Arquitectura del sistema

Facultad de Ingeniería – Consejo de Educación Técnico Profesional

Tecnólogo Informático

Montevideo – Uruguay

Índice

[Introducción 4](#_Toc374655504)

[Modelo de dominio 5](#_Toc374655505)

[Vista de Casos de Uso 6](#_Toc374655506)

[Casos de uso críticos 6](#_Toc374655507)

[Módulo Web 6](#_Toc374655508)

[Módulo Móvil 7](#_Toc374655509)

[Diagrama de casos de uso críticos 8](#_Toc374655510)

[Vista Lógica 8](#_Toc374655511)

[Arquitectura general de la aplicación 9](#_Toc374655512)

[Estilo Arquitectónico 9](#_Toc374655513)

[Subsistemas de la arquitectura 9](#_Toc374655514)

[Diagramas de Interacción 15](#_Toc374655515)

[Vista de Deployment 18](#_Toc374655516)

[Escenario de deploy para desarrollo 18](#_Toc374655517)

[Escenario de deploy para producción 18](#_Toc374655518)

[Vista de Implementación 20](#_Toc374655519)

**Índice de imágenes**

[Imagen 1: Modelo de dominio del sistema 5](#_Toc373788998)

[Imagen 2: Diagrama de casos de uso críticos 8](#_Toc373788999)

[Imagen 3: Arquitectura general de la aplicación 9](#_Toc373789000)

[Imagen 4: Arquitectura en Capas Lógicas 11](#_Toc373789001)

[Imagen 5: Capa de Servicios 13](#_Toc373789002)

[Imagen 6: Capa de Negocio 13](#_Toc373789003)

[Imagen 7: Capa de Persistencia 14](#_Toc373789004)

[Imagen 8: Diagrama de Secuencia Login 15](#_Toc373789005)

[Imagen 9: Diagrama de Secuencia Login – Móvil 15](#_Toc373789006)

[Imagen 10: Diagrama de Secuencia Login – Web 16](#_Toc373789007)

[Imagen 11: Diagrama de Secuencia Buscar contenidos 16](#_Toc373789008)

[Imagen 12: Diagrama de Secuencia Buscar contenidos – Web 16](#_Toc373789009)

[Imagen 13: Diagrama de Secuencia Buscar contenidos - Móvil 17](#_Toc373789010)

[Imagen 14: Diagrama de Despliegue de la aplicación (producción) 19](#_Toc373789011)

[Imagen 15: Diagrama de Despliegue de la aplicación (desarrollo) 19](#_Toc373789012)

[Imagen 16: Diagrama de Despliegue de Implementación 20](#_Toc373789013)

# Introducción

En este documento se describe la arquitectura del sistema a desarrollar, utilizando un enfoque basado en “vistas”, que representan al sistema bajo diferentes perspectivas y con diferente énfasis en un determinado aspecto.

Éste enfoque se denomina “4+1 View Model of Architecture” y presenta al sistema bajo 5 vistas: la Vista de Casos de Uso, la Vista Lógica, la Vista de Distribución, la vista de Implementación, y la vista de Procesos.

Los diagramas presentados en el presente documento son presentados en lenguaje UML en su versión 2.0.

# Modelo de dominio

Como resultado de la etapa de análisis se llegó a un modelo conceptual que presenta los principales conceptos del dominio del Marketplace a implementar. El mismo se presenta con el fin principal de dar un panorama general de la aplicación y a su vez complementar la siguiente sección correspondiente a la Vista de Casos de Uso.

