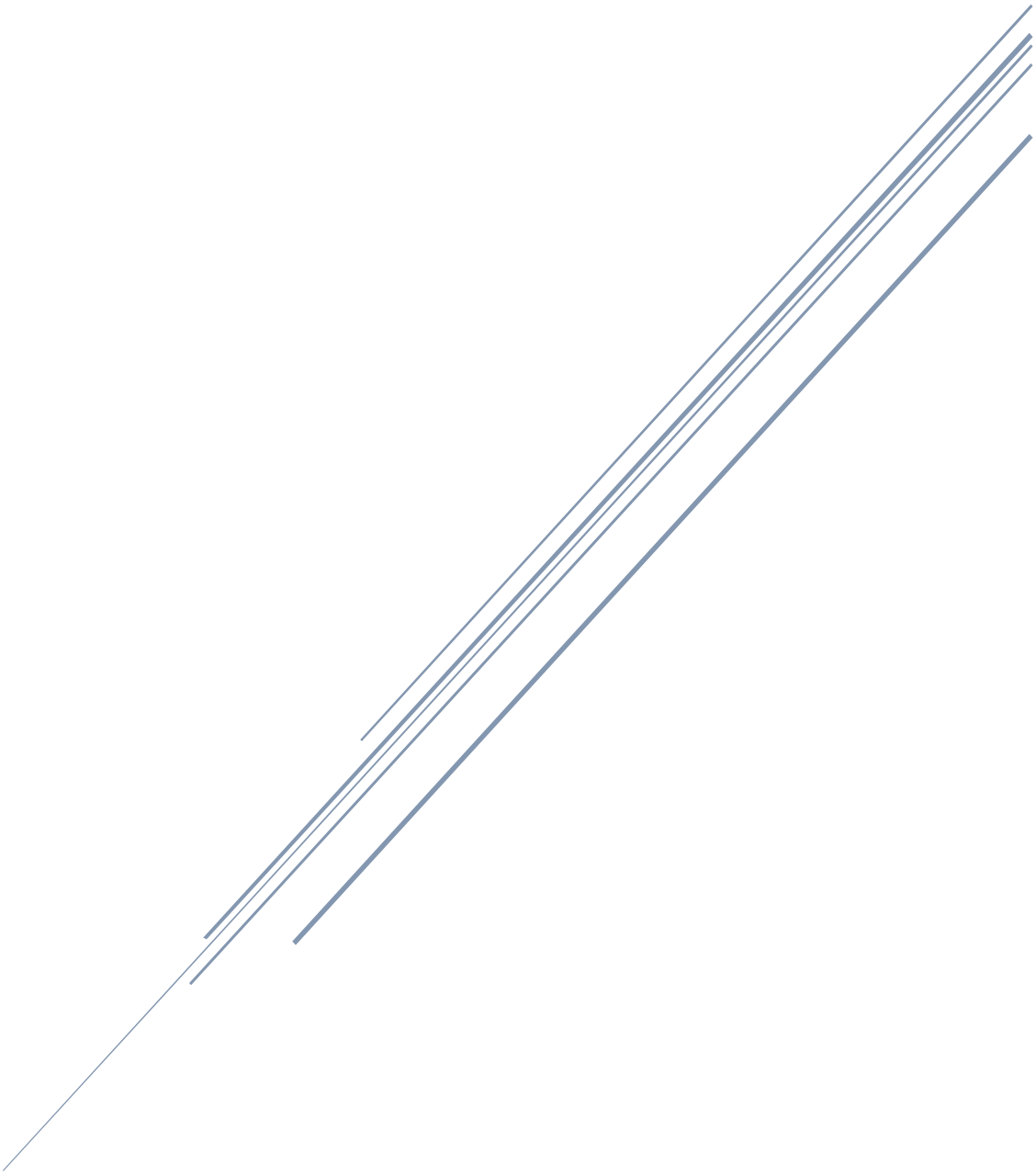
PORTAFOLIO

Tecnología Orientada a Objetos



Universidad de La Rioja, 2017-2018

Carlota Alti Palacios, Daniel Gómez Ochagavía y Samuel Ojeda Delgado

