Documento de ANÁLISIS

Sagres

Sistema de Administración y Gestión de RESTaurantes

Adrián Víctor Pérez Lopera
Sergio Rodríguez Lumley
Nabil Sabeg

V1.1

ÍNDICE DE CONTENIDO

Apartado de control de versiones

Todas las versiones están especificadas a fondo en el apartado de "Apéndices", al final de este documento, cada apéndice se corresponde en nombre con su número de versión. Por ejemplo, el "Apéndice 0.1" se corresponde con la versión v0.1. Para ver los cambios realizados sobre cada versión, hay que ir deshaciendo los cambios desde el final.

Versión	Fecha	Descripción
V1.0	23/04/10	Se ha creado el documento de análisis.
V1.1	07/05/10	Se han corregido fallos en algunos contratos y diagramas de colaboración.

1. Modelado estático

Dentro del proceso unificado, el siguiente paso es el análisis. Para ello lo primero es realizar un modelo estático de nuestro sistema. El modelo estático o diagrama de clases contiene todos los elementos que juegan un papel importante y merecen ser representados como clases en el sistema. Se compone de la definición de las clases, sus atributos, métodos y las relaciones entre ellas.

Descripción de clases conceptuales

Clase	Descripción
Pedido	Representa un pedido de cliente. Guarda toda la información relevante al sistema de un pedido.
Factura	En esta clase se guardan todas las facturas de los clientes.
Restaurante	En esta clase se guardan los datos del restaurante (Nombre, dirección, razón social...)
ElementoPedido	Representa un elemento de un pedido determinado seleccionado por el cliente. Almacena la información relevante del plato o bebida que representa.
ElementoColaBar	Es un elemento de pedido de tipo bebida. Representa las bebidas seleccionadas por el cliente en el pedido.
ElementoColaCocina	Es un elemento de pedido de tipo plato. Representa los platos seleccionados por el cliente en el pedido.
ElementoBebida	Elemento de tipo bebida. Representa a las bebidas que se ofrecen en la carta.
ElementoPlato	Elemento de tipo plato. Representa a los platos de comida que se forman con la combinación de distintos ingredientes.
Elemento	Representa a cada elemento de la carta del restaurante. Almacena la información asociada al plato o bebida que representa.
SeccionComida	Es un tipo de sección. Su función es organizar los elementos de la carta que son platos de comida.
SeccionBebida	Es un tipo de sección. Su función es organizar los elementos de la carta que son bebidas.
Seccion	La carta del restaurante se encuentra divida en secciones. Está clase se encarga de dividir los distintos elementos que componen la carta según el tipo de cada uno.
Carta	Esta clase es la encargada de almacenar, controlar y gestionar toda la información referente a la carta del restaurante.

Relación entre clases conceptuales

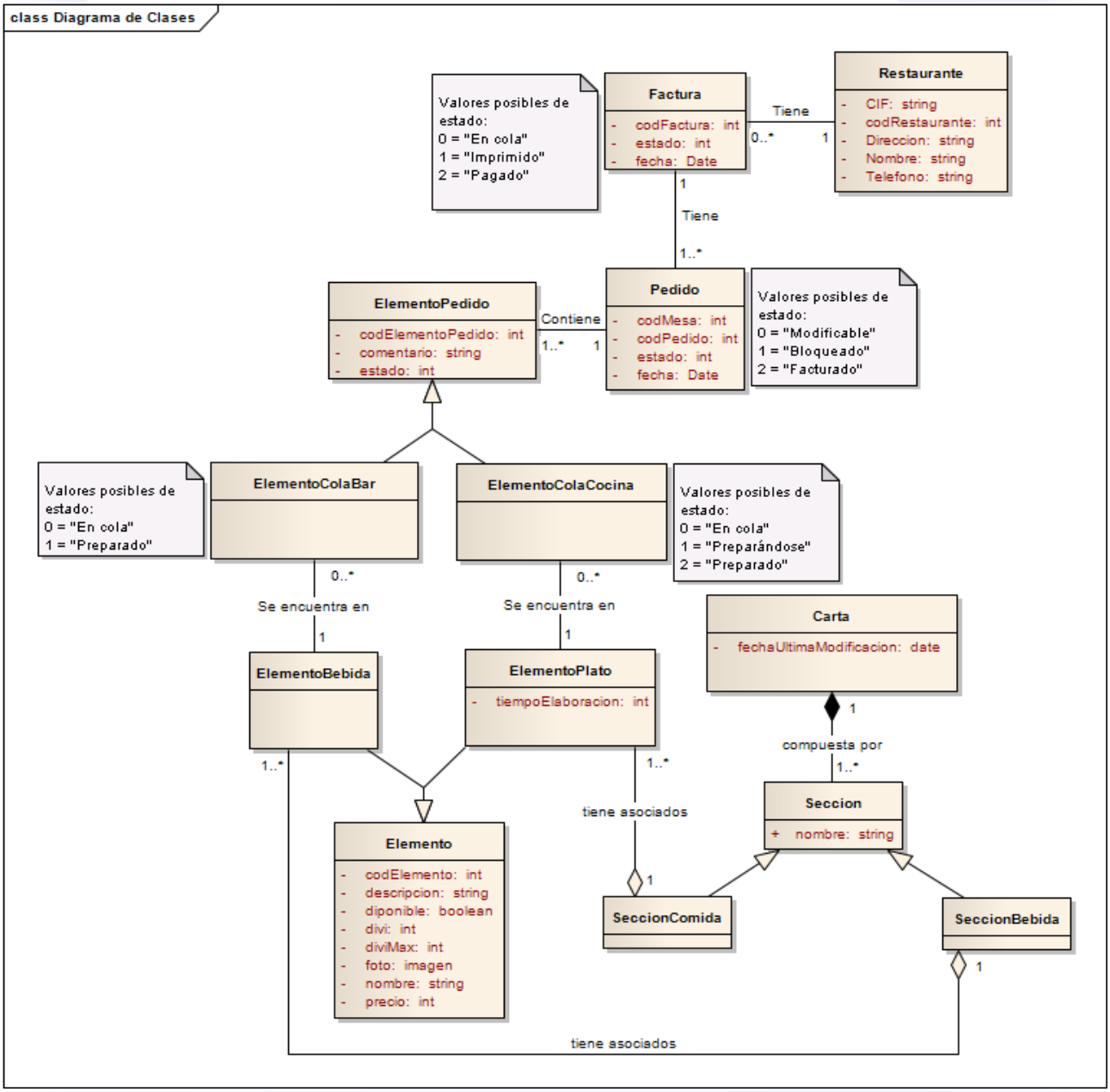
Clases	Relación
Pedido - Factura	Es una relación de asociación entre pedido y factura. Una factura se refiere a uno o varios pedidos.
Restaurante - Factura	Es una relación de asociación entre restaurante y factura. así la factura puede obtener la información necesaria del restaurante.
Pedido - ElementoPedido	Es una relación de asociación, la cual establece que un pedido contiene uno o muchos elementos pedidos por el cliente.
ElementoColaBar - ElementoPedido - ElementoColaCocina	Se establece una generalización entre estas tres clases en la que ElementoPedido es la clase padre y las otras dos son las que heredan de ella.
ElementoColaBar - ElementoBebida	Es una relación de asociación en la que un ninguno o muchos ElementoColaBar hacen referencia a un único ElementoBebida.
ElementoColaCocina - ElementoPlato	Es una relación de asociación en la que un ninguno o muchos ElementoColaCocina hacen referencia a un único ElementoPlato.
ElementoBebida - Elemento - ElementoPlato	Se establece una generalización entre estas tres clases en la que Elemento es la clase padre y las otras dos son las que heredan de ella.
ElementoPlato - SeccionComida	Está claro que una sección de la carta organiza a una serie de elementos de un tipo concreto. En este caso, una sección de comida organiza elementos de tipo plato, es decir, elementos de la carta que están realizados con ingredientes. La relación que se establece es de agregación, puesto que aunque los elementos se vinculan con un "todo" como es la sección, estos pueden existir de manera independiente.
ElementoBebida - SeccionBebida	Se da la misma situación que para la relación anterior.
SeccionComida - Seccion - SeccionBebida	Se establece una generalización entre estas tres clases en la que Sección es la clase padre y las otras dos son las que heredan de ella.
Carta - Seccion	Como se ha definido anteriormente una carta está compuesta por secciones. Se trata de una relación de composición puesto que la carta se trata como un "todo" del cual las secciones forman parte y no tendrían sentido por sí solas si este "todo" no existiese.