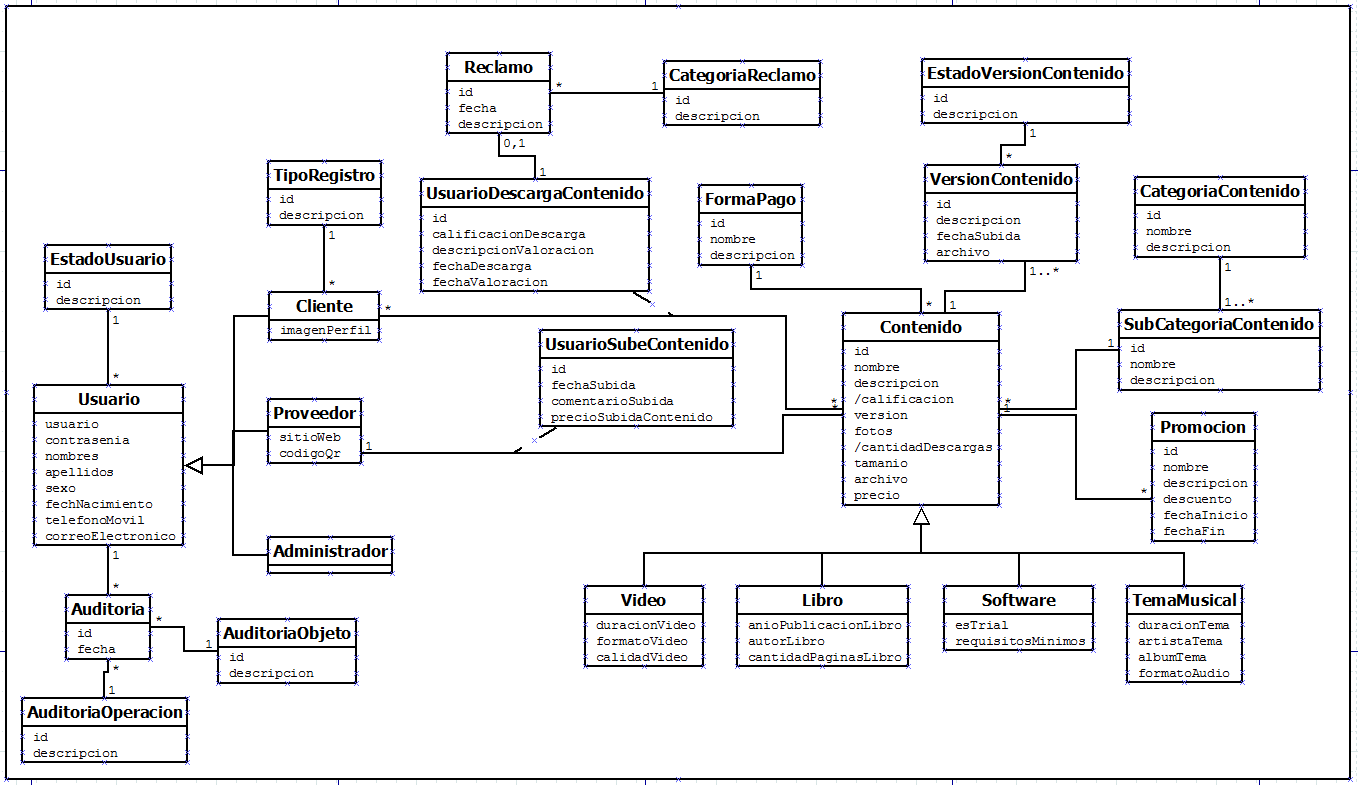


Imagen 1: Modelo de dominio del sistema

# Vista de Casos de Uso

La Vista de Casos de uso tiene como objetivo presentar un subconjunto del total de los casos de uso definidos para el sistema al cual se llamará “casos de uso críticos para la arquitectura del sistema”. Éstos casos de uso críticos se eligieron utilizando criterios de cobertura (que accionen el mayor número de componentes de la arquitectura), disparidad (que no hayan dos casos de uso similares), prioridad (que sean prioritarios dentro de todos los casos de uso definidos), complejidad (que posean lógica compleja) y riesgo tecnológico (que tengan una alta dependencia de módulos propios o de terceros que puedan representar riesgo).

Los casos de uso críticos para la arquitectura del sistema serán prototipados en una etapa temprana del desarrollo del sistema con el fin de validar la arquitectura.

## Casos de uso críticos

### Módulo Web

|  |  |
| --- | --- |
| **Identificador** | LOGIN\_WEB |
| **Nombre** | Login web |
| **Descripción** | Un usuario anónimo se autentica en el sistema a través de la página web. Para ello ingresa su usuario y contraseña. |

|  |  |
| --- | --- |
| **Identificador** | BC\_WEB |
| **Nombre** | Buscar contenidos web |
| **Descripción** | Un usuario anónimo o autenticado en el sistema busca un contenido a través de la página web, con la posibilidad de seleccionar la categoría del contenido entre las disponibles o ingresar el nombre del mismo. |

|  |  |
| --- | --- |
| **Identificador** | LOGOUT\_WEB |
| **Nombre** | Logout web |
| **Descripción** | Un usuario previamente autenticado en el sistema hace clic en el botón salir de la interfaz web del sistema quedando automáticamente como un usuario anónimo. |

## Módulo Móvil

|  |  |
| --- | --- |
| **Identificador** | LOGIN\_MOVIL |
| **Nombre** | Login a través de dispositivo móvil |
| **Descripción** | Un usuario anónimo se autentica en el sistema a través de la aplicación móvil del portal instalada en su dispositivo móvil Android. Para ello ingresa su usuario y contraseña. |

|  |  |
| --- | --- |
| **Identificador** | BC\_MOVIL |
| **Nombre** | Buscar contenidos desde dispositivo móvil |
| **Descripción** | Un usuario anónimo o autenticado en el sistema busca un contenido a través de la aplicación móvil, con la posibilidad de seleccionar la categoría del contenido entre las disponibles o ingresar el nombre del mismo. |

|  |  |
| --- | --- |
| **Identificador** | LOGOUT\_MOVIL |
| **Nombre** | Desincronización del dispositivo móvil |
| **Descripción** | Un usuario cuyo dispositivo móvil estaba sincronizado con la tienda hace clic en el botón desvincular quedando instantáneamente sin conectividad con la cuenta del usuario. |

