SensaCar

Especificación de Requisitos

UNIVERSIDAD
COMPLUTENSE
MADRID

Ignacio Baena Kuroda
Cristóbal Saraiba Torres
Ignacio Vítores Sancho

Contenido

1 - Introducción

1.1 Propósito

El objetivo de este documento de especificación de requisitos es determinar una descripción completa del sistema que vamos a implementar.

El documento consta de una clara definición de las funcionalidades del programa y de los casos de uso pedidos por el cliente.

Este documento está creado para confirmar que los requisitos especificados por el cliente coinciden con el programa a realizar.

1.2 Alcance

El software en específico se llama ***SensaCar.*** Dicho software controlará las ventas de dos tipos de vehículos, en concreto coches y motos, y mantendrá relación directa entre el vendedor y el cliente.

Por otra parte, la empresa tendrá un control de todos sus empleados gestionando sus salarios dependiendo del tipo de contratación que tenga y se distinguirán sus competencias individuales.

Esta aplicación deberá ser manejada por el personal de la empresa, siendo así un software de tipo empresarial y no de uso cotidiano. Un beneficio directo será un control exacto del cálculo de salarios dependiendo del tipo de empleado.

1.3 Definiciones acrónimos y abreviaturas

- **JDBC** - Java Database Connectivity, es una API que permite la ejecución de operaciones sobre bases de datos desde el lenguaje de programación Java.
- **DAO** - Un componente de software que suministra una interfaz común entre la aplicación y uno o más dispositivos de almacenamiento de datos, tales como una Base de datos o un archivo
- **JPA** - Es un framework del lenguaje de programación Java que maneja datos relacionales en aplicaciones usando la Plataforma Java en sus ediciones Standard (Java SE) y Enterprise (Java EE).

- **MariaDB**- MariaDB es un sistema de gestión de bases de datos relacional.
- **DNI** -Documento Nacional de Identidad.
- **API** - La interfaz de programación de aplicaciones, conocida también por la sigla **API** del inglés application programming interface, es un conjunto de subrutinas, funciones y procedimientos

1.4 Referencias
A continuación, se proporcionarán una lista completa de todos los documentos referenciados en este SRS.
Las definiciones se han extraído de:
https://es.wikipedia.org/wiki/
Diapositivas del profesor.

1.5 Resumen
A continuación se muestra una descripción general de todo el desarrollo que se va a llevar a cabo en cuestión de requisitos software, se dictaminarán las funciones del producto, sus características y sus correspondientes restricciones, una serie de supuestos y dependencias que representan los factores que pueden afectar a dichos requisitos y finalmente en este apartado, unos requisitos futuros con vistas a versiones futuras de la aplicación.

Más adelante en el tercer apartado, se mostrarán un ejemplo de la ejecución del programa mostrando una maqueta de las interfaces de usuario, las funciones que se deben llevar a cabo en la validez del producto, una serie de requisitos de rendimiento, un **modelo de dominio,** las restricciones de diseño y los atributos del sistema software.

2 – Descripción General

2.1 – Perspectiva del producto
El producto ha sido ideado con la intención de ofrecer todas las funcionalidades necesarias para un concesionario de vehículos, unificando la gestión de las ventas y de los empleados bajo un mismo software dividido en dos subsistemas sin conexión entre ellos.

Este software está pensado para funcionar en un computador de oficina que disponga de la última versión de Java como una aplicación de escritorio.

2.1.1 Interfaces del sistema:
Se compondrá de 2 subsistemas sin ninguna conexión más allá del menú de inicio del programa. El sistema tendrá que manejar **6 entidades en total, 3 por subsistema:**

- El primer subsistema se encargará de la gestión de ventas, con 3 entidades a manejar (**Venta, Modelo, Cliente**) y DAOs para acceder a la base de datos.

- El segundo subsistema será el encargado de manejar los datos de los empleados, también con 3 entidades (**Empleado, Competencia, Departamento**) utilizando JPA para implementar el almacén del dominio.

2.1.2 Interfaces de usuario:
La interfaz de todo el producto será lo más intuitiva posible, anteponiendo la funcionalidad al diseño.

2.1.3 Interfaces hardware:
Solo será requerido teclado, ratón y un monitor para manejar el producto.

2.1.4 Interfaces software:
Será necesario: Windows 10, Java 2.8 o superior, MariaDB 10.3

2.1.5 Interfaces de comunicación:
El producto no se comunicará mediante servicio Web, pero deberá tener acceso a la base de datos.

2.1.6 Memoria:
No será necesario controlar ni el espacio en disco ni la memoria RAM, pero se limitará el consumo de la segunda, dentro de lo posible, evitando acceder a datos no solicitados.

2.1.7 Operaciones:
Es necesario que se disponga de operaciones CRUD (Create, Read, Update, Delete, Read All) de las 3 entidades de la parte DAO (**Cliente, Modelo y Ventas**). Además, los módulos Cliente y Modelo incluyen una query.

Respecto a la parte implementada con JPA, el módulo **Empleado** tendrá todas las operaciones CRUD mientras que **Competencia** tendrá las operaciones Create, Delete, Asignar competencia a empleado, Desasignar competencia a empleado, Actualizar competencia de empleado y Mostrar empleados de competencia y el módulo **Departamento** tiene las operaciones Create, Delete, Read y Calcular nómina.

El usuario deberá tener acceso a la aplicación en todo momento, de modo que cualquier acción contra la base de datos hará uso de *commit* y *rollback* para mantener la consistencia de los datos.

2.1.8 Requisitos de adaptación:
No existen requisitos en este sentido.

2.2 - Funciones del producto

Cliente

Alta cliente.

Leer cliente.

Leer lista de clientes.

Actualizar cliente.

Eliminar cliente.

Consultar clientes con N compras desde una fecha.

Venta

Abrir venta.

Añadir vehículo a venta.

Quitar vehículo de venta.

Cerrar venta.

Actualizar venta.

Eliminar venta.

Leer venta.

Leer lista de ventas.

Obtener Detalles Ventas.

Modelo

Crear modelo.

Leer modelo.

Actualizar modelo.

Borrar modelo.

Consultar lista de modelos.

Consultar modelos con N ventas desde una fecha.

Competencia

Añadir competencia.

Eliminar competencia.

Asignar competencia a un empleado.

Desasignar competencia a un empleado

Actualizar competencia de un empleado

Mostrar todos los empleados que tienen una competencia.

Departamento

Añadir departamento.

Eliminar departamento.

Leer departamento.

Calcular total de nóminas de departamento.

Empleado

Añadir empleado.

Eliminar empleado.

Leer empleado.

Leer lista de empleados.

Actualizar empleado.

2.3 Características del usuario

El usuario varía en función de la parte de la aplicación

Parte 1

Gerente: Usuario que podrá acceder a los datos de todos los vendedores. Contará con estudios en Administración y Gestión de Empresas. Con un nivel de experiencia avanzado.

Vendedor: Usuarios con experiencia en ventas y con conocimientos generales sobre vehículos automotrices sin un nivel de estudios específico.

Parte 2

Recursos Humanos: Serán los usuarios encargados de gestionar los datos de los empleados de la empresa. Estudios requeridos: Grado en Relaciones Laborales y Recursos Humanos. Con un nivel de experiencia básico a intermedio.

2.4 – Restricciones

- Interfaz para ser usada mediante Eclipse.
- El sistema se desarrolla bajo el paradigma cliente/servidor.
- Lenguaje de programación: el sistema se desarrollará en lenguaje Java, siendo la base de datos implementada con ayuda de JDBC y JPA.
- La aplicación debe mantener los datos de manera consistente en la base de datos en todo momento. No se permite introducir datos erróneos por parte del usuario.
- El sistema garantiza el tratamiento de concurrencia de manera correcta.
- El sistema se dividirá en dos subsistemas independientes: uno con la capa de recursos implementada mediante el patrón DAO y otro mediante JPA.
- El sistema tendrá una operación polimórfica en la parte implementada mediante JPA (cálculo de la nómina de un empleado).
- El sistema permitirá realizar consultas sobre la base de datos con más de un filtro en la parte que hace uso del patrón DAO.
- No se implementará ningún tipo de seguridad adicional sobre el uso de la aplicación más allá de las precondiciones de cada caso de uso.

2.5 Supuestos y dependencias

Para el funcionamiento completo del sistema, se requiere tener preinstalado el entorno de Java. Recomendado la versión Java 8. Se necesita tener instalado también el cliente de MariaDB.

2.6 Requisitos futuros

- Optimizar la aplicación para sistemas operativos macOs (Apple).
- Desarrollar la versión para dispositivos móviles.
- Servicios de Terminal

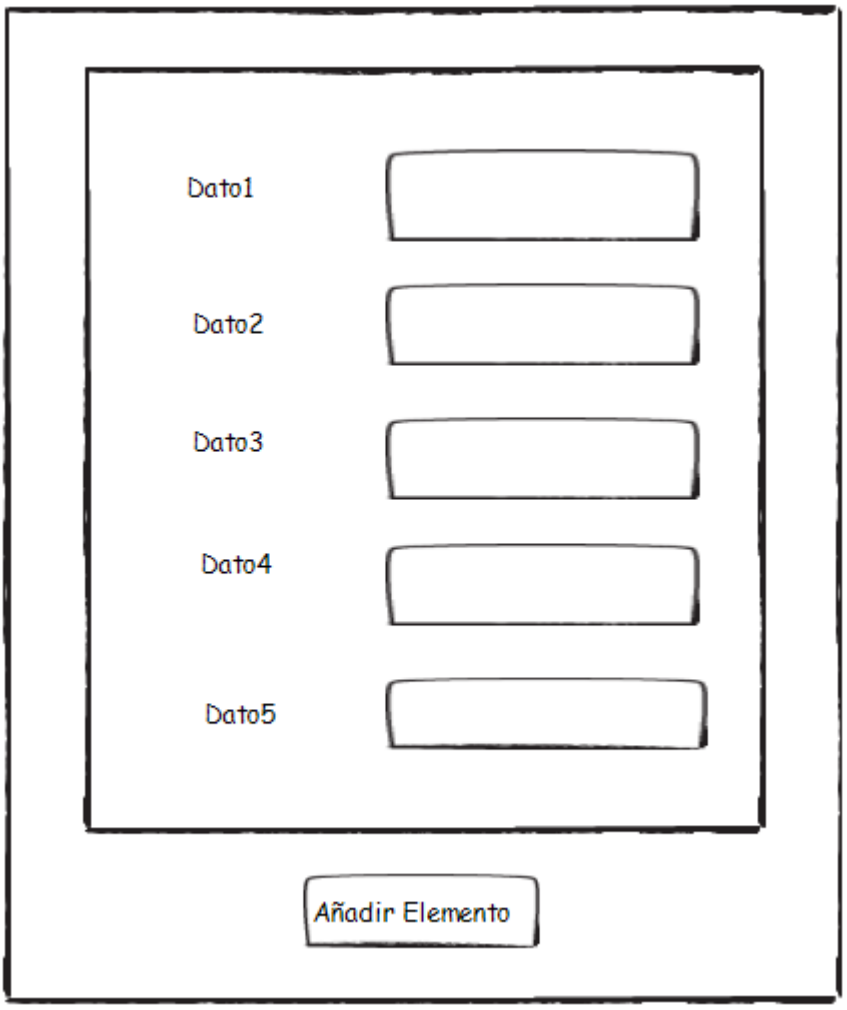
3 – Requisitos específicos

3.1 Interfaces externas

- **Nombre del elemento:** Create
- **Descripción del propósito:** Añadir nuevo elemento
- **Origen de entrada o destino de salida:**

  Entrada: Los datos del elemento.

  Salida: Mensaje de éxito o error.
- **Rango válido, precisión y/o tolerancia:**
  - DNI (8 números y 1 letra), teléfono (9 cifras), nombre (solo letras).
  - ID (10 cifras, positivo), precio (mayor que 0).
  - Stock (mayor o igual que 0).
  - Nivel (mayor que 0).
- **Organización/formatos de ventana:**



- **Formato de fechas:** dd/mm/aaaa
- **Formato de comandos:** No hay comandos va por interfaz gráfica

- **Mensajes de fin:**

  "Elemento añadido."

  "Error: elemento ya existe."

  "Error: datos inválidos."

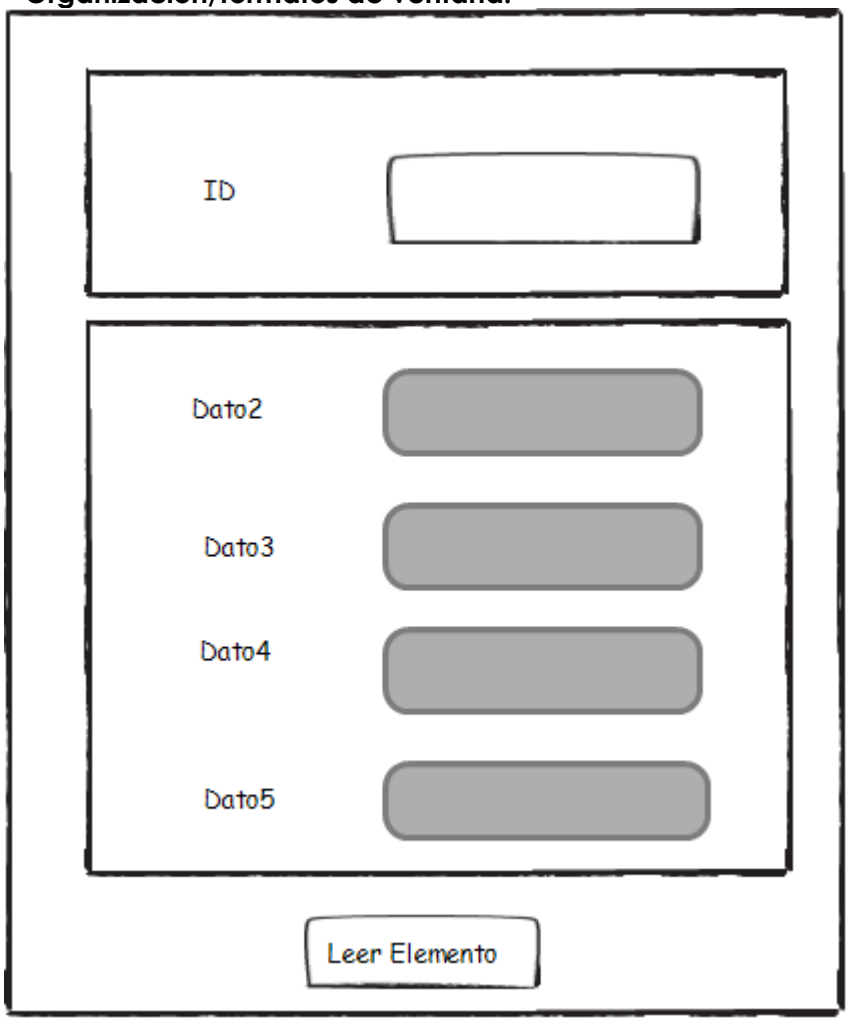
- **Nombre del elemento:** Read
- **Descripción del propósito:** Leer un elemento
- **Origen de entrada o destino de salida:**

  Entrada: El identificador del elemento.

  Salida: Información del elemento o mensaje de error.
- **Rango válido, precisión y/o tolerancia:**

  - DNI (8 números y 1 letra).

  - ID (10 cifras, positivo).
- **Organización/formatos de ventana:**



- **Formato de fechas: dd/mm/aaaa**
- **Formato de comandos:** No hay comandos va por interfaz gráfica

- Mensajes de fin:

"Información del elemento."

"Error: elemento no existe."

"Error: ID inválido."

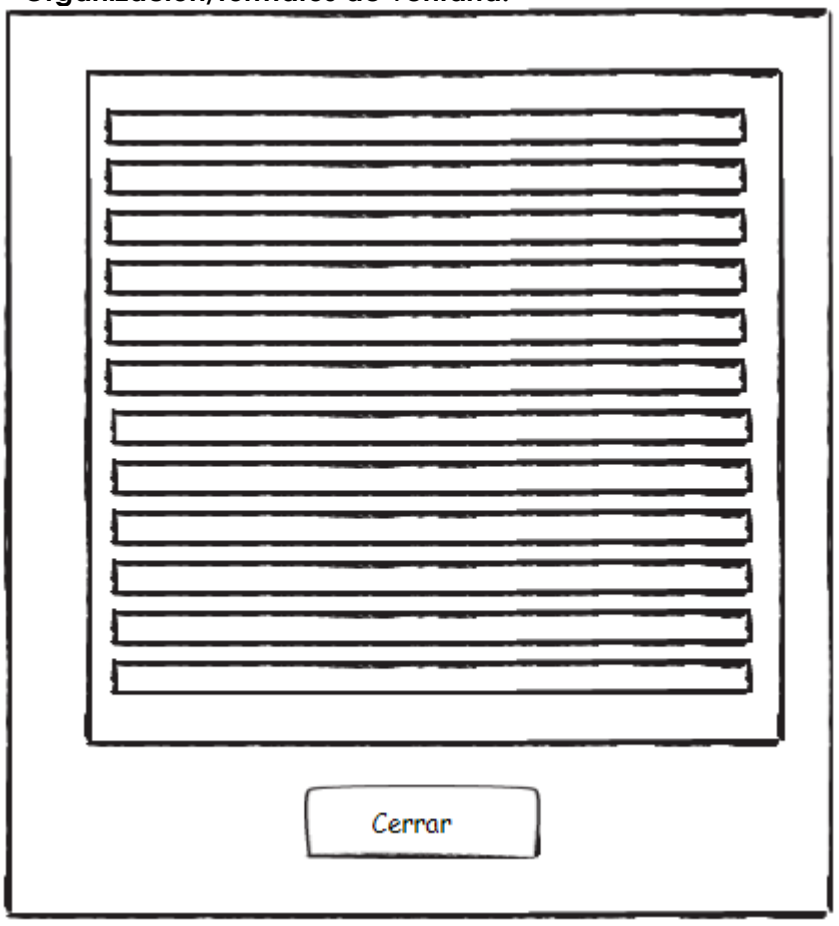
- Nombre del elemento. Read All.

- Descripción del propósito. Devuelve todos los elementos de la base de datos por pantalla.

- Origen de entrada o destino de salida. Salida de datos por pantalla.

- Rango válido, precisión y/o tolerancia.

- Organización/formatos de ventana.



- Formato de fechas. dd/mm/aaaa

- Formato de comandos. No cuenta con comandos, utiliza una interfaz gráfica

- Mensajes de fin.

"No existen elementos en la base de datos."

- Nombre del elemento. Update.

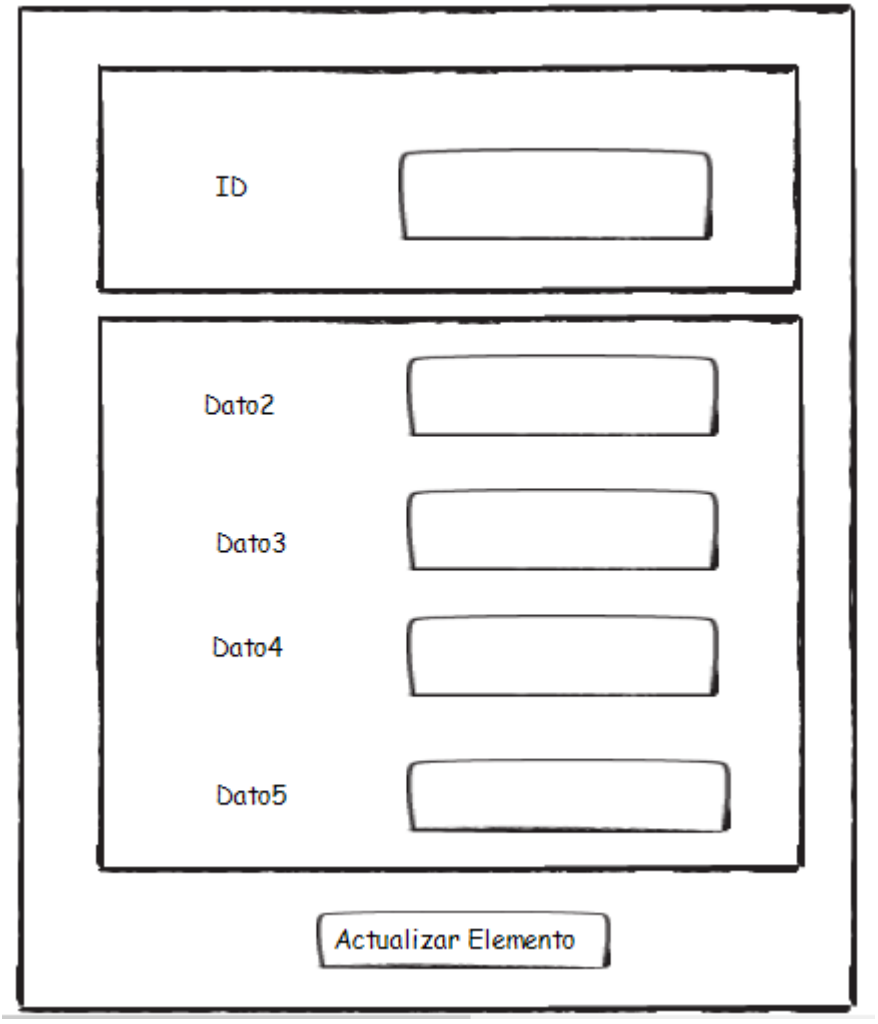
- Descripción del propósito. Actualiza los datos de un elemento de la base de datos.

- Origen de entrada o destino de salida. Entrada del elemento por teclado. Salida base de datos.

- Rango válido, precisión y/o tolerancia.

- DNI (8 números y 1 letra), teléfono (9 cifras), nombre (solo letras).
- ID (10 cifras, positivo), precio (mayor que 0).
- Stock (mayor o igual que 0).
- Nivel (mayor que 0).

- Organización/formatos de ventana.



- Formato de fechas. dd/mm/aaaa

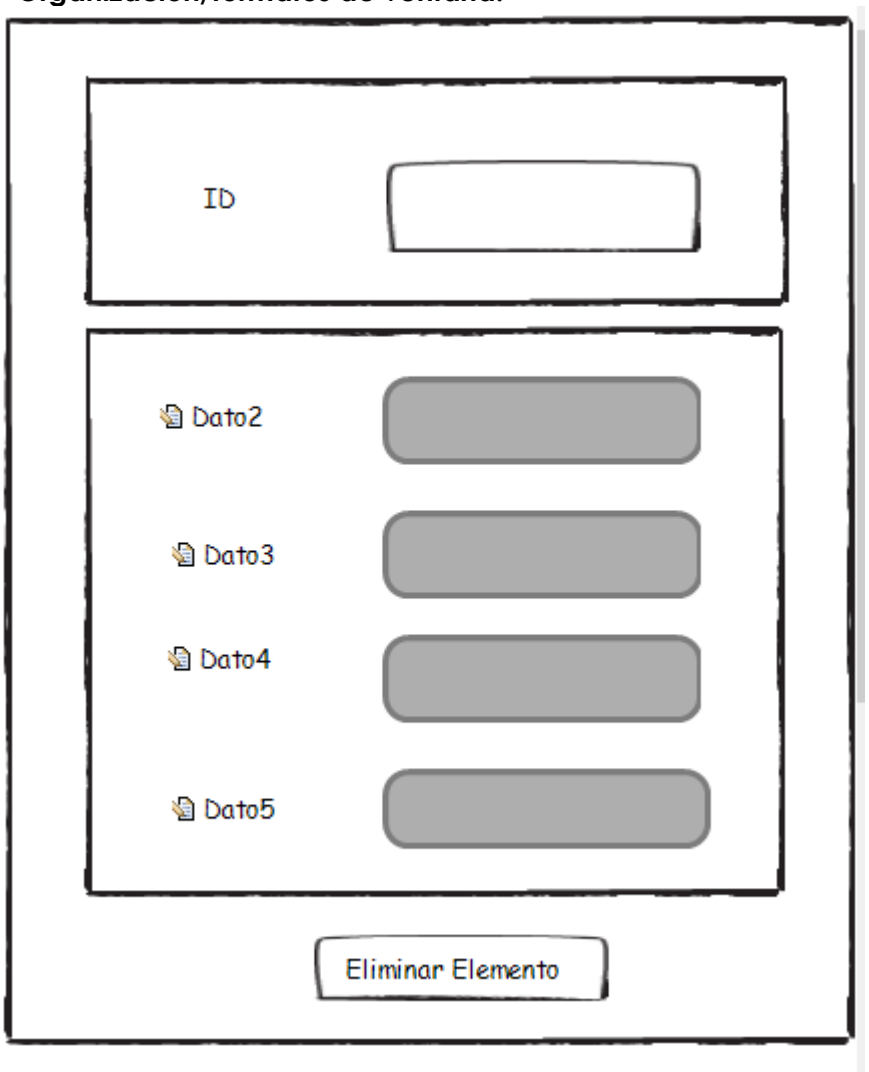
- Formato de comandos. No cuenta con comandos, utiliza una interfaz gráfica

- Mensajes de fin.

