 01

La información almacenada podrá ser consultada y actualizada permanentemente y simultáneamente, sin que se afecte el tiempo de respuesta.

### SIGE – CONF – 01

El Sistema estará disponible las 24 horas del día durante el año, siempre y cuando se tenga la conexión con las dependencias definidas.

### SIGE – FAC-SOP – 01

El estándar de codificación de los programas será Java para el desarrollo del Sistema de Escritorio, JavaScript y TypeScript para el Sistema Web y Java para el aplicativo Móvil, siguiendo el estándar del documento “Estándares de Programación en JAVA – Versión 0.9.3”.

### SIGE – FAC-SOP – 02

La base de datos a la que se conectará el Sistema es MongoDB, siguiendo el estándar del documento “MongoDB Applied Design Patterns”.

## Riesgos

Los riesgos más significativos para el sistema son:

* La no disponibilidad del servicio RESTful API (proporcionado por un servidor Amazon Web Services AWS).
* La no disponibilidad del servicio de base de datos proporcionado por Amazon Web Services AWS (Alojado en otra instancia)

## Restricciones Especiales.

No aplica.

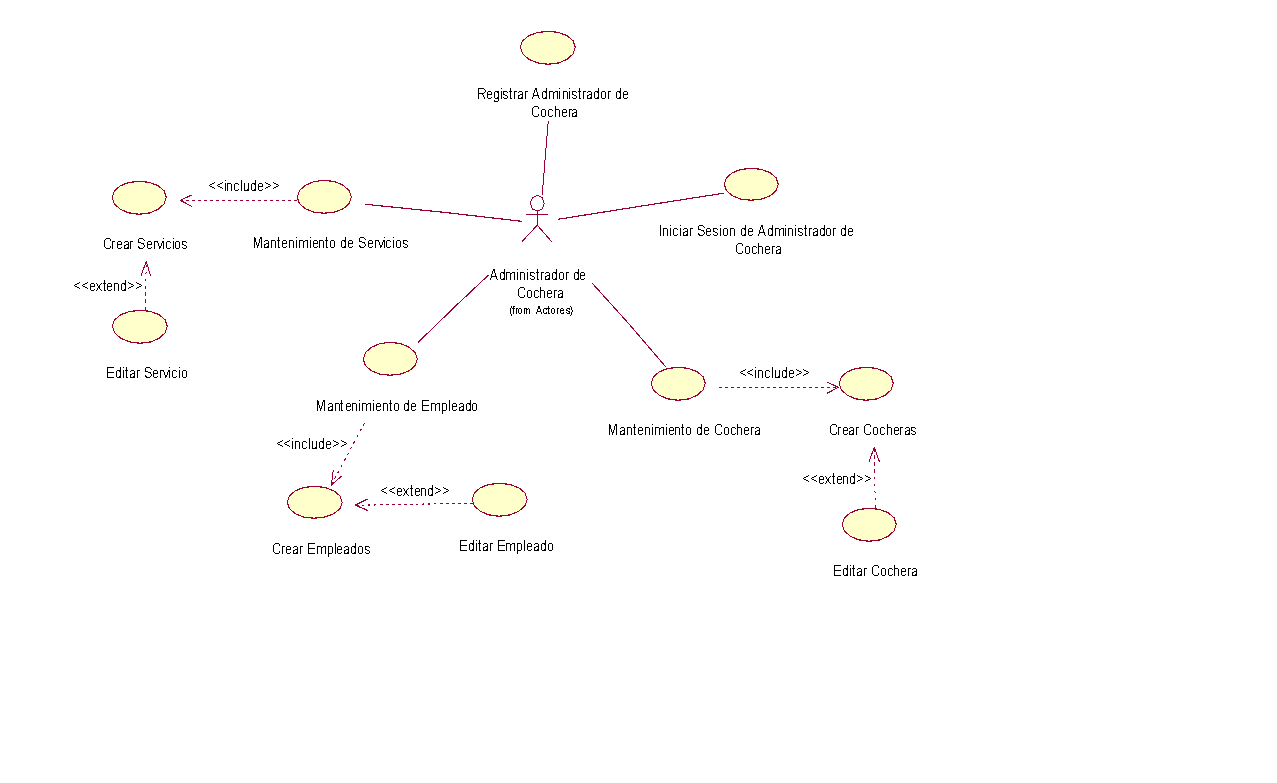
# Vista de Casos de Uso

En esta parte del documento se listan los casos de usos y los escenarios del modelo de casos de uso que representan requerimientos arquitectónicamente significativos basados en los aspectos siguientes:

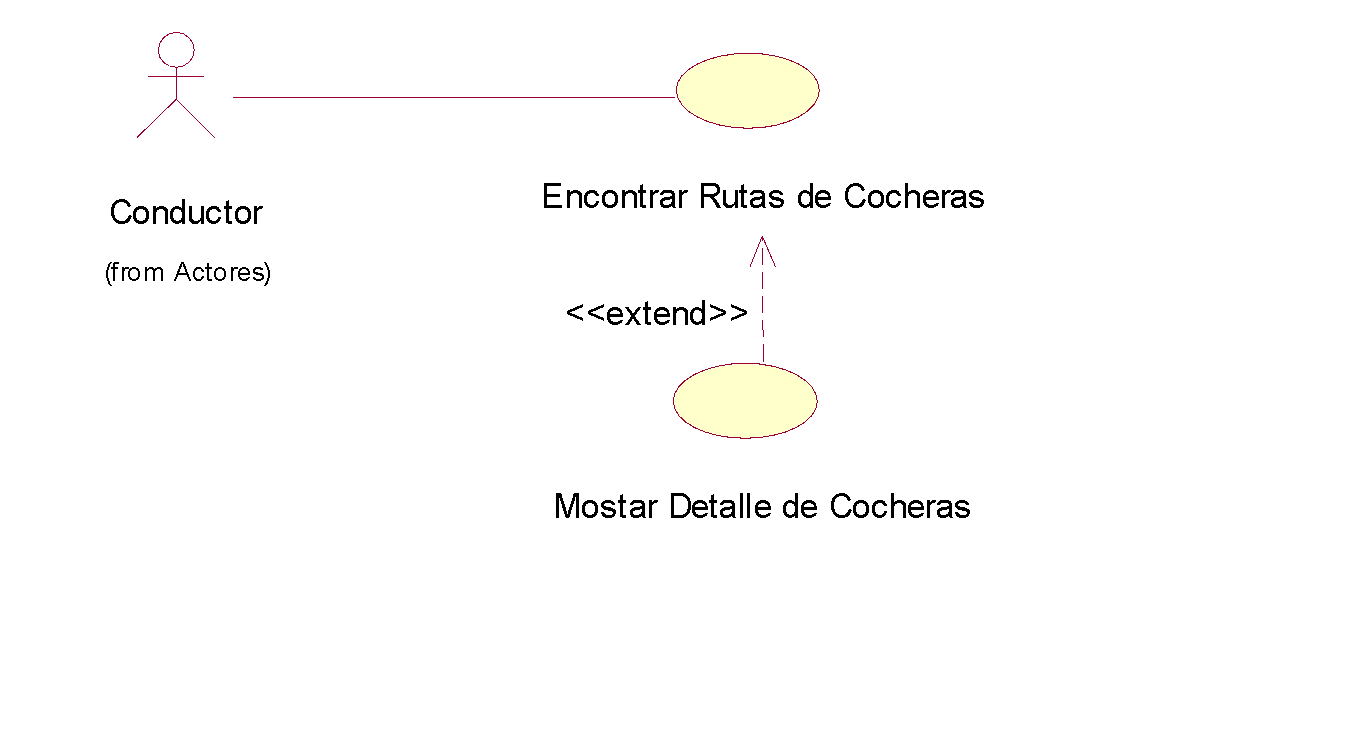
* Si representan una funcionalidad central y significativa del sistema final.
* Tienen una amplia cobertura de la arquitectura del producto, porque ejercitan muchos de elementos de la arquitectura, o si ellos someten a alto stress o ilustran puntos delicados de la arquitectura.