## Diagrama de casos de uso críticos

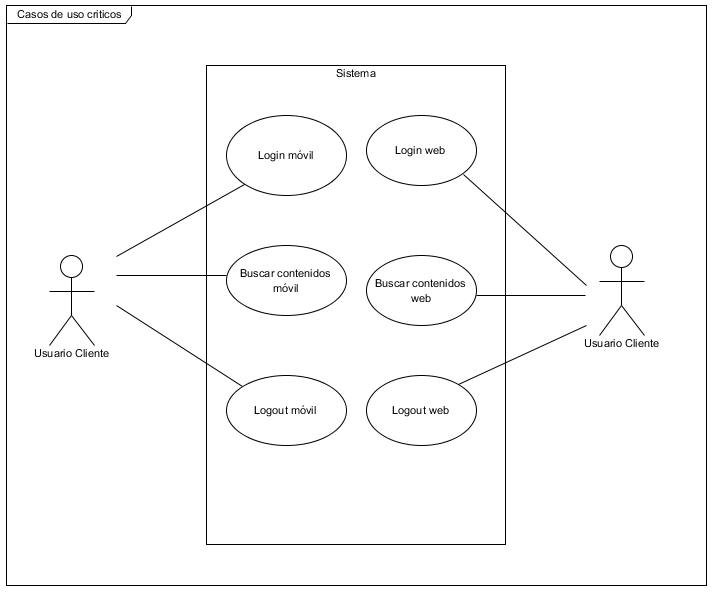


Imagen 2: Diagrama de casos de uso críticos

## Vista Lógica

La Vista Lógica permite describir el sistema en base a abstracciones fundamentales del diseño orientado a objetos para dar soporte a los requerimientos funcionales establecidos. Adoptando un enfoque *top-down*descompondremos al sistema en un conjunto de subsistemas, como pueden ser las capas lógicas, y a través de sucesivos refinamientos se llegará a las unidades lógicas más pequeñas.

## Arquitectura general de la aplicación

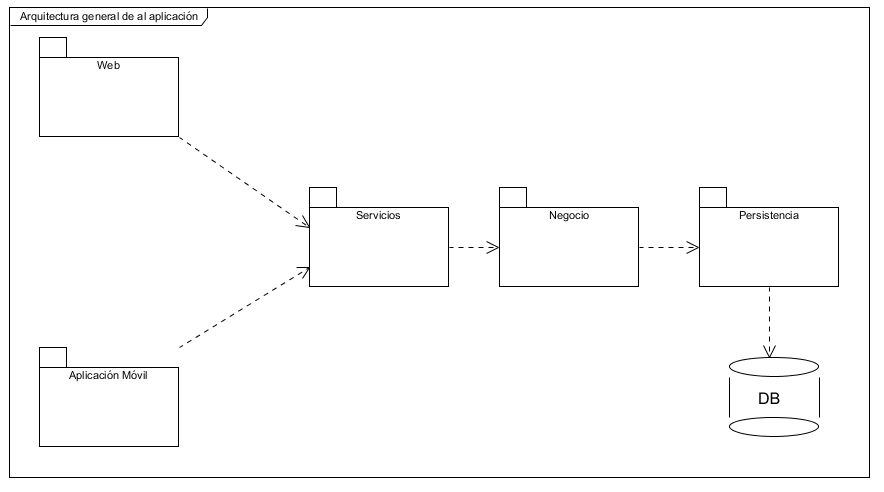


Imagen 3: Arquitectura general de la aplicación

## Estilo Arquitectónico

El estilo arquitectónico adoptado para el desarrollo de la solución es el de Capas Estrictas, donde cada capa consume interfaces (tiene visibilidad) de su inmediata inferior. En la siguiente imagen () se presenta el diagrama de componentes de la arquitectura previamente mencionada, en la cual se identifican cuatro subsistemas: Capa de Persistencia, Capa de Negocio, Capa de Servicios y Capa de Presentación estando las dos últimas a un mismo nivel ya que ambos dependen de la Capa de negocio y no tienen dependencias entre sí.

### Subsistemas de la arquitectura

En esta sección se dará una breve introducción cada Subsistema identificado para la arquitectura adoptada, así como una reseña de las unidades lógicas y funcionalidades que contienen.

#### Capa de Presentación

La capa de presentación contiene los componentes de la aplicación que conforman la interfaz de usuario y manejan las interacciones con los mismos.

Dado que la aplicación a desarrollar constará tanto de un modulo web como un modulo móvil, vale destacar algunas particularidades. En el caso de la aplicación web la Capa de presentación estará compuesta por paginas html 5, librerías de estilos así como la lógica necesaria para la invocación de los web services rest expuestos por la capa de servicios (lógica de presentación mayormente conformada por componentes JQuery).

En el caso del modulo de administración, se optó por la tecnología *JSF* (ó *Java Server Faces*) de presentación. Dicha tecnología se conforma básicamente por páginas .xhtml y *Managed Beans* (clases java) detrás de las mismas mapeándose a los datos.

Por otra parte la aplicación móvil estará conformada por vistas xml con elementos propios del lenguaje de programación Android. Detrás de las mismas, clases que extienden *Activity* escuchando los eventos desencadenados por los usuarios de la aplicación.

#### Capa de Servicios

La Capa de Servicios tiene como finalidad ofrecer un subconjunto de la lógica de negocio a clientes remotos, como pueden ser paginas html a través de pedidos ajax o la aplicación móvil para la recarga de contenidos. Dichos servicios serán expuestos a través de Servicios Web REST utilizando las tecnologías JAX-RS (implementación particular: RestEasy).

Los clientes remotos serán tanto la aplicación web como la aplicación móvil que consumirán dicha lógica con el fin de recargar contenido, llevar a cabo operaciones como pueden ser compra y descarga, calificación de contenidos, entre otros.

En el caso de la aplicación móvil, la capa de servicios se encarga de consumir los servicios REST expuestos por el servidor principal utilizando la api de Apache para las operaciones sobre http *Apache Http Components.*

En la se ilustra el diseño de la Capa de Servicios.

