RESTAURANTE "EL TEJAO"

SEGUNDO ENTREGABLE DEL TRABAJO PRÁCTICO PARA LA ASIGNATURA IISSI (INTRODUCCIÓN A LA INGENIERÍA DEL SOFTWARE Y LOS SISTEMAS DE INFORMACIÓN)

MIEMBROS DEL GRUPO:

- ASEN RANGELOV BAYKUSHEV

TUTOR DEL GRUPO:

- JAVIER TROYA CASTILLA

I. Introducción del problema

I. 1 Descripción

Este proyecto trata de un restaurante ubicado en Isla Mayor, un municipio español de la provincia de Sevilla, Andalucía. El restaurante se llama "El Tejao". Es un restaurante de especialidades[1] que ofrece a precios asequibles una gran variedad de mariscos y pescados, de entre los que sobresale el cangrejo rojo, el cual es considerado el plato más típico de Isla Mayor. La cocina del restaurante refleja la cultura gastronómica de su entorno. Isla Mayor se encuentra en la llanura marismeña del río Guadalquivir y está situado muy cerca del parque natural de Doñana. Por tanto, los ingredientes con los que el restaurante "El Tejao" elabora sus platos provienen de la flora y la fauna silvestres de las marismas. Además, el municipio Isla Mayor es el primer productor de arroz en España y el tercer productor de cangrejo rojo a nivel mundial. Esta riqueza natural hace que "El Tejao" atraiga a muchos visitantes que desean probar la comida típica de este rincón de España.

"El Tejao" es un restaurante de cuarta clase[2] (1 tenedor). Está dirigido por una gerente, que es la persona clave del restaurante. Ella se encarga de la toma de decisiones y se ocupa de la parte administrativa de las operaciones que se realizan en el restaurante. Visto que "El Tejao" es un restaurante relativamente pequeño, en él no existe el puesto de responsable de compras. Es la gerente la que desempeña su función. Su misión principal como responsable de compras es procurar que no falte producto[3] en el almacén[4] del restaurante, pero también se ocupa de la buena calidad de los productos y de minimizar los costes de consumos.

La gerente del restaurante tiene a su disposición un equipo de cocineros profesionales. Son siete personas: tres cocineros de línea, que son

responsables directos de la producción de alimentos, dos cocineros auxiliares para la realización de tareas básicas como cortar verduras, limpiar vajilla, etc., un pastelero y un jefe de cocina, cuya tarea es dirigir el trabajo en la cocina[5] y supervisar a los demás cocineros. El jefe de cocina es la única persona de la cocina que tiene derecho a ir al almacén del restaurante para traer los productos que falten.

El restaurante dispone de un almacén en el que trabaja una persona, el encargado de almacén. Él se ocupa de que los artículos salgan sólo en presencia del jefe de cocina y los más antiguos del almacén sean los que se utilicen en primer lugar. Además, avisa a la responsable de compras cuando quedan pocas unidades de un cierto artículo.

En el servicio de atención al cliente trabajan 13 personas. 12 camareros, que trabajan por turnos, y un encargado de reservas[6], que atiende a los clientes, bien de forma presencial o por teléfono.

"El Tejao" también dispone de un equipo de limpieza profesional, que consta de seis personas.

I. 2 Situación actual

Desde sus comienzos por el año 1995, el restaurante "El Tejao" ha gozado de muy buenas opiniones por parte de sus clientes. El objetivo principal del restaurante siempre ha sido evolucionar y tratar de mejorar sus servicios. Últimamente, la gerente del restaurante ha hecho grandes avances a fin de popularizarlo. "El Tejao" ya tiene su propia página web y perfil en Facebook. Hoy en día, es el restaurante más popular y exitoso de Isla Mayor. "El Tejao" es colaborador de la película ganadora de 10 Goyas en los Goyas 2015, "La Isla Mínima". El año 2016 ha sido un año muy duro para el restaurante. A principios del año el Tribunal Supremo prohibió la pesca y la venta del cangrejo rojo, lo que supuso una gran amenaza para el negocio del restaurante. Sin embargo, en agosto esta sentencia ha sido anulada.

I. 3 Problemas

Actualmente "El Tejao" experimenta muchas dificultades con la gestión. Varios de los procesos de negocio del restaurante conllevan el uso de registros[7] para el almacenamiento de datos. La falta de sistema de información del restaurante obliga a que todos los trabajadores tengan que llevar estos registros manualmente, lo cual resulta muy incómodo y lioso, y más todavía a la hora de extraer los datos necesarios de ellos. Además, a veces se cometen errores. Por ejemplo, en mayo de 2014 una mesa se reservó para la misma fecha y hora a nombre de dos clientes distintos. Y el encargado de reservas se dio cuenta de esto dos horas antes de que los clientes llegasen. En julio de 2016 la gerente del restaurante, que también es la responsable de compras, calculó mal la cantidad de huevos que tenía que comprar para abastecer el almacén. Ella compró mucho más de la cuenta y, al final, cuando caducaron, tuvo que tirar más de la mitad a la basura.

I. 4 Expectativas

Es obvio que se necesita hacer reformas en el "El Tejao". La gerente del restaurante admite que es hora de que se introduzca un sistema de información que sustituya las hojas sueltas utilizadas en la mayoría de los procesos de negocio del restaurante. La gerente espera que dicho sistema de información les pueda facilitar el trabajo a ella y a todos sus empleados, y, lo que es más importante, que con la introducción del sistema de información se reduzcan los errores cometidos durante los procesos de negocio.

II. Glosario de términos

A continuación se ha mostrado el glosario de términos del proyecto. Las definiciones dadas sólo son válidas dentro del contexto del proyecto.

Término	Definición*	[i]
Almacén	Espacio físico del restaurante donde se guardan los productos que no caben en la cocina	[4]
Artículo/ producto	Ingredientes que se utilizan para la elaboración de los platos (huevos, sal, etc.) o alimentos que se sirven directamente al cliente (pan, bebidas, etc.)	[3]
Cancelación	La acción de anular una reserva previamente hecha	[15]
Carta	Compendio de todos los platos que ofrece el restaurante junto con sus respectivos precios	[9]
Cocina	Espacio físico del restaurante donde se los cocineros elaboran los platos y donde se guarda una pequeña parte de los productos	[5]
Cuarta clase	Según la clasificación de los restaurantes por tenedores, un restaurante de cuarta clase (o de 1 tenedor tiene el comedor independiente de la cocina, plaque inoxidable, loza irrompible, cristalería sencilla en buen estado de conservación, servilleta de tela o papel, servicios sanitarios decorosos y personal perfectamente aseado. Su carta o menú, aunque sencillo, ofrecerá platillos de no más de tres diferentes tiempos.	[2]
Cuenta	Recibo que indica el importe total que el cliente tiene que pagar por la comida consumida en el restaurante	[11]
Datos de contacto	Consisten en: el nombre, los apellidos y el número de teléfono del cliente	[13]
Datos de reserva	Consisten en: los datos de contacto, el pedido (si tiene) y los requisitos del cliente	[8]
Factura	Documento en el que se detalla los servicios recibidos junto con su importe que se entrega al cliente del restaurante	[14]
Fianza	Cantidad de dinero que el cliente paga por adelantado para asegurar el cumplimiento de su reserva en el restaurante. Se le devuelve al pagar la cuenta.	[12]
Pedido	La comida y las bebidas que el cliente quiere consumir en el restaurante	[10]

Registro	Conjunto de datos almacenados para su posterior consulta	[7]
Reserva	La acción de destinar una o varias mesas del restaurante para el uso exclusivo de una persona determinada	[6]
Restaurante de especialidades	Restaurante que ofrece una variedad limitada o estilo de cocina y muestra en su carta una extensa variedad de su especialidad, ya sean mariscos, aves, carnes o pastas, entre otros posibles.	[1]

III. Modelos de procesos de negocio

A continuación se muestran las descripciones en lenguaje natural de los distintos procesos de negocio del restaurante “El Tejao”. Sus diagramas BPMN se adjuntan en el apéndice

III. 1 Gestión del restaurante (descripción en lenguaje natural)

En este apartado se ha descrito en lenguaje natural el proceso de negocio del restaurante "El Tejao". Como se ve en el diagrama BPMN, algunas de las tareas más complejas de este proceso contienen subprocesos de negocio y, por tanto, tienen sus propios diagramas y descripciones.

El cliente empieza a sentir hambre y se dirige al restaurante "El Tejao". Nada más entrar, el cliente es recibido por el camarero. Acto seguido, el camarero le pregunta si tiene reserva. Si el cliente responde afirmativamente, el camarero comprueba su nombre y los datos de la reserva[8] (que se habían almacenado previamente en un registro durante el subproceso: "Hacer reserva" y se le habían proporcionado al camarero durante el subproceso "Confirmar la reserva"), acomoda al cliente en su mesa y le trae la carta[9].

Si el cliente contesta que no tiene reserva, el camarero le pregunta si quiere hacer una. Si el cliente dice que quiere hacer una reserva, se procede a la tarea "Hacer reserva". Dada la complejidad de la tarea "Hacer reserva", ésta se ha explicado detalladamente en el subproceso "Hacer reserva". Si el cliente contesta que no quiere hacer reserva, es decir, ha venido para comer sin haber reservado mesa, el camarero deja que el cliente elija la mesa a su gusto. Cuando el cliente se sienta, pide la carta y el camarero se la trae. El cliente empieza a leer la carta y pensar en qué comer. Esta tarea puede tardar mucho, de ahí que el camarero la interrumpa cada 10 minutos, preguntándole al cliente si ya ha elegido lo que va a pedir. Evidentemente, si la tarea ha sido interrumpida, el cliente contesta que no está listo. Cuando el cliente ya está listo, llama al camarero y él, a su vez, lo escucha y apunta su pedido[10]. Una vez que el camarero tiene el pedido del cliente, se lo lleva a la cocina y empieza a preparar la comida. Esta tarea es tan complicada que se ha explicado en otro subproceso "Llevar a cabo el pedido", pero conviene adelantar que no es el camarero el que realmente prepara la comida. El cliente queda a la espera de que el camarero le traiga la comida. Sin embargo, puede ocurrir que su paciencia se agote, en cuyo caso mete

prisa al camarero. Si esto sucede, el camarero inmediatamente lo calma, se disculpa por la tardanza y continúa la tarea "Llevar a cabo el pedido". Esto puede ocurrir varias veces, las mismas acciones se repiten, pero en realidad, como se ve en el diagrama, las quejas del cliente no surten ningún efecto sobre el subproceso "Llevar a cabo el pedido". En cuanto este subproceso termina y la comida está lista, el camarero se la trae al cliente. Luego, el camarero espera a que el cliente pida más comida (en cuyo caso se repiten los pasos anteriores) o pida la cuenta[11]. El camarero calcula la cuenta basándose en los datos del pedido que había apuntado antes. Si el cliente había reservado la mesa, la fianza[12] se le resta a la cuenta (más detalles en el subproceso "Hacer reserva)". Al final, el cliente recibe la cuenta, la paga y se va del restaurante.

III. 2 Hacer reserva (descripción en lenguaje natural)

En la descripción del proceso de negocio del restaurante muchas veces se hace referencia a las reservas. Su diagrama BPMN contiene una tarea "Hacer reserva", que realmente es un proceso de negocio distinto. Este subproceso se ha explicado de forma separada y mucho más detallada a continuación.

El cliente quiere hacer una reserva. Para esto, el restaurante le propone dos opciones: ir al restaurante y hablar en persona con el encargado de reservas, o bien llamarle por teléfono. Si va al restaurante, uno de los camareros lo va a dirigir al encargado de reservas y si llama por teléfono, se pone en contacto con él de forma directa. Una vez realizada la comunicación entre los dos, el cliente le explica toda la información acerca de la reserva que quiere hacer. Es obligatorio que el cliente le informe al encargado de reservas de la fecha y la hora de llegada al restaurante, así como del número de personas que van a asistir y de la cantidad de mesas que va a pedir. Luego, si quiere, el cliente puede añadir más requisitos (por ejemplo, puede requerir que la mesa esté al lado de la ventana o en el exterior, o que no

esté junto al aire acondicionado). Conforme el cliente proporciona la información, el encargado de reservas va comprobando en el registro si hay mesas disponibles que cumplan con todos los requisitos del cliente. Si se pueden satisfacer todos los requisitos, la tarea termina. Si el encargado de reservas se encuentra con que alguno de los requisitos no se puede cumplir, le pide al cliente que lo cambie (por ejemplo, le pide que cambie la hora de llegada porque tienen todas las mesas reservadas a esta hora). Si al cliente no le conviene cambiar ninguno de los requisitos, se despide y el subproceso "Hacer reserva" termina. Si está dispuesto a cambiarlos, le informa al encargado de reservas de los cambios, y él, a su vez, después de investigar los nuevos requisitos, le dice si ya se puede hacer la reserva o no. Esto se repite hasta que el encargado de reservas consiga encontrar mesas que satisfagan las necesidades del cliente o hasta que al cliente ya no le convenga retirar los requisitos.

Luego, el encargado de reservas le pregunta al cliente si quiere pedir la comida de antemano, para que cuando entre en el restaurante, toda la comida esté preparada. Si el cliente le contesta afirmativamente, el encargado de reservas le pide los datos del pedido y los almacena en el registro. Después, le pide los datos de contacto[13] (nombre, apellidos y número de teléfono). En caso contrario, le pide directamente los datos de contacto.

Luego, el encargado de reservas procede a almacenar los datos de contacto del cliente en el registro. Acto seguido, calcula la fianza que el cliente tiene que pagar para que se realice la reserva. La fianza es una cuantía que se fija por la gerente del restaurante, pero si el cliente ha decidido pedir la comida de antemano, a la hora de hacer la reserva, a esta cuantía fija se suma el 30% del valor del pedido. Tal como se ha explicado anteriormente, la fianza se le devuelve al cliente a la hora de pagar la cuenta en el restaurante. El encargado de reservas le envía al cliente la factura[14] de la fianza, el cliente la paga, y el encargado de reservas recibe el dinero.

No obstante, puede que al cliente le surjan circunstancias imprevistas que le impidan ir al restaurante en la fecha y a la hora acordada. Si se da esta condición, el cliente debe cancelar[15] la reserva para no perder la fianza. El cliente se pone en contacto con el encargado de reservas y los pasos se

repiten en orden inverso: el encargado de reservas borra del registro los datos de la reserva y le devuelve la fianza al cliente.

Si el cliente no cancela la reserva, a las 24 horas antes de la hora de reserva tendrá una última oportunidad para hacerlo sin perder la fianza. Pero esta vez es el encargado de reservas el que se pone en contacto con el cliente. Más detalles sobre la tarea "Confirmar la reserva" se explican a continuación.

III. 3 Confirmar la reserva (descripción en lenguaje natural)

Ya se ha visto que la tarea "Hacer reserva" es un subproceso del proceso de negocio del restaurante "El Tejao". Al final de este subproceso se realiza una tarea llamada "Confirmar la reserva", que implica la participación tanto del cliente como del encargado de reservas. Esta tarea, puesto que es compleja, contiene su propio proceso que hereda su nombre: el subproceso "Confirmar la reserva".

A las 24 horas antes de la hora de reserva, el encargado de reservas le llama al cliente a no ser que el cliente ya haya cancelado su reserva (esto ya se ha explicado en el subproceso "Hacer reserva"). Cuando el cliente contesta, el encargado de reservas le pregunta si la reserva sigue en pie. En caso de que el cliente conteste negativamente, el encargado de reservas borra del registro los datos de la reserva y le devuelve la fianza al cliente. El cliente recibe la fianza de vuelta y el proceso termina. Si el cliente contesta afirmativamente, es decir, confirma la reserva, el encargado de reservas extrae del registro los datos de contacto del cliente, su pedido (si tiene) y los requisitos sobre la mesa. Luego, le pasa todos estos datos a un camarero y al cabo de 24 horas borra del registro los datos de la reserva (porque se supone que el cliente ya se está atendiendo en el restaurante). El camarero que recibe los datos de la reserva espera 18 horas y a las 6 horas antes de

la hora de reserva empieza a preparar la mesa (la limpia y pone un letrero que indica que la mesa está reservada. Si la reserva del cliente incluye pedido de antemano, el camarero lo lleva a cabo). Una vez preparada la mesa, el cliente ya puede acomodarse en el restaurante. El proceso termina.

