Universidad Rafael Landívar
Facultad de Ingeniería
Análisis y Diseño II
Inge. Dhaby Eugenio Xiloj Curruchiche

Tema:

Informe de Proyecto Final Empresa

“RESTAURANTE DE COMIDA TÍPICA “CALLE REAL””

- Alex Omar Tzul Tax 1587217
- Elmer Gustavo Pú Tzunux 1535017
- Dominga Del Rosario Gómez Sac. 1643417
- Fredy Mat Xhun Mateo Juan 1571617

Repositorio:
https://github.com/elmergustavo/SistemaPuntoVenta

Quetzaltenango, 12/07/ 2021

INTRODUCCIÓN

En las empresas y microempresas poseen necesidades ya que a pasar el tiempo se van ampliando por lo que ya no les da abasto la forma en la que se venía trabajando para controlar todos sus servicios que prestan y por ende sus ganancias ya que dichas necesidades se vuelven la causa de muchos problemas. Una empresa grande posee problemas grandes.

Para empezar, se debe de realizar un análisis a la empresa en general con la única finalidad de estudiar, analizar y proponer nuevos métodos, cambiando procesos o implementando herramientas que ayuden en el desempeño de la empresa que se estudia y solucionar dichos problemas.

En el presente informe se realizó el análisis para la empresa Restaurante de comida típica “CALLE REAL” la cual está ubicada en 4ta calle D12-09 zona 1 Quetzaltenango, sector terciario o de servicio al cual se pertenece, dicha empresa se catalogó como empresa en crecimiento y desarrollo el propietario accedió a la propuesta planteada por los estudiantes del curso de análisis y Diseño.

Dichos estudiantes realizaron la correspondiente entrevista al propietario para conocer la empresa en si su funcionamiento en las actividades que se realiza a diario y mensual para poder analizar lo que realmente se necesita para implementar y así automatizar sus actividades para luego realizar los estudios correspondientes según el temario del curso.

DATOS GENERALES

Nombre de la empresa

Restaurante de comida típica Calle Real

Ubicación

4ta calle D12-09 zona 1 Quetzaltenango, sector terciario o de servicio al cual se pertenece

Contacto

7725-0397, Julio Francisco, Chef a cargo

Inscripción

Pequeño contribuyente

Descripción general

Calle Real es un restaurante ubicado en la zona 1 de la ciudad de Quetzaltenango, con tres años de funcionamiento desde su fundación, inspirado en las raíces guatemaltecas su entorno es totalmente colonial, con decoraciones preponderantes regionales, mostrando algo de cultura nacional y gastronómica. Actualmente el menú es estacionario y variado, tomando gastronomía extranjera y nacional, pero como eje principal la nacional, dando platillos preponderantes de la región aledaña del departamento, tomando como base recetas caceras y brindando una atención como si estuviera en el comedor de su hogar. El nombre restaurante y comida típica quiso dar a conocer que en dicha empresa no solo se pueden encontrar platillos regionales guatemaltecos (Comida típica), sino también una variedad de platillos demandantes que no son tradicionales dentro de la gastronomía meramente guatemalteca, por ello lleva al frente del nombre la palabra “Restaurante”

Breve Descripción de que tratará el proyecto

Nos enfocaremos en crear un sistema para el control, gestión y operación del Restaurante Calle Real, el cual se encuentra ubicado en 4ta calle D12-09 zona 1 Quetzaltenango, este sistema tiene como objetivo el poder ser utilizado para uso diario del restaurante mientras desarrolla sus operaciones, desde el registro del pedido ya sea para llevar o consumir en el local, así mismo el cobro de ellos y el apartado de facturación, también el registro y control de compras, almacenar la información de clientes, proveedores, ingredientes, recetas, así mismo realizar búsquedas de criterios según campos como direcciones, nombres, teléfonos, el sistema también podrá mostrar o actualizar el inventario. Para ingresar a dicho software y un aparto para administrar a los usuarios (crear, modificar o eliminar).

Personas que interactúa con el sistema:
Administrador/Dueño

Módulos en que se divide el proyecto y explicación sobre su función de cada uno:

Módulo 1: Ventas

En este módulo se realizarán las ventas en la caja, tanto las ventas rápidas como las ventas del servicio para llevar en el restaurante, durante la realización de las ventas se van a generar la cantidad de pedidos que el cliente vaya a realizar, eso depende del numero de mesa en la que se van a registrar.

Módulo 2: Inventario

Llevaremos a cabo el control de inventario de la empresa sobre el producto que ingresa, así como los avisos cuando falten productos, se administran los tipos de platillos que se van a registrar, los tipos de categorías para que cada suministro tenga su categoría correspondiente.

Módulo 3: Cotizaciones

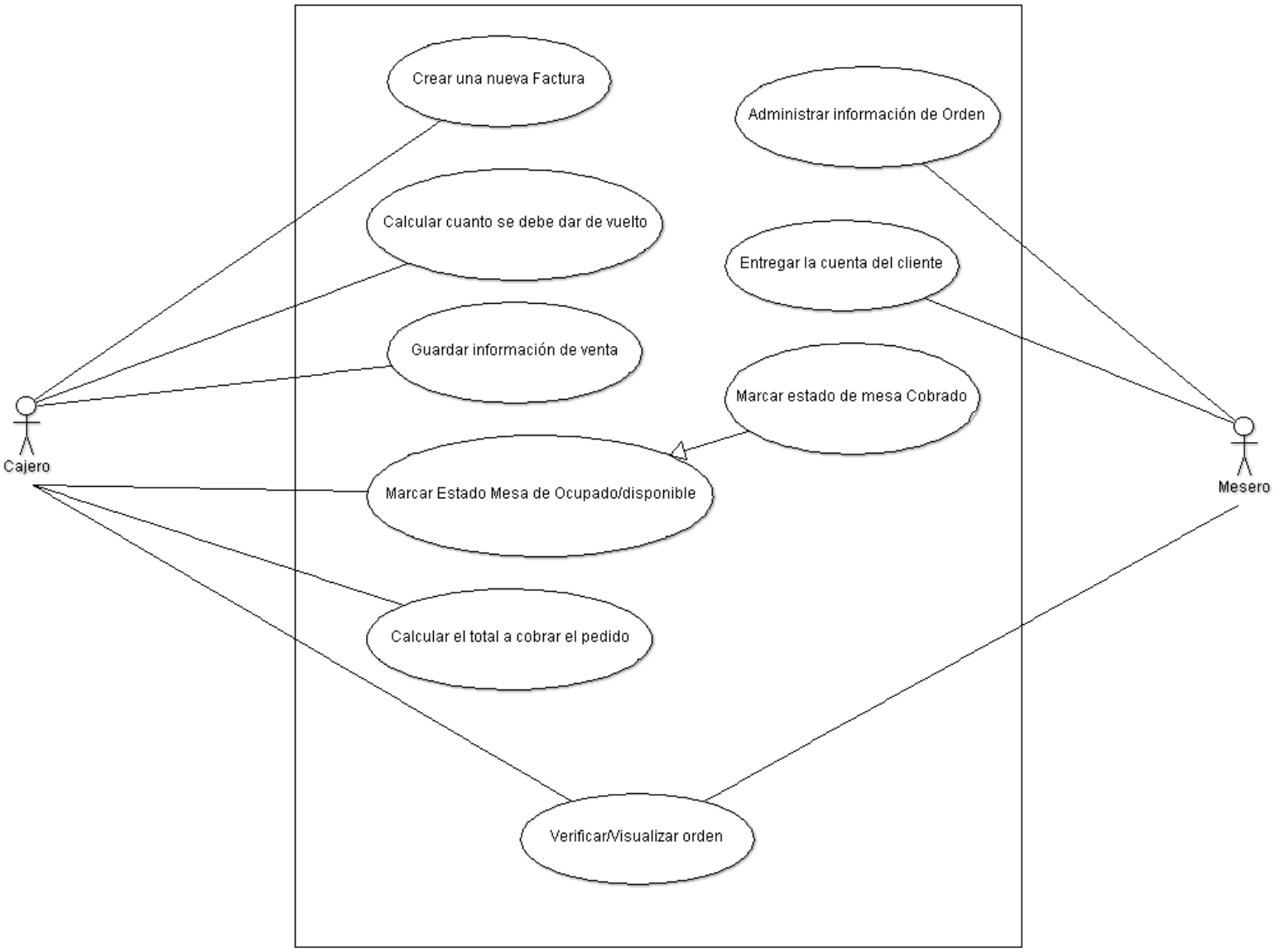
El cual consiste en poder crear un documento con la información para el solicitante acerca de los precios especiales y/o servicio de catering