Atributos asociados a clases y/o relaciones

Clase	Atributo	Descripción
Pedido	codMesa : int	Código que identifica a una mesa. Por mesa, se entiende un terminal de cliente.
	codPedido: int	Código único que identifica a un pedido.
	estado : int	Estado de un pedido, este puede variar entre "Modificable", "Bloqueado" o "Facturado".
	fecha : Date	Fecha, hora, minuto y segundo en el cual fue creado el pedido.
Factura	codFactura : int	Código único identificador de la factura.
	estado : int	Indica el estado de la factura, 0=en cola, 1=imprimido, 2=pagado, 3=anulado.
	fecha : Date	La fecha en la que se ha pagado la factura.
Restaurante	CIF : string	Código de identificación fiscal del restaurante.
	CodRestaurante : int	Código del restaurante.
	Direccion : string	Dirección del restaurante.
	Nombre : string	Nombre del restaurante.
	Telefono : string	Teléfono del restaurante.
ElementoPedido	codElementoPedido : int	Código identificador del elemento de pedido.
	comentario : string	Comentario realizado por el cliente a la hora de seleccionar el pedido. Se permite que haga anotaciones al cocinero como "Sin cebolla, por favor".
	estado : int	Estado del elemento de pedido, puede variar, según si es un plato ("En cola", "Preparándose", "Preparado") o si es una bebida ("En cola", "Preparado").
Elemento	codElemento : int	Número que identifica a un elemento.
	descripción : int	Breve descripción del elemento.
	disponible : int	Nos dice si un elemento está disponible para su pedido o está invalidado por cualquier motivo.
	diviMax : int	Número de divisiones que se pueden hacer del elemento (en el caso de un plato raciones, en el caso de una bebida vasos).
	foto : imagen	Imagen para visualizar el elemento.
	nombre : string	Nombre del elemento que se visualizará en la carta.

	precio : int	Precio de venta al público del elemento.
ElementoPlato	tiempoElaboracion : int	Tiempo en minutos que se tarda en elaborar el plato. Solo se tiene en cuenta en el caso de que no haya raciones.
Seccion	Nombre : string	Nombre de la sección.
Carta	fechaUltimaModificacion : Date	Indica la fecha más reciente en la que la carta fue modificada. Este atributo se utilizará para controlar que la carta no sufra cambios importantes hasta pasados los 6 meses desde la última modificación.

Diagrama de clases



Descripción de contratos del sistema

Nombre:	Pedido = getSiguientePedidoCocinaEnCola()
Responsabilidad:	Obtener el siguiente pedido que lleve más tiempo en cola de cocina, es decir, con la fecha más lejana (el más antiguo), que tenga algún ElementoColaCocina en estado "En cola".
Tipo:	Sistema
Referencias:	Caso de uso "Cambia Estado Plato"
Nota:	
Excepción:	• Si no hay ningún pedido con algún ElementoColaCocina en estado "En cola".
Precondición:	
Postcondición:	
Salida:	Un pedido con algún ElementoColaCocina en estado "En cola", con la fecha más antigua posible.

Realizado por Sergio Rodríguez Lumley

Nombre:	Pedido = getPedidosCocinaPreparandose()
Responsabilidad:	Obtener todos aquellos pedidos con algún plato en estado “Preparandose”.
Tipo:	Sistema
Referencias:	Caso de uso “Cambia Estado Plato”
Nota:	
Excepción:	• Si no hay ningún pedido con algún ElementoColaCocina en estado “Preparandose”.
Precondición:	
Postcondición:	
Salida:	Todos los pedidos que tengan algún plato en estado “Preparandose”.

Realizado por Sergio Rodríguez Lumley

Nombre:	Bool = seleccionaPlato(Pedido pedido, ElementoColaCocina plato)
Responsabilidad:	Seleccionar un plato del pedido “pedido” y cambiar el estado del plato. Si estaba “En cola” lo cambia a “Preparandose”, el del pedido a “Bloqueado” y actualiza la cantidad de productos asociados a ese plato. Si estaba “Preparandose” se cambiará a “Preparado”.
Tipo:	Sistema
Referencias:	Caso de uso “Cambia Estado Plato”
Nota:	
Excepción:	• Si alguno de los parámetros recibidos es incoherente (pedido nulo, plato nulo, etc). • Si el plato no pertenece al pedido.
Precondición:	
Postcondición:	• Se modificó un pedido. • Se modificó el estado de un plato al valor “Preparandose” o “Preparado”. • Se modificaron los ingredientes asociados al plato.
Salida:	Booleano con valor “Verdadero” si ha cambiado el estado del plato, “Falso” en caso contrario.

Realizado por Sergio Rodríguez Lumley

Nombre:	Pedido = getSiguientePedidoBar()
Responsabilidad:	Obtener el siguiente pedido que lleve más tiempo en cola de bar, es decir, con la fecha más lejana (el más antiguo), que tenga algún ElementoColaBar en estado "En cola".
Tipo:	Sistema
Referencias:	Caso de uso "Cambia Estado Bebida"
Nota:	
Excepción:	• Un pedido con algún ElementoColaBar en estado "En cola", con la fecha más antigua posible.
Precondición:	
Postcondición:	
Salida:	Un pedido con algún ElementoColaBar en estado "En cola", con la fecha más antigua posible.

Realizado por Sergio Rodríguez Lumley

Nombre:	Bool = seleccionaBebida(Pedido pedido, ElementoColaBar bebida)
Responsabilidad:	Seleccionar una bebida del pedido "pedido", cambiar el estado del pedido a "Bloqueado" y también el de la bebida. Si esta estaba "En cola" se cambiará a "Preparado" y reducirá la cantidad de bebidas en stock.
Tipo:	Sistema
Referencias:	Caso de uso "Cambia Estado Bebida"
Nota:	
Excepción:	• Si alguno de los parámetros recibidos es incoherente (pedido nulo, bebida nula, etc). • Si la bebida no pertenece al pedido.
Precondición:	
Postcondición:	• Se modificó un pedido. • Se modificó el estado de una bebida al valor "Preparado". • Se modificaron las bebidas asociadas al producto.
Salida:	Booleano con valor "Verdadero" si ha cambiado el estado de la bebida, "Falso" en caso contrario.

Realizado por Sergio Rodríguez Lumley

Nombre:	elementosCarta = obtieneElementosCarta()
Responsabilidad:	Operación encargada de devolver las secciones de la carta con sus correspondientes elementos.
Tipo:	Sistema.
Referencias:	Nuevo Pedido, Modifica Pedido.
Nota:	
Excepción:	
Precondición:	
Postcondición:	
Salida:	• Una lista de elementos tipo par sección de carta, lista de elementos de la sección.