## Diagrama de Caso de Uso

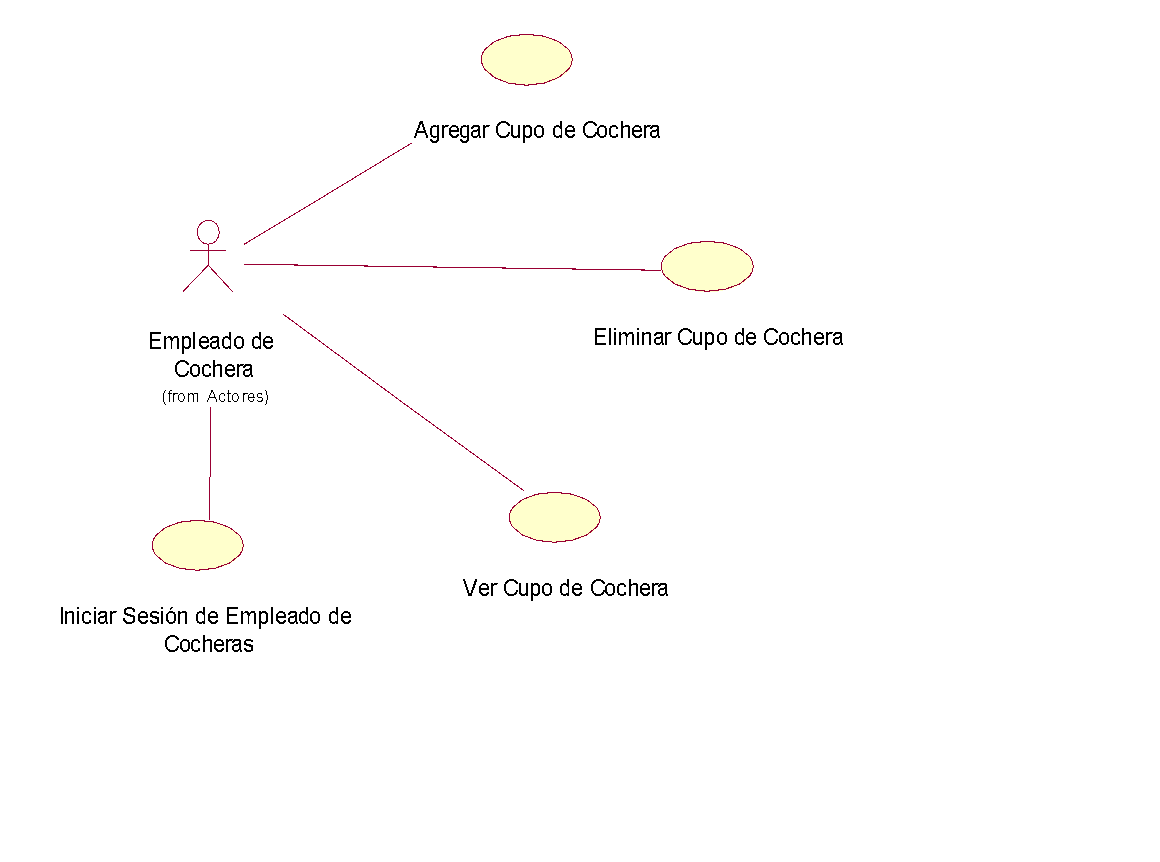
Paquete Registro de Cochera:



Paquete Consultas:



Paquete Gestión de cocheras:



## Casos de Uso Significativos de la Arquitectura

Los casos de uso significativos de arquitectura son aquellos que tienen restricciones es decir están vinculados a requerimientos no funcionales y/o riesgos.

|  |  |
| --- | --- |
| Caso de Uso Significativo | Requerimientos No Funcionales y/o Riesgos |
| Registrar Administrador de cocheras | La información almacenada podrá ser consultada y actualizada permanentemente y simultáneamente, sin que se afecte el tiempo de respuesta. |
| Iniciar Sesión de administrador de cochera |
| Mantenimiento de cocheras |
| Mantenimiento de servicios |
| Agregar cupo de cochera |
| Encontrar rutas de cochera |

# Vista Lógica

## Generalidades

En esta parte del documento se describe la descomposición del modelo de diseño en término de jerarquía de paquetes y sus capas. A nivel arquitectural la aplicación web se compondrá en cuatro capas: Capa de Presentación, Capa controladora, Capa de la Lógica del Negocio y Capa de Persistencia.

A continuación, se describe la vista para el subsistema Seguridad siendo similar para los demás subsistemas, donde en cada capa se presentarán las mismas relaciones.

## Paquetes de Diseño Arquitectónicamente Significativos

Para cada paquete significativo, se describe el nombre del paquete y su descripción, y uno o varios diagramas con todas las clases y paquetes significativos contenidos dentro del paquete. Para cada clase significativa en el paquete, se incluye el nombre, descripción y opcionalmente, una descripción de sus principales responsabilidades, operaciones y atributos.



### Controladora

### Lógica del Negocio

Esta capa contiene a las clases java representantes de las entidades de la aplicación, para su uso en la lógica del negocio y que se corresponden con las clases de persistencia de la aplicación.

## Interpretaciones de los Casos de Uso

# Vista de Procesos

# Vista de Despliegue

El Sistema SIGE se hospedará y ejecutará desde un servidor principal de aplicaciones, junto con el manejador de base de datos y los datos de la aplicación. Los usuarios tendrán acceso al sistema mediante el uso de estaciones de trabajo conectadas directamente al servidor (mediante una red interna, por ejemplo) o a través de Internet (asumiendo que conozcan la dirección I.P. del servidor en donde se ejecute el sistema SIGE). Dichos clientes deben ejecutar un navegador web para poder hacer uso del sistema. Se recomienda tener un servidor de respaldo, el cual monitoree el servidor principal y pueda detectar cuando haya fallas en el mismo.

