Institución:	
Carrera:	Licenciatura en Ing. De Software
Ciclo:	2025-I
Grupo:	1101
Asignatura:	Diseño de software
Título de actividad:	Estándar IEEE 1016-Structura e Interacción.
Integrantes:	Ramírez García Oswaldo. Rios Carrera Jesús Vicente. Vargas Angeles Uriel.
Matriculas:	21-011-1167 19-011-0599 19-011-1318
Profesor:	Máximo Eduardo Sánchez Gutiérrez.

Índice

5.9 Estructura del Sistema.

La estructura del sistema de comanda digital para restaurantes define la organización interna de los componentes que lo conforman.

Esta vista de la arquitectura muestra cómo los módulos principales y secundarios se interrelacionan entre sí, asegurando una distribución lógica de responsabilidades, favoreciendo la reutilización de componentes y facilitando la escalabilidad y el mantenimiento del sistema.

La composición estructural ha sido diseñada para minimizar el acoplamiento entre módulos y maximizar su cohesión, buscando una solución flexible y sostenible a largo plazo.

5.9.1 Organización estructural del sistema.

El sistema de comanda para restaurantes como se mencionó antes, se estructura mediante una composición organizada de componentes principales y secundarios. Cada módulo del sistema tiene una responsabilidad claramente definida, lo que permite su reutilización en otros contextos similares.

La arquitectura se enfoca en garantizar un bajo acoplamiento y una alta cohesión entre los módulos, facilitando la extensibilidad, el mantenimiento y la evolución del sistema.

La estructura interna favorece la reutilización de componentes finos (como el manejo de inventario y la generación de tickets) y la composición de componentes de mayor tamaño (como el flujo de pedidos de los clientes).

5.9.2 Diseño de Elementos.

El diseño de elementos en el sistema de comanda digital para restaurantes involucra la definición detallada de las entidades que componen su estructura interna, así como las relaciones entre ellas. Estas entidades están organizadas de forma que garantizan una alta cohesión dentro de cada módulo y una mínima dependencia entre los diferentes componentes del sistema. Este enfoque asegura que el sistema sea flexible, escalable y mantenible a lo largo del tiempo.

Entidades de Diseño

El sistema está compuesto por una serie de clases e interfaces que representan los diferentes componentes y actores dentro del flujo de trabajo de la comanda. Entre las entidades clave se incluyen las clases Pedido, Carrito, Platillo, Ticket, Cocinero e Inventario, cada una con responsabilidades claramente definidas dentro del sistema.

La clase Pedido tiene como propósito gestionar la información de los pedidos realizados por los clientes, incluyendo detalles como el estado del pedido, la fecha de creación y una lista de platillos seleccionados. Por otro lado, la clase Carrito es responsable de contener los platillos seleccionados por el cliente antes de confirmar la compra. Ambas clases están estrechamente relacionadas, ya que el Carrito forma parte del Pedido.

La clase Platillo representa los elementos del menú disponibles, con atributos como el nombre, el precio y una lista de ingredientes. La clase Ticket, por su parte, es la entidad encargada de generar el comprobante de la transacción, que puede ser electrónico o impreso, y refleja los detalles del pedido y su estado.

Las interfaces como GestiónDeInventario definen las operaciones necesarias para interactuar con los componentes que gestionan los recursos, como los platillos disponibles, permitiendo su actualización o consulta conforme se realiza el pedido.

Relaciones de Diseño

El sistema está estructurado mediante relaciones bien definidas entre sus componentes. Las entidades no existen de manera aislada, sino que interactúan entre sí a través de conexiones lógicas y organizadas.

Por ejemplo, la relación entre el Cliente y el Pedido se establece a través de un conector que vincula el inicio de un pedido con su seguimiento hasta su finalización.

El Carrito es un componente crucial dentro del flujo del pedido. Permite al cliente seleccionar los platillos que desea pedir, y una vez que este proceso se confirma,

el Carrito se convierte en una parte integral del Pedido. En este momento, el Carrito transmite la lista de platillos seleccionados y otros detalles como la cantidad y los precios, los cuales se integran al Pedido. Este, a su vez, gestiona el estado del pedido, la fecha de creación y la relación con el Ticket, el cual actúa como el comprobante de la transacción finalizada.

Por otro lado, la clase Ticket envuelve a un Pedido una vez que el cliente ha confirmado su compra, proporcionando un registro de la transacción finalizada, que puede ser impresa o electrónica.

Además, las interfaces como GestiónDeInventario permiten a los componentes del sistema acceder a la información sobre la disponibilidad de platillos y actualizarla conforme se realiza el pedido. Esta interacción asegura que el sistema funcione de manera coherente, evitando inconsistencias y errores durante el proceso de compra.