Realizado por Adrián Víctor Pérez Lopera

Nombre:	exito = nuevoPedido (codMesa, elementosPedido)
Responsabilidad:	Operación encargada de incluir en el sistema un nuevo pedido (con sus elementos) asociado a una mesa.
Tipo:	Sistema.
Referencias:	Nuevo Pedido.
Nota:	El parámetro "codMesa" de la operación, se especificaba en el DMR como "mesa". Se ha cambiado ya que en este punto ya sí sabemos que las mesas van a ser referenciadas por un código.
Excepción:	
Precondición:	• El parámetro "codMesa" es correcto. • Los miembros del parámetro "elementosPedido" son correctos.
Postcondición:	• Se creó un pedido con estado "modificable". • Se crearon elementos de pedido con estado "en cola". • Se crearon enlaces entre el pedido y los elementos de pedido. • Se crearon enlaces entre los elementos de pedido y los elementos plato y/o elementos bebida de la carta.
Salida:	• True si el pedido fue realizado con éxito, false en caso contrario.

Realizado por Adrián Víctor Pérez Lopera

Nombre:	pedidosMesa = obtienePedidosMesaModificables(codMesa)
Responsabilidad:	Operación encargada de obtener los pedidos con estado "modificable" de una mesa.
Tipo:	Sistema.
Referencias:	Modifica Pedido.
Nota:	El parámetro "codMesa" de la operación, se especificaba en el DMR como "mesa". Se ha cambiado ya que en este punto ya sí sabemos que las mesas van a ser referenciadas por un código.
Excepción:	
Precondición:	• El parámetro "codMesa" es correcto.
Postcondición:	
Salida:	• Una lista de pedidos.

Realizado por Adrián Víctor Pérez Lopera

Nombre:	elementosPedido = obtieneElementosPedido(pedido)
Responsabilidad:	Operación encargada de obtener los elementos de un pedido con su cantidad.
Tipo:	Sistema.
Referencias:	Modifica Pedido.
Nota:	
Excepción:	
Precondición:	• El parámetro "pedido" es correcto.
Postcondición:	
Salida:	• Una lista de elementos de pedido.

Realizado por Adrián Víctor Pérez Lopera

Nombre:	exito = modificaPedido (pedido, elementosPedido)
Responsabilidad:	Operación encargada de destruir el pedido al que hace referencia "pedido" y crear un nuevo pedido con los elementos de "elementosPedido".
Tipo:	Sistema.
Referencias:	Nuevo Pedido.
Nota:	
Excepción:	
Precondición:	• El parámetro "pedido" es correcto. • Los miembros del parámetro "elementosPedido" son correctos.
Postcondición:	• Se eliminó un pedido y sus elementos asociados. • Se creó un pedido con estado "modificable". • Se crearon elementos de pedido con estado "en cola". • Se crearon enlaces entre el pedido y lo elementos de pedido. • Se crearon enlaces entre los elementos de pedido y los elementos plato y/o elementos bebida.
Salida:	• True si el pedido fue modificado con éxito, false en caso contrario.

Realizado por Adrián Víctor Pérez Lopera

Nombre:	Confirmación = pideFactura(codMesa,listaElementos)
Responsabilidad:	Inserta la petición de imprimir la facturas de pedido(s) que contiene listaElementos en la cola de facturas.
Tipo:	Sistema
Referencias:	Caso de uso “Pide factura”
Nota:	
Excepción:	• El código de mesa es incorrecto. • La lista de de elementos está vacia.
Precondición:	• El cliente ha hecho un pedido.
Postcondición:	• Inserción de la petición en la cola de facturas.
Salida:	

Realizado por nabil

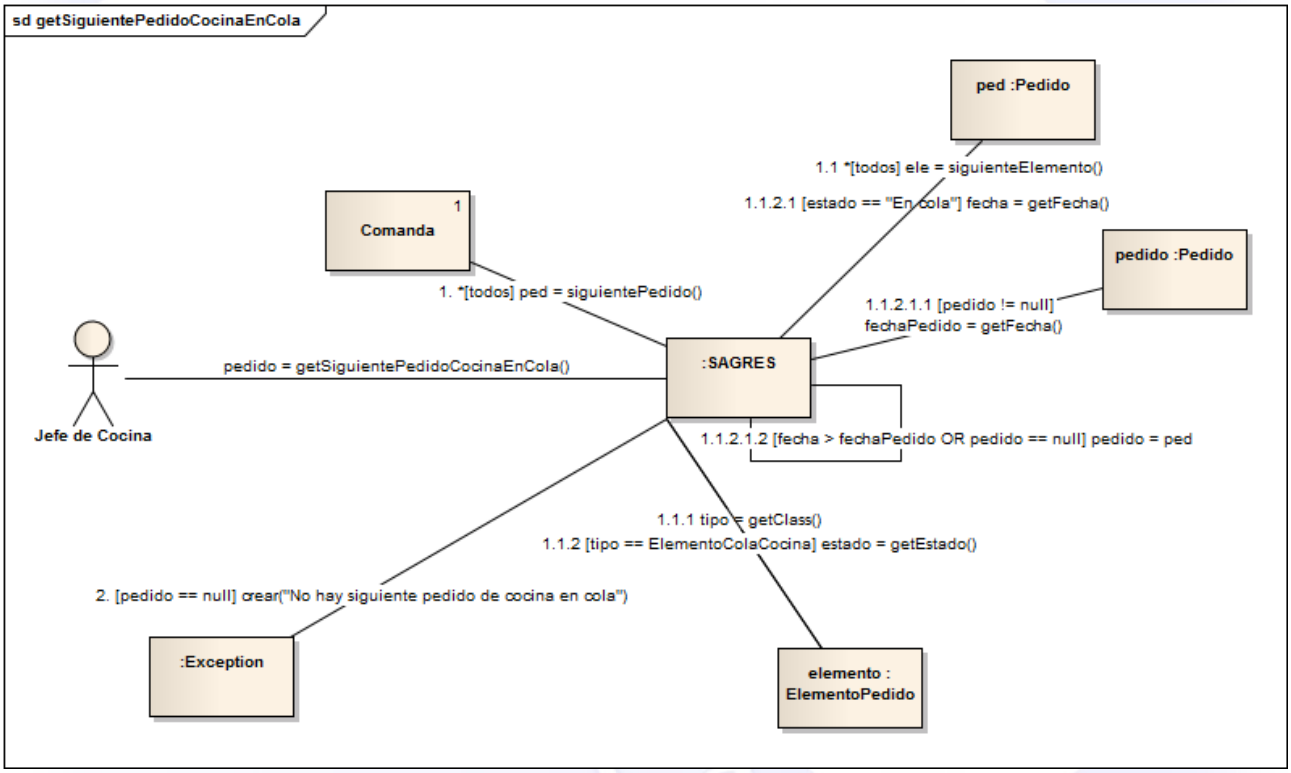
Nombre:	imprimeFactura(codMesa)
Responsabilidad:	Imprime la factura de la mesa con el código (codMesa)
Tipo:	sistema
Referencias:	Caso de uso “Cambia Estado Factura”
Nota:	
Excepción:	Código de mesa incorrecto.
Precondición:	
Postcondición:	
Salida:	

