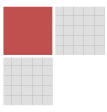
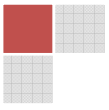


Índice

Descripción del Negocio 3
Contexto 3
Alcance 3
Glosario 5
Arquitectura propuesta 7
Manejo de sesiones 7
Implementación de persistencia y transaccionalidad 8
Manejo de concurrencia 8
Manejo de excepciones 8
Vista de Casos de Uso 9
Vista Lógica 10
Diagramas de clases 10
Vista de Componentes 15
Vista Física o de Despliegue 16
Vista de Procesos 17



Descripción del Negocio

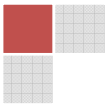
Contexto

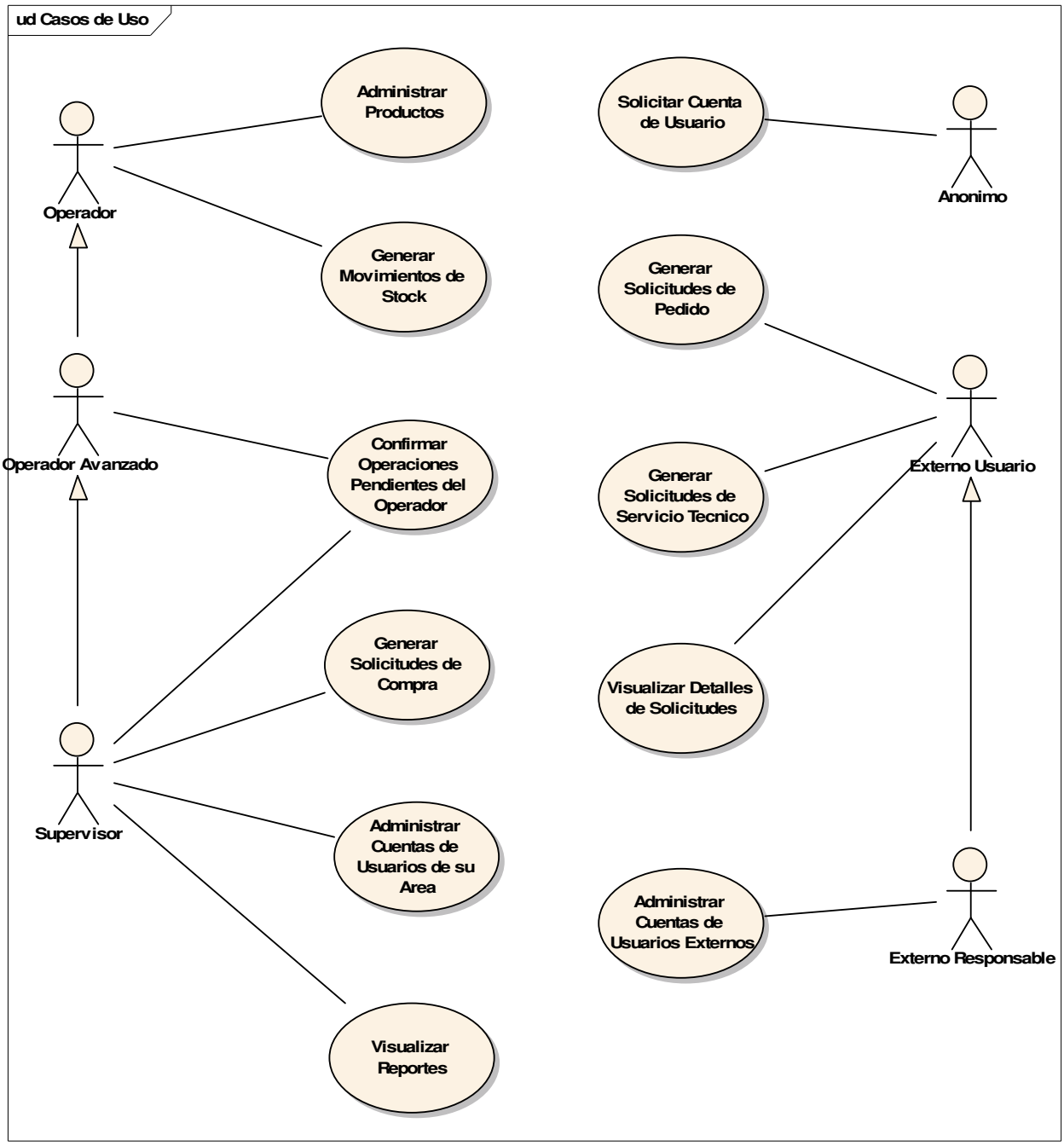
El proyecto tiene por objetivo desarrollar una herramienta y/o conjunto de herramientas informáticas que sirvan para dar soporte al proceso de manejo de inventario y stock.

Alcance

A continuación se listan los requerimientos basados en el EGR:

- Administración de los distintos usuarios del sistema
- Administración de productos e insumos
- Administración de proveedores
- Administración de áreas, puertas y depósitos
- Movimientos de stock
- Ingresos y egresos de stock
- Solicitud de Servicio Técnico
- Solicitud de Pedido de Productos
- Solicitud de Compra



