UTN
Facultad Regional Córdoba

TUP

Tecnicatura Universitaria en Programación

METODOLOGÍA DE SISTEMAS I

Unidad Temática N.º 4:
U.M.L.

Guía Práctica
2° Año – 3° Cuatrimestre

Índice

Introducción 2

Diagrama de clases 3

Problema 1 “Lavadero” (Ejemplo) 3

1. Identificación de clases del dominio 3
2. Modelado de clases del dominio 8
3. Identificar atributos 13
4. Identificar métodos/operaciones 14

Problema 2: “Comida Argentina” 16

Problema 3: “CAR WASH” (Resuelto) 17

Modelo de máquina de estado 21

Problema 4: “E-Commerce” 21

1. Revisar el Diagrama de Clases 21
2. Identificar los Estados Importantes 22
3. Determinar los Eventos y Transiciones 23
4. Crear el Diagrama de Máquina de Estado 23
5. Refinar el Diagrama con Condiciones Guardas (Opcional) 24
6. Revisar y Validar 24
7. Completar el diagrama de clases 24

Problema 5: “Fórmula Uno” 25

Solución Problema 5: “Fórmula Uno” 26

Diagrama de Clases del Campeonato de Fórmula Uno (con clase abstracta Etapa) .. 26

Diagrama de Clases del Campeonato de Fórmula Uno (sin clase Etapa) 29

Diagrama de Estados de CARRERAS del Campeonato de Fórmula Uno 31

Introducción

Un modelo ayuda al equipo de trabajo a comunicar la visión del sistema que se está construyendo. Permite especificar la estructura y conducta del sistema.

Un modelo permite documentar la estructura y conducta de un sistema antes de que sea transformado en código.

Objetivos de la Unidad 3

Que el alumno sea capaz de:

- Analizar un caso práctico y representar el problema con un modelo utilizando UML.
- Reconocer e identificar los bloques de construcción y reglas de sintaxis de UML.
- Analizar un caso práctico y construir un modelo consistente con UML.
- Modelar clases del dominio del problema.
- Modelar estados de un objeto.
- Modelar secuencia de mensajes entre clases.

Modelo del dominio del problema

Un modelo de dominio es una representación de las clases conceptuales del mundo real, no de componentes de Software. No se trata de un conjunto de diagramas que describen clases software, u objetos software con responsabilidades.

(Larman, 2002)

El modelo del dominio del problema se puede representar mediante un diagrama de clases y un diagrama de casos de uso. Vamos a comenzar definiendo cómo hacer un modelo del dominio con un diagrama de clases. Para ello, tomaremos de ejemplo el primer ejercicio de la guía:

Diagrama de clases

Problema 1 “Lavadero” (Ejemplo)

Un servicio de lavado rápido realiza distintos tipos de servicio (referidos a distintas formas de lavado: lavado a seco, con máquina lavadora, etc.) a las prendas que habitualmente traen sus clientes.

Cuando un cliente llega, se le toman sus datos personales: apellido y nombre del cliente, domicilio (calle, nro, dpto y piso), barrio y teléfono. A continuación, se le pregunta el tipo de servicio requerido y la/s prenda/s que va a dejar para ello, de esta forma se le genera un número de pedido, y se registra la fecha de pedido, el tipo de prenda, tipo de servicio para esa prenda, la cantidad de prendas que deja por cada tipo de servicio, y se le informa al cliente el precio unitario, y el precio total de lo pedido, informando la fecha posible de entrega.

Un cliente puede requerir en un mismo pedido, distintos servicios para cada prenda. En general un tipo de servicio puede aplicarse a cualquier tipo de prenda.

Consigna:

1. Lea detenidamente los ejercicios y subraye los objetos y clases.
2. Represente gráficamente, mediante UML, cada una de las clases identificadas y asigne un Nombre, los Atributos y las Operaciones.
3. Relacione las clases según corresponda.

1. Identificación de clases del dominio

El primer paso en la definición de la esencia del problema es seleccionar las clases del sistema. Los siguientes consejos nos ayudarán a identificar las clases de nuestro dominio de problema:

a) Leer cuidadosamente la especificación de requerimientos (el enunciado en nuestro caso), buscando frases sustantivas.

Frase sustantiva: Una frase sustantiva es un grupo de palabras que tiene como núcleo un sustantivo, y funciona como un sustantivo dentro de la oración. Puede estar compuesta por un sustantivo solo o acompañado de modificadores (como adjetivos, determinantes o complementos) que agregan información sobre el sustantivo. Ejemplo: “El libro rojo”, “Una casa en la playa”, “El perro negro”, “Los niños que juegan en el parque”.

Algunas frases sustantivas de nuestro ejercicio:

- Un servicio de lavado rápido
- Distintos tipos de servicio
- Distintas formas de lavado
- Lavado a seco
- Máquina lavadora
- Las prendas
- Sus clientes
- Sus datos personales
- Apellido y nombre del cliente
- Domicilio
- Barrio y teléfono
- Un número de pedido
- La fecha de pedido
- El tipo de prenda
- La cantidad de prendas
- El precio unitario
- El precio total
- La fecha posible de entrega

b) Cambiar todos los plurales por singular y hacer una lista preliminar:

- Servicio
- Forma de lavado
- Lavado a seco
- Dato Personal
- Tipo de servicio
- Máquina lavadora
- Barrio
- Teléfono
- Fecha de pedido
- Cantidad de prenda
- Precio unitario
- Precio total
- Fecha de posible entrega
- Prenda
- Cliente
- Domicilio
- Pedido
- Tipo de Prenda

c) Observando la lista se dividirán los ítems en tres categorías: clases obvias, absurdas y dudosas. Por supuesto deben descartarse las absurdas. De las otras dos categorías pueden obtenerse clases candidatas.

Clases obvias	*Clases dudosas*	*Clases absurdas*
Servicio	Cliente	Máquina lavadora
Tipo de servicio	Domicilio	Forma de lavado
Prenda		Lavado a seco
Pedido		Dato personal
Detalle de pedido		Barrio
		Teléfono
		Fecha de pedido
		Precio unitario
		Precio total

Tabla (1) Elaboración propia

- **Clases obvias**: Estas son clases que claramente representan objetos del dominio. Estas clases parecen ser fundamentales para el sistema y tienen un papel activo en la interacción con el usuario o en la lógica del negocio.
- **Clases dudosas:** Son aquellas que requieren un análisis más profundo para decidir si deben ser clases o no. Por ejemplo, en una lavandería, Cliente es dudosa porque su relación puede ser indirecta, gestionada por pedidos. Un ejemplo donde Cliente sea una clase obvia es en una tienda en línea. Aquí, Cliente es fundamental porque realiza compras, gestiona envíos y pagos.
- **Clases absurdas**: Estas son entidades que no tienen sentido como clases dentro del contexto del dominio que estamos modelando, que son más bien detalles específicos de implementación o funcionalidad.

Es **importante** ajustar la identificación de clases en función del análisis del dominio específico que se está modelando. ***Lo que es obvio en un dominio puede no serlo en otro, y lo dudoso en un caso puede ser central en otro contexto***. Por eso, este análisis iterativo de clases ayuda a refinar el modelo y asegurar que las clases candidatas representan correctamente las entidades del dominio.

Otro criterio, aparte del sentido común, que se puede usar para decidir qué frases sustantivas son candidatas significativas para ser clases es buscar entre estas categorías de abstracciones:

- Objetos tangibles o físicos: Avión, asiento, billete, equipaje, tarjeta de embarque.
- Roles: Papeles desempeñados por personas u organizaciones: Piloto, agente de ventas, pasajero.
- Incidentes: Representan la ocurrencia de un evento, algo que sucede en un momento determinado: Cancelación de vuelo, vuelo, aterrizaje, colisión.
- Interacciones: transacciones o contratos que se dan entre dos o más objetos del modelo: Reserva, venta de billete, pago.
- Líneas de las transacciones: Línea de venta.
- Especificaciones propias de aplicaciones de inventario o fabricación, relacionados con los estándares o definiciones de elementos.
- Descripciones: Descripción del vuelo.
- Organizaciones: Departamento de ventas, compañía aérea.
- Lugares: Tienda.
- Contenedores: Avión, tienda, lata.
- Cosas contenidas: Artículo, pasajero.
- Conceptos abstractos: Ansiedad, claustrofobia.
- Otros sistemas informáticos externos al sistema: Control de tráfico aéreo, sistema de autorización de pago a crédito.