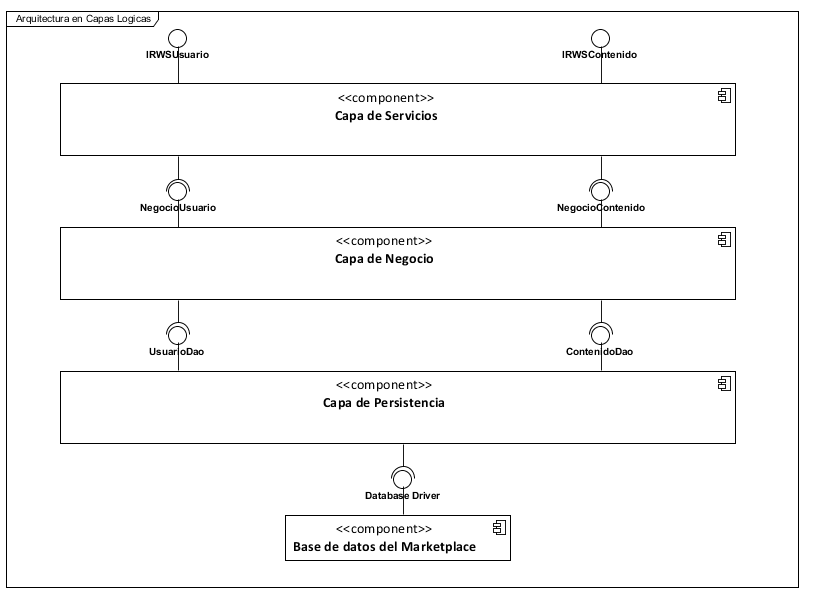


Imagen 4: Arquitectura en Capas Lógicas

#### Capa de Negocio

Este subsistema encapsula los componentes del sistema que realizar las operaciones que implementan las funcionalidades propiamente dichas. También conocida como Capa Lógica o Capa de dominio ésta capa contiene las clases de lógica que constituyen la fachada de la aplicación, ofreciendo una interfaz simplificada de las operaciones ofrecidas para cada elemento del dominio.

La fachada de la aplicación se implementa como un conjunto de clases (como son por ejemplo NegocioUsuario y NegocioContenido) que siguen el patrón *Session Facade*. Estas clases se implementan como *Statless Beans* de java e implementan las operaciones de casos de uso relacionados entre sí.

En la se ilustra el diseño de la Capa de Negocio o Capa Lógica.

#### Capa de Persistencia

El subsistema Persistencia se encarga del acceso a datos consumidos por la aplicación. Ofrece interfaces a la Capa de Negocios y es donde se realiza el mapeo de clases persistentes (comúnmente llamadas *Entities*) a las tablas de la base de datos PostgreSQL.

Para el caso de la aplicación móvil el objetivo de esta capa es la de respaldar la información y sesiones de usuario para permitir interactuar con la aplicación en caso de no tener conexión con el servidor principal. Se optó por la base de datos SQLite por la fácil integración con las tecnologías móviles Android.

En la se ilustra el diseño de la Capa Persistencia.