Realizado por nabil

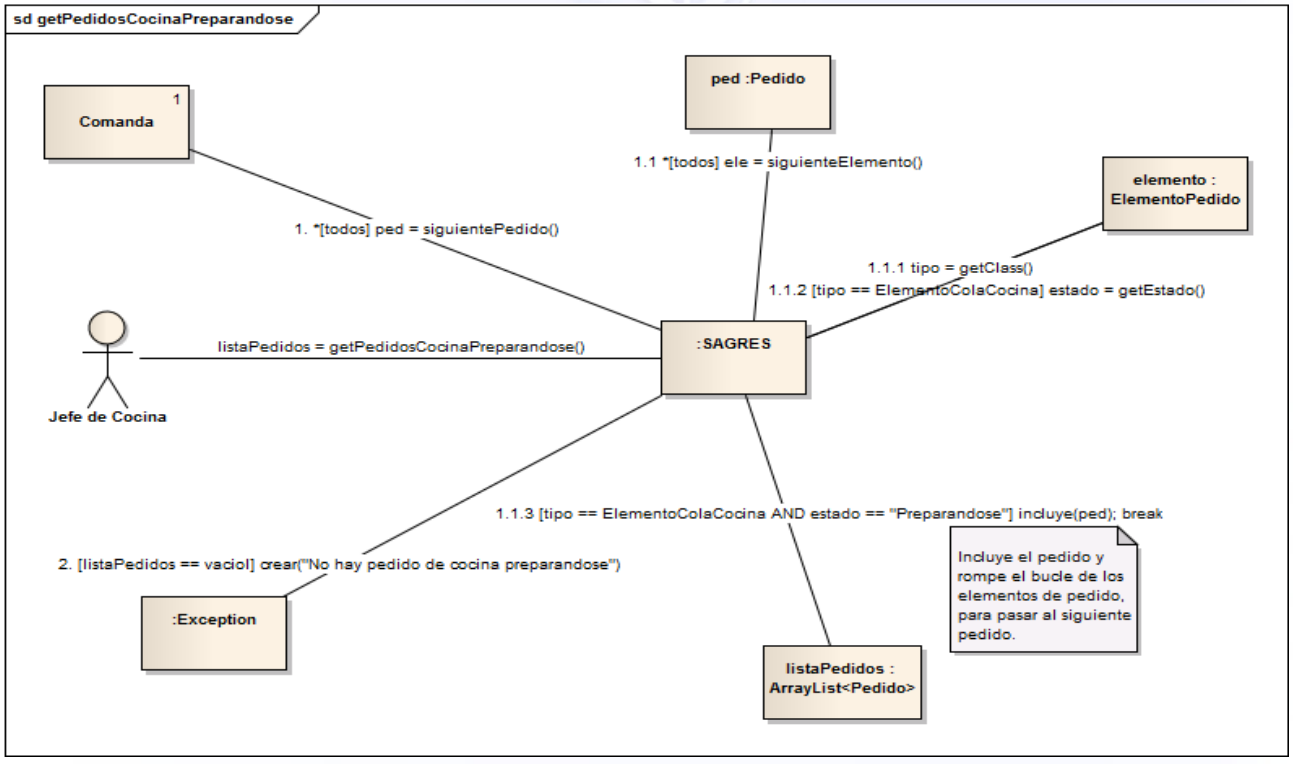
Nombre:	ConfirmaPagoFactura(codMesa)
Responsabilidad:	Confirmar el pago de una factura de la mesa con el código (codMesa) y buscar si hay otro(s) elementos(s) servido(s) después de la impresión de esta factura. En caso de que esto ocurra se imprima la factura nueva que contiene todos los elementos con el total.
Tipo:	sistema
Referencias:	Caso de uso “Cambia Estado Factura”
Nota:	
Excepción:	
Precondición:	
Postcondición:	• Cola de facturas actualizada.
Salida:	

Realizado por nabil

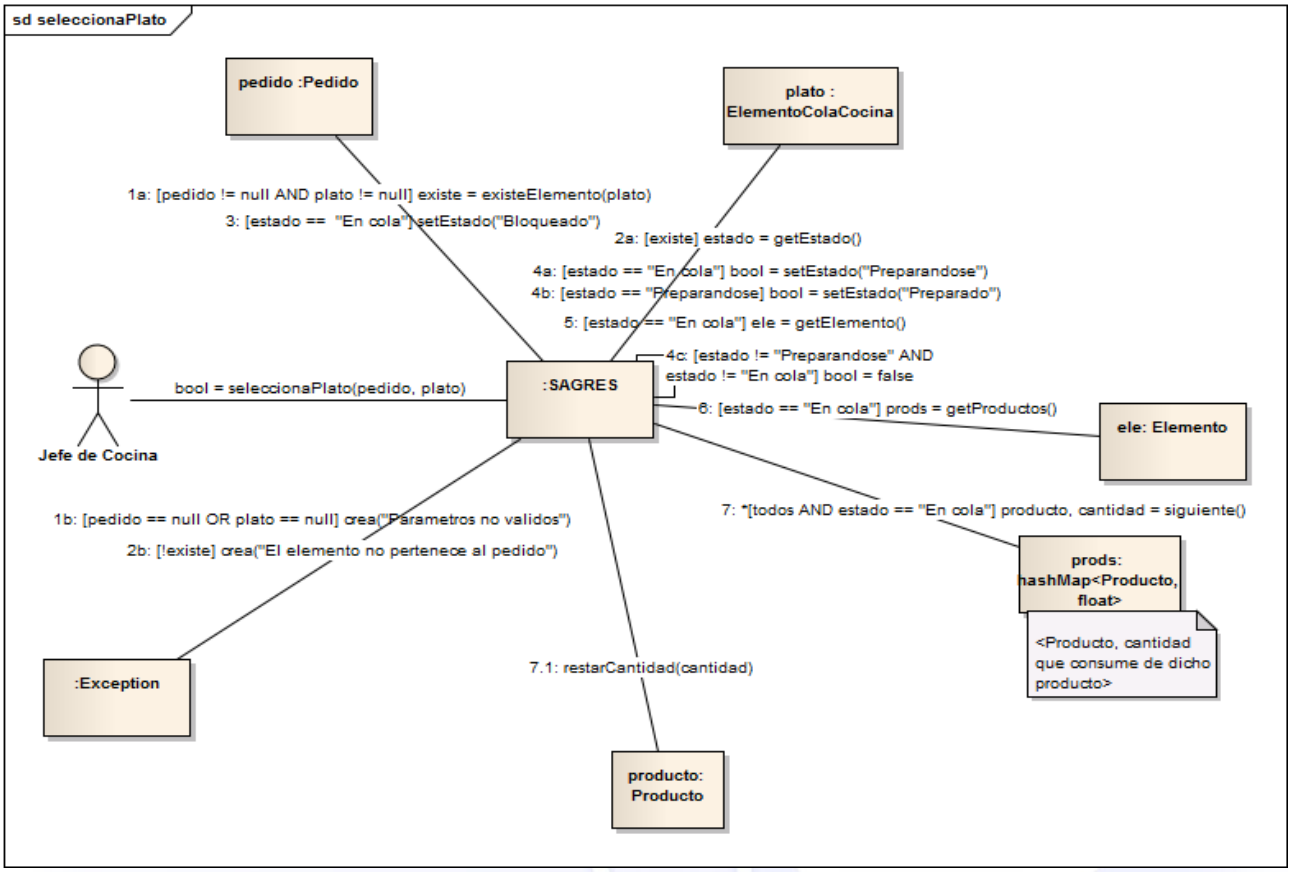
Diagramas de colaboración para cada contrato