Ahora, ¿cómo identificamos cuáles de estos sustantivos candidatos realmente describen clases en nuestro dominio de problema? Una propuesta muy utilizada, es chequear cada sustantivo candidato contra una serie de preguntas simples, como se muestra en la siguiente figura:

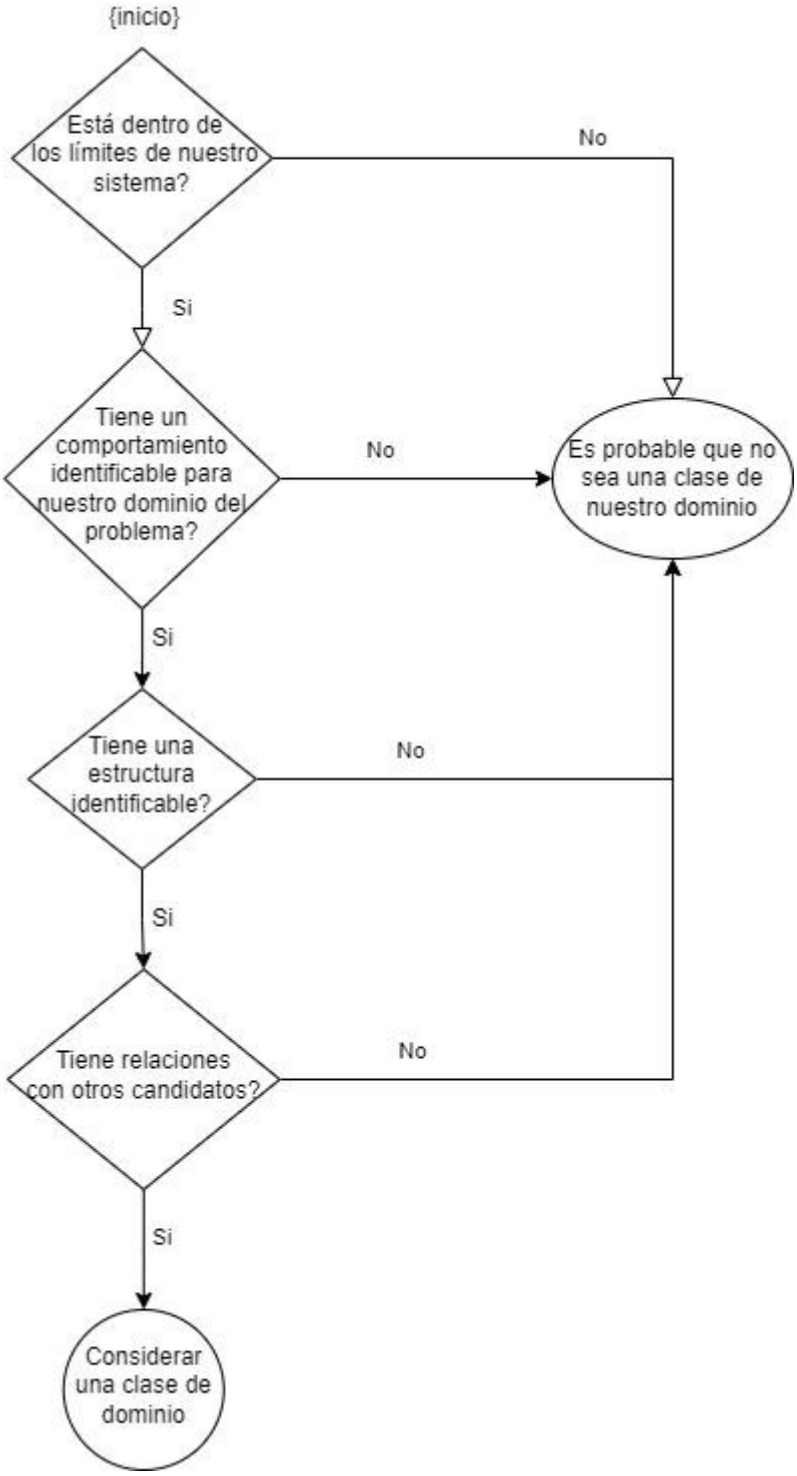

Figura (1) Elaboración propia

Consideraciones para la identificación de clases **significativas** del dominio:

- *"Es mejor especificar en exceso un modelo de dominio con muchas clases conceptuales de grano fino que especificar por defecto" (Larman, 2002).*
- El modelo no es mejor por tener pocas clases conceptuales.
- Es normal obviar clases conceptuales durante la identificación inicial para descubrirlas más tarde (incluso en diseño) al considerar atributos y asociaciones.
- Que los requisitos no indiquen la necesidad obvia de registrar información sobre una clase o porque la clase conceptual no tenga atributos no significa que debamos excluirla del modelado.

2. Modelado de clases del dominio

El modelo de dominio muestra las clases conceptuales o vocabulario del dominio, tal que informalmente una clase conceptual es una idea, cosa u objeto.

"Formalmente, una clase conceptual puede considerarse en términos de su símbolo, intensión y extensión" (Martin y Odell, 1995)

- Símbolo: palabras o imágenes que representan una clase conceptual.
- Intención: la definición de una clase conceptual.
- Extensión: el conjunto de ejemplos a los que se aplica la clase conceptual.

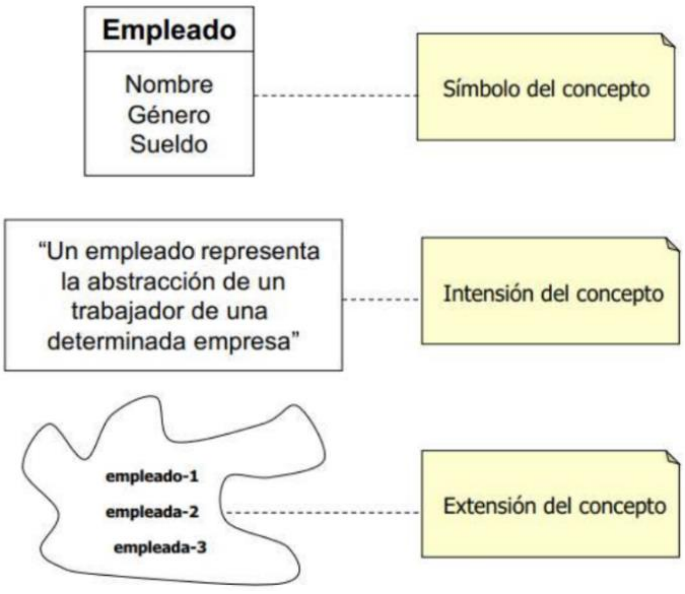

Figura (2) Extraída de internet

Modelado de clases abstractas

Es útil identificar las clases abstractas en el modelo del dominio porque esto restringe las clases que pueden tener instancias concretas y se clarifican las reglas del dominio del problema.

¿Cómo las identificamos? Si cada miembro de una clase A puede ser también miembro de una subclase, entonces A es una clase conceptual abstracta.

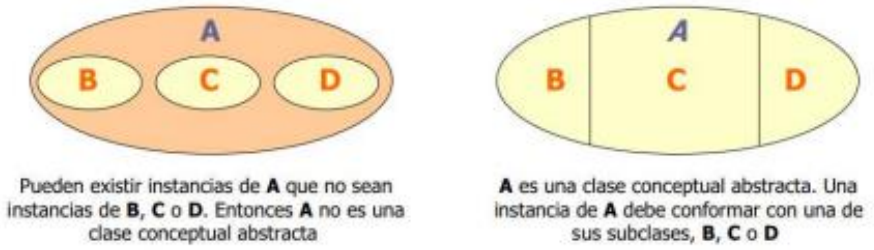

Figura (3) Extraída de internet

Modelado de los cambios de estado

En los diagramas de clases, los cambios de estado no se modelan de forma detallada ya que, como vimos en clase, estamos modelando un estadío estático del sistema en un momento determinado.

Para modelar No se debe modelar el estado de un concepto X como subclases de X, sino que se debe seguir una de estas dos estrategias:

- Definir una jerarquía de estados y asociar los estados con X.
- Ignorar la representación de los estados de un concepto en el modelo de dominio; en lugar de esto representar los estados en diagramas de estados.

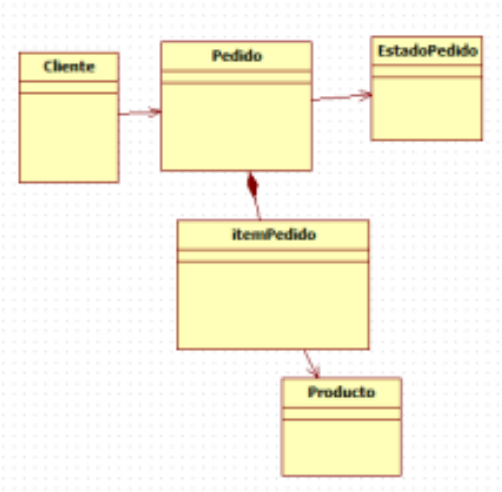

Figura (4) Elaboración propia

Modelando relaciones de generalización

No se debe utilizar la relación de generalización para modelar situaciones estructurales que son susceptibles de cambiar con el paso del tiempo ya que la relación de generalización es estática y es muy frecuente que los modelos tengan que reflejar situaciones estructurales que cambian con el paso del tiempo. Se debe valorar modelar mediante roles.

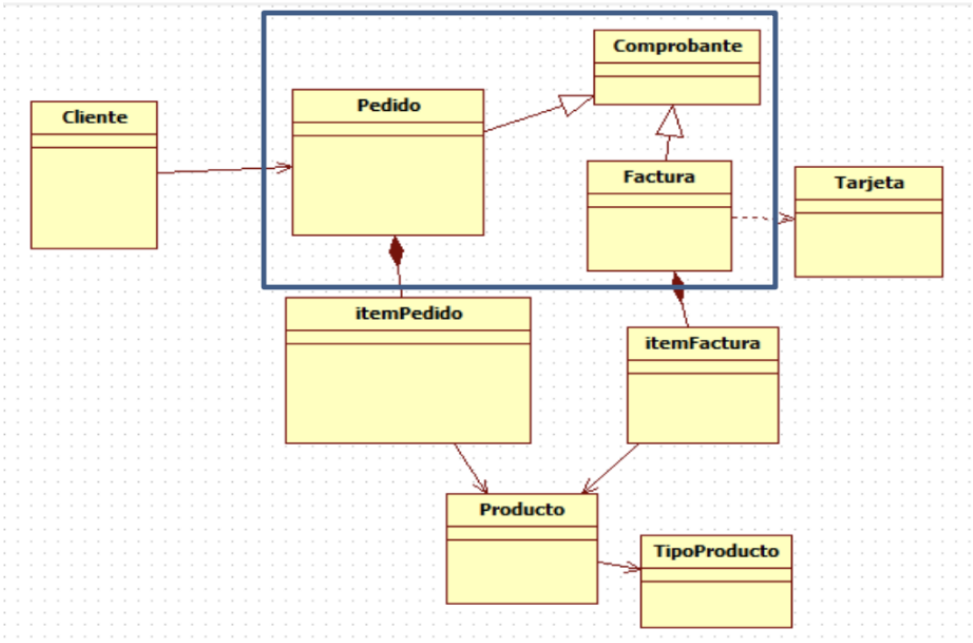

Figura (5) Elaboración propia

Modelado de relación TODO-PARTE

Una relación todo-parte puede utilizarse cuando el tiempo de vida de la parte está ligado al tiempo de vida del todo y, además:

- Existe una dependencia de creación-eliminación de la parte en el todo.
- Existe un ensamblaje obvio todo-parte físico o lógico.
- Alguna propiedad del compuesto se propaga a las partes, como la ubicación.
- Las operaciones que se aplican sobre el compuesto se propagan a las partes, como la destrucción, movimiento o grabación.