"Elemento actualizado."

"Error. Elemento no encontrado."

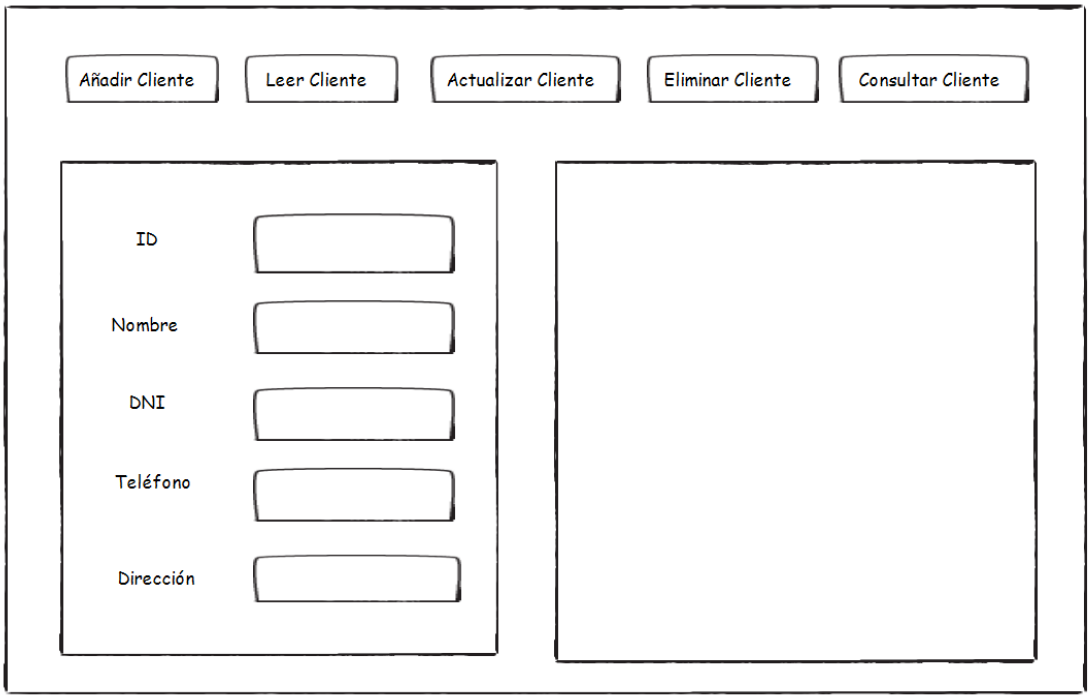
- **Nombre del elemento.** Delete.
- **Descripción del propósito.** Eliminar un elemento de la base de datos.
- **Origen de entrada o destino de salida.** Entrada del identificador del elemento por teclado. Salida base de datos.
- **Rango válido, precisión y/o tolerancia.**
    - DNI (8 números y 1 letra).
    - ID (10 cifras).
- **Organización/formatos de ventana.**



- **Formato de fechas.** dd/mm/aaaa
- **Formato de comandos.** No cuenta con comandos, utiliza una interfaz gráfica
- **Mensajes de fin.**
    "Elemento eliminado de la base de datos."
    "Error. Elemento no encontrado."

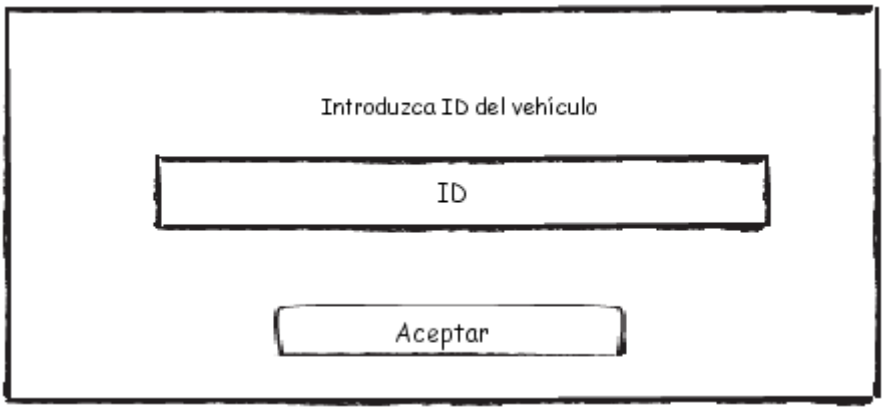
Pantallas específicas

Módulo cliente

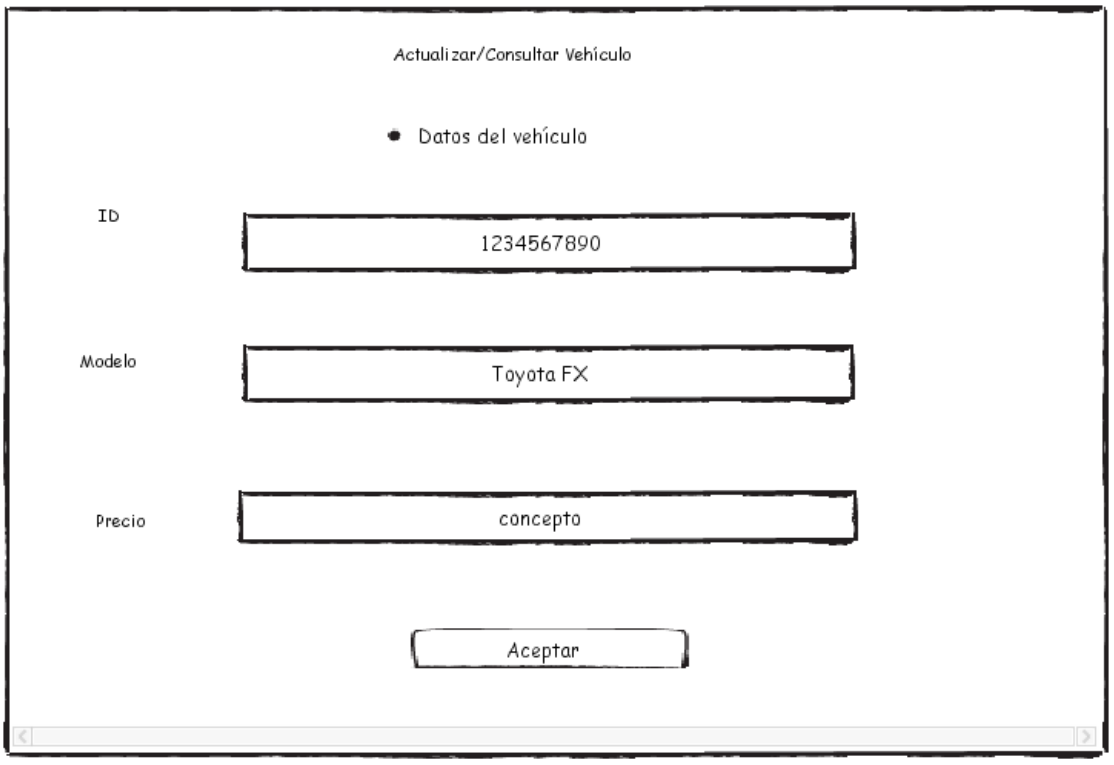


Modulo Modelo

Consultar Modelo

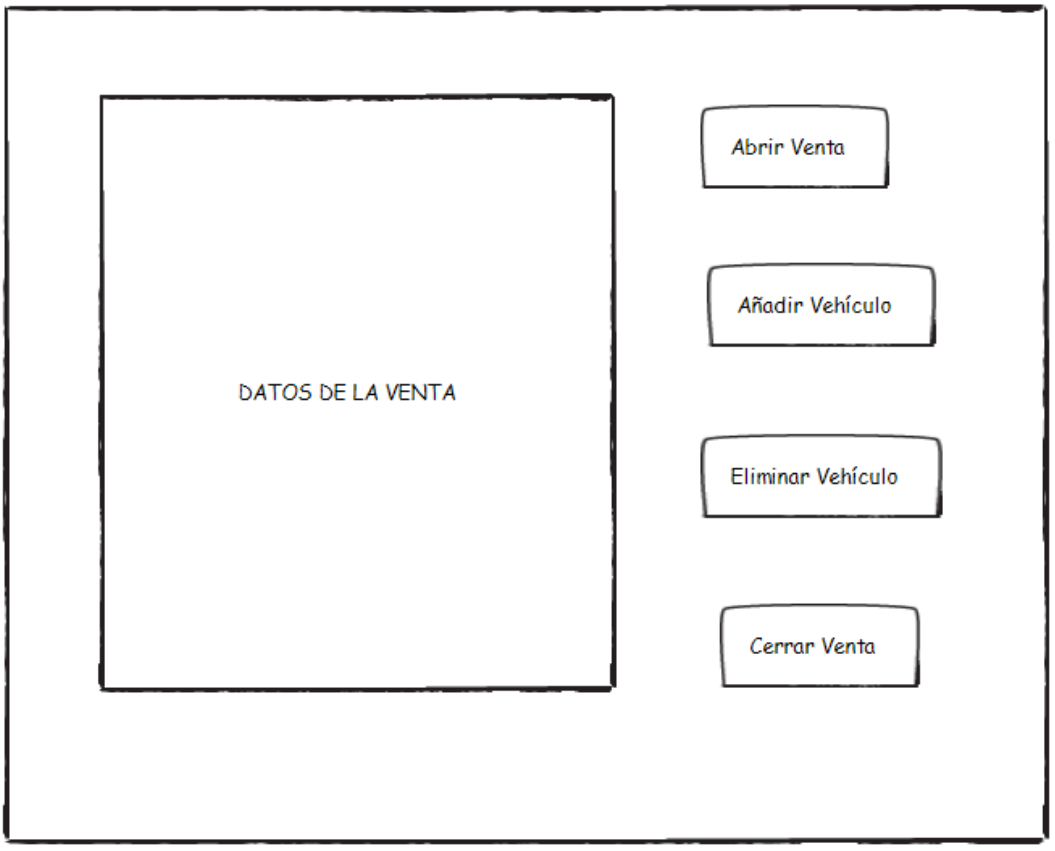


Actualizar/Consultar Modelo

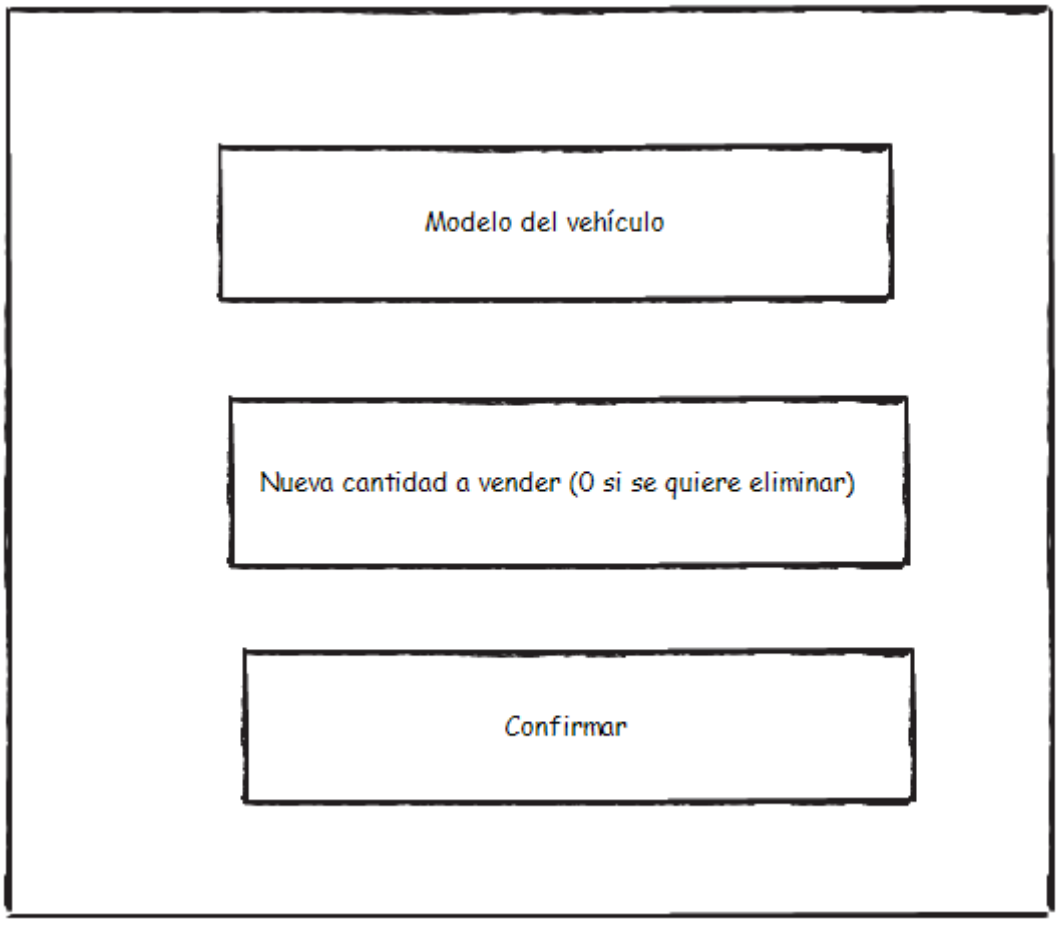


Modulo Venta

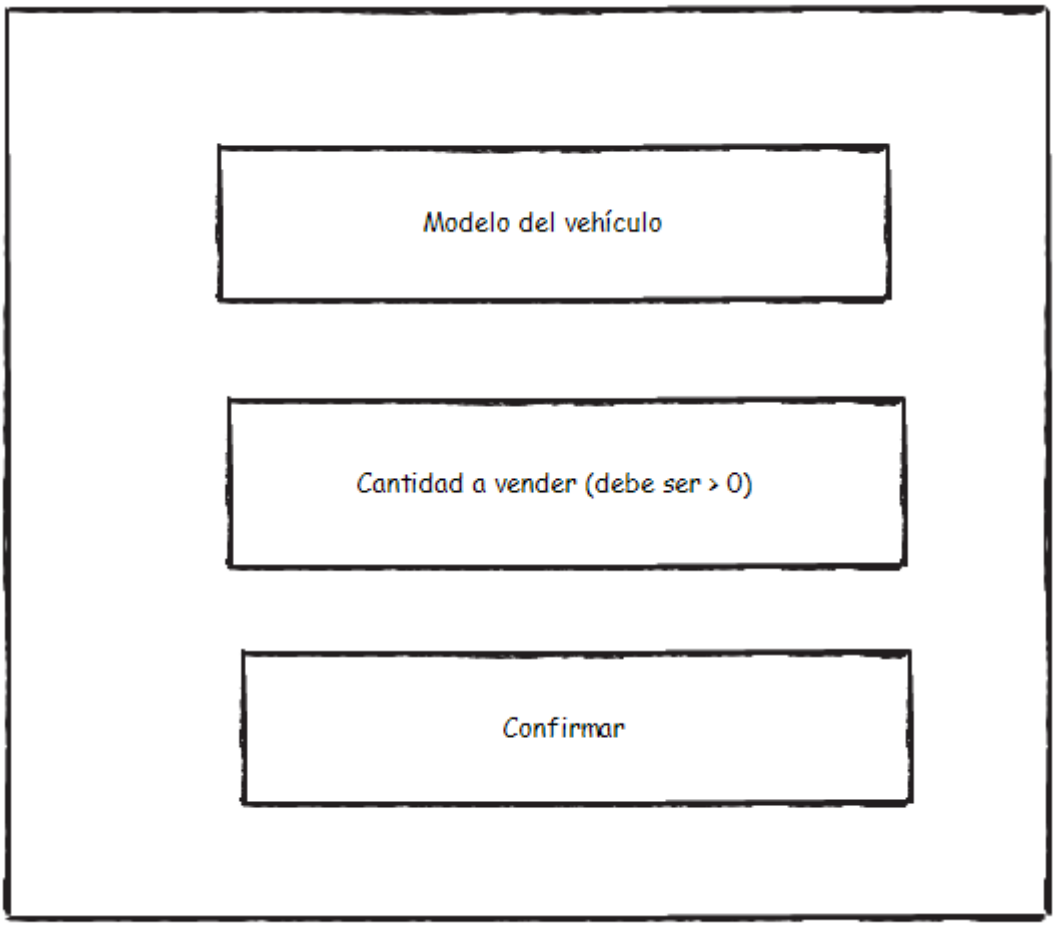
Crear Venta



Crear Venta ❼ Añadir Vehiculo

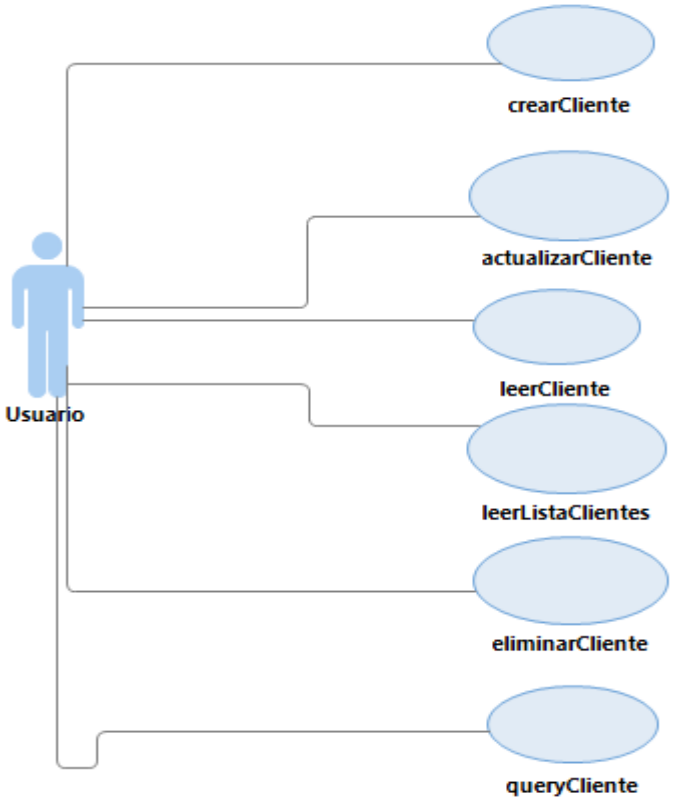


Crear Venta ❼ Eliminar Vehículo

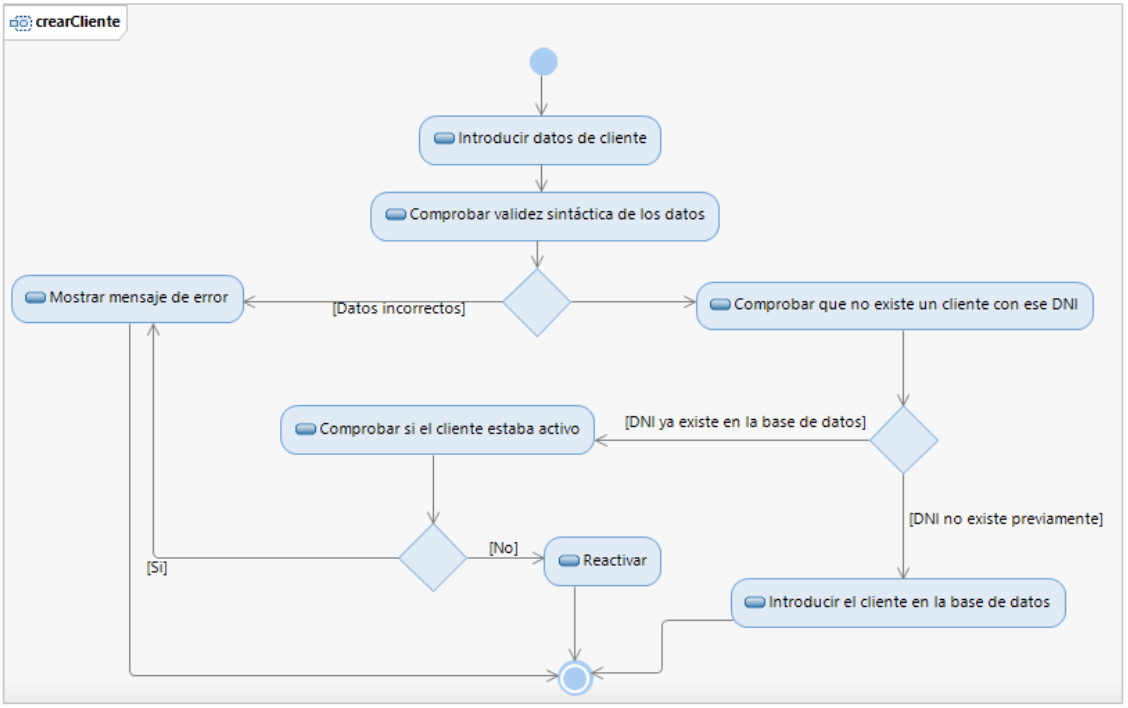


3.2 Funciones.

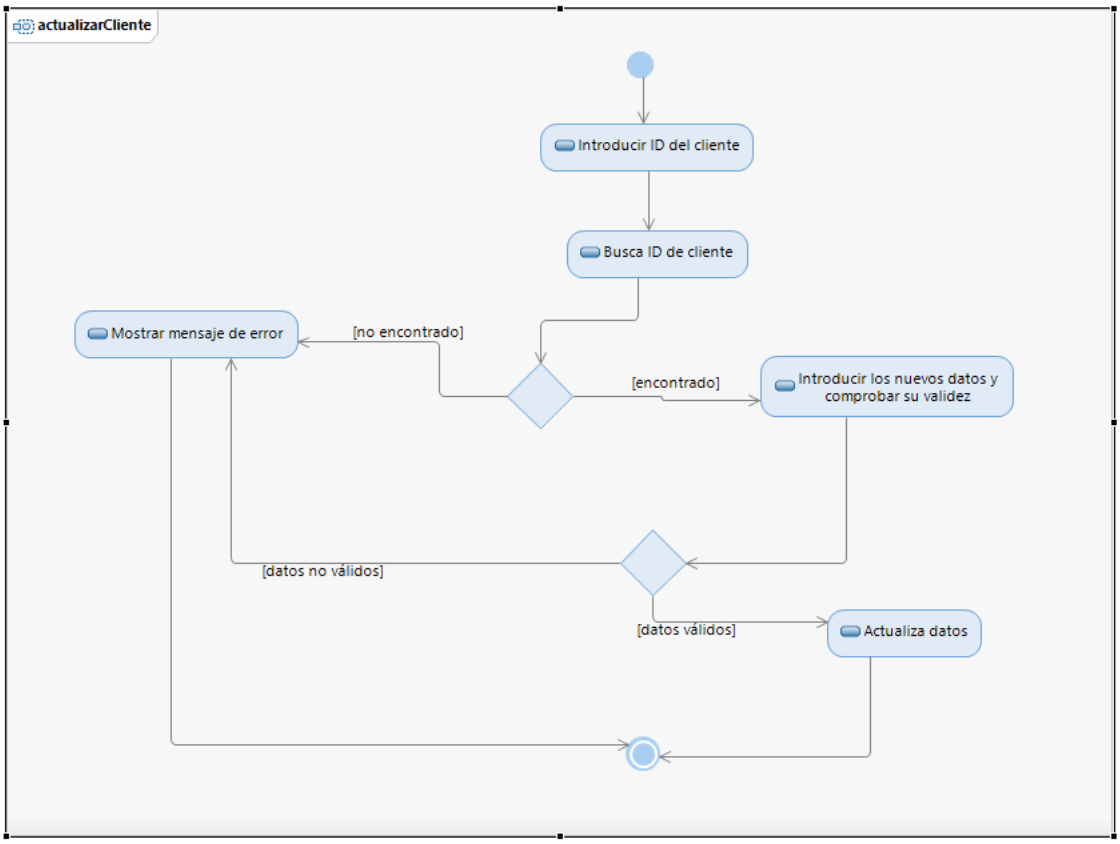
Modulo cliente.