Atributos de Diseño

Cada elemento del sistema está dotado de atributos específicos que determinan su funcionamiento dentro de la arquitectura general. Los nombres de las clases y sus métodos reflejan sus responsabilidades y roles dentro del sistema, lo que ayuda a mantener la claridad y la organización del código.

El tipo de cada componente está estrechamente relacionado con su propósito funcional. Por ejemplo, las clases Pedido y Carrito son tipos de entidades que gestionan colecciones de objetos relacionados, como platillos o tickets. Los atributos, como el id_pedido, fecha_pedido, y estado_pedido, están definidos con el fin de facilitar el seguimiento y la gestión de los pedidos a lo largo de todo su ciclo de vida.

Restricciones de Diseño

El diseño también está regido por una serie de restricciones que garantizan la correcta interacción y la escalabilidad del sistema. Las restricciones de interfaz aseguran que los módulos del sistema se comuniquen adecuadamente entre sí,

respetando las expectativas de las operaciones definidas por las interfaces. Por ejemplo, la interfaz GestiónDeInventario tiene restricciones claras sobre cómo deben gestionarse las actualizaciones en la disponibilidad de los platillos, permitiendo que cualquier clase que implemente esta interfaz lo haga de manera coherente.

Las restricciones de reusabilidad son fundamentales, ya que algunos módulos están diseñados para ser reutilizados en diferentes contextos. La clase Ticket, por ejemplo, puede ser utilizada para generar distintos tipos de tickets en el sistema, independientemente de su formato o método de presentación.

Por último, las restricciones de dependencia aseguran que las relaciones entre los componentes sean lo más desacopladas posible. El Cocinero, por ejemplo, depende del Pedido para saber qué platillos preparar, pero no requiere conocer detalles sobre el Cliente o el Carrito. Esta separación de responsabilidades facilita la mantenibilidad y la extensión del sistema.

5.9.3 Diagramas Estructurales de Diseño.

En esta sección se presentan los diagramas, los cuales documentan los elementos, relaciones y composiciones internas del sistema, representando su organización y estructura de manera formal.

Se emplean tres tipos de diagramas sugeridos en el estándar:

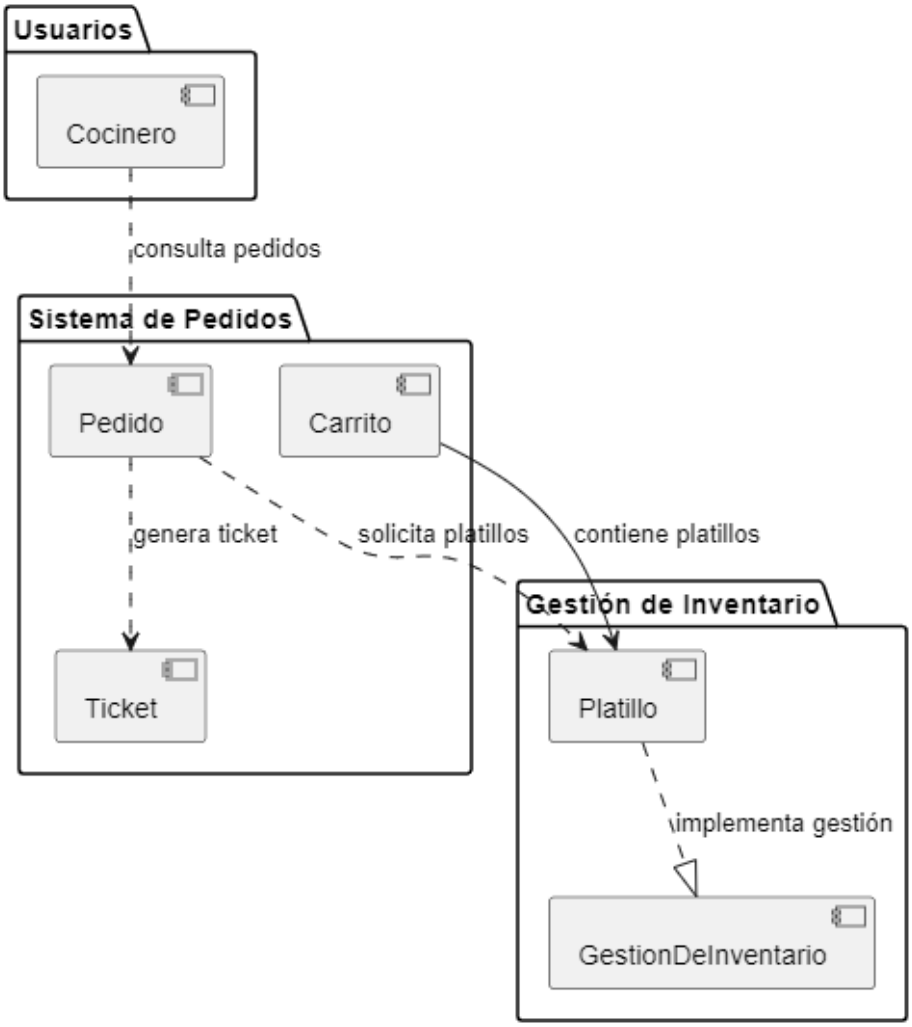
- Diagrama de Clases (UML Class Diagram)
- Diagrama de Paquetes (UML Package Diagram)
- Diagrama de Estructura Compuesta (UML Composite Structure Diagram)

Estos elementos aseguran la correcta visualización de las preocupaciones de diseño como: la composición, la modularidad, la reutilización de componentes y la definición de interfaces.