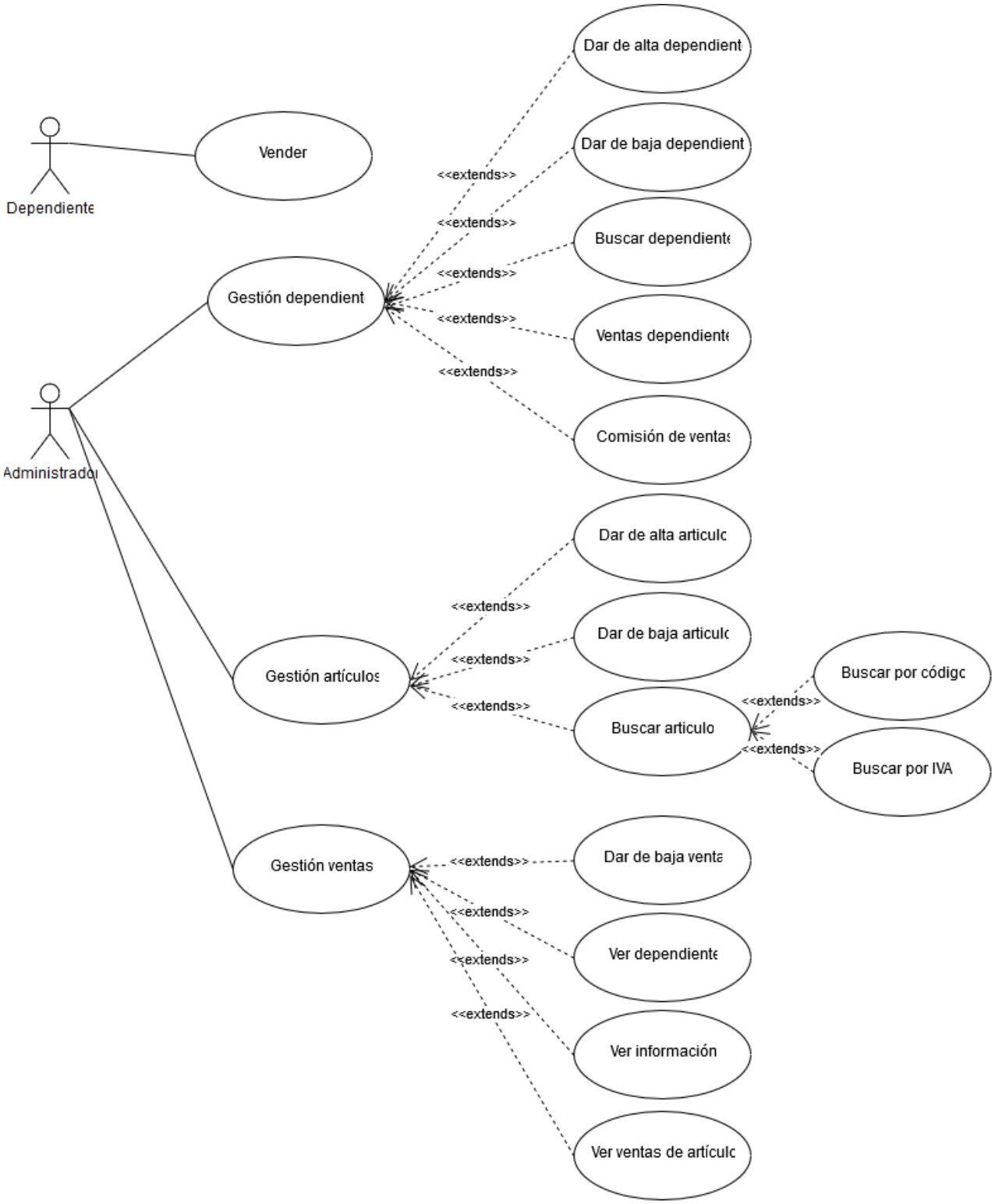
Índice

Requisitos iniciales

El sistema debe poder gestionar información de dependientes, artículos y ventas realizadas por los dependientes. La gestión la llevará a cabo un administrador.