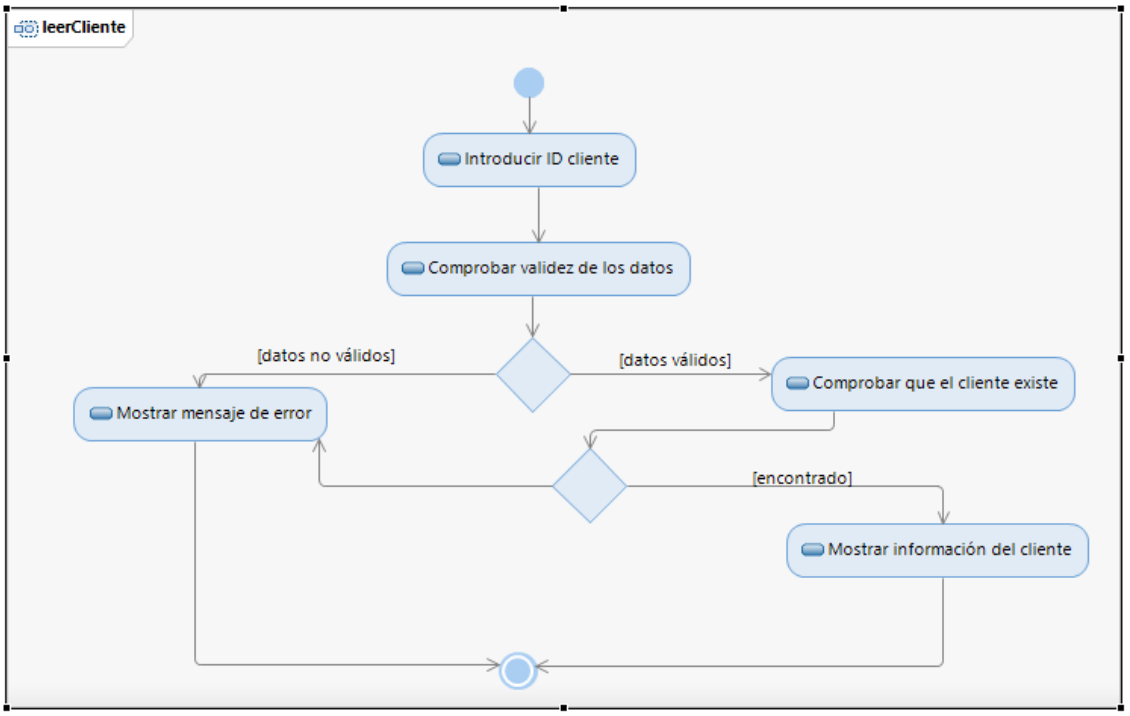
Requisito 1.1	crearCliente
Descrita por	Ignacio Vítores
Prioridad	Alta
Estabilidad	Alta.
Descripción	Añadir un nuevo cliente al base de datos
Entradas	Los datos del cliente
Salidas	Mensaje de éxito o error.
Origen	Interfaz de usuario – Vendedor.
Destino	Sistema.
Necesita	Base de datos de clientes.
Acciones	El vendedor introduce los datos de clientes. El sistema registra el nuevo cliente en el base de datos.
Precondición	No existen IDs ni DNI duplicados en la base de datos.
Pos condició	Valores en la BBDD consistente.
Efectos laterale	En caso de que los datos introducidos no son correctos o en caso de que ya existe el cliente, se mostrará un mensaje de



Requisito 1.2	actualizaCliente
Descrita por	Ignacio Vitores
Prioridad	Alta
Estabilidad	Alta.
Descripción	Actualizar los datos del cliente al base de datos
Entradas	Los datos del cliente
Salidas	Mensaje de éxito o error.
Origen	Interfaz de usuario – Vendedor.
Destino	Sistema.
Necesita	Base de datos de clientes.
Acciones	El vendedor introduce nuevos datos del cliente. El sistema actualiza los datos del cliente en el base de datos.
Precondición	Existe ID en la base de datos.
Pos condició	
Efectos laterale	En caso de que los datos introducidos no son correctos o en caso de que no existe el cliente, se mostrará un mensaje de

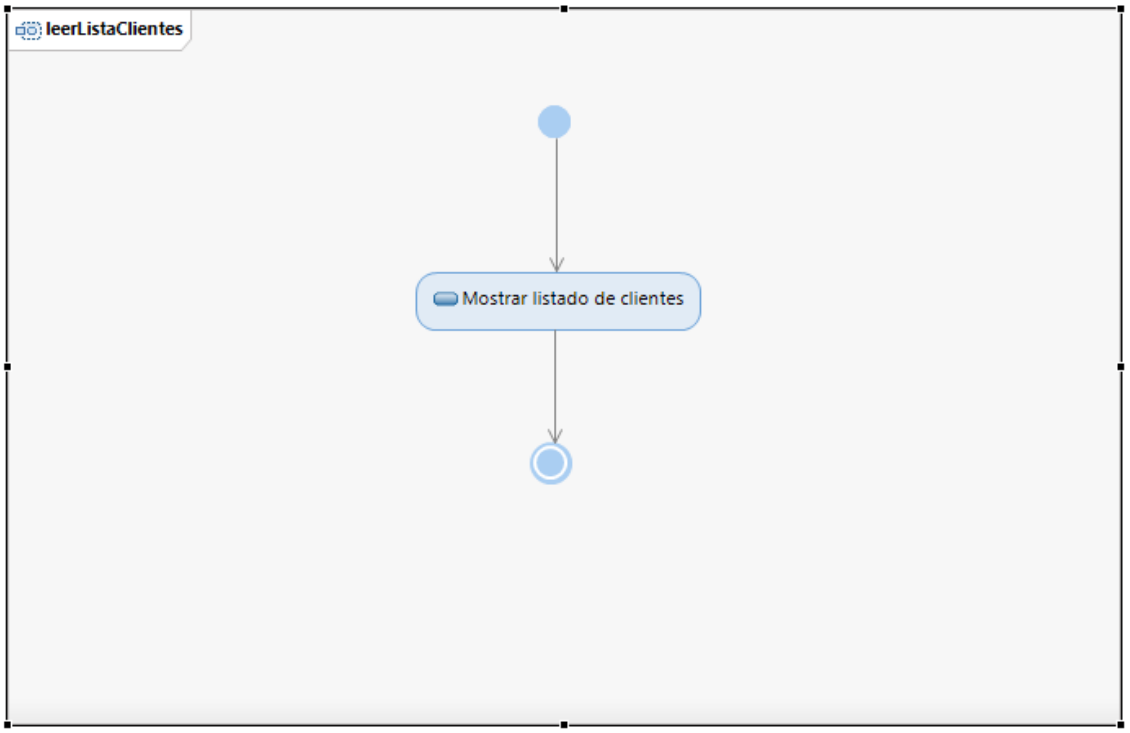


Requisito 1.3	leerCliente
Descrita por	Ignacio Vitores
Prioridad	Media
Estabilidad	Alta.
Descripción	Leer los datos de un cliente
Entradas	El ID del cliente
Salidas	Mensaje con la información del cliente
Origen	Interfaz de usuario – Vendedor.
Destino	Sistema.
Necesita	Base de datos de clientes.
Acciones	El vendedor introduce el ID del cliente. El sistema muestra la información del cliente por pantalla.
Precondición	Existe ID del cliente en la base de datos.
Poscondició	No se modifica la base de datos.
Efectos laterale	En caso de que el ID introducido no es correcto o en caso de que no existe el cliente, se mostrará un mensaje de error.



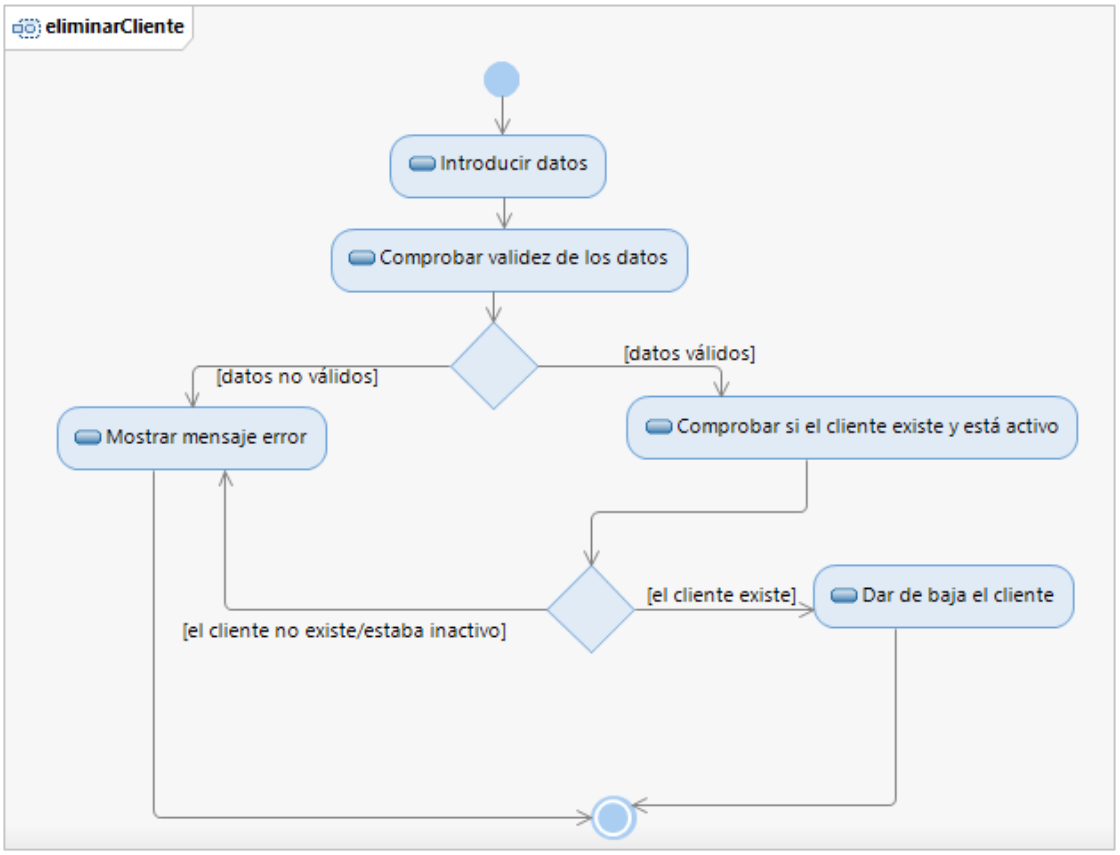
Requisito 1.4	leerTodoCliente
Descrita por	Ignacio Vitores
Prioridad	Media
Estabilidad	Alta.
Descripció	Leer los datos de todos los clientes
Entradas	Ninguna
Salidas	Mensaje con las informaciones de todos los clientes
Origen	Interfaz de usuario – Vendedor.
Destino	Sistema.
Necesita	Base de datos de clientes.
Acciones	El vendedor pide las informaciones de todos los clientes. El sistema muestra las informaciones de todos los clientes por
Precondición	La base de datos no es vacío.
Pos condició	No se modifica la base de datos.

Efectos laterale	En caso de que la base de datos es vacío, se mostrará un mensaje de error.



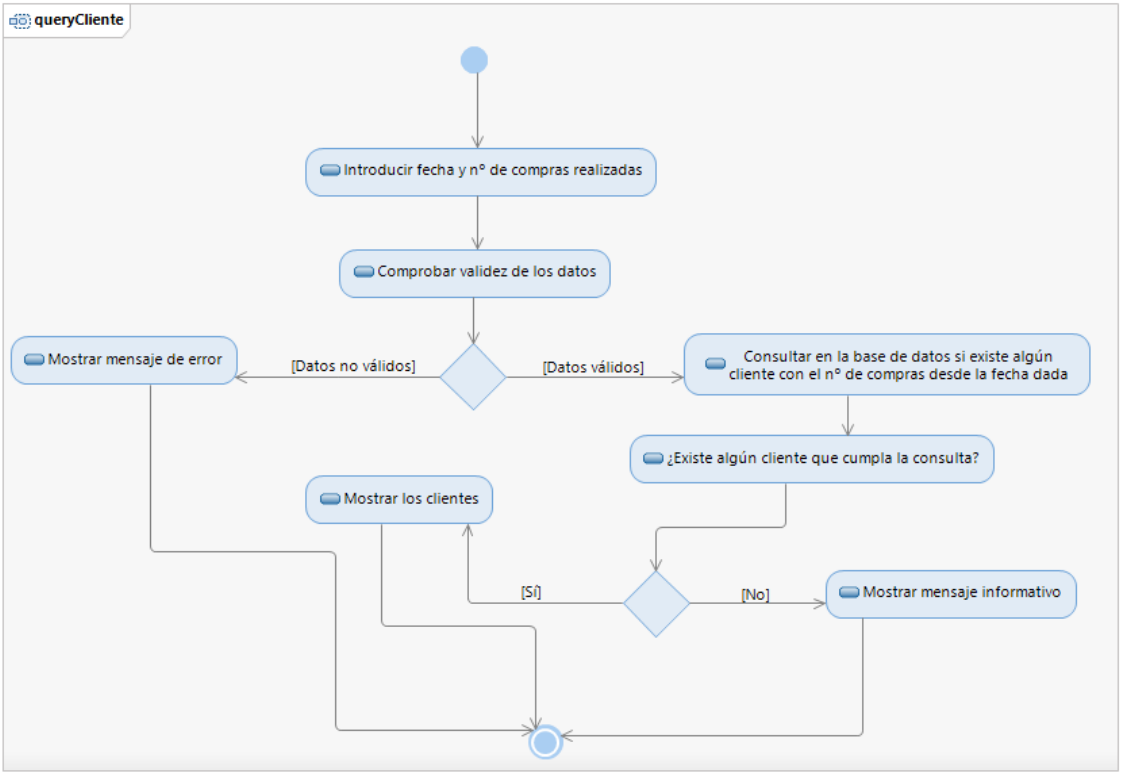
Requisito 1.5	eliminaCliente
Descrita por	Ignacio Vitores
Prioridad	Alta
Estabilidad	Alta.
Descripción	Eliminar el cliente al base de datos
Entradas	El ID del cliente
Salidas	Mensaje de éxito o error.
Origen	Interfaz de usuario – Vendedor.
Destino	Sistema.
Necesita	Base de datos de clientes.

Acciones	El vendedor introduce el ID del cliente. El sistema elimina los datos del cliente en el base de datos.
Precondición	Existe ID en la base de datos.
Pos condició	
Efectos laterale	En caso de que el ID introducido no es correcto o en caso de que no existe el cliente, se mostrará un mensaje de error.

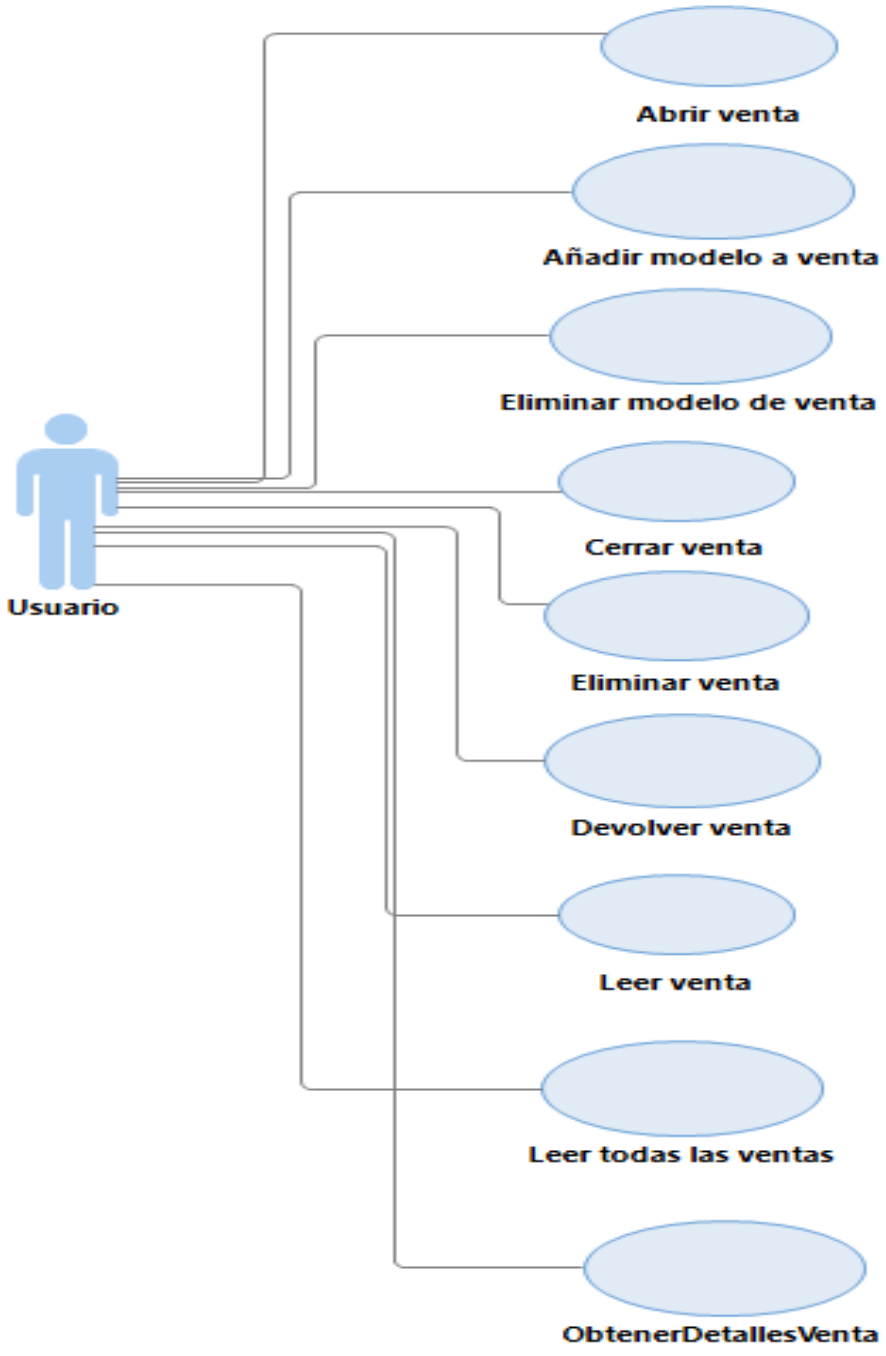


Requisito 1.6	consultaComprasCliente
Descrita por	Ignacio Vitores
Prioridad	Media
Estabilidad	Alta.
Descripción	Consultar N clientes que realidos compras desde la fecha indicada

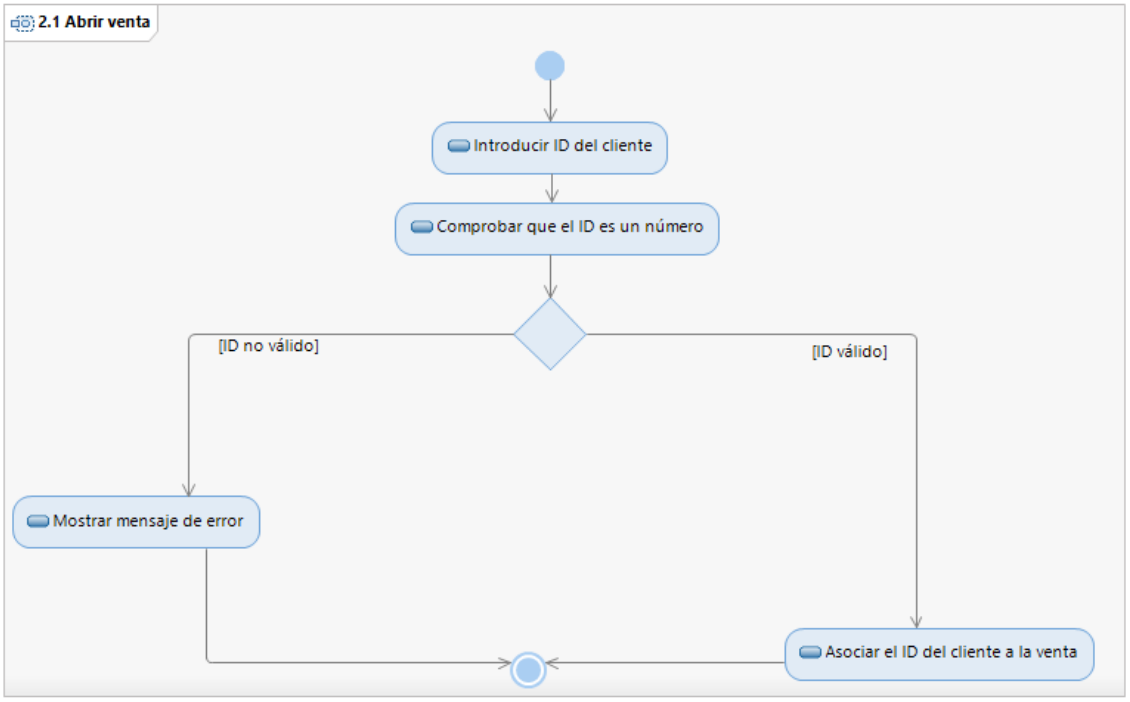
Entradas	El número de clientes que quieres consultar y la fecha
Salidas	Mensaje con la información de los clientes
Origen	Interfaz de usuario – Vendedor.
Destino	Sistema.
Necesita	Base de datos de clientes.
Acciones	El vendedor introduce el número y la fecha. El sistema muestra la información de los clientes.
Precondición	La fecha introducida es correcta.
Pos condición	No se modifica la base de datos.
Efectos laterales	En caso de que la fecha introducida no es correcta.