Módulo 4: Administración Financiera

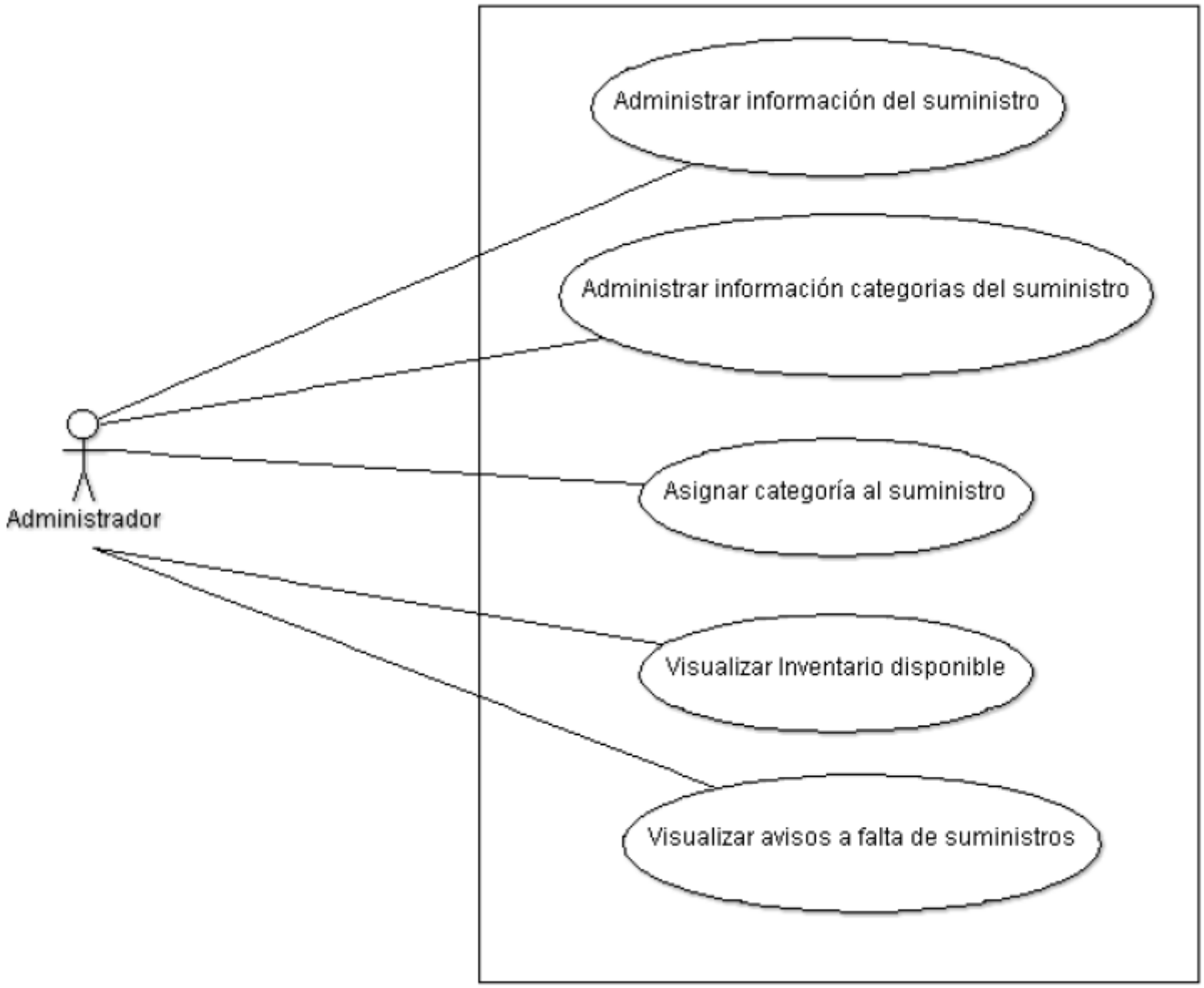
Controlar los cobros recibidos y gestionar los impuestos pagados o por pagar