Existen dos tipos de relación: Agregación y Composición

Agregación: Un objeto agregado es un objeto construido a partir de otros objetos y el agregado es mayor que la suma de sus partes

- Todas las interacciones realizadas con el conjunto de los objetos agregados se realizan a través de la interfaz del objeto agregado.
- Los objetos componentes están ocultos o encapsulados dentro del agregado.

Composición: Es una forma fuerte de agregación en donde el ciclo de vida de las partes depende del ciclo de vida del agregado

- Las partes no existen fuera de su participación en el agregado.
- La pertenencia fuerte implica objetos físicos que se unen para formar el compuesto.

Ejemplos de cada una:

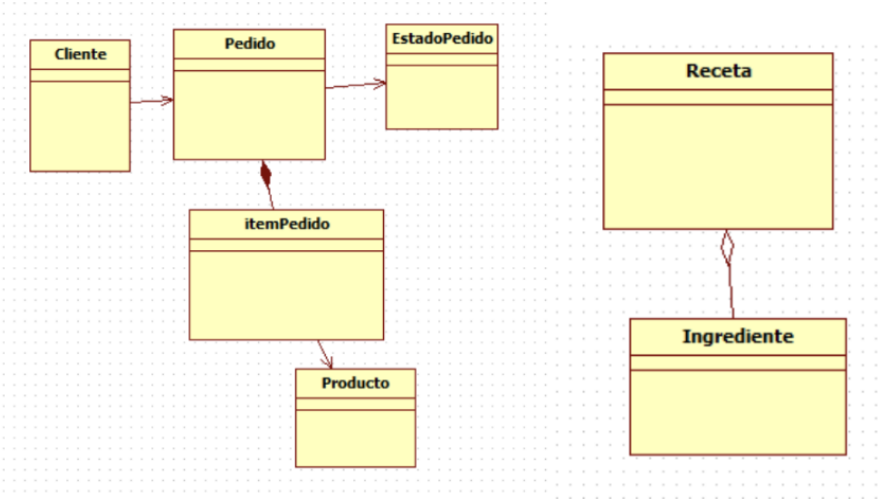

Figura (6) Elaboración propia

Agregación	Composición
La multiplicidad en el extremo de las partes es variable según el caso.	La multiplicidad en el extremo de las partes es variable según el caso.
La multiplicidad en el extremo del agregado es variable según el caso.	La multiplicidad en el extremo de la composición es 1.
Las partes pueden existir después de que el agregado sea “desmontado” o destruido.	Si la composición es “desmontada” o destruida, las partes no tienen existencia propia.
Las partes pueden cambiar de un agregado a otro.	Las partes no se pueden mover de una composición a otra.

Tabla (2) Elaboración propia

Es muy importante aclarar, que la relación de agregación y composición al ser muy similares, a veces esto está delimitado según el dominio del problema.

Continuando con el modelado, recordemos que el objetivo de esta práctica es crear un modelo de dominio de clases conceptuales **significativas del dominio de interés**, para ello:

A. Una vez listadas las clases conceptuales candidatas relacionadas con los requisitos actuales en estudio (primer paso de este proceso), representar las clases en un modelo de dominio.

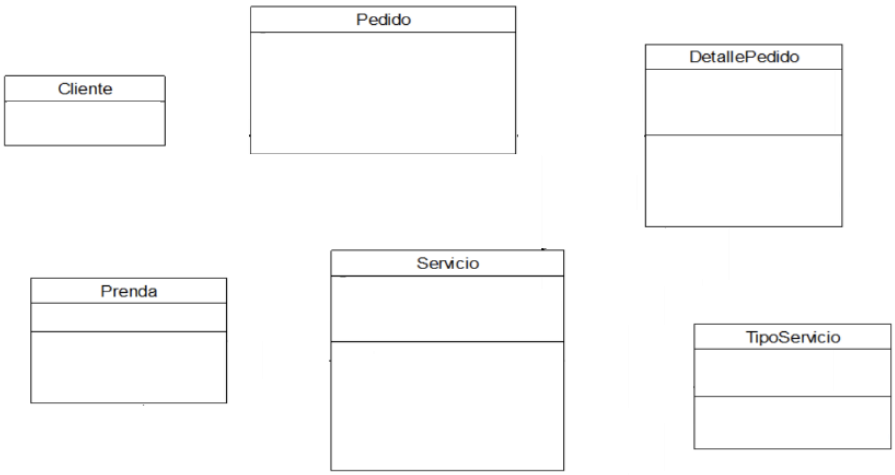

Figura (4) Elaboración propia

B. Añadir las asociaciones necesarias para registrar las relaciones que hay que mantener en memoria.

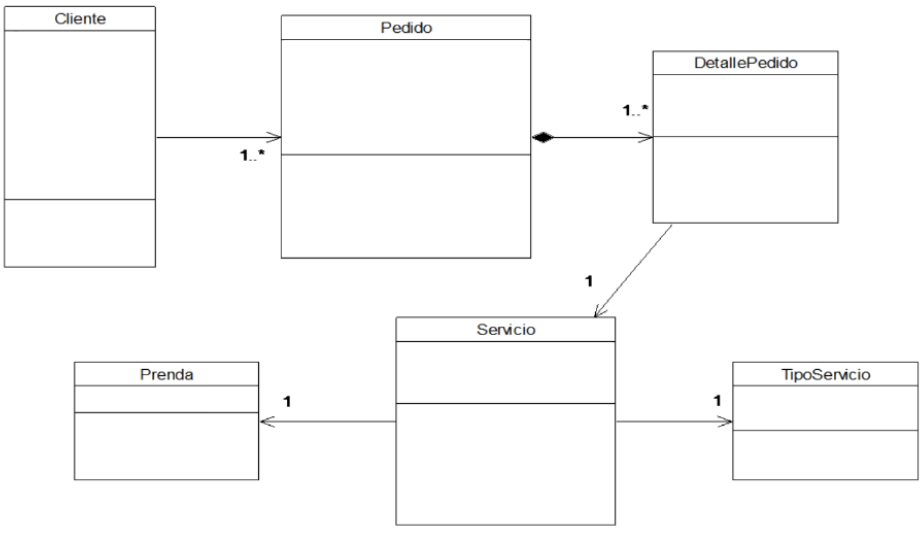

Figura (5) Elaboración propia

3. Identificar atributos

Son las propiedades relevantes de los objetos individuales. Una vez modeladas parcialmente las clases y sus relaciones, podemos completar el modelado de las clases mediante la identificación y especificación de sus atributos. Para ello, es importante tener en cuenta que:

- La mayoría de los atributos simples son los que conocen como tipos de datos primitivos
- El tipo de un atributo no debería ser un concepto de dominio complejo.
- Los atributos deben ser, generalmente, tipos de datos: *"Un tipo de dato para UML implica un conjunto de valores para los cuales no es significativa una identidad única (en el contexto del modelo o sistema en el que se está trabajando)" (Rumbaugh et al., 1999)*
- *"Se deberían incluir en un modelo de dominio aquellos atributos para los que los requisitos sugieren o implican una necesidad de registrar la información" (Larman, 2002)*
- En caso de duda es mejor definir algo como una clase conceptual en lugar de como un atributo y se debe representar lo que podría considerarse inicialmente como un tipo de dato, como una clase no primitiva si:
  - Está compuesta de secciones separadas.
  - Habitualmente hay operaciones asociadas con él, como análisis sintáctico o validación.
  - Tiene otros atributos.
  - Es una cantidad con una unidad.
  - Es una abstracción de uno o más tipos con alguna de estas cualidades.