Modulo Venta

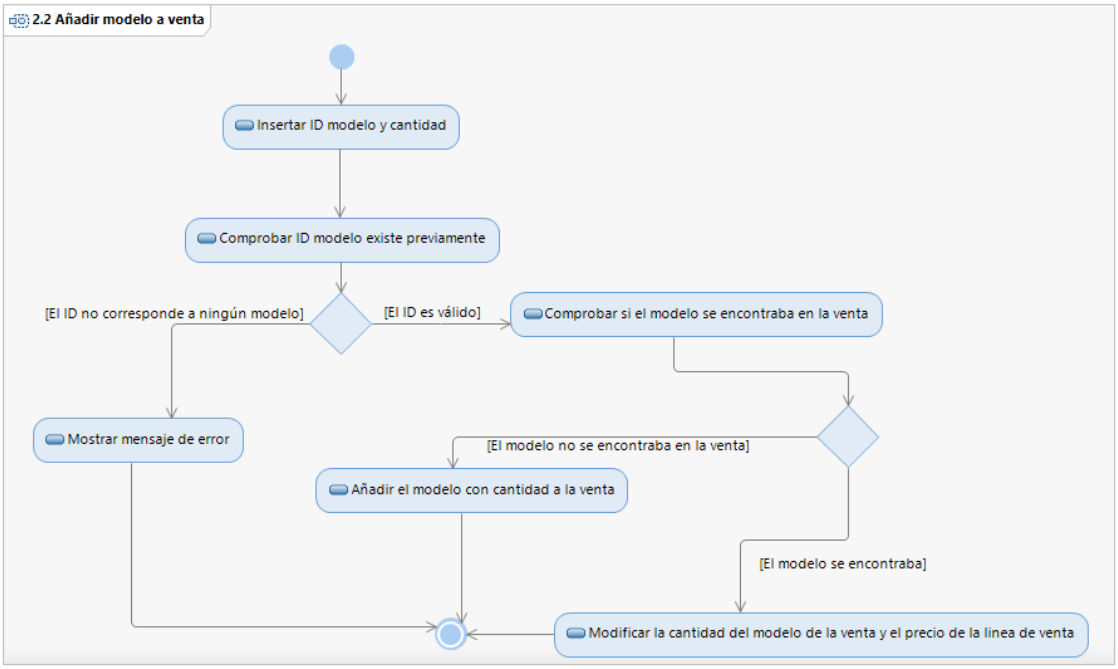


Requisito	abrirVenta
Descrita por	Ignacio Vítores Sancho
Prioridad	Alta.
Estabilidad	Alta.
Descripción	Función que abre una nueva venta.
Entradas	ID del cliente.
Salidas	Ventana que muestre el estado de la venta.
Origen	Interfaz de usuario.
Destino	Sistema.
Necesita	Base de datos de ventas y clientes.
Precondició	Ninguna
Pos	El id del cliente es valido
Efecto s	Si no se puede confirmar la venta, se mostrará un mensaje de error.

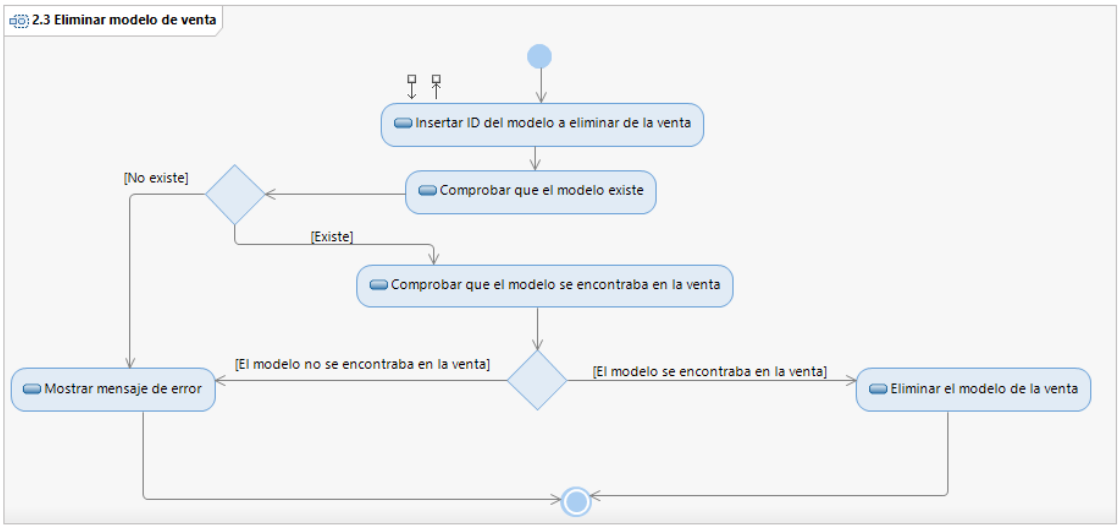


Requisito 2.1.2	añadirModeloAVenta
Descrita	Ignacio Vítores Sancho
Prioridad	Alta.
Estabilidad	Alta.
Descripció	Función que añade un vehículo a una venta.
Entradas	Venta, Id y cantidad del modelo.
Salidas	Venta con el nuevo estado
Origen	Interfaz de usuario.
Destino	Sistema.
Necesita	Base de datos de ventas y modelos.

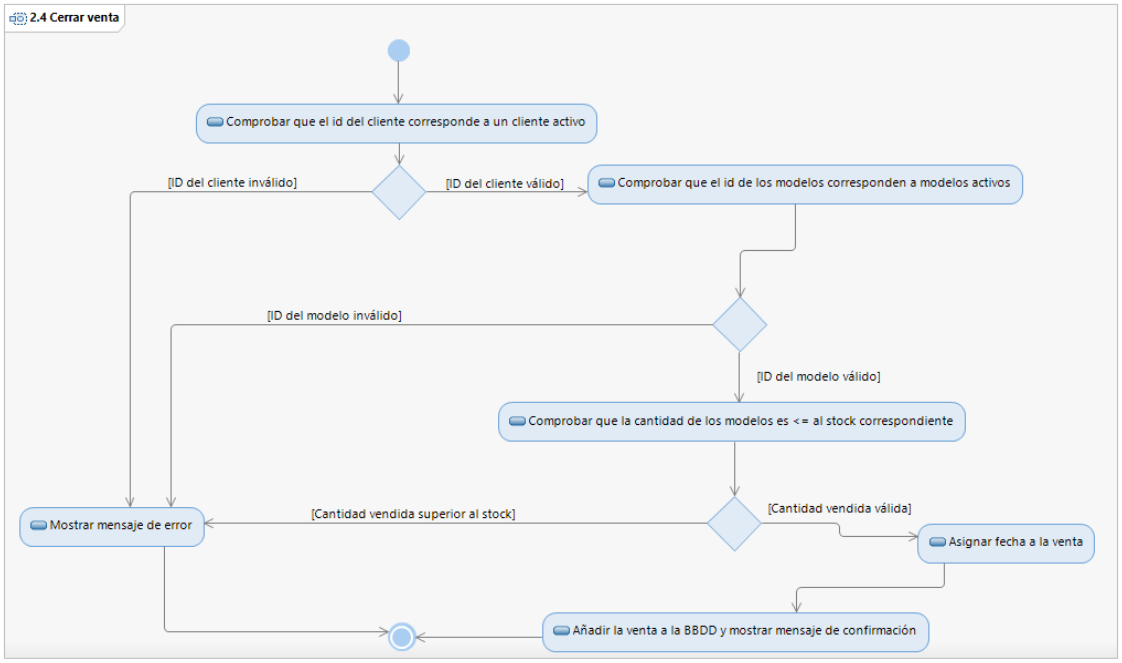
Precondi ció n	La venta, el id del modelo y la cantidad son válidos.
Pos condici	No se deja la cantidad de ningún modelo en el stock < 0 ni se venden modelos inactivos.
Efecto s	Si no se puede añadir el modelo, se mostrará un mensaje de error.



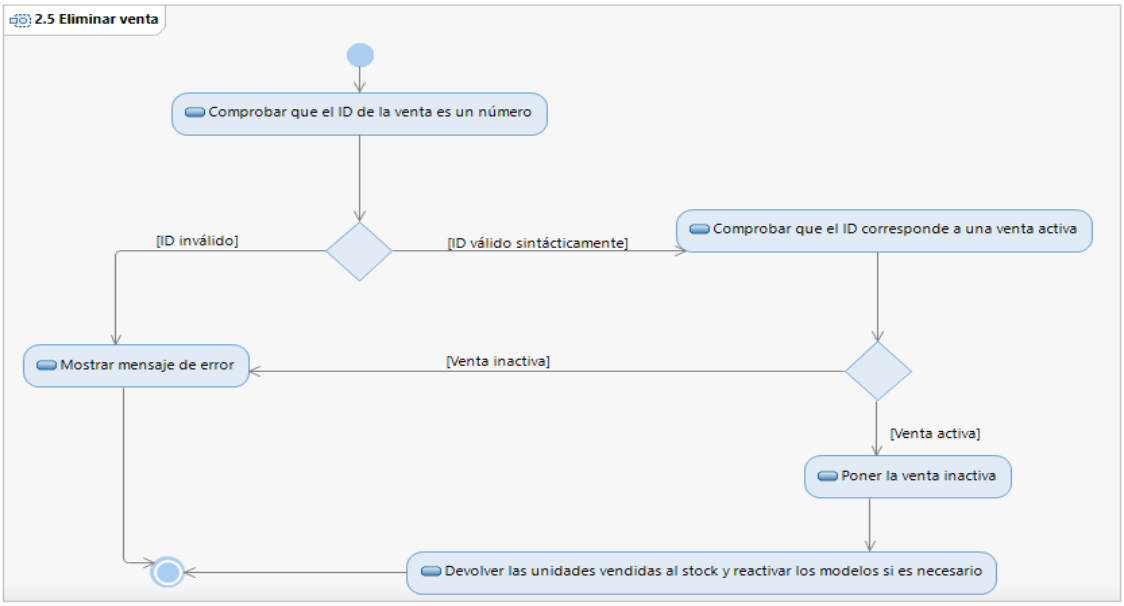
Requisito 2.1.3	eliminarModeloDeVenta
Descrita por	Ignacio Vítores Sancho
Prioridad	Alta.
Estabilidad	Alta.
Descripción	Función que quita un vehículo a una venta.
Entradas	Id del vehículo, la venta y la cantidad.
Salidas	Ventana que muestre el estado de la venta.
Origen	Interfaz de usuario.
Destino	Sistema.
Necesita	Base de datos de ventas y modelos.
Precondició n	La venta debe tener algún modelo. La cantidad debe ser positiva. El modelo debe
Pos condición	No hay IDs de venta repetidos en la base de datos. No se deja la cantidad de ningún
Efectos laterales	Si no se puede eliminar el modelo, se mostrará un mensaje de error.



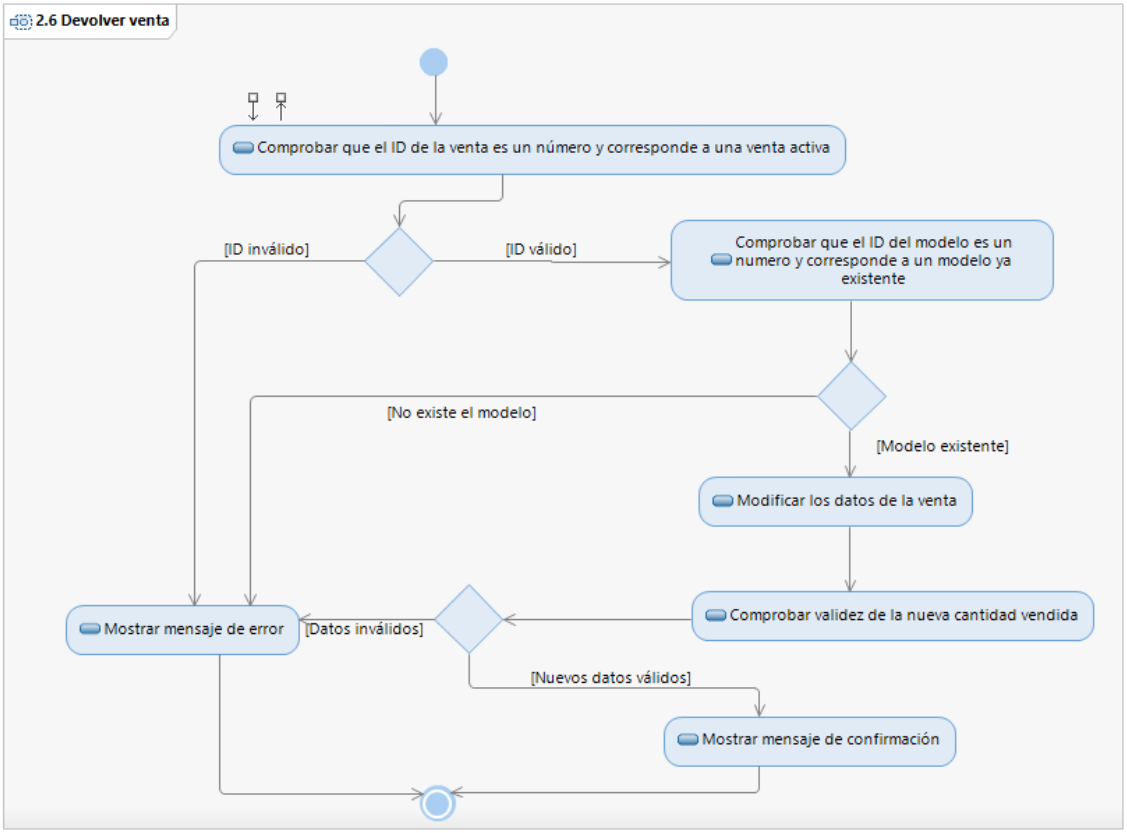
Requisito 2.1.4	cerrarVenta
Descrita por	Ignacio Vítores Sancho
Prioridad	Alta.
Estabilidad	Alta.
Descripción	Función que cierra y confirma una nueva venta.
Entradas	Venta.
Salidas	Ventana de confirmación de venta.
Origen	Interfaz de usuario.
Destino	Sistema.
Necesita	Base de datos de ventas, clientes y modelos.
Precondició n	La venta no tiene cantidades negativas.
Pos condición	No hay IDs de venta repetidos en la base de datos. No se deja stock de ningún modelo de
Efectos laterales	Si no se puede confirmar la venta, se mostrará un mensaje de error.



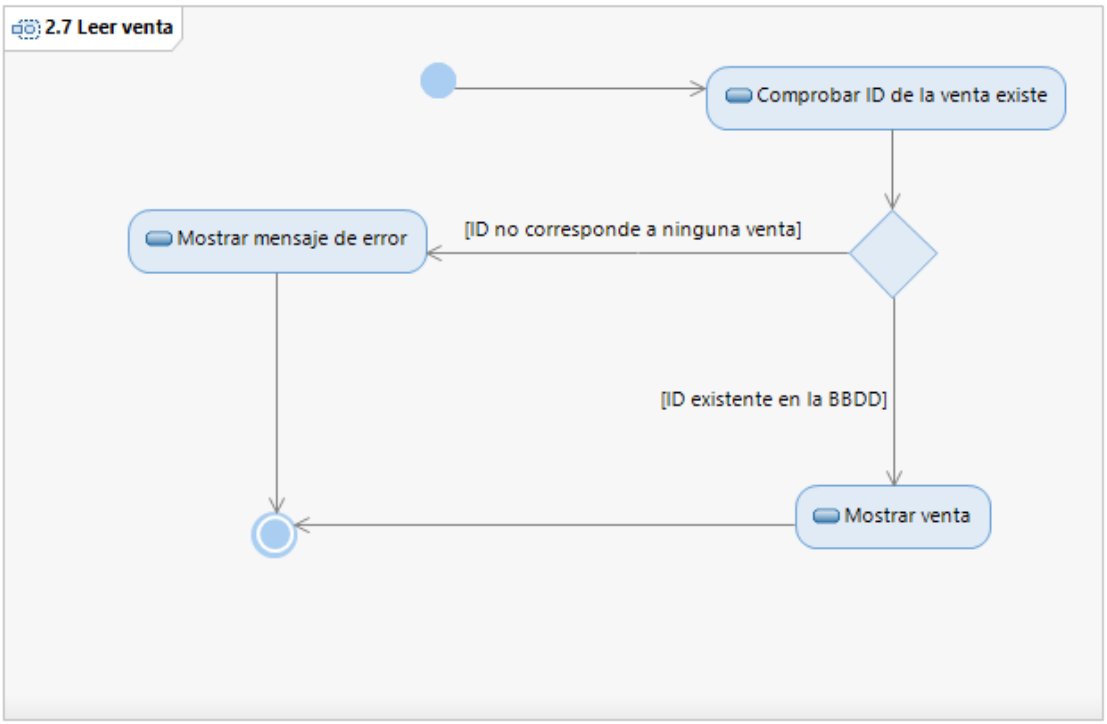
Requisito 2.3	eliminarVenta
Descrita	Ignacio Vítores Sancho
Prioridad	Alta.
Estabilidad	Alta.
Descripción	Función que elimina una venta de la base de datos, modificando los campos necesarios en las entidades cliente y vehículo (número de compras, vehículos
Entradas	ID de la venta.
Salidas	Se mostrará un mensaje confirmando la cancelación de la venta en caso de ser posible.
Origen	Interfaz de usuario.
Destino	Sistema.
Necesita	Base de datos de ventas, modelos y clientes.
Precondición	Ninguna.
Pos condició	El estado de la venta pasa a ser inactivo si existía. El ID de la venta existía previamente.
Efectos laterale	En caso de que no se encuentre la venta, se mostrará un mensaje de error.



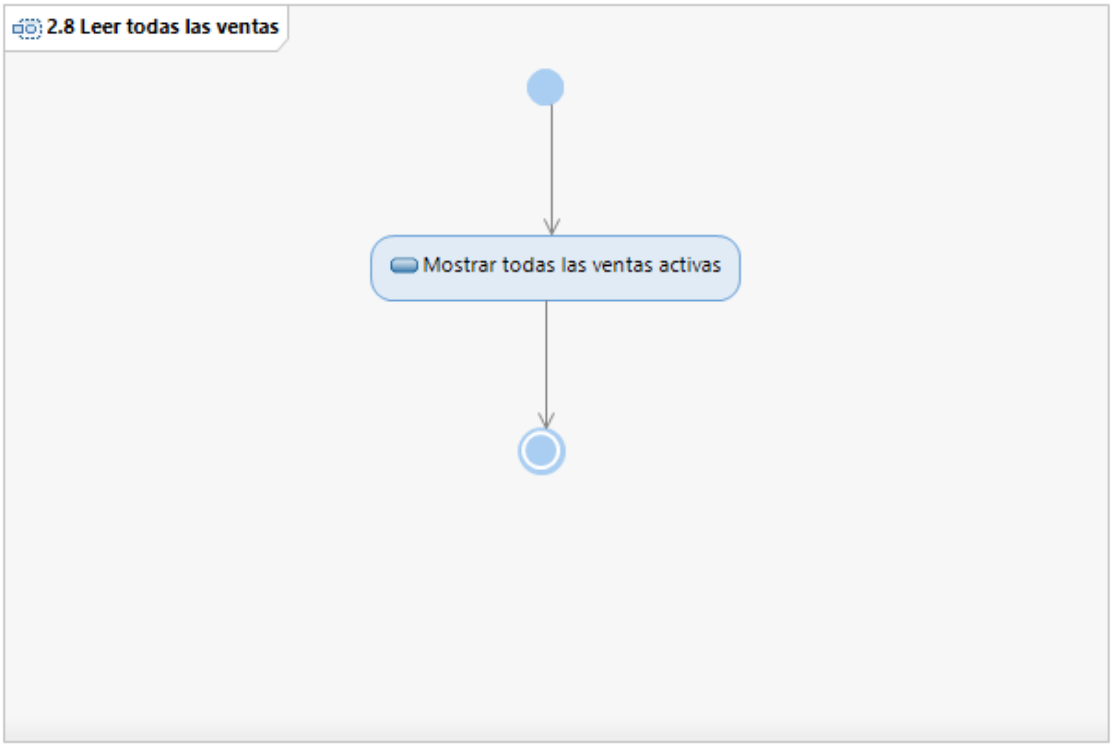
Requisito 2.2	devolverVenta
Descrita	Ignacio Vítores Sancho
Prioridad	Media.
Estabilidad	Alta.
Descripció	Función que modifica los datos de una venta y devuelve un
Entradas	ID de la venta.
Salidas	En caso de que los datos nuevos sean correctos, se mostrará un mensaje de éxito. Si no, se mostrará un
Origen	Interfaz de usuario
Destino	Sistema.
Necesita	Base de datos de ventas, modelos y clientes.
Precondició n	El ID de la venta no es nulo.
Pos condició n	No hay IDs de venta repetidos, el ID de la venta existía previamente y no se deja la cantidad de stock de ningún modelo de vehículo ni la cantidad vendida < 0.
Efectos laterale	Si algún dato se introduce en el formato incorrecto, se mostrará un mensaje de error.



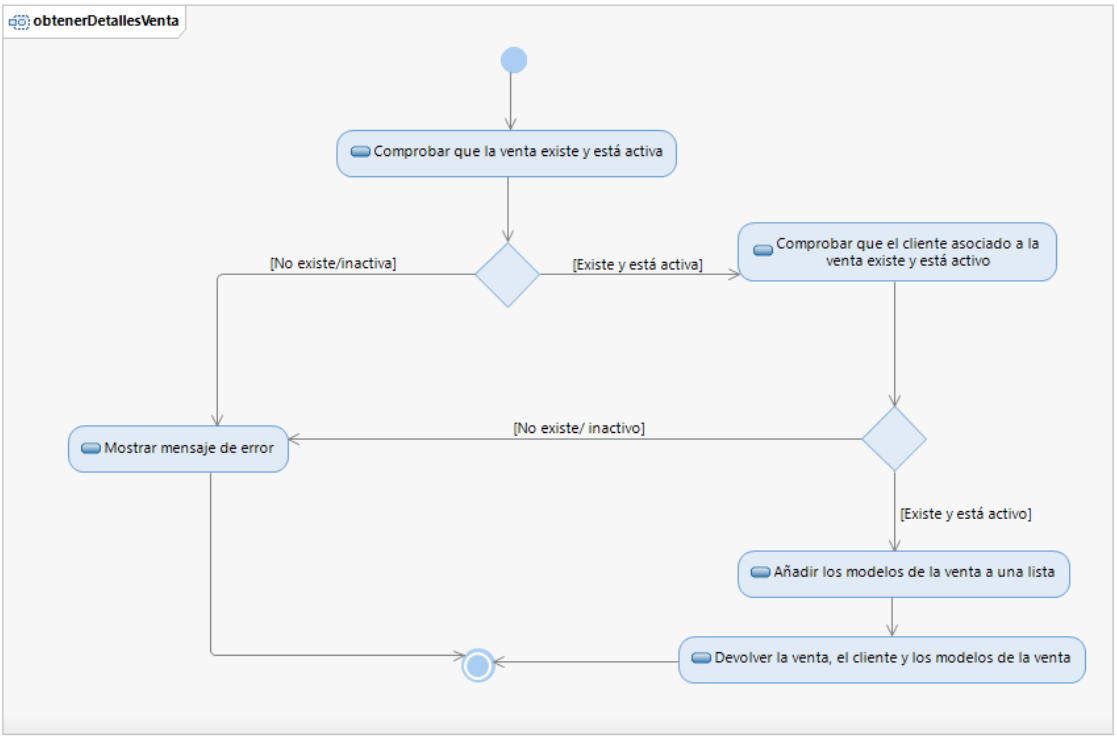
Requisito	leerVenta
Descrita por	Ignacio Vítores Sancho
Prioridad	Media.
Estabilidad	Alta.
Descripción	Función que muestra por pantalla la información correspondiente a una venta.
Entradas	ID de la venta de la que se quiere ver la información.
Salidas	Se mostrará una tabla con la información de la venta
Origen	Interfaz de usuario
Destino	Sistema.
Necesita	Base de datos de ventas.
Precondició	Ninguna.
Pos condición	La base de datos no se modifica. El ID de la venta existía previamente.
Efectos laterale	En caso de que la venta no se encuentre, se mostrará un mensaje de error.