DIAGRAMAS DE CASO DE USO

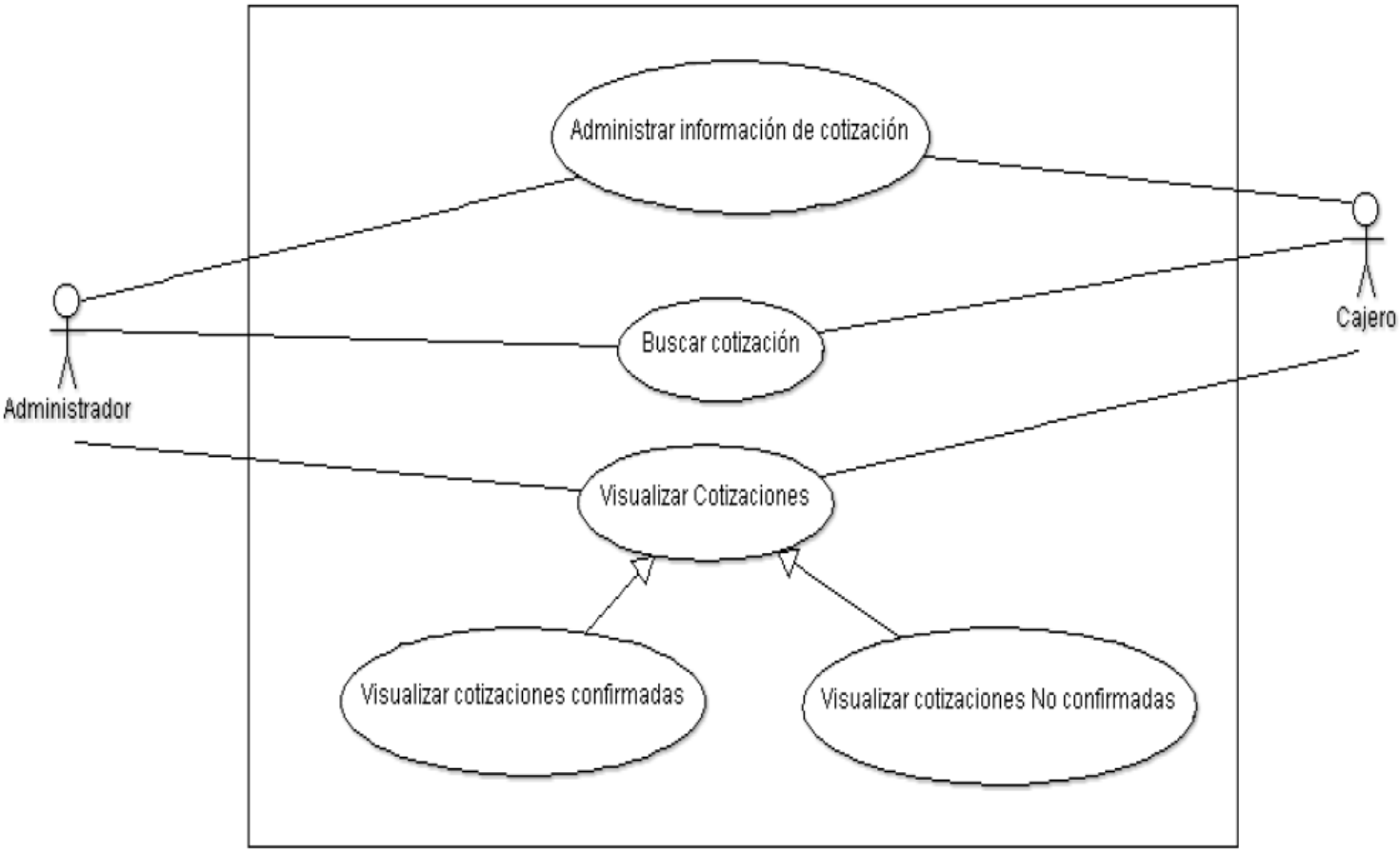
Caso de Uso: Modulo Venta



Caso de Uso: Modulo Inventario



Caso de Uso: Modulo Cotización



Caso de Uso: Modulo administración Financiera

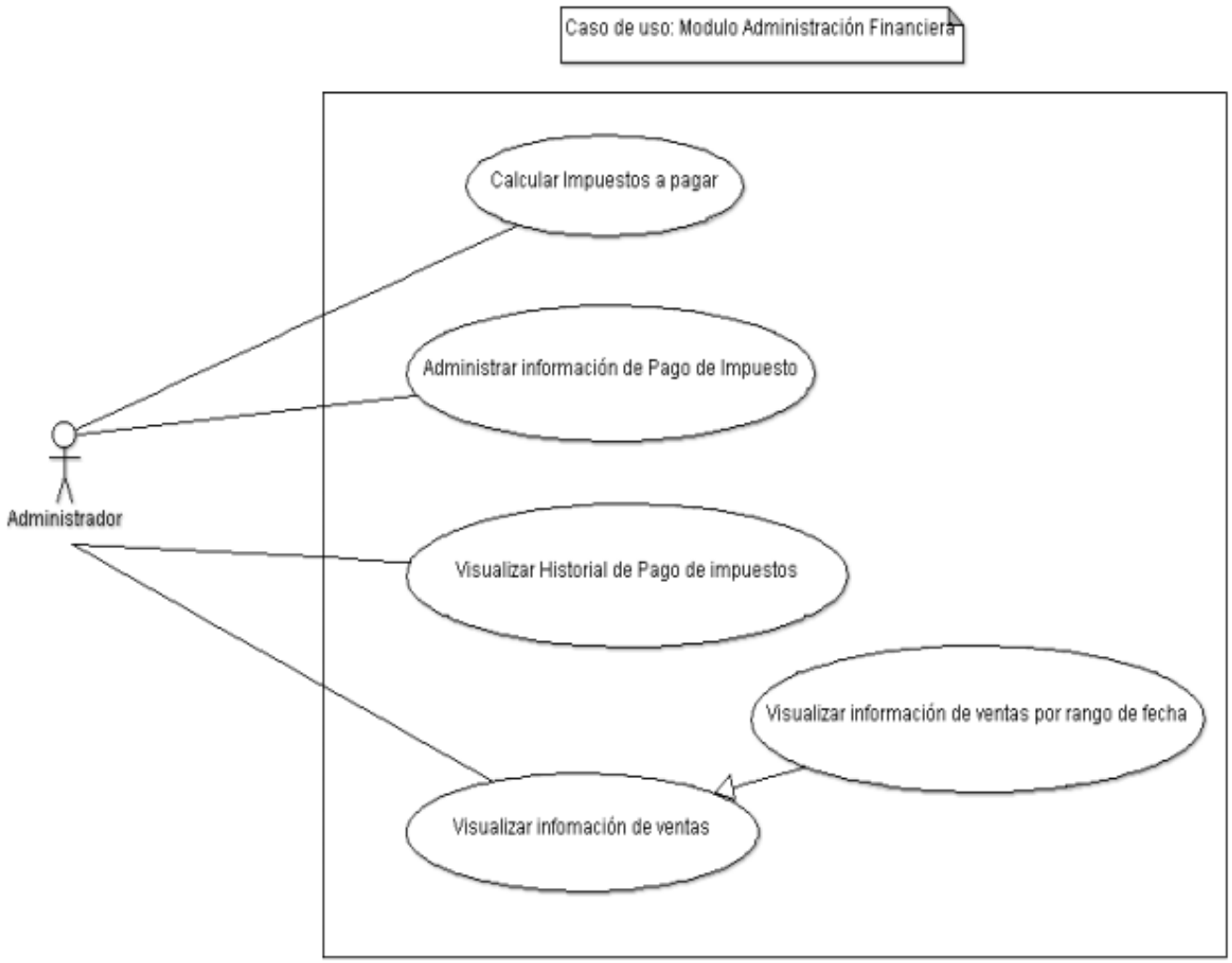
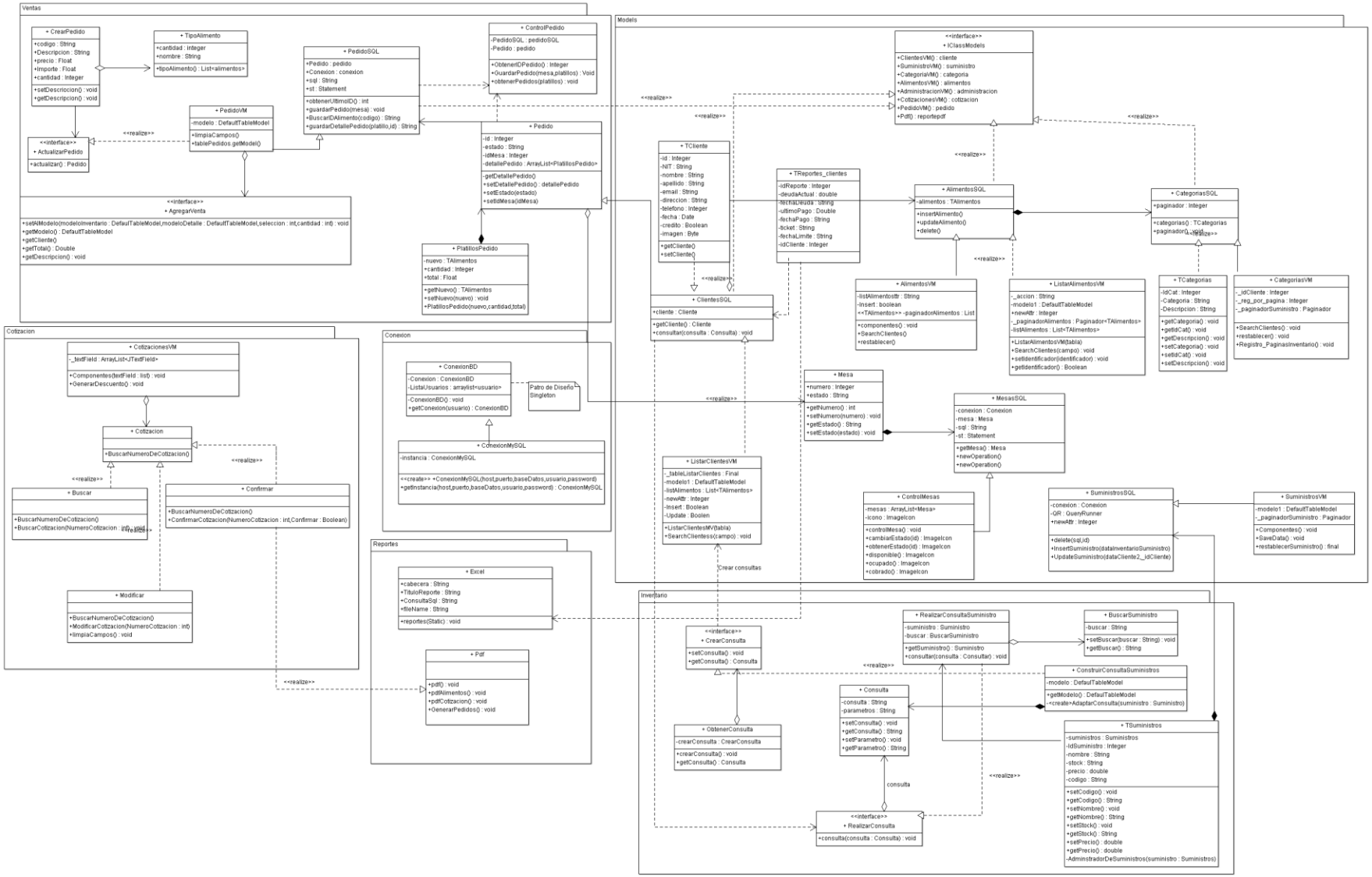