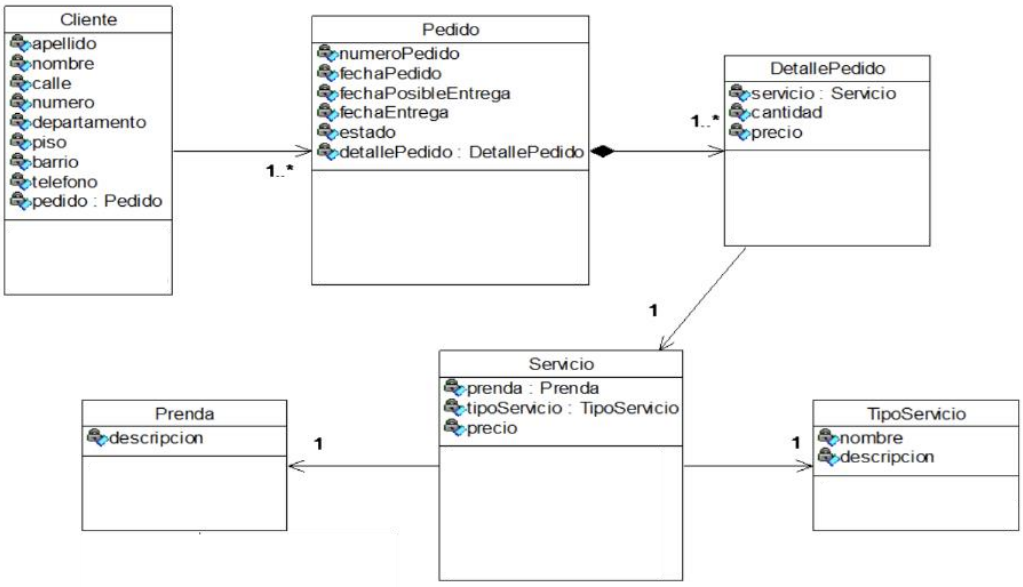

Figura (6) Elaboración propia

4. Identificar métodos/operaciones

Las operaciones son esenciales para definir el comportamiento de las clases y cómo interactúan entre sí. A continuación, se describen los pasos y consideraciones para llevar a cabo esta identificación a partir de la especificación de requerimientos:

1. Identificación de Verbos Clave: Busca verbos en los requerimientos que indiquen acciones. Los verbos suelen estar relacionados con las operaciones que deben realizar las clases. Por ejemplo, en un requerimiento que dice "el cliente debe poder realizar un pedido", el verbo "realizar" indica que la clase relacionada (como Pedido) tendrá un método llamado realizarPedido().
2. Definición de Responsabilidades: Para cada clase identificada en el modelo de dominio, determina su responsabilidad. Pregunta: "¿Qué debe hacer esta clase?" o "¿Qué acciones debe poder ejecutar?" Esto te ayudará a definir métodos que reflejen estas responsabilidades.
3. Consideración de las Relaciones entre Clases: Examina cómo las clases están relacionadas entre sí. Las relaciones como asociación, agregación y composición pueden sugerir operaciones. Por ejemplo, si una clase Cliente está asociada con una clase Pedido, podría ser útil incluir métodos en Cliente como crearPedido() o verHistorialDePedidos().

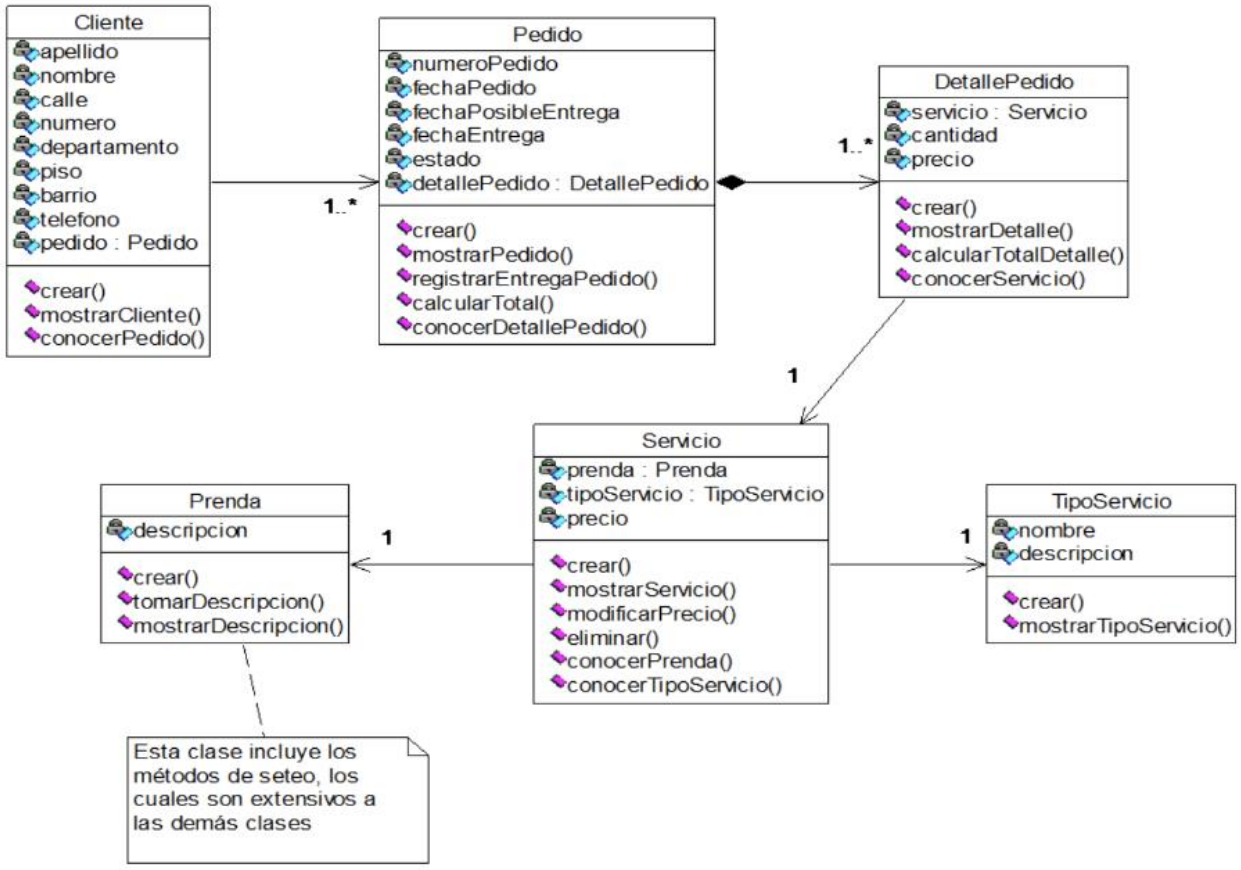

Figura (7) Elaboración propia

Teniendo en cuenta todos estos aspectos, como conclusión podemos llegar a diferentes Diagramas de Clases, ya que dependiendo como se modele es el resultado obtenido.

El objetivo de esta materia no es abordar en estos aspectos, sino en el entendimiento y concepción de un diagrama.

Siguiendo con el ejemplo, otra manera de representar el diagrama de clases es la siguiente.

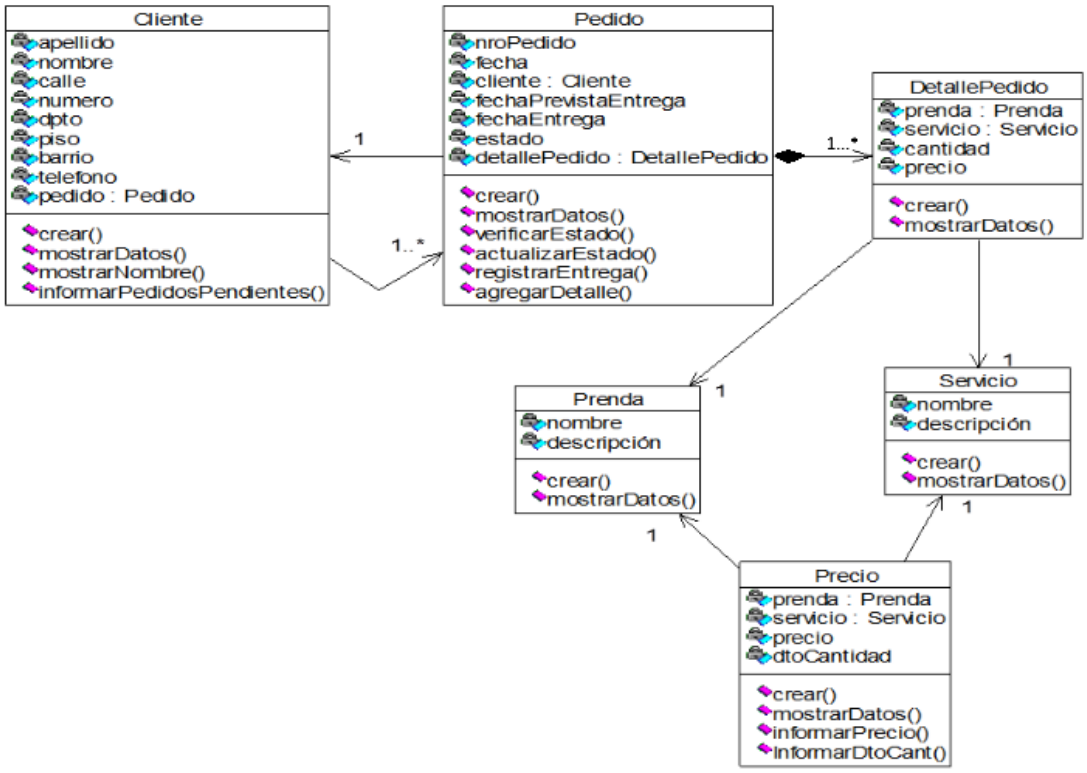

Figura (8) Elaboración propia

Problema 2: “Comida Argentina”

La casa de comidas rápidas “Comida Argentina” se dedica a la elaboración y comercialización de diversos menús.

Entre los productos que se preparan se encuentran sándwich de hamburguesas, lomitos, papas, ensaladas. El proceso de producción y venta de menú es el siguiente:

Cuando el cliente se acerca al mostrador es atendido por el Empleado de Ventas quien le consulta el menú que desea. El cliente selecciona el menú e indica la cantidad deseada –el comprador puede incluir más de un tipo de menú en un pedido -. El Empleado de Ventas genera entonces el ticket para el pedido, con lo cual se emite una copia del mismo en la cocina del local, en donde el cocinero preparará cada menú para el ticket en cuestión. El Empleado de Ventas le cobra al cliente y le informa que espere su pedido en el mostrador de entregas.