En caso de que el cliente confirme la reserva, pero después cambie de opinión, él puede llamar y cancelar la reserva. El encargado de reservas informa al camarero de la cancelación y si él ya había empezado a preparar la mesa, su tarea se interrumpe. Acto siguiente, el camarero borra los datos de la reserva, pero no le devuelve la fianza al cliente.

III. 4 Llevar a cabo el pedido (descripción en lenguaje natural)

En el proceso de negocio del restaurante se ha visto que el camarero desempeña una tarea que se llama "Llevar a cabo el pedido". A continuación se va a explicar el subproceso que contiene esta tarea. Además, se va a describir la gestión del almacén del restaurante "El Tejao".

Para que este subproceso comience, el camarero tiene que tener el pedido de cliente. Como ya se ha explicado anteriormente, el camarero apunta el pedido mientras el cliente pide la comida, o lo recibe por parte del encargado de reservas en caso de que el cliente haya querido pedir la comida al hacer la reserva.

Una vez que el camarero está en posesión del pedido, entra en la cocina del restaurante y se lo da a alguno de los cocineros. Nada más recibir el pedido, el cocinero comprueba si puede preparar la comida disponiendo únicamente de los productos alimenticios que en este momento estén en la cocina. Si se cumple la condición de que todos los productos necesarios estén en existencia en la cocina, el cocinero se pone a preparar la comida.

Cuando termina, le entrega la comida al camarero, que, a su vez, se la lleva de la cocina y se la sirve al cliente.

Sin embargo, es posible que en la cocina se hayan agotado las existencias de uno o más de los productos alimenticios necesarios, en cuyo caso, el cocinero rápidamente elabora una nota, en la cual indica los productos que necesita, así como sus cantidades. El cocinero le entrega esta nota al jefe de cocina, que va al almacén del restaurante para obtener los productos.

En el almacén el jefe de cocina le da la nota al encargado de almacén, cuya tarea es leerla, guardarla y proporcionarle al jefe de cocina los productos de la nota en sus respectivas cantidades, eligiendo entre todas las unidades que están disponibles en el almacén aquellas, cuya fecha de caducidad sea más próxima a la fecha actual. Si durante esta tarea el encargado de almacén se encuentra con que alguna de las unidades de un cierto tipo de producto está caducada, la tira a la basura.

Cuando el jefe de cocina recibe los productos que le faltan al cocinero, vuelve a la cocina y se los entrega. El cocinero ya puede empezar a preparar la comida. (Como se ve en el diagrama del modelo BPMN, esta tarea puede repetirse, es decir, puede ocurrir que el cocinero note que le falta otro producto, del que no se había dado cuenta antes, en cuyo caso vuelve a elaborar una lista y se la da al jefe de cocina).

Una vez que el encargado de almacén le ha proporcionado al jefe de cocina los productos de la nota, él comprueba si en el almacén todavía quedan suficientes unidades en existencia para los productos que acaba de entregar al jefe de cocina. En caso de que queden pocas unidades de ciertos productos, el encargado de reservas lo avisa a la responsable de compras.

Cada tres días la responsable de compras va al almacén del restaurante, recoge todas las notas que el encargado de almacén había guardado y a partir de ellas crea una lista de compras en la cual indica tanto los productos que hay que comprar como sus respectivas cantidades. Luego la responsable

de compras va a comprar los productos de la lista y vuelve al restaurante para abastecer primero la cocina y luego el almacén.

No obstante, si antes de que pasen tres días la responsable de compras recibe un aviso por parte del encargado de almacén de que quedan pocas unidades de ciertos productos, ella reacciona de forma inmediata, es decir, tan pronto como recibe el aviso, va al almacén, recoge las notas, elabora la lista y va de compras para evitar que el encargado de almacén se vea incapaz de proporcionarle al jefe de cocina los productos que el cocinero necesita.

IV. Visión general del sistema

IV. 1 Descripción general del sistema a desarrollar

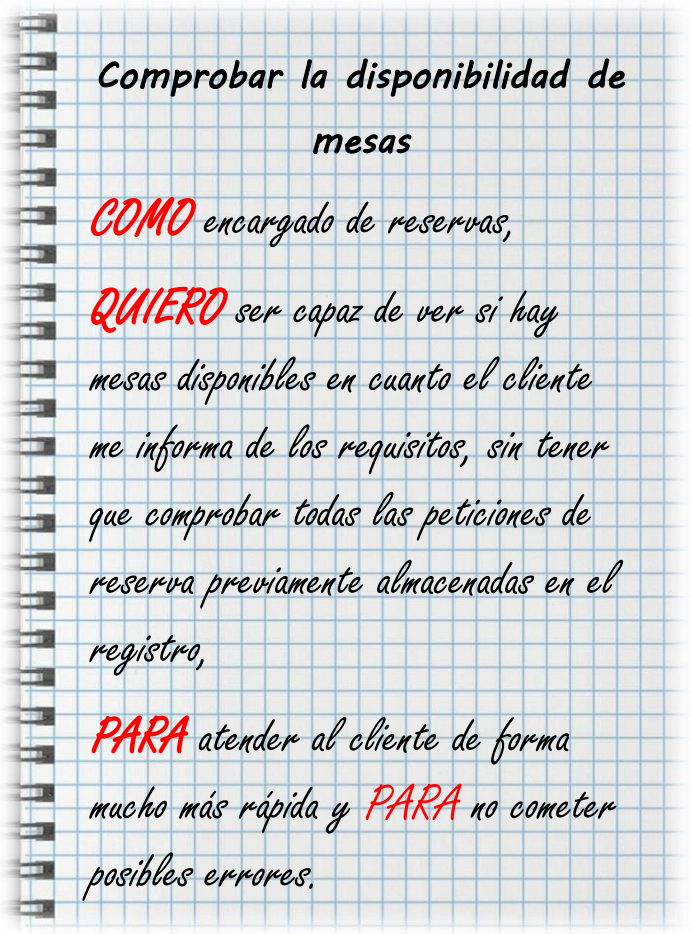
Como se ha visto en el apartado anterior, en la gestión del restaurante "El Tejao" se utilizan muchos registros. Estos registros sirven a los trabajadores para almacenar datos que luego van a necesitar para llevar a cabo varias tareas. No obstante, todos los registros del restaurante se llevan manualmente, es decir, los trabajadores escriben a mano los datos usando bolígrafos y hojas sueltas. Esto supone muchas dificultades para la posterior consulta, actualización o modificación de estos datos. Se pierde tiempo y, además, es probable que se cometan errores.

La idea principal de este proyecto es el desarrollo de un sistema de información que facilite todos los procesos de negocio, sustituyendo los registros llevados manualmente. La introducción de dicho sistema beneficiará a la mayoría de los trabajadores, otorgándoles no sólo mayor comodidad al llevar los registros, sino también opciones como consulta, actualización o modificación de datos, todo de forma automática. Más aún, les avisará cuando tienen que cumplir ciertas tareas.

IV. 2 Historias de usuario (El encargado de reservas)

Sin duda, al encargado de reservas le tocan las tareas más liosas. Cuando el cliente se pone en contacto con él y termina de explicar todos los requisitos que tiene sobre la mesa que quiere reservar, el encargado de reservas tiene

que ir comprobando, hoja a hoja, los datos de todas las reservas previamente hechas para asegurarse de que todos los requisitos del cliente se pueden satisfacer (por ejemplo, si el cliente quiere reservar una mesa en el exterior para cierta fecha y hora, el encargado de reservas debe comprobar que hay al menos una mesa disponible en el exterior para dicha fecha y hora. Si en el registro hay 30 reservas pendientes, el encargado de reservas tiene que revisar cada una de ellas antes de contestar al cliente). Está claro que el encargado de reservas tiene que pasar a usar un sistema de información para no perder el tiempo del cliente y no cometer errores.



Si el cliente decide pedir la comida al hacer la reserva, le va a costar bastante tiempo al encargado de reservas calcular la fianza y expedir la factura porque para hacer esto, él tiene que consultar los precios de cada plato que el cliente haya pedido. Esta tarea se puede suprimir utilizando el sistema de información.

Generar la factura de la fianza

COMO encargado de reservas,

QUIERO que el sistema de información me genere la factura de la fianza que tengo que enviar al cliente automáticamente,

PARA no perder tiempo en calcular la fianza y hacer la factura manualmente.

Tal como se ha explicado en el apartado anterior, a las 24 horas antes de la hora de reserva el encargado de reservas tiene que llamar al cliente para preguntarle si la reserva sigue en pie. Si en el registro hay 30 reservas pendientes, el encargado tiene que hacer 30 llamadas en fechas y horas distintas. La probabilidad de que se le olvide hacer alguna llamada es bastante alta. El sistema de información le puede ayudar, recordándole la hora a la que él tiene que llamar a determinado cliente.

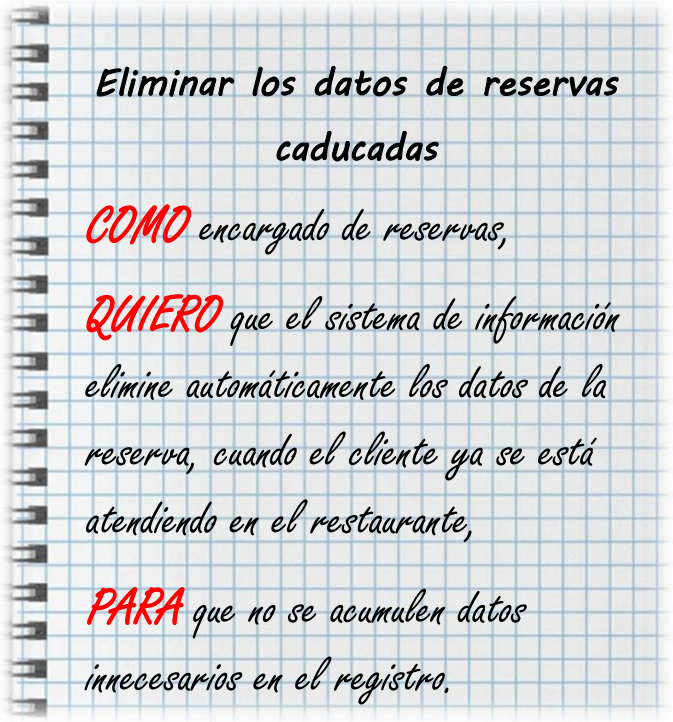
Avisar 24 antes de la hora de reserva

COMO encargado de reservas,

QUIERO que el sistema de información me avise a las 24 horas antes de la llegada de determinado cliente al restaurante,

PARA no olvidarme de llamar a dicho cliente para preguntarle si la reserva sigue en pie.

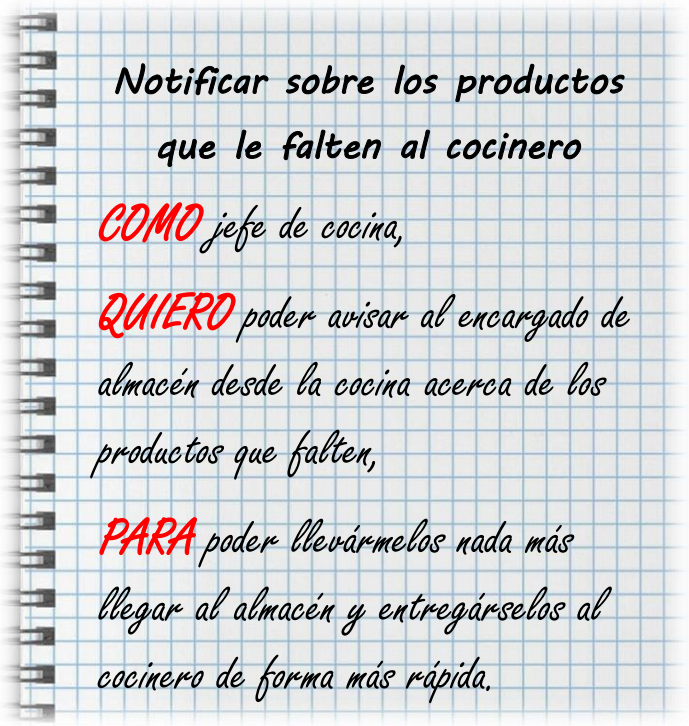
A menudo al encargado de reservas se le olvida borrar del registro los datos de reservas caducadas, que ya no sirven para nada, lo cual provoca que el registro se llene de datos innecesarios. Si el encargado de reservas deja las hojas y pasa a utilizar un sistema de información, los datos innecesarios se podrán borrar automáticamente.



IV. 3 Historias de usuario (El jefe de cocina)

Si en la cocina faltan productos necesarios para que el cocinero pueda preparar la comida, el cocinero toma nota de los productos que faltan y le da la nota al jefe de cocina. Después el jefe de cocina tiene que ir al almacén, entregarle al encargado de almacén esta nota, escrita a mano, y esperar a que el encargado de almacén la lea y le proporcione los productos. Es obvio que el jefe de cocina va a tardar bastante en darle al cocinero los productos. Sin embargo, es crucial no perder mucho tiempo porque mientras estas tareas se realizan, el cliente espera a que el camarero le traiga la comida. La introducción de un sistema de información puede hacer que estas tareas se realicen más rápido. Si el jefe de cocina utiliza un sistema de información

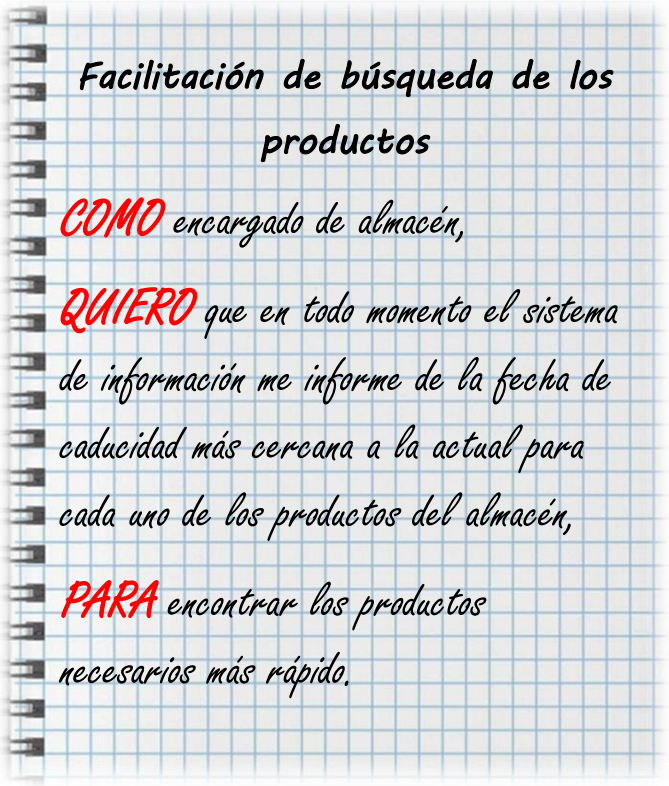
que le permita introducir los productos necesarios y sus respectivas cantidades, y después mandar esta información al encargado de almacén, se ahorrará mucho tiempo. Él tendrá que ir al almacén para obtener los productos, pero mientras tanto el encargado de almacén podrá buscarlos, de forma que cuando el jefe de cocina llegue al almacén, él podrá cogerlos y llevárselos a la cocina sin tener que esperar.