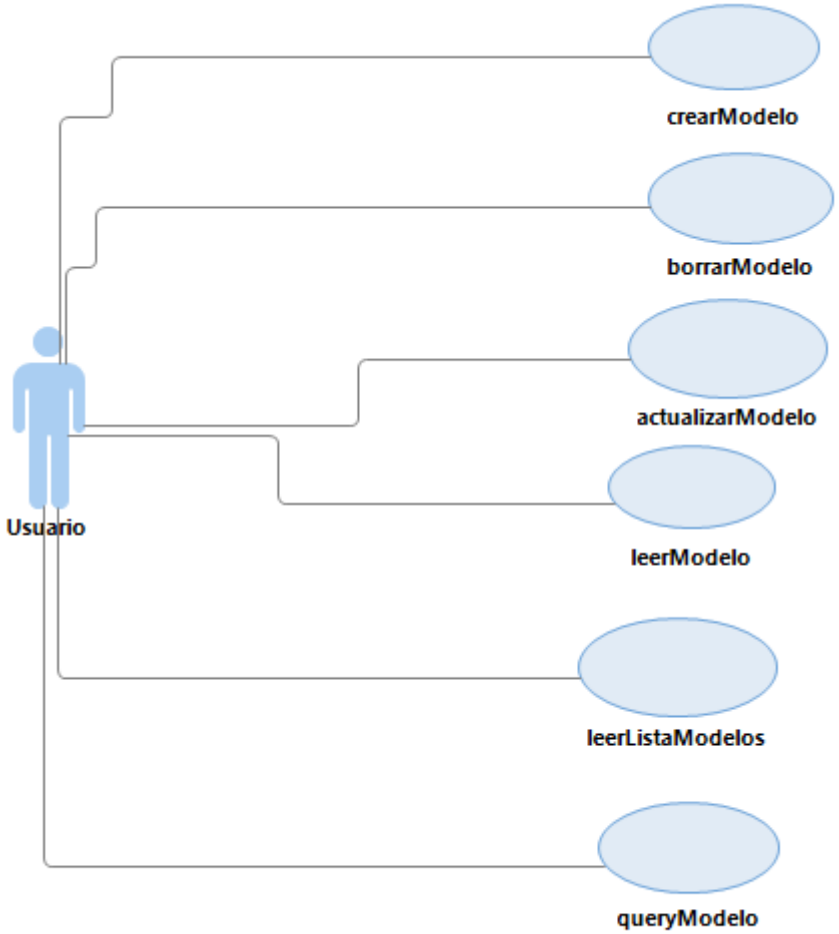
Requisito 2.5	leerTodasLasVentas
Descrita por	Ignacio Vítores Sancho
Prioridad	Media.
Estabilidad	Alta.
Descripción	Función que muestra por pantalla la información de todas las ventas que se encuentran en la base de
Entradas	Ninguna.
Salidas	Se mostrará por pantalla una lista con todas las ventas de la base de datos.
Origen	Interfaz de usuario
Destino	Sistema.
Necesita	Base de datos de ventas.
Precondición	Ninguna.
Pos condició	No se ha modificado ningún campo de la base de datos. Hay alguna venta introducida en la BBDD previamente.
Efectos laterale	En caso de que haya error al leer las ventas se mostrará un mensaje de error.



Requisito 2.6	ObtenerDetallesVenta
Descrita por	Ignacio Baena
Prioridad	Media.
Estabilidad	Alta.
Descripción	Funcion que muestra todos los detalles de una venta, los datos del cliente y los modelos
Entradas	ID de la venta de la que se quiere ver la información.
Salidas	Se mostrara por pantalla los datos del cliente y de los modelos
Origen	Interfaz de usuario
Destino	Sistema.
Necesita	Base de datos de ventas., clientes y modelos
Precondició n	Ninguna.
Pos condición	No se ha modificado ningún campo de la base de datos. Hay alguna venta introducida en la BBDD
Efectos laterales	En caso de que haya error al leer las ventas se mostrará un mensaje de error.

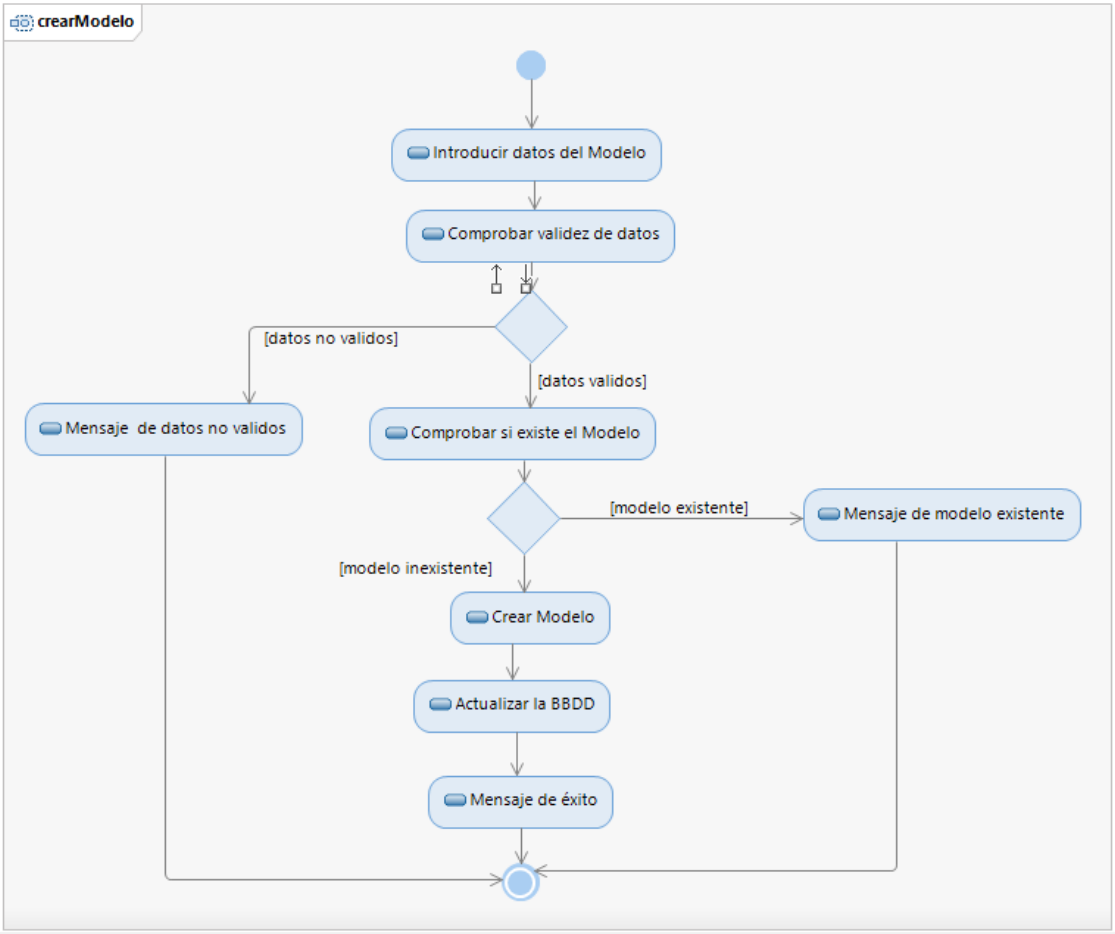

obtenerDetallesVenta
Comprobar que la venta existe y está activa
[No existe/inactiva]
[Existe y está activa]
Comprobar que el cliente asociado a la venta existe y está activo
Mostrar mensaje de error
[No existe/ inactivo]
[Existe y está activo]
Añadir los modelos de la venta a una lista
Devolver la venta, el cliente y los modelos de la venta

Módulo Modelo



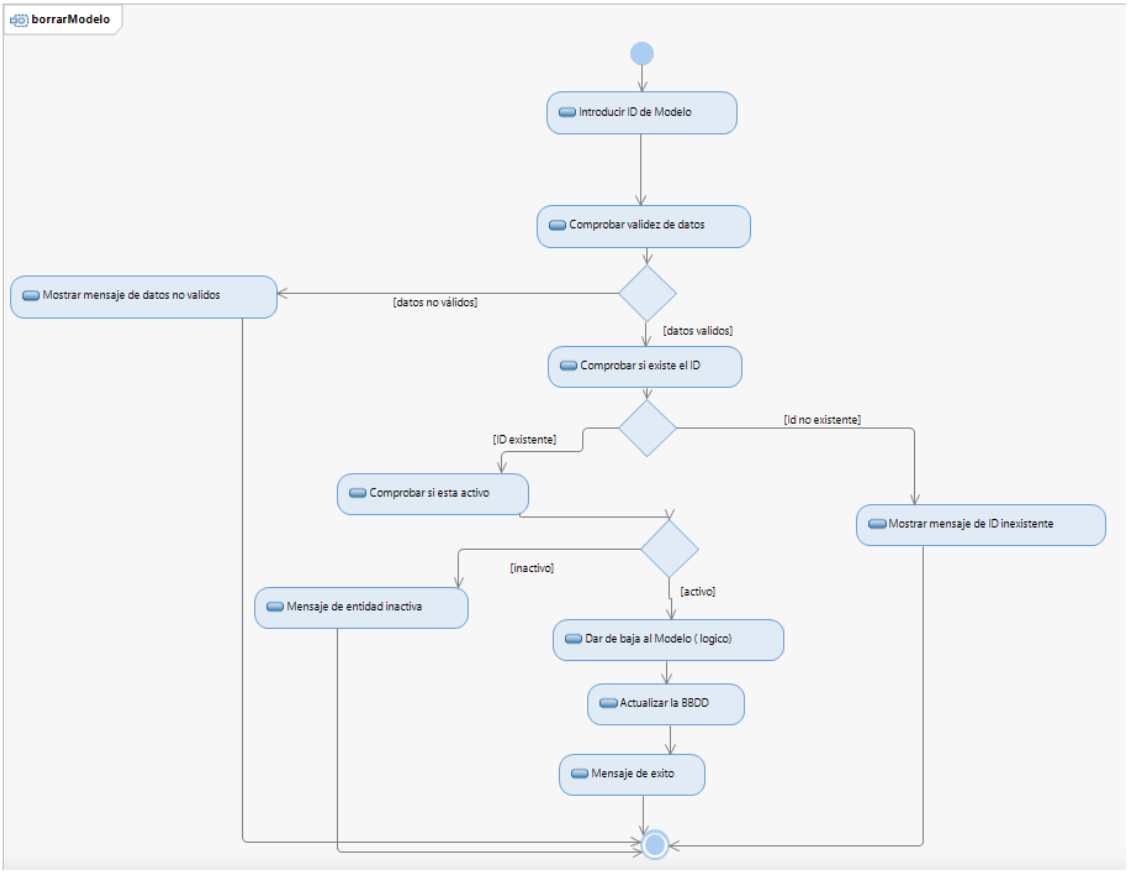
Requisitos	**Nuevo Modelo**
Descrita por	Ignacio Baena Kuroda
Prioridad	Alta.
Estabilidad	Alta.
Descripción	Añade un nuevo modelo a base de datos.
Actor	Encargado o Vendedor.
Entradas	ID.
Salidas	Mensaje confirmando la inserción.
Origen	Interfaz de usuario - Encargado/Vendedor.
Destino	Base de datos de modelos.
Necesita	Acceso a la base de datos de modelos.

Acciones	El sistema busca si existe un modelo con el ID dado. En caso negativo asigna un ID al modelo y lo inserta en la base de datos. Tras la inserción pasa al caso de uso 2.3
Precondició	No existen IDs repetidos en la base de datos.
Postcondició n	No existen IDs repetidos en la base de datos. Un nuevo modelo en la base de datos.
Efectos lateral	Si existe algún error durante la inserción se muestra un mensaje indicándolo.



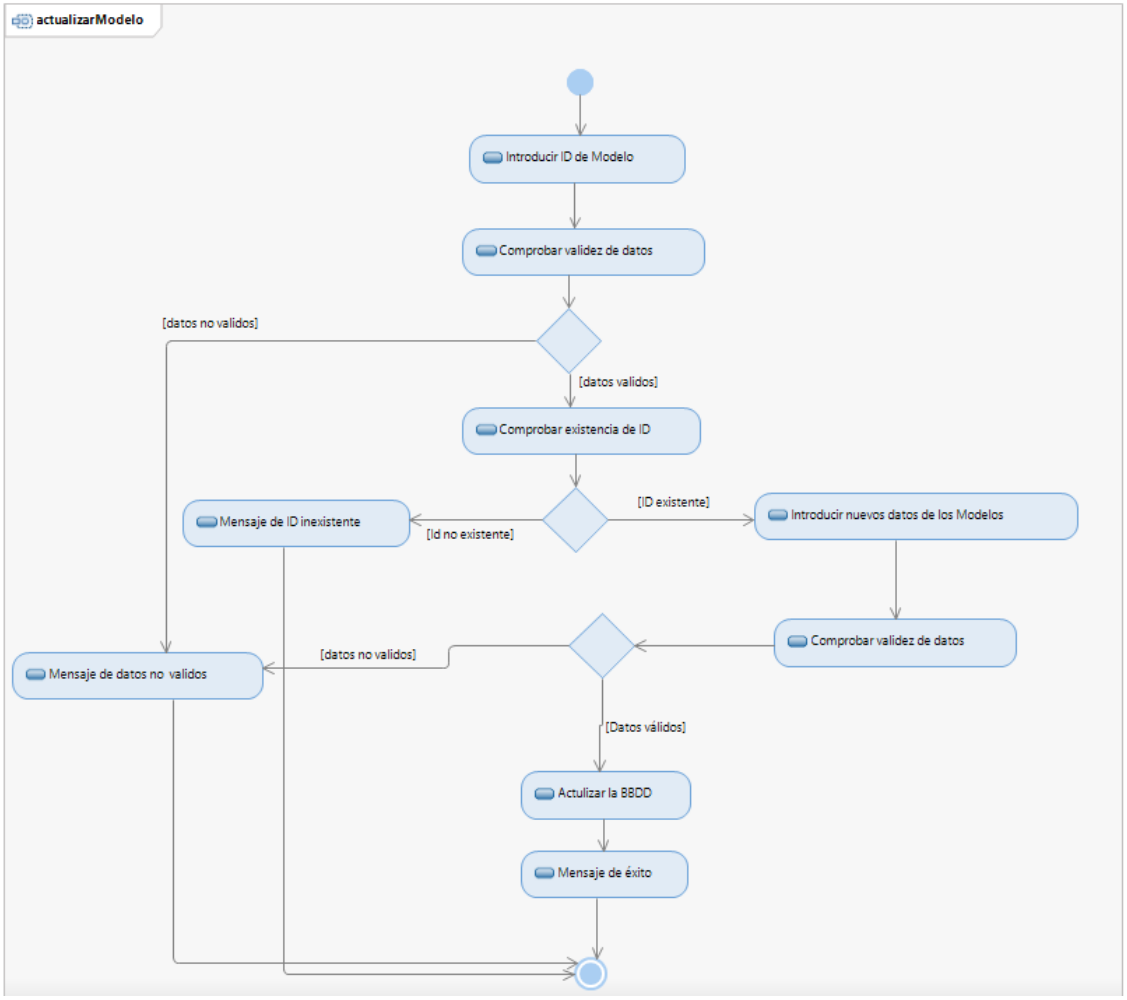
Requisitos 3.4	**Borrar Modelo**
Descrita por	Ignacio Baena Kuroda
Prioridad	Media.
Estabilidad	Alta.
Descripción	Elimina un modelo.

Actor	Encargado o Vendedor.
Entradas	ID de un modelo.
Salidas	Mensaje confirmando el borrado.
Origen	Interfaz de usuario - Encargado/Vendedor.
Destino	Base de datos de modelos.
Necesita	Acceso a la base de datos de modelos.
Precondición	No existen IDs repetidas en la base de datos.
Postcondición	No existen IDs repetidos en la base de datos. El modelo solicitado desaparece de la base de datos.
Efectos lateral	Si el ID no existe se muestra un mensaje indicándolo.



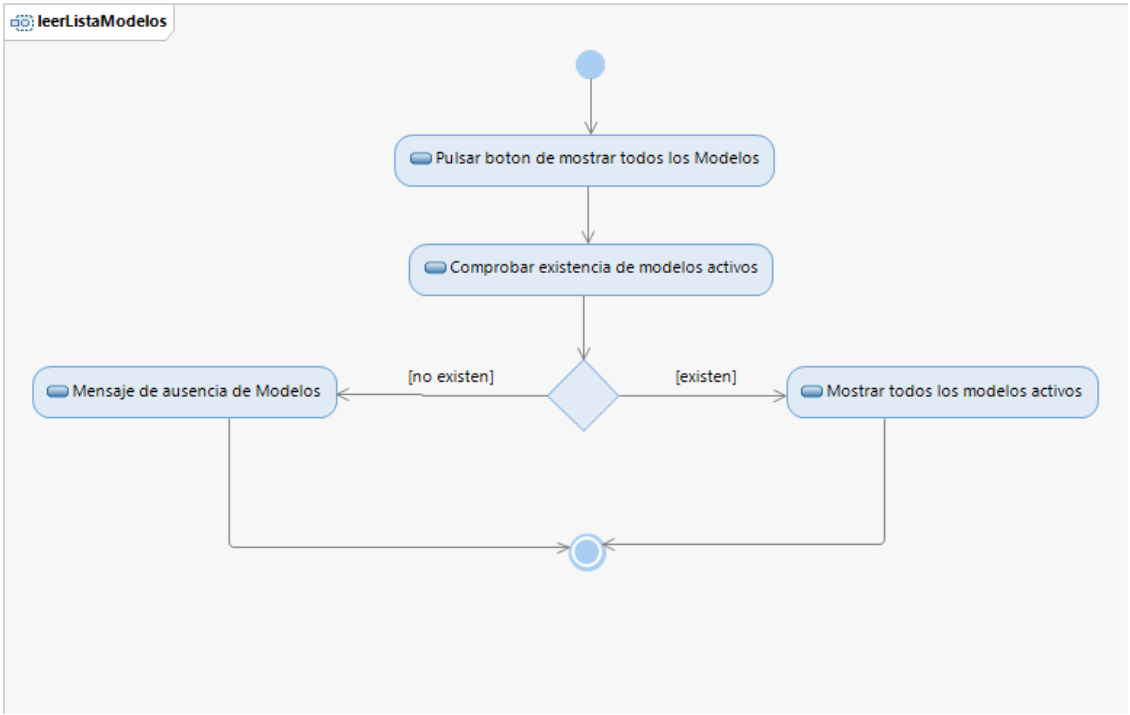
Requisitos	**Actualizar Modelo**
Descrita por	Ignacio Baena Kuroda
Prioridad	Alta.
Estabilidad	Alta.

Descripción	Modifica los datos de un modelo.
Actor	Encargado o Vendedor.
Entradas	ID de un modelo.
Salidas	Mensaje confirmando la modificación.
Origen	Interfaz de usuario - Encargado/Vendedor.
Destino	Base de datos de modelos.
Necesita	Acceso a la base de datos de modelos.
Precondició	No existen IDs repetidos en la base de datos.
Postcondició n	No existen IDs repetidas en la base de datos. No se crea ni se borra ningún modelo. El modelo
Efectos lateral es	Si no existe ningún modelo con ese ID se muestra un mensaje indicándolo.

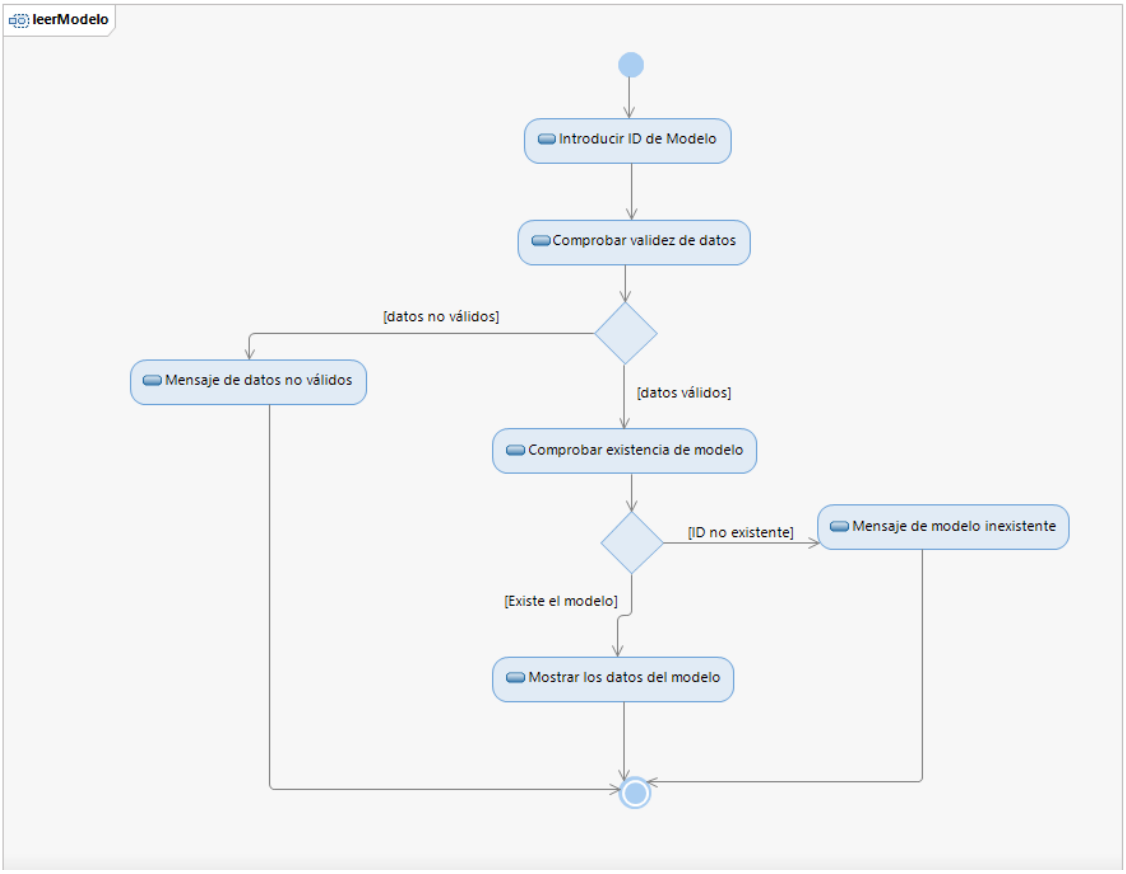


Requisitos 3.5	**Consultar Lista Modelos**
Descrita por	Ignacio Baena Kuroda
Prioridad	Media.