UML Package Diagram.

El Diagrama de Paquetes siguiente organiza las clases del sistema en áreas funcionales específicas; como Inventario que hace gestión de platillos y su disponibilidad, Pedidos lleva a cabo la gestión de carritos y de pedidos, y finalmente Cocina la cual hace la preparación de platillos.

Cada paquete agrupa lógicamente las clases que comparten una misma responsabilidad funcional.



El diagrama muestra la organización general del sistema en tres paquetes principales: "Sistema de Pedidos", "Gestión de Inventario" y "Usuarios". Dentro de cada paquete se agrupan las clases relacionadas, permitiendo observar las dependencias entre ellas. Por ejemplo, "Pedido" solicita platillos y genera un ticket,

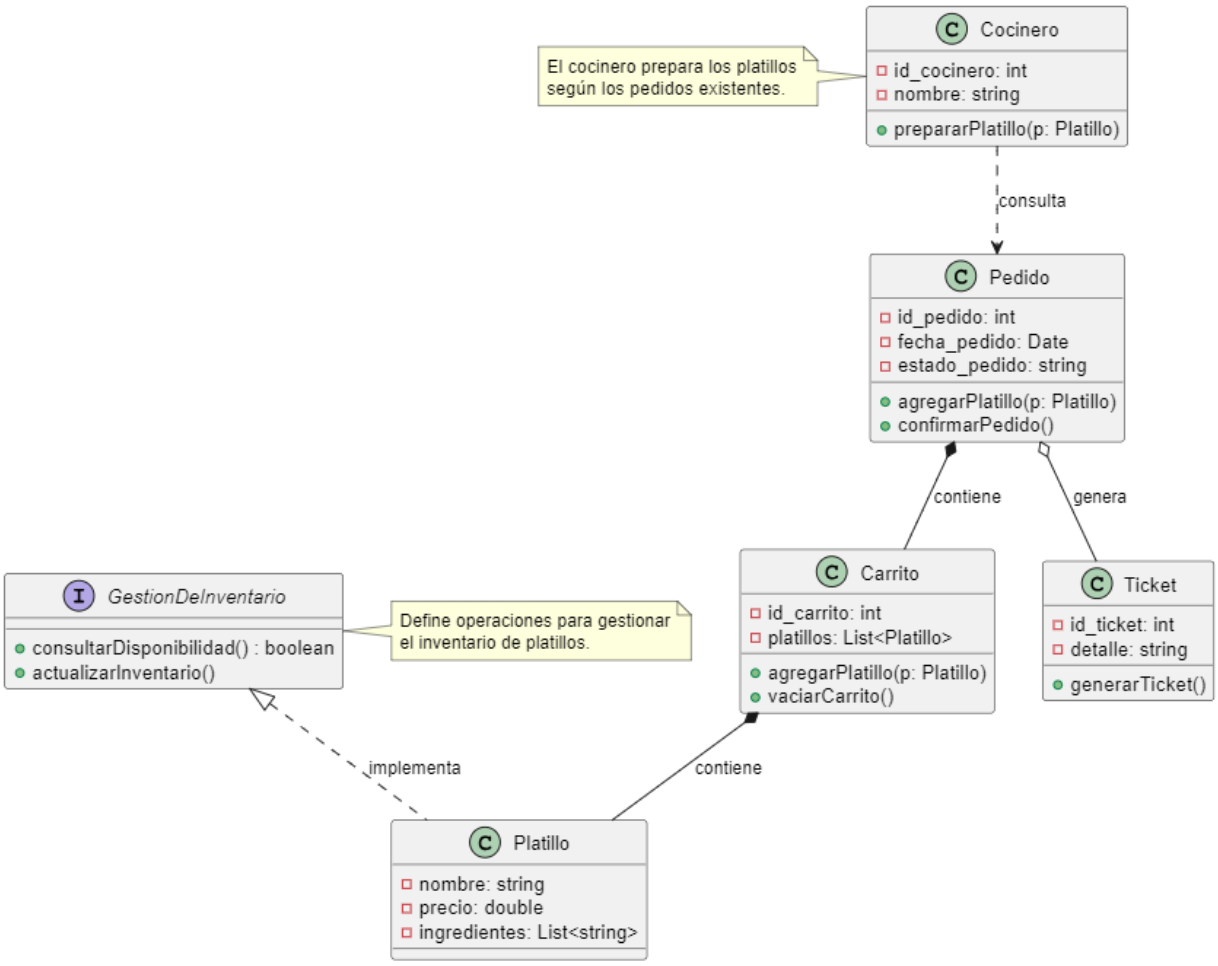
mientras que "Platillo" está relacionado con la gestión de inventario. Además, el "Cocinero" puede consultar los pedidos existentes. Esta estructura ayuda a modularizar el sistema y facilita el mantenimiento.