IV. 4 Historias de usuario (El encargado de almacén)

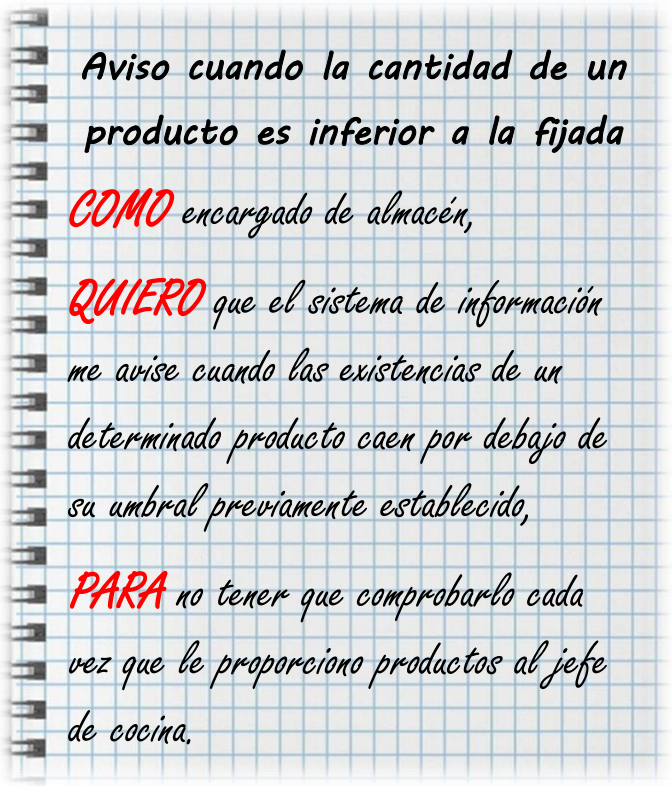
Cuando el encargado de almacén ha sido avisado de los productos que faltan en la cocina, él empieza a buscarlos para que el jefe de cocina se los lleve. Como se ha explicado en el apartado anterior, el encargado de almacén le proporciona al jefe de cocina las unidades que tengan la fecha de caducidad más cercana a la actual. Esto se hace para que, en la medida de lo posible, se evite tener que tirar productos alimenticios a la basura por estar

caducados. No obstante, esta tarea suele tardar mucho. Por ejemplo, si el encargado de almacén le tiene que proporcionar diez huevos al jefe de cocina, él primero tiene que comprobar las fechas de caducidad de todos los huevos en el almacén, encontrar la más cercana a la actual, y coger 10 huevos que tengan dicha fecha. Él encargado de almacén siempre va ordenando los productos a medida que la responsable de compras se los va suministrando, pero aun así le costará trabajo encontrar 10 huevos que tengan la fecha de caducidad más cercana porque él no sabe esta fecha. Y conviene recordar otra vez que el tiempo que él tarde en entregarle al jefe de cocina los productos afecta directamente al tiempo que el cliente tiene que esperar para que el camarero le sirva la comida. Lo que le ayudará mucho al encargado de almacén es un sistema de información que para todos los productos del almacén le diga cuál es la fecha de caducidad más cercana a la actual.



El encargado de almacén siempre debe estar atento a que no se agoten todas las unidades de un determinado producto. Por tanto, siempre que nota que quedan pocas unidades de un artículo, lo avisa a la responsable de compras. Sin embargo, a él le va a ser mucho más cómodo confiar en los servicios de un sistema de información. Sólo bastará con introducir una

cantidad mínima fijada para cada producto y si las existencias de un producto del almacén caen por debajo de su umbral, el sistema de información le avisará.



IV. 5 Historias de usuario (La responsable de compras)

Cada tres días la responsable de compras va al almacén del restaurante, recoge todas las notas que se habían acumulado, y a partir de ellas, elabora una lista de compras para saber qué artículos tiene que comprar y en qué cantidades. Son cuentas que resultan muy pesadas y ella las hace muy rápido, por lo que de vez en cuando comete errores y compra más o menos de lo que es necesario. Un sistema de información le permitirá sacar la lista de compras de forma inmediata, ya que para cada producto que ha salido del almacén desde el último abastecimiento se calculará automáticamente la cantidad a comprar.

Elaborar la lista de compras

COMO responsable de compras,

QUIERO que el sistema de información me informe del tipo y la cantidad de cada producto que ha salido del almacén desde su último abastecimiento,

PARA elaborar la lista de compras sin tener que hacer cuentas, evitando así cometer errores.

V. Catálogo de requisitos

V. 1 Requisitos de información

RI-001 Información sobre mesas:

Como encargado de reservas
Quiero disponer de la siguiente información sobre las *mesas* de las que dispone el restaurante: el número identificador de la mesa, la capacidad (para cuántas personas es), la ubicación (si la mesa está en el interior o el exterior del restaurante), si es para fumadores o no fumadores, el número de mesas que tienen las mismas susodichas características (salvo el identificador, que es distinto para cada mesa) y, además, para cada mesa del restaurante, las reservas que tiene asignadas
Para saber si hay mesas disponibles que satisfagan todas las necesidades de un determinado cliente que quiera hacer reserva en el restaurante.

RI-002 Información sobre clientes:

Como encargado de reservas
Quiero disponer de la siguiente información sobre los *clientes* del restaurante. En concreto, el nombre, los apellidos y el número de teléfono
Para poder relacionar cada mesa reservada con un determinado cliente y poder ponerme en contacto con dicho cliente cuando hace falta.

RI-003 Información sobre peticiones de reserva:

Como encargado de reservas
Quiero disponer de la información necesaria para poder relacionar a un determinado cliente con una mesa del restaurante (o varias) que cumpla todos los requisitos de dicho cliente. En concreto: el cliente que hace la petición, la fecha para la que dicho cliente desea reservar mesa, la hora, el número de personas que van a asistir y el número de mesas que quiere reservar. Además, si el cliente exige que la mesa esté en el interior o el exterior del restaurante, que sea para fumadores o no fumadores o quiere incluir el pedido en la petición de reserva, se debe conocer dicha información.
Para ver con claridad cuáles son los requisitos del cliente y poder comprobar si dichos requisitos se pueden cumplir.

RI-004 Información sobre reservas:

Como encargado de reservas
Quiero disponer de la siguiente información sobre las *reservas*: el código, la fecha, la hora, el cliente que ha hecho la reserva y la mesa reservada
Para saber cuáles son las reservas asignadas a cada mesa en todo momento.

RI-005 Información sobre pedidos:

Como encargado de reservas
Quiero disponer de la siguiente información sobre los *pedidos* en caso de que los clientes quieran pedir la comida al hacer la reserva. Se debe conocer el código y, para cada plato que pida el cliente, el nombre, la ración pedida (media o completa), las unidades pedidas y el precio de una unidad.
Para poder consultar dicha información cuando me hace falta.

RI-006 Información sobre facturas:

Como encargado de reservas
Quiero disponer de la siguiente información sobre las *facturas*: la fecha de elaboración, el código, la cuantía fija, la reserva a la que corresponde y, además, si dicha petición de reserva incluye pedido, la factura tiene que contener toda la información del pedido (salvo el código)
Para poder calcular las fianzas y expedir las facturas de manera automática.

RI-007 Información sobre productos:

Como encargado de almacén
Quiero disponer de la siguiente información sobre los *productos*: el nombre, la unidad en la que se mide (los productos alimenticios del almacén se suelen medir en unidades, gramos, kilogramos o litros), el umbral bajo el cual es necesario avisarle a la responsable de compras que quedan pocas existencias. También me gustaría tener los productos del almacén agrupados según sus fechas de caducidad y saber cuántos productos hay en cada grupo.
Para manejar la gestión del almacén de forma mucho más fácil.

RI-008 Información sobre notas:

Como jefe de cocina
Quiero disponer de la siguiente información sobre las *notas* que utilizan los cocineros para informarle al encargado de almacén de los productos que faltan en la cocina. En concreto: el código de la nota, la fecha en la que se ha elaborado y para cada producto que falte en la cocina, la cantidad que el encargado de almacén tiene que proporcionar.
Para poder reaccionar rápidamente ante la falta de productos en la cocina.

RI-009 Información sobre listas de compras:

Como responsable de compras
Quiero disponer de la siguiente información sobre las *listas de compras*: el código, la fecha en la que se ha creado, la fecha de referencia y todas las notas que se han elaborado desde la fecha de la última compra. “Fecha de referencia” es un término que me he inventado yo y representa la última vez que fui de compras.
Para saber los productos que hay que comprar y en qué cantidades.

V. 2 Reglas de negocio

RN-001 Margen de 4 horas entre reservas:

Como encargado de reservas
Quiero que un cliente pueda reservar una mesa del restaurante sólo si hay al menos 4 horas de diferencia entre la hora a la que dicho cliente quiera reservar la mesa y la hora de las demás reservas que tenga asignadas la mesa
Para evitar que un cliente se encuentre con que al llegar al restaurante la mesa que ha reservado está ocupada o evitar tener que meterle prisa a un cliente cuando la mesa que ha reservado tiene asignada otra reserva.

RN-002 Metodología para calcular fianzas:

Como encargado de reservas
Quiero que se cumpla la siguiente regla de negocio: se expide una factura a un determinado cliente sin importar el número de mesas que haya reservado. El coste de la fianza es una cuantía fijada por la gerente del restaurante (aunque esta cuantía puede cambiar en el futuro). En caso de que el cliente haya pedido la comida al hacer la reserva, a esta cuantía se le suma el 30% del valor del pedido.
Para respetar las normas que ha establecido la gerente.

RN-003 Metodología para crear listas de compras:

Como responsable de compras
Quiero que se cumpla la siguiente regla de negocio: en una lista de compras se anotan los productos que estén presentes solamente en las notas elaboradas desde la fecha de la última compra, así como sus respectivas cantidades a comprar. No se incluyen productos de notas anteriores.
Para mantener el almacén abastecido siempre de la misma cantidad de productos en la medida de lo posible.

V. 3 Requisitos funcionales

RF-001 Informes sobre la disponibilidad de mesas:

Como encargado de reservas
Quiero que el sistema de información me informe de las mesas del restaurante que cumplan todos los requisitos del cliente y estén disponibles para la fecha y la hora elegida
Para decidir si puedo acomodar al cliente en una mesa del restaurante.

RF-002 Informes sobre reservas:

Como encargado de reservas
Quiero poder consultar la fecha y la hora de las reservas que una mesa tenga asignada
Para responder al cliente en caso de que me pregunte cuándo una determinada mesa estará libre

RF-003 Informes sobre pedidos:

Como encargado de reservas
Quiero imprimir los pedidos de los clientes que deciden pedir la comida de antemano
Para luego dárselos a los camareros (para que ellos los lleven a cabo).

RF-004 Cálculo de fianzas:

Como encargado de reservas
Quiero calcular automáticamente la fianza que un determinado cliente tiene que pagar
Para ahorrar tiempo y no cometer errores.

RF-005 Elaboración de facturas:

Como encargado de reservas
Quiero imprimir la factura correspondiente a un determinado cliente
Para dársela.

RF-006 Avisos de reservas pendientes:

Como encargado de reservas
Quiero recibir aviso a las 24 horas antes de la llegada de determinado cliente al restaurante
Para no olvidarme de llamar a dicho cliente para preguntarle si la reserva sigue en pie.

RF-007 Avisos de reservas pendientes:

Como encargado de reservas
Quiero poder borrar toda la información relacionada con reservas inactuales (facturas, pedidos, etc.). Sin embargo, **quiero** llevar un registro con todos los clientes del restaurante que han hecho reservas
Para evitar tomar nota de los datos de clientes que visitan muy a menudo nuestro restaurante

RF-008 Actualización de la información sobre las existencias:

Como encargado de almacén
Quiero consultar la información actualizada sobre las existencias de cada producto del almacén
Para saber cuántas unidades quedan de cada producto y cuáles son sus respectivas fechas de caducidad después de cada entrega de productos al jefe de cocina.

RF-009 Avisos de productos caducados:

Como encargado de almacén
Quiero recibir aviso si hay productos caducados en el almacén
Para tirarlos a la basura.

RF-010 Avisos de falta de existencias:

Como encargado de almacén
Quiero recibir aviso cuando las existencias de un determinado producto caen por debajo de su umbral establecido
Para no tener que comprobarlo cada vez que le proporciono productos al jefe de cocina.

RF-011 Informes de fechas de caducidad:

Como encargado de almacén
Quiero que el sistema de información me informe de la fecha de caducidad más cercana a la actual para las unidades de cualquier producto del almacén
Para entregarle al jefe de cocina siempre aquellas unidades cuya fecha de caducidad es más cercana a la actual y, de esta forma, evitar tener que tirar productos a la basura.

RF-012 Elaboración de listas de compras:

Como responsable de compras
Quiero que el sistema de información cree la lista de compras automáticamente de tal forma que incluya los productos a comprar y sus respectivas cantidades basándose en las notas elaboradas por los cocineros desde la fecha de la última compra
Para no tener que elaborarla yo y evitar que se cometan errores.

V. 4 Requisitos no funcionales

RNF-001 Avisos de reservas pendientes:

Como encargado de almacén
Dado que la responsable de compras siempre compra las mismas marcas de productos y casi siempre los envases son iguales, para nosotros, los trabajadores, será mucho más fácil que para algunos productos se utilice el tipo de envase como unidad de medida. Por lo tanto, **quiero** que además de las unidades de medida: litro, gramo, kilogramo y unidad, se utilizan también los términos: “bote”, “botella”, “lata”, “paquete” y “bolsa”
Para referirnos a las cantidades de los productos más fácilmente.

VI. Pruebas de aceptación

RN-001 Margen de 4 horas entre reservas:

Como encargado de reservas
Quiero que un cliente pueda reservar una mesa del restaurante sólo si hay al menos 4 horas de diferencia entre la hora a la que dicho cliente quiera reservar la mesa y la hora de las demás reservas que tenga asignadas la mesa
Para evitar que un cliente se encuentre con que al llegar al restaurante la mesa que ha reservado está ocupada o evitar tener que meterle prisa a un cliente cuando la mesa que ha reservado tiene asignada otra reserva.

Pruebas de aceptación:

- ✓ El encargado de reservas asocia una petición de reserva para la fecha 25/12/2016 a las 18:00 con una mesa que tiene asignada reserva para la fecha 25/12/2016 a las 14:01. Debe salir mensaje de error.
- ✓ El encargado de reservas asocia una petición de reserva para la fecha 25/12/2016 a las 12:00 con una mesa que tiene asignada reserva para la fecha 25/12/2016 a las 15:59. Debe salir mensaje de error.
- ✓ El encargado de reservas asocia una petición de reserva para la fecha 25/12/2016 a las 18:00 con una mesa que tiene asignada reserva para la fecha 24/12/2016 a las 18:00. No tiene que salir ningún mensaje de error.

RN-002 Metodología para calcular fianzas:

Como encargado de reservas
Quiero que se cumpla la siguiente regla de negocio: se expide una factura a un determinado cliente sin importar el número de mesas que haya reservado. El coste de la fianza es una cuantía fijada por la gerente del restaurante (aunque esta cuantía puede cambiar en el futuro). En caso de que el cliente haya pedido la comida al hacer la reserva, a esta cuantía se le suma el 30% del valor del pedido.
Para respetar las normas que ha establecido la gerente.

Pruebas de aceptación:

- ✓ Se crea una factura con cuantía fija distinta a la establecida por la gerente. Debe salir mensaje de error.
- ✓ La gerente ha establecido que la cuantía sea de 10€. Se crea una factura con cuantía fija de 10€. El cliente para el que se expide la factura ha pedido una ración de ensalada mixta (que vale 3€) y una ración de pulpo a la gallega (que vale 10€). El coste de la fianza debe ser 6,90€.

RN-003 Metodología para crear listas de compras:

Como responsable de compras
Quiero que se cumpla la siguiente regla de negocio: en una lista de compras se anotan los productos que estén presentes solamente en las notas elaboradas desde la fecha de la última compra, así como sus respectivas cantidades a comprar. No se incluyen productos de notas anteriores.
Para mantener el almacén abastecido siempre de la misma cantidad de productos en la medida de lo posible.

Pruebas de aceptación:

- ✓ Hay tres notas. La primera se ha creado hace 5 días, la segunda – hace 3 días y la tercera – ayer. La última vez que la responsable de compras fue a hacer la compra era hace 4 días. En la lista de compras deben aparecer los productos de la segunda y la tercera nota con sus respectivas cantidades.

VII. Modelo conceptual

Los conceptos relevantes del dominio del problema se han representado mediante diagramas UML. La información necesaria para elaborar los diagramas se extrae del catálogo de requisitos (en concreto: los requisitos de información y las reglas de negocio). Para mayor legibilidad, se han creado tres distintos diagramas UML. Dos de ellos representan conceptos relacionados con la reserva de mesas en el restaurante y son conexos. El tercero representa la gestión de almacén del restaurante y no tiene ninguna conexión con los otros dos.

Los diagramas UML, junto con los distintos escenarios de prueba, se adjuntan en el apéndice.