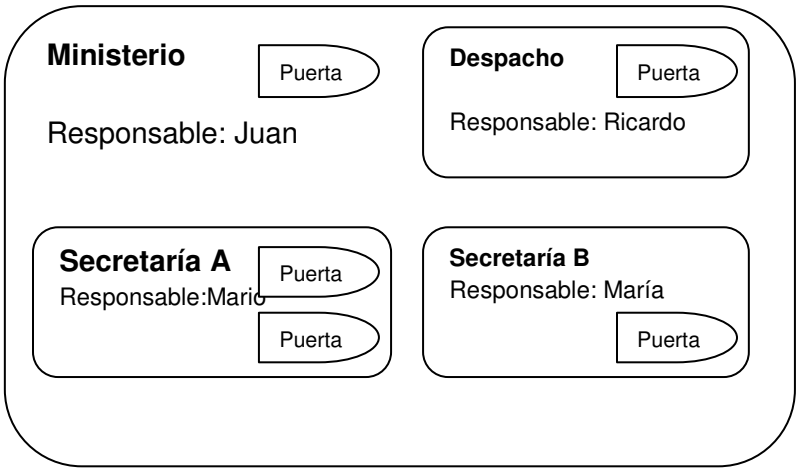

ud Casos de Uso
Administrar Productos
Generar Movimientos de Stock
Confirmar Operaciones Pendientes del Operador
Generar Solicitudes de Compra
Administrar Cuentas de Usuarios de su Area
Visualizar Reportes
Solicitar Cuenta de Usuario
Generar Solicitudes de Pedido
Generar Solicitudes de Servicio Tecnico
Visualizar Detalles de Solicitudes
Administrar Cuentas de Usuarios Externos
Operador
Operador Avanzado
Supervisor
Anonimo
Externo Usuario
Externo Responsable

Glosario

Áreas

El ministerio, las secretarías, las direcciones, los despachos, etc, son tipos de estructura lógicas, que llamaremos áreas, que tienen un único responsable (Firmante). Dichas áreas lógicas tienen puertas físicas asignadas. Un área lógica, por ejemplo un ministerio puede contener otras áreas lógicas como por ejemplo secretarías, direcciones, etc. Cada una siempre posee un responsable y puede estar compuesta por otras áreas lógicas que poseen puertas.

Ejemplo:



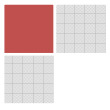
Puertas

Las puertas son ubicaciones físicas, que pertenecen a alguna área lógica. Tienen un responsable asignado. Los productos se encuentran asignados a las puertas, por lo tanto tienen un responsable asignado, que es el responsable del área donde esta la puerta.

Depósitos

Los depósitos son un caso especial de las áreas lógicas. También poseen puertas físicas. Los proveedores también pueden tener depósitos y puertas asignados, para registrar cuando un producto vuelve al proveedor para ser reparado por ejemplo. Tales depósitos son indicados como EXTERNOS.

Proveedores



Personas o empresas que venden o entregan productos.

Productos

Los productos / insumos representan todo artículo en los depósitos o áreas. Están agrupados por categorías. Todo producto tiene asociada una puerta en la que está físicamente, también puede estar asociado a otro producto, si es que forma parte del mismo.

Garantías

Cada producto puede tener asociada una garantía, la misma contiene la información de la garantía del proveedor sobre el mismo.

Nivel de Stock Crítico

Cada genérico de producto cuenta con un nivel de stock mínimo aconsejable. Por debajo del mismo se recomienda realizar pedidos de compra.

Movimiento de Stock

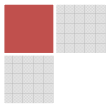
Movimiento de producto realizado entre dos puertas de algún área.

Ingreso de Stock

Al ingresar mercadería a algún depósito de stock.

Baja de Stock

Egreso de mercadería del stock.



Arquitectura propuesta

El sistema a implementar para responder a los requerimientos funcionales definidos puede ser descripto en función de las siguientes necesidades:

- Ser accesible a múltiples usuarios.
- Permitir que los usuarios puedan interactuar con el sistema cumpliendo diferentes roles y garantizar que ciertas funcionalidades estén disponibles sólo para algunos roles.
- Realizar validaciones sobre las operaciones que los usuarios intentan llevar a cabo, de manera de garantizar el cumplimiento de las reglas de negocio y la consistencia de la información almacenada.
- Permitir operaciones tanto de actualización como de consulta, de lo cual se deriva que se debe almacenar el resultado de las operaciones efectuadas en algún medio persistente.

Se decidió utilizar un patrón de arquitectura LAYERS, en el cual se definieron los siguientes *layers* (a partir de ahora llamados capas):

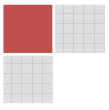
- **Capa de Persistencia:** Ofrece servicios de persistencia y recuperación de información a la capa de negocio.
- **Capa de Negocio:** Implementa procesos de negocio identificados durante el análisis funcional
- **Capa de Servicio:** Expone las funcionalidades necesarias para soportar los casos de uso definidos para la aplicación. Los mismos serán consumidos por la capa de presentación.
- **Capa de Presentación:** Presenta la información al usuario permitiéndole interactuar con la aplicación, y traduce los eventos generados en pedidos a la capa de servicio.

Se tomó la determinación de aislar lo más posible a la capa de Negocio de las demás. Son las otras capas las que dependen de la capa de negocios. La capa de negocios sólo conoce a ciertas interfaces que sirven de comunicación entre los componentes, pero no conoce a sus implementaciones específicas.

Las relaciones que se establecen entre las capas están descriptas en la vista de paquetes.

Manejo de sesiones

Para el manejo de sesiones de usuario se utilizaron las herramientas provistas por el framework de .NET. Se decide utilizar sólo sesión en la capa de presentación, para permitir que las llamada a métodos de servicio, no dependa de datos de la sesión, sino que sea independiente una de otra.



Implementación de persistencia y transaccionalidad

La aplicación implementa persistencia y manejo de transacciones mediante el uso de NHibernate.
Ventajas de su utilización:

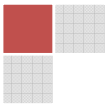
- Permite conexión a múltiples base de datos relacionales.
- Oculta la complejidad del acceso y manejo a la base de datos al programador.
- Maneja un pool de conexiones para permitir accesos concurrentes a la base de datos y controla que las transacciones no interfieran unas con otras.
- Provee manejo y control de concurrencia a través del manejo de versiones de los datos.