UML Class Diagram.

El Diagrama de Clases siguiente describe las principales entidades del sistema, incluyendo Pedido, Carrito, Platillo, Ticket y Cocinero.

Se modelan atributos y operaciones relevantes, así como las relaciones de composición, asociación, e implementación de interfaces.

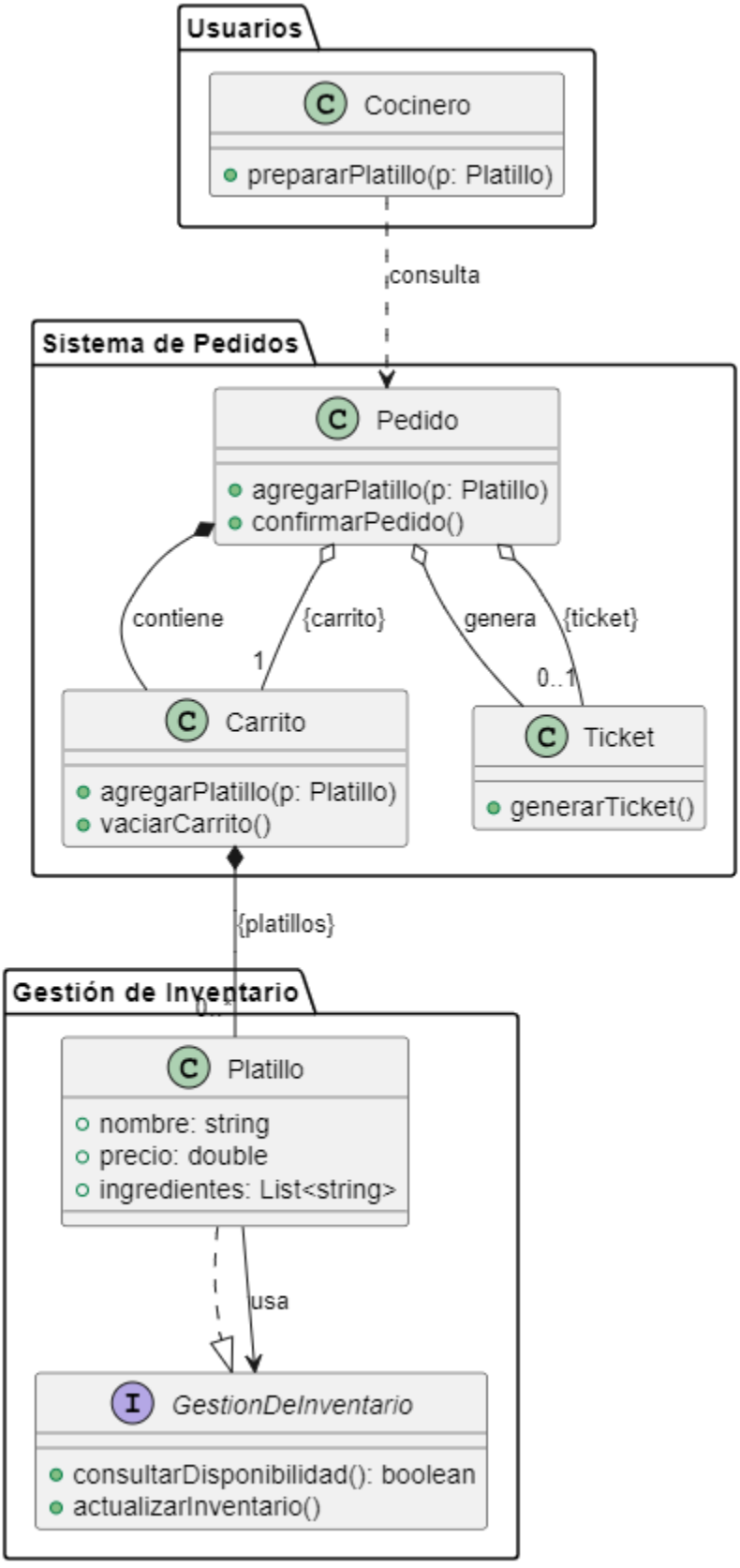
Este diagrama refleja la estructura lógica del sistema y la interacción entre los objetos principales.