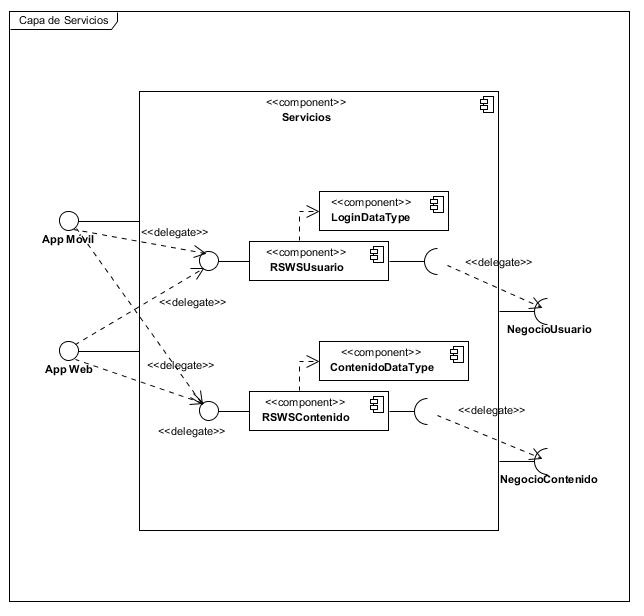


Imagen 5: Capa de Servicios

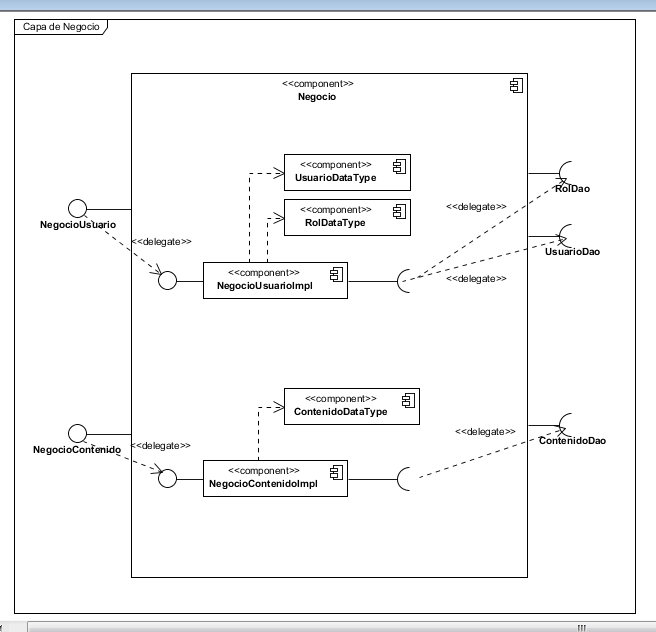


Imagen 6: Capa de Negocio

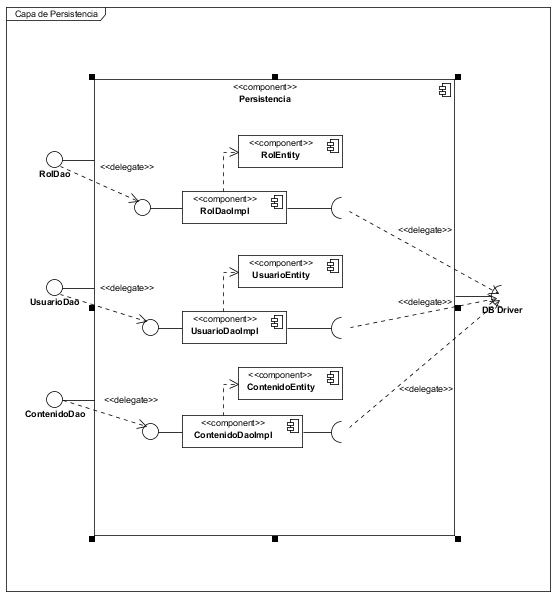


Imagen 7: Capa de Persistencia

### Diagramas de Interacción

En la presente sección se presentan las interacciones entre los diferentes componentes lógicos que conforman la aplicación para la realización de los casos de uso críticos para la arquitectura del sistema.

Para representar las interacciones entre dichos componentes se utilizarán los Diagramas de Secuencia del Sistema.