Los dependientes tienen sus datos personales y se les asigna una comisión, los artículos tienen sus datos identificativos y de precios, y las ventas guardan el dependiente que la realizó, código, fecha e importe.

Modelo de Casos de Uso



Especificación de Casos de Uso

Dependiente

1. Vender

Nombre: Vender
Actor principal: Dependiente
Actor secundario: Sistema
Objetivo: Se introduce una nueva venta en el sistema
Actor principal: Dependiente
Actor secundario: Sistema
Precondición: Los datos introducidos deberán de ser válidos
Postcondición: La venta queda registrada en la base de datos
Descripción del proceso:

1. El dependiente introduce su identificador
2. El sistema crea una venta, dándole un código respecto al resto (que será único) y la fecha de ese momento.
3. El dependiente introducirá el código de los artículos de esa venta
4. El sistema obtendrá el precio de venta de cada uno, lo sumará y mostrará el importe por pantalla
5. El dependiente dirá si el método de pago es en tarjeta o al contado
6. El sistema, si el método introducido es con tarjeta, almacenará también el número de tarjeta. Y si el pago es al contado, guardará un 0.
7. Si el pago es al contado, el dependiente introducirá la cantidad aportada por el cliente.
8. El sistema mostrará el cambio por pantalla.

Flujo alternativo:

1.1 Si el dependiente introduce mal el identificador, se mostrará un mensaje y se le dejará volver a introducirlo

3.1 Si el dependiente no introduce artículos, se creará una venta vacía. Si el código de algún artículo no existe, se mostrará un mensaje y se dejará volver a introducirlo.

Administrador

1. Gestión de dependientes

Nombre: Gestión de dependientes
Actor principal: Administrador
Actor secundario: Sistema
Objetivo: El sistema muestra una interfaz con los datos de dependientes y las opciones posibles de gestión.
Evento de activación: El dependiente selecciona gestionar dependientes
Precondición: ninguna
Escenario principal:

- El sistema muestra la interfaz

Postcondición: ninguna

1.1. Dar de alta dependiente

Nombre: Dar de alta dependiente
Actor principal: Administrador
Actor secundario: Sistema
Objetivo: Añadir un dependiente al sistema
Evento de activación: El administrador selecciona dar de alta
Precondición: Ninguna

Escenario principal:

- El administrador suministra los datos del nuevo dependiente
- El sistema los registra en la base de datos

Postcondición: Si el dependiente estaba ya en el sistema no hay cambios, si no estaba se introduce en el sistema.

1.2. Dar de baja dependiente

Nombre: Dar de baja dependiente
Actor principal: Administrador
Actor secundario: Sistema
Objetivo: Dar de baja un dependiente del sistema
Evento de activación: El administrador selecciona dar de baja
Precondición: El dependiente ha de estar en la base de datos del sistema
Escenario principal:

- El administrador selecciona dar de baja un dependiente
- El servidor solicita confirmación
- El administrador provee la confirmación
- El sistema elimina el vendedor y sus ventas relacionadas

Postcondición: No hay dependientes ni ventas con el ID de dependiente de aquel que ha sido dado de baja.

1.3. Buscar dependiente <<abstract>>

Nombre: Buscar vendedor
Actor principal: Administrador
Actor secundario: Sistema
Objetivo: Muestra los datos del dependiente con el NSS provisto
Evento de activación: El administrador selecciona buscar por NSS
Precondición: Se provee un NSS con formato adecuado y el dependiente está en el sistema
Escenario principal:

- El administrador selecciona buscar por NSS introduciéndolo
- El sistema devuelve los datos del vendedor correspondiente

Postcondición: Se devuelve el vendedor con NSS igual al provisto

1.4. Ventas dependiente

Nombre: Ventas dependiente
Actor principal: Administrador
Actor secundario: Sistema
Objetivo: Obtener el volumen de ventas del último mes de un dependiente
Evento de activación: El administrador selecciona obtener el volumen de ventas por dependiente
Precondición: Se selecciona un dependiente perteneciente al sistema
Escenario principal:

- El administrador suministra los datos del vendedor en cuestión (seleccionado a partir de una lista, por ejemplo)
- El sistema muestra el volumen de ventas del último mes para ese dependiente