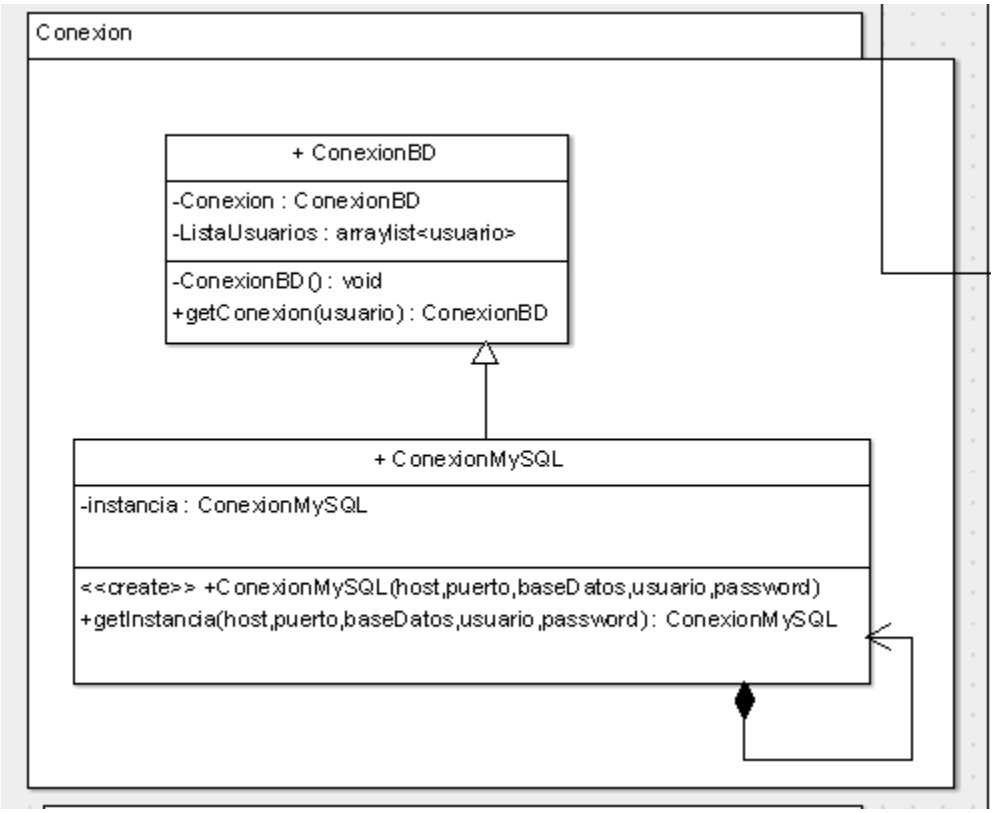
DIAGRAMA DE CLASES

Diseño en el MVC: se crearon modelos para crear objetos, vista es la presentación que se genera en la pantalla, que sería los Jframe, y en los controladores nos ayuda a definir el modo en que la interfaz reacciona a la entrada del usuario.



Patrones de Diseño implementados en el sistema.

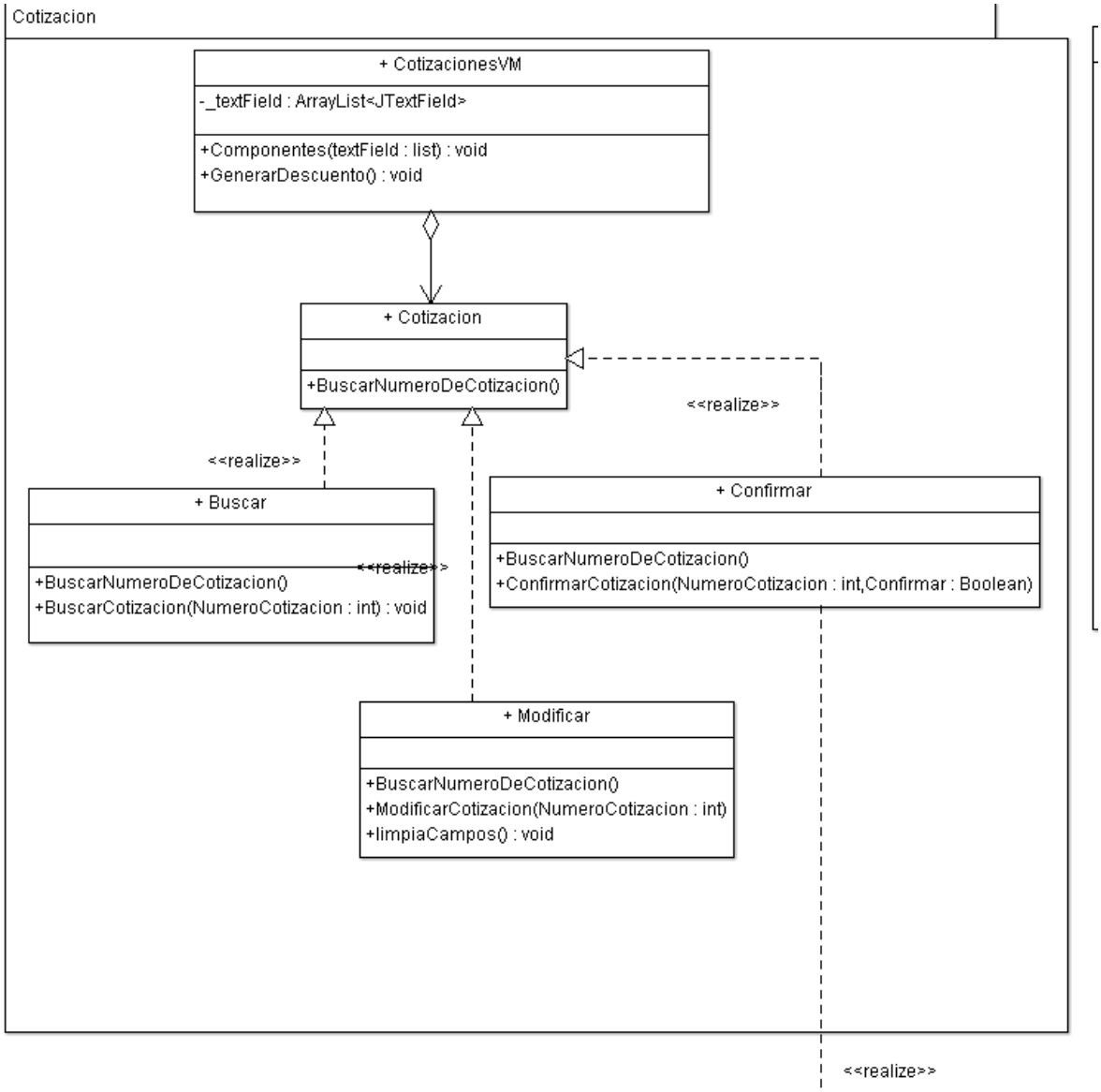
Singleton



En el paquete de conexión se utilizó Patrón de diseño de la rama Creacional en este caso fue el Singleton el cual nos asegura que una clase solo puede ser instanciada una vez, y además de proveer un punto de acceso a esta; en este caso la única instancia se implementó en la clase conexión, esta está encargada únicamente de realizar la conexión.

El fin para poder usar este patrón es solo tener una instancia en todo el proyecto así mismo para poder asegurar que solo esa instancia sea utilizada en el momento necesario, tomando en cuenta no instanciar muchas veces la conexión a la BD como acostumbramos

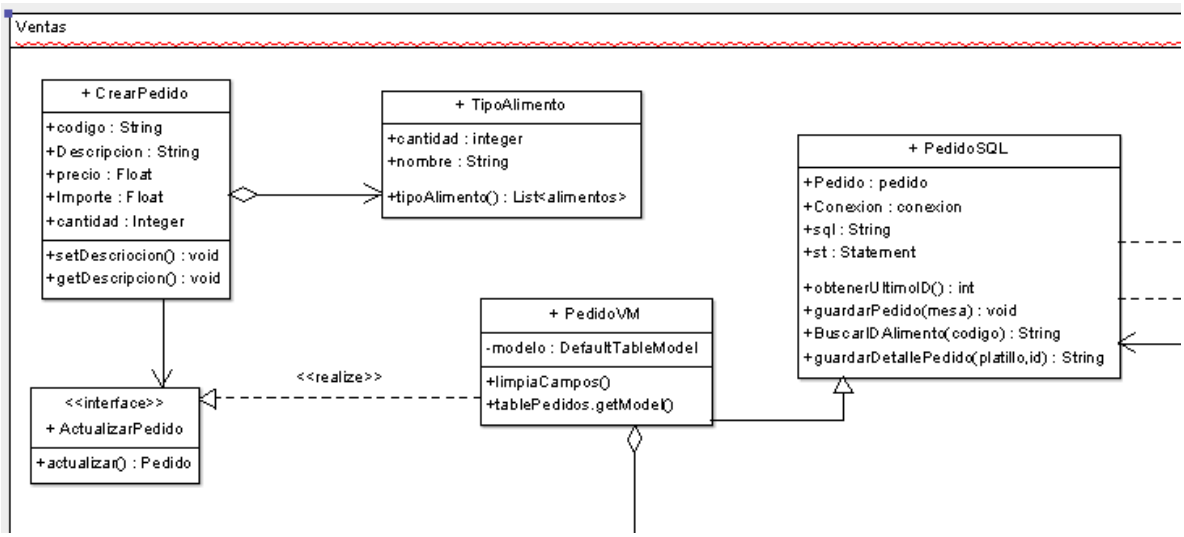
Strategy



Utilizamos strategy puesto que en cotizaciones separamos las diferentes responsabilidades que tiene esta clase en específico y como solo difieren en su comportamiento, solo cambian las funciones que deben de tener cada uno puesto que llegamos al mismo punto de poder ir trabajándolas por separado sin afectar el funcionamiento