VIII. Matriz de trazabilidad

	RI-001	RI-002	RI-003	RI-004	RI-005	RI-006	RI-007	RI-008	RI-009	RN-001	RN-002	RN-003
Cliente		X										
MesaConcreta	X											
Mesa	X											
PeticiónReserva			X									
Reserva				X								
Pedido					X							
LíneaDePedido					X							
Plato					X							
Factuta						X						
Producto							X					
Nota								X				
LíneaDeNota								X				
ListaDeCompras									X			
GrupoProductos							X					
representa	X											
tiene	X											
esRealizadaPor				X								
hace			X									
referencia					X							
incluye						X						
correspondeA						X						
vaDirigidaA						X						
Comp. entre Pedido y Factura						X						
Comp. entre LíneaDePedido y Plato					X							
Comp. entre LíneaDePedido y Pedido					X							
esUnGrupoDe							X					
haceReferenciaA								X				
Comp. entre LíneaDeNota y Nota								X				
Comp. entre Nota y ListaDeCompras									X			
número de mesas										X		
cuantíaFija											X	
composición de una lista de compras												X

IX. Modelo relacional en 3FN

Relaciones	PK	AK	FK
Clientes (OID_C, nombre, apellidos, teléfono)	OID_C	(nombre, apellidos, teléfono)	
PeticionesReserva (OID_PR, fechaYHoraReserva, numPersonas, ubicacionMesa, tipoMesa, OID_C)	OID_PR		OID_C / Clientes
Mesas (OID_M, maxPersonas, ubicación, tipo, numMesasConcretas)	OID_M	(maxPersonas, ubicación, tipo)	
MesasConcretas (identificador, OID_M)	identificador		OID_M / Mesas
Reservas (código, fechaYHora, identificador, OID_C)	código	(fechaYHora, identificador)	identificador / MesasConcretas OID_C / Clientes
Facturas (códigoFactura, fechaElaboración, cuantíaFija, código)	códigoFactura		código / Reservas
Platos (nombre, precioMediaRación, precioRación)	nombre		
Pedidos (códigoPedido, OID_PR, códigoFactura)	códigoPedido	(OID_PR), (códigoFactura)	OID_PR / PeticionesReserva códigoFactura / Facturas
LíneasDePedido (códigoPedido, nombre, raciónPedida, unidadesPedidas)	códigoPedido, nombre, raciónPedida		códigoPedido / Pedidos nombre / Platos

Relaciones	PK	AK	FK
Productos (nombre, unidadMedida, umbralExistencias)	nombre		
GruposDeProductos (OID_GP, nombre, fechaCaducicad, cantidadExistencia)	OID_GP	(nombre, fechaCaducidad)	nombre / Productos
ListasDeCompras (códigoLista, fechaCreación)	códigoLista	fechaCreación	
Notas (códigoNota, fechaElaboración, códigoLista)	códigoNota		códigoLista / ListasDeCompras
LíneasDeNota (nombre, cantidad, códigoNota)	nombre, códigoNota		códigoNota / Notas

- No se identificaron jerarquías en el modelado conceptual, por lo que en los diagramas no aparecen clasificaciones;

- Se le ha propuesto al encargado de reservas la posibilidad de que el sistema de información automáticamente escoja cuál es la mesa más adecuada para una determinada reserva entre todas las mesas que cumplan los requisitos del cliente y realice la reserva sobre ella. Sin embargo, se ha hablado con el encargado de reservas y él prefiere tomar estas decisiones por sí mismo. Es decir, es el encargado de reservas el que va a elegir una mesa y la va a reservar. El sistema de información no debería quitarle este derecho.

- Se le ha propuesto a la responsable de compras que ya no se utilicen fechas de referencia en las listas de compras. En vez de usar fechas de referencia, el sistema de información va a borrar las listas de compras antiguas y las notas que las compongan cada vez que se cree nueva lista de compras. Ella se va a elaborar a partir de todas las notas existentes (esto es, que no hayan sido borradas). La responsable de compras está de acuerdo.

X. Modelo tecnológico

X. 1 Scripts de creación de tablas y restricciones

```
CREATE TABLE Clientes (
OID_C INTEGER NOT NULL,
Nombre VARCHAR2(30) NOT NULL,
Apellidos VARCHAR2(50) NOT NULL,
Telefono CHAR(9) NOT NULL,
PRIMARY KEY(OID_C),
UNIQUE(Nombre, Apellidos, Telefono)
);

CREATE TABLE PeticionesReserva (
OID_PR INTEGER NOT NULL,
fechaYHoraReserva TIMESTAMP NOT NULL,
numPersonas SMALLINT NOT NULL,
ubicacionMesa CHAR(8) CHECK(ubicacionMesa IN ('interior', 'exterior')),
tipoMesa VARCHAR2(11) CHECK (tipoMesa IN ('fumadores', 'noFumadores')),
OID_C INTEGER NOT NULL,
PRIMARY KEY (OID_PR),
FOREIGN KEY (OID_C) REFERENCES Clientes
);

CREATE TABLE Mesas (
OID_M INTEGER NOT NULL,
maxPersonas SMALLINT NOT NULL,
ubicacion CHAR(8) CHECK(ubicacion IN ('interior', 'exterior')) NOT NULL,
tipo VARCHAR2(11) CHECK (tipo IN ('fumadores', 'noFumadores')) NOT NULL,
numMesasConcretas SMALLINT  DEFAULT 0 NOT NULL,
PRIMARY KEY (OID_M),
UNIQUE (maxPersonas, ubicacion, tipo)
);
```

```
CREATE TABLE MesasConcretas (
identificador INTEGER NOT NULL,
OID_M INTEGER NOT NULL,
PRIMARY KEY (identificador),
FOREIGN KEY (OID_M) REFERENCES Mesas
);

CREATE TABLE Reservas (
codigo VARCHAR2(6)  NOT NULL,
fechaYHora TIMESTAMP NOT NULL,
identificador INTEGER NOT NULL,
OID_C INTEGER NOT NULL,
PRIMARY KEY (codigo),
UNIQUE (fechaYHora, identificador),
FOREIGN KEY (identificador) REFERENCES MesasConcretas,
FOREIGN KEY (OID_C) REFERENCES Clientes
);

CREATE TABLE Facturas (
codigoFactura VARCHAR2(6) NOT NULL,
fechaElaboracion DATE NOT NULL,
cuantiaFija NUMBER DEFAULT 10.00 NOT NULL,
codigo VARCHAR2(6),
PRIMARY KEY (codigoFactura),
FOREIGN KEY (codigo) REFERENCES Reservas  ON DELETE CASCADE

);

CREATE TABLE Platos (
nombre VARCHAR2(60) NOT NULL,
precioMediaRacion NUMBER(2),
precioRacion NUMBER(2) NOT NULL,
PRIMARY KEY (nombre)
);

CREATE TABLE Pedidos (
codigoPedido VARCHAR2(6) NOT NULL,
OID_PR INTEGER NOT NULL UNIQUE,
codigoFactura VARCHAR2(8) NOT NULL UNIQUE,
PRIMARY KEY (codigoPedido),
FOREIGN KEY (OID_PR) REFERENCES PeticionesReserva ON DELETE CASCADE,
FOREIGN KEY (codigoFactura) REFERENCES Facturas ON DELETE CASCADE
);

CREATE TABLE LineasDePedido (
codigoPedido VARCHAR2(6) NOT NULL,
nombre VARCHAR(30) NOT NULL,
racionPedida VARCHAR2(8) CHECK (racionPedida IN ('media', 'completa')) NOT NULL,
unidadesPedidas SMALLINT NOT NULL,
PRIMARY KEY (codigoPedido, nombre, racionPedida),
FOREIGN KEY (codigoPedido) REFERENCES Pedidos ON DELETE CASCADE,
FOREIGN KEY (nombre) REFERENCES Platos
);
```

```
CREATE TABLE Productos (
nombre VARCHAR2(30) NOT NULL,
unidadMedida VARCHAR2(10) NOT NULL CHECK (unidadMedida IN (
'gramos', 'kilogramos', 'botes', 'botellas', 'paquetes', 'latas', 'unidades', 'bolsas', 'litros')),
umbralExistencias NUMBER NOT NULL,
PRIMARY KEY (nombre)
);

CREATE TABLE GruposDeProductos (
OID_GP INTEGER NOT NULL,
nombre VARCHAR2(30) NOT NULL,
fechaCaducidad DATE NOT NULL,
cantidadExistencia NUMBER NOT NULL,
PRIMARY KEY (OID_GP),
UNIQUE (nombre, fechaCaducidad),
FOREIGN KEY (nombre) REFERENCES Productos
);

CREATE TABLE ListasDeCompras(
codigoLista VARCHAR2(8) NOT NULL,
fechaCreacion DATE NOT NULL UNIQUE,
PRIMARY KEY (codigoLista)
);

CREATE TABLE Notas (
codigoNota VARCHAR2(8) NOT NULL,
fechaElaboracion DATE NOT NULL,
codigoLista VARCHAR2(8),
PRIMARY KEY (codigoNota),
FOREIGN KEY (codigoLista) REFERENCES ListasDeCompras ON DELETE CASCADE
);

CREATE TABLE LineasDeNota (
nombre VARCHAR2(30) NOT NULL,
cantidad NUMBER NOT NULL,
codigoNota VARCHAR2(8) NOT NULL,
PRIMARY KEY (nombre, codigoNota),
FOREIGN KEY (codigoNota) REFERENCES Notas ON DELETE CASCADE
);
```

X. 2 Scripts de creación de triggers

```
CREATE OR REPLACE TRIGGER numeroMesasConcretas
AFTER INSERT ON MesasConcretas
FOR EACH ROW
BEGIN
UPDATE Mesas SET numMesasConcretas = numMesasConcretas + 1 WHERE OID_M = :NEW.OID_M;
END;
/
CREATE OR REPLACE TRIGGER numeroMesasConcretas2
AFTER DELETE ON MesasConcretas
FOR EACH ROW
BEGIN
UPDATE Mesas SET numMesasConcretas = numMesasConcretas - 1 WHERE OID_M = :OLD.OID_M;
END;
/
CREATE OR REPLACE TRIGGER numeroMesasConcretas3
AFTER UPDATE OF OID_M ON MesasConcretas
FOR EACH ROW
BEGIN
UPDATE Mesas SET numMesasConcretas = numMesasConcretas + 1 WHERE OID_M = :NEW.OID_M;
UPDATE Mesas SET numMesasConcretas = numMesasConcretas - 1 WHERE OID_M = :OLD.OID_M;
END;
```

```
CREATE OR REPLACE TRIGGER evitarSolaparReservas
BEFORE INSERT ON Reservas
FOR EACH ROW
BEGIN
IF sePuedeReservar(:NEW.identificador, :NEW.fechaYHora) = 'N' THEN
raise_application_error
(-20600,:NEW.codigo||' La mesa no está disponible para esta fecha y hora.');
END IF;
END;
/
CREATE OR REPLACE TRIGGER evitarSolaparReservas2
BEFORE UPDATE OF fechaYHora ON Reservas
FOR EACH ROW
BEGIN
IF sePuedeReservar(:OLD.identificador, :NEW.fechaYHora) = 'N' THEN
raise_application_error
(-20600,:NEW.codigo||' La mesa no está disponible para esta fecha y hora.');
END IF;
END;
```

```
CREATE OR REPLACE TRIGGER reservarVariasMesasALaVez
BEFORE INSERT ON Reservas
FOR EACH ROW
DECLARE
CURSOR cur12 IS SELECT fechaYHora, OID_C,  codigoPedido, codigo FROM Facturas F NATURAL JOIN Reservas LEFT JOIN Pedidos P on F.codigoFactura = P.codigoFactura;
registro cur12%ROWTYPE;
BEGIN
OPEN cur12;
LOOP
FETCH cur12 INTO registro;
EXIT WHEN cur12%NOTFOUND;

IF registro.fechaYHora = :NEW.fechaYHora AND registro.OID_C = :NEW.OID_C AND registro.codigoPedido IS NULL THEN
DELETE FROM Facturas WHERE  codigo = registro.codigo;
END IF;

IF registro.fechaYHora = :NEW.fechaYHora AND registro.OID_C = :NEW.OID_C AND registro.codigoPedido IS NOT NULL THEN
DBMS_OUTPUT.put_line('Mensaje de error: ');
DBMS_OUTPUT.put_line('Si un cliente quiere reservas varias mesas para la misma fecha y hora, y, además, quiere pedir la comida de antemano, el pedido tiene que corresponder a la última petición de reserva del cliente.');
DBMS_OUTPUT.put_line('Solución: Borre las facturas y las reservas que se han creado. Vuelva a ejecutar los métodos reservarMesa, pero esta vez procure que el último en ejecutarse sea el método reservarMesa con la petición de reserva que incluya el pedido.');
END IF;


END LOOP;

CLOSE cur12;
END;
```

```
CREATE OR REPLACE TRIGGER evitarPedirMediaRacion
BEFORE INSERT ON LineasDePedido
FOR EACH ROW
DECLARE
CURSOR cur7 IS SELECT precioMediaRacion FROM Platos WHERE nombre = :NEW.nombre;
w_precioMediaRacion NUMBER(2);
BEGIN
OPEN cur7;
FETCH cur7 INTO w_precioMediaRacion;

IF w_precioMediaRacion IS NULL AND :NEW.racionPedida = 'media' THEN
raise_application_error
(-20600,:NEW.nombre||' No se puede pedir media ración de este plato.');
END IF;
CLOSE cur7;
END;
```