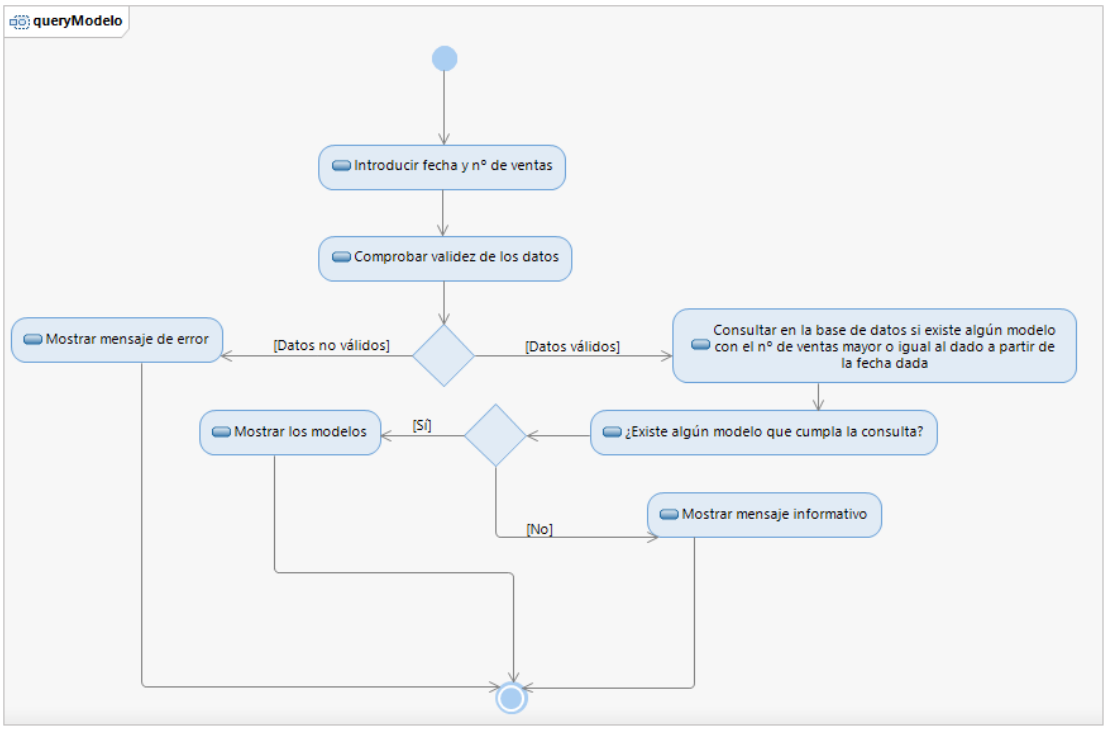
Estabilidad	Media.
Descripción	Muestra una lista con los datos de todos los modelos en la base de datos.
Actor	Encargado o Vendedor.
Entradas	Ninguna.
Salidas	Lista con datos de los modelos por pantalla.
Origen	Interfaz de usuario - Encargado/Vendedor.
Destino	Base de datos de modelos.
Necesita	Acceso a la base de datos de modelos.
Precondición	No existen IDs repetidas en la base de datos.
Postcondición	No existen IDs repetidas en la base de datos. La base de datos no sufre ningún cambio.
Efectos lateral	Si no existe ningún modelo se muestra un mensaje indicándolo.



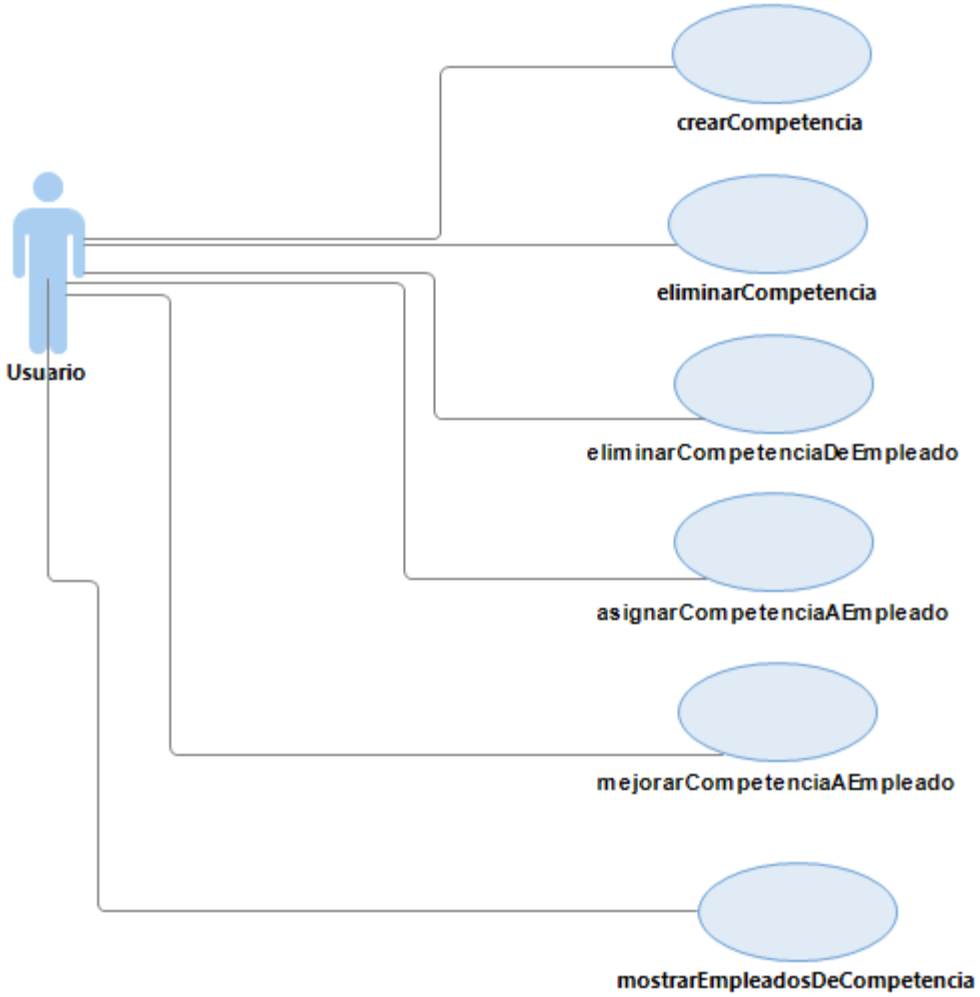
Requisitos 3.2	**Consultar Modelo**
Descrita por	Ignacio Baena Kuroda
Prioridad	Media.
Estabilidad	Alta.
Descripción	Muestra los datos de un modelo a partir de un ID dado.
Actor	Encargado o Vendedor.
Entradas	ID de un modelo.
Salidas	Datos del modelo por pantalla.
Origen	Interfaz de usuario - Encargado/Vendedor.
Destino	Base de datos de modelos.
Necesita	Acceso a la base de datos de modelos.
Acciones	El sistema busca en la base de datos el modelo con el ID dado. Si lo encuentra devuelve el modelo y muestra
Precondición	No existen IDs repetidos en la base de datos.
Postcondición	No existen IDs repetidos en la base de datos. La base de datos no sufre ningún cambio.
Efectos laterales	Si no existe ningún modelo con ese ID se muestra un mensaje indicándolo.



Requisitos	**Obtener lista modelos con N ventas en este año**
Descrita por	Ignacio Baena Kuroda
Prioridad	Baja.
Estabilidad	Media.
Descripción	Muestra por pantalla la lista de modelos con una cantidad de ventas igual a la dada. Si no hay ninguno se mostrará el menor con ventas superiores a la dada.
Actor	Encargado o Vendedor.
Entradas	Ventas deseadas.
Salidas	Modelos con N ventas este año.
Origen	Interfaz de usuario - Encargado/Vendedor.
Destino	Base de datos de modelos.
Necesita	Acceso a la base de datos de modelos.
Acciones	El sistema busca en la base de datos los modelos vendidos en este año y calcula cuáles son los modelos con un número de ventas igual al introducido. En caso de no encontrarlo buscará el menor de los que tengan ventas superiores a la deseada. Si lo encuentra lo muestra por
Precondició	No existen IDs repetidos en la base de datos.
Postcondició n	No existen IDs repetidos en la base de datos. La base de datos no sufre ningún cambio.
Efectos lateral	Si no existen modelos con ventas iguales o superiores a la introducida se indicará por pantalla.

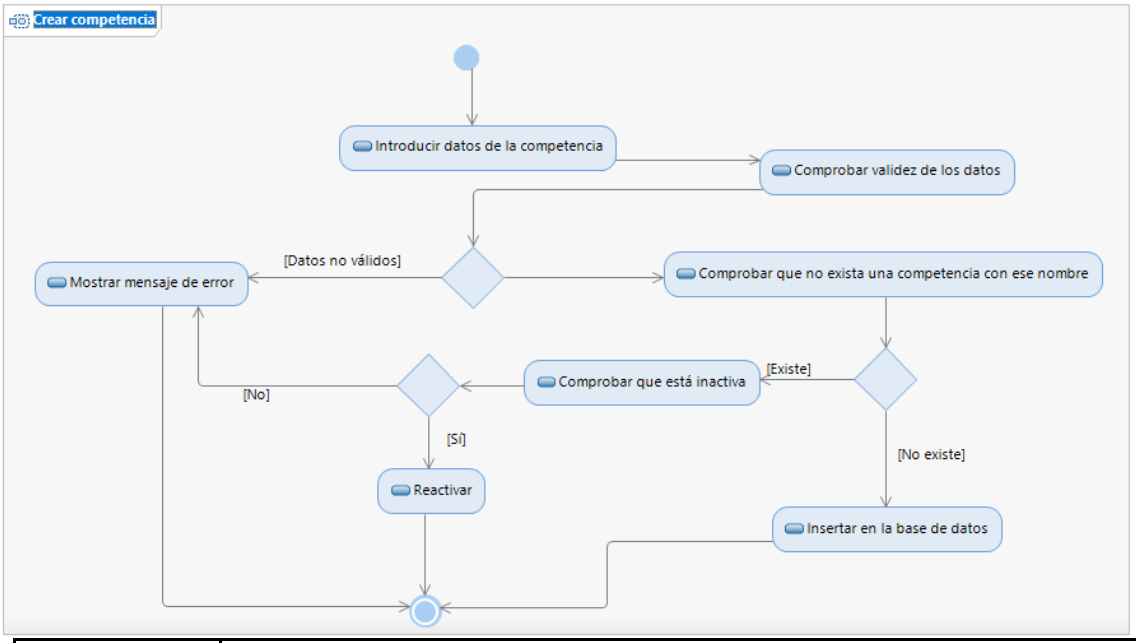

queryModelo
Introducir fecha y nº de ventas
Comprobar validez de los datos
Mostrar mensaje de error
[Datos no válidos]
[Datos válidos]
Consultar en la base de datos si existe algún modelo con el nº de ventas mayor o igual al dado a partir de la fecha dada
Mostrar los modelos
[Sí]
¿Existe algún modelo que cumpla la consulta?
Mostrar mensaje informativo
[No]

Módulo Competencia

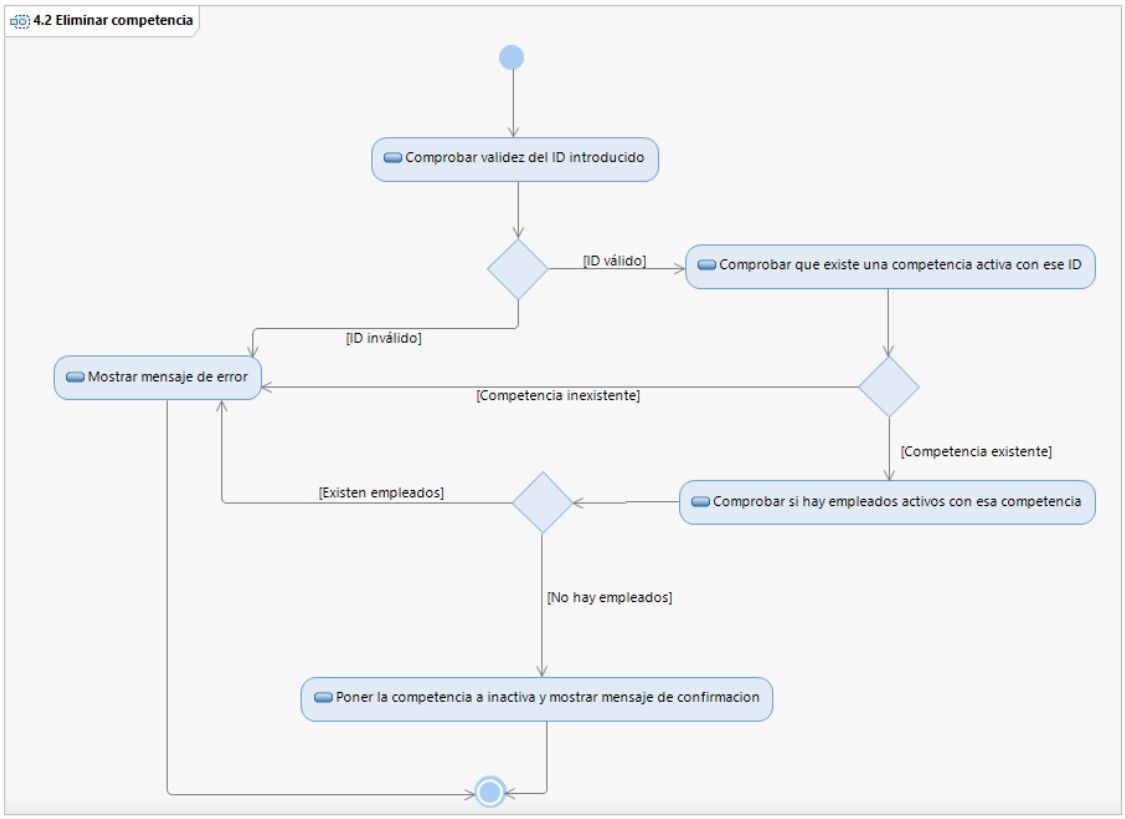


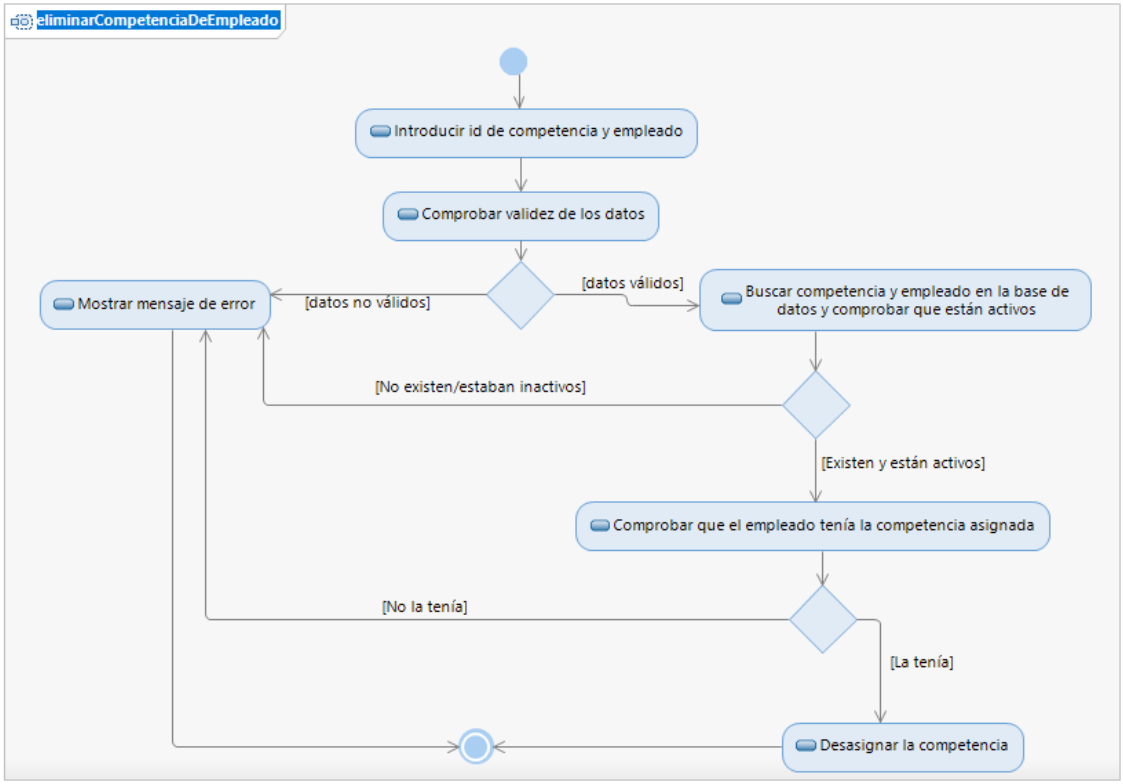
Requisito 4.1	Añadir Competencia
Descrita	Ignacio Baena Kuroda
Prioridad	Alta
Estabilidad	Alta.
Descripción	Función que añade una nueva competencia a la base de datos asignándole un ID
Entradas	Datos de la competencia (descripcion)
Salidas	Una competencia con todos los datos introducidos
Origen	Interfaz de usuario – Administrador.
Destino	Sistema.
Necesita	Base de datos de competencias.
Precondición	No exista competencia con el mismo nombre
Pos condició	Una competencia con todos sus datos

Efectos laterale	En caso de que los datos introducidos no sean correctos o contengan caracteres no válidos se mostrará un



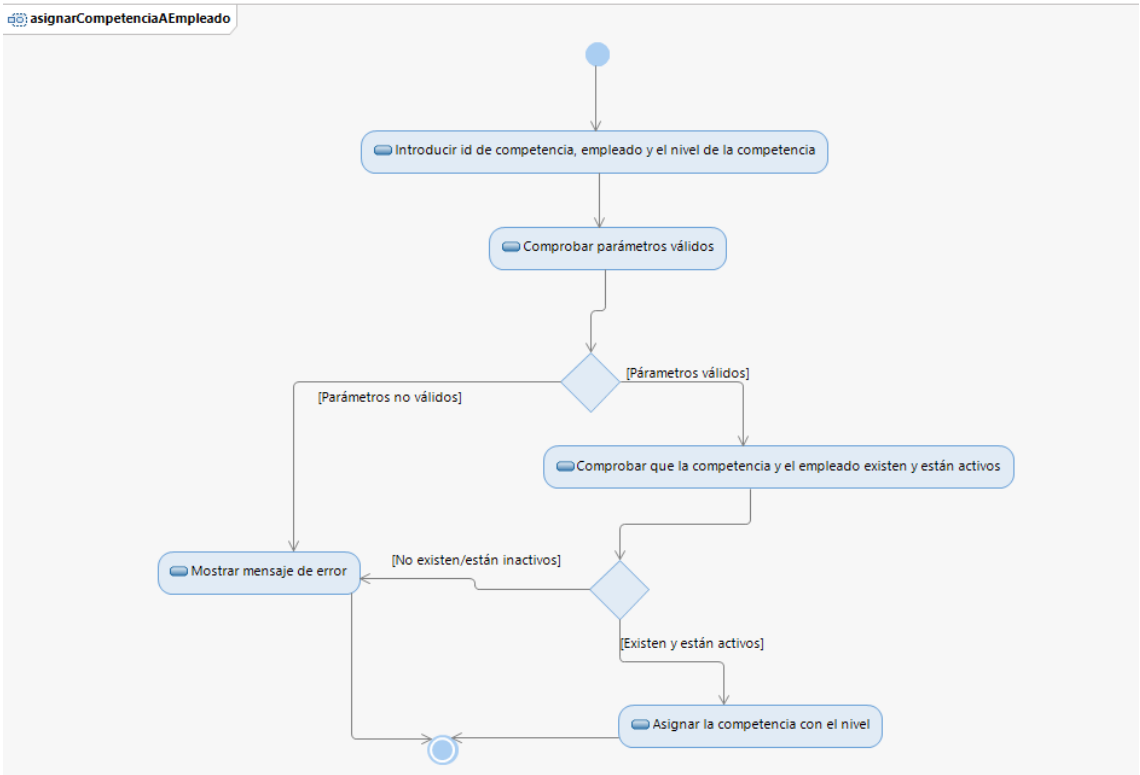
Requisito 4.2	Eliminar Competencia
Descrita por	Ignacio Baena Kuroda
Prioridad	Alta
Estabilidad	Alta.
Descripción	Función da de baja una competencia de la base de datos
Entradas	El ID de la competencia
Salidas	Mensaje con la confirmación de la competencia dada de
Origen	Interfaz de usuario – Administrador.
Destino	Sistema.
Necesita	Base de datos de empleados.
Precondición	No existen IDs duplicados en la base de datos.
Pos condició	Si existen empleados con dicha competencia se eliminarán también.
Efectos laterale	En caso de que ya este eliminado o no se encuentre mostrará un mensaje de error





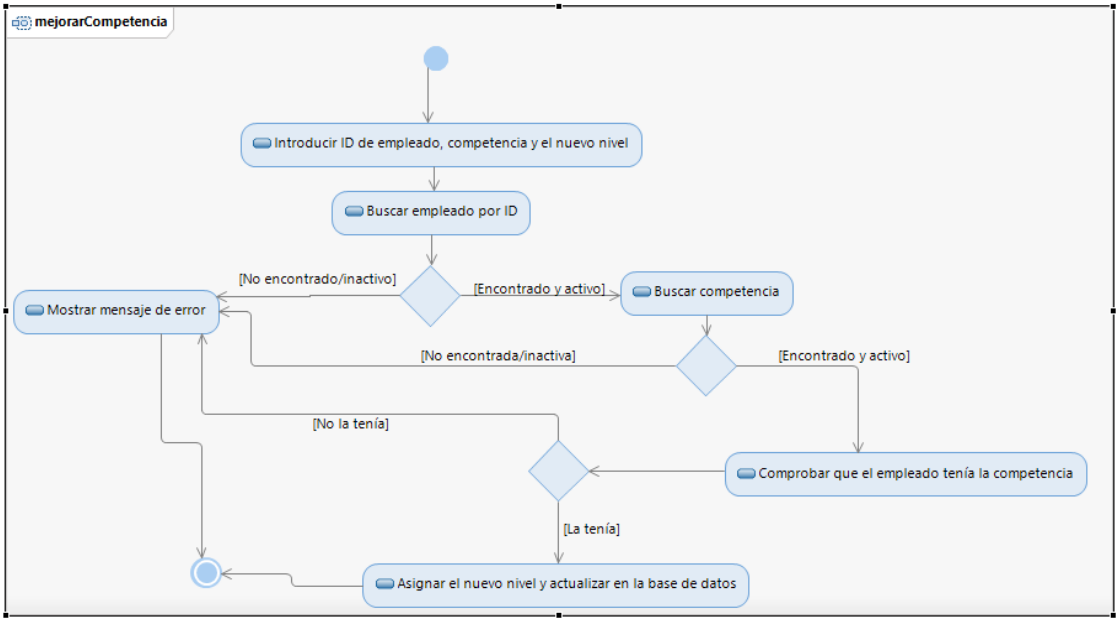
Requisito 4.6	Añadir Competencia a un Empleado
Descrita	Ignacio Baena Kuroda
Prioridad	Alta
Estabilidad	Alta.