Manejo de concurrencia

Para el manejo de concurrencia también se utilizaron los servicios del framework de .NET.
.NET está basado en una API de C#, la cual propone un modelo básico para el procesamiento de HTTPRequest y Response preparado para soportar concurrencia.

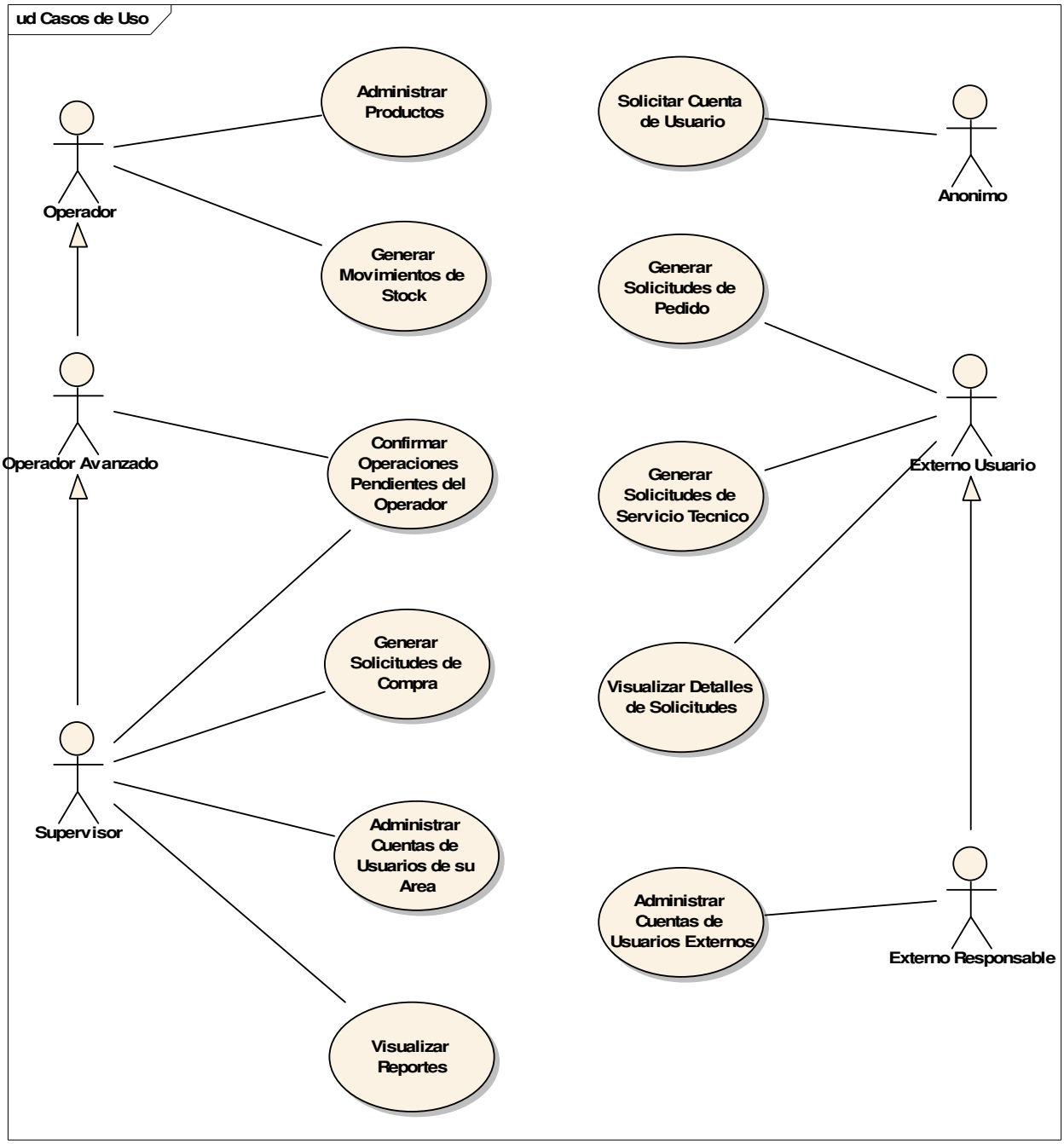
Manejo de excepciones

Las excepciones se utilizan como mecanismo de comunicación de errores entre las diferentes capas de la arquitectura.
Se maneja una jerarquía de excepciones basadas en dos excepciones base: BusinessLogicException y ApplicationException, la primer jerarquía especializa las excepciones de reglas de negocio, por lo que las mismas si llegan a la capa de presentación son visualizadas al usuario para que tome una medida al respecto. Mientras que la segunda jerarquía modela las excepciones de aplicación, como una conexión caída, la base de datos no disponiblem, etc, que inhabilitan continuar con la operación al usuario, están son visualizadas de forma genérica al operador para que notifique al administrador del sistema del mismo.



A continuación se incluyen las diferentes vistas a través de las cuales se describe la arquitectura propuesta para la presente aplicación.