X. 3 Scripts de creación de procedimientos

```
CREATE OR REPLACE PROCEDURE encontrarMesa (
w_numPersonas IN PeticionesReserva.numPersonas%TYPE,
w_ubicacionMesa IN PeticionesReserva.ubicacionMesa%TYPE,
w_tipoMesa IN PeticionesReserva.tipoMesa%TYPE,
w_fechaYHoraReserva IN PeticionesReserva.FechaYHoraReserva%TYPE)
IS
CURSOR cur IS SELECT identificador, maxPersonas FROM MesasConcretas NATURAL JOIN Mesas WHERE w_numPersonas <= maxPersonas AND
(w_ubicacionMesa IS NULL OR w_ubicacionMesa = ubicacion) AND
(w_tipoMesa IS NULL OR w_tipoMesa = tipo) ORDER BY maxPersonas ASC;
registro cur%ROWTYPE;
BEGIN
DBMS_OUTPUT.put_line('Lista de las mesas que cumplen los requisitos del cliente (se recomienda acomodar al cliente en la primera mesa disponible que aparezca en la lista):');
OPEN cur;
LOOP
FETCH cur INTO registro;
EXIT WHEN cur%NOTFOUND;

IF sePuedeReservar(registro.identificador, w_fechaYHoraReserva) = 'Y' THEN
DBMS_OUTPUT.put_line('El cliente puede acomodarse en la mesa ' ||registro.identificador|| ' que tiene capacidad ' ||registro.maxPersonas);
ELSE DBMS_OUTPUT.put_line('La mesa ' ||registro.identificador|| ' cumple los requisitos, pero no está disponible para esta fecha y hora.');
DBMS_OUTPUT.put_line('Pida al cliente que cambie la fecha o la hora. Puede consultar las reservas que tiene asignadas la mesa ' ||registro.identificador|| ' ejecutando el comando consultarReservasMesa( ' ||registro.identificador|| ' )' );
END IF;

END LOOP;

IF CUR%ROWCOUNT = 0 THEN
DBMS_OUTPUT.put_line('No hay mesas en el restaurante que cumplan los requisitos. El problema no está en que las mesas ya estén reservadas, sino en que no existen tales mesas.');
END IF;

CLOSE cur;
END encontrarMesa;
```

```
CREATE OR REPLACE PROCEDURE llamadaAEncontrarMesa (
w_OID_PR IN PeticionesReserva.OID_PR%TYPE)
IS
CURSOR cur3 IS SELECT numPersonas, ubicacionMesa, tipoMesa, fechaYHoraReserva FROM PeticionesReserva WHERE OID_PR = w_OID_PR;
registro cur3%ROWTYPE;
BEGIN
OPEN cur3;
FETCH cur3 INTO registro;
encontrarMesa(registro.numPersonas, registro.ubicacionMesa, registro.tipoMesa, registro.fechaYHoraReserva);
CLOSE cur3;
END llamadaAEncontrarMesa;
```

```
CREATE OR REPLACE PROCEDURE reservarMesa(
w_codigo IN Reservas.codigo%TYPE,
w_OID_PR IN INTEGER,
w_identificador IN MesasConcretas.identificador%TYPE)
IS
CURSOR cur4 IS SELECT fechaYHoraReserva, OID_C FROM PeticionesReserva NATURAL JOIN Clientes WHERE OID_PR = w_OID_PR;
registro cur4%ROWTYPE;
BEGIN
OPEN cur4;
FETCH cur4 INTO registro;
INSERT INTO Reservas VALUES (w_codigo, registro.FechaYHoraReserva, w_identificador, registro.OID_C);
DBMS_OUTPUT.put_line('La mesa ' ||w_identificador|| ' se ha reservado para ' ||registro.fechaYHoraReserva|| ' por el cliente ' ||registro.OID_C);
INSERT INTO Facturas VALUES (w_OID_PR, SYSDATE, DEFAULT, w_codigo);
DBMS_OUTPUT.put_line('Se ha elaborado una factura para esta reserva.');
CLOSE cur4;
END reservarMesa;
```

```
CREATE OR REPLACE PROCEDURE consultarReservasMesa(
w_identificador IN MesasConcretas.identificador%TYPE)
IS
CURSOR cur5 IS SELECT fechaYHora FROM Reservas WHERE identificador = w_identificador;
registro cur5%ROWTYPE;
BEGIN
DBMS_OUTPUT.put_line('Lista de todas las reservas que la mesa ' ||w_identificador|| ' tiene asignadas:');
OPEN cur5;
LOOP
FETCH cur5 INTO registro;
EXIT WHEN cur5%NOTFOUND;

DBMS_OUTPUT.put_line('Tiene asignada una reserva el ' ||registro.fechaYHora);
END LOOP;

IF CUR5%ROWCOUNT = 0 THEN
DBMS_OUTPUT.put_line('La mesa ' ||w_identificador|| ' de momento no tiene asignada ninguna reserva.');
END IF;

CLOSE cur5;
END consultarReservasMesa;
```

```
CREATE OR REPLACE PROCEDURE reservasAConfirmar
IS
CURSOR cur6 IS SELECT DISTINCT fechaYHora, nombre, apellidos, telefono FROM Reservas NATURAL JOIN Clientes WHERE fechaYHora BETWEEN CURRENT_TIMESTAMP AND CURRENT_TIMESTAMP + INTERVAL '1' DAY;
w_telefono  Clientes.telefono%TYPE;
w_nombre Clientes.nombre%TYPE;
w_apellidos Clientes.apellidos%TYPE;
w_fechaYHora Reservas.fechaYHora%TYPE;
BEGIN
DBMS_OUTPUT.put_line('Tiene que llamar a los siguientes clientes:');
OPEN cur6;
LOOP
FETCH cur6 INTO w_fechaYHora, w_nombre, w_apellidos, w_telefono ;
EXIT WHEN cur6%NOTFOUND;

DBMS_OUTPUT.put_line(w_telefono|| ' ( ' ||w_nombre|| ' ' ||w_apellidos|| ' ) ha reservado una mesa para ' ||w_fechaYHora);
END LOOP;

IF CUR6%ROWCOUNT = 0 THEN
DBMS_OUTPUT.put_line('No hay reservas pendientes a confirmar.');
END IF;

CLOSE cur6;
END reservasAConfirmar;
```

```
CREATE OR REPLACE PROCEDURE imprimirPedido(
w_codigoPedido IN Pedidos.codigoPedido%TYPE)
IS
CURSOR cur8 IS SELECT nombre, racionPedida, unidadesPedidas, identificador, fechaYHora FROM LineasDePedido NATURAL JOIN Pedidos NATURAL JOIN Facturas NATURAL JOIN Reservas WHERE codigoPedido = w_codigoPedido;
registro cur8%ROWTYPE;
BEGIN
OPEN cur8;
LOOP
FETCH cur8 INTO registro;
EXIT WHEN cur8%NOTFOUND;

IF CUR8%ROWCOUNT = 1 THEN
DBMS_OUTPUT.put_line('Se tienen que servir los siguientes platos en la mesa  ' ||registro.identificador|| ' el día ' ||registro.fechaYHora);
END IF;

DBMS_OUTPUT.put_line(registro.nombre|| ' (' ||registro.racionPedida|| ') x ' ||registro.unidadesPedidas);
END LOOP;


CLOSE cur8;
END imprimirPedido;
```

```
CREATE OR REPLACE PROCEDURE imprimirFactura(
w_codigoFactura IN Facturas.codigoFactura%TYPE)
IS
CURSOR cur11 IS SELECT codigoPedido
FROM Pedidos NATURAL JOIN Facturas WHERE codigoFactura = w_codigoFactura;
w_codigoPedido Pedidos.codigoPedido%TYPE;

BEGIN

OPEN cur11;
FETCH cur11 INTO w_codigoPedido;
IF cur11%NOTFOUND THEN
imprimirFacturaSinPedido(w_codigoFactura);
ELSE
imprimirFacturaConPedido(w_codigoFactura);
END IF;
CLOSE cur11;
END imprimirFactura;
```

```
CREATE OR REPLACE PROCEDURE imprimirFacturaConPedido(
w_codigoFactura IN Facturas.codigoFactura%TYPE)
IS
CURSOR cur9 IS SELECT fechaElaboracion, cuantiaFija, nombre, racionPedida, unidadesPedidas, precioMediaRacion, precioRacion,  calcularPrecio(racionPedida, unidadesPedidas, precioMediaRacion, precioRacion)
FROM Facturas NATURAL JOIN Pedidos NATURAL JOIN LineasDePedido NATURAL JOIN Platos WHERE codigoFactura = w_codigoFactura;
w_fechaElaboracion Facturas.fechaElaboracion%TYPE;
w_cuantiaFija Facturas.cuantiaFija%TYPE;
w_nombre Platos.nombre%TYPE;
w_racionPedida LineasDePedido.racionPedida%TYPE;
w_unidadesPedidas LineasDePedido.unidadesPedidas%TYPE;
w_precioMediaRacion Platos.precioMediaRacion%TYPE;
w_precioRacion Platos.precioRacion%TYPE;
w_precio NUMBER;
acumulador NUMBER := 0;
cuantia NUMBER :=0;
precioTotal NUMBER := 0;
BEGIN
OPEN cur9;
LOOP
FETCH cur9 INTO w_fechaElaboracion, w_cuantiaFija, w_nombre, w_racionPedida, w_unidadesPedidas, w_precioMediaRacion, w_precioRacion, w_precio;
EXIT WHEN cur9%NOTFOUND;
IF CUR9%ROWCOUNT = 1 THEN
DBMS_OUTPUT.put_line('                FACTURA                ');
DBMS_OUTPUT.put_line('          Restaurante El Tejao          ');
DBMS_OUTPUT.put_line('codigo: ' ||w_codigoFactura|| '          fecha: ' ||w_fechaElaboracion);
END IF;
DBMS_OUTPUT.put_line(w_nombre|| ' (' ||w_racionPedida|| ') x ' ||w_unidadesPedidas|| ' = ' ||w_precio|| '€');
acumulador := acumulador + w_precio;
cuantia := w_cuantiaFija;
END LOOP;
precioTotal := (acumulador*30)/100 + cuantia;
DBMS_OUTPUT.put_line('Coste total: ' ||precioTotal|| '€');
CLOSE cur9;
END imprimirFacturaConPedido;
```

```
CREATE OR REPLACE PROCEDURE imprimirFacturaSinPedido(
w_codigoFactura IN Facturas.codigoFactura%TYPE)
IS
CURSOR cur10 IS SELECT fechaElaboracion, cuantiaFija
FROM Facturas WHERE codigoFactura = w_codigoFactura;
w_fechaElaboracion Facturas.fechaElaboracion%TYPE;
w_cuantiaFija Facturas.cuantiaFija%TYPE;

BEGIN

OPEN cur10;
FETCH cur10 INTO w_fechaElaboracion, w_cuantiaFija;
DBMS_OUTPUT.put_line('                FACTURA                ');
DBMS_OUTPUT.put_line('          Restaurante El Tejao          ');
DBMS_OUTPUT.put_line('codigo: ' ||w_codigoFactura|| '          fecha: ' ||w_fechaElaboracion);
DBMS_OUTPUT.put_line('Coste total: ' ||w_cuantiaFija|| '€');
CLOSE cur10;
END imprimirFacturaSinPedido;
```

```
CREATE OR REPLACE PROCEDURE borrarDatosObsoletos
IS

BEGIN

DELETE FROM PeticionesReserva WHERE fechaYHoraReserva < CURRENT_TIMESTAMP;
DELETE FROM Reservas WHERE fechaYHora < CURRENT_TIMESTAMP;


END borrarDatosObsoletos;
```

```
CREATE OR REPLACE PROCEDURE checkExistenciasProducto(
w_nombre  IN GruposDeProductos.nombre%TYPE)
IS
CURSOR cur14 IS SELECT umbralExistencias, cantidadExistencia FROM GruposDeProductos NATURAL JOIN Productos WHERE nombre = w_nombre;
registro cur14%ROWTYPE;
w_acumulador NUMBER := 0;
w_umbral NUMBER := 0;
BEGIN
OPEN cur14;
LOOP
FETCH cur14 INTO registro;
EXIT WHEN cur14%NOTFOUND;
w_acumulador := w_acumulador + registro.cantidadExistencia;
w_umbral := registro.umbralExistencias;
END LOOP;

IF w_acumulador < w_umbral OR cur14%ROWCOUNT = 0 THEN
DBMS_OUTPUT.put_line('Hay que comprar ' ||w_nombre);
END IF;
CLOSE cur14;
END checkExistenciasProducto;
```

```
CREATE OR REPLACE PROCEDURE comprobarExistencias
IS
CURSOR cur15 IS SELECT DISTINCT nombre FROM Productos;
w_nombre Productos.nombre%TYPE;
BEGIN
OPEN cur15;
LOOP
FETCH cur15 INTO w_nombre;
EXIT WHEN cur15%NOTFOUND;
checkExistenciasProducto(w_nombre);
END LOOP;
CLOSE cur15;
END comprobarExistencias;
```

```
CREATE OR REPLACE PROCEDURE comprobarFechaCaducidad
IS
CURSOR cur16 IS SELECT fechaCaducidad, OID_GP, cantidadExistencia, unidadMedida, nombre FROM GruposDeProductos NATURAL JOIN Productos;
registro cur16%ROWTYPE;
BEGIN
OPEN cur16;
LOOP
FETCH cur16 INTO registro;
EXIT WHEN cur16%NOTFOUND;
IF registro.fechaCaducidad < SYSDATE THEN
DBMS_OUTPUT.put_line('Los/Las ' ||registro.cantidadExistencia|| ' ' ||registro.unidadMedida|| ' de ' ||registro.nombre|| ' ya han caducado. Tiene que tirar los productos del grupo ' ||registro.OID_GP|| ' a la basura') ;
END IF;
END LOOP;
CLOSE cur16;
END comprobarFechaCaducidad;
```

```
CREATE OR REPLACE PROCEDURE consultarNota (
w_codigoNota IN Notas.codigoNota%TYPE
)
IS
CURSOR cur17 IS SELECT nombre, cantidad, unidadMedida FROM Notas NATURAL JOIN LineasDeNota NATURAL JOIN Productos unidadMedida WHERE codigoNota = w_codigoNota;
registro cur17%ROWTYPE;
BEGIN
OPEN cur17;
LOOP
FETCH cur17 INTO registro;
EXIT WHEN cur17%NOTFOUND;
DBMS_OUTPUT.put_line('El jefe de cocina necesita ' || registro.cantidad|| ' '||registro.unidadMedida|| ' de ' ||registro.nombre);
encontrarGrupo(registro.nombre, registro.cantidad, registro.unidadMedida);
END LOOP;
CLOSE cur17;
END consultarNota;
```

```
CREATE OR REPLACE PROCEDURE encontrarGrupo (
w_nombre IN LineasDeNota.nombre%TYPE,
w_cantidad IN LineasDeNota.cantidad%TYPE,
w_unidadMedida IN Productos.unidadMedida%TYPE
)
IS
CURSOR cur18 IS SELECT cantidadExistencia, fechaCaducidad, OID_GP FROM GruposDeProductos WHERE nombre = w_nombre ORDER BY fechaCaducidad ASC;
registro cur18%ROWTYPE;
BEGIN
DBMS_OUTPUT.put_line('********** Los/las ' || w_cantidad|| ' '||w_unidadMedida|| ' de ' ||w_nombre|| ' se pueden obtener de los siguientes grupos (es recomendable coger primero los productos del primer grupo que aparezca en la lista): ');
OPEN cur18;
LOOP
FETCH cur18 INTO registro;
EXIT WHEN cur18%NOTFOUND;
DBMS_OUTPUT.put_line('          El grupo ' ||registro.OID_GP|| ', donde hay ' ||registro.cantidadExistencia|| ' ' ||w_unidadMedida|| ' de ' ||w_nombre|| ' con fecha de caducidad: ' ||registro.fechaCaducidad);
END LOOP;
CLOSE cur18;
END encontrarGrupo;
```

```
CREATE OR REPLACE PROCEDURE entregarProductos (
w_OID_GP IN GruposDeProductos.OID_GP%TYPE,
w_cantidad IN LineasDeNota.cantidad%TYPE
)
IS

BEGIN
UPDATE GruposDeProductos SET cantidadExistencia = cantidadExistencia - w_cantidad WHERE OID_GP = w_OID_GP;
DELETE FROM GruposDeProductos WHERE cantidadExistencia = 0;
DBMS_OUTPUT.put_line('Se han cogido ' ||w_cantidad|| ' unidades del grupo ' ||w_OID_GP);
END entregarProductos;
```

```
CREATE OR REPLACE PROCEDURE elaborarListaDeCompras (
w_codigoLista IN ListasDeCompras.codigoLista%TYPE)
IS
CURSOR cur19 IS SELECT DISTINCT nombre FROM LineasDeNota NATURAL JOIN Notas;
w_nombre LineasDeNota.nombre%TYPE;
BEGIN
DELETE FROM ListasDeCompras WHERE fechaCreacion < SYSDATE;
DBMS_OUTPUT.put_line('Se han borrado todas las listas de compras anteriores junto con sus correspondientes notas');
INSERT INTO ListasDeCompras VALUES (w_codigoLista, SYSDATE);
DBMS_OUTPUT.put_line('Se ha creado nueva lista de compras. Su código es: ' ||w_codigoLista);
UPDATE Notas SET codigoLista = w_codigoLista;
DBMS_OUTPUT.put_line('Todas las notas existentes ya corresponden a la nueva lista de compras');
DBMS_OUTPUT.put_line('++++   Lista de compras:   ++++');
OPEN cur19;
LOOP
FETCH cur19 INTO w_nombre;
EXIT WHEN cur19%NOTFOUND;
procedimientoAuxiliar(w_nombre);
END LOOP;
CLOSE cur19;
END elaborarListaDeCompras;
```

```
CREATE OR REPLACE PROCEDURE procedimientoAuxiliar (
w_nombre IN LineasDeNota.nombre%TYPE)
IS
CURSOR cur20 IS SELECT cantidad, unidadMedida FROM LineasDeNota NATURAL JOIN Notas NATURAL JOIN Productos WHERE nombre = w_nombre;
w_cantidad LineasDeNota.cantidad%TYPE;
w_unidadMedida Productos.unidadMedida%TYPE;
acumulador NUMBER := 0;
BEGIN
OPEN cur20;
LOOP
FETCH cur20 INTO w_cantidad, w_unidadMedida;
EXIT WHEN cur20%NOTFOUND;
acumulador := acumulador + w_cantidad;
END LOOP;
DBMS_OUTPUT.put_line(acumulador|| ' ' ||w_unidadMedida|| ' de ' ||w_nombre);
CLOSE cur20;
END procedimientoAuxiliar;
```

```
CREATE OR REPLACE PROCEDURE copiaPruebas (
w_codigoLista IN ListasDeCompras.codigoLista%TYPE,
w_fechaCreacion IN ListasDeCompras.fechaCreacion%TYPE)
IS
CURSOR cur19 IS SELECT DISTINCT nombre FROM LineasDeNota NATURAL JOIN Notas;
w_nombre LineasDeNota.nombre%TYPE;
BEGIN
DELETE FROM ListasDeCompras WHERE fechaCreacion < w_fechaCreacion;
DBMS_OUTPUT.put_line('Se han borrado todas las listas de compras anteriores junto con sus correspondientes notas');
INSERT INTO ListasDeCompras VALUES (w_codigoLista, w_fechaCreacion);
DBMS_OUTPUT.put_line('Se ha creado nueva lista de compras. Su código es: ' ||w_codigoLista);
UPDATE Notas SET codigoLista = w_codigoLista;
DBMS_OUTPUT.put_line('Todas las notas existentes ya corresponden a la nueva lista de compras');
DBMS_OUTPUT.put_line('++++   Lista de compras:   ++++');
OPEN cur19;
LOOP
FETCH cur19 INTO w_nombre;
EXIT WHEN cur19%NOTFOUND;
procedimientoAuxiliar(w_nombre);
END LOOP;
CLOSE cur19;
END copiaPruebas;
```

```
CREATE OR REPLACE PROCEDURE nuevoCliente(
w_OID_C IN Clientes.OID_C%TYPE,
w_nombre IN Clientes.nombre%TYPE,
w_apellidos IN Clientes.apellidos%TYPE,
w_telefono IN Clientes.telefono%TYPE)
IS
CURSOR cur0 IS SELECT OID_C FROM Clientes WHERE nombre = w_nombre AND apellidos = w_apellidos AND telefono = w_telefono;
registro Clientes.OID_C%TYPE;
BEGIN
OPEN cur0;
FETCH cur0 INTO registro;
IF cur0%NOTFOUND THEN
INSERT INTO Clientes VALUES (w_OID_C, w_nombre, w_apellidos, w_telefono);
DBMS_OUTPUT.PUT_LINE('Cliente ' ||w_OID_C|| ' del restaurante. Sus datos son: ' ||w_nombre|| ' ' ||w_apellidos|| ' (' ||w_telefono|| ') . ');
ELSE
DBMS_OUTPUT.PUT_LINE('Este cliente ya existe en la base de datos. Su número identificador es ' ||registro);
END IF;
END nuevoCliente;
```

```
CREATE OR REPLACE PROCEDURE requisitosMesa(
w_OID_PR IN PeticionesReserva.OID_PR%TYPE,
w_fechaYHoraReserva IN PeticionesReserva.fechaYHoraReserva%TYPE,
w_numPersonas IN PeticionesReserva.numPersonas%TYPE,
w_ubicacionMesa IN PeticionesReserva.ubicacionMesa%TYPE,
w_tipoMesa IN PeticionesReserva.tipoMesa%TYPE,
w_OID_C IN Clientes.OID_C%TYPE)
IS
BEGIN
INSERT INTO PeticionesReserva VALUES (w_OID_PR, w_fechaYHoraReserva, w_numPersonas, w_ubicacionMesa, w_tipoMesa, w_OID_C);
DBMS_OUTPUT.PUT_LINE('Cliente ' ||w_OID_C|| ' del restaurante quiere reservar una mesa para ' ||w_fechaYHoraReserva|| ' con las siguientes caracterisitcas: ');
DBMS_OUTPUT.PUT_LINE('Número de personas: ' ||w_numPersonas);
DBMS_OUTPUT.PUT_LINE('Interior/exterior: ' ||w_ubicacionMesa);
DBMS_OUTPUT.PUT_LINE('Fumadores/no fumadores: ' ||w_tipoMesa);
DBMS_OUTPUT.PUT_LINE('Esta petición de reserva llevará el número identificador: ' ||w_OID_PR);
END requisitosMesa;
```

```
CREATE OR REPLACE PROCEDURE introducirNuevoTipoMesa(
w_OID_M IN Mesas.OID_M%TYPE,
w_maxPersonas IN Mesas.maxPersonas%TYPE,
w_ubicacion IN Mesas.ubicacion%TYPE,
w_tipo IN Mesas.tipo%TYPE)
IS
BEGIN
INSERT INTO Mesas VALUES (w_OID_M, w_maxPersonas, w_ubicacion, w_tipo, DEFAULT);
DBMS_OUTPUT.PUT_LINE('Se ha introducido un nuevo tipo de mesa (tipo ' ||w_OID_M|| ') - para ' ||w_maxPersonas|| ' personas como mucho, para ' ||w_tipo|| ' situada en el ' ||w_ubicacion|| ' del restaurante.');
END introducirNuevoTipoMesa;
```

```
CREATE OR REPLACE PROCEDURE nuevaMesa(
w_identificador IN MesasConcretas.identificador%TYPE,
w_OID_M IN Mesas.OID_M%TYPE)
IS
BEGIN
INSERT INTO MesasConcretas VALUES (w_identificador, w_OID_M);
DBMS_OUTPUT.PUT_LINE('Colocamos una mesa de tipo ' ||w_OID_M|| ' . El número identificador de esta mesa es: ' ||w_identificador);
END nuevaMesa;
```

```
CREATE OR REPLACE PROCEDURE incluirPedido(
w_codigoPedido IN Pedidos.codigoPedido%TYPE,
w_OID_PR IN PeticionesReserva.OID_PR%TYPE)
IS
BEGIN
INSERT INTO Pedidos VALUES (w_codigoPedido, w_OID_PR, w_OID_PR);
DBMS_OUTPUT.PUT_LINE('Se ha creado un nuevo pedido que corresponde a la petición de reserva ' ||w_OID_PR|| '. El código del pedido es: ' ||w_codigoPedido);
END incluirPedido;

/

CREATE OR REPLACE PROCEDURE pedirPlato(
w_codigoPedido IN Pedidos.codigoPedido%TYPE,
w_nombre IN Platos.nombre%TYPE,
w_racionPedida IN LineasDePedido.racionPedida%TYPE,
w_unidadesPedidas IN LineasDePedido.unidadesPedidas%TYPE)
IS
BEGIN
INSERT INTO LineasDePedido VALUES (w_codigoPedido, w_nombre, w_racionPedida, w_unidadesPedidas);
DBMS_OUTPUT.PUT_LINE('El plato se ha añadido correctamente al pedido del cliente.');
END pedirPlato;
```

```
CREATE OR REPLACE PROCEDURE introducirNuevoPlato(
w_nombre IN Platos.nombre%TYPE,
w_precioMediaRacion IN Platos.precioMediaRacion%TYPE,
w_precioRacion IN Platos.precioRacion%TYPE
)
IS
BEGIN
INSERT INTO Platos VALUES (w_nombre, w_precioMediaRacion, w_precioRacion);
DBMS_OUTPUT.PUT_LINE('Nuevo plato en la carta del restaurante: ' ||w_nombre|| '. Precio para media ración: ' ||w_precioMediaRacion|| ' €. Precio para una ración completa: ' ||w_precioRacion|| ' €.');
END introducirNuevoPlato;

/

CREATE OR REPLACE PROCEDURE nuevoProducto(
w_nombre IN Productos.nombre%TYPE,
w_unidadMedida IN Productos.unidadMedida%TYPE,
w_umbralExistencias IN Productos.umbralExistencias%TYPE
)
IS
BEGIN
INSERT INTO Productos VALUES (w_nombre, w_unidadMedida, w_umbralExistencias);
DBMS_OUTPUT.PUT_LINE('Nuevo producto: ' ||w_nombre);
DBMS_OUTPUT.PUT_LINE('La cantidad de este producto se mide en : ' ||w_unidadMedida);
DBMS_OUTPUT.PUT_LINE('El encargado de almacén tiene que informarle inmediatamente a la responsable de compras si en el almacén quedan menos de ' ||w_umbralExistencias|| ' ' ||w_unidadMedida);
END nuevoProducto;
```

```
CREATE OR REPLACE PROCEDURE nuevoGrupoDeProductos (
w_OID_GP IN GruposDeProductos.OID_GP%TYPE,
w_nombre IN GruposDeProductos.nombre%TYPE,
w_fechaCaducidad IN GruposDeProductos.fechaCaducidad%TYPE,
w_cantidadExistencia IN GruposDeProductos.cantidadExistencia%TYPE)
IS
CURSOR cur13 IS SELECT unidadMedida FROM Productos WHERE  nombre = w_nombre;
w_unidadMedida Productos.unidadMedida%TYPE;
BEGIN
OPEN cur13;
FETCH cur13 INTO w_unidadMedida;
INSERT INTO GruposDeProductos VALUES (w_OID_GP, w_nombre, w_fechaCaducidad, w_cantidadExistencia);
DBMS_OUTPUT.PUT_LINE('El almacén se ha abastecido de ' ||w_cantidadExistencia|| ' ' ||w_unidadMedida|| ' de ' ||w_nombre|| ' que van a caducar el ' ||w_fechaCaducidad);
DBMS_OUTPUT.PUT_LINE('El número identificador de este grupo de productos es: ' ||w_OID_GP);

CLOSE cur13;
END nuevoGrupoDeProductos;

/

CREATE OR REPLACE PROCEDURE tirarProductos (
w_OID_GP IN GruposDeProductos.OID_GP%TYPE)
IS
BEGIN
DELETE  FROM GruposDeProductos WHERE OID_GP = w_OID_GP;
END tirarProductos;
```

```
CREATE OR REPLACE PROCEDURE crearNuevaNota(
w_codigoNota IN Notas.codigoNota%TYPE
)
IS
BEGIN
INSERT INTO Notas VALUES (w_codigoNota, SYSDATE, NULL);
DBMS_OUTPUT.PUT_LINE('Se ha creado nueva nota.');
DBMS_OUTPUT.PUT_LINE('Código de la nota: ' ||w_codigoNota);
DBMS_OUTPUT.PUT_LINE('Fecha de elaboración ' ||SYSDATE);
DBMS_OUTPUT.PUT_LINE('La nota todavía no forma parte de ninguna lista de compras.');
END crearNuevaNota;

/

CREATE OR REPLACE PROCEDURE escribirEnLaNota(
w_codigoNota IN LineasDeNota.codigoNota%TYPE,
w_nombre IN LineasDeNota.nombre%TYPE,
w_cantidad IN LineasDeNota.cantidad%TYPE
)
IS
BEGIN
INSERT INTO LineasDeNota VALUES (w_nombre, w_cantidad, w_codigoNota);
DBMS_OUTPUT.PUT_LINE('Se ha añadido una línea en la nota.');
END escribirEnLaNota;
```

