Pedidos Online - DUSA
Descripción de la Arquitectura
Versión 2.1

Historia de revisiones

Fecha	Versión	Descripción	Autor
28/08/2013	1.0	Versión Inicial	Joana Miller
31/08/2013	1.1	Modificaciones Generales	Joana Miller
01/09/2013	1.2	Revisión Vista del Modelo Implementación	Joana Miller
01/09/2013	1.3	Revisión final Responsable SQA	Verónica Gamarra
15/09/2013	2.0	Modificaciones Generales, Patrón MVC, Vista Modelo Diseño	Joana Miller
15/09/2013	2.1	Revisión final Responsable SQA	Verónica Gamarra

Índice

1. Introducción 4

1.1. Propósito 4
1.2. Alcance 4
1.3. Definiciones, siglas y abreviaturas. 4
1.4. Referencias 4
1.5. Visión general 4

2. Vista del Modelo de Casos de Uso 5

2.0. Sobre los Usuarios Identificados En el Modelo de Casos de Uso se identificaron los siguientes usuarios: 5
2.1. Diagrama de Casos de Uso relevantes a la Arquitectura 5
2.2. Casos de Uso relevantes a la Arquitectura 6

2.2.1. Iniciar Pedido 6
2.2.2. Confirmar Pedido 7
2.2.3. Buscar Producto 7
2.2.4. Ver Historial de Pedidos 7
2.2.5. Configurar Notificaciones 7

3. Trazabilidad desde el Modelo de Casos de Uso al Modelo de Diseño 8

3.1. Iniciar Pedido 8
3.2. Confirmar Pedido 9
3.3. Buscar Producto 9
3.4. Ver Historial de Pedidos 10
3.5. Configurar Notificaciones 10

4. Vista del Modelo de Diseño 11

4.0. Descripción de Arquitectura: Estilo Cliente - Servidor 11
4.1. Descomposición en Subsistemas 12

4.1.1. Subsistema: WEB-APP 13
4.1.2. Subsistema LogicServer 16
4.1.3. Subsistema SOLR 17

4.2. Diseño de Clases 17
4.3. Diseño de Casos de Uso 18

5. Vista del Modelo de Implementación 18

5.1. SubSistemas 20

5.1.1. LogicServer 20
5.1.2. WEB-APP 20
5.1.3. SearchEngine 20

5.2. Componentes 20

5.2.1. User Controller 20
5.2.2. Pedidos Controller 20
5.2.3. Product Controller 20
5.2.4. Admin Controller 21
5.2.5. AccessDB 21
5.2.6. Services 21
5.2.7. FactoryController 21

5.3. Interfases 21

5.3.1. WebServices 21
5.3.2. SOLRInterface 22
5.3.3. IPersistence 22

6. Vista del Modelo de Distribución 22
6.1. Diagramas de Distribución 22
6.2. Nodos 23
6.2.1. Servidor Felicia 23
6.2.2. Servidor 10 23
6.2.3. Servidor 11 24
6.2.4. Sistemas Externos (Sistema Farmacias & Sistemas de D.U.S.A.) 24
6.3. Conexiones 24
6.3.1. WebServer Felicia - Servidor 10 24
6.3.2. Servidor 10 - Servidor 11 (Base Datos) 24
6.3.3. Servidor 10 - Nodos Externos (Sistemas Externos) 24
6.3.4. Diagrama de Red 24

1. Introducción

1.1. Propósito

Este documento brinda un acercamiento global y comprensible de la Arquitectura del sistema a desarrollar, utilizando diferentes puntos de vista para mostrar los distintos aspectos del sistema.

Se detallarán las decisiones de Arquitectura críticas que se realizaron para cumplir con los requerimientos del cliente, para ello se busca presentar el diseño del sistema mediante distintas vistas conceptuales, desde la vista del Modelo de Casos de Uso, Modelo de Diseño, Modelo de Implementación, Modelo de Distribución y las trazas entre estos.

1.2. Alcance

Este documento está enfocado principalmente a los implementadores y diseñadores del sistema para que profundicen en las tecnologías y los patrones de diseño que se consideran en la solución del sistema, ampliando la base de conocimiento de los actores responsables del futuro mantenimiento del sistema.

1.3. Definiciones, siglas y abreviaturas.

Se recomienda recurrir al Glosario del Proyecto, de toda forma la siguiente tabla describe los términos propios de este documento.

Concepto	Descripción
Aplicación Web	Referencia al front end del Sistema
LogicServer	Referencia al Servidor de Lógica de Negocio

1.4. Referencias

En la siguiente tabla se listan nombre y descripción de los documentos que son referenciados en la presente Descripción de la Arquitectura.

Nombre	Descripción
RQDRQG03v4.3	Especificación de requerimientos
RQMODG03v3.1	Modelo de Casos de Uso
DSMDIG03v1.2	Modelo de Diseño

1.5. Visión general

Teniendo como entrada el documento de Modelo de Caso de Uso, se seleccionaron los casos de uso que resultan relevantes para la definición de la Arquitectura, en las siguientes secciones se presentan los casos de uso relevantes y la trazabilidad de los mismos a lo largo de las vistas mencionadas en el Propósito de este documento.

2. Vista del Modelo de Casos de Uso

En las siguientes secciones se describe la Arquitectura del sistema desde el punto de vista del modelo de casos de uso, se ilustran los diagramas de los casos de uso que se consideran funcionalmente importantes para el cliente, y que ayudan a mitigar riesgos técnicos.

2.0. Sobre los Usuarios Identificados

En el Modelo de Casos de Uso se identificaron los siguientes usuarios:

- Usuario Administrador
- Usuario-Web
    - Usuario-Dueño
    - Usuario-Empleado

Estos usuarios de alguna forma condicionan ciertos aspectos del diseño de la Arquitectura, a continuación se listan algunas condiciones que deben de ser contempladas por la Arquitectura:

- La categorización de Usuarios fuerza a diseñar la mayoría de casos de uso con un actor central: el usuario (Usuario-Web en la mayoría de casos de uso) de esta forma el diseño tiene que contemplar que sea un Usuario-Dueño o Usuario-Empleado el que realice un caso de uso que aplica a Usuario-Web, se debe resolver de la misma forma, reutilizando toda la lógica que sea posible.

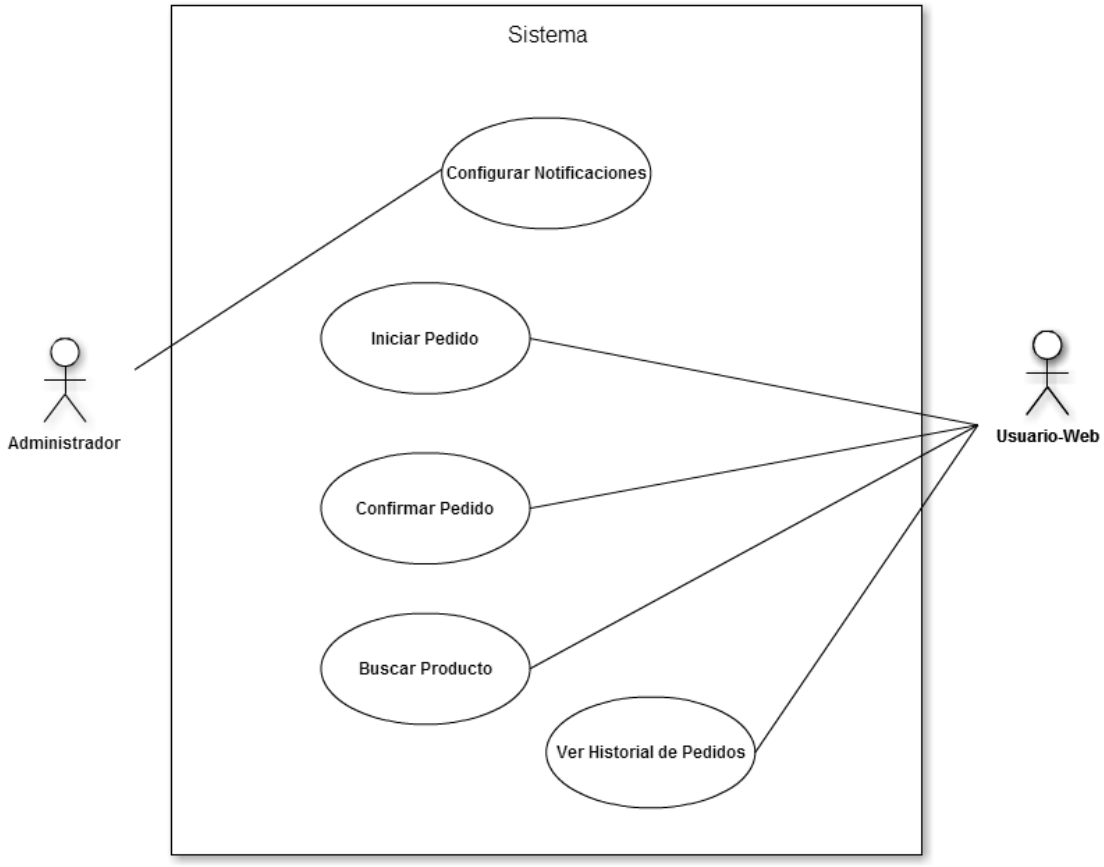
- Aquellos casos de uso que tienen como actor principal al Administrador deben ser contemplados en un módulo independiente, dado que el Producto Final servirá de base para futuros desarrollos de D.U.S.A.. De esta forma se garantiza que las funcionalidades del Usuario Administrador puedan ser extensibles y modificables en un mismo módulo dentro del Sistema, sin necesidad de modificar otros componentes, salvo aquellos que publican los servicios hacia el fron-end.