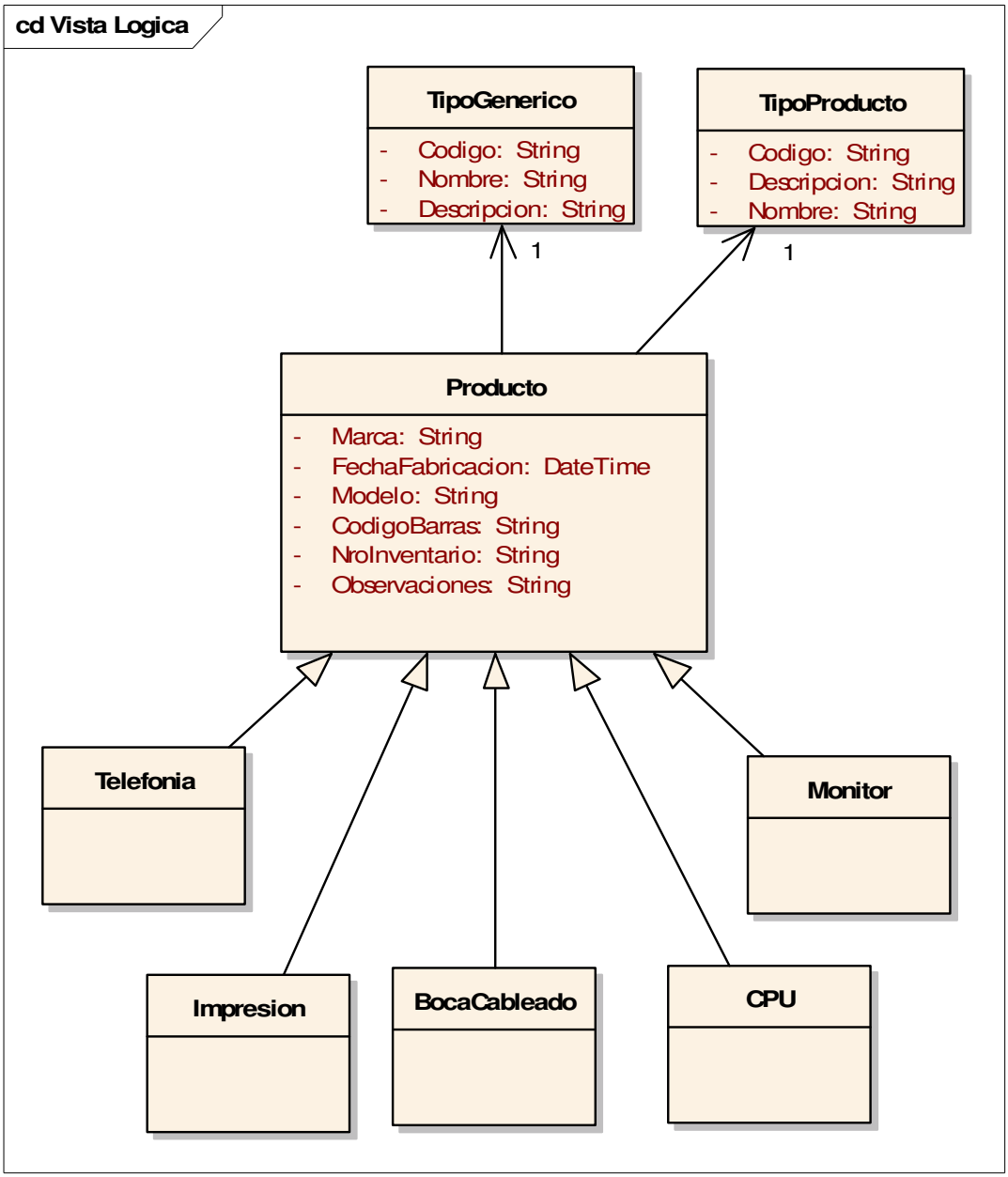
Vista de Casos de Uso

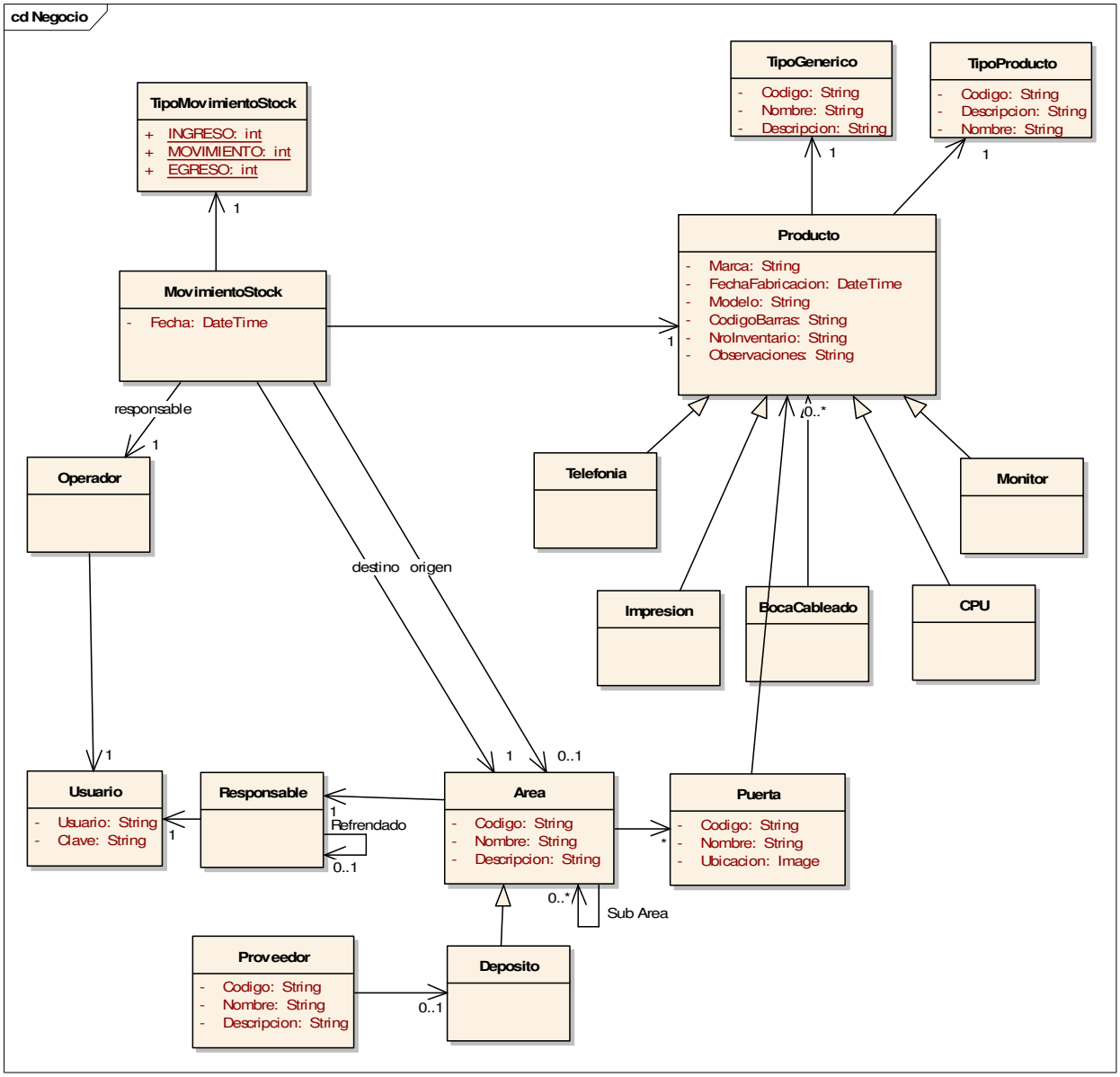


Vista Lógica

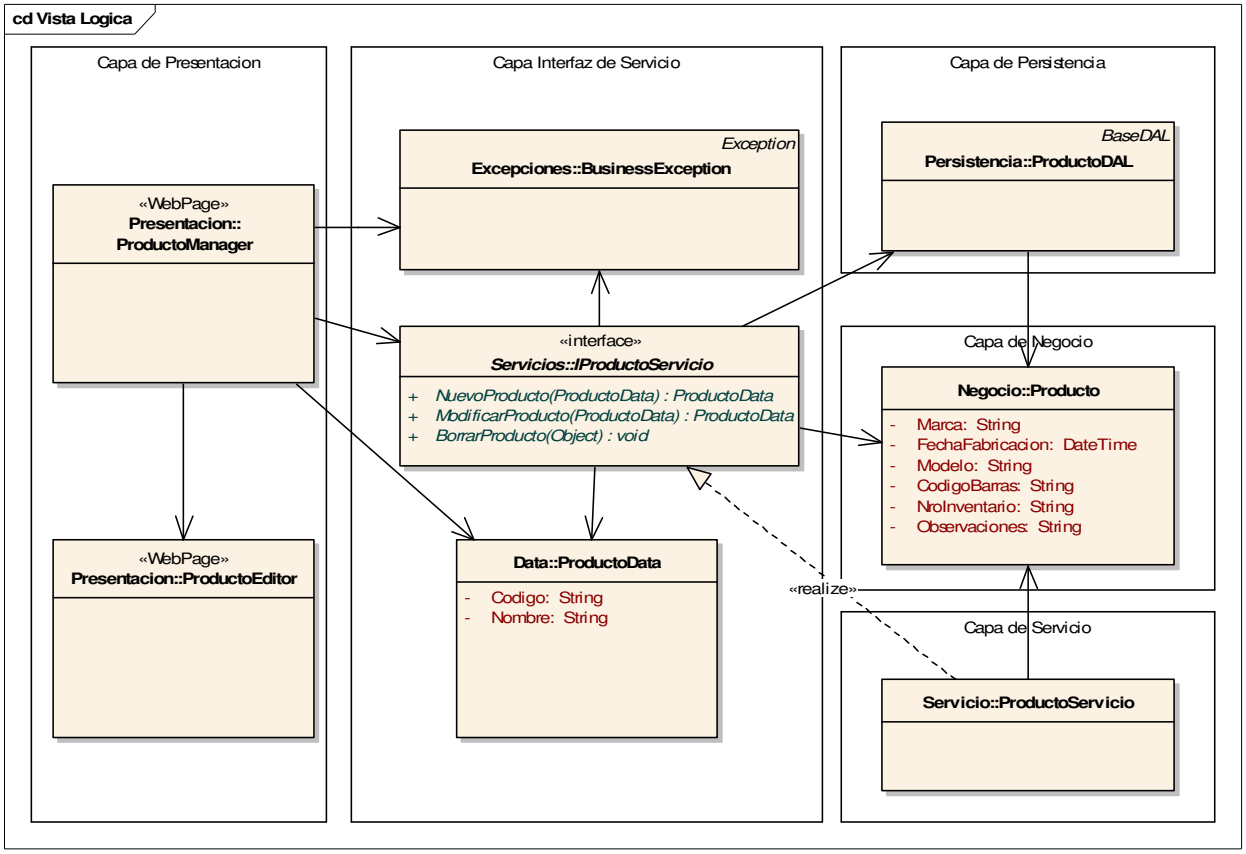
Diagramas de clases

En estos siguientes diagramas se muestran las clases principales que integran cada capa de la arquitectura propuesta. Dichas clases irán ajustándose a medida de avanzar con la aplicación, las mismas están definidas de modo básico para poder definir la arquitectura a utilizar. Se entrará en mayor detalle de las mismas al momento de realizar el diseño mas fino del modelo.