El diagrama representa las clases principales que conforman el sistema y sus relaciones. Se definen atributos y métodos relevantes para "Pedido", "Carrito", "Platillo", "Ticket" y "Cocinero". También se incluye una interfaz llamada "GestionDeInventario" que es implementada por la clase "Platillo". Se puede observar cómo "Pedido" contiene un "Carrito" y genera un "Ticket", mientras que el "Carrito" almacena varios "Platillos". Este diseño busca reflejar las responsabilidades de cada entidad de forma clara y organizada.

UML Composite Structure Diagram.

El Diagrama de Estructura Compuesta que se muestra a continuación, representa la estructura interna de las entidades del sistema, visualizando las relaciones entre componentes, interfaces y puertos de comunicación. Se describen explícitamente las composiciones internas y las conexiones, estableciendo de manera detallada cómo interactúan las partes del sistema en tiempo de ejecución.


Usuarios
Cocinero
prepararPlatillo(p: Platillo)
consulta
Sistema de Pedidos
Pedido
agregarPlatillo(p: Platillo)
confirmarPedido()
contiene
{carrito}
genera
{ticket}
1
0..1
Carrito
agregarPlatillo(p: Platillo)
vaciarCarrito()
Ticket
generarTicket()
{platillos}
Gestión de Inventario
Platillo
nombre: string
precio: double
ingredientes: List<string>
usa
GestionDeInventario
consultarDisponibilidad(): boolean
actualizarInventario()

En este diagrama se muestra cómo están compuestos internamente los elementos del sistema. "Pedido" se conecta con un "Carrito" y un "Ticket", indicando que puede tener uno o ninguno en el caso del ticket. El "Carrito" puede contener varios "Platillos" y estos a su vez interactúan con la interfaz de gestión de inventario. El "Cocinero" tiene acceso a los pedidos para preparar los platillos. Esta vista permite entender la composición y las interacciones internas entre los distintos componentes.

Los diagramas presentados permiten visualizar de forma clara la organización interna del sistema, sus relaciones de composición y dependencia, así como la definición explícita de interfaces y componentes reutilizables.

5.10 Interacción del Sistema.

La vista de interacción del sistema de comanda digital para restaurantes define las estrategias y mecanismos mediante los cuales las entidades del sistema se comunican entre sí. Esta vista describe cómo, cuándo, por qué y a qué nivel se producen las acciones en el sistema, permitiendo comprender el comportamiento dinámico de los componentes durante la ejecución.

Su propósito es detallar la asignación de responsabilidades, las colaboraciones entre objetos y el flujo de mensajes necesario para cumplir con los casos de uso más relevantes. Esta visión es especialmente útil en sistemas reactivos, interactivos y distribuidos, donde la coordinación entre actores es esencial para garantizar un funcionamiento correcto.

5.10.1 Distribución de interacción del Diseño.

Las principales preocupaciones de diseño en esta vista se centran en evaluar cómo se distribuyen y coordinan las responsabilidades entre los objetos que colaboran en el sistema. En particular, se estudia la forma en que las entidades intercambian mensajes, desencadenan eventos y reaccionan ante acciones externas, como la confirmación de un pedido o la preparación de un platillo.

Esta vista es especialmente útil para validar el uso correcto de patrones de diseño y asegurar que los flujos de interacción estén bien alineados con los requisitos funcionales. También permite describir situaciones de concurrencia, coordinación en tiempo real y transiciones de estado entre componentes.

Para el sistema de comanda digital, este enfoque es esencial para representar cómo se procesan las órdenes desde que el cliente selecciona los platillos hasta que son preparados y entregados, incluyendo eventos como actualizaciones de inventario y generación de tickets.

5.10.2 Elementos de Interacciones del Diseño

El diseño de las interacciones del sistema de comanda digital para restaurantes se basa en una combinación de elementos que permiten describir cómo los componentes del sistema colaboran y reaccionan frente a eventos internos o externos. A continuación, se describen los principales elementos de diseño utilizados:

Clases y métodos: Las clases como Pedido, Carrito, Platillo y Cocinero incluyen métodos que definen comportamientos importantes para la interacción del sistema. Por ejemplo, Pedido.confirmarPedido() y Carrito.agregarPlatillo() representan acciones clave disparadas por eventos generados por el usuario.

Eventos y señales: Las interacciones se activan a partir de eventos como la confirmación de un pedido, el inicio de la preparación de un platillo o la actualización del inventario. Estos eventos desencadenan mensajes o señales entre componentes, como el paso del pedido a la cocina o la reducción de ingredientes disponibles.

Estados: Algunos elementos del sistema atraviesan diferentes estados. Por ejemplo, un Pedido puede estar en estado pendiente, en preparación, o entregado. Estos estados permiten definir la lógica de transición entre acciones del sistema.