Postcondición: se devuelve el volumen de ventas del último mes para el dependiente solicitado

1.5. Comisión de ventas

Nombre: Obtener comisión de ventas
Actor principal: Administrador
Actor secundario: Sistema
Objetivo: Obtener comisión de ventas de un dependiente
Evento de activación: El administrador selecciona obtener comisión de ventas
Precondición: Se selecciona un dependiente perteneciente al sistema
Escenario principal:

- El administrador suministra los datos del vendedor en cuestión (seleccionado a partir de una lista, por ejemplo)
- El sistema muestra la comisión para ese dependiente

Postcondición: se devuelve la comisión para el dependiente solicitado

2. Gestión de artículos

Nombre: Gestión de artículos
Actor principal: Administrador
Actor secundario: Sistema
Objetivo: El sistema muestra una interfaz con los datos de artículos y las opciones posibles de gestión.
Evento de activación: El dependiente selecciona gestionar artículos
Precondición: ninguna
Escenario principal:

- El sistema muestra la interfaz

Postcondición: ninguna

2.1. Dar de alta artículo

Nombre: Dar alta artículo.
Actor principal: Sistema.
Actores secundarios: Empleado.
Objetivo: El empleado dará de alta un articulo
Evento de activación: El empleado selecciona en el sistema dar de alta un artículo.
Precondición: Los datos introducidos deben ser datos correctos que se especificarán más adelante.
Escenario principal:

- El empleado introduce una descripción del artículo, el coste de fábrica y su IVA.
- El sistema guarda en la base de datos la información, asignándole al producto un ID.
- El sistema muestra al usuario el ID asignado a los datos introducidos.

Postcondición: El articulo está dado de alta en la base de datos y el empleado conoce su ID asignado a dicho artículo.

2.2. Dar de baja artículo

Nombre: Dar de baja un artículo.
Actor principal: Sistema.
Actores secundarios: Empleado.
Objetivo: El empleado quiere dar de baja un artículo, quedando en la base de datos registrado esta baja, pero sin eliminar la información del artículo.
Evento de activación: El empleado selecciona en el sistema dar de baja un artículo.
Precondición: El ID del artículo tendrá que ser correcto, además de que dicho artículo tendrá que estar dado de alta.

Escenario principal:

- El empleado introduce el ID para dar de baja el artículo.
- El sistema da de baja el artículo en la base de datos.
- El sistema informa al empleado que la operación ha concluido.

Postcondición: El articulo queda dado de baja en la base de datos.

2.3. Buscar artículo <<abstract>>

Nombre: Buscar artículo

Objetivo: Agrupa las distintas opciones de búsqueda entre artículos

2.3.1. Buscar artículo por código

Nombre: Buscar artículo por código.

Actor principal: Sistema.

Actores secundarios: Empleado.

Objetivo: El empleado conocerá la información del artículo que deseé.

Evento de activación: El empleado selecciona buscar artículo en la aplicación.

Precondición: El ID introducido debe ser correcto.

Escenario principal:

- El empleado introduce el ID del artículo que desea buscar.
- El sistema obtendrá los datos en la base de datos.
- El sistema muestra al empleado los datos del artículo en cuestión.

Postcondición: El empleado conoce la información del artículo deseado.

2.3.2. Buscar artículos por IVA

Nombre: Obtener artículos con un IVA.

Actor principal: Sistema.

Actores secundarios: Empleado.

Objetivo: EL empleado conocerá los artículos con el IVA especificado.

Evento de activación: El empleado selecciona en el sistema buscar por IVA

Precondición: El IVA especificado por el empleado debe ser o 20, o 10 o 4 %.

Escenario principal:

- El empleado selecciona uno de los IVA s que el sistema trata.
- El sistema busca en la base de datos los artículos con el IVA determinado.
- El sistema muestra al empleado la información de los artículos con el IVA especificado.

Postcondición: EL empleado conoce los artículos con el IVA especificado.