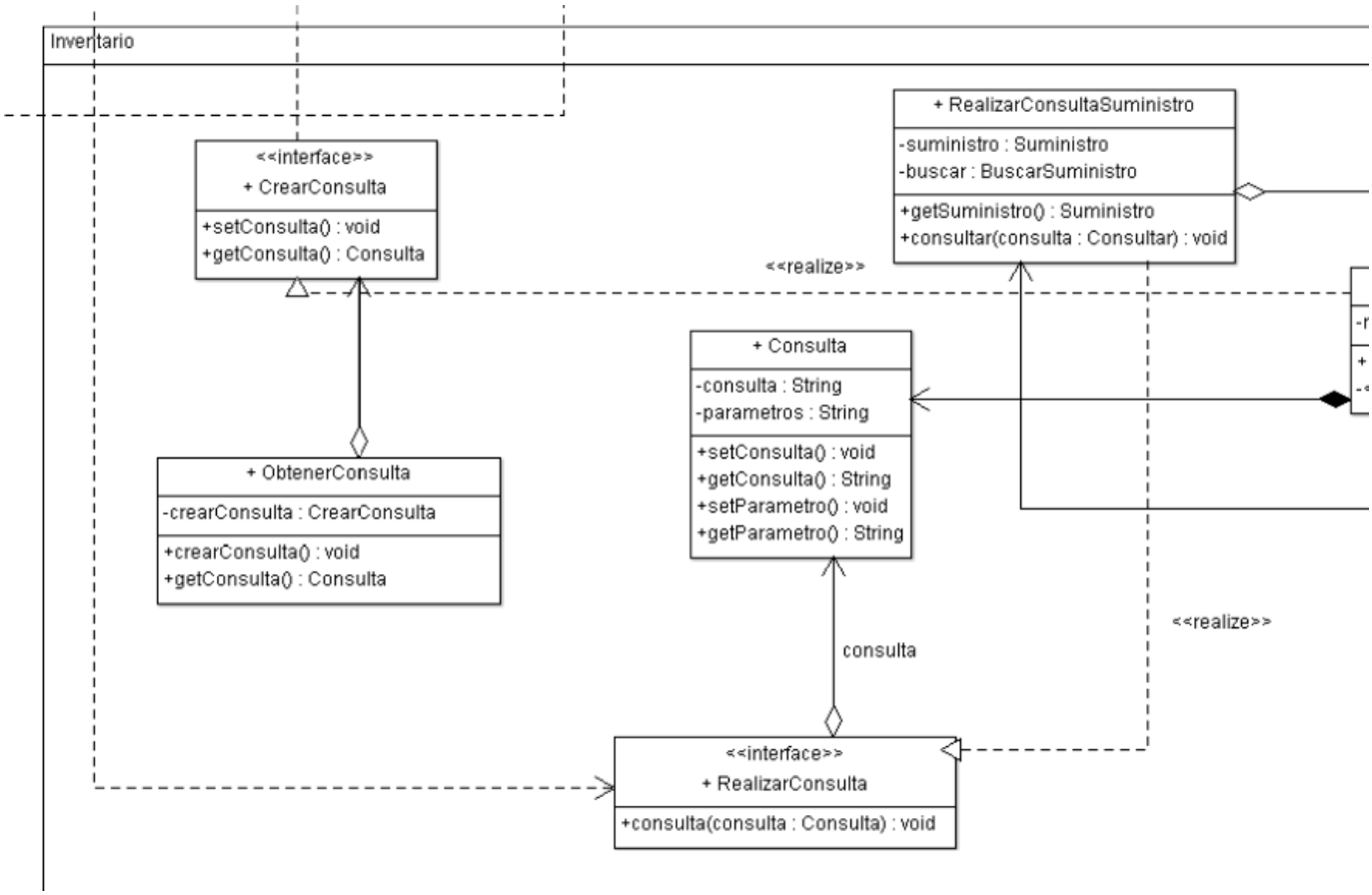
Aquí separamos las responsabilidades asi mismo separamos los modelos a la hora de poder guardar en la base de datos

El principio de Delegar las clases



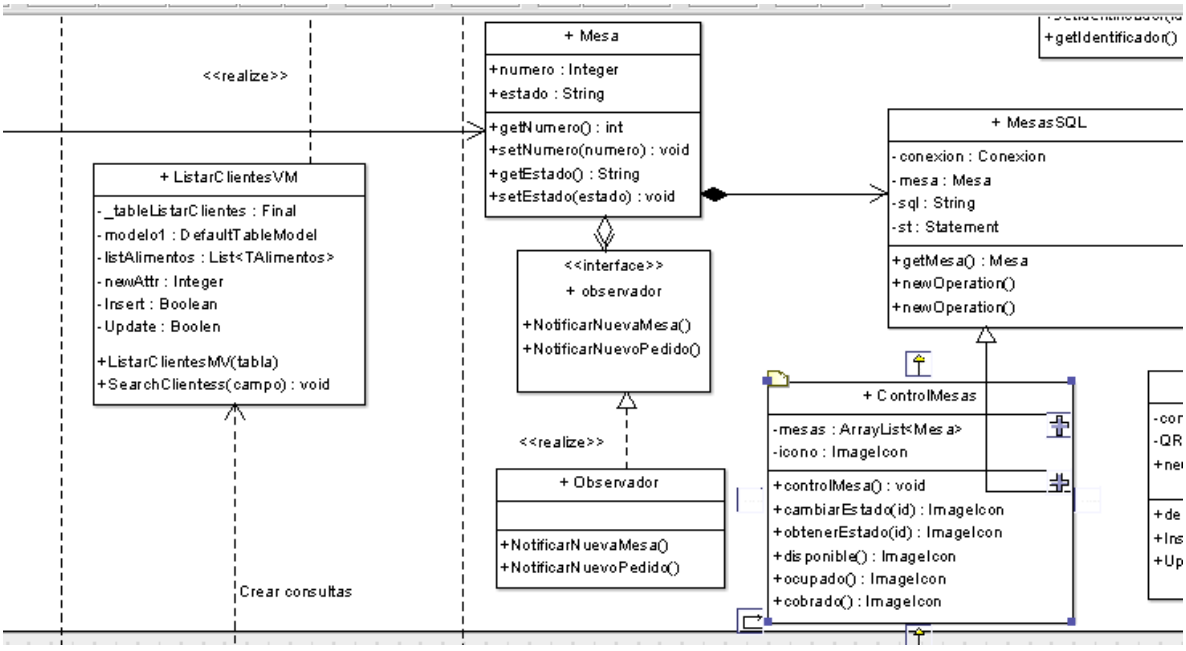
Aquí vamos a separar las responsabilidades para no dejar que solo una clase realice todos como por ejemplo Actualizar Pedido, lo separamos en pedido para poder asi mismo no cargar con muchas responsabilidades al pedido, con el fin de poder cumplir con SOLID asi mismo tenemos en cuenta que debemos está al tanto de separar las responsabilidades con la única responsabilidad que debe de tener cada uno