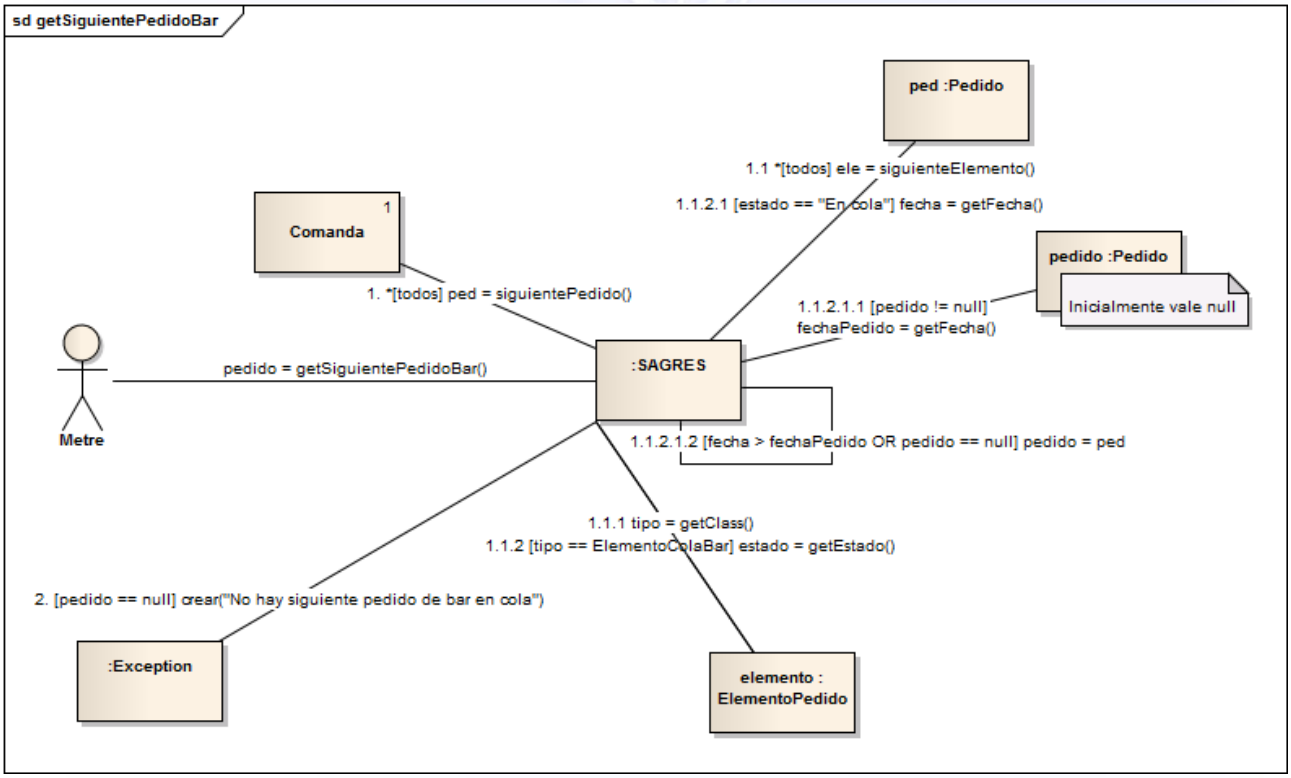
Realizado por Sergio Rodríguez Lumley



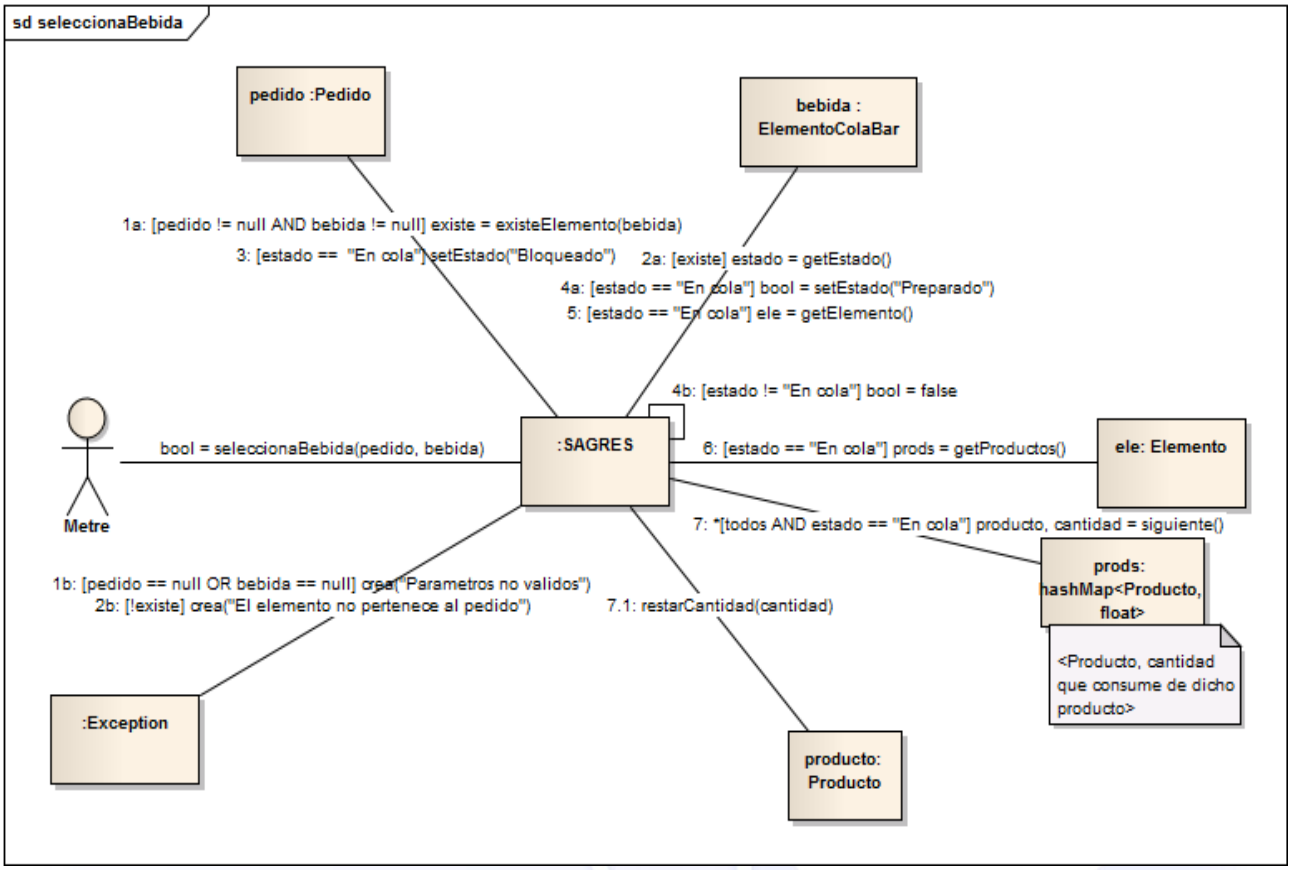
Realizado por Sergio Rodríguez Lumley



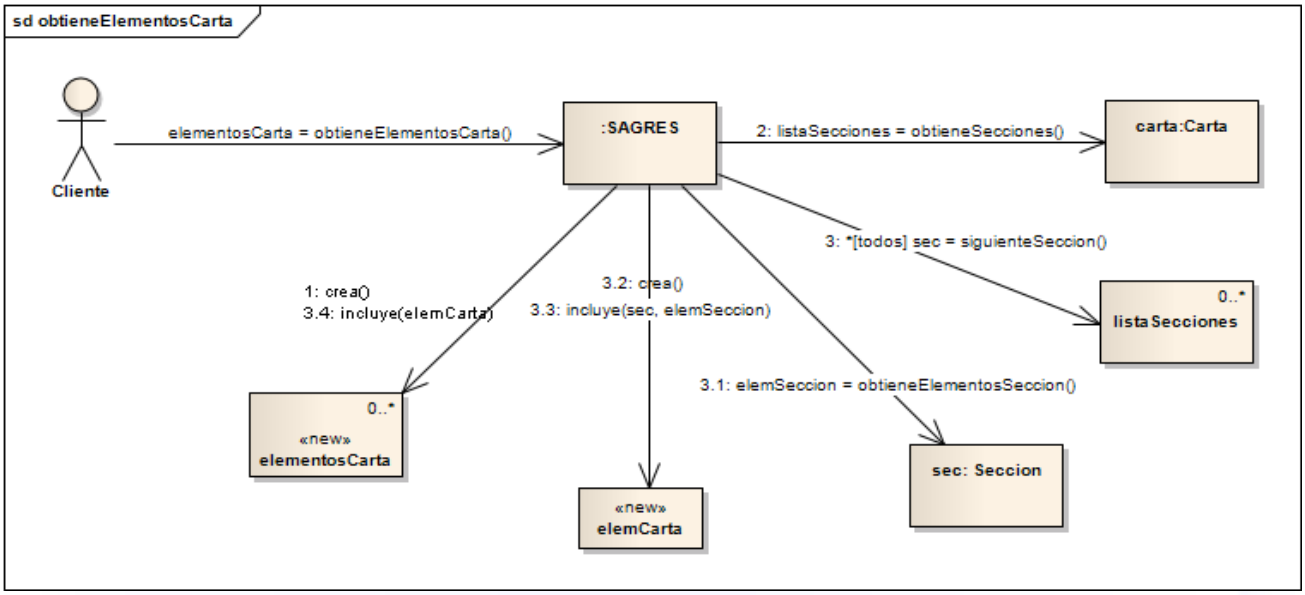