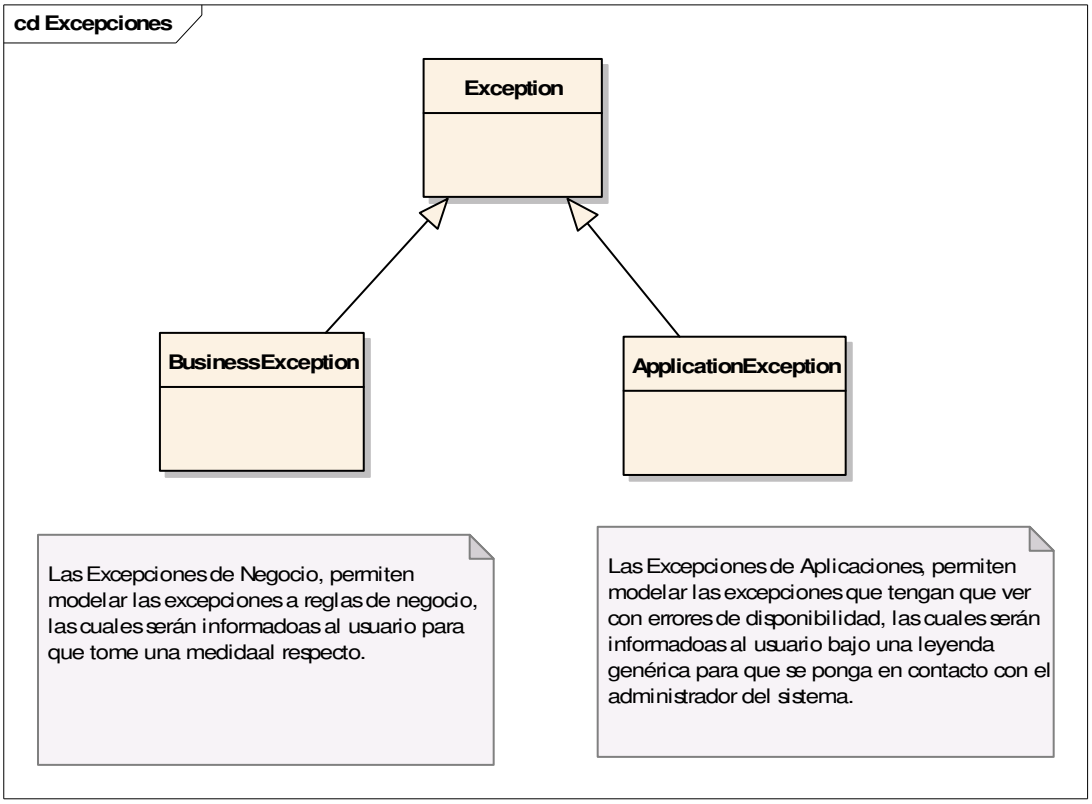

cd Negocio
TipoGenerico
- Codigo: String
- Nombre: String
- Descripcion: String
TipoProducto
- Codigo: String
- Descripcion: String
- Nombre: String
TipoMovimientoStock
+ INGRESO: int
+ MOVIMIENTO: int
+ EGRESO: int
Producto
- Marca: String
- FechaFabricacion: DateTime
- Modelo: String
- CodigoBarras: String
- NroInventario: String
- Observaciones: String
MovimientoStock
- Fecha: DateTime
responsable
Operador
destino
origen
Telefonia
Monitor
Impresion
BocaCableado
CPU
Usuario
- Usuario: String
- Clave: String
Responsable
Refrendado
Area
- Codigo: String
- Nombre: String
- Descripcion: String
Puerta
- Codigo: String
- Nombre: String
- Ubicacion: Image
Sub Area
Proveedor
- Codigo: String
- Nombre: String
- Descripcion: String
Deposito