Strategy



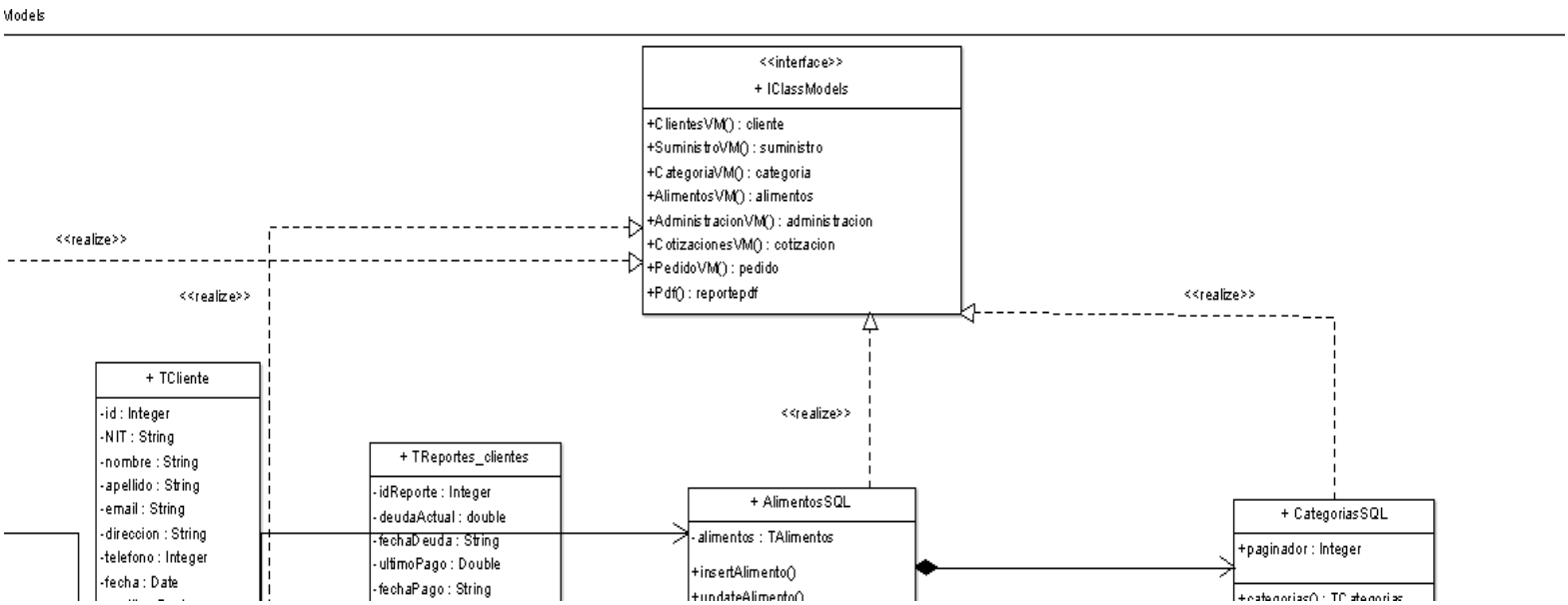
Aquí podemos definir una familia de algoritmos como es el caso de consultas el cual usamos uno acá porque separamos las responsabilidades porque cargábamos mucho una sola clase con muchas funciones como podía ser el de consultas en este caso en específico, tenemos varios tipos de consultas para las inserciones que necesitamos en la base de datos y puesto que varían en muchos varios en su comportamiento lo definimos como un strategy porque así podemos mejorar el funcionamiento

Observer



Dejamos preparado el modulo observador para cuando sea necesario poder mostrar o notificar a un usuario o en este caso puede ser el cocinero o el chef recibe la notificación de una nueva mesa y esta puede ser notificado de una nueva orden para que la empiece a preparar, esto no está implementado pero está preparado para su uso con tal de poder mejorar el funcionamiento puesto que en la empresa aun no cuentan con los recursos para más pantallas y usarlas dentro de la cocina puesto que no contemplaban más gastos a los esperados y así dejarlo como una actualización futura, con tal de poder ver la mejor manera funcional a futuro

Principio delegar



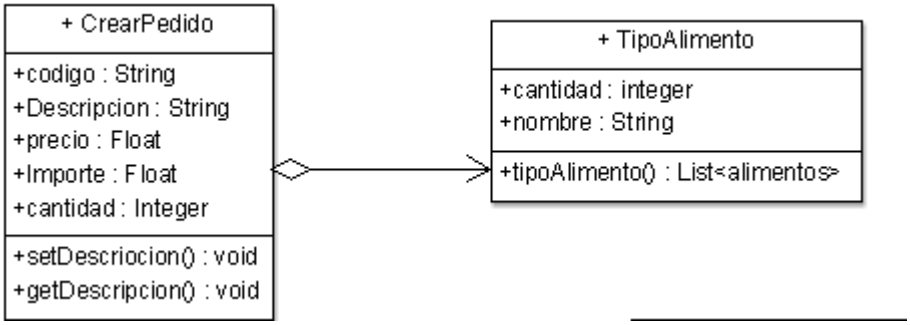
Delegamos características que tenemos que usar en específico a los demás modelos para poder separar las responsabilidades a los demás objetos y así mismo dejar que los objetos que posteriormente vamos a crear sean de la mejor manera implementados con la mejor exactitud

Con tal de poder mejorar y dejar que todos tengan una función en específico y delegando cada una de las responsabilidades

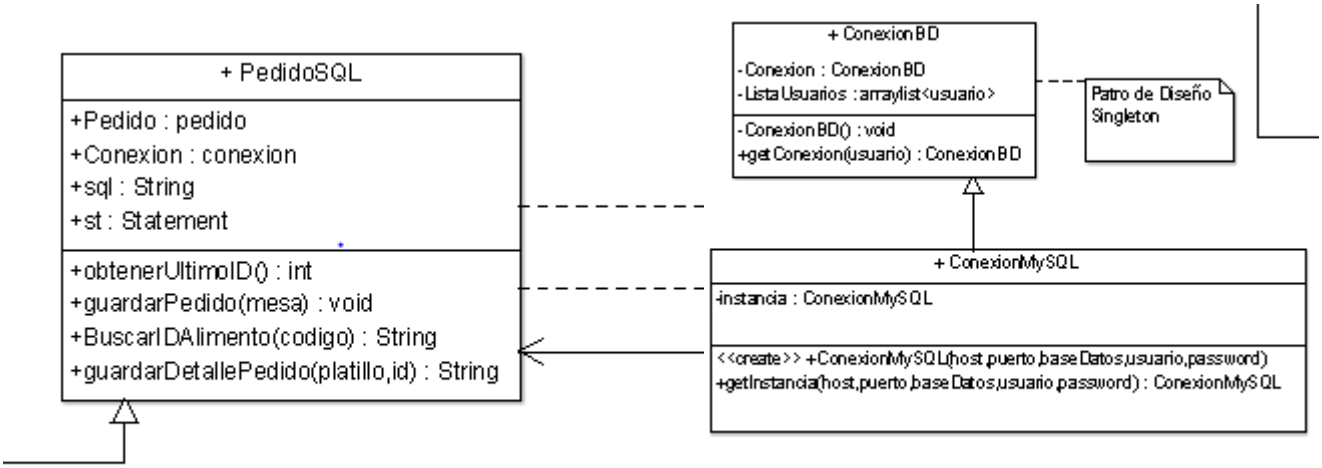
Clases:

Principio SOLID

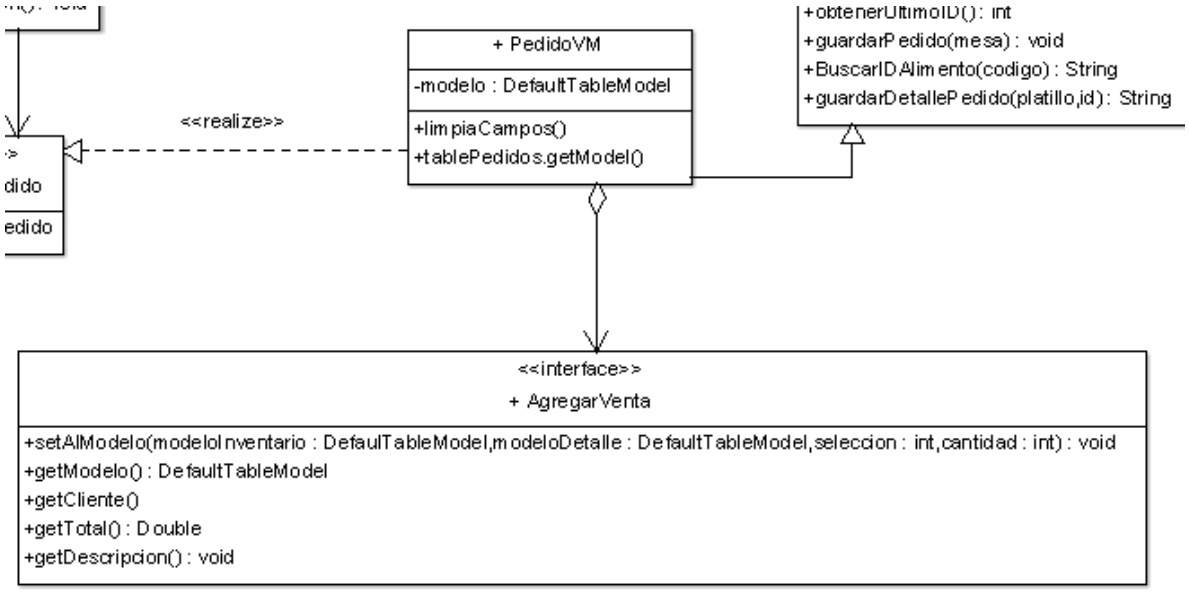
Responsabilidad Única



En la clase crear pedido, se aplicaron todas sus responsabilidades que le pertenecen a esta dicha clase, esta clase tiene una cohesión alta por lo tanto solo realizar métodos que solo le corresponde a esta clase, donde capturamos todos los campos correspondientes para realizar un pedido, en la clase de tipo alimento obtenemos los platillos que se van a agregar al pedido que se está creando.

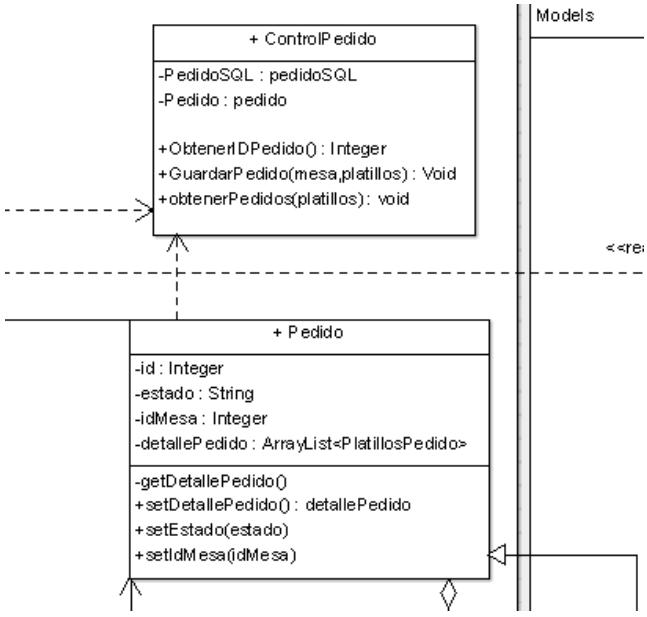


La clase pedidoSQL tiene sus responsabilidades de gestionar todo lo que tenga que ver con las consultas SQL, tanto como ingresar modificar o liminar de la base de datos.



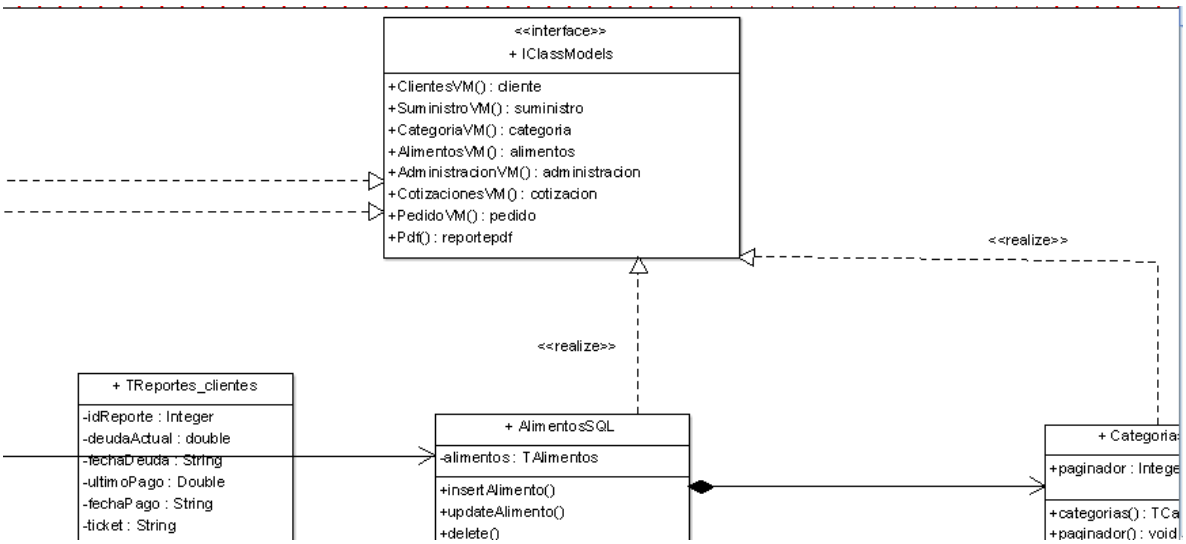
En la clase del pedidoVM determinamos todas las acciones que solo esta clase puede realizar evitando darle responsabilidad que no le correspondan.

Principio abierto / cerrado



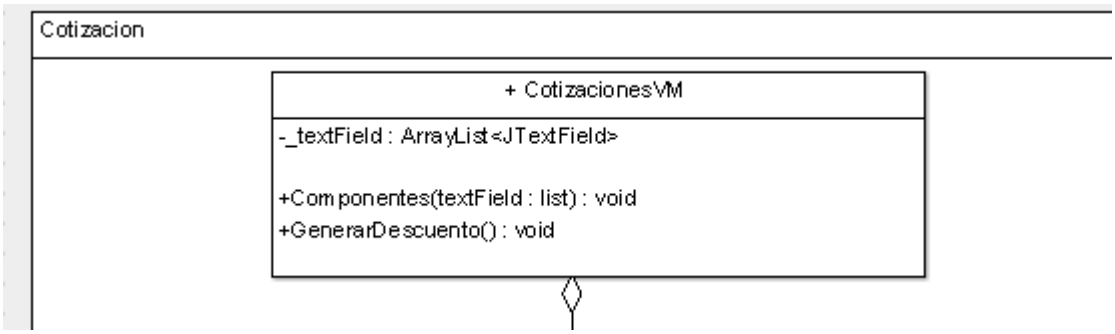
Aquí establecimos que los componentes del software deben estar abiertos para extender a partir de ellos, pero cerrados para evitar que se modifiquen en cualquier parte del sistema.

Principio de substitución de Liskov

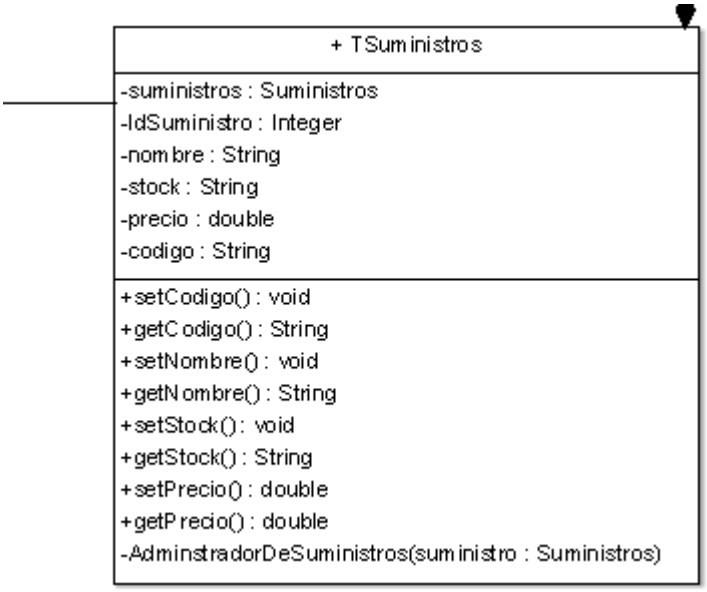


Este principio establece que una subclase puede ser sustituida por su superclase. Es decir se instanciaron todos los objetos que van a ser utilizado en el Jframe principal para que solo pueda ser llamado para realizar sus acciones correspondientes