Realizado por Sergio Rodríguez Lumley



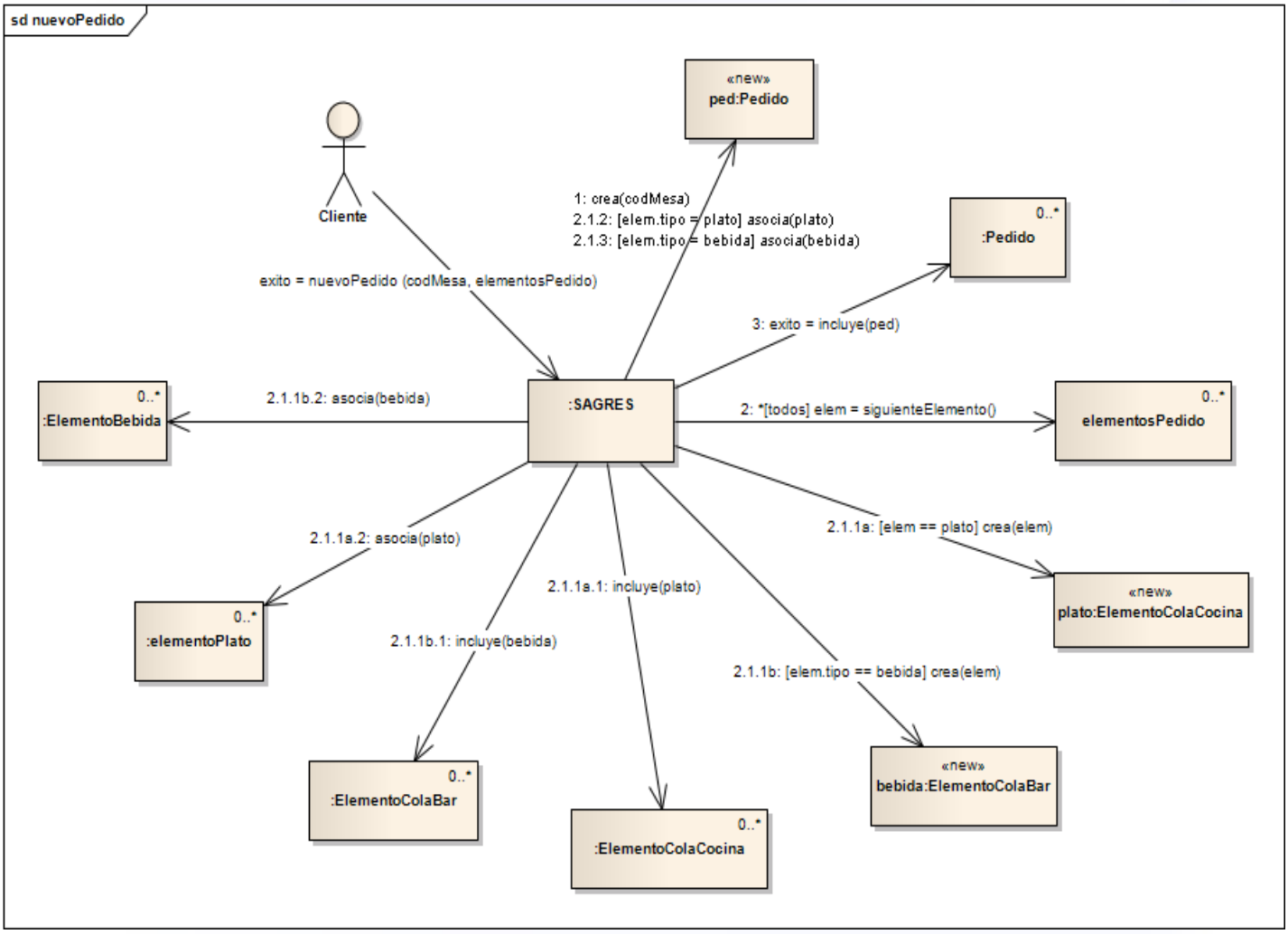
Realizado por Sergio Rodríguez Lumley



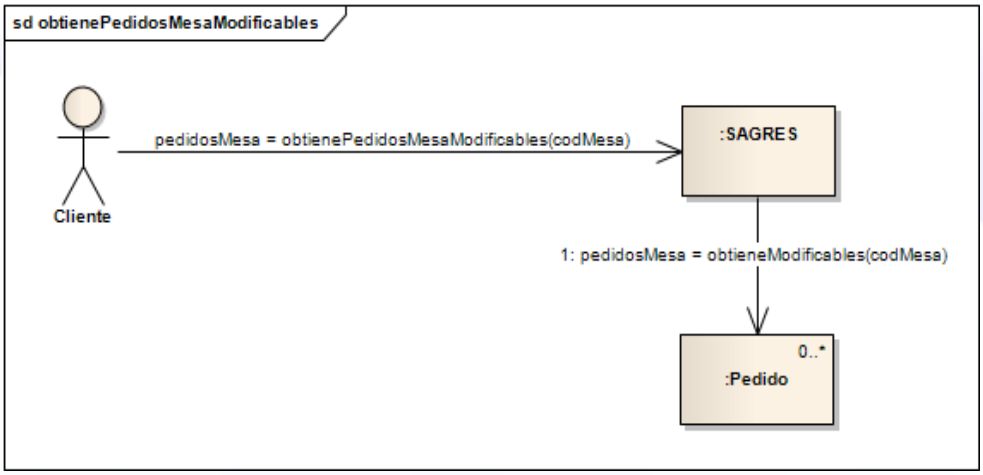
Realizado por Sergio Rodríguez Lumley



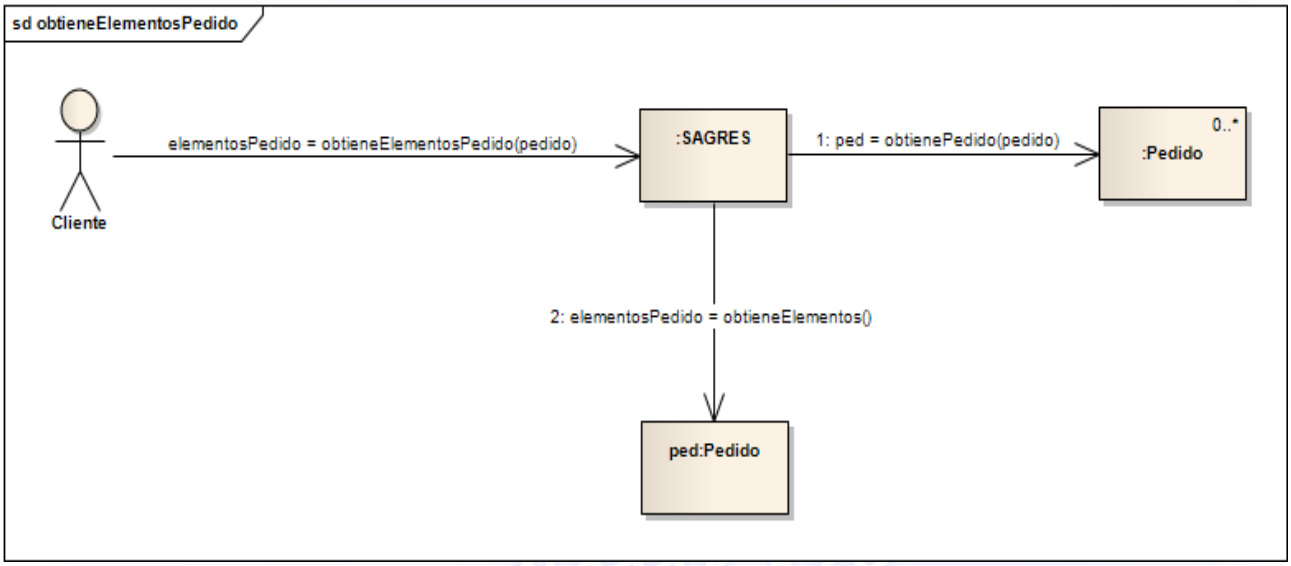