Descripció	Función que asigna una competencia a un Empleado.
Entradas	ID de empleado y ID de competencia.
Salidas	Un empleado con una competencia nueva
Origen	Interfaz de usuario – Administrador.
Destino	Sistema.
Necesita	Base de datos de competencias. Base de datos de
Precondici ó n	No existan IDs duplicados de competencia ni empleados
Pos condició	Una empleado completamente actualizado.
Efectos laterale	En caso de que los datos introducidos no sean correctos o contengan caracteres no válidos se mostrará un

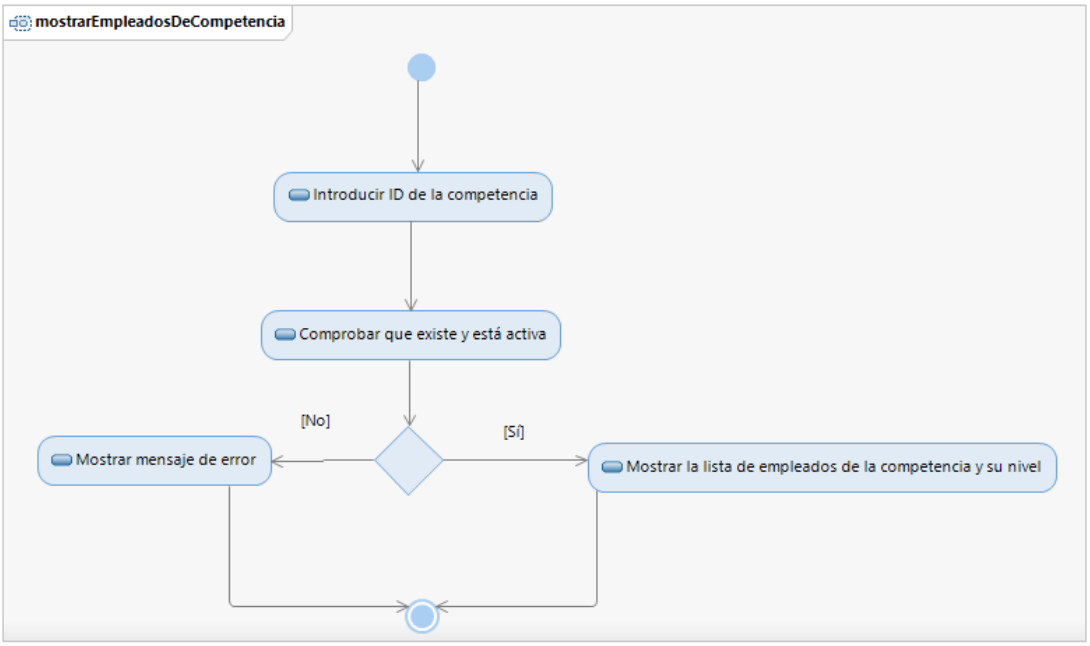


Requisit o 4.7	Mejorar Competencia de un Empleado
Descrita	Ignacio Baena Kuroda
Prioridad	Alta
Estabilidad	Alta.
Descripció	Función que actuliaza el nivel de una competencia a un
Entradas	ID de empleado y ID de competencia. Y nivel
Salidas	Un empleado con una competencia nueva
Origen	Interfaz de usuario – Administrador.
Destino	Sistema.
Necesita	Base de datos de competencias. Base de datos de
Precondici ó n	No existan IDs duplicados de competencia ni empleados
Pos condició	Una empleado completamente actualizado.

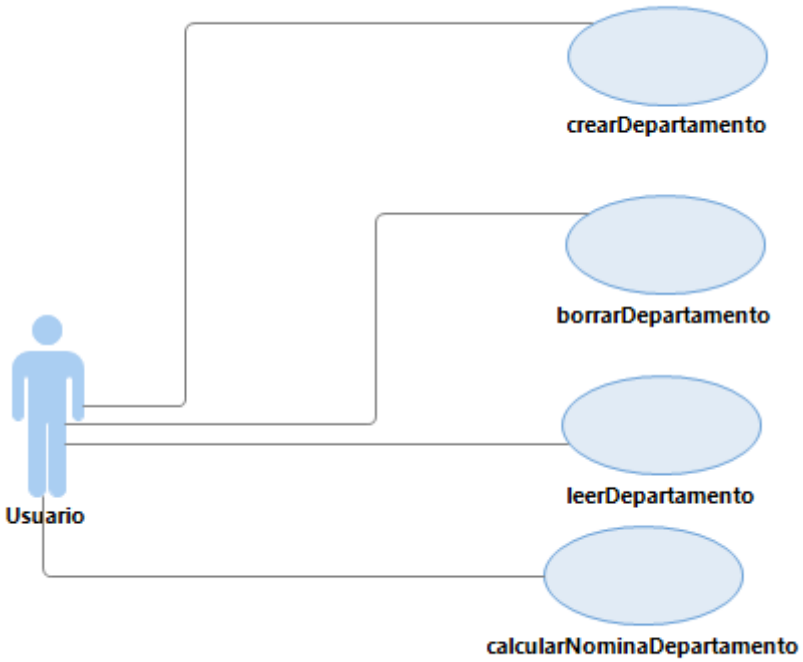
Efectos laterale	En caso de que los datos introducidos no sean correctos o contengan caracteres no válidos se mostrará un



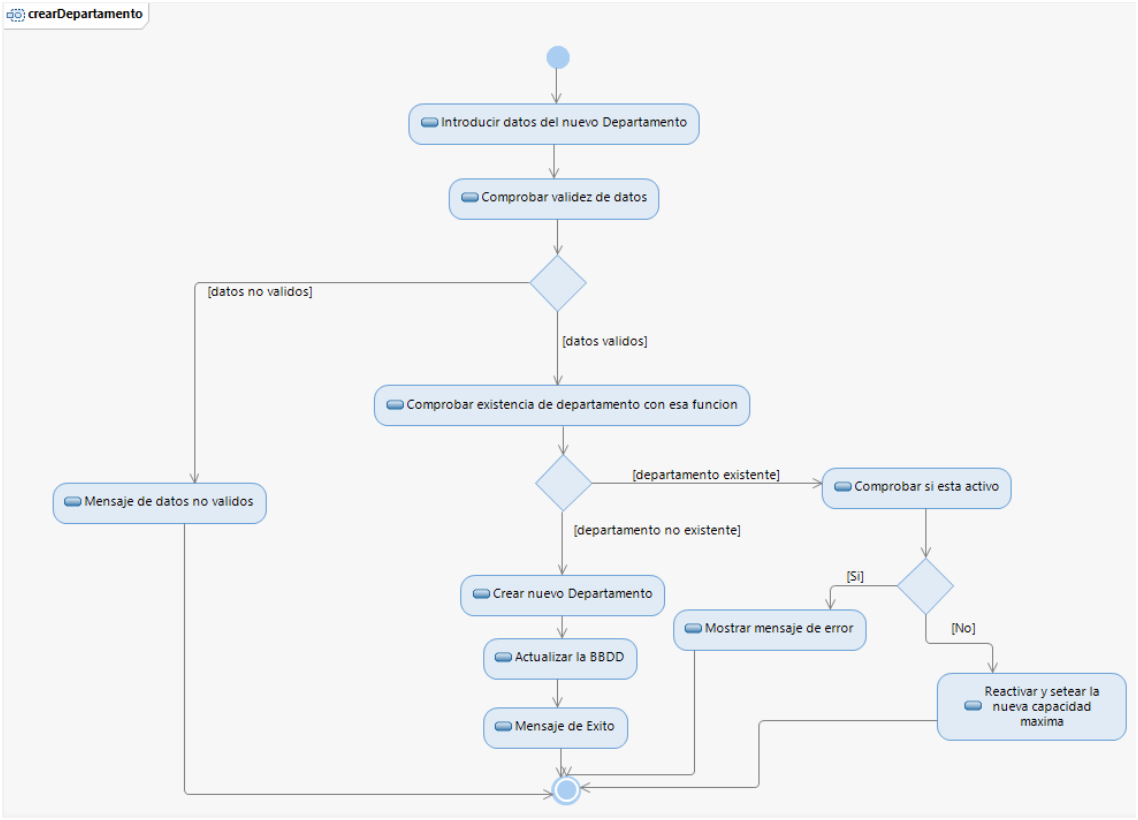
Requisito 4.7	Mostrar Empleados de Competencia
Descrita	Ignacio Baena Kuroda
Prioridad	Alta
Estabilidad	Alta.
Descripción	Función que muestra los empleados poseedores de una determinada compentencia
Entradas	ID de competencia.
Salidas	Lista de empleados
Origen	Interfaz de usuario – Administrador.
Destino	Sistema.
Necesita	Base de datos de competencias. Base de datos de
Precondición	No existan IDs duplicados de competencia ni empleados
Pos condició	Una empleado completamente actualizado.
Efectos laterale	En caso de que los datos introducidos no sean correctos o contengan caracteres no válidos se mostrará un



Módulo Departamento

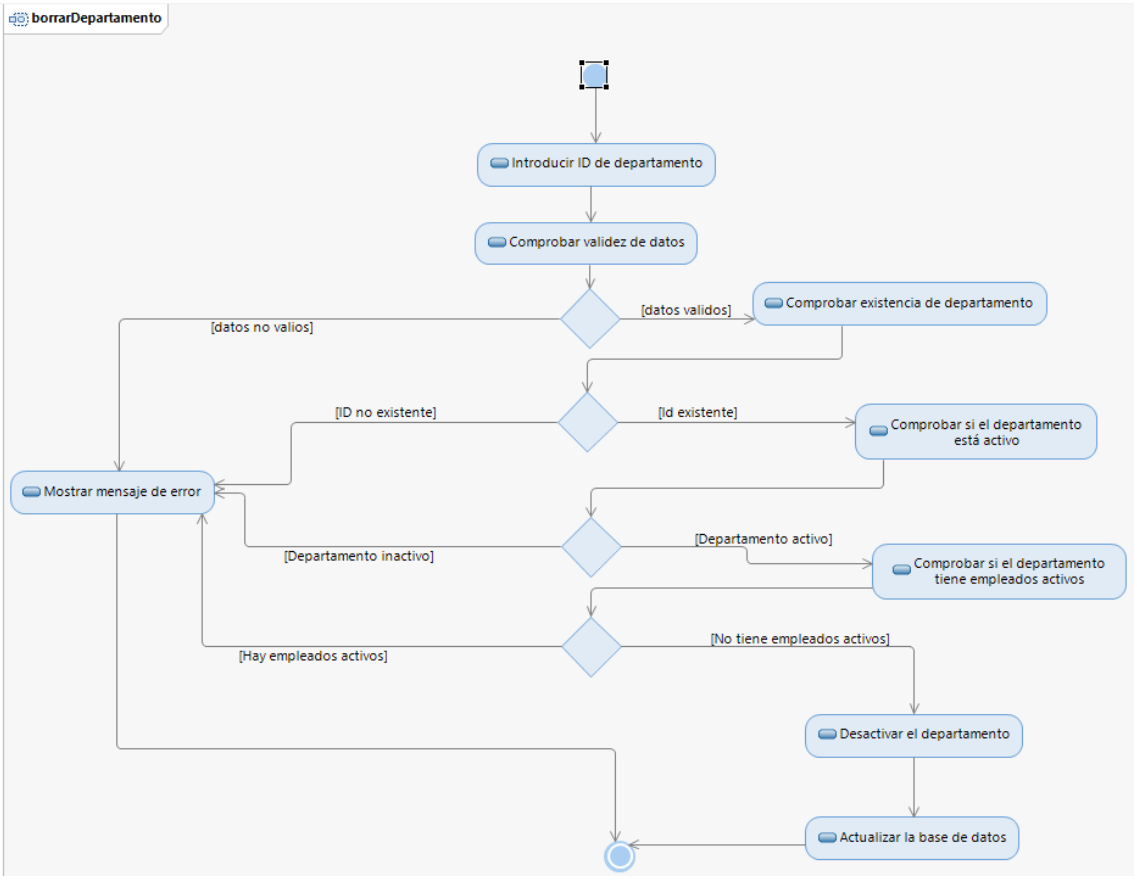


Requisito 5.1	Crear Departamento
Descrita	Cristóbal Saraiba Torres
Prioridad	Alta.
Estabilidad	Alta.
Descripción	Función que añade un nuevo departamento a la base de datos asignándole un ID.
Entradas	Datos del departamento.
Salidas	Un departamento con todos los datos introducidos.
Origen	Interfaz de usuario – Administrador.
Destino	Sistema.
Necesita	Base de datos de departamento.
Precondición	No exista departamento con el mismo nombre.
Pos condició	Un departamento con todos sus datos.
Efectos laterale	En caso de que los datos introducidos no sean correctos o contengan caracteres no válidos se mostrará un



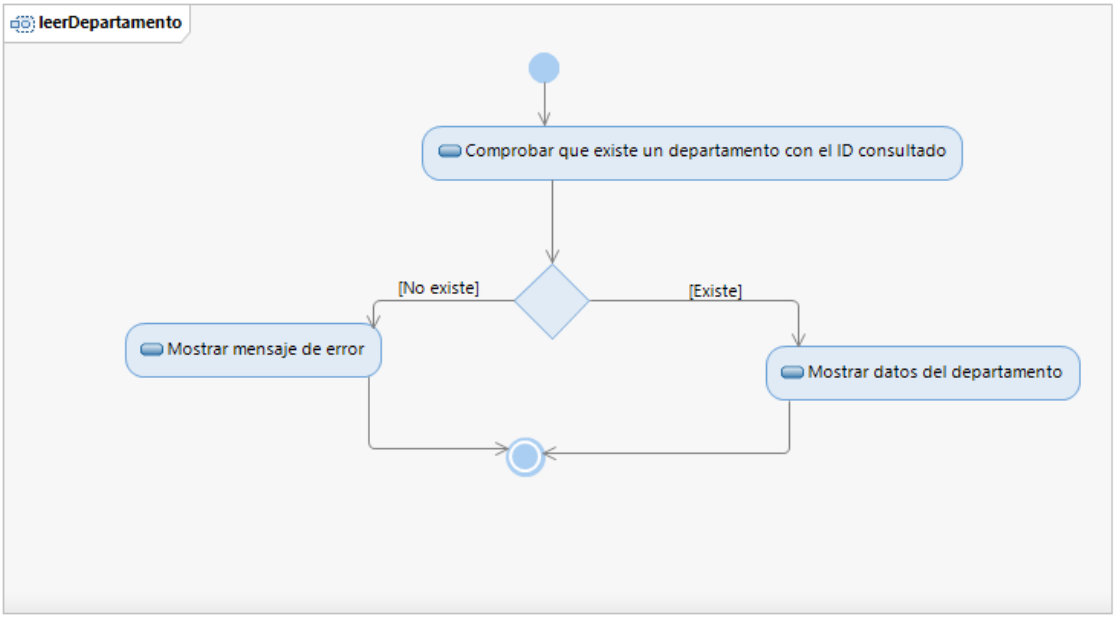
Requisito 5.2	Eliminar Departamento
Descrita por	Cristóbal Saraiba Torres
Prioridad	Alta

Estabilidad	Alta.
Descripción	Función elimina un departamento de la base de datos
Entradas	El ID del departamento
Salidas	Mensaje con la confirmación de que el departamento ha sido dado de baja
Origen	Interfaz de usuario – Administrador.
Destino	Sistema.
Necesita	Base de datos de departamento.
Precondición	No existen IDs duplicados en la base de datos.
Pos condició	Si existen empleados con dicho departamento, se le eliminará su conexión con el departamento.
Efectos laterale	En caso de que ya este eliminado o no se encuentre mostrará un mensaje de error.

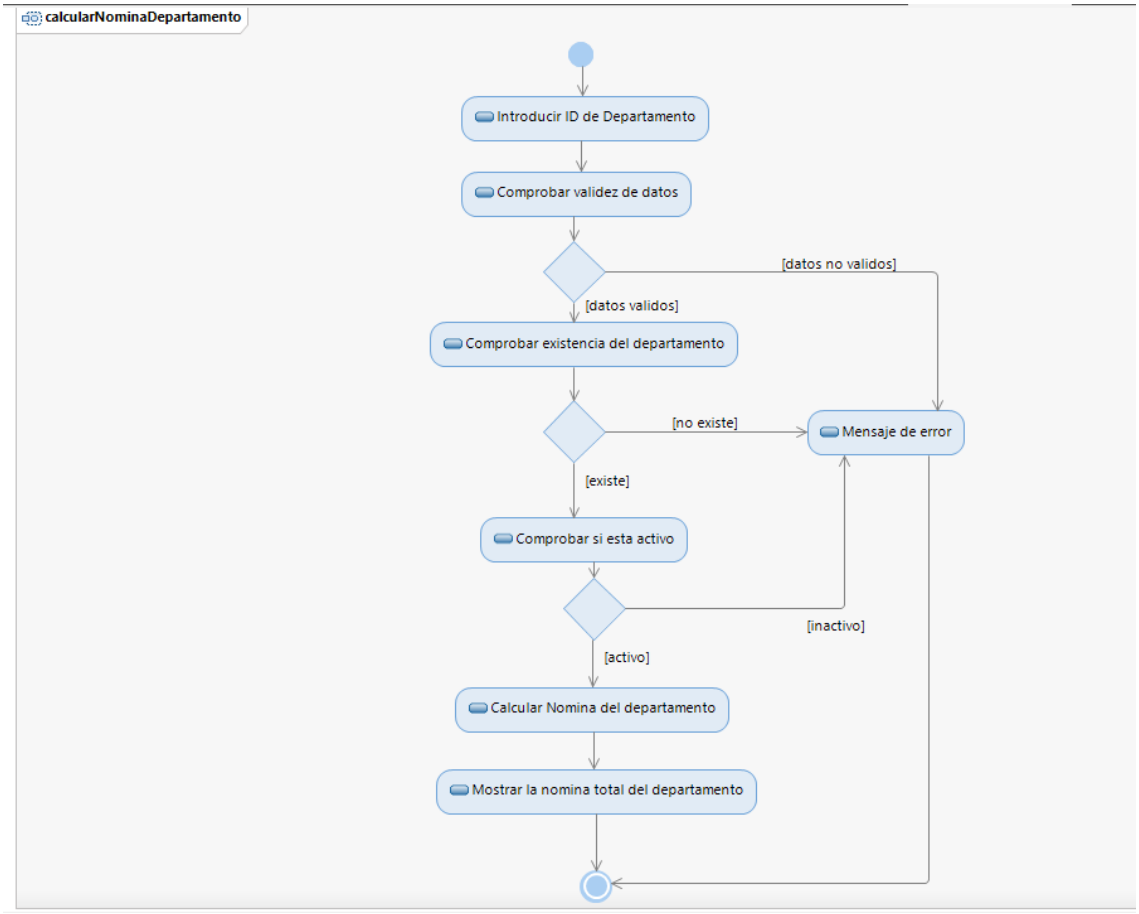


Requisito 5.4	Leer Departamento
Descrita por	Cristóbal Saraiba Torres
Prioridad	Media.
Estabilidad	Alta.
Descripción	Función que muestra por pantalla la información correspondiente a un departamento.
Entradas	El ID de la departamento.
Salidas	Mensaje con la información específica de cada

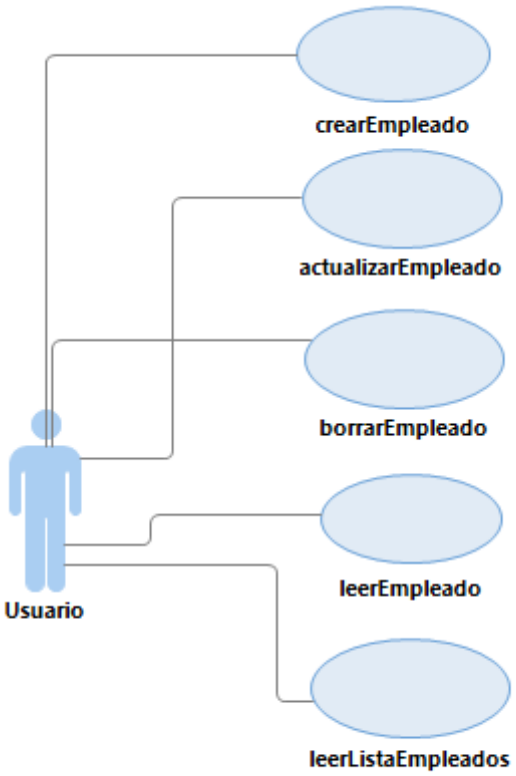
Origen	Interfaz de usuario – Administrador.
Destino	Sistema.
Necesita	Base de datos de departamento.
Precondición	No existen IDs duplicados en la base de datos.
Pos condició	No se modifica la base de datos.
Efectos laterale	En caso de que no se encuentre el departamento, se mostrará un mensaje de error.



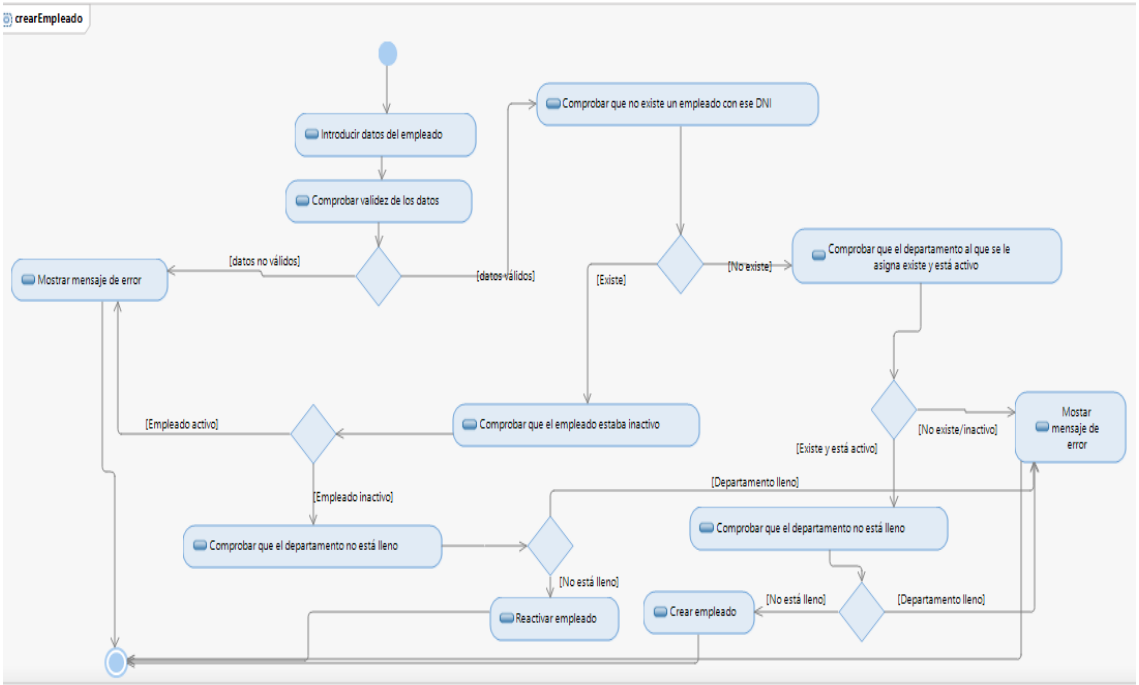
Requisito 5.6	Calcular Nóminas Departamento
Descrita	Cristóbal Saraiba Torres
Prioridad	Media.
Estabilidad	Alta.
Descripción	Función que calcula la suma de todas las nóminas de un departamento.
Entradas	ID del departamento.
Salidas	La suma de todas las nóminas.
Origen	Interfaz de usuario – Administrador.
Destino	Sistema.
Necesita	Base de datos de departamento. Base de datos de
Precondición	No existan IDs duplicados de un departamento ni de empleados.
Pos condició	No se modifica la base de datos.
Efectos laterale	En caso de que los datos introducidos no sean correctos o contengan caracteres no válidos se mostrará un


calcularNominaDepartamento
Introducir ID de Departamento
Comprobar validez de datos
[datos no validos]
[datos validos]
Comprobar existencia del departamento
[no existe]
Mensaje de error
[existe]
Comprobar si esta activo
[inactivo]
[activo]
Calcular Nomina del departamento
Mostrar la nomina total del departamento