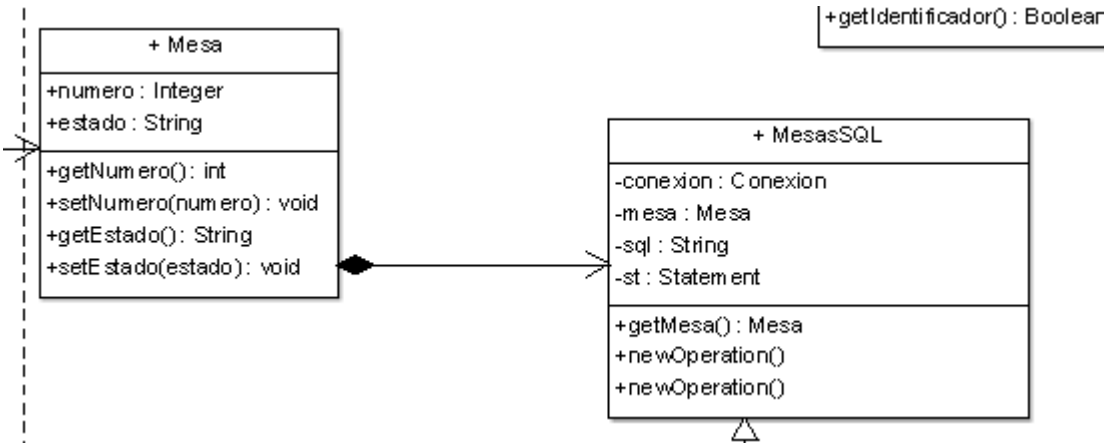
Explicación de clases



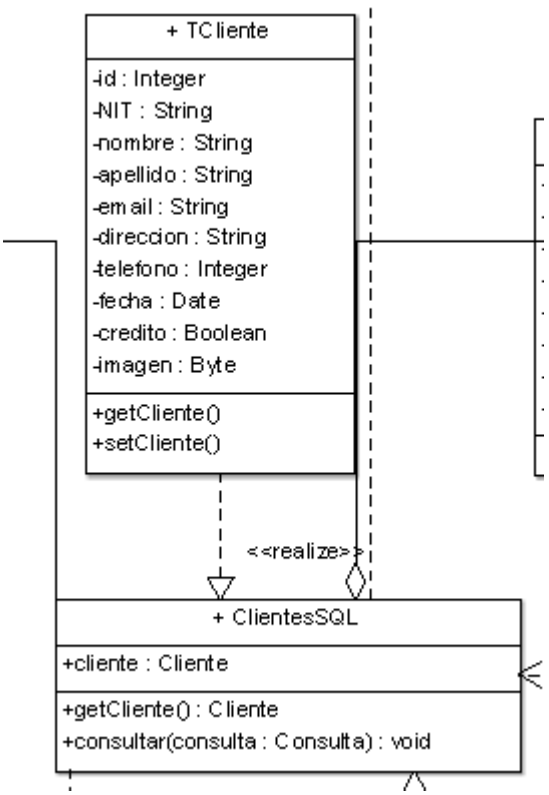
Esta clase pertenece a los controladores en la cual solo realizar la acción que el usuario va a realizar con las cotizaciones.



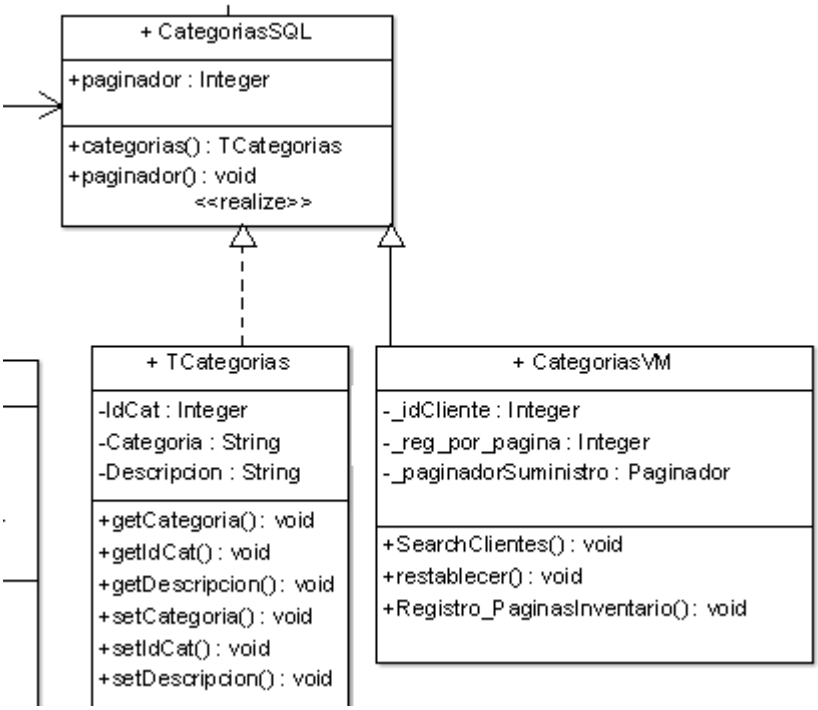
En esta clase se genera el modelo de los suministros, generando todos los atributos necesarios para poder registrar a la base de datos.



En la clase mesa es el controlador que nos permite hacer las acciones cuando el usuario vaya interactuar con el sistema, y por lo tanto va a necesitar de la mesaSQL donde se va a realizar todos los registros a la base de datos.



En la clase TCliente es el modelo de los atributos que se van a generar cuando se registre un cliente, y la parte de clienteSQL se va a encargar de hacer los registros a la base de datos.



En la categoriaSQL se va a encargar de hacer todas la consultas a la base de datos, realizando sus acciones correspondientes, extendiendo de la clase TCategoria donde se encuentran todos sus atributos, y la clase categoríaVM es el controlador donde se encargar de hacer las diferentes acciones cuando el usuario vaya a interactuar con el sistema.

Principios

Unas de las primeras cosas que buscamos corregir al realizar el proyecto fue cumplir con los principios SRP y OCP , es decir, buscar que mis clases tuvieran solo una responsabilidad, evitar llenar las clases con métodos que tuvieran funciones muy diferentes unos de los otros. Primero se creó un modelo general de las clases, para luego ir viendo que era lo que necesitaba ser cambiado para cumplir con los patrones. Consideramos que el diseño cumple con OCP debido a que se tiene “espacio” para seguir creciendo, es decir, es posible agregar más características si eso se quisiera, no habría problemas en esto, y no tendría que modificar ninguna clase ya existente para realizar esto.

Para poder cumplir con el principio de liskov. Si se deseara generar una nueva forma de realizar una consulta, solo tendría que crear una nueva clase que contenga el algoritmo nuevo y relacionarlo con la interface adecuada, de nuevo, no tendría que modificar ni la clase consulta, ni ninguna clase de algoritmo existente.

Con el principio SRP consideramos que se cumplio , debido a que se busco colocar distintas clases o interfaces que contuvieran los métodos que no eran necesarios o no cumplían con lo que la clase especifica necesitaba,

CONCLUSION

Gracias al análisis hacia la empresa Restaurante de comida típica “CALLE REAL” previo al desarrollo del Sistema de información y el haber contemplado sus componentes o pasos se pudieron identificar las necesidades y nos ayudó a traducir dichas necesidades en un modelo de Sistema que utiliza varios componentes, entre ellos el Software, hardware, personas, base de datos, documentación y procedimientos.

Hemos llegado a la conclusión que antes de comenzar con el desarrollo de cualquier proyecto, es necesario conducir un estudio de Sistemas para detectar todos los detalles de la situación actual de la empresa.

La información recopilada con el análisis y estudio sirve como base para crear varias estrategias de Diseño, ya que se entiende el funcionamiento o procesos que la empresa emplea, el flujo de información y así se puede saber cómo mejorar el rendimiento de la empresa, optimizar el uso de los recursos y satisfacer las necesidades, Dentro del diseño es donde se fomenta la calidad del Proyecto, el diseño es la única manera de materializar con precisión los requerimientos del cliente para tener ese boceto o plano en el que nos da una vista de los requerimientos.

Repositorio:

https://github.com/elmergustavo/SistemaPuntoVenta.git