Realizado por Adrián Víctor Pérez Lopera



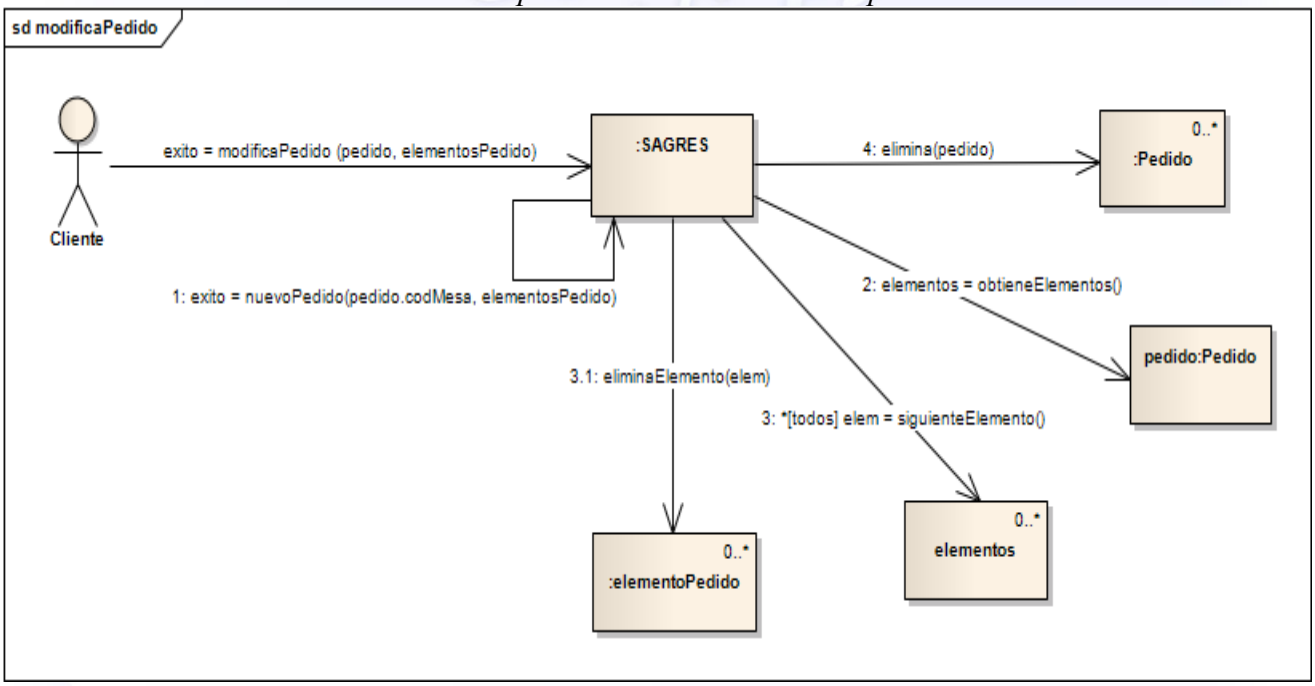
Realizado por Adrián Víctor Pérez Lopera



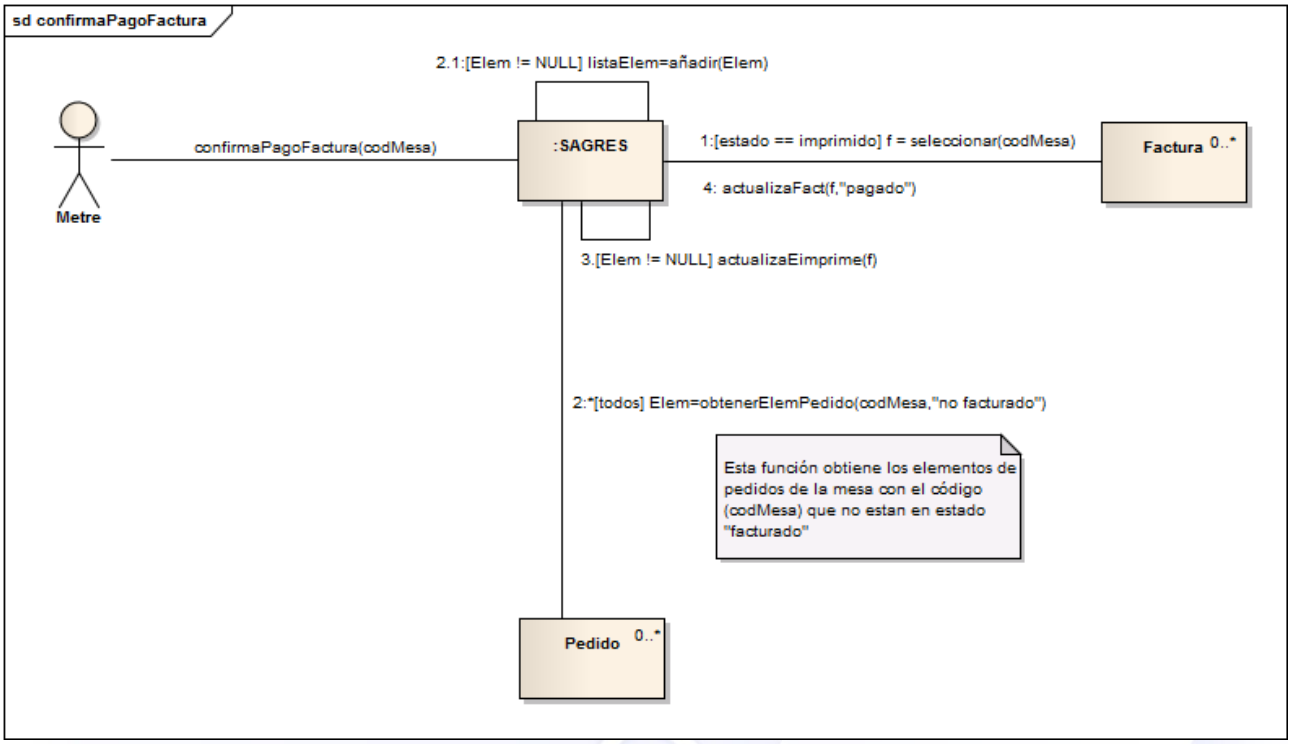
Realizado por Adrián Víctor Pérez Lopera