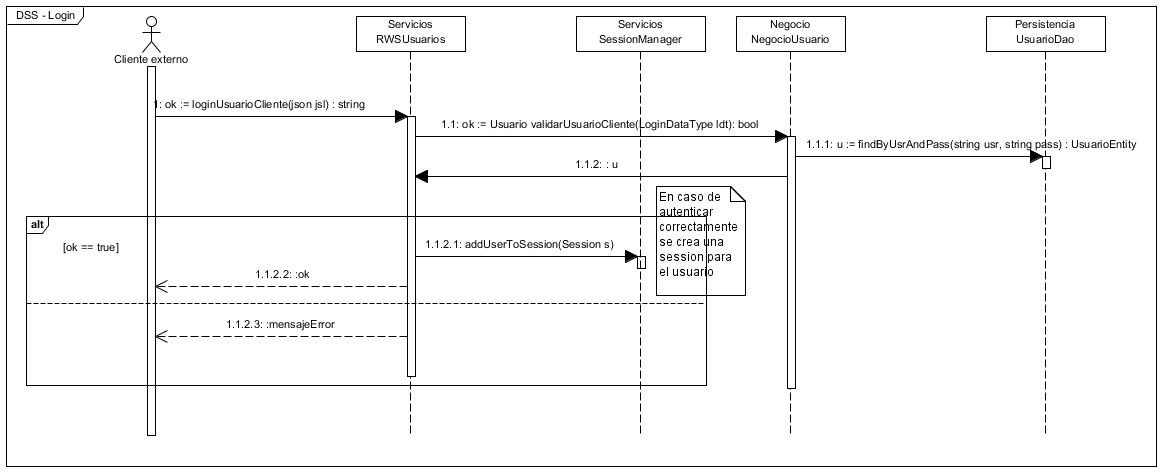


Imagen 8: Diagrama de Secuencia Login

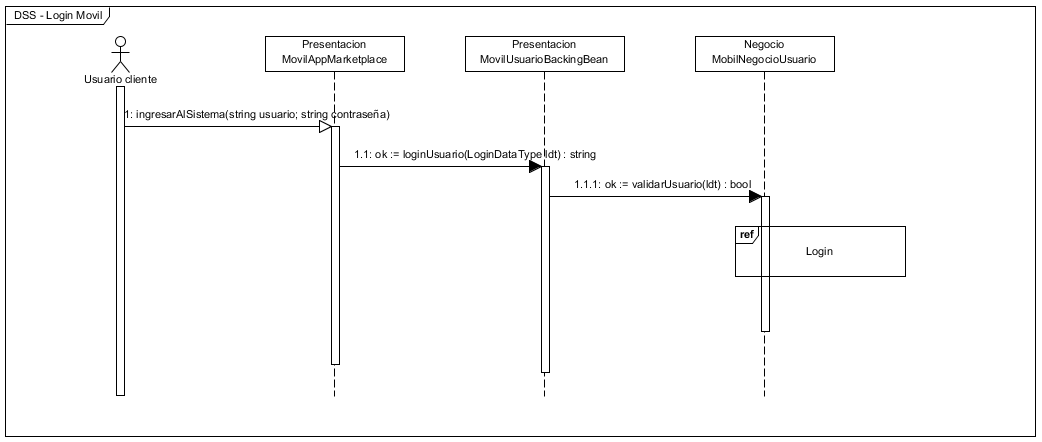


Imagen 9: Diagrama de Secuencia Login – Móvil

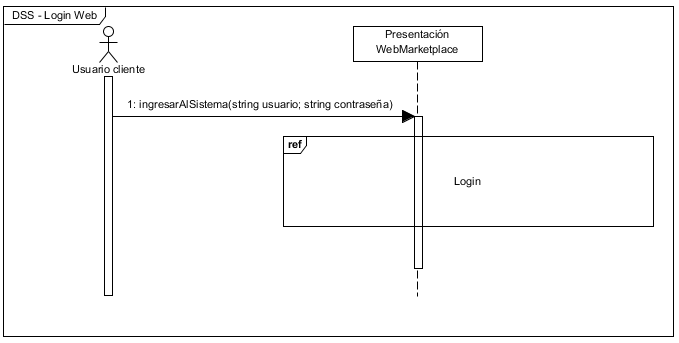


Imagen 10: Diagrama de Secuencia Login – Web

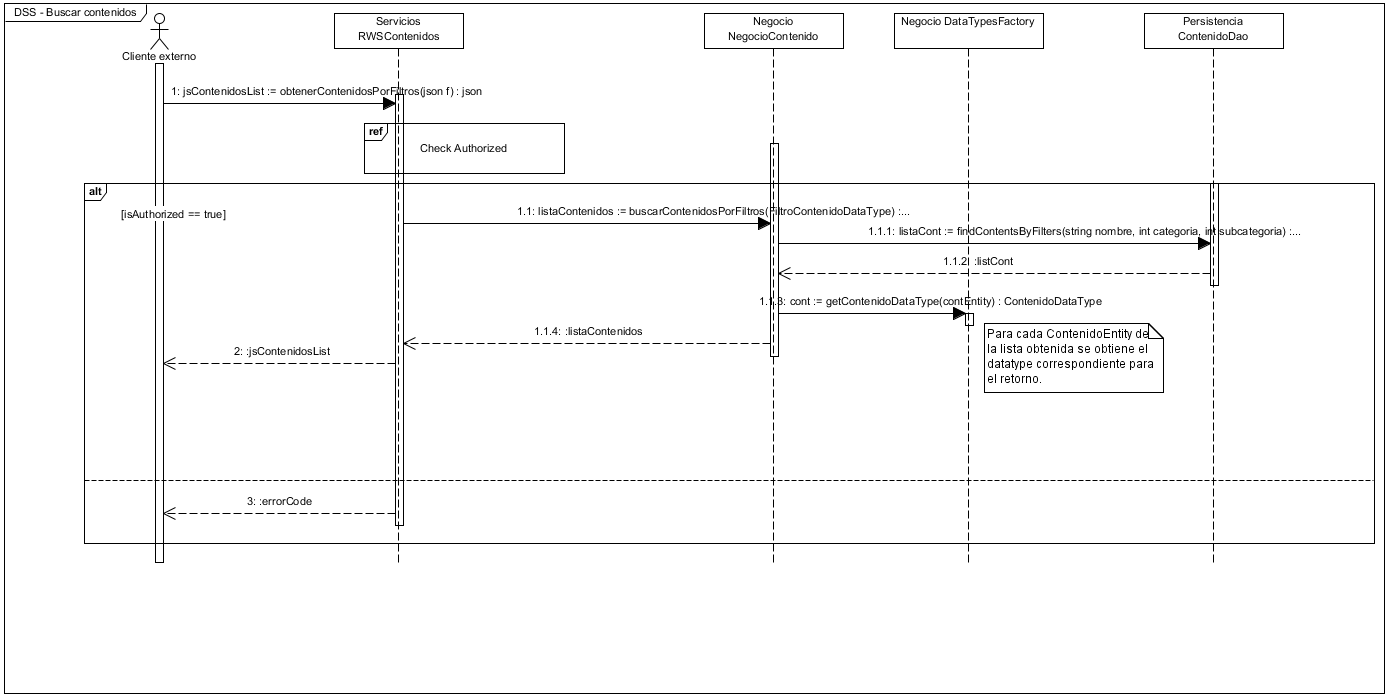


Imagen 11: Diagrama de Secuencia Buscar contenidos

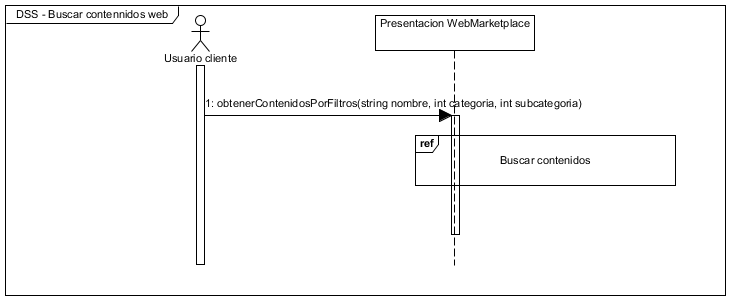


Imagen 12: Diagrama de Secuencia Buscar contenidos – Web

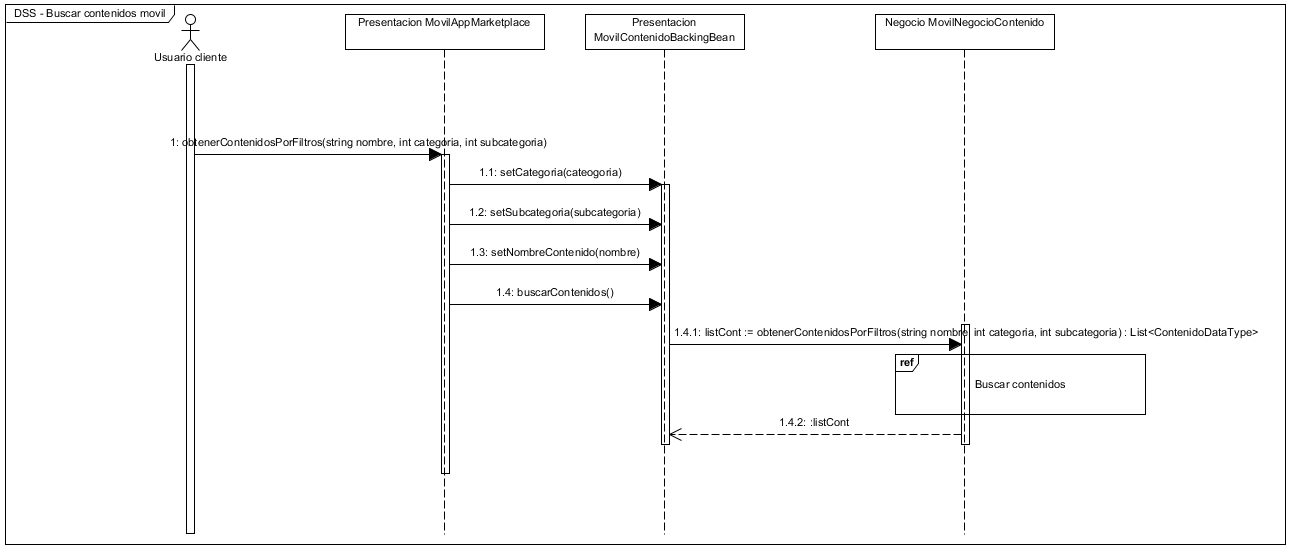


Imagen 13: Diagrama de Secuencia Buscar contenidos - Móvil

# Vista de Deployment

La Vista de Deployment o Vista de Distribución del sistema permite visualizar los posibles escenarios de asignación de componentes lógicos de la aplicación en diferentes nodos físicos o tiers.

En el caso del Marketplace a implementar podemos identificar los siguientes nodos físicos que conformarán el diagrama de distribución:

* **Cliente Web:** Computador que tiene el navegador que consumirá la interfaz web.
* **Browser:** un navegador web estándar instalado en el cliente. Ejemplos de éste tier pueden ser Google Chrome, Mozilla Firefox, Internet Explorer, entre otros.
* **Cliente Móvil:** dispositivo móvil que tiene instalada la aplicación móvil del Marketplace.
* **Aplicación Android:** Aplicación que consume directamente los web services expuestos por el sistema para dicha plataforma.