2.1. Diagrama de Casos de Uso relevantes a la Arquitectura

En esta sección se describen los casos de uso que fueron condicionantes a la hora de diseñar la Arquitectura del Sistema. La principal razón para seleccionar estos casos de uso fue que la resolución de estos obliga a que los datos pasen por varias capas lógicas (de presentación, de lógica de negocio, comunicación, persistencia y otras). De esta forma se comienza a definir el nivel de abstracción que tendrá el diseño de la Arquitectura.

A continuación se presenta el diagrama de casos de uso, que describe los casos de uso mencionados en el párrafo anterior.

En la sección siguiente se describe brevemente cada caso y se presenta un refinamiento de la solución para cada uno, este refinamiento servirá como base para definir los paquetes de software necesarios en el diseño del Sistema.



- Diagrama 1-

2.2. Casos de Uso relevantes a la Arquitectura

En las siguientes subsecciones se presenta una breve descripción de los casos de uso que son considerados en el Diagrama 1 y un refinamiento planteado para la solución de los mismos. En estos refinamientos se hace mención a componentes de software que formarían parte de la solución del caso de uso.

2.2.1. Iniciar Pedido

Descripción:

El sistema permitirá a un Usuario-Web iniciar un pedido donde podrá agregar/quitar productos y modificar la información propia del pedido (como reparto asignado, condición de compra, fecha).

Refinamiento:

La resolución de este caso de uso puede darse por medio de 3 capas de procesamiento, la primera la Aplicación Web que dispara el casos de uso, luego una capa de procesamiento de lógica de negocio que podríamos identificar como LogicServer, que registrará o leerá de una base de datos la información necesaria para ejecutar el caso.

La dificultad de este caso de uso reside en tener un control de concurrencia de usuarios, dado que existirá un único pedido activo (iniciado) por farmacia, mientras que cada farmacia puede tener hasta 5 Usuarios-Web asociados.

2.2.2. Confirmar Pedido

Descripción:

El sistema permitirá a un Usuario-Web confirmar el pedido activo asociado a la sesión del Usuario-Web.

Refinamiento:

En este caso de uso participarán componentes de todas las capas del Sistema, la aplicación web que disparará el caso de uso, el LogicServer que resolverá el caso de uso invocando a paquetes de otras capas como ser base de datos y servicios hacia D.U.S.A., para informar que se debe despachar un nuevo pedido.

2.2.3. Buscar Producto

Descripción:

El sistema permitirá a un Usuario-Web realizar la búsqueda de un producto sobre el catálogo de productos ofrecido por D.U.S.A., el Usuario accederá a una búsqueda indexada por nombre, laboratorio, descripción de producto.

Refinamiento:

Este caso de uso, será invocado desde la capa de presentación (la Aplicación Web) pero resuelto en su mayoría sobre la capa de procesamiento (LogicServer) que mediante un componente local (será parte del LogicServer en un principio) realizará las configuraciones necesarias para poder implementar una búsqueda sobre otro componente que tendrá cargada la información indexada sobre la que se efectúa la búsqueda. Para este caso de uso se hará uso de una tecnología ágil como SOLR, un servidor para indexación de documentos (VER DIAGRAMA DE PAQUETE)

2.2.4. Ver Historial de Pedidos

Descripción:

El sistema permitirá a un Usuario-Web consultar el estado de los pedidos realizados en la última semana respecto de la fecha actual.

Refinamiento:

Este caso de uso, será invocado desde la capa de presentación (la Aplicación Web) pero resuelto en su mayoría sobre la capa de procesamiento (LogicServer) que tendrá que interactuar con dos capas (de comunicación hacia sistema de D.U.S.A.) y de acceso a base de datos para relevar la información necesaria sobre el estado de los pedidos a mostrar en el histórico.

2.2.5. Configurar Notificaciones

Descripción:

El sistema permitirá a un Usuario-Administrador configurar las notificaciones que recibirán los Usuario-Web al iniciar sesión, estas notificaciones pueden ser

disparadas a todos los Usuarios-Web o a algunos en particular, con mensajes personalizados.

Refinamiento:

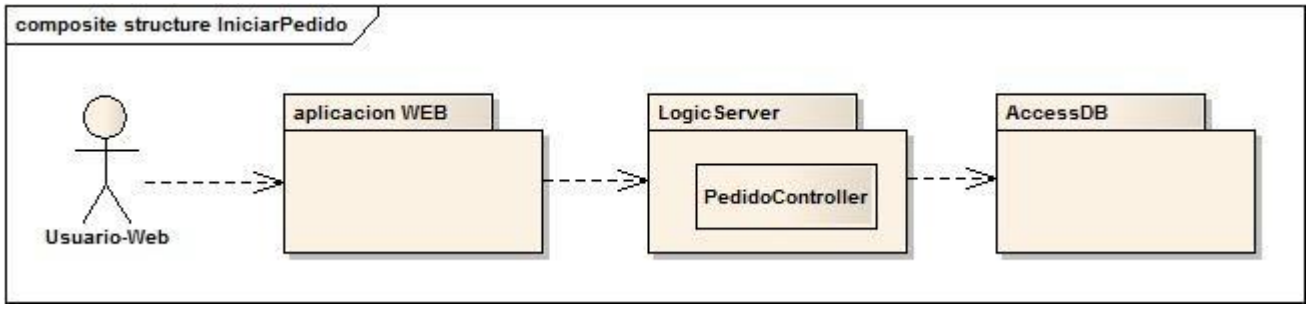
Este caso de uso, será invocado desde la capa de presentación (la Aplicación Web) pero resuelto en su mayoría sobre la capa de procesamiento (LogicServer) donde el módulo destinado para las funcionalidades del Administrador del sitio se encargará de comunicarse con la base de datos indicada para obtener las ofertas a notificar, o la información del cliente a notificar. El acceso a la información bien podría ser por la base de datos propia del Sistema "Pedidos Online - DUSA" donde se obtiene información sobre los Usuarios-Web como también acceder a la base de datos de D.U.S.A. donde se puede obtener información sobre ofertas, campañas, bonificaciones de productos.

3. Trazabilidad desde el Modelo de Casos de Uso al Modelo de Diseño

En esta sección se presentan los diagramas de paquetes de software que describen el proceso que se presentó en los refinamientos de los casos de usos mencionados en la sección anterior.