## Cliente web

Las configuraciones físicas necesarias con la que debe contar el cliente se especifican a continuación:

* Sistema Operativo: Windows XP, Linux o macOS o nuevas versiones a partir de ella.
* Navegador web: Google Chrome 50, Safari 9 o nuevas versiones a partir de ella
* RAM: 1 GB

## Cliente móvil

Las configuraciones físicas necesarias con la que debe contar el cliente se especifican a continuación:

* Sistema Operativo: Android 5 o nuevas versiones a partir de ella.
* Navegador web: Google Chrome 50, Safari 9 o nuevas versiones a partir de ella
* RAM: 1 GB

## Cliente escritorio

Las configuraciones físicas necesarias con la que debe contar el cliente se especifican a continuación:

* Sistema Operativo: Windows XP o nuevas versiones a partir de ella.
* RAM: 1 GB

## Servidor Web

Las configuraciones físicas necesarias con la que debe contar el Servidor Web en el que estará alojada la aplicación se especifican a continuación:

* Sistema Operativo: Linux 3 o nuevas versiones a partir de ella.
* RAM: 4 GB

# Vista de Implementación

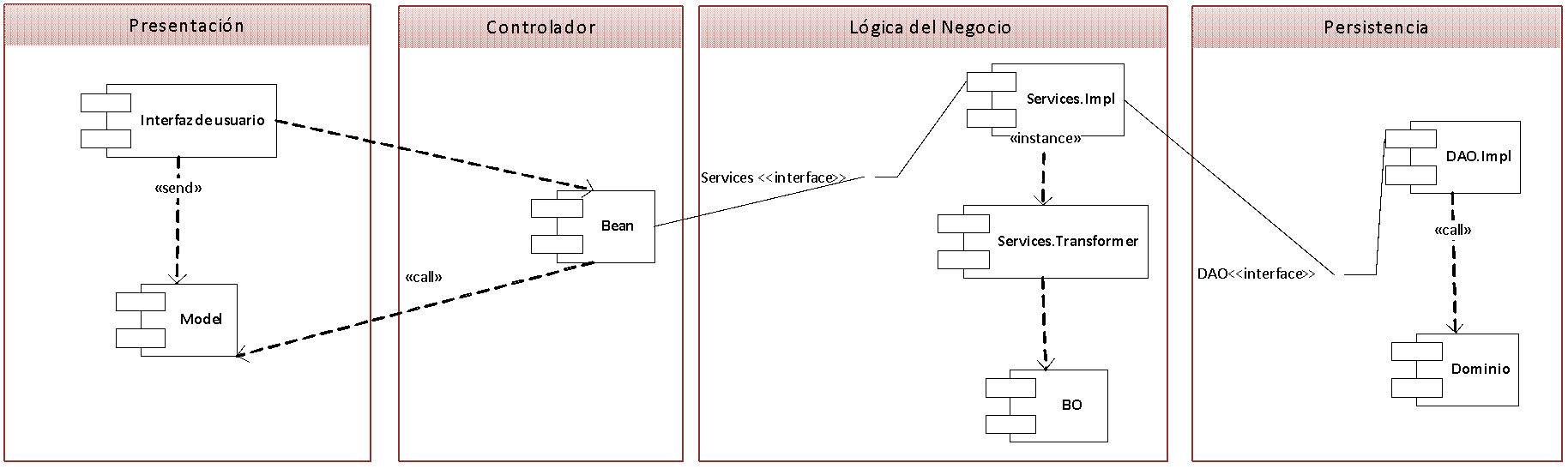
En esta sección se describe la estructura completa del Modelo de Implementación, la descomposición del software en capas y subsistemas en el Modelo de Implementación, y cualquier componente arquitectónicamente significativo.