X. 4 Scripts de creación de funciones

```
CREATE OR REPLACE FUNCTION calcularPrecio (
w_racionPedida IN LineasDePedido.racionPedida%TYPE,
w_unidadesPedidas IN LineasDePedido.unidadesPedidas%TYPE,
w_precioMediaRacion IN Platos.precioMediaRacion%TYPE,
w_precioRacion IN Platos.precioRacion%TYPE)
RETURN NUMBER
IS
BEGIN
IF w_racionPedida = 'media' THEN
RETURN w_unidadesPedidas*w_precioMediaRacion;
ELSE
RETURN w_unidadesPedidas*w_precioRacion;
END IF;

END calcularPrecio;
```

```
CREATE OR REPLACE FUNCTION sePuedeReservar (
w_identificador IN MesasConcretas.identificador%TYPE,
w_fechaYHoraReserva IN PeticionesReserva.fechaYHoraReserva%TYPE)
RETURN CHAR
IS w_fechaYHora TIMESTAMP;

CURSOR cur2 IS SELECT fechaYHora FROM Reservas
WHERE identificador = w_identificador AND (fechaYHora BETWEEN w_fechaYHoraReserva - 4/24 AND w_fechaYHoraReserva + 4/24);
BEGIN
OPEN cur2;
LOOP
FETCH cur2 INTO w_fechaYHora;
EXIT WHEN cur2%NOTFOUND;
END LOOP;
IF cur2%ROWCOUNT > 0 THEN
RETURN 'N';
ELSE
RETURN 'Y';
END IF;
CLOSE cur2;
END sePuedeReservar;
```

XI. Apéndice

I. 1 Fotos

(de la página web del restaurante)

(de la primera, la segunda y la tercera entrevista con la gerente – Joaquina Castro)

Carnes

	1/2 Ración	Ración
Solomillo a la casera	6,00€	10,00€
Solomillo al roquefort	6,00€	10,00€
Solomillo al whisky	6,00€	10,00€
Solomillo a la brasa		12,00€
Churrasco de cerdo		8,00€
Pechuga de pollo		6,00€
Costillitas ibéricas deshuesadas	6,00€	10,00€
Pluma ibérica		10,00€
Secreto ibérico		10,00€
Entrecot de ternera		12,00€
Solomillo de ternera		14,00€
Chuletas de cordero	9,00€	15,00€
Solomillo de pato		8,00€

Entradas

	1/2 Ración	Ración
Revueltos		
Revuelto de colas de cangrejo	4,50€	8,00€
Revuelto ibérico	4,50€	8,00€
Revuelto de ajetes y gambas	4,50€	8,00€
Panes de la casa		
Pan de jamón		8,00€
Pan de albur marinado		8,00€
Pan de bacalao		8,00€

Entradas

	1/2 Ración	Ración
Ensalada mixta		persona/ 3,00€
Ensalada El Tejao		persona/ 5,00€
(Lechuga, tomate cherry, pasas,aguacate, melocotón, gambas y salsa rosa)		
Aliño de pimientos	6,00€	10,00€
Cóctel de marisco	6,00€	10,00€
Ensaladilla rusa	6,00€	10,00€
Croquetas caseras de puchero y gambas	6,00€	10,00€
Tortillitas de camarón	6,00€	10,00€
Tortillitas de bacalao y gambas	6,00€	10,00€
Frito de la casa	6,00€	10,00€
Camarones fritos con aliño de pimientos	6,00€	10,00€
Cangrejos a la plancha	6,00€	10,00€
Colas de cangrejo al ajillo/fritas	6,00€	10,00€
Espinacas con garbanzos	6,00€	10,00€

Pescados / Marisco

	1/2 Ración	Ración
Chocos fritos	6,00€	10,00€
Puntillitas	6,00€	10,00€
Chipirón plancha/frito	8,00€	12,00€
Pulpo a la gallega	6,00€	10,00€
Pescada a la plancha		5,00€
Mero a la plancha	6,00€	10,00€
Hueva plancha/frita	6,00€	10,00€
Pez espada plancha		10,00€
Frito variado	10,00€	14,00€
Sepia plancha		7,00€
Lubina		9,00€
Ventresca de atún		8,00€
Albur plancha	8,00€	12,00€
Gambas al ajillo	6,00€	10,00€
Gambón plancha		Unidad/ 1,50€
Coquinas		10,00€

Arroces

Arroz con pato		9,00€
Arroz caldoso con cola de cangrejo		9,00€
Arroz caldoso con marisco		9,00€
Arroz negro *(mínimo 2 personas)*	persona/	7,00€
Arroz en salsa Doñana		12,00€
Crujiente de arroz en salsa Doñana		12,00€
Arroz frito con angulas y gambas		12,00€
Arroz al horno *(por encargo / mínimo 2 personas)*	persona/	6,00€
Paella		5,00€
Paella por encargo de carne o verduras *(mín. 2 pax)*	persona/	7,50€
Paella por encargo de marisco *(mínimo 2 personas)*	persona/	9,00€

Vinos

Tinto

Arnegui reserva D.O. Rioja	17,00€
Arnegui crianza D.O. Rioja	12,00€
Beronia crianza D.O. Rioja	12,00€
Marqués de Cáceres crianza D.O. Rioja	13,00€
Condado de Oriza Reserva D.O. Ribera del Duero	18,00€
Condado de Oriza Roble D.O. Ribera del Duero	12,00€
Protos Roble D.O. Ribera del Duero	13,00€