Jerarquía y concurrencia: Las entidades están organizadas jerárquicamente (por ejemplo, Carrito forma parte de un Pedido), y ciertas actividades pueden ejecutarse

de forma concurrente. Mientras un cliente realiza una nueva orden, otro pedido puede estar siendo preparado por un cocinero, lo que ilustra un diseño concurrente y distribuido.

Sincronización: Se utilizan mecanismos de sincronización implícitos para asegurar que ciertas acciones no se superpongan o generen conflictos. Por ejemplo, un platillo no puede prepararse si el inventario no ha confirmado la disponibilidad de los ingredientes.

A continuación, se presenta una tabla con ejemplos clave de los elementos de interacción del diseño:

Elemento	Ejemplo en el sistema
Clase	Pedido, Carrito, Platillo, Cocinero
Método	Pedido.confirmarPedido(), Cocinero.prepararPlatillo()
Estado	Pedido: *pendiente*, *en preparación*, *servido*
Evento	Cliente presiona “Confirmar Pedido” en la interfaz
Señal	Sistema emite “Orden enviada a cocina” cuando el pedido se confirma
Jerarquía	Pedido contiene un Carrito, y el Carrito una lista de Platillo
Concurrencia	Varios Cocineros pueden preparar diferentes platillos al mismo tiempo
Sincronización	Los Pedidos se procesan de forma ordenada en la cocina para evitar colisiones

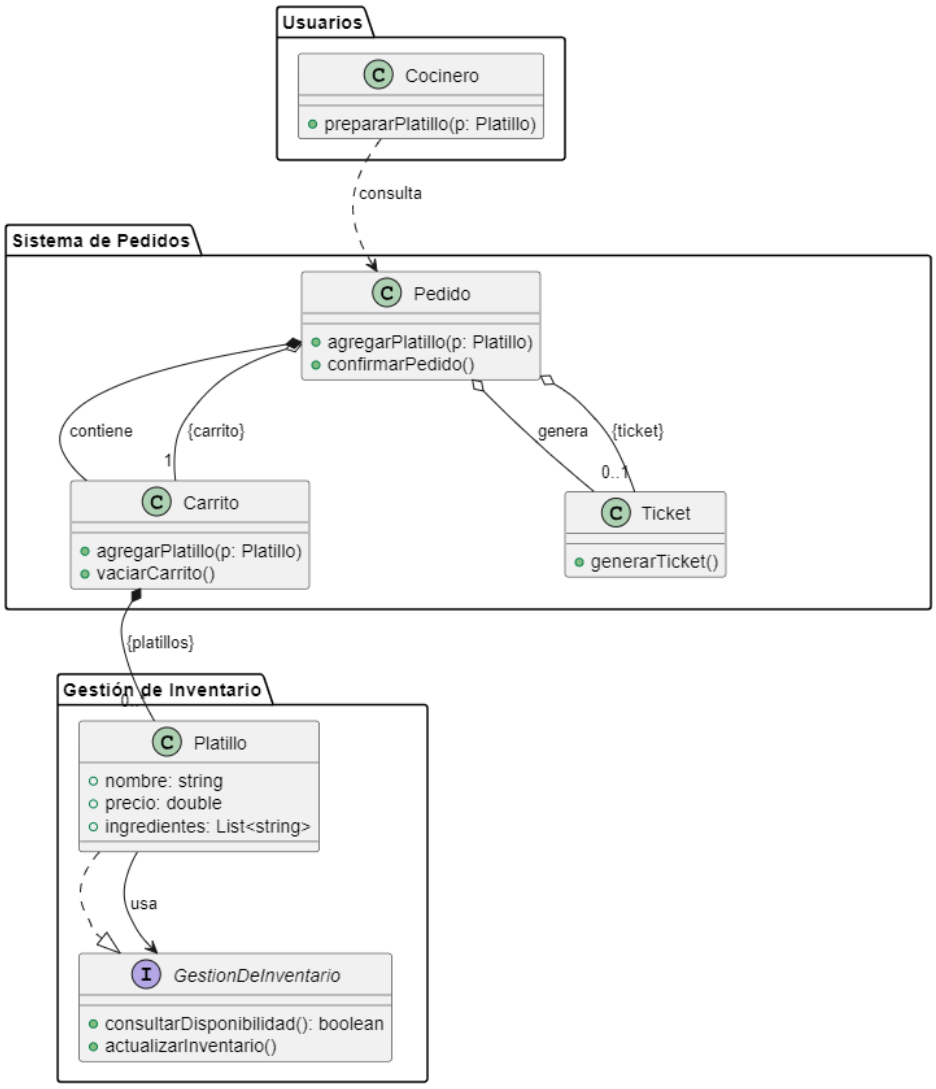
Estos elementos en conjunto permiten modelar adecuadamente las interacciones en tiempo real, asegurar la coordinación entre módulos y garantizar la respuesta adecuada del sistema ante diferentes escenarios operativos.