## Generalidades

Las capas con las que contará el sistema en su implementación serán las siguientes:

* Capa de Presentación:
* Capa de Control:
* Capa de lógica-de-negocio:
* Capa de Persistencia:

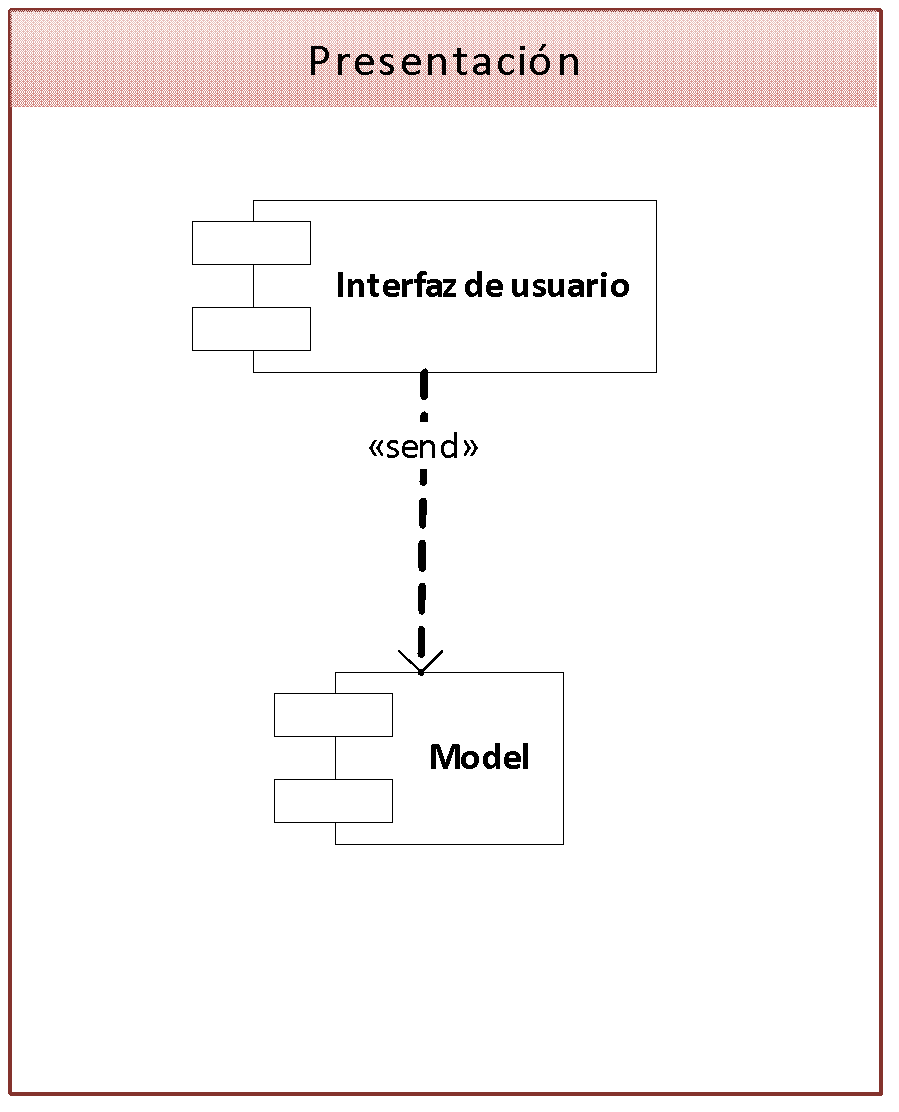
Modelo de Implementación: (Diagrama de componentes en capas)



## Capas

[Se deberá proveer para cada capa una sección con su nombre y la enumeración de los subsistemas asignados a la capa, así como un diagrama de componentes donde se muestren los componentes que conforman la capa, las dependencias entre ellos. Las interfaces requeridas y proporcionadas por cada componente, a fin de describir con suma precisión la integración.]