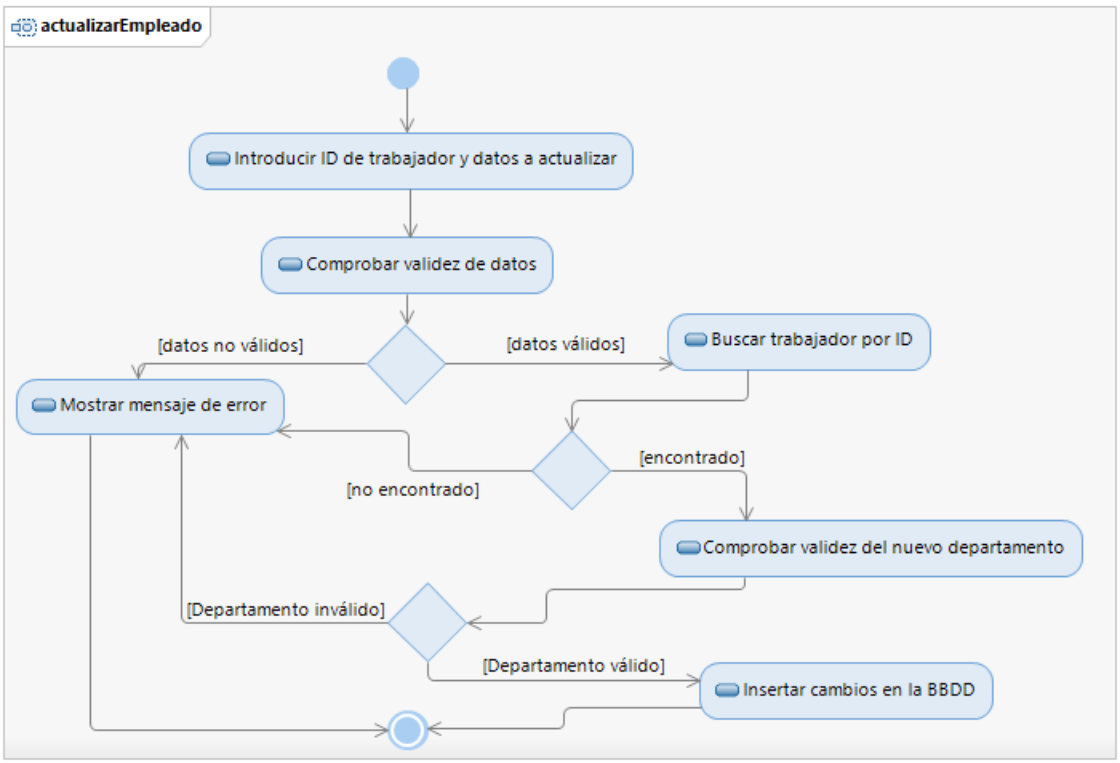
Módulo Empleado



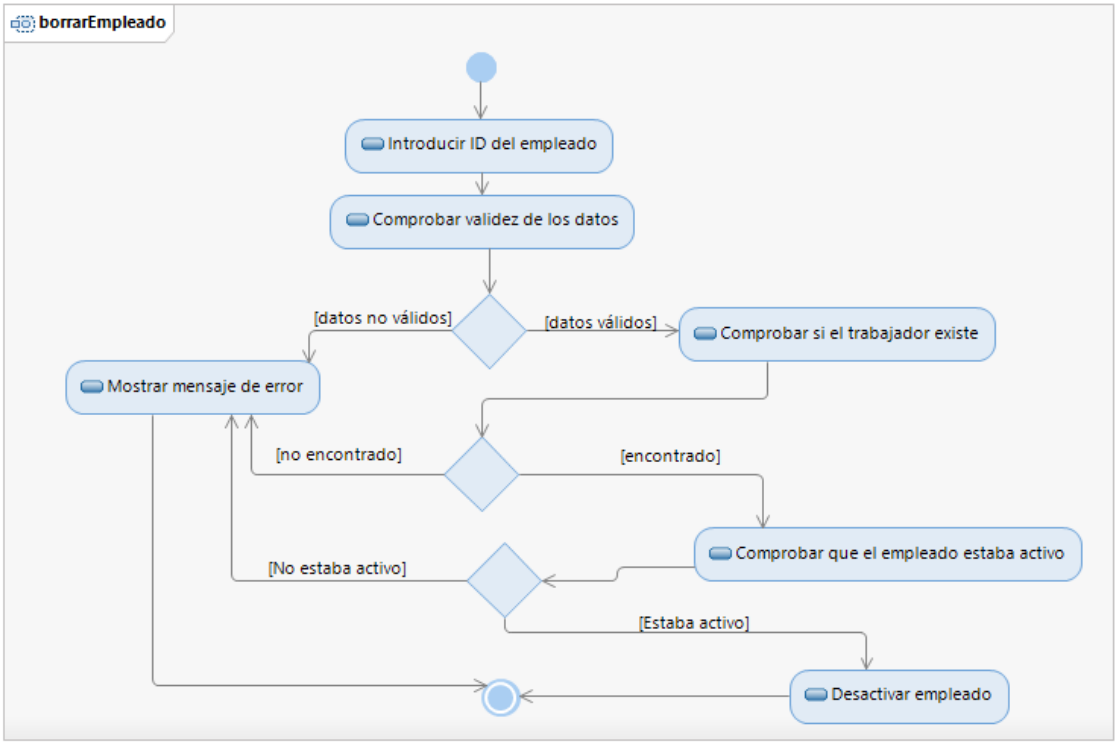
Requisitos 6.1	**altaEmpleado**
Descrita por	Ignacio Baena Kuroda
Prioridad	Alta
Estabilidad	Alta
Descripción	Activa al empleado (sino existe lo crea)
Actor	Recursos Humanos
Entradas	Datos del empleado (DNI, Horario, Telefono, Actividad, tipo)
Salidas	Mensaje de confirmación
Origen	Interfaz de usuario - Empleados
Destino	Base de datos de empleados
Necesita	Acceso a la base de datos de empleados
Precondición	No existe el DNI del empleado en la base de datos o se encuentra inactivo
Postcondición	No existen DNI's repetidos en la base de datos. Existe el nuevo empleado en la base de datos.
Efectos laterales	Si se encuentra el DNI en la base de datos devuelve un mensaje de error y cancela la operación



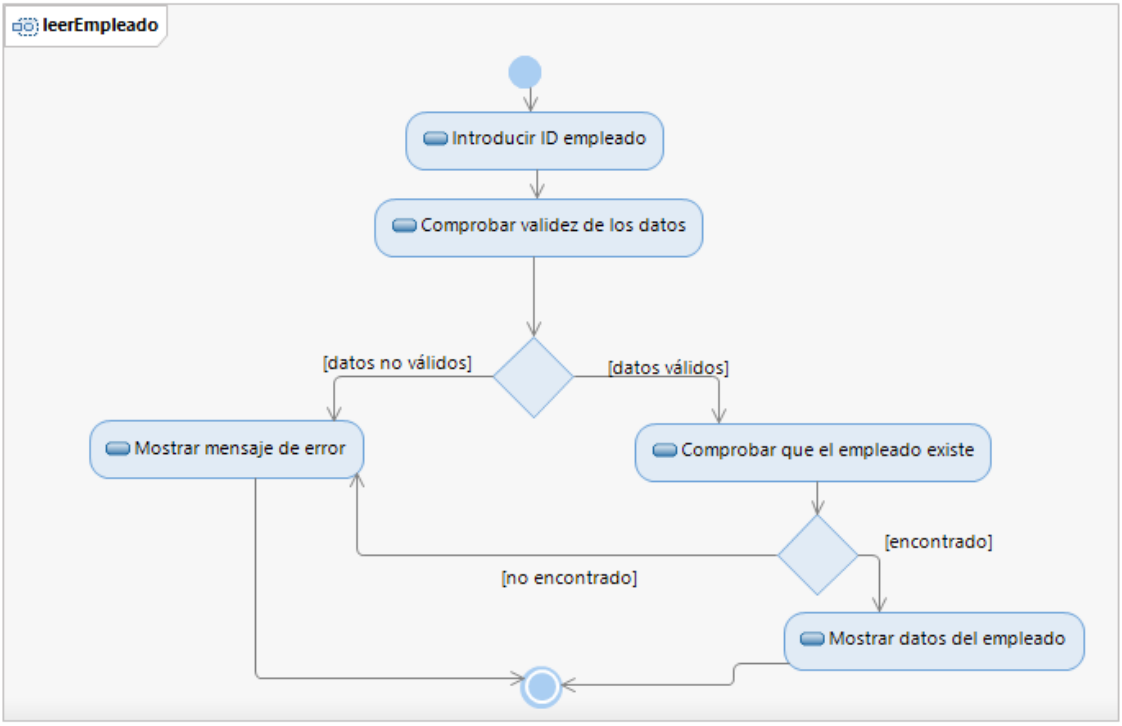
Requisitos 6.4	**actualizarEmpleado**
Descrita por	Ignacio Baena Kuroda
Prioridad	Alta
Estabilidad	Alta
Descripción	Modifica los datos de un empleado
Actor	Recursos Humanos
Entradas	ID de un empleado
Salidas	Mensaje de confirmación
Origen	Interfaz de usuario – Empleados
Destino	Base de datos de empleados
Necesita	Acceso a la base de datos de empleados
Precondición	No existen ID's repetidos en la base de datos
Postcondición	No existen ID's repetidos en la base de datos. No se crea ni elimina el empleado. Se actualizan los datos del empleado
Efectos laterales	Si no existe empleado con el ID introducido muestra un mensaje de error



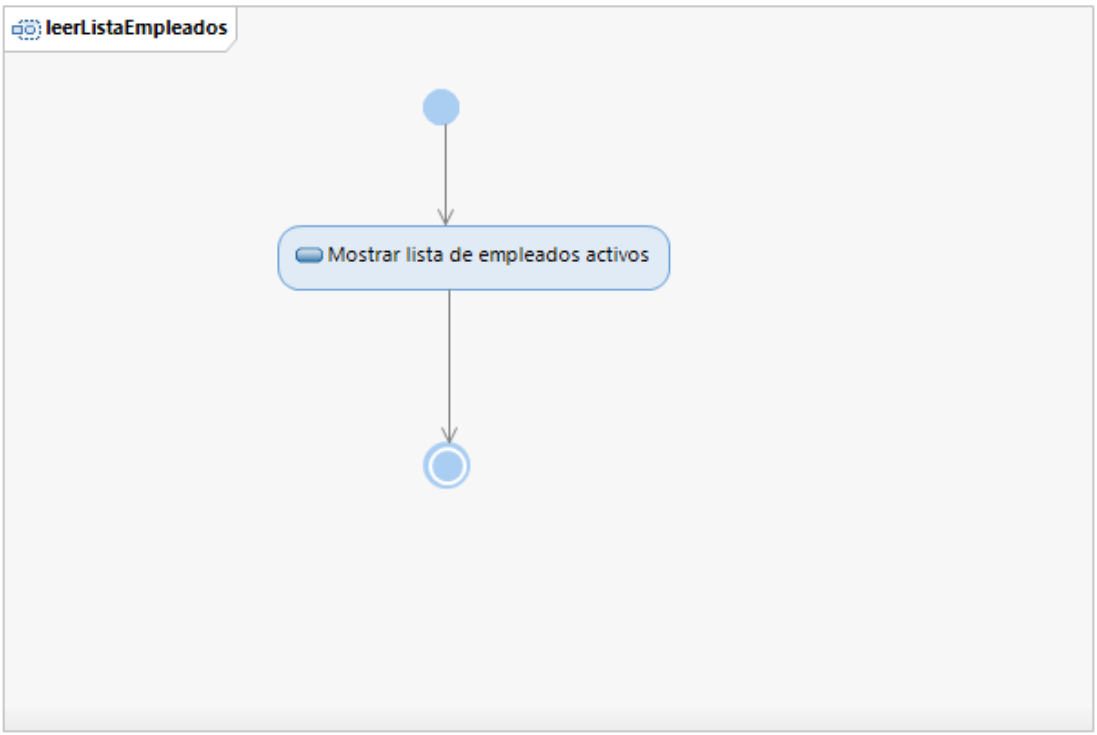
Requisitos 6.5	**bajaEmpleado**
Descrita por	Ignacio Baena Kuroda
Prioridad	Media
Estabilidad	Alta
Descripción	Elimina a un empleado de la base de datos
Actor	Recursos Humanos
Entradas	ID de un empleado
Salidas	Mensaje de confirmación
Origen	Interfaz de usuario – Empleados
Destino	Base de datos de empleados
Necesita	Acceso a la base de datos de empleados
Precondición	No existen ID's repetidos en la base de datos
Postcondición	No existen ID's repetidos en la base de datos. El empleado es eliminado de la base de datos
Efectos laterales	Si no encuentra el ID muestra un mensaje de error



Requisitos 6.2	**consultarEmpleado**
Descrita por	Ignacio Baena Kuroda
Prioridad	Media
Estabilidad	Alta
Descripción	Muestra los datos de un empleado por pantalla
Actor	Recursos Humanos
Entradas	ID del empleado
Salidas	Datos del empleado por pantalla
Origen	Interfaz de usuario – Empleados
Destino	Base de datos de empleados
Necesita	Acceso a la base de datos de empleados
Precondición	No existen ID's repetidos en la base de datos
Postcondición	No existen ID's repetidos en la base de datos. No se modifica la base de datos
Efectos laterales	Si no se encuentra el ID muestra un mensaje de error por pantalla



Requisitos 6.3	**consultarListaEmpleados**
Descrita por	Ignacio Baena Kuroda
Prioridad	Media
Estabilidad	Media
Descripción	Muestra una lista con los datos de todos los empleados en la base de datos
Actor	Recursos Humanos
Entradas	Ninguna
Salidas	Lista de datos de los empleados por pantalla
Origen	Interfaz de usuario – Empleados
Destino	Base de datos de empleados
Necesita	Acceso a la base de datos de empleados
Precondición	No existen ID's repetidos en la base de datos
Postcondición	No existen ID's repetidos en la base de datos. La base de datos no se modifica
Efectos laterales	Si la base de datos está vacía se muestra un mensaje indicándolo


leerListaEmpleados
Mostrar lista de empleados activos

3.3 Requisitos de Rendimiento.

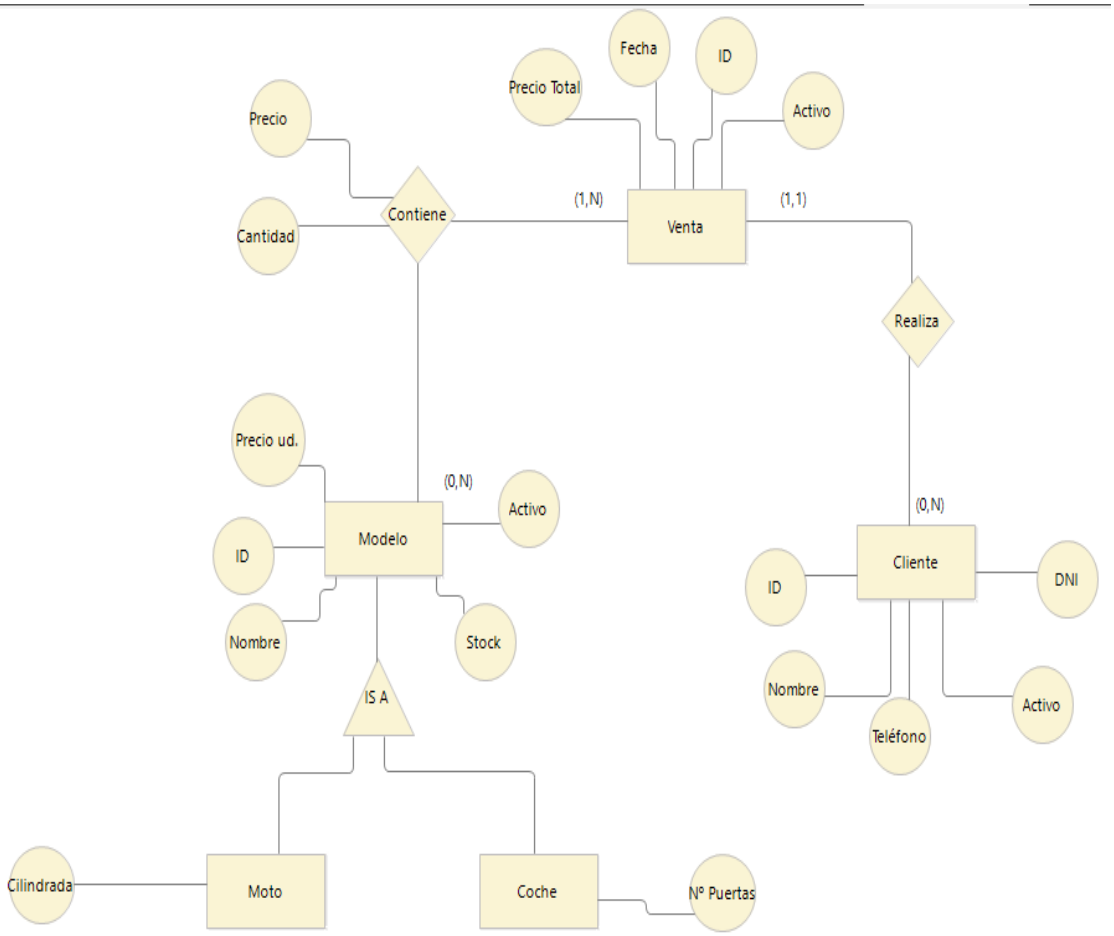
Al realizar un operación, el coste de uso de la CPU de no debe sobrepasar el 50 %.
El 90 % de las transacciones deben realizarse en menos de 1 minuto como máximo en cualquier caso.
El número de usuarios será limitado pero el sistema soportará la concurrencia de procesos simultáneos.
El uso de disco duro, CPU, porcentaje de rendimiento del CPU, memoria y la optimización de los recursos al máximo, ayudan al rendimiento del sistema.

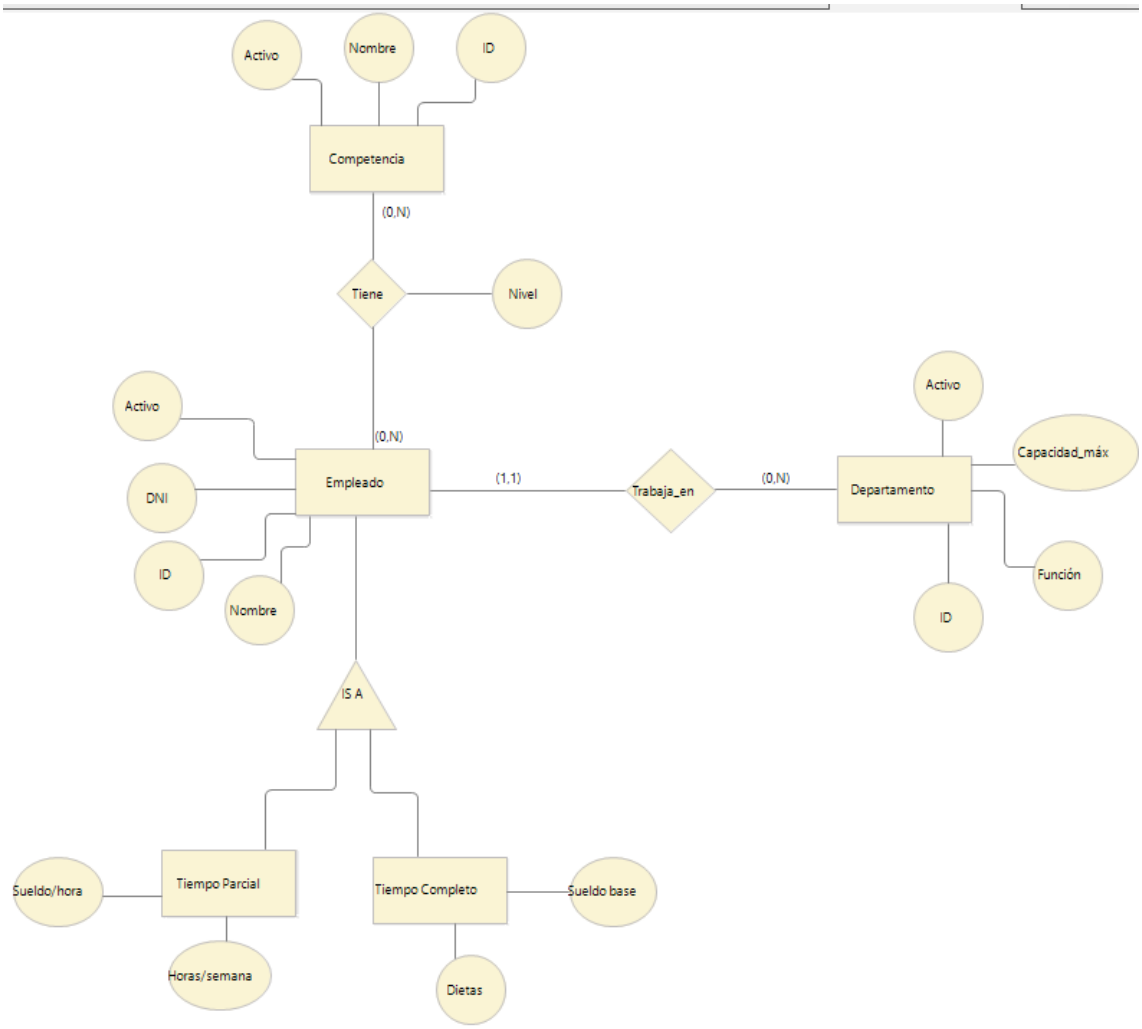
3.4 – Logical Database Requirements

3.4.1 Entidades de datos, relaciones y formato

El sistema debe manejar y mantener los datos de las 6 entidades del dominio (Venta, Modelo, Cliente, Empleado, Departamento y Competencia).

Todos los datos serán almacenados en una base de datos siguiendo el siguiente modelo:





Siendo las claves ID en el caso de Venta, Modelo, Departamento y Competencia y DNI en el caso de Cliente y Empleado.

Los ID deberán ser un número entero de una longitud máxima de 10 dígitos, mientras que los DNI serán un código alfanumérico con 8 dígitos y una letra.

El atributo "activo" será un booleano indicando que el objeto se encuentra disponible (no ha sido borrado), ya que los borrados son lógicos y no se realizarán sobre la base de datos.

Nota: Los atributos de entidades y relaciones podrán sufrir cambios durante el desarrollo del proyecto, pero la estructura del dominio no.

3.4.2 Capacidades de acceso y frecuencia de uso

Los datos de un subsistema no podrán ser accedidos desde el otro pues son independientes.

El sistema solo accederá a la información necesaria para cada operación limitando el acceso a datos innecesarios, de modo que el uso de la base de dato será constante para todas las entidades.

3.5 - Restricciones de diseño

La aplicación que se ha diseñado estará basada en el lenguaje de implementación Java (en su versión 8) y el lenguaje de diseño UML. Al utilizar Java como lenguaje de programación, hemos aplicado la POO (Programación orientada a objetos), aprovechando las características de dicho modelo como son la encapsulación, la abstracción, la ocultación, la herencia y el polimorfismo.

3.6 - Atributos del Sistema Software

- Fiabilidad: la introducción de datos incorrectos por parte del usuario estará vigilada mediante excepciones, comprobando que los datos que finalmente se guardan en la BBDD siempre serán fiables.

- Disponibilidad: al no tener un servidor web, la aplicación estará disponible en cualquier ordenador que se ejecute pero los datos se almacenarán en el disco duro interno del terminal.

- Seguridad: no se establece ningún tipo de seguridad para acceder a la aplicación.

- Mantenibilidad: tras la finalización del proyecto no se publicarán nuevas versiones.

- Portabilidad: se garantiza que el sistema es compatible con el sistema operativo Windows 10.