A modo de ejemplo, se detalla cómo se conforma cada menú. Menú 1: hamburguesa con queso, lechuga y huevo; papas y gaseosa. Menú 2: hamburguesa de pollo con queso y lechuga, papas y gaseosa. Menú 3: ensalada cesar y agua,

entre otros. Cada menú se conforma de una hamburguesa, lomito o ensalada; una porción de papas y una bebida.

Cuando el menú está listo el cocinero se lo entrega al vendedor y éste llama al cliente por su número de ticket para que retire la bandeja del mostrador.

Se solicita:

Construir el Diagrama de Clases abreviado, teniendo en cuenta:

1. Identificar clases y graficar.
2. Identificar relaciones entre clases y representar gráficamente

Problema 3: "CAR WASH" (Resuelto)

La empresa **CAR WASH** se dedica al lavado de vehículos y ha decidido adquirir un Sistema de Información cuyos alcances son los siguientes:

- Administración de máquinas de lavado.
- Administración de tipos de servicio de lavado
- Administración de Clientes.
- Gestión de turnos.
- Gestión de Cobro de Servicios de lavado.
- Generación de reportes vinculados con los servicios que presta y los ingresos asociados.

Un cliente puede requerir:

- Lavar su vehículo en ese momento con turno previo.
- Lavar su vehículo en ese momento sin turno previo. Esto es posible si hay máquinas disponibles, teniendo prioridad las reservas existentes.
- Pedir un turno para lavar un vehículo en otro momento (día y hora)
- Cancelar un turno ya pedido.

Las modificaciones se tratan como cancelaciones y se crea una nueva reserva. Las tarifas de los servicios se determinan con una vigencia y varían dependiendo del tipo de lavado (completo, solo carrocería, solo motor, etc.), y del tipo de vehículo (auto, moto o camioneta). También influyen las promociones que son establecidas por la Administración para los clientes, a los que se les asigna un tipo

de cliente. Cada cliente tiene asociado un único tipo de cliente, por ejemplo: remiseros, taxis, clientes frecuentes, etc.

Cuando concluye el lavado se cobra y se entrega el vehículo acompañado con su comprobante de cobro. Queda excluido del alcance la facturación.

La Gerencia ha manifestado la necesidad de tener entre otros informes:

- Informe que incluya la cantidad de lavados por tipo de lavado y por tipo de vehículo.
- Importes cobrados por tipo de lavado en un período de tiempo.
- Tipos de Cliente que utilizan los servicios con más frecuencia.

Se pide:

1) Construir el **Modelo de Objetos del Dominio** indicando atributos y métodos para las clases y especificando navegabilidad y multiplicidad para las relaciones.

Solución:

La empresa CAR WASH se dedica al lavado de vehículos y ha decidido adquirir un Sistema de Información cuya definición es la siguiente:

1. Objetivo del SI: Gestionar la reserva de turnos y el lavado de vehículos, administrando clientes y sus vehículos y brindando información resultante de la gestión.
2. Alcances
   - Administración de máquinas de lavado.
   - Administración de tipos de lavado.
   - Administración de Clientes y sus vehículos.
   - Gestión de reserva de turnos.
   - Gestión de tarifas y cobro por servicios de lavado.
   - Generación de reportes vinculados con los servicios que presta y los ingresos asociados.
3. No contempla
   - Facturación y emisión de comprobante de cobro.
   - Administración de usuarios y perfile