A continuación se muestran las relaciones entre clases de distintos paquetes

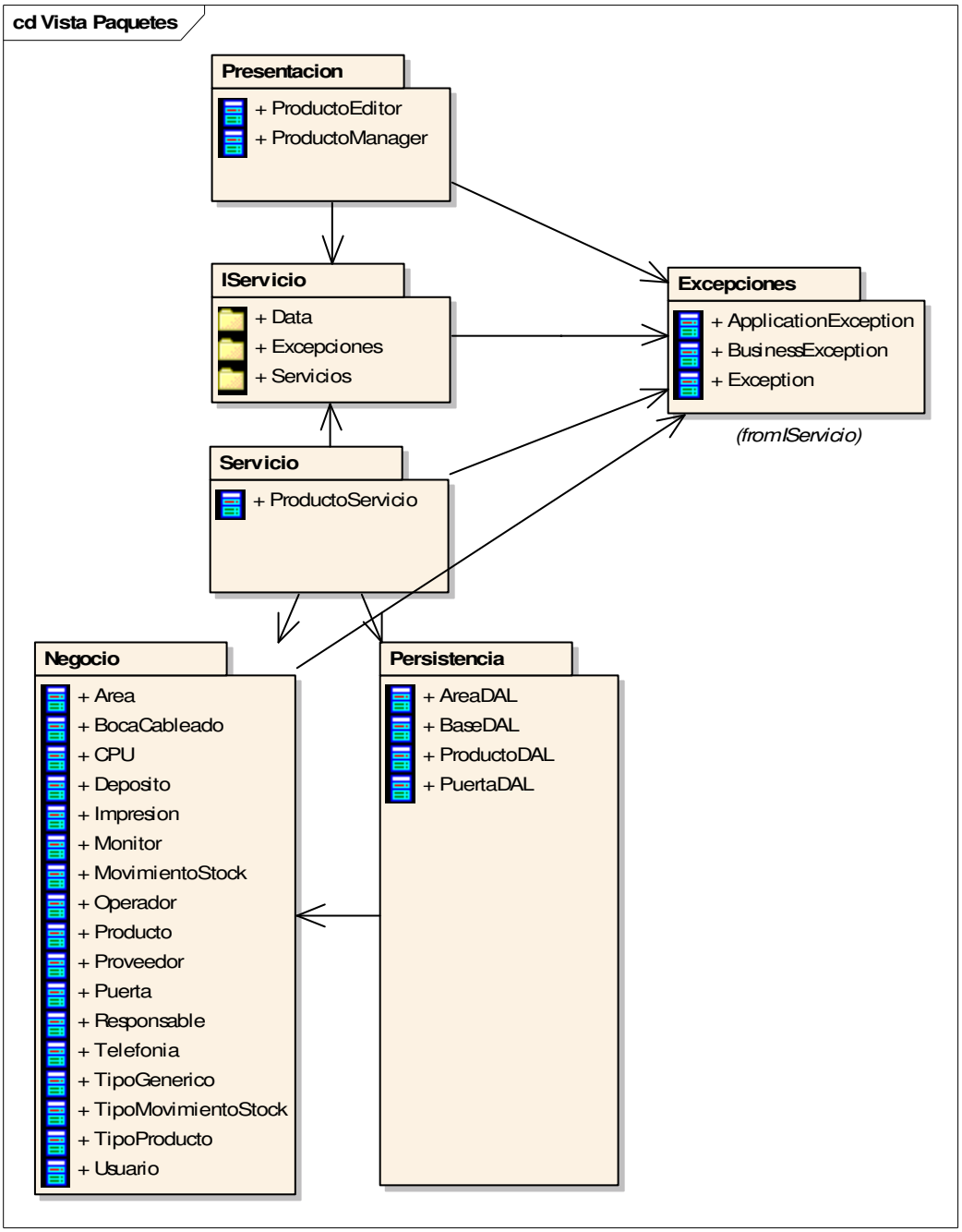


Jerarquía de Excepciones

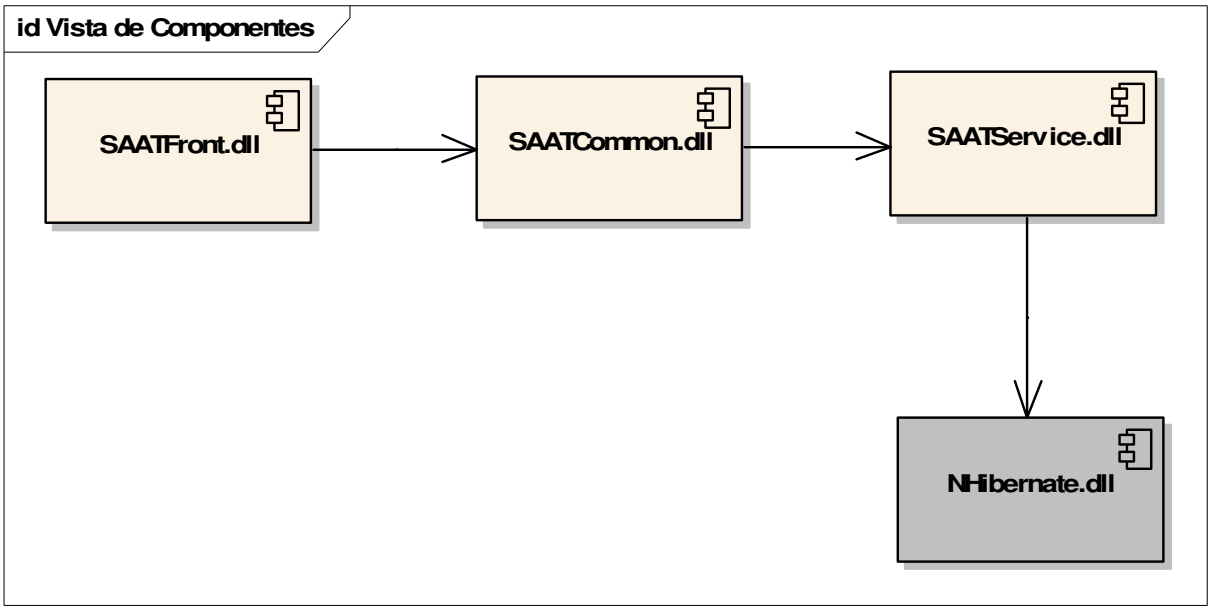
En el siguiente gráfico se ven las dos Excepciones base de donde heredarán las excepciones de negocio y de aplicación necesarias.



Como puede observarse, los paquetes de presentación y persistencia son dependientes del paquete de Negocios, con lo cual un cambio en la capa de presentación o en la forma de almacenar la información no afecta a la capa de negocios. A su vez los paquetes intercambian información por medio de objetos simples. Estos últimos están incluidos dentro del paquete Interfaz de Servicio y son utilizados por todas las capas.