5.10.3 Interacción del sistema.

Para ilustrar las interacciones del sistema, se presentan tres diagramas que muestran diferentes perspectivas del comportamiento dinámico entre los componentes principales de la aplicación de comanda digital para restaurantes.

Diagrama de Estructura Compuesta.

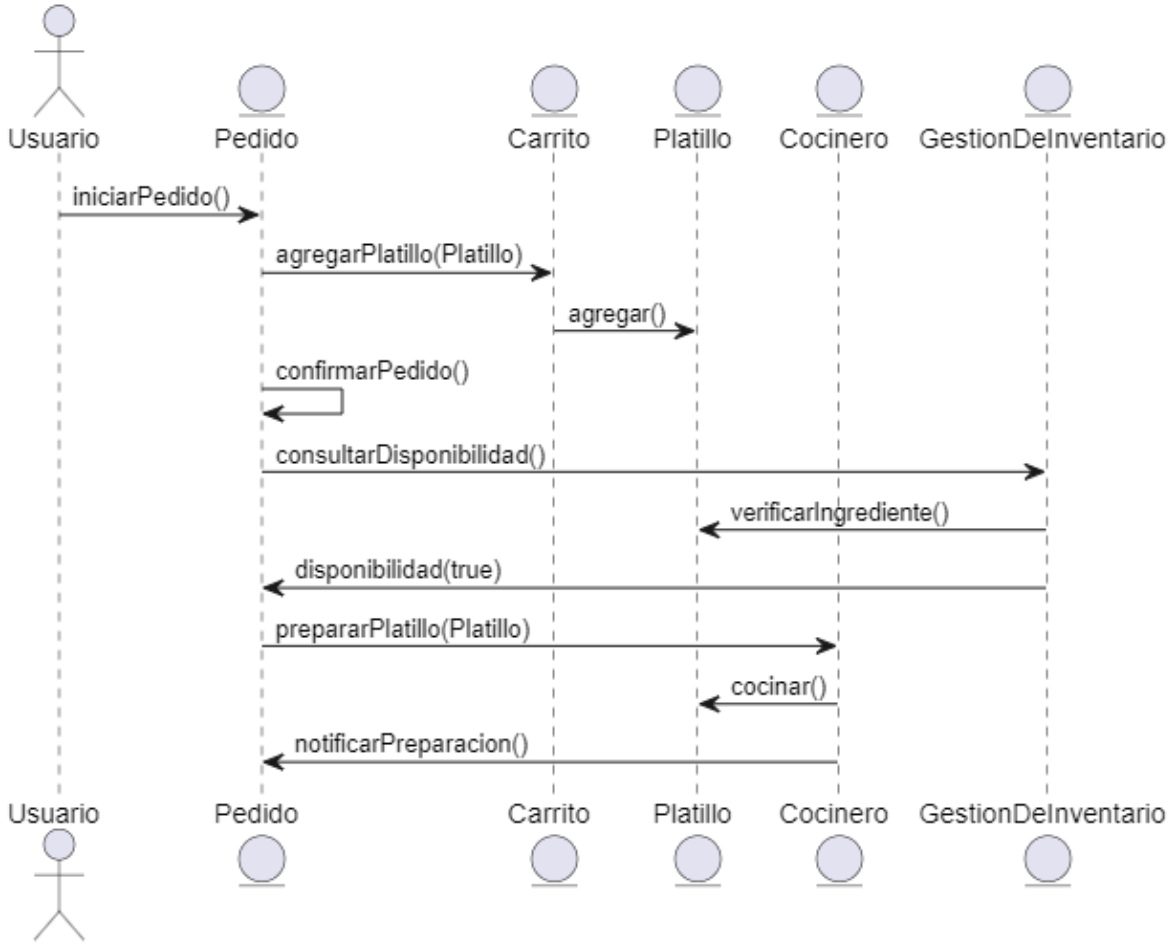
El primer diagrama corresponde a una vista estructural interna que permite comprender cómo se organizan y se comunican los elementos del sistema desde una perspectiva estática. Se representa la composición del objeto Pedido, el cual contiene un Carrito y puede generar un Ticket. A su vez, el Carrito se relaciona con varios Platillos, los cuales interactúan con la interfaz GestionDeInventario. Esta estructura visualiza las posibles conexiones entre los objetos y la existencia de relaciones jerárquicas y de colaboración dentro del sistema.



Este diagrama permite comprender dónde ocurren las interacciones y quiénes están involucrados.

Diagrama de Secuencia (Interacción normal).