### Capa de Presentación



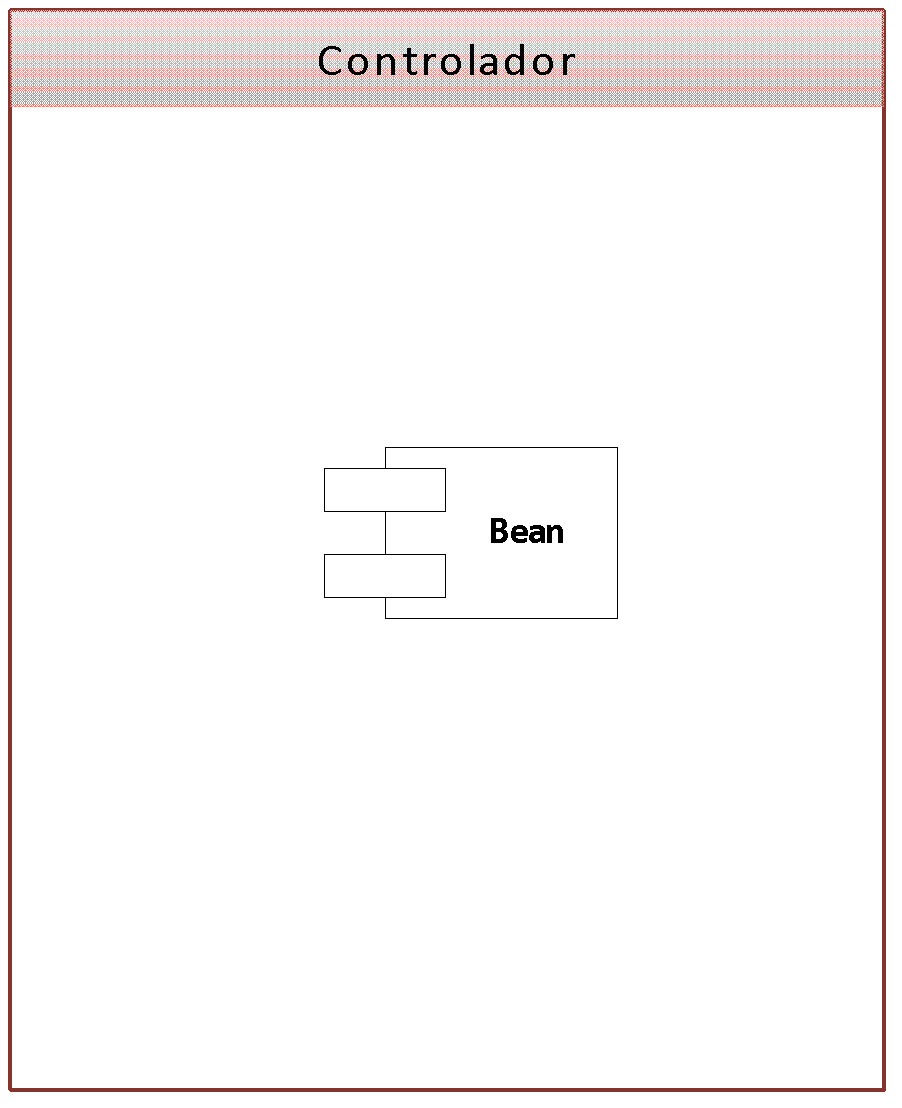
#### Interfaz:

#### Comprende de las páginas .xhtml para la vista por parte del usuario.

#### Model:

#### Contiene de las clases Java con los atributos de los formularios de las paginas xhtml.

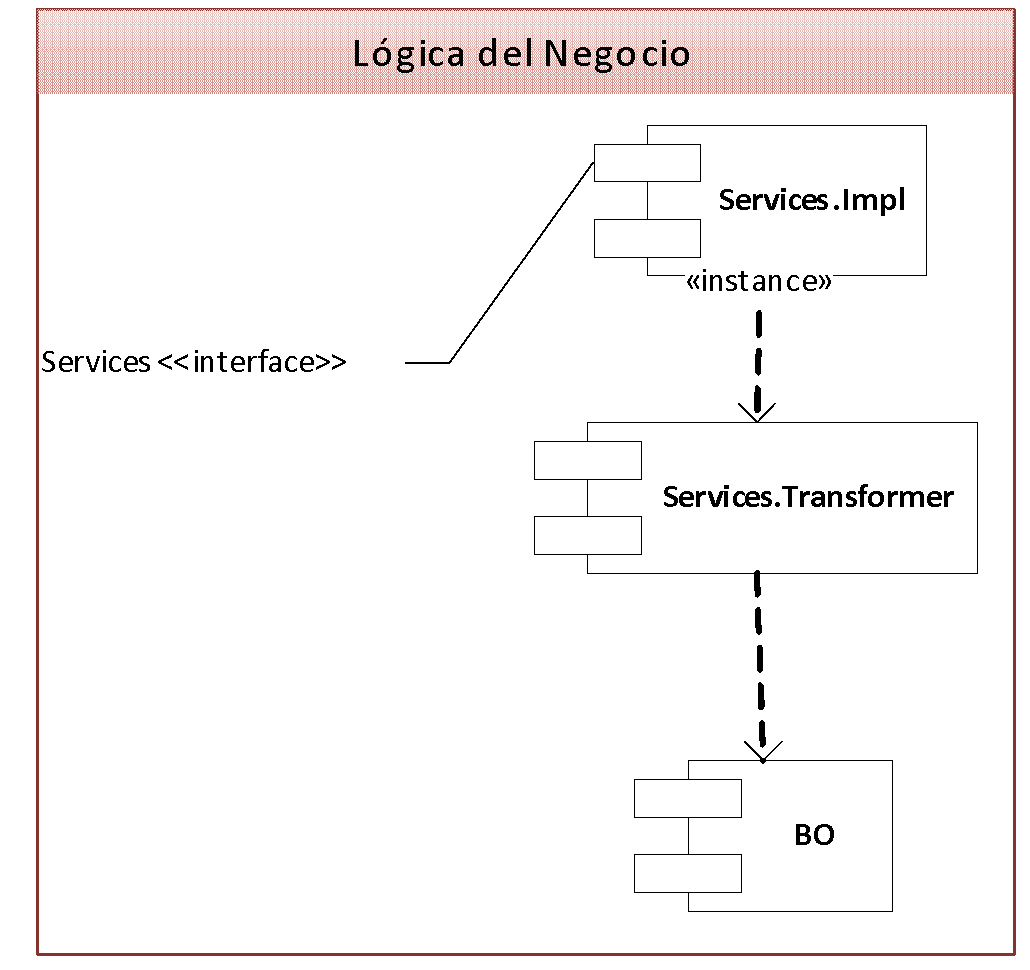
### Capa Controladora



#### Bean:

Clases Managed Bean encargadas de la lógica de la aplicación.

### Capa Lógica del Negocio



#### Services:

#### Son las clases de tipo interfaz encargadas de la declaración de los métodos de la lógica del negocio.

#### Services.Impl:

Clases Java que implementan la lógica del negocio.

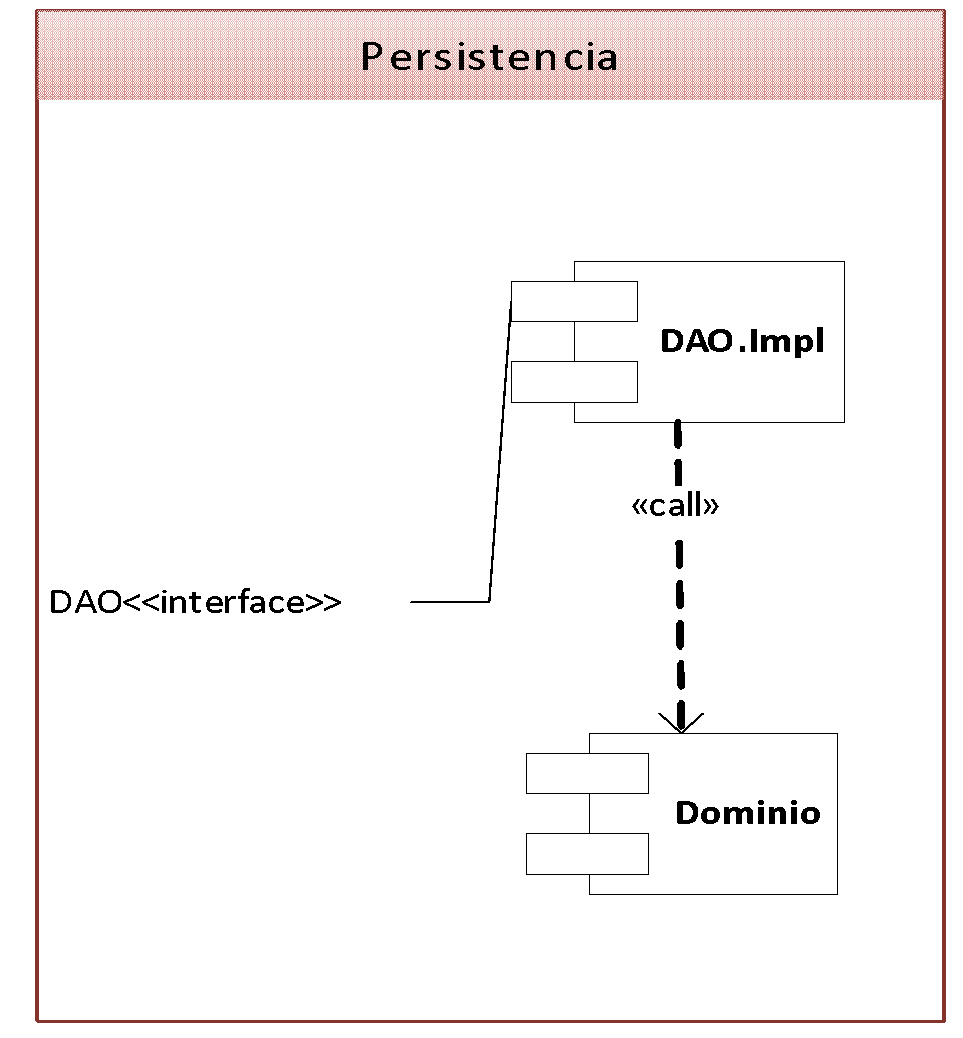
#### Services.Transformer:

Clases correspondientes con cada entidad de la aplicación que se encargan de la transformación de una clase BO a una clase entidad de la BD y de la misma forma, de una entidad de BD a una clase BO.