Mapa Conceptual para análisis de dominio del problema

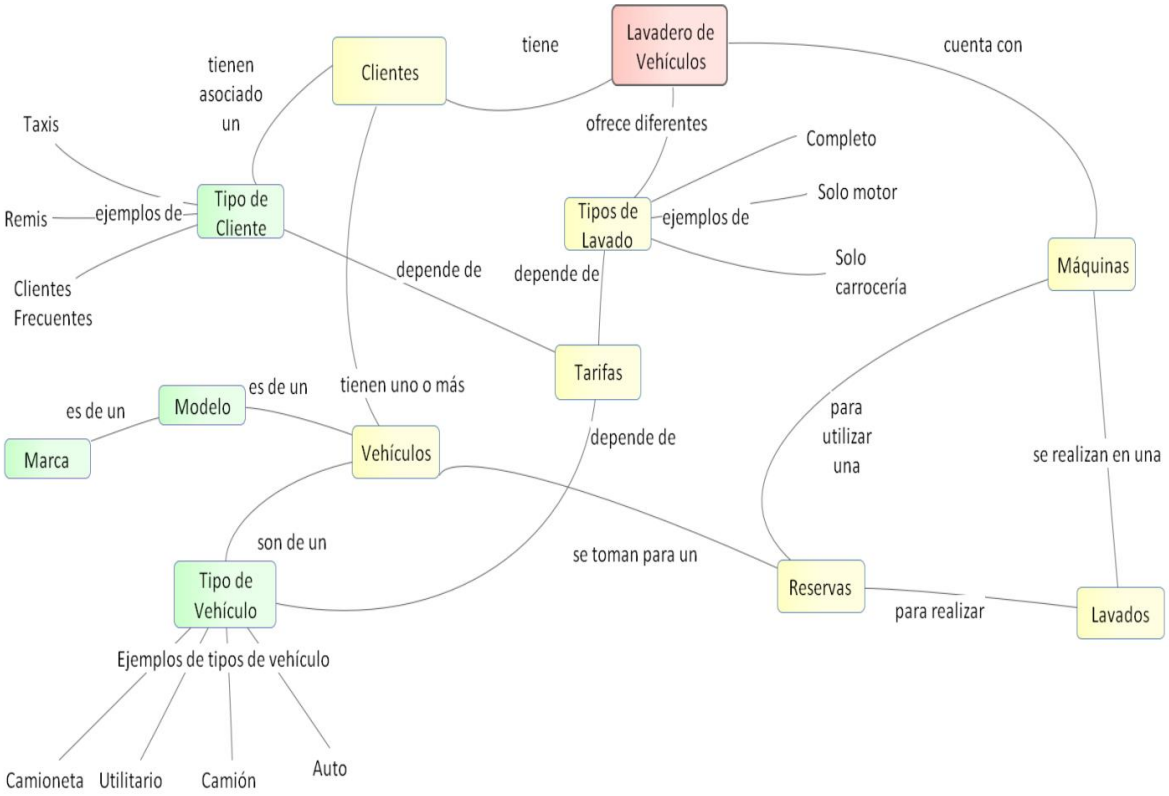

Figura (9) Elaboración propia

Solución Propuesta: Modelo de Dominio

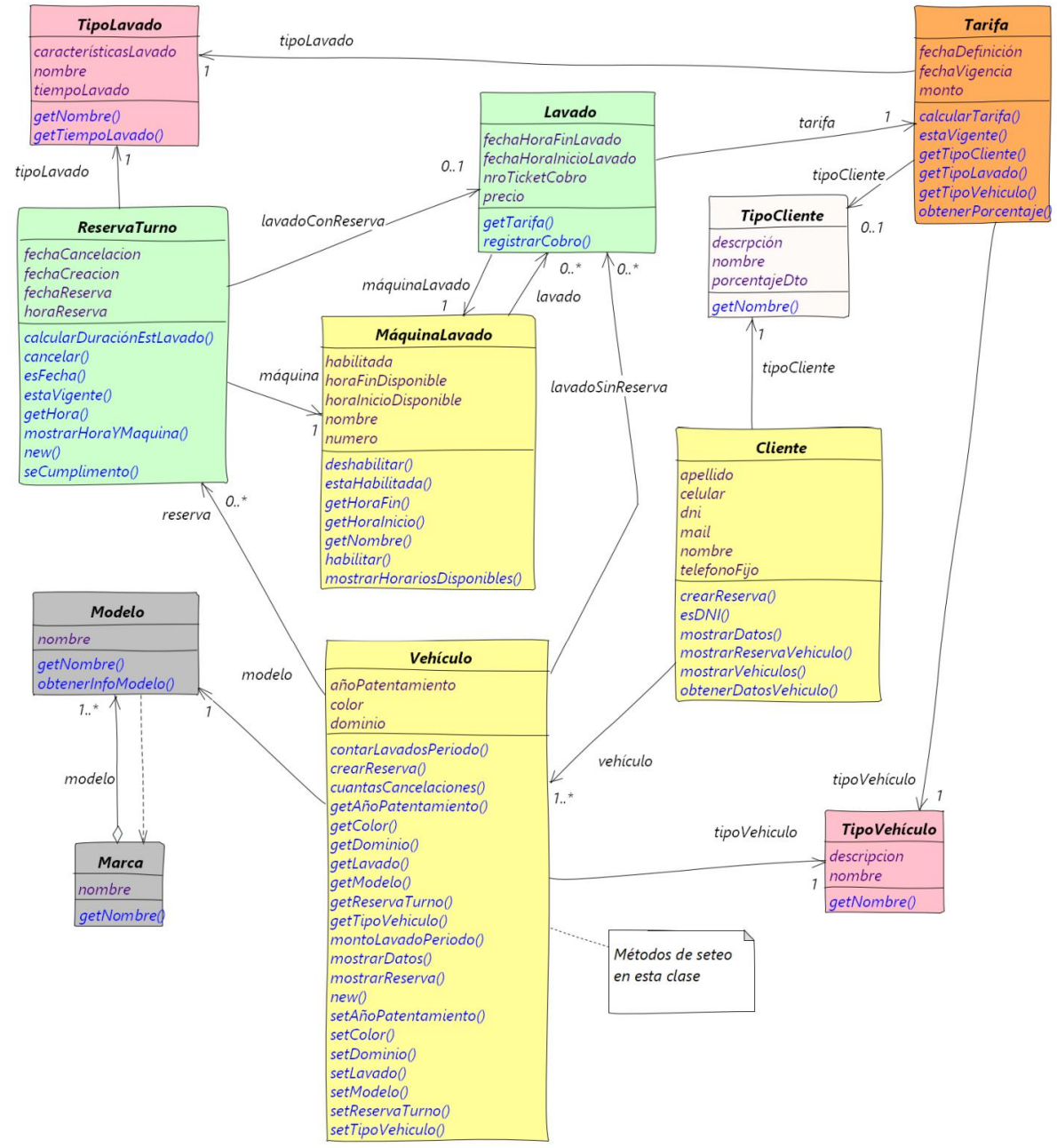

Figura (10) Elaboración propia

Modelo de máquina de estado

Los diagramas de máquina de estado son una herramienta visual utilizada en ingeniería de software para modelar el comportamiento dinámico de un sistema o de una clase específica. Representan los diferentes estados por los que puede pasar un objeto a lo largo de su ciclo de vida, así como las transiciones que ocurren entre estos estados en respuesta a eventos. Su objetivo principal es mostrar cómo un objeto cambia de estado en función de las interacciones y eventos que se producen en el sistema, ayudando a entender mejor su comportamiento en situaciones complejas.

A continuación, vamos a trabajar en un ejemplo para comprender cómo confeccionar estos diagramas, paso a paso:

Problema 4: “E-Commerce”

En un sistema de gestión de pedidos de un e-commerce, la clase Pedido representa el ciclo de vida de una orden realizada por un cliente, la cual pasa por varios estados a medida que avanza. Inicialmente, un Pedido se encuentra en estado Pendiente, lo que indica que ha sido creado, pero aún no se ha procesado. Este estado cambia a Procesado una vez que el cliente confirma el pago. Luego, el pedido pasa al estado Enviado cuando es preparado y despachado. Si el cliente recibe el pedido, el estado cambia a Entregado, lo que marca la finalización exitosa del ciclo de vida del pedido. Sin embargo, si en cualquier momento antes del despacho el pedido es cancelado, se activa la transición hacia el estado Cancelado, cerrando también su ciclo de vida. Este seguimiento detallado de los estados y eventos permite un control preciso del proceso de cada pedido.

1. Revisar el Diagrama de Clases

En el sistema de gestión de pedidos, la clase **Pedido** es un buen candidato para un diagrama de máquina de estado, ya que su comportamiento cambia dinámicamente a lo largo de su ciclo de vida. Desde su creación hasta su finalización, un **Pedido** pasa por varios estados, lo que debe modelarse en el diagrama.

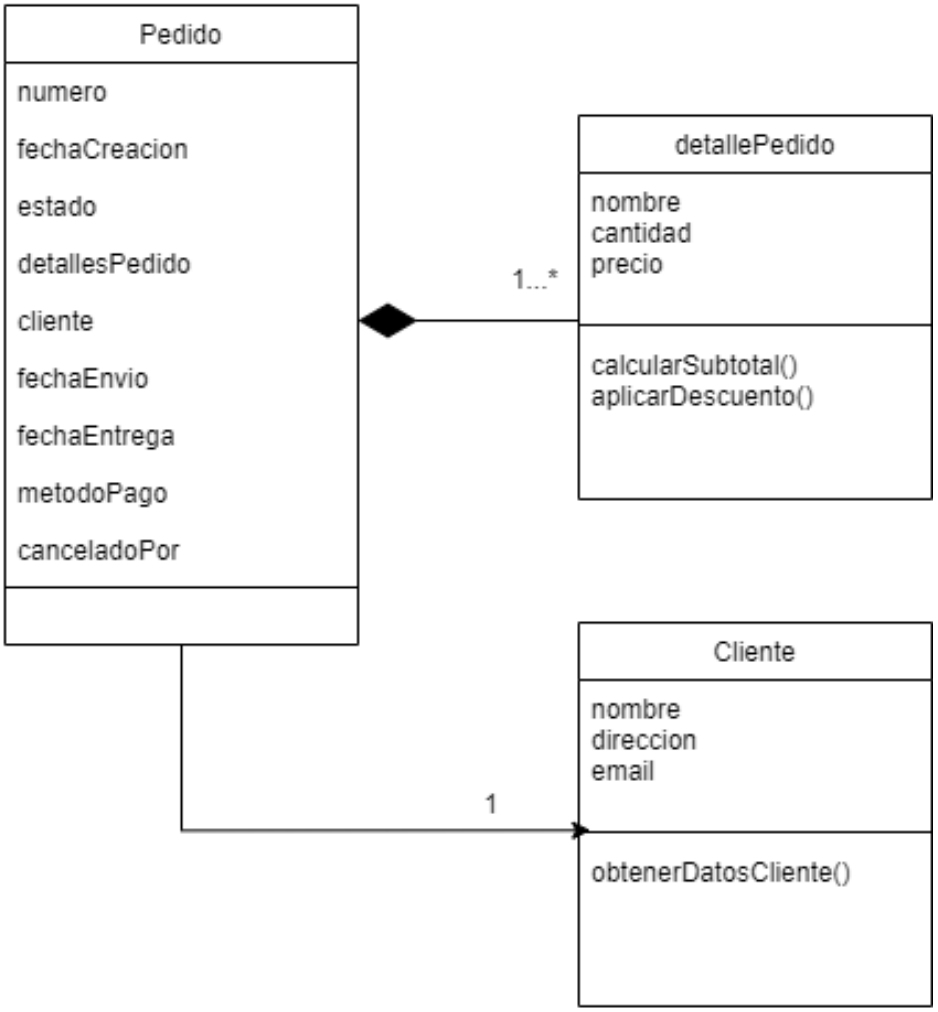

Figura (11) Elaboración propia

2. Identificar los Estados Importantes

De acuerdo con el enunciado, los estados por los que pasa un Pedido son:

- Pendiente: El pedido ha sido creado, pero no ha sido procesado aún.
- Procesado: El pedido ha sido pagado y está listo para ser enviado.
- Enviado: El pedido ha sido despachado y está en camino al cliente.
- Entregado: El pedido ha sido recibido por el cliente, completando exitosamente su ciclo de vida.
- Cancelado: El pedido ha sido cancelado antes de su despacho.

3. Determinar los Eventos y Transiciones

Cada cambio de estado en el ciclo de vida del Pedido es provocado por un evento:

- De Pendiente a Procesado: El evento "Pago confirmado" cambia el estado a Procesado cuando el cliente paga.
- De Procesado a Enviado: El evento "Pedido despachado" ocurre cuando el pedido está listo para su envío.
- De Enviado a Entregado: El evento "Entrega confirmada" marca que el cliente ha recibido el pedido, cambiando el estado a Entregado.
- A Cancelado: El evento "Cancelación solicitada" puede ocurrir en cualquier momento antes de que el pedido sea enviado, finalizando el pedido en el estado Cancelado.

4. Crear el Diagrama de Máquina de Estado

- **Estado inicial**: El diagrama comienza con un círculo sólido que representa el estado inicial (*Pendiente*) cuando el pedido es creado.
- **Estados intermedios y finales**: Representa cada estado relevante con un rectángulo redondeado, como *Procesado*, *Enviado*, *Entregado*, y *Cancelado*.
- **Transiciones**: Conecta los estados usando flechas para mostrar cómo y cuándo se producen las transiciones. Etiqueta cada flecha con el evento correspondiente (por ejemplo, "Pago confirmado" entre *Pendiente* y *Procesado*).
- **Estado final**: Representa el estado final con un círculo doble, indicando que el ciclo de vida del **Pedido** puede terminar en *Entregado* o *Cancelado*.

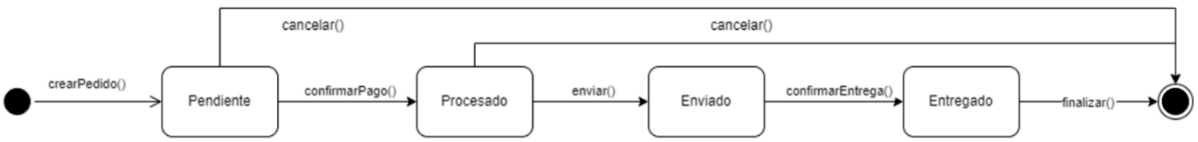

Figura (12) Elaboración propia

5. Refinar el Diagrama con Condiciones Guardas (Opcional)

Si es necesario, puedes agregar *guardas* que especifiquen condiciones bajo las cuales una transición ocurre. Por ejemplo, la transición de *Procesado* a *Enviado* podría tener una guarda que diga "[Pedido listo para despachar]".

6. Revisar y Validar

Revisa el diagrama para asegurarte de que los estados y transiciones cubren todos los posibles casos en el ciclo de vida de un pedido. También, verifica que los eventos y condiciones sean coherentes con la lógica del sistema.

7. Completar el diagrama de clases

Una vez finalizado el diagrama, es importante validar que los métodos de la clase a modelar sean coherentes con los eventos y transiciones definidas en el diagrama de máquinas de estado. Se deben definir como métodos de la clase a todos los eventos que permitirían cambiar el estado de la clase según las transiciones del diagrama.