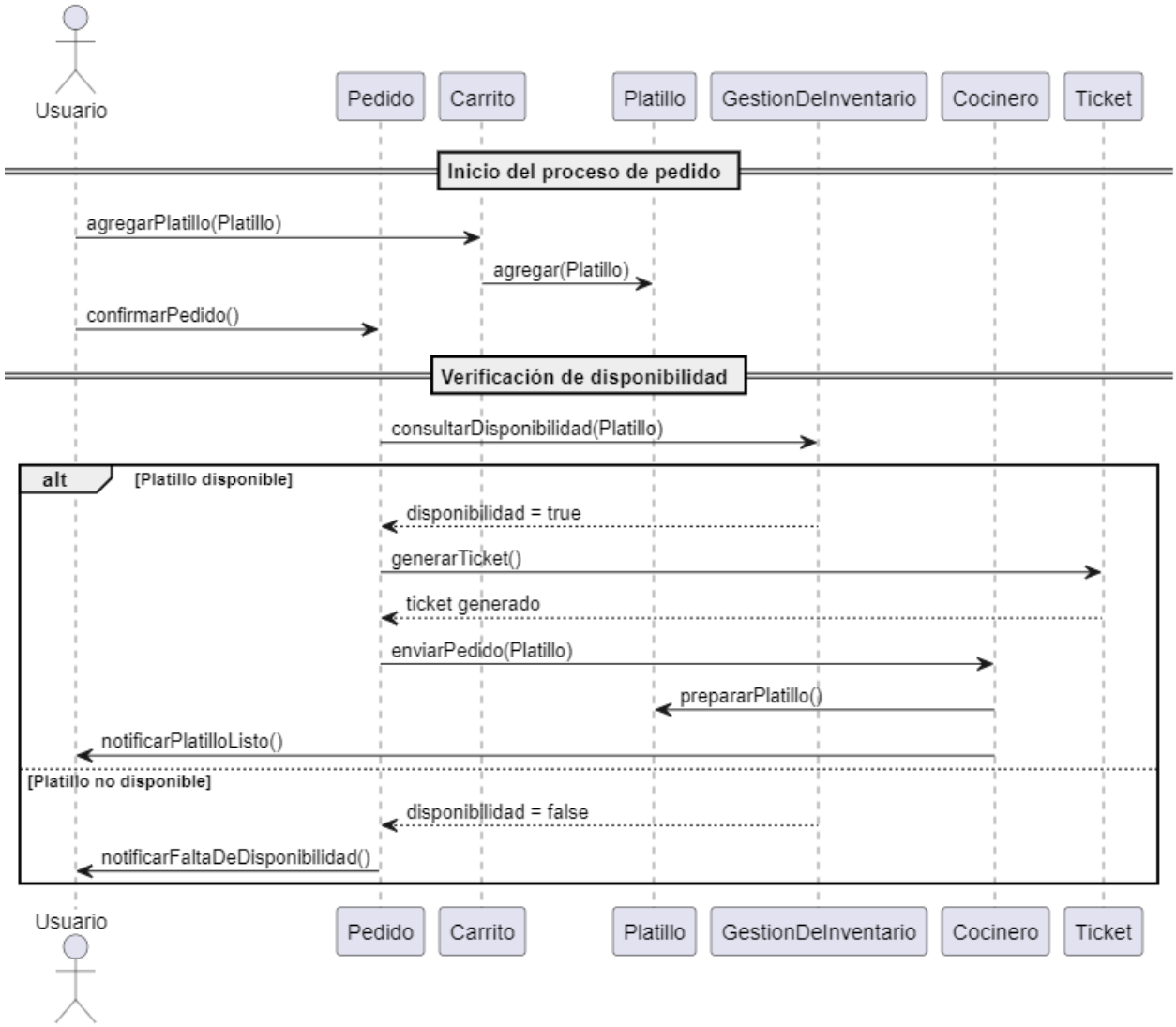
A continuación, se presenta un diagrama de secuencia simple que describe el flujo de interacción entre el Cliente, el sistema, y el Cocinero. El Cliente agrega platillos al Carrito, confirma su pedido y este es enviado al Cocinero. Este flujo muestra la asignación de responsabilidades de manera secuencial y clara, ayudando a entender cómo ocurren las acciones en el sistema durante una operación típica.



Esta vista facilita observar el orden temporal de los eventos y cómo colaboran los distintos componentes para cumplir una acción.

Diagrama de Secuencia (Interacción con condición de disponibilidad).

Finalmente, se presenta un segundo diagrama de secuencia más completo, que agrega condiciones de disponibilidad del platillo antes de confirmar un pedido. Aquí se detalla la validación con GestionDeInventario y se contempla el caso en el que el platillo no está disponible. Este enfoque permite visualizar escenarios reales del sistema, con bifurcaciones lógicas que representan decisiones dinámicas en tiempo de ejecución.



Este diagrama es especialmente útil para describir concurrencia, decisiones, estados y sincronización.

En conjunto, estos diagramas permiten validar y refinar las decisiones de diseño, ofreciendo una representación clara y precisa del comportamiento interactivo del sistema.