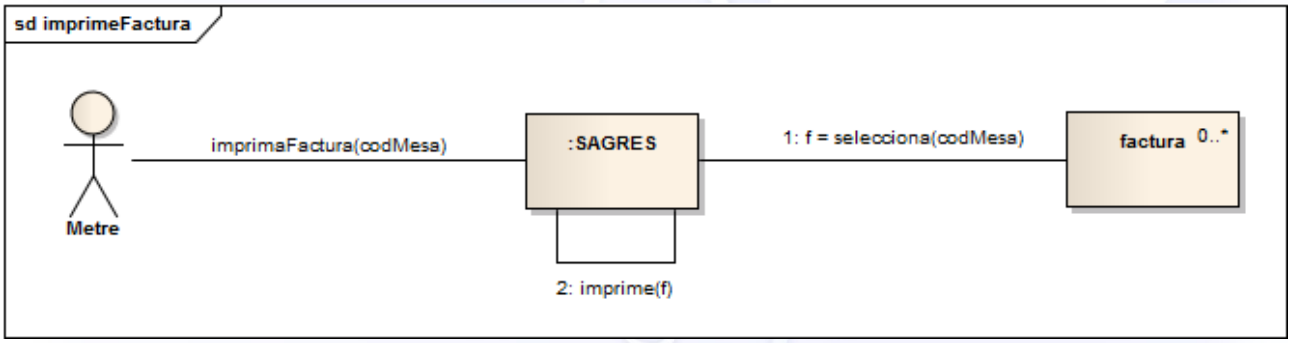
Realizado por Adrián Víctor Pérez Lopera



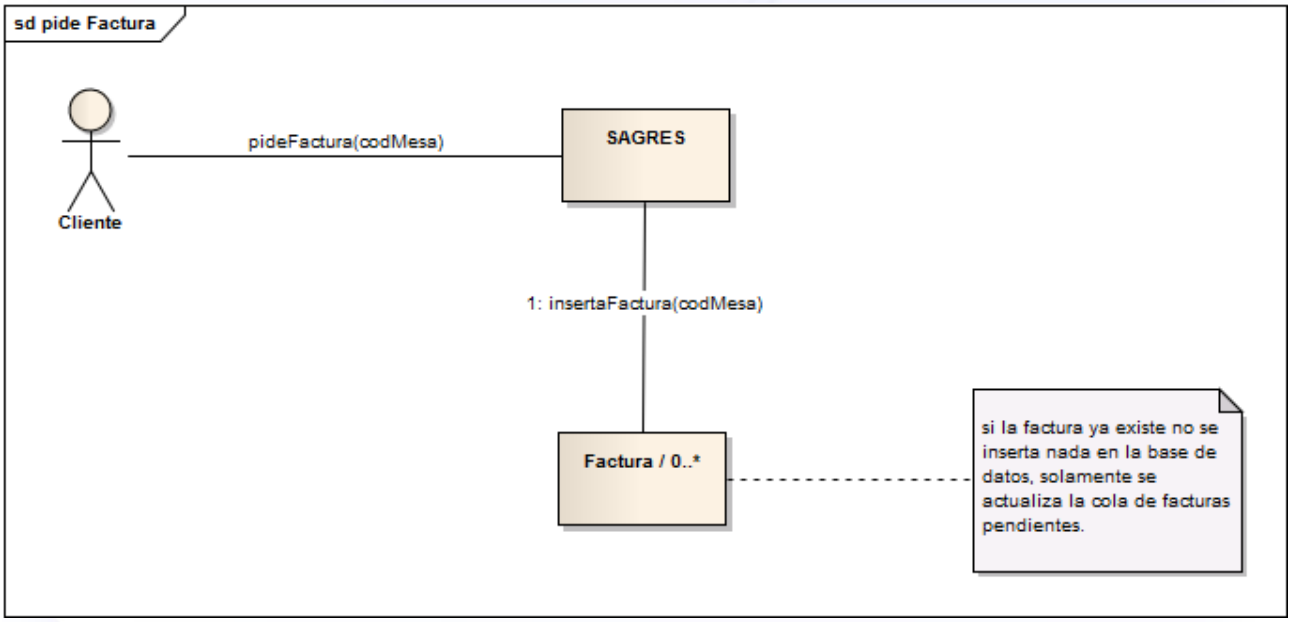
Realizado por Adrián Víctor Pérez Lopera



Realizado por Nabil Sabeg



Realizado por Nabil Sabeg



Realizado por Nabil Sabeg

Apéndice 1.0

Fecha	23/04/10
Descripción del problema	
Impacto del problema	
Soluciones adoptadas	• Se ha creado el documento de análisis.
Anexos a la versión	

Apéndice 1.1

Fecha	07/05/10
Descripción del problema	1. Los contratos y los diagramas de colaboración correspondientes a la realización de nuevos pedidos y a la modificación de los mismos por parte del cliente tenían fallos que afectaban a la lógica de la aplicación. 2. No se mostraban en los diagramas de colaboración de colas parte de la funcionalidad de las operaciones, al igual que en sus contratos. 3. Los cambios hechos en el modelado de requisitos llevaban a eliminar el contrato y diagrama de colaboración de "imprime Facturas". 4. Se ha añadido el diagrama de colaboración "pide factura" 5. La clase "Comanda" no aparece en ninguno de los diagramas de colaboración, por lo que no tiene ninguna utilidad. 6. Los cambios hechos en el DMR llevaban a eliminar los contratos y diagramas de colaboración de "iniciaNuevoPedido", "iniciaMofidicaPedido" y "obtieneElementos", además de añadir los contratos y diagramas de colaboración de "obtieneElementosCarta", "obtienePedidosMesaModificables" y "obtieneElementosPedido".
Impacto del problema	1. Se arrastraban incoherencias en ciertas operaciones a la hora de realizar el diseño. 2. Imposibilidad de conocer la funcionalidad correcta y posible error en la especificación del diseño. 3. Aparece una clase en el diagrama de clases que no se utiliza, con lo que se están introduciendo incoherencias en el sistema. 4. El análisis y el DMR no coincidían.
Soluciones adoptadas	• Se han corregido los contratos y diagramas de colaboración pertinentes. • Se ha añadido la funcionalidad completa a los diagramas de colaboración de colas, así como actualizadas sus funciones y contratos. • Se ha eliminado la clase "Comanda". • Se ha quitado el diagrama de colaboración y contrato de "imprime Facturas" • Se ha añadido el diagrama de colaboración "pide factura" • Se han quitado los diagramas de colaboración y los contratos de "iniciaNuevoPedido", "iniciaModificaPedido" y "obtieneElementos", y se han añadido los diagramas de colaboración y los contratos de "obtieneElementosCarta", "obtienePedidosMesaModificables" y "obtieneElementosPedido".
Anexos a la versión	