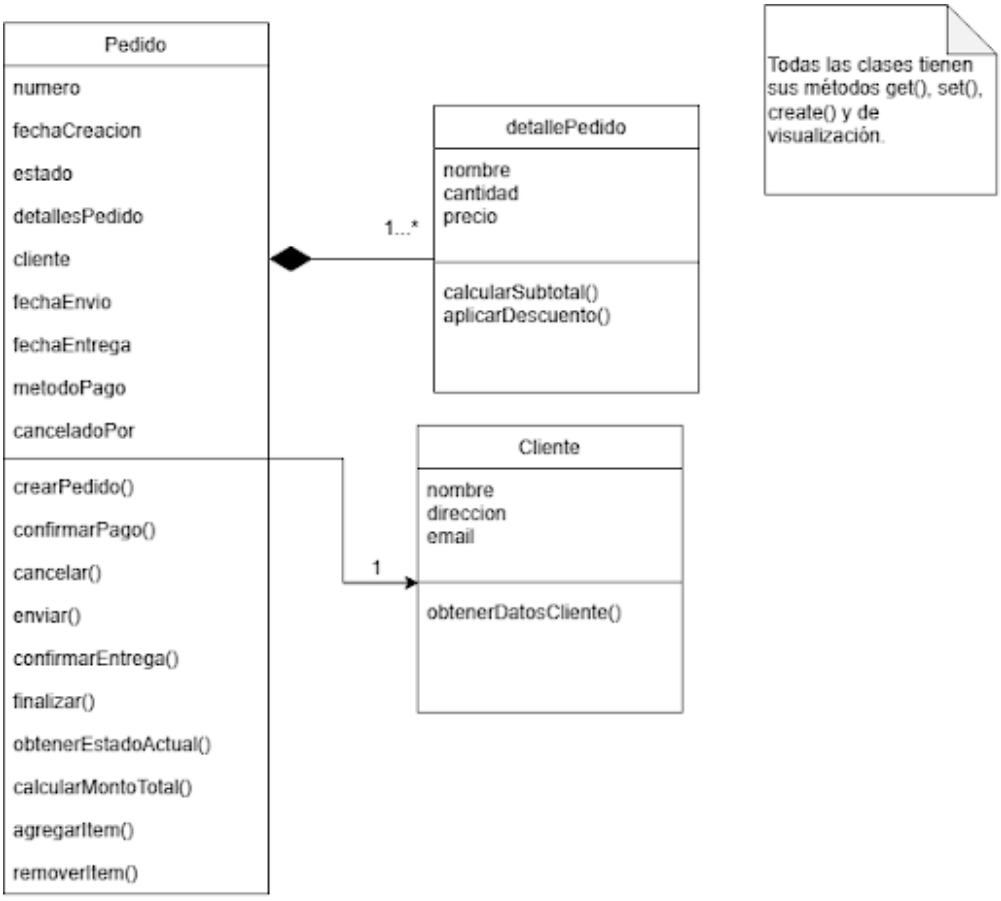

Figura (13) Elaboración propia

Problema 5: “Fórmula Uno”

Un cliente nos solicitó una aplicación para llevar el control del Campeonato de la Fórmula Uno. Para ello, pudimos recabar la siguiente información. Los campeonatos se realizan de manera anual. En este campeonato, varios equipos, llamados escuderías, compiten en una serie de carreras que se llevan a cabo en diversos circuitos de todo el mundo. Cada equipo está compuesto por dos pilotos, en donde cada uno de ellos conduce un automóvil de Fórmula 1.

Las carreras se llevan a cabo en un Gran Premio (GP), que es un evento que tiene características únicas, incluyendo el nombre del circuito en el que se lleva a cabo, la ubicación geográfica y otras características distintivas. Los Grandes Premios que se van a disputar en el campeonato se definen al comienzo del año junto con una fecha determinada. Cada Gran Premio se realiza en un fin de semana e incluye distintas etapas: las prácticas, la clasificación y la carrera, que se llevan a cabo en un día y horario específico.

Durante cada carrera, los pilotos compiten por acumular puntos, que se otorgan a los que terminan en las primeras posiciones. El ganador de la carrera recibe la mayor cantidad de puntos, y luego los puntos disminuyen a medida que se baja en la clasificación. Estos puntos se acumulan a lo largo de la temporada para determinar al campeón.

Antes de cada carrera, los pilotos participan en una sesión de calificación que determina el orden de largada en la parrilla de salida. La carrera en sí implica una cantidad de vueltas al circuito, con paradas en boxes para cambiar neumáticos y hacer ajustes en el coche. El objetivo de cada piloto es completar la distancia total de la carrera lo más rápido posible.

Debido a condiciones externas (por ejemplo, el clima, desastres naturales, etc.), en muchas ocasiones las carreras se ven afectadas, ya sea antes del inicio o durante las mismas. Esto lleva a que la carrera sea suspendida. Si la situación es temporal y las condiciones mejoran, se procede a dar inicio en el punto en el que se había suspendido. En caso contrario, si no es posible continuar, entonces se comunica que la carrera fue cancelada. Cabe aclarar que, si el calendario lo permite, una carrera puede ser reprogramada para una nueva fecha.

Se solicita:

1. Diseñe el diagrama de clases asociado al problema
2. Realice la máquina de estados asociada a la clase **Carrera**

Solución Problema 5: “Fórmula Uno”

Diagrama de Clases del Campeonato de Fórmula Uno (con clase abstracta Etapa)

Este documento detalla el diseño del modelo orientado a objetos del sistema para gestionar el Campeonato de Fórmula Uno, incluyendo el uso de una clase abstracta Etapa, de la cual heredan las clases Carrera, Clasificación y Práctica.

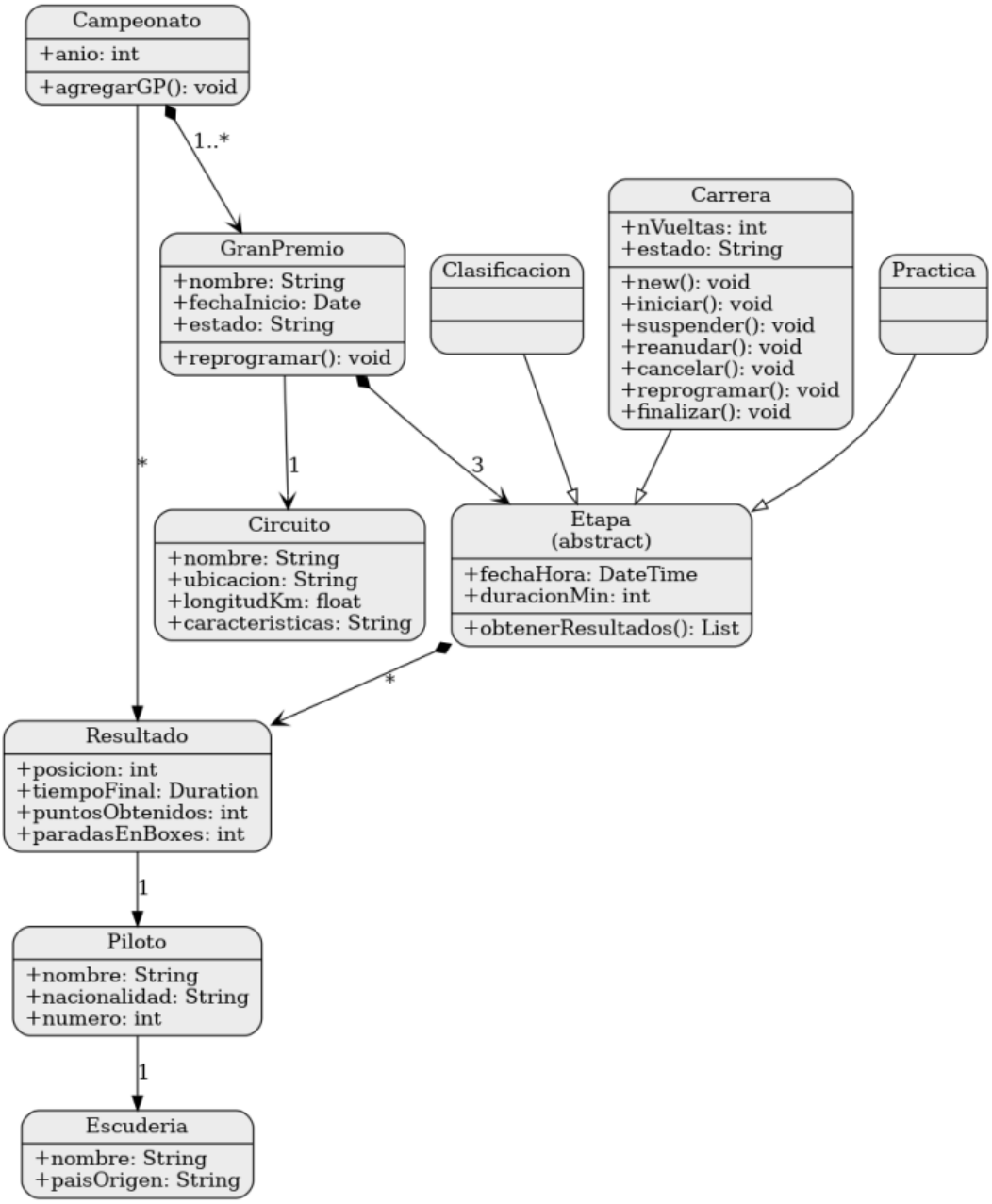

Figura (14) Elaboración propia diagrama con etapas

- **Importante**: Faltaría agregar una nota que todas las clases contienen los métodos **set()**, **get()**, **create()** y **conoce_a()**

Justificación de Clases y Relaciones

1. Campeonato

Del enunciado: "Los campeonatos se realizan de manera anual."