3.1. Iniciar Pedido

Diagrama de Paquete:

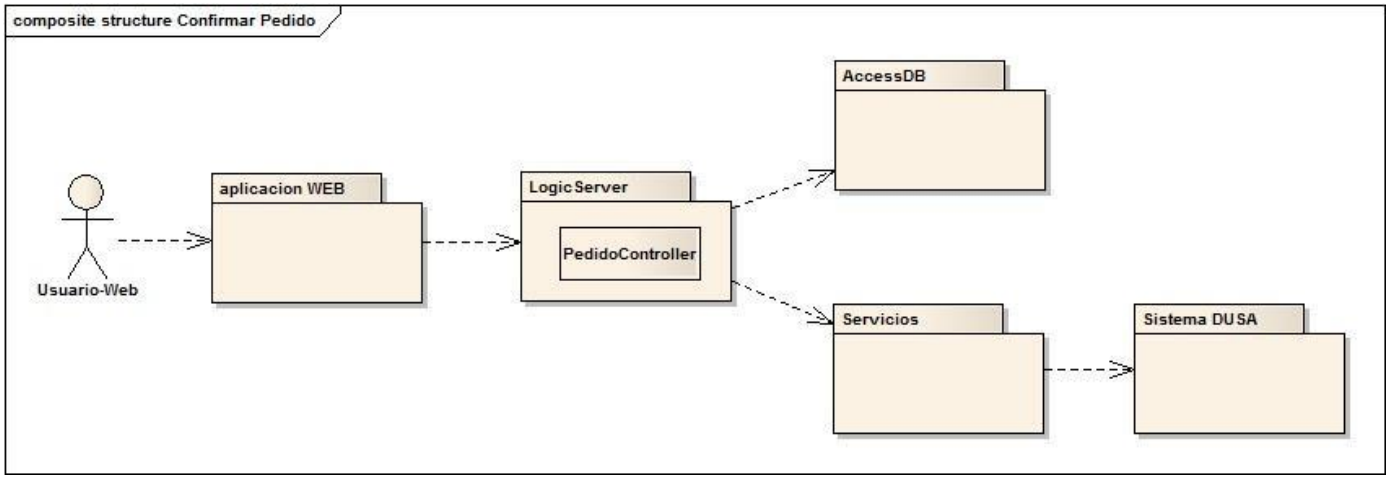


El componente Pedido Controller será el encargado de resolver la lógica de negocio de este caso de uso.

El paquete AccessDB refiere al acceso a la base de datos local al Sistema "Pedidos Online - DUSA", donde se persistirán la información propia de la aplicación, formará parte del Sistema de Base de Datos de D.U.S.A., pero el acceso estará brindado desde el módulo AccesDB para el Sistema "Pedidos Online - DUSA".

3.2. Confirmar Pedido

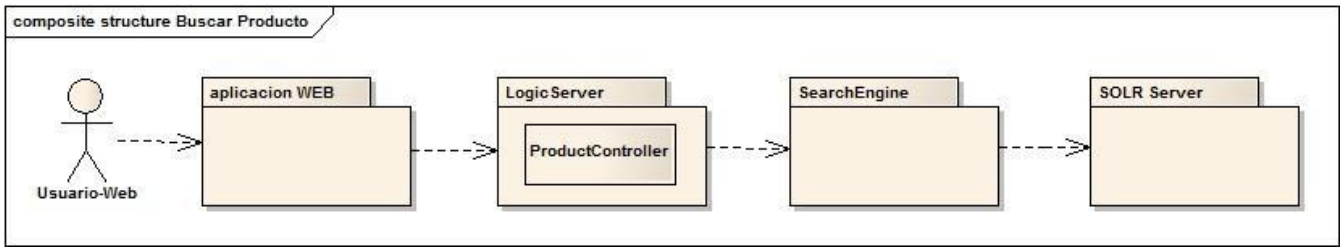
Diagrama de Paquete:



Una vez confirmado un pedido, el módulo PedidoController resolverá la lógica de negocio, registrará el pedido en la base de datos mediante AccessDB y enviará el pedido confirmado al Sistema de Despacho de Pedidos de D.U.S.A. mediante el paquete de Servicios (que bien podría ser mandar un web service, o registrar en una base de datos que el Sistema de D.U.S.A. consulte, en este sentido el diseño contempla cualquiera de las implementaciones mencionadas)

3.3. Buscar Producto

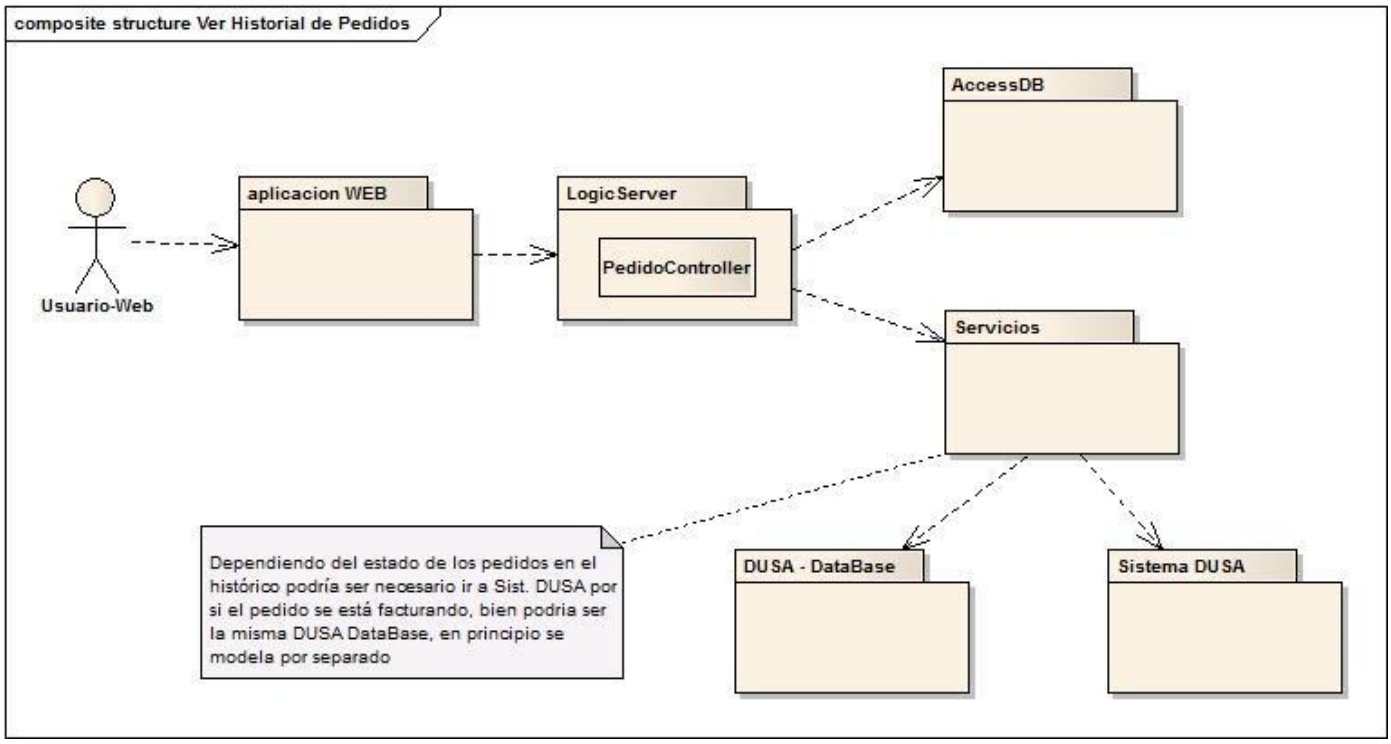
Diagrama de Paquete:



La búsqueda de un producto se resolverá entre los paquetes SearchEngine, que instanciará y cargará la consulta en SQLServer de quien recibirá los productos a mostrar en la búsqueda. Este caso de uso es disparado desde el LogicServer por el módulo ProductController.

3.4. Ver Historial de Pedidos

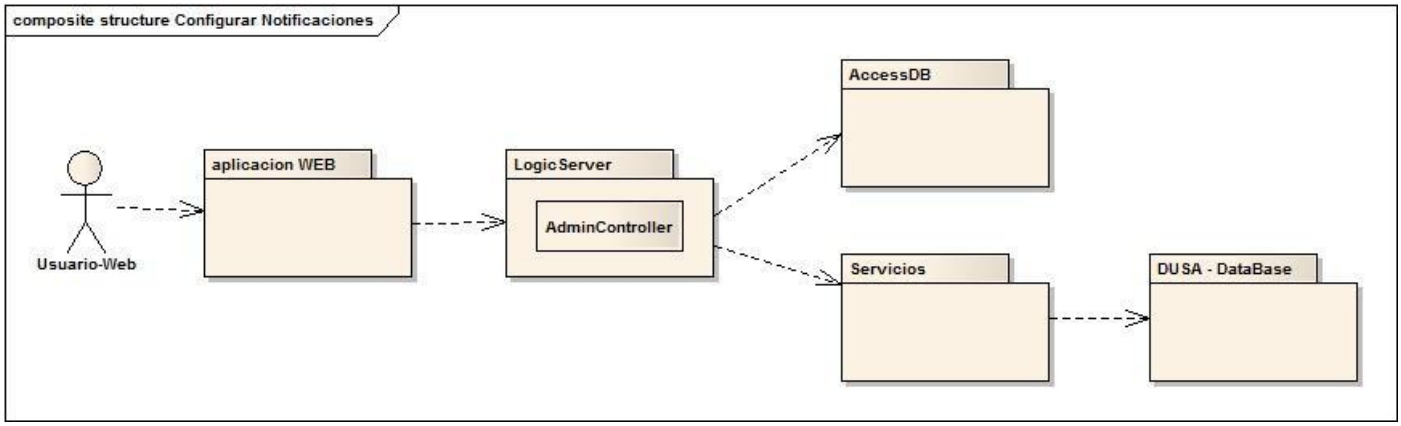
Diagrama de Paquete:



Dependiendo del estado de los pedidos a listar en el histórico, puede ser necesario consultar al Sistema de Despacho de pedidos de D.U.S.A. sobre el estado del pedido, o bien directamente a la base de datos de D.U.S.A., en cualquiera de los casos el diseño contempla las dos posibles implementaciones, de esta forma, desde Servicios se obtendría la información del estado del pedido y desde AccessDB la información sobre el Usuario y sesión que solicitó ver el historial.

3.5. Configurar Notificaciones

Diagrama de Paquete:



Este caso de uso como enfatizamos será resuelto por el módulo AdminController dedicado a implementar las funcionalidades del Usuario-Administrador del sitio, la

información que sea necesaria para las notificaciones será consumida directamente sobre la base de datos de D.U.S.A. mediante el módulo de Servicios.

4. Vista del Modelo de Diseño

4.0. Descripción de Arquitectura: Estilo Cliente - Servidor

Esta sección describe la Arquitectura considerada para la solución de "Pedidos Online - DUSA"

La Arquitectura de la solución se basará en parte en el estilo Cliente-Servidor en 4 Niveles, realizando una distribución de las capas lógicas de la aplicación bien distintiva, donde la gestión de los datos no sea gestionada por clientes ni servidores, de esta forma logramos reducir el tráfico en la red y otorgar mayor seguridad a los datos del cliente.

Por otro lado la centralización del acceso a la Base de Datos aporta la facilidad de actualización de los métodos de acceso, si en un futuro se realizara una migración o cualquier actualización que impacte en la forma de acceder a los datos, alcanzaría con actualizar la capa de procesamiento, para el resto de las capas esta actualización debería ser transparente.

La independencia de la lógica de procesamiento en una capa específica, centraliza la actualización de reglas y lógica de negocio que pudieran ser manifestadas por el Cliente, ofrece también facilidad de integración con futuros módulos que implementen nuevas funcionalidades como ser *Reclamos & Tracking de Reclamos* estos nuevos módulos serían integrados a la capa de procesamiento y luego se reflejarían las funcionalidades nuevas en las interfaces correspondientes.

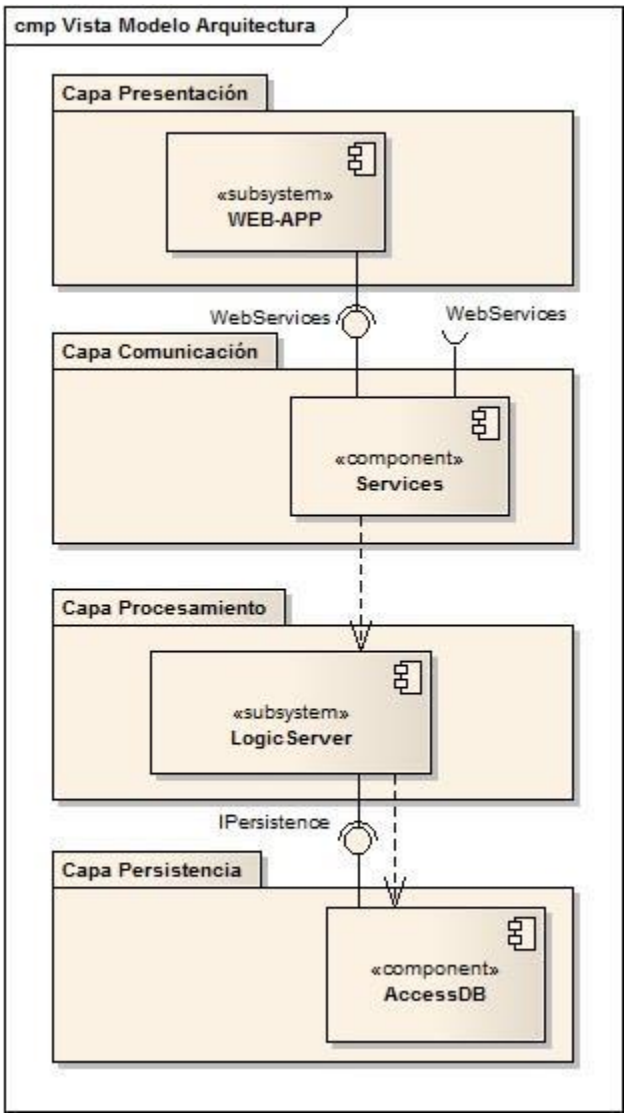
La centralización de front-ends en la capa de Presentación brinda una prolija solución para incorporar futuros desarrollos de aplicaciones destinadas a usuarios finales como una versión para Dispositivos Móviles

Uno de los requerimientos no funcionales relevados consistía en diseñar una Arquitectura que fuera lo suficientemente flexible para integrar el producto final con otros Sistemas Externos (propios de DUSA o incluso de los clientes de DUSA como Sistemas de Facturación de las farmacias). Para satisfacer este requerimiento se decide establecer una capa de comunicación que sirva como interfaz para comunicar el Sistema-PODUSA con otros Sistemas. *

De esta forma quedan identificados los Niveles de la siguiente manera:

- Un primer nivel que corresponde a la capa de presentación donde se encuentra la aplicación web (WEB-APP el front-end visible para los usuarios finales).
- El segundo nivel corresponde a la capa de comunicación donde se encuentra un componente encargado de establecer la comunicación con la aplicación web y los sistemas de D.U.S.A. (Services)
- En un tercer nivel la capa de procesamiento donde se encuentra el LogicServer que envuelve toda la lógica de negocio y procesamiento de información necesaria para cumplir con los requerimientos funcionales del sistema.
- Un cuarto nivel que se corresponde con la capa de Persistencia, donde se encuentran las componentes para el Acceso a Base de Datos propia del

El siguiente diagrama ilustra las capas descritas:



- Diagrama Capas Lógicas -

Los Subsistemas y componentes que aparecen en el diagrama de capas lógicas serán detallados en la siguiente sección.

4.1. Descomposición en Subsistemas

En esta sección se presenta la descomposición del Sistema en Subsistemas, las interfaces y dependencias que existen entre los mismos, y la ubicación de cada subsistema en las capas de la Arquitectura mencionadas en la sección anterior se describirán en detalle en la sección 5 (Diagrama de Componentes)

4.1.1. Subsistema: WEB-APP

Este subsistema corresponde a la aplicación web que será utilizada por los usuarios finales del Sistema.

Realidad de Ambiente Ejecución:

Los ambientes de ejecución del sistema final presentan ciertas características que no se pueden dejar pasar por alto, las terminales de las farmacias por lo general cumplen con las siguientes características:

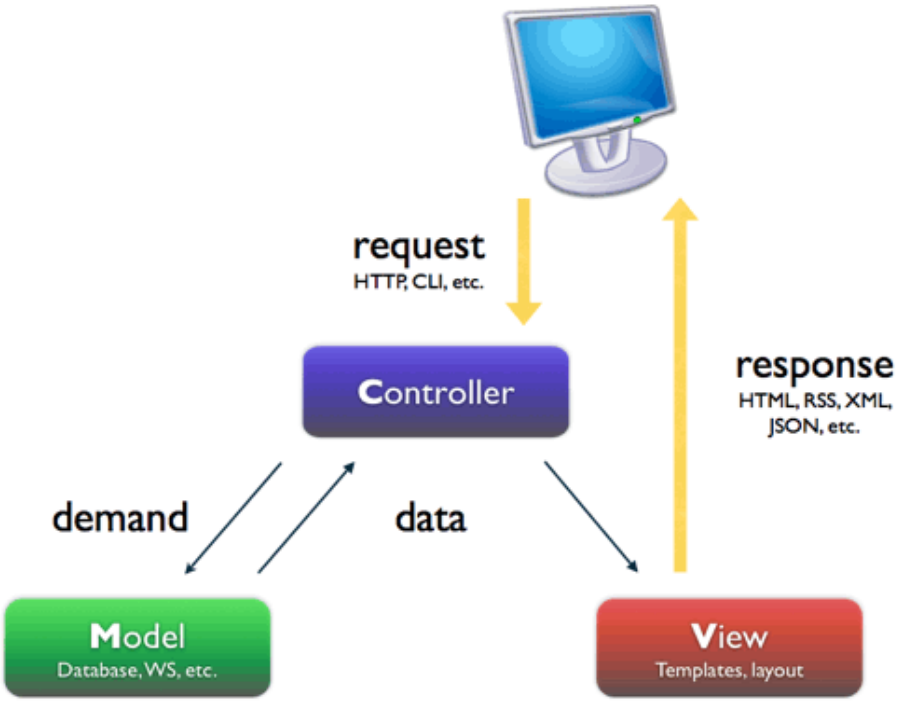
- Conexión lenta a internet.
- Versiones antiguas de navegadores (Internet Explorer en su mayoría, Firefox, y se estima una minoría en Chrome)
- Versiones antiguas de Sistemas Operativos (se estima en su mayoría Windows XP)
- El usuario cuenta con poco conocimiento informático

Entonces, considerando esta realidad, la aplicación web no solo debe ser fácil de usar, sino que debe contar con una buena performance, estimando que será el caso de uso Buscar Producto el que más usabilidad tendrá.
Bajo estas consideraciones es que se torna necesario contar con un diseño propio para este subsistema, que garantice poco procesamiento del lado del cliente (navegadores) donde la tasa de conexión no influya fuertemente en la performance del tráfico de datos, en particular al buscar productos.

Diseño Propio:

Teniendo en cuenta la realidad de los futuros ambientes de ejecución se decidió utilizar el patrón MVC (Model View Controller) para diseñar la aplicación web e integrarla con la lógica de negocio.

Este patrón se ajusta con las necesidades relevadas, se plantea desarrollar el subsistema en 3 componentes:



- Diagrama Arquitectura de patrón MVC -

La Vista (View)

Será el componente donde se implementarán las páginas, las pantallas que el usuario verá en el sitio, aquellas que desplegarán la información.

El Controlador (Controller)

Corresponde a la lógica intermedia entre la Vista y el Modelo, es el componente que tomando la entrada de la Vista mapea las solicitudes a métodos de la lógica y los invoca, notifica al modelo sobre cambios y espera la respuesta del mismo para retornar una respuesta a la Vista.

El Modelo (Model)

Corresponde a la lógica de negocio de la aplicación, pero mucho más que eso podríamos considerar la base de datos, la capa de comunicación. El modelo responde a las solicitudes de datos del Controller, las procesa, realiza los cambios debidos y responde al Controller para que la Vista pueda ser actualizada.

Virtudes del MVC:

- Resuelve el problema de modularizar funcionalmente la interfaz de usuario, permitiendo modificar fácilmente cada parte.
- Permite modificar la interfaz de usuario más a menudo que la lógica de negocio, actualizar la interfaz consiste en modificar la vista y mapear en el controlador las nuevas invocaciones de métodos.
- Permite mostrar la misma información de varias maneras diferente, de esta forma se pueden definir vistas para cada tipo de cliente

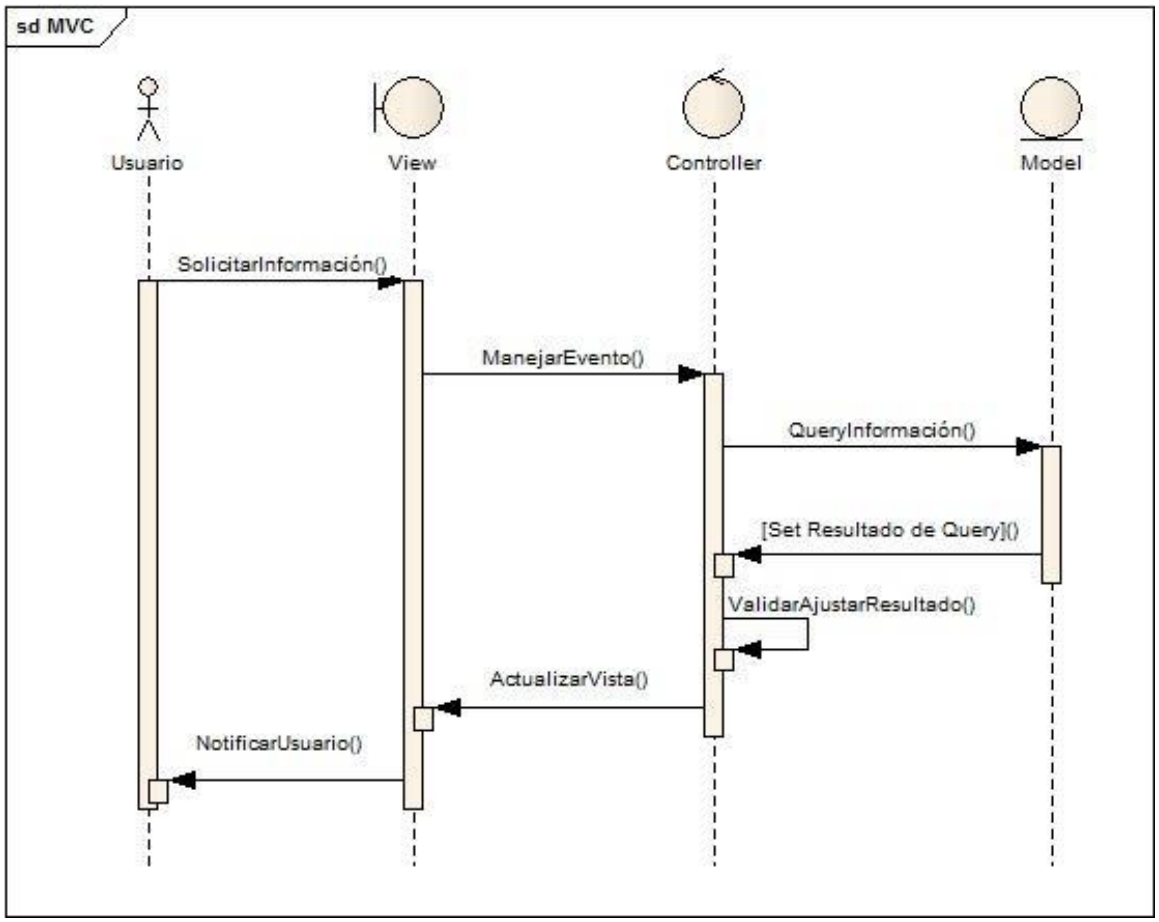
(aquellos que tengan mejores recursos de hardware/software en sus terminales, y aquellos que siguen apegados a los sistemas tradicionales)

- La interfaz de usuario generalmente consisten en presentar y actualizar información, para la presentación se envía información al modelo por medio de controlador y se presenta en la vista, para la actualización se delega la responsabilidad al modelo, el controlador notificará a la vista qué mostrar cuando ya esté actualizado, de esta forma la vista sólo se encargará de la presentación.

- Dado que la interfaz de usuario es más dependiente del dispositivo sobre el cual se corre, con MVC es fácil extender la vista a distintos tipos de dispositivos (como móviles) el Controlador sigue siendo el mismo para los distintos dispositivos.

La principal virtud es la i**ndependencia de la vista con el modelo**, nos aporta flexibilidad y brinda un abanico de posibilidades de interfaz de usuario acordes a los distintos dispositivos (pc, celular, tablet) que puedan correr en el futuro la aplicación, no solamente mediante navegadores web.

En el Diagrama a continuación se describe un ejemplo de invocación de un método del Usuario sobre un diseño MVC.



- Diagrama Secuencia Ejemplo MVC -

4.1.2. Subsistema LogicServer

Corresponde al subsistema de lógica de negocio, engloba los componentes necesarios para resolver los casos de uso identificados en el Modelo de Casos de Uso.

Requisitos No funcionales deseables:

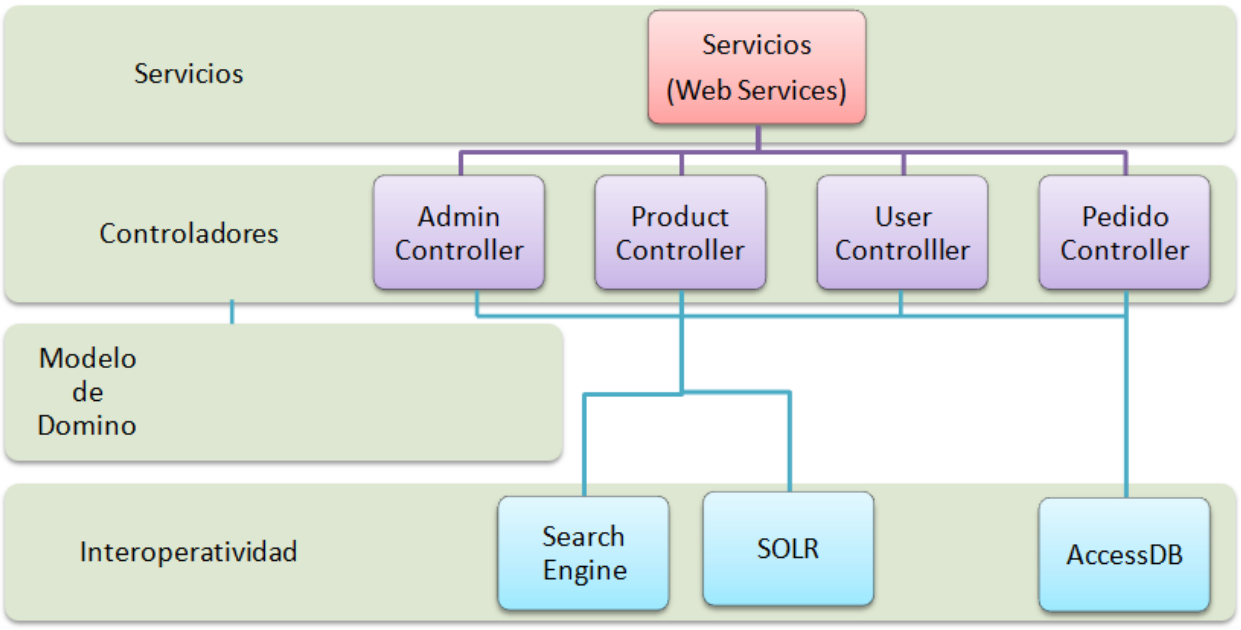
Dentro de los requerimientos del sistema se identificaron mediante las reuniones con el Cliente varios del estilo no-funcional que condicionan las decisiones de diseño de la aplicación, entre ellos podemos mencionar:

- Bajo costo de mantenimiento.
- Reusabilidad de código.
- Fácil integración con otros sistemas.
- Una Arquitectura extensible, que sirva como base para desarrollar nuevos módulos.

Considerando estos requerimientos es que se optó que este subsistema sea orientado a proveer y consumir servicios.

Diseño Propio:

El siguiente diagrama ilustra el diseño que se utilizó para ubicar las principales componentes del LogicServer



- Diagrama Diseño LogicServer -

El Servidor de la lógica de negocio se compone de un módulo de Servicios, mediante el cual publica web services (REST) con los métodos asociados a los Controladores que resuelven los casos de uso.

Los controladores dependerán del modelo de dominio para la implementación de las funcionalidades.

Los módulos de interoperabilidad corresponden a los utilizados para la Búsqueda de Productos (SearchEngine & SOLR) y para el acceso a la base de datos del LogicServer (AccessDB)

Los módulos para la Búsqueda de producto son los que engloban la lógica que resuelve la indexación de información a buscar en la aplicación web.

Ventajas de este diseño:

- Bajo costo de mantenimiento, en el sentido que agregar o quitar una funcionalidad nueva implica modificar el módulo donde esa funcionalidad estará (o se quitará) y dar de alta (o de baja) el web service que expone esa funcionalidad.

- Reusabilidad de código, si en un futuro quisiera cambiar la forma de acceder a la base de datos, o hubiera una migración de la base de datos alcanza con modificar el módulo AccessDB, para el resto de los componentes el cambio debiera ser transparente pues la comunicación a la base es únicamente por medio de AccessDB

- Reusabilidad de código, si en un futuro quisiera cambiar el Buscador de Productos, optimizarlo, o modificar las indexaciones, el cambio es a nivel de los componentes SearchEngine & SOLR, para el resto de los componentes el cambio debiera ser transparente.

- La integración con otros sistemas se podría dar por medio del módulo de Servicios, donde se consumirían los servicios de Sistemas Externos y por donde los mismos pueden consumir servicios con LogicServer, nada impide que la forma de comunicación pudiera ser por medio de base de datos, alcanza con modificar el módulo de Servicios, adaptarlo a los estándares de comunicación que desee el cliente (web services, base de datos, etc.)

- Este diseño muestra flexibilidad al querer extender la Arquitectura, si interesare agregar nuevos módulos, nuevas funcionalidades, podrían colocarse a nivel de los Controladores e ir publicando en las capas siguientes los métodos que se quieran utilizar, ya sea para el Controller de la WEB-APP como para módulo Servicios si lo fuera a consumir un Sistema Externo (como de Facturación por ejemplo)

En la sección 5 se detallará más a nivel de componentes este subsistema.

4.1.3. Subsistema SOLR

No se adentrará en detalles sobre este subsistema para esta iteración, en la próxima entrega se profundizará sobre el diseño. Consiste en un subsistema que está enfocado a indexar información y realizar búsquedas performantes sobre la información a la que accede. Presenta características muy buenas dada la realidad de los ambientes de ejecución finales para el sistema, pero serán detalladas con detenimiento en una próxima entrega de este documento.

4.2. Diseño de Clases

Exponer en este documento un diagrama de clases de diseño (DCD) no aportaría más de lo que se puede ver en el documento Modelo de Diseño, para ver en detalle

las clases del sistema, las dependencias entre ellas se sugiere revisar el documento de Modelo de Diseño.

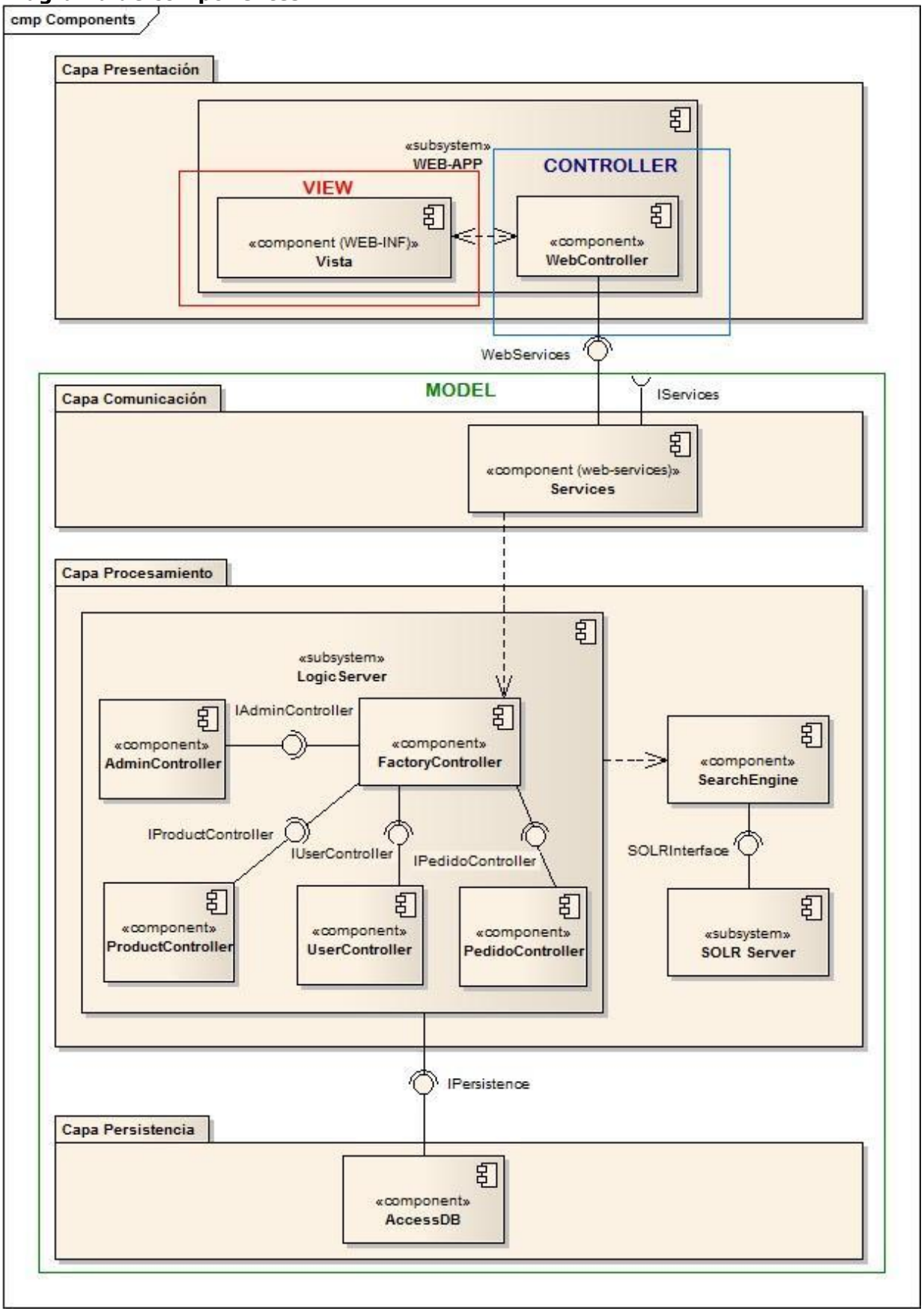
4.3. Diseño de Casos de Uso

Según el plan de desarrollo se realizaron los diseño de los casos de uso que deben comenzar a implementarse en la primer semana de Fase de Elaboración. El diseño de estos casos de uso y los siguientes se remitirán al documento de Modelo de Diseño.

5. Vista del Modelo de Implementación

En esta sección se describe la descomposición del sistema en componentes de software y las interfaces que permiten la comunicación entre las mismas. A continuación se presenta el Diagrama de Componentes, en las subsecciones siguientes se describen brevemente cada módulo, subsistema que aparece en el Diagrama.

Diagrama de componentes



- Diagrama Componentes Preliminar -

5.1. SubSistemas

Se describen brevemente los subsistemas reflejados en el diagrama de componentes.

5.1.1. LogicServer

Es el subsistema que comprende los módulos que implementan las principales funcionalidades que debe cumplir el producto final. Contará con un diseño lógico propio orientado a patrones de diseño (como ser Observer, State y otros) que sirvan para distribuir los módulos y clases de una forma eficiente, con bajo acoplamiento y alta cohesión para un ágil mantenimiento.

5.1.2. WEB-APP

Es el subsistema fron-end que actuará de presentación hacia el usuario final, tanto Web como Administrador, proveerá las interfaces gráficas, captará la información de entrada al Subsistema Servidor "Pedidos Online - DUSA" y reflejará los resultados de las funcionalidades implementadas por los módulos.

5.1.3. SearchEngine

Este componente refiere a un motor de búsqueda sobre los productos que aparecen en la base de datos de D.U.S.A., dado que actualmente existen 14000 entradas en la tabla Stock de productos, este módulo deberá ser diseñado e implementado de forma tal que las búsquedas realizadas por los clientes desde el sitio sean ágiles e intuitivas, para ello se utilizarán algoritmos de indexación que serán abordados en la próxima fase. Por el momento no se ahondará en mayores especificaciones.

5.2. Componentes

Se describen brevemente los módulos reflejados en los subsistemas que se están en el diagrama de componentes, se definen módulos por concepto funcionales, donde cada módulo implementa las funcionalidades que refieran al concepto que representan.

5.2.1. User Controller

Módulo que encapsula las funcionalidades ofrecidas a un Usuario a través del sitio web En esta primera instancia están consideradas las funcionalidades de: ver detalle de cuenta, iniciar y finalizar sesión, modificar contraseña

5.2.2. Pedidos Controller

Este módulo concentra la lógica necesaria para implementar las funcionalidades que refieren a los pedidos, iniciar un pedido, cancelarlo, modificar información de un pedido.

5.2.3. Product Controller

Este módulo encapsula funcionalidades básicas sobre los productos respecto de los pedidos como agregar, quitar productos a un pedido, consultar detalle de un producto, buscar un producto en el sitio (sea para consultar detalle como para agregarlo a un pedido activo). Si en una próxima instancia el cliente tuviera interés de agregar funcionalidades que atañen a este concepto deberían desarrollarse en este módulo.

5.2.4. Admin Controller

Este módulo engloba las funcionalidades que serán provistas para el administrador de la aplicación, en un principio contempla habilitar/deshabilitar de Usuario-Web, configuración de notificaciones a usuarios. La razón principal de encapsular estas funciones en un módulo propio para el administrador residen en brindar una ágil integración de nuevas funcionalidades que apliquen a este actor, como ser: obtener estadísticas sobre pedidos, sobre productos y otras que pudieran incorporarse en el futuro.

5.2.5. AccessDB

Este módulo implementa la lógica necesaria para acceder a la Base de Datos (las tablas propias del Sistema "Pedidos Online - DUSA" y las tablas propias de la Base Datos actual de D.U.S.A.) implementa los métodos de lectura y escritura de la base y transforma la información obtenida desde la Base en el tipo de objeto que espera procesar el módulo que invoca las primitivas de acceso a la Base.

5.2.6. Services

Es el módulo encargado de brindar acceso entre el Sistema-PODUSA y los Sistemas Externos (que podrían ser propios de D.U.S.A. o no, como por ejemplo: Sistemas de Facturación de Farmacias)

Para que el Sistema-PODUSA pueda acceder a otros Sistemas Externos este módulo requiere la interfaz "InterOperacion SistDUSA" para consumir los servicios de los Sistemas Externos.

Este módulo bien podría considerarse como una API, dependiendo de las funcionalidades que se le quieran delegar, como encriptar datos, transformar la información recibida a otros formatos, realizar backups a la Base de Datos, puede llegar a ser tan complejo como se quiera, tener un diseño lógico propio dependiendo de las restricciones que se apliquen. Remitiéndose a la realidad planteada para este proyecto se podría considerar implementarlo como una API simplemente, sin necesidad de agregar mayor complejidad.

5.2.7. FactoryController

Es el módulo encargado de instanciar los controladores que el módulo de Services necesite para publicar los métodos asociados a ese controlador. Este módulo instancia los controladores por medio de Interfaces "*IcontrollerName"* para reutilizar código al acceder a los controladores, sin importarle a Services qué clase, qué módulo implementa la interfaz por la que accede a los métodos a ser publicados.

5.3. Interfases

Se describen brevemente las interfaces entre los componentes y subsistemas.

5.3.1. WebServices

Es la interfaz donde se exponen los web services que consumirá la Aplicación Web para comunicarse con el Servidor de "Pedidos Online - DUSA".

Esta interfaz también podría utilizarse para que los Sistemas Externos puedan interactuar con el Sistema de "Pedidos Online - DUSA".

La imposición de una capa de comunicación permite que las modificaciones que se tengan que hacer para las interacciones con los SistemasExternos, se realicen a

nivel del módulo Servicios, siendo transparente para el resto del Sistema-PODUSA la forma en que la conexión con otros Sistemas se establece (sea por web-services, acceso a base de datos u otros mecanismos de conexión)

5.3.2. SOLRInterface

Es la interfaz que comunica el servidor SOLR con el componente SearchEngine, para obtener el resultado de las query enviadas hacia el catálogo de productos, para el caso de uso Buscar Producto.

5.3.3. IPersistence

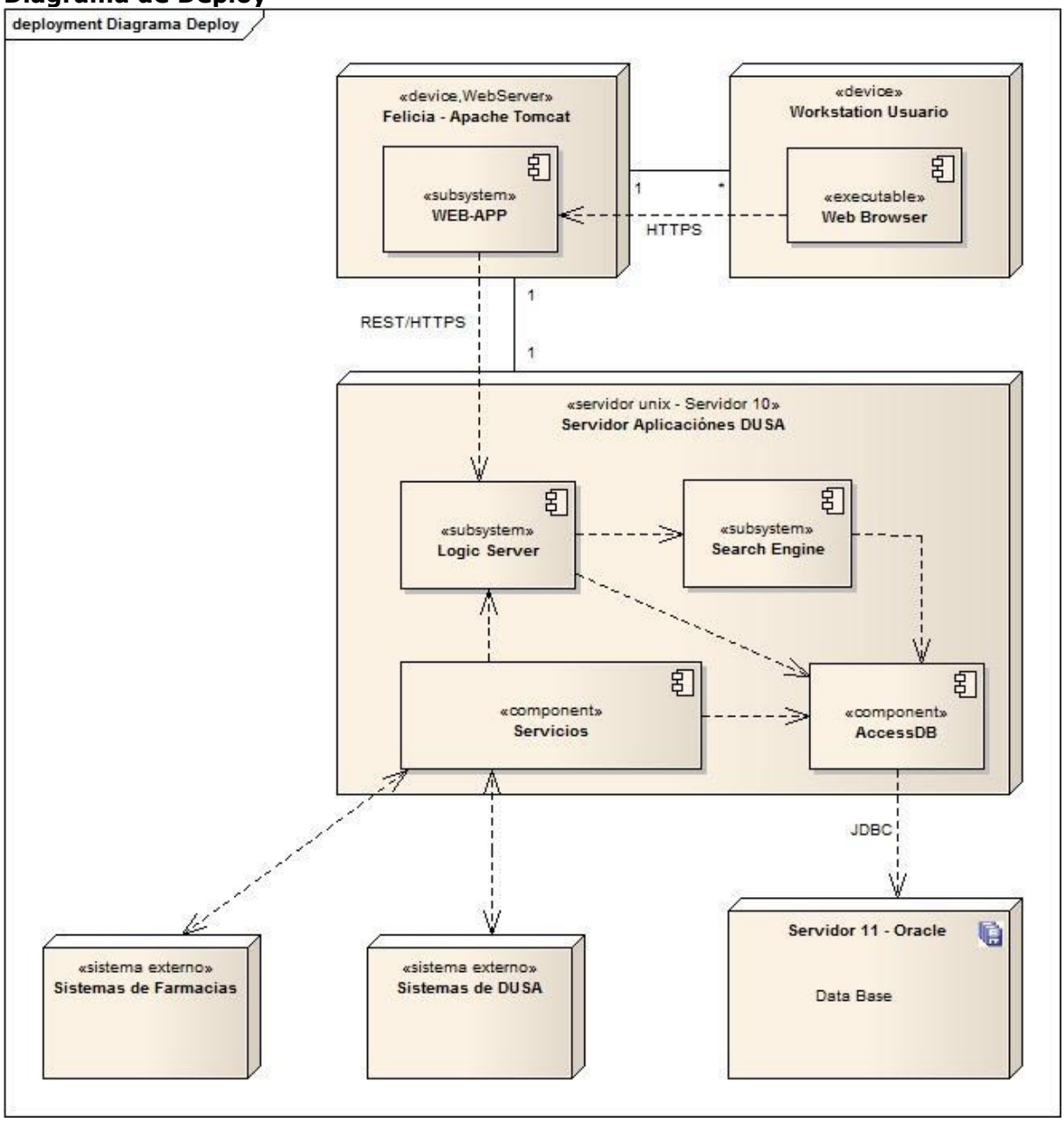
Es la interfaz provista por el módulo de Acceso a Base de Datos para interactuar con el DBMS de D.U.S.A..

6. Vista del Modelo de Distribución

En esta sección se presenta distribución física de los subsistemas presentados en las secciones anteriores, se describen las conexiones entre los distintos nodos involucrados.

6.1. Diagramas de Distribución

Diagrama de Deploy



6.2. Nodos

En esta sección se da más detalle de los nodos relevantes a la Arquitectura del sistema.

6.2.1. Servidor Felicia

Corresponde al Web Server donde estará deployada la WEB-APP aplicación web, es un servidor Apache Tomcat v6.

6.2.2. Servidor 10

Corresponde al servidor producción donde están instalados los sistemas de D.U.S.A., donde se instalará el Sistema "Pedidos-Online - DUSA".

6.2.3. Servidor 11

Corresponde al Servidor que aloja las Bases de Datos de D.U.S.A..

6.2.4. Sistemas Externos (Sistema Farmacias & Sistemas de D.U.S.A.)

Estos nodos corresponden a los sistemas externos al Sistema-PODUSA que podrían realizar intercambio de información por medio de servicios a través del componente Servicios.

6.3. Conexiones

En esta sección se describen las conexiones entre los nodos, protocolos de mensajería, de aplicación.

6.3.1. WebServer Felicia - Servidor 10

La conexión de estos nodos para la solución del Sistema-PODUSA será mediante web services utilizando el estándar REST mediante HTTPS.

6.3.2. Servidor 10 - Servidor 11 (Base Datos)

La conexión entre estos nodos para la solución del Sistema-PODUSA será mediante la API JDBC.

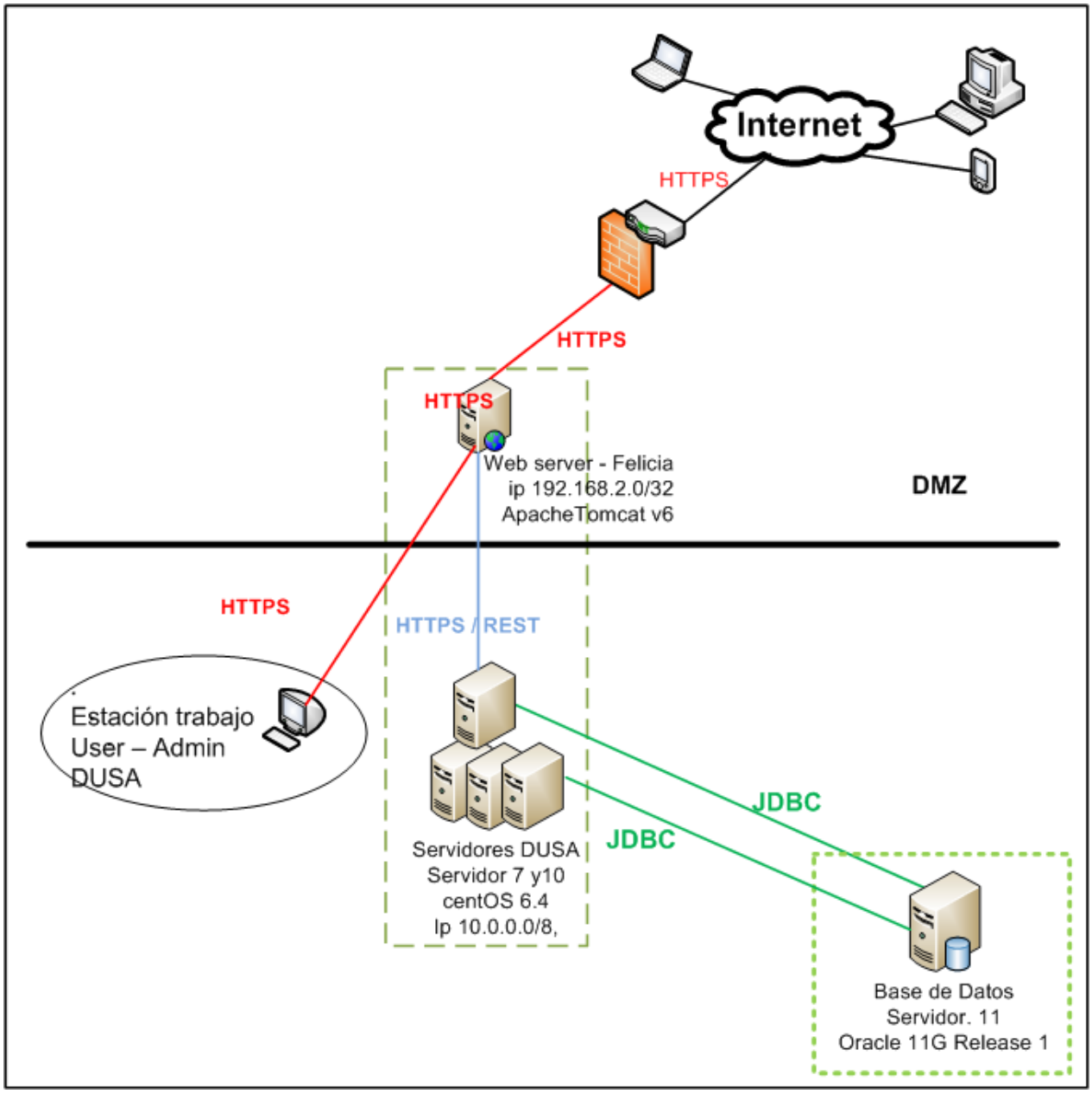
6.3.3. Servidor 10 - Nodos Externos (Sistemas Externos)

La conexión entre el Servidor de "Pedidos Online - DUSA" y aquellos Sistemas Externos que pudieran ser integrados en un futuro se realizará mediante el componente Servicios, donde se fijarán estándares de comunicación, como ser protocolo de transporte, de mensajería. Bien podría implementarse mediante accesos a Base de Datos, o por medio de web-services, según lo prefiera el Cliente.

La principal idea es que al centralizar estas conexiones en un módulo específico es que para el resto del Sistema-PODUSA la forma de conexión sea transparente y no implique cambio en la lógica de negocio de los restantes módulos.

6.3.4. Diagrama de Red

Para fijar más las ideas aquí expresadas se agrega un Diagrama de Red que se actualizará en cada entrega de este documento.


Internet
HTTPS
HTTPS
HTTPS
Web server - Felicia
ip 192.168.2.0/32
ApacheTomcat v6
DMZ
HTTPS
HTTPS / REST
Estación trabajo
User – Admin
DUSA
JDBC
JDBC
Servidores DUSA
Servidor 7 y10
centOS 6.4
Ip 10.0.0.0/8,
Base de Datos
Servidor. 11
Oracle 11G Release 1