#### BO:

Clases Business Object correspondientes a cada entidad de BD de la aplicación.

### Capa de Persistencia



#### DAO:

#### Son las clases de tipo interfaz que poseen los métodos que interactúan con las clases entidad de la BD.

#### DAO.Impl:

Contiene la implementación de los métodos del paquete DAO.

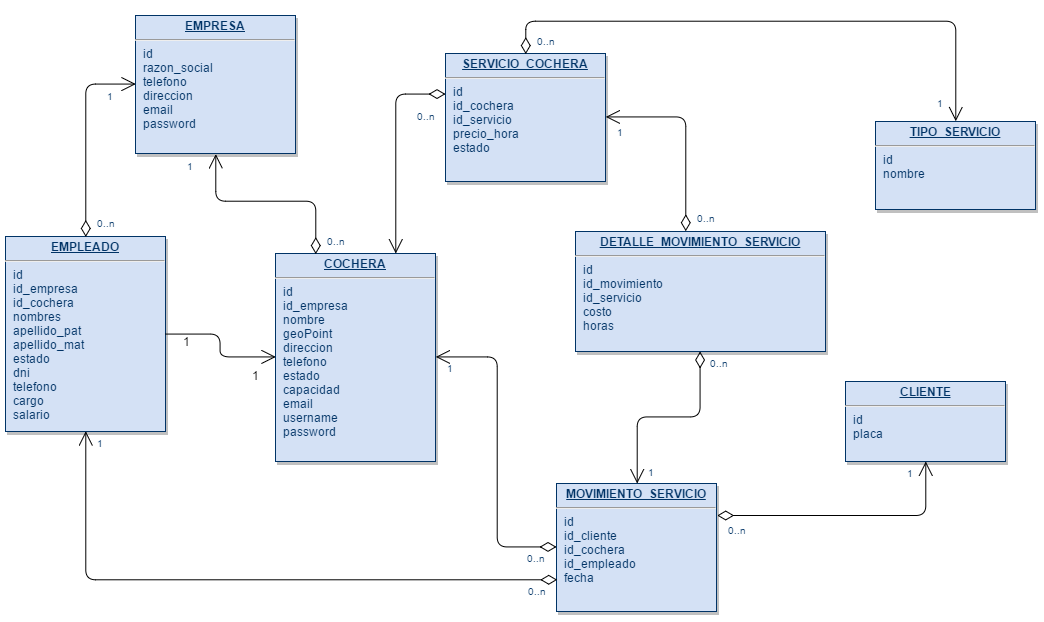
#### Dominio:

Contiene las clases java que representan a las entidades resultados del mapeo de la BD.

# Vista de Datos

En esta vista se incorpora la perspectiva del almacenamiento de datos del sistema para soportar los requerimientos de persistencia de la información en el tiempo.

Modelo Lógico de datos.



# Tamaño y desempeño

Aspectos relacionados a requerimientos no funcionales tales como desempeño, tiempos de respuestas, entre otros.

1. El sistema permitirá el acceso concurrente de varios usuarios, de modo que pueda realizarse transacciones simultáneas.
2. El programa a ser ejecutado en las computadoras denominadas terminales clientes del sistema Web, no debe de demandar más recursos que los que tendría una computadora como la especificada dentro de los requisitos de hardware mínimos para las computadoras clientes definidas en el Documento de Visión.
3. La Base de Datos del sistema será diseñada de acuerdo a patrones que permita una mayor eficiencia en las transacciones.

# Calidad

La estructura y los componentes del sistema están diseñados de manera que la confiabilidad del sistema sea mayor, pues se ha usado la Metodología RUP en donde la arquitectura es desarrollada en iteraciones durante las fases de incepción y elaboración, además de la Arquitectura especificada en el Doc. Arquitectura de Software del Banco de la Nación. Permite que a través de su diseño sean soportadas requerimientos funcionales y no funcionales detallados en el SIGE- Documento de Requerimientos de Software.

Historia de las Revisiones

|  |  |  |  |
| --- | --- | --- | --- |
| Fecha | Versión | Descripción | Autor |
| 21/05/2017 | 1.0 | Creación del Documento | Carlos Ramírez Vera |
|  |  |  |  |
|  |  |  |  |