# **CASOS DE USO**

***Actividades***

1. *Tareas:* Las presentes tareas deben realizarse en orden -o no- prestando atención a la relación entre las lecturas realizadas.

1. Leer el Prefacio de UML Gota a Gota
2. Leer la sección “Cómo llegamos hasta aquí” de UML Gota a Gota y crea un documento que expanda lo escrito hasta nuestros días.
3. Aprende UML utilizando Visual Paradigm <https://www.visual-paradigm.com/guide/> y crea un resumen conciso de los conceptos más relevantes trabajados en clase. Explora los tutoriales y guías disponibles en el sitio web, enfocándote en los temas que se relacionan con los conceptos discutidos en clase. A medida que avances, toma notas y organiza un resumen estructurado de los conceptos clave de UML. Finalmente, comparte y discute tus hallazgos con tus compañeros de clase para mejorar la comprensión de UML y su aplicación en proyectos futuros.

2. Considere el ejercicio 2 del teórico 2.1:

*Se pide:*

c) Identifique los casos de uso del sistema y elabore un diagrama de casos de uso.

d) Escriba el escenario principal correspondiente a cada uno de ellos.

e) Escriba el desarrollo expandido para cada caso de uso del sistema (o sea, para cada uno de ellos, escriba una ficha que detalle actores, puntos de extensión y de inclusión, precondiciones, poscondiciones, flujo principal y flujos alternativos).

**🔹 Punto a) Prefacio de *UML Gota a Gota***

En el **prefacio** se explica que UML (*Unified Modeling Language*) nació como un esfuerzo de estandarización en los años 90, cuando existían muchos lenguajes y notaciones distintas para modelar software (Booch, OMT, OOSE, etc.). UML se planteó como un **lenguaje visual unificado** que sirviera a:

* Analistas, diseñadores y desarrolladores para comunicarse con un mismo “idioma gráfico”.
* Representar de manera clara sistemas complejos.
* Brindar diagramas que abarcan distintos puntos de vista (estructural, de comportamiento, de interacción).

El texto destaca que UML no es un **método de desarrollo**, sino un **lenguaje de modelado** que puede usarse con diferentes metodologías (ágiles, tradicionales, etc.).

En resumen: **El prefacio justifica la necesidad de UML como un estándar universal de modelado que facilite comunicación y documentación en proyectos de software.**

**🔹 Punto b) Sección *“Cómo llegamos hasta aquí”* y expandir hasta nuestros días**

En esa sección se relata la **evolución histórica**:

* Años 80–90: coexistían varios métodos de modelado orientado a objetos (Booch, OMT de Rumbaugh, OOSE de Jacobson).
* Mediados de los 90: Grady Booch, James Rumbaugh e Ivar Jacobson (“los tres amigos”) unifican sus notaciones bajo UML.
* 1997: UML es adoptado como estándar por la OMG (Object Management Group).

**Expansión hasta la actualidad (2025):**

* UML ha evolucionado en varias versiones: UML 1.x (1997–2003), UML 2.x (2005–actual).
* Hoy sigue siendo estándar de la OMG, con 14 tipos de diagramas (estructurales y de comportamiento).
* UML se combina con otros lenguajes/modelos:
  + **SysML** (Systems Modeling Language) para ingeniería de sistemas.
  + **BPMN** (Business Process Model and Notation) para procesos de negocio.
  + **ArchiMate** para arquitectura empresarial.
* En metodologías ágiles, UML se usa de manera **ligera**, sobre todo con **diagramas de casos de uso, de clases y de secuencia**, evitando la sobrecarga documental.
* Herramientas modernas (Visual Paradigm, Enterprise Architect, StarUML, PlantUML) permiten integrar UML con pipelines DevOps, generar código automáticamente o enlazar con historias de usuario.
* Aunque han surgido enfoques más ágiles y ligeros (como C4 Model o diagramas ad-hoc), UML sigue siendo una **referencia universal** en proyectos que requieren modelado formal o documentación estándar.

En resumen: **“Cómo llegamos hasta aquí” explica la unificación histórica de lenguajes; la expansión muestra que UML sigue vigente, pero adaptado a entornos ágiles y coexistiendo con nuevos lenguajes de modelado y notaciones modernas.**

## 🔹 Punto c) Resumen de UML a partir de Visual Paradigm

## 1. Introducción

El **Lenguaje Unificado de Modelado (UML, Unified Modeling Language)** es un estándar creado para representar de forma gráfica los sistemas de software y su comportamiento. No es una metodología de desarrollo, sino un **lenguaje visual** que facilita la comunicación entre analistas, diseñadores, programadores y clientes.

Su objetivo principal es **unificar** las distintas notaciones de modelado que existían en los años 90 y ofrecer un marco común para describir sistemas desde diferentes perspectivas.

## 2. Conceptos principales aprendidos en clase y reforzados con Visual Paradigm

### **Actores**

* Representan **roles externos** (usuarios, otros sistemas) que interactúan con el sistema.
* No son personas concretas, sino funciones o roles.

### **Casos de uso**

* Son los **objetivos o funcionalidades** que el sistema ofrece a los actores.
* Se representan con óvalos dentro del rectángulo del sistema.
* Deben escribirse con un nombre claro: verbo + sustantivo (ej. “Realizar pago”).

### **Relaciones**

* **Asociación**: línea simple entre actor y caso de uso.
* **Include (<<include>>)**: cuando un caso de uso **siempre incluye** la ejecución de otro (ej. “Realizar pago” incluye “Procesar pago”).
* **Extend (<<extend>>)**: cuando un caso de uso **puede extenderse opcionalmente** con otro (ej. “Agregar al carrito” extiende “Realizar pago”).
* **Generalización**: herencia entre actores o casos de uso (ej. Usuario y Administrador).

### **Diagramas UML más relevantes**

1. **Diagrama de Casos de Uso**: describe qué hace el sistema desde la perspectiva del usuario.
2. **Diagrama de Clases**: muestra la estructura estática (atributos, métodos y relaciones entre clases).
3. **Diagrama de Secuencia**: representa la interacción entre objetos en el tiempo, mostrando mensajes enviados y respuestas.
4. **Diagrama de Actividades**: describe flujos de procesos, útil para modelar lógica de negocio.

## 3. Buenas prácticas señaladas por Visual Paradigm

* Mantener los diagramas **simples y claros** (no sobrecargarlos).
* Usar nombres consistentes y fáciles de entender.
* Documentar cada caso de uso con escenarios principales y alternativos.
* Organizar diagramas de manera que los actores estén siempre **fuera** y los casos de uso **dentro** del límite del sistema.
* En proyectos ágiles, usar UML de forma **ligera**: centrarse en casos de uso y diagramas de secuencia para complementar historias de usuario.

## 4. Utilidad práctica de UML hoy

* Permite **comunicación efectiva** entre distintas partes de un proyecto.
* Ayuda a **documentar y analizar requerimientos**.
* Es útil tanto en proyectos tradicionales como ágiles (aunque se usa de manera más resumida en estos últimos).
* UML se integra con herramientas modernas como Visual Paradigm, PlantUML o StarUML, lo que facilita la generación de diagramas y su integración con procesos de desarrollo.

## 5. Conclusión

Aprender UML con Visual Paradigm permite reforzar la teoría con práctica visual e interactiva. El lenguaje sigue siendo relevante porque ofrece una **visión compartida y estandarizada