* **Servidor de aplicaciones:** Servidor que contiene el la aplicación Java Enterprise Edition, en este caso Jboss 7.1.1 Aplication Server.
* **Jboss 7.1.1 Application Server:** servidor de aplicaciones para sistemas desarrollados sobre Java Enterprise Edition.
* **Servidor de Base de Datos:** Computador que corre el motor de base de datos para la aplicación. En este caso PostgreSQL 9.2.
* **PostgreSQL 9.2 DataBase:** motor de base de datos elegido para el desarrollo.

## Escenario de deploy para desarrollo

El escenario para etapa de desarrollo involucra un único nodo, en el cual se ejecutan todos los componentes del sistema, como son Web browser, simulador Windows Phone, Servidor de Aplicaciones así como el motor de base de datos. Este esquema presenta gran simplicidad y bajo costo al momento de preparar el ambiente de desarrollo. Como contrapartida, presentaría problemas de performance, escalabilidad y seguridad para la aplicación si se considerara para un ambiente de producción.

## Escenario de deploy para producción

Este escenario presenta los componentes implantados en nodos totalmente independientes, un nodo corriendo el Servidor de Aplicaciones, otro con el Servidor de Base de Datos y otros nodos (clientes) ejecutando los browsers y las aplicaciones móviles consumiendo la aplicación. Esto promueve la escalabilidad y performance del sistema, así como la rápida recuperación del sistema ante posibles fallas en alguno de los nodos.