3. Gestión de ventas

Nombre: Gestión de ventas

Objetivo: Agrupa las distintas opciones de búsqueda entre artículos

3.1. Dar de baja venta

Nombre del caso de uso: Dar de baja venta

Actor principal: Administrador

Actor secundario: Sistema

Precondición: El código de la venta introducido debe ser válido

Postcondición: Elimina una venta del sistema

Descripción del proceso:

1. El administrador introduce el código de la venta
2. El sistema busca la venta y muestra un diálogo de confirmación
3. El administrador acepta

4. El sistema efectúa la eliminación

Flujo alternativo:

1.1 Si el código introducido no existe, se mostrará un mensaje

3.2. Ver dependiente

Nombre del caso de uso: Obtener información del dependiente de una venta
Actor principal: Administrador
Actor secundario: Sistema
Precondición: El código de venta introducido debe ser válido
Postcondición: Devuelve los datos del dependiente que ha realizado esa venta
Descripción del proceso:

1. El administrador introduce el código de la venta
2. El sistema busca la venta y muestra por pantalla los datos del dependiente que la ha realizado

Flujo alternativo:

1.1 Si el código introducido no existe, el sistema mostrará un mensaje y dejará volver a introducir otro.

3.3. Ver información de venta

Nombre del caso de uso: Ver información de venta
Actor principal: Administrador
Actor secundario: Sistema
Precondición: El código de la venta introducido debe ser válido
Postcondición: Devuelve los datos de la venta que tiene el código que ha sido introducido
Descripción del proceso:

5. El administrador introduce el código de la venta
6. El sistema busca la venta y muestra por pantalla los datos de la venta con ese código

Flujo alternativo:

1.1 Si el código introducido no existe, se mostrará un mensaje y dejará volver a introducir otro

3.4. Obtener ventas de un dependiente

Nombre del caso de uso: Obtener ventas de un dependiente
Actor principal: Administrador
Precondición: El identificador de dependiente debe de ser válido
Postcondición: Devuelve una lista ordenada por fecha, con todas las ventas de un dependiente
Descripción del proceso:

1. El administrador introduce el identificador de un dependiente
2. El sistema busca las ventas realizadas por ese dependiente
3. El sistema muestra por pantalla las ventas del dependiente ordenadas

Flujo alternativo:

2.1 Si el dependiente no ha realizado ninguna venta, se mostrará por pantalla un mensaje advirtiéndolo

3.5. Ver ventas de un artículo

Nombre del caso de uso: Ver ventas de un artículo
Actor principal: Administrador
Precondición: El identificador de artículo introducido debe de ser válido

Postcondición: Devuelve una lista con todas las ventas en las que se ha vendido el artículo introducido

Descripción del proceso:

1. El administrador introduce el identificador de un artículo
2. El sistema busca las ventas que contienen ese artículo
3. El sistema muestra por pantalla las ventas con ese artículo

Flujo alternativo:

2.1 Si el artículo no se ha vendido en ninguna venta, se mostrará por pantalla un mensaje advirtiéndolo

Listado completo de requisitos

El ID introducido en la función “Dar de baja un articulo” debe existir en la base de datos.

Los datos del articulo introducidos en “Dar alta articulo” deben ser correctos, a saber:

- Descripción: Debe ser una cadena de caracteres con tamaño no superior a 255 caracteres.
- Coste fabrica: Debe ser un real positivo.
- IVA: Será 21, 10 o 4. El sistema interpretará que es un valor en %.
- El ID será asignado por el sistema automáticamente. Será una cadena de caracteres de 8 dígitos.
- La descripción será una cadena de caracteres con tamaño no superior a 255 caracteres.

En la operación "Obtener artículos con un IVA", el usuario tendrá que introducir uno de los tres valores aceptables para el valor IVA, a saber, elegirá entre 20, 10 o 4.

El NSS introducido en la función “Dar de alta dependiente” debe no estar registrado ya en la base de datos para poder introducir el nuevo dependiente, y debe estar registrado para “Dar de baja dependiente”, “Buscar por NSS”, “Ver ventas” y “Obtener comisión de ventas”.