Blancos

Barbadillo	8,00€
Mioro	8,00€
Analivia Rueda	10,00€
Manzanilla la Guita 1/2	5,00€

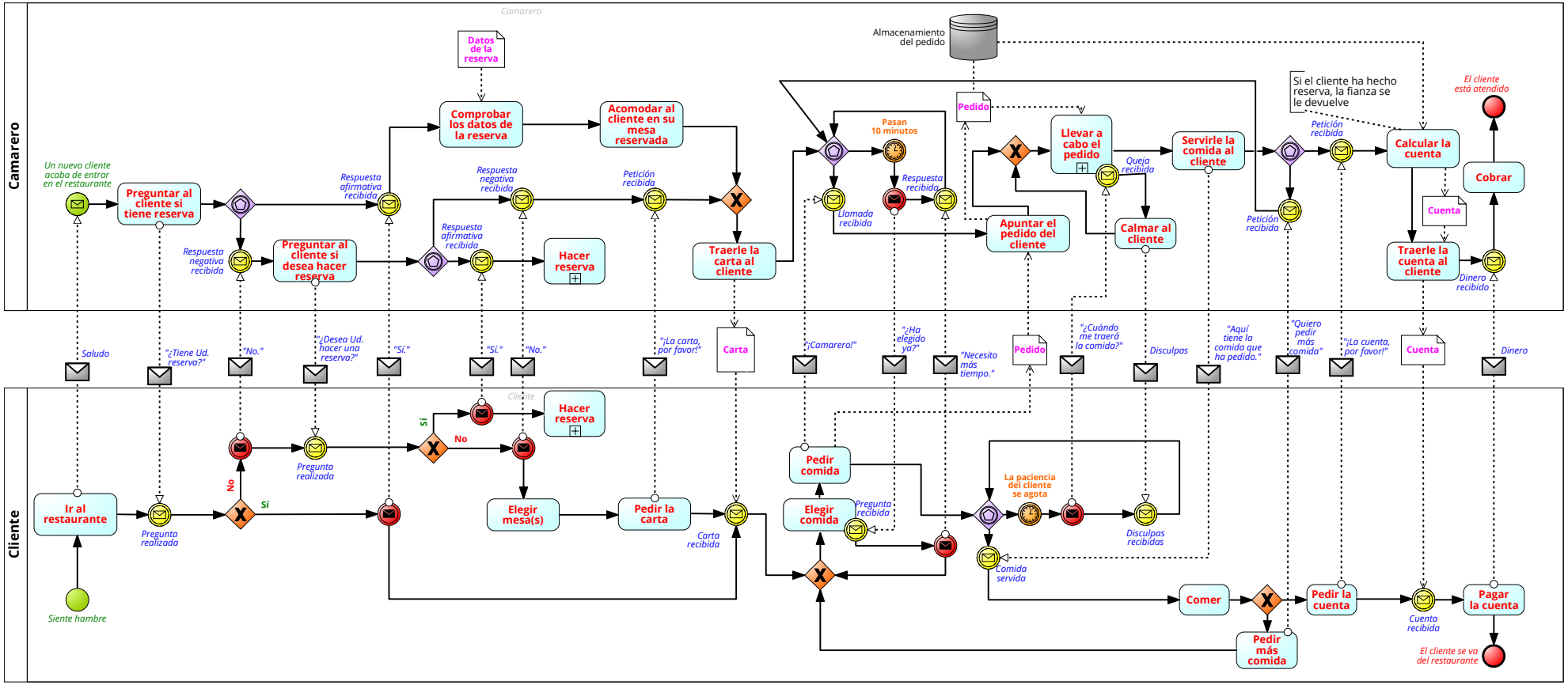
Rosados

Lambrusco	6,00€
Frissé	6,00€

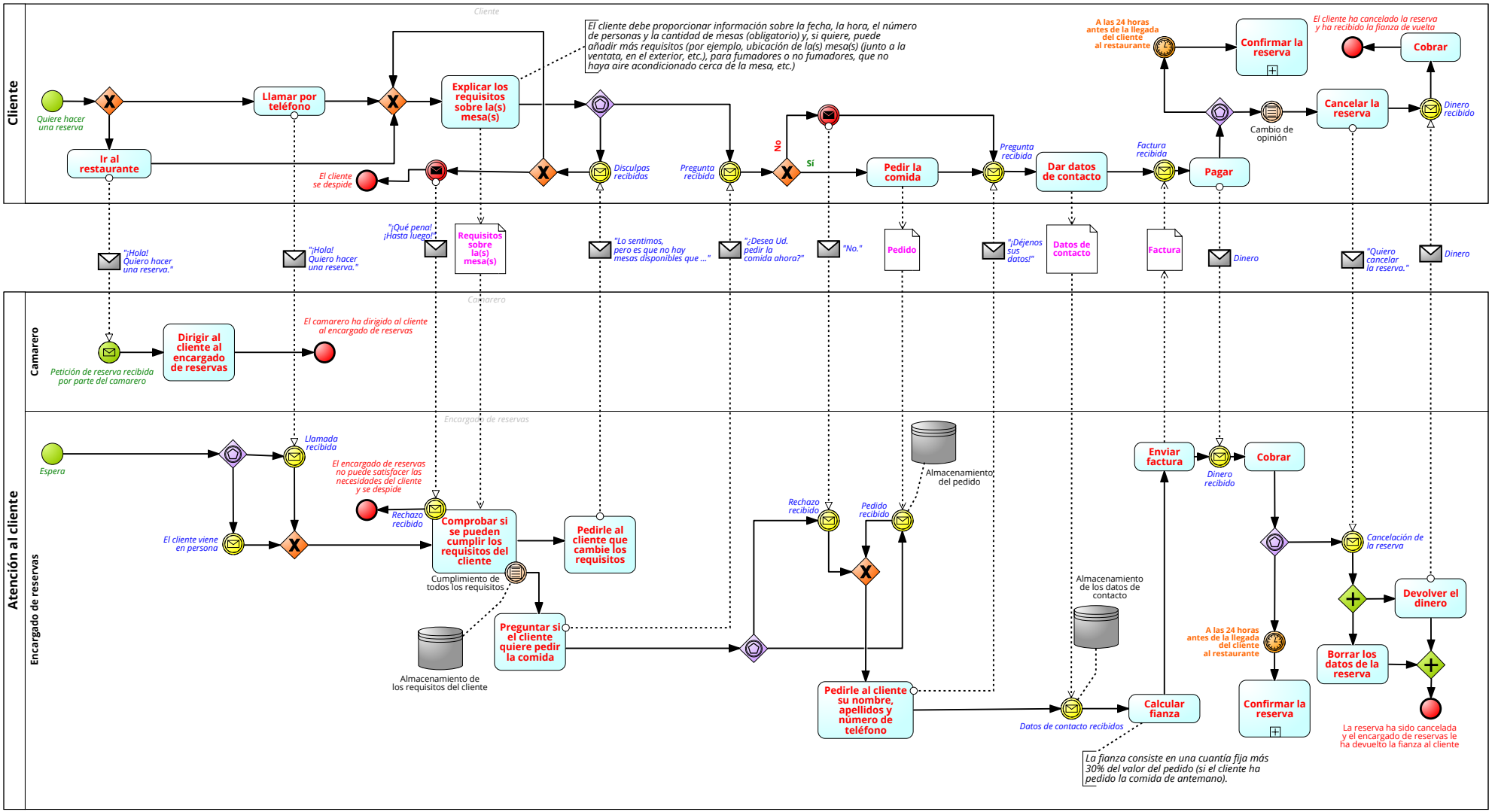
Postres

Natillas con galletas	3,00€
Arroz con leche	3,00€
Arroz con leche con dulce de leche	3,00€
Flan de huevo	3,00€
Panacota	3,50€
Tres chocolates	3,00€
Tocino de cielo	3,00€
Tarta de galleta, chocolate y vainilla	3,50€
Fruta del tiempo	1,50€
Fresas con nata	3,00€
Flan de dulce de leche	3,00€
Flan de queso	3,00€

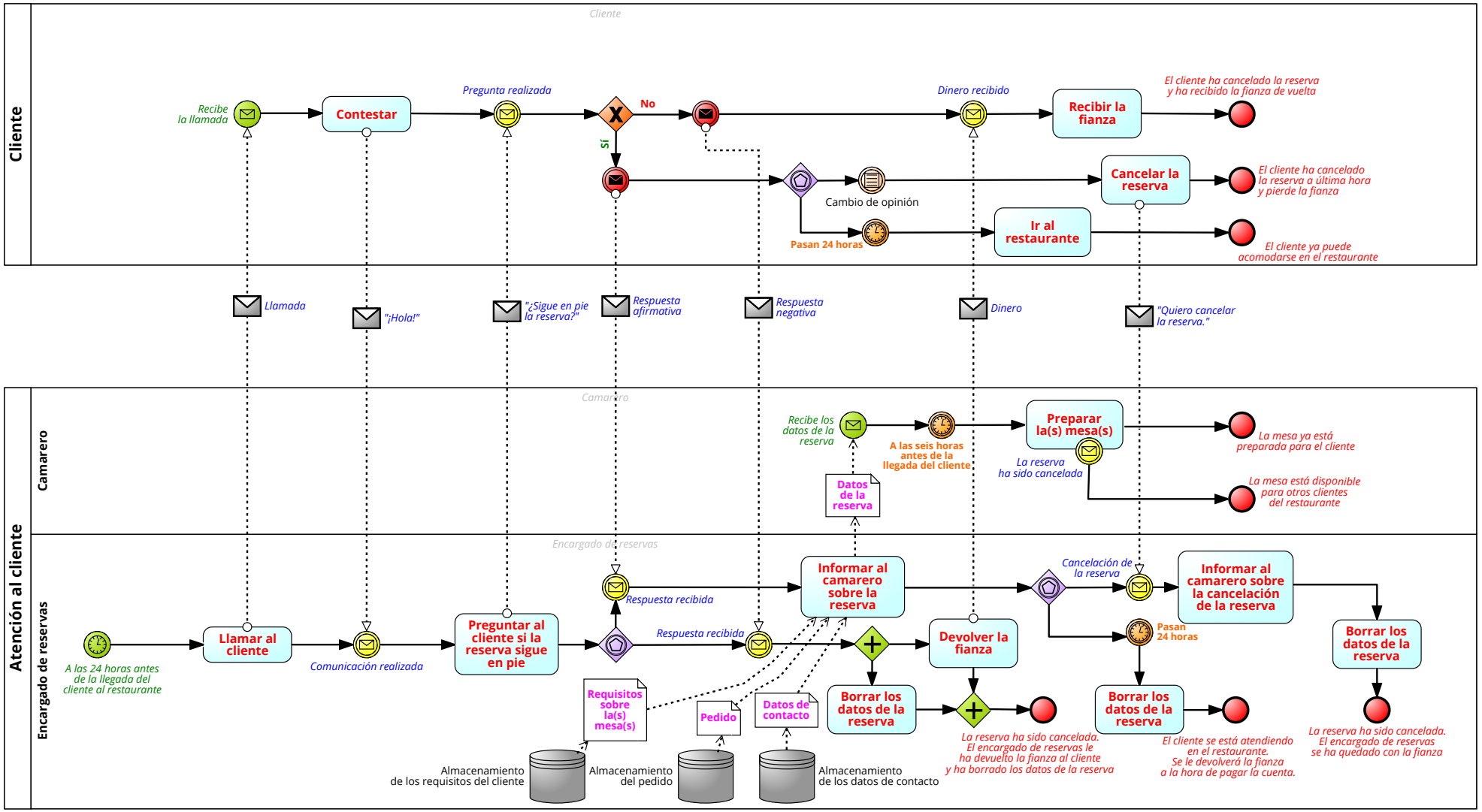
2. Gestión del restaurante (diagrama BPMN)



3. Hacer reserva (diagrama BPMN)



4. Confirmar la reserva (diagrama BPMN)



5. Llevar a cabo el pedido (diagrama BPMN)

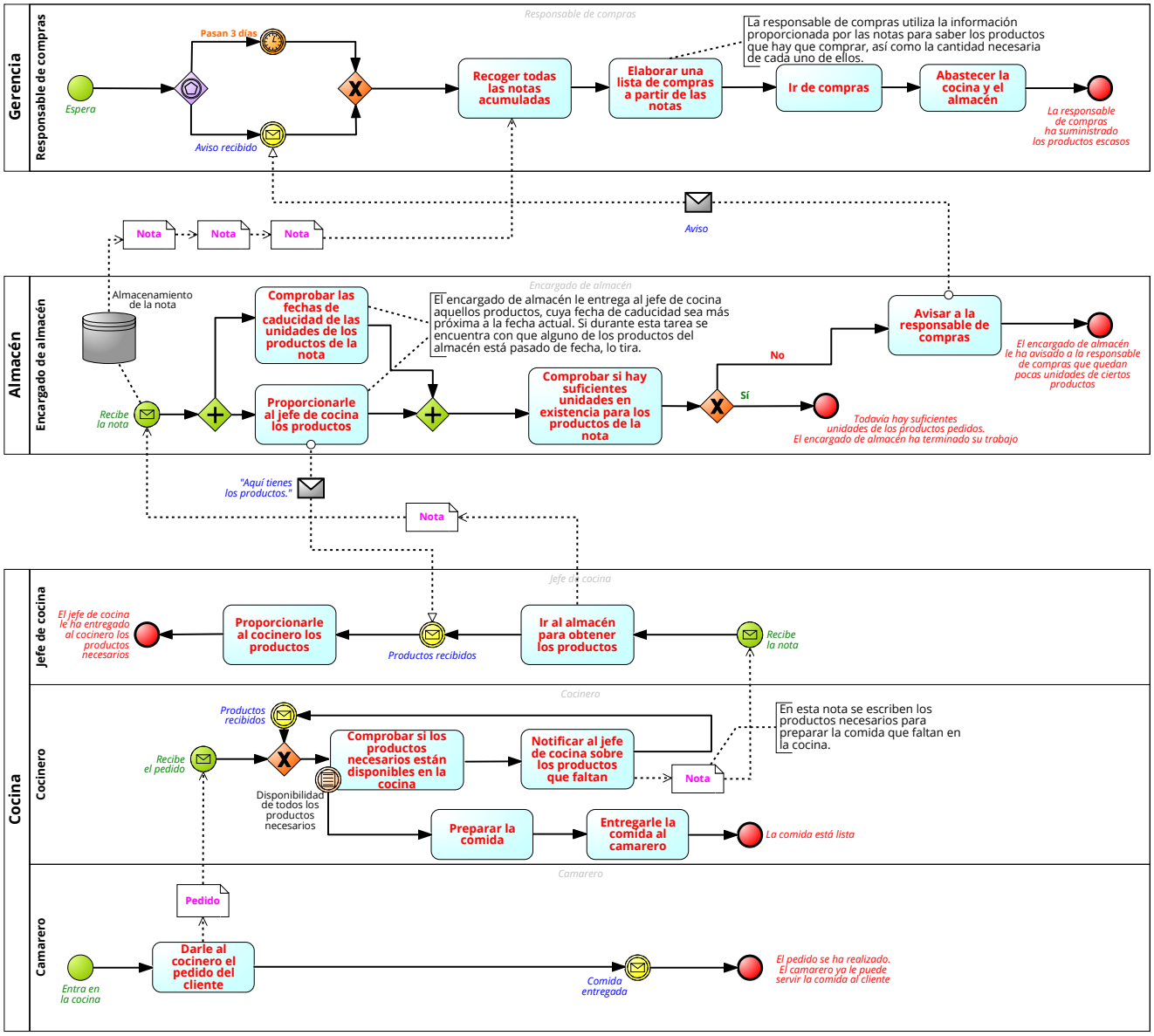


Diagrama UML 1 (Modelado de RI-001, RI-002, RI-003, RI-004 y RN-001

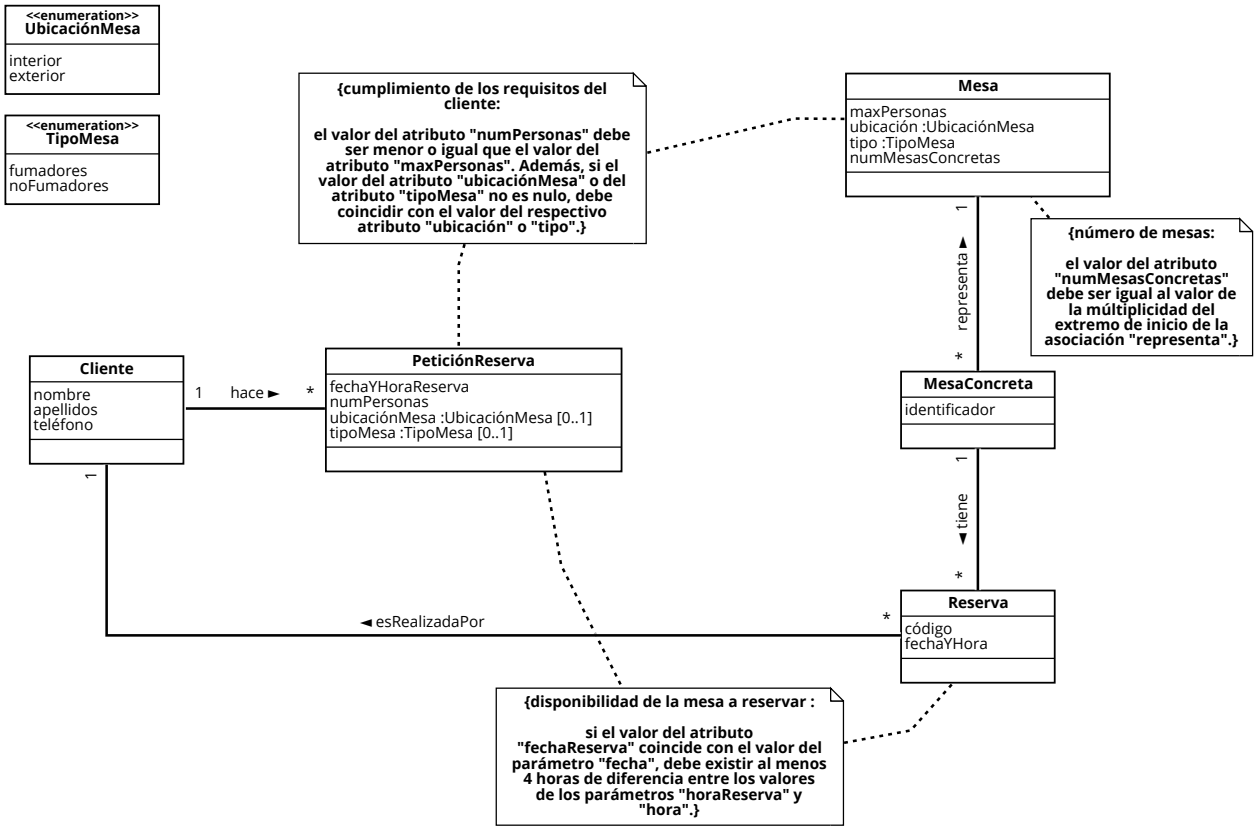


Diagrama UML 2 (Modelado de RI-005, RI-006 y RN-002)