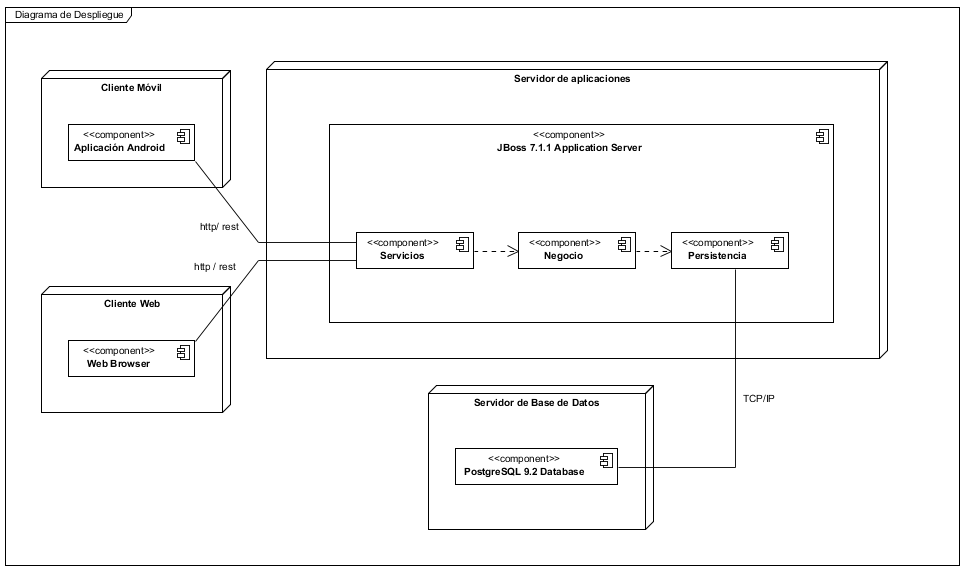


Imagen 14: Diagrama de Despliegue de la aplicación (producción)

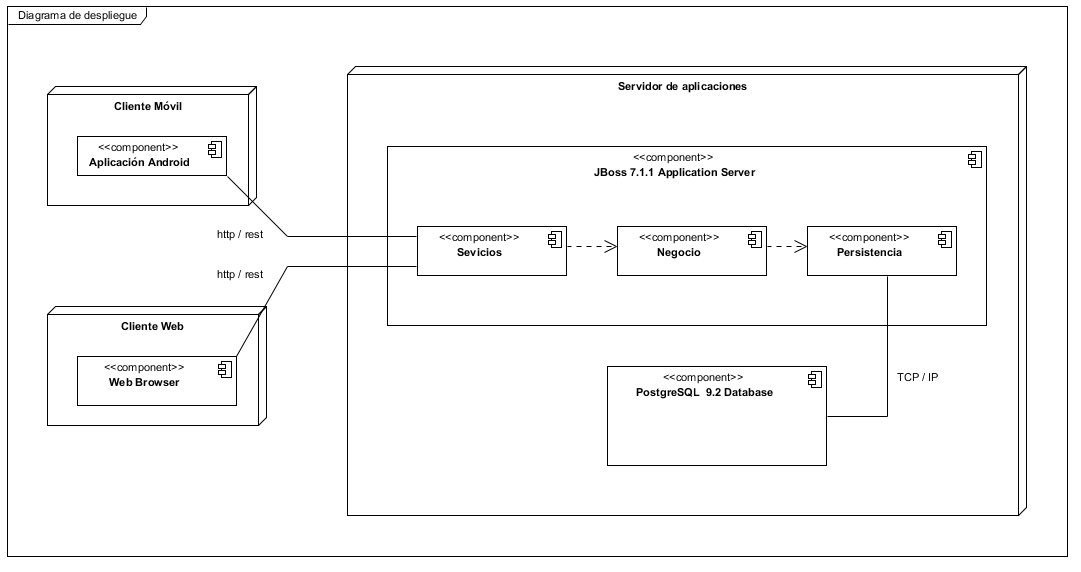


Imagen 15: Diagrama de Despliegue de la aplicación (desarrollo)

# Vista de Implementación

La Vista de implementación se focaliza en los componentes que forman parte del sistema, en otras palabras el resultado de la implementación de los diferentes subsistemas que conforman la solución. En el siguiente diagrama se ilustra el empaquetado de los módulos que componen la aplicación Java Enterprise Edition, los cuales corresponden con las Capas (o subsistemas) de la vista lógica.

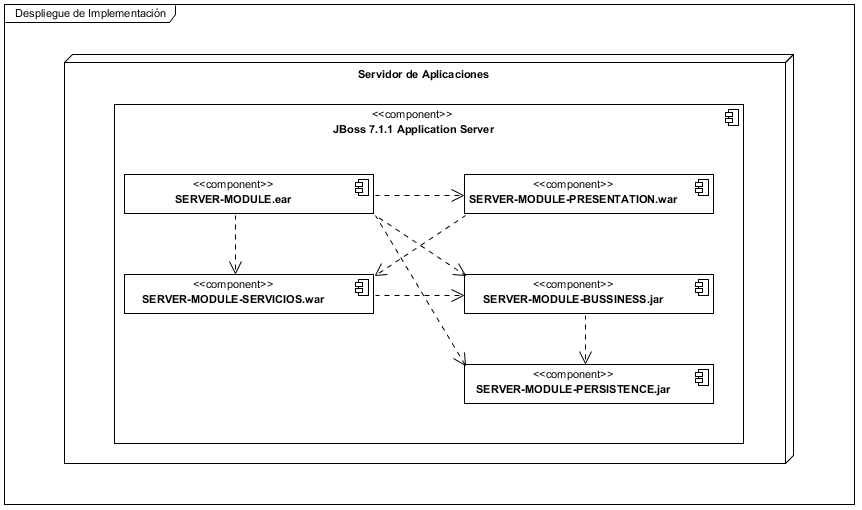


Imagen 16: Diagrama de Despliegue de Implementación