Los datos introducidos en “Dar de alta dependiente” deben cumplir:

- NSS: ha de ser una cadena de caracteres. Lo consideramos único e inalterable una vez asignado. Cumple el siguiente formato:
- Nombre y apellidos: cadenas de caracteres modificables y no vacías.
- Comisión: número entero como un tanto por ciento modificable, positivo o cero y no nulo. Puede ser distinto para cada dependiente y se aplica a todas sus ventas por igual.

El identificador de dependiente introducido en “Crear Venta” y en “Obtener ventas de un dependiente” debería existir en la base de datos, así como los identificadores de los artículos introducidos en “Crear Venta”.

Los datos de la venta introducidos en “Crear Venta” deben ser correctos, adaptándose a:

- Código (identificador) de venta: será asignado por el sistema automáticamente. Será una cadena de caracteres de 8 dígitos.
- Fecha: será la fecha del momento de creación de la venta, que será de tipo DateTime.
- Líneas de venta: lista de líneas de venta. Cada línea de venta contendrá el artículo comprado (solo puede aparecer el mismo artículo en una línea de venta por cada venta) y la cantidad.
- Importe: valor calculado como la suma de importes de artículos y sus cantidades.

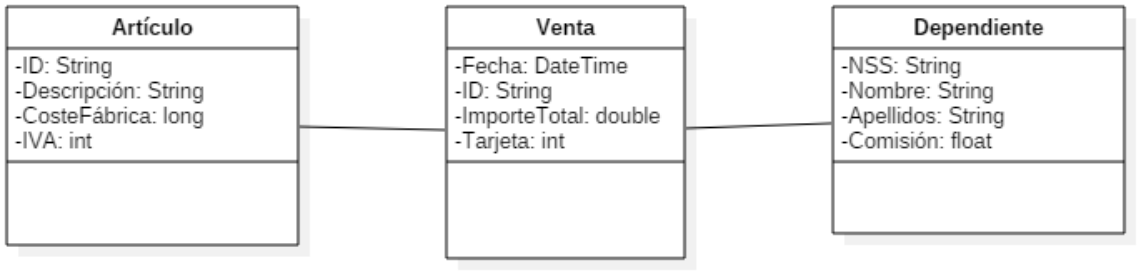
- Tarjeta de crédito: deberá ser un número entero positivo de 16 dígitos
- Contado: deberá ser un número positivo. (Se almacenará en la base de datos con un cero, para saber que no ha pagado con tarjeta de crédito)
- Dependiente: su identificador debe ser válido según el catálogo de requisitos de Dependientes.

Una línea de venta debe cumplir que contiene un artículo de la base de datos y la cantidad es un entero mayor que 0.

El identificador de venta introducido en “Obtener información del dependiente de una venta” y en “Obtener datos de una venta”, deberá existir en la base de datos. El identificador de artículo introducido en “Obtener ventas de un artículo” deberá existir en la base de datos.