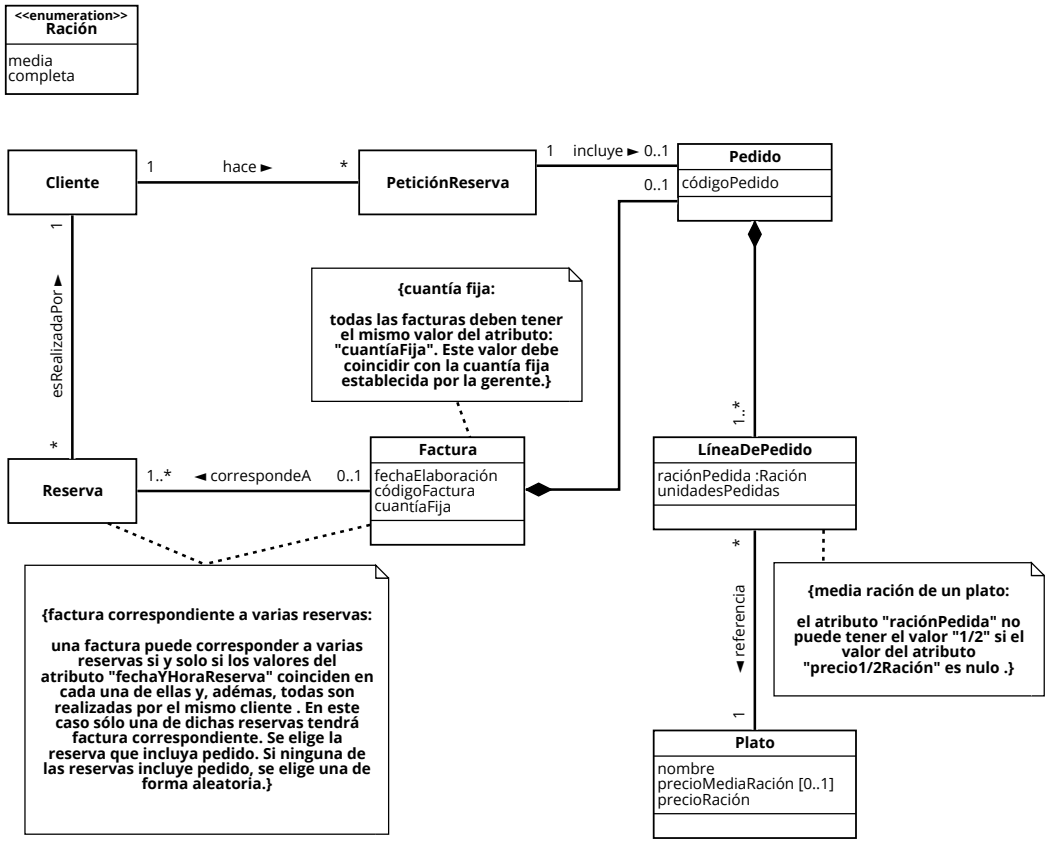
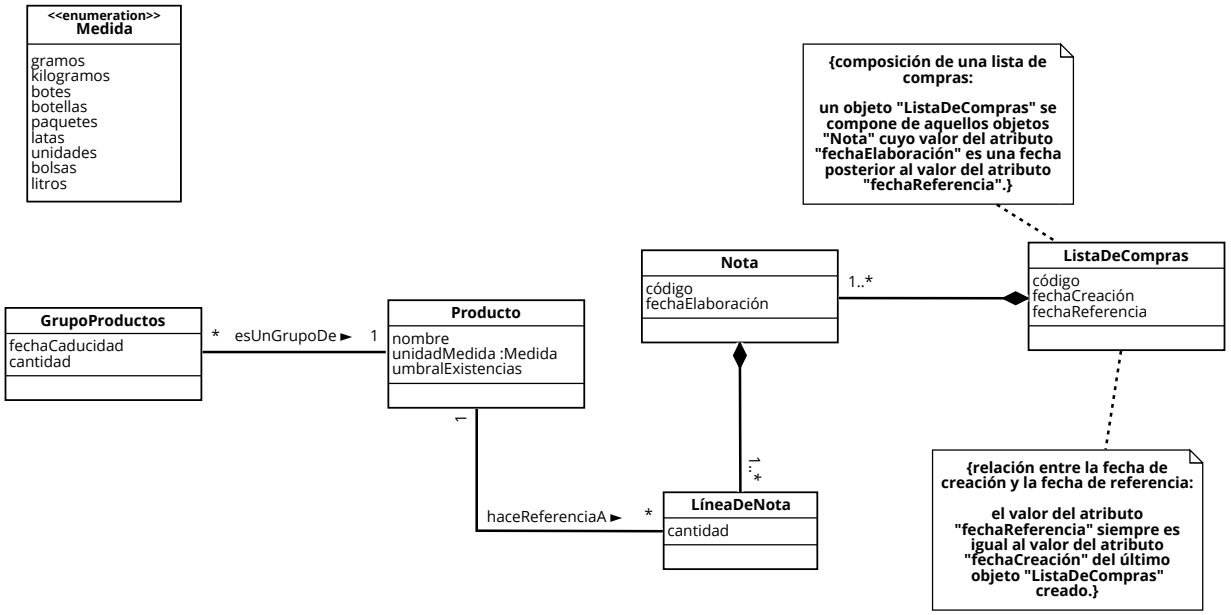
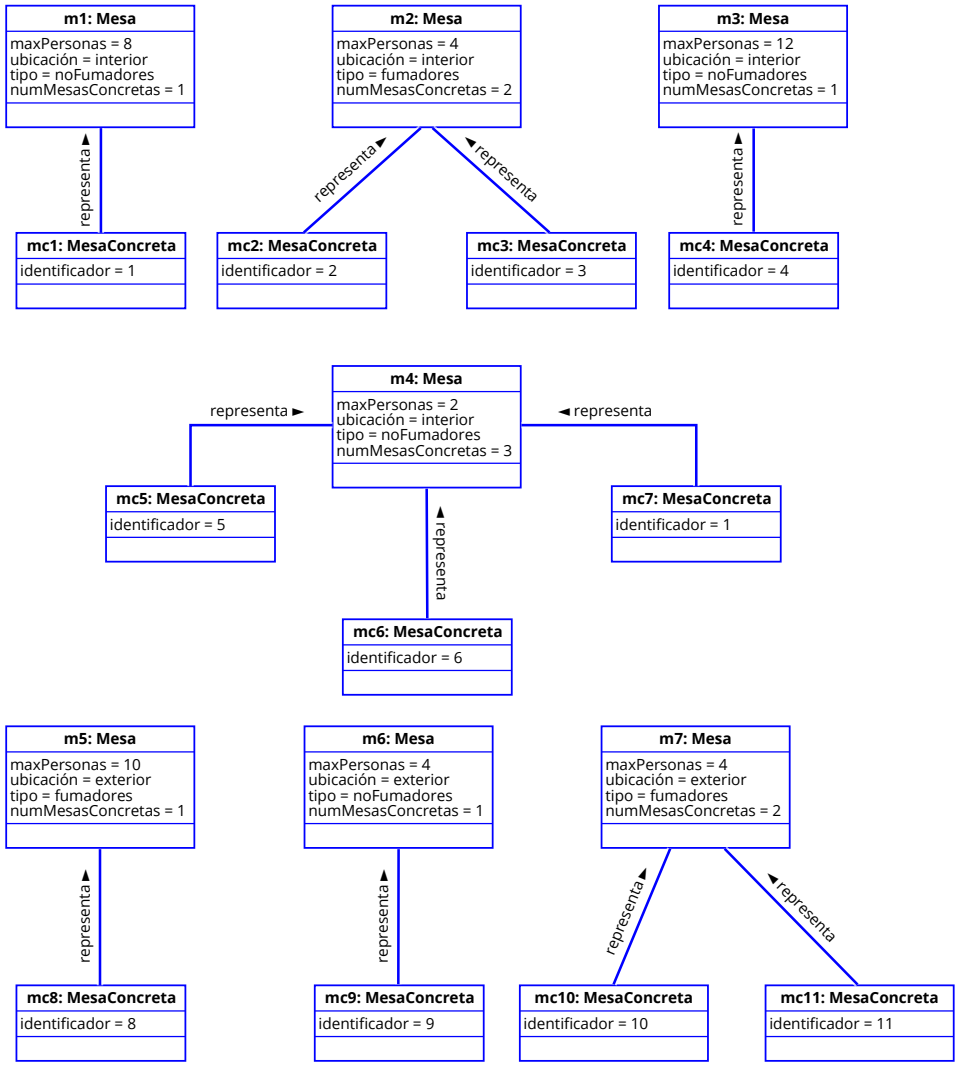


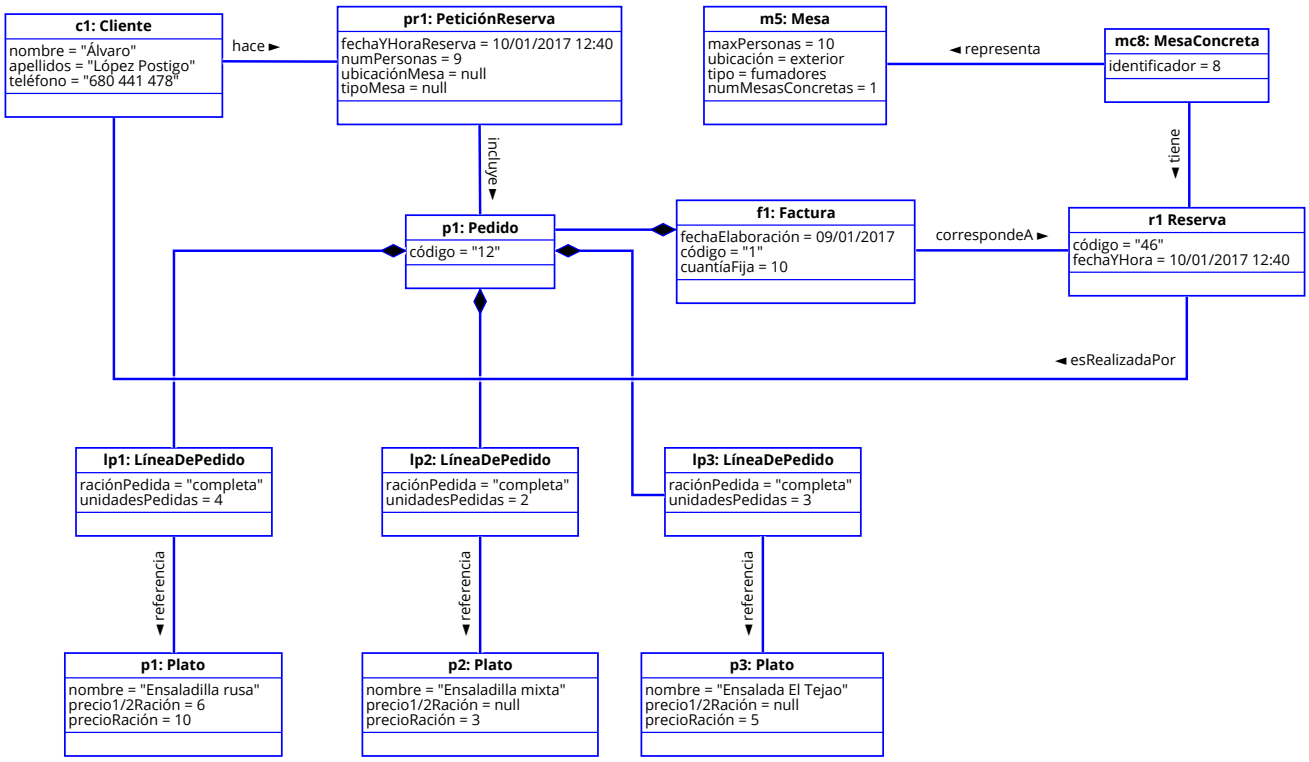
Diagrama UML 3 (Modelado de RI-007, RI-008, RI-009 y RN-003)



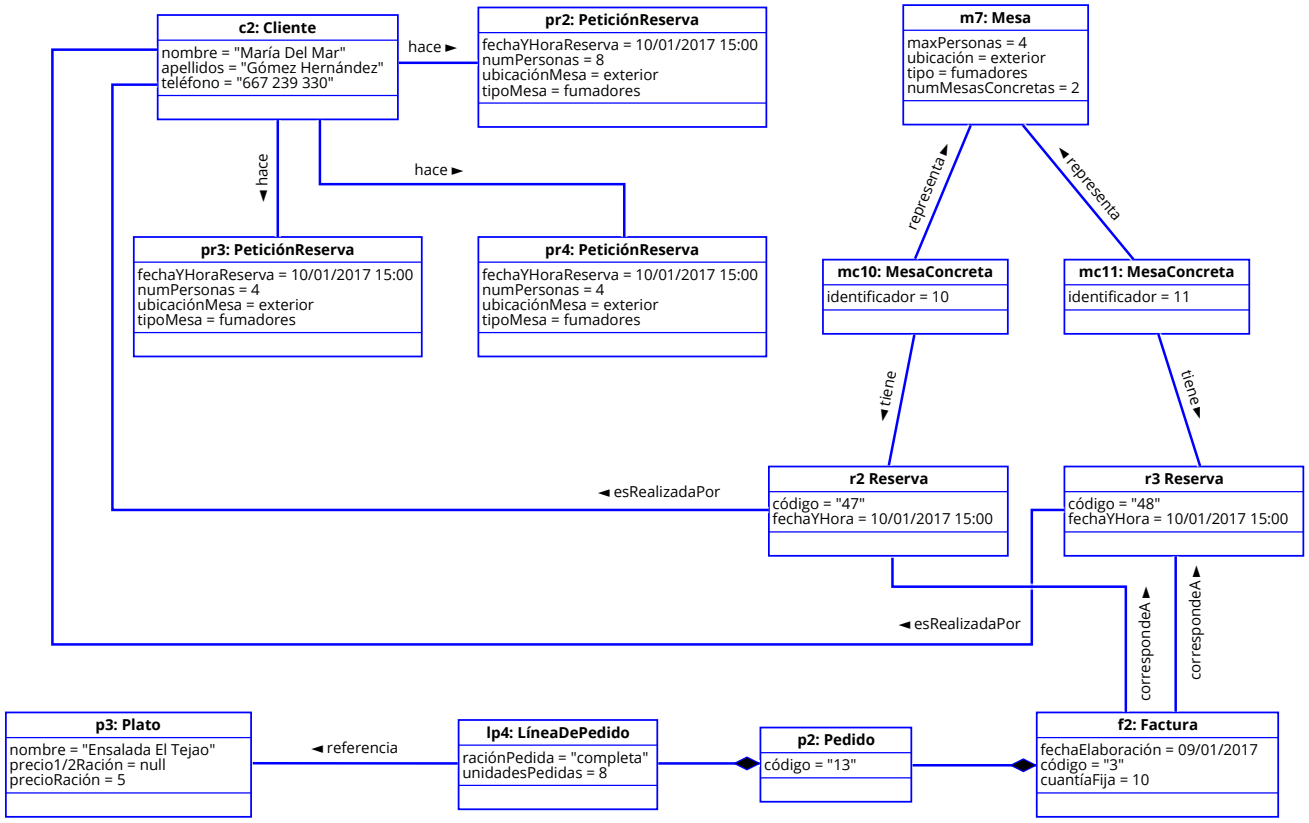
Escenario de Prueba 1 (las mesas de El Tejao)



Escenario de Prueba 1 (el cliente hace reserva)



Escenario de Prueba 1 (el cliente reserva vasrias mesas)



Escenario de prueba 2