- anio: int: representa el año del campeonato.
- agregarGP(): para añadir Grandes Premios.

Relaciones:

- Composición con GranPremio (1..*).
- Asociación con Resultado (para acumular puntos).

2. GranPremio

Del enunciado: "Las carreras se llevan a cabo en un Gran Premio (GP)..."

- Atributos: nombre, fechaInicio, estado.
- Método: reprogramar().

Relaciones:

- Composición con Etapa (3 instancias: una de cada subclase).
- Asociación con Circuito.

3. Etapa (abstracta)

Del enunciado: "Cada Gran Premio se realiza en un fin de semana e incluye distintas etapas: las prácticas, la clasificación y la carrera..."

Clase abstracta: Etapa

- Atributos comunes: fechaHora, duracionMin.
- Método: obtenerResultados()

Subclases:

a. **Carrera**

- Atributos: nVueltas, estado.
- Métodos: new(), iniciar(), suspender(), reanudar(), cancelar(), reprogramar(), finalizar().
- **Del enunciado:** menciona suspensiones, cancelaciones, reanudaciones, finalización y reprogramación.

b. **Clasificación**

- Puede tener métodos como determinarOrdenLargada().
- **Del enunciado:** "...determina el orden de largada en la parrilla de salida."

c. Practica

- Atributos sugeridos: nroSesion, clima, mejorTiempo.
- No detallados en el enunciado, pero comunes en la práctica real.

Relaciones:

- Composición entre Etapa y Resultado.

4. Resultado

Del enunciado: "...los pilotos compiten por acumular puntos..."

- Atributos: posicion, tiempoFinal, puntosObtenidos, paradasEnBoxes.
- Asociado a un Piloto, compuesto por una Etapa.

5. Piloto

- Atributos: nombre, nacionalidad, numero.
- Asociación con Escudería.

6. Escudería

- Atributos: nombre, paisOrigen.
- Representa el equipo de Fórmula Uno.

7. Circuito

Del enunciado: "...nombre del circuito, ubicación geográfica y características distintivas."

- Atributos: nombre, ubicacion, longitudKm, caracteristicas.
- Asociado 1:1 con GranPremio.

Este enfoque permite aprovechar herencia y polimorfismo, al factorizar comportamientos comunes en Etapa y delegar lo específico en las subclases.

Diagrama de Clases del Campeonato de Fórmula Uno (sin clase Etapa)

Este documento describe el diseño del sistema de información para gestionar el Campeonato de Fórmula Uno sin emplear una clase abstracta para representar las etapas. En este enfoque, las clases Carrera, Clasificación y Práctica se modelan como entidades independientes.

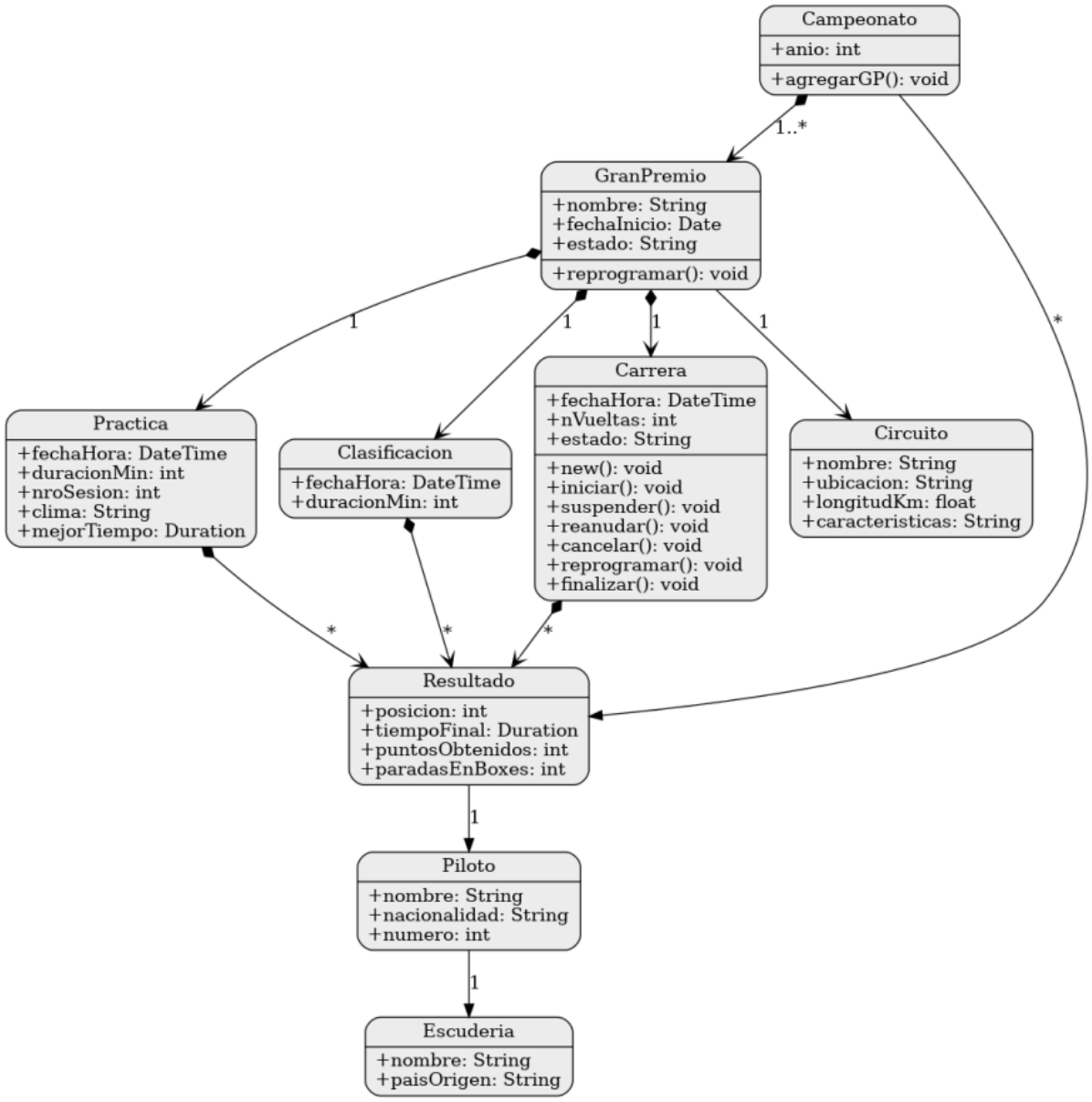

Figura (15) Elaboración propia diagrama sin etapas

- **Importante**: Faltaría agregar una nota que todas las clases contienen los métodos **set()**, **get()**, **create()** y **conoce_a()**

Justificación de Clases y Relaciones

1. Campeonato

Del enunciado: "Los campeonatos se realizan de manera anual."

- Atributo: anio
- Método: agregarGP()

Relaciones:

- Composición con GranPremio (1..*).
- Asociación con Resultado para acumulación de puntos.

2. GranPremio

Del enunciado: "Las carreras se llevan a cabo en un Gran Premio (GP)..."

- Atributos: nombre, fechaInicio, estado
- Método: reprogramar()

Relaciones:

- Composición con Carrera, Clasificación y Práctica (cada una como clase separada)
- Asociación con Circuito

3. Carrera

Del enunciado: "Durante cada carrera, los pilotos compiten por acumular puntos..."

- Atributos: fechaHora, nVueltas, estado
- Métodos: new(), iniciar(), suspender(), reanudar(), cancelar(), reprogramar(), finalizar()
- Composición con Resultado

4. Clasificación

Del enunciado: "...los pilotos participan en una sesión de calificación que determina el orden de largada..."

- Atributos: fechaHora, duracionMin
- Puede incluir método determinarOrdenLargada() (implícito)
- Composición con Resultado

5. Práctica

Del enunciado: "...las prácticas [...] se llevan a cabo en un día y horario específico."

- Atributos: fechaHora, duracionMin, nroSesion, clima, mejorTiempo
- Composición con Resultado

6. Resultado

- Atributos: posicion, tiempoFinal, puntosObtenidos, paradasEnBoxes
- Asociado a Piloto, compuesto por una etapa (Carrera, Clasificación o Práctica)

7. Piloto

Del enunciado: "Cada equipo está compuesto por dos pilotos..."

- Atributos: nombre, nacionalidad, numero
- Asociación con Escudería

8. Escudería

- Atributos: nombre, paisOrigen

9. Circuito

Del enunciado: "...nombre del circuito, ubicación geográfica y características distintivas."

- Atributos: nombre, ubicacion, longitudKm, caracteristicas
- Asociación con GranPremio (1:1)

Este enfoque mantiene un modelo simple y directo, ideal cuando las etapas tienen diferencias significativas o si no se requiere polimorfismo.

Diagrama de Estados de CARRERAS del Campeonato de Fórmula Uno

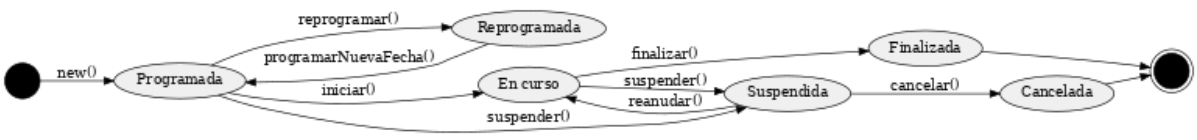

Figura (16) Elaboración propia