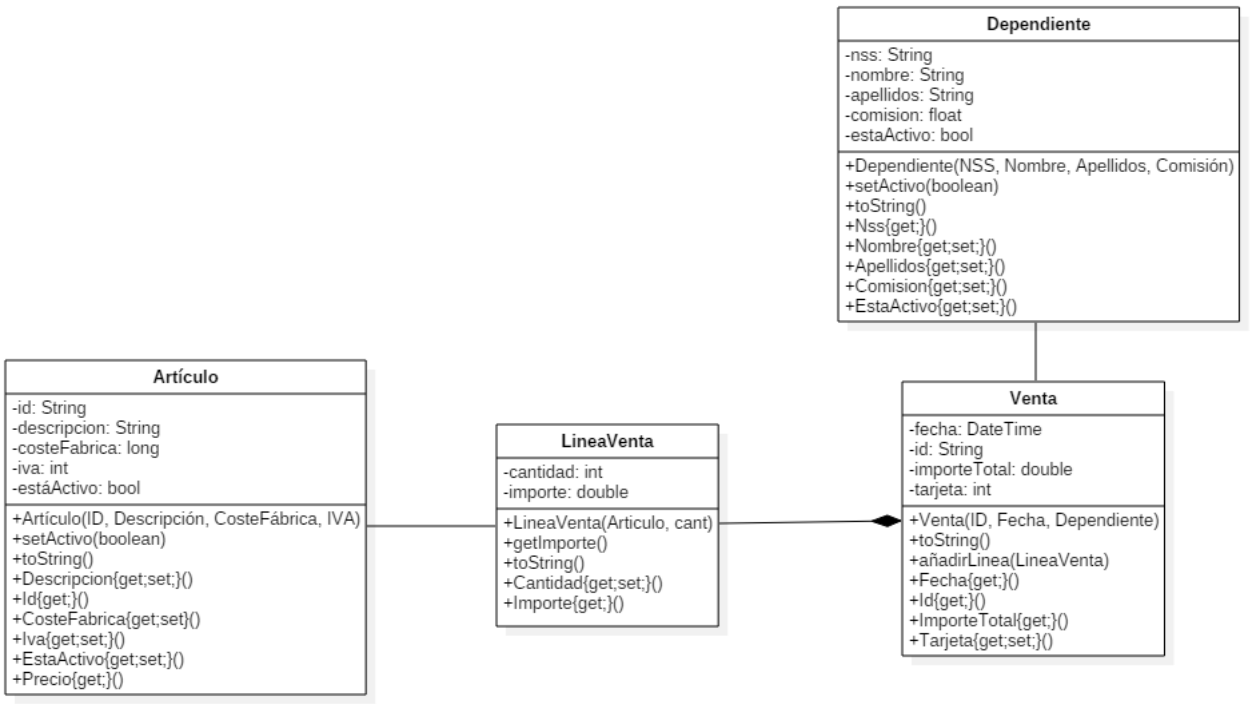
Modelo conceptual

Nuestro sistema tendrá una estructura basada en estas tres clases.



Modelo de clases

Una vez analizado el modelo anterior, hemos añadido las clases que consideramos oportunas para hacer nuestro modelo de negocio factible. Para ello, introducimos una línea de venta que representa un artículo y su cantidad en una venta.



Interfaz pública de la lógica de negocio

Hemos analizado el sistema y hemos separado en tres clases los servicios que ofrecerá nuestro sistema a partir del modelo de negocio creado anteriormente. Estas tienen por interfaces:

ServiciosArticulo:

- bool anadirArticulo(Articulo a)
- bool modificarArticulo(Articulo original, Articulo nuevo)
- bool modificarArticulo(String clave, String nombre, float coste, int iva)
- bool borrarArticulo(Articulo a)
- Articulo getArticulo(Articulo a)
- Articulo getArticulo(String a)
- List<Articulo> getTodosArticulos()
- bool existeArticulo(Articulo v)
- bool existeArticulo(String v)
- List<Articulo> getArticulosConIva(int iva)

SeviciosDependiente:

- bool anadirDependiente(Dependiente d)
- bool modificarDependiente(Dependiente original, Dependiente nuevo)
- bool modificarDependiente(String clave, String nombre, String apellidos, int comision)
- bool borrarDependiente(Dependiente d)
- bool borrarDependiente(String d)
- bool existeDependiente(string clave)
- bool existeDependiente(Dependiente d)
- Dependiente getDependiente(Dependiente d)
- Dependiente getDependiente(String clave)
- List<string> getNSS(Dependiente d)
- double calcularComision(Dependiente d, DateTime date)
- List<Venta> obtenerVentasMes(Dependiente d, DateTime fecha)
- List<Dependiente> getDependientesTienda()

ServiciosVenta

- bool anadirVenta(Venta v)
- Venta getVenta(Venta v)
- Venta getVenta(String v)
- List<Venta> getVentasDeArticulo(Articulo a)
- List<Venta> getVentasDeArticulo(String a)
- Dependiente getDependienteDeVenta(Venta v)
- Dependiente getDependienteDeVenta(String a)
- bool borrarVenta(Venta v)
- bool borrarVenta(String v)
- bool existeVenta(Venta v)
- bool existeVenta(String id)
- List<Venta> getVentas()

Repositorio de GitHub

Para agilizar la tarea hemos utilizado una plataforma de versionado para realizar el proyecto:

https://github.com/sod1497/ProyectoTOO