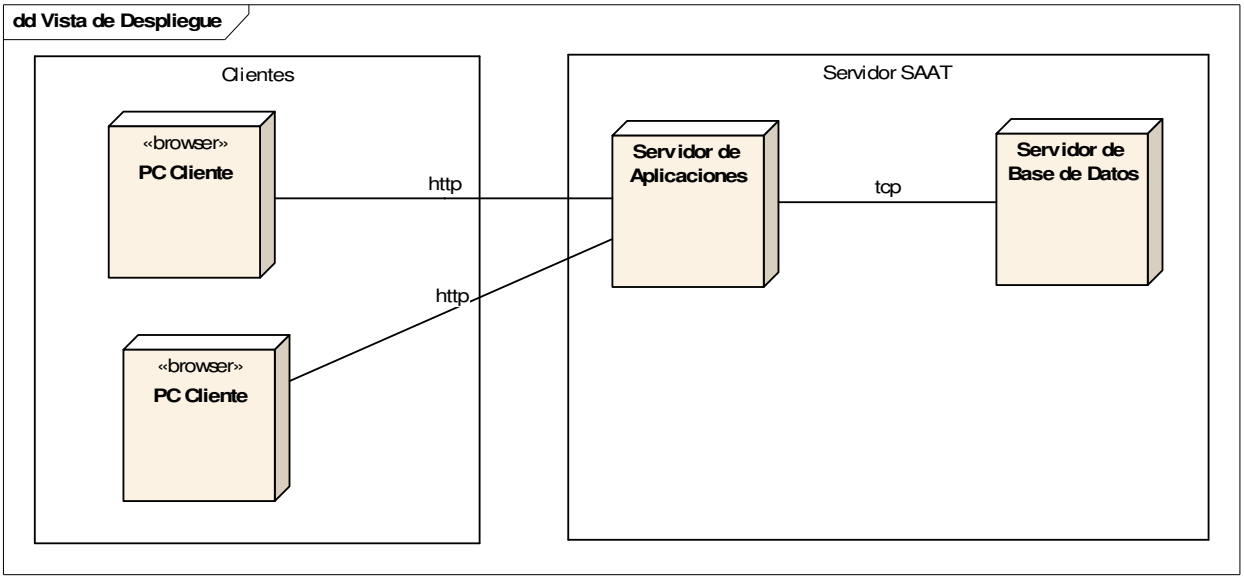
Vista de Componentes



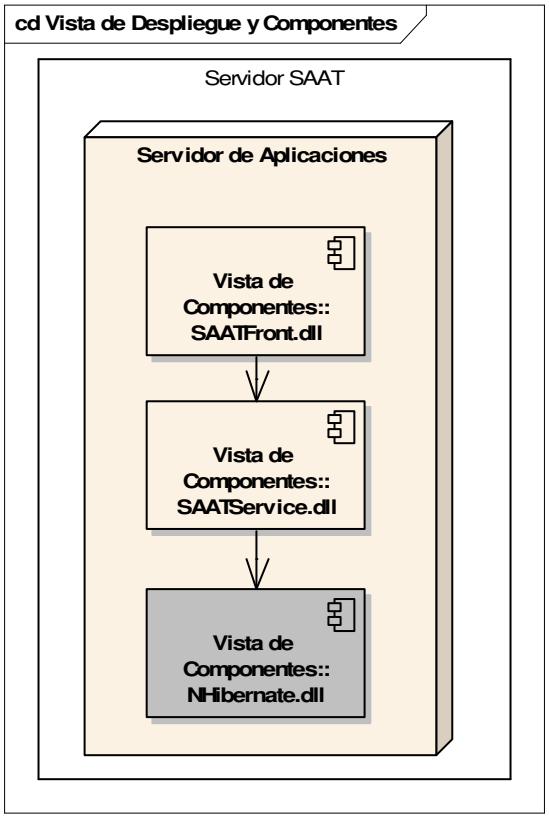
Los componentes principales son:

- **SAATFront:** Agrupa las clases encargadas de mostrar la información y permitir interactuar al usuario con la aplicación. Define las pantallas a mostrar según el estado de ejecución, controla la navegabilidad del sistema. Formatea los datos de salida. Valida el formato de los datos de entrada para permitir que el usuario pueda enterarse de las validaciones antes de que su solicitud llegue a la capa de lógica de negocios.
- **SAATService:** Agrupa las clases que definen las reglas de negocio. Publica los servicios de negocio y controla su acceso. Incluye las clases de manejo de persistencia.

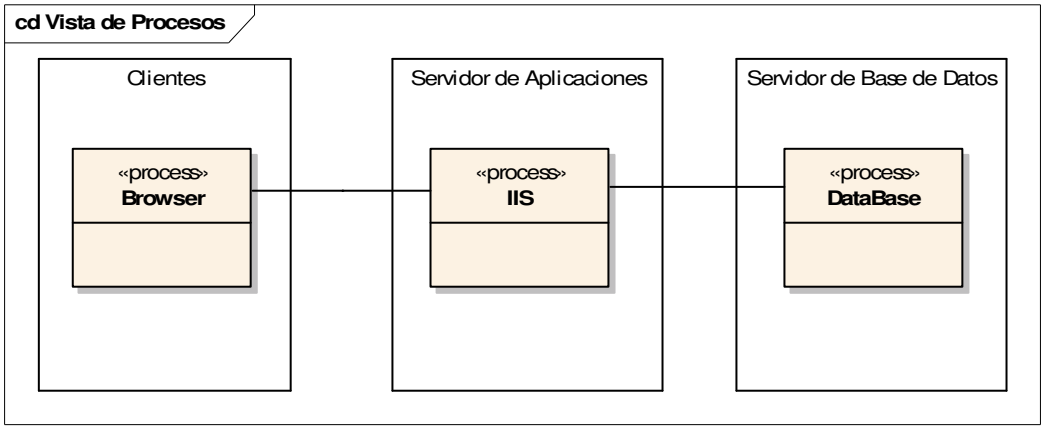
Vista Física o de Despliegue



A continuación se muestra como se despliegan los componentes físicos dentro de los servidores analizados en la vista de depsliegue.



Vista de Procesos



En el diagrama se pueden observar los procesos que son necesarios para el funcionamiento de la aplicación SAAT.
Tenemos un primer proceso, llamado Browser, el cual representa al proceso de los browsers clientes que acceden a la aplicación, comunicándose con el proceso llamado IIS, el cual hostea todos los componentes de la aplicación SAAT. Existe otro proceso, llamado DataBase Process, que refleja el proceso del servidor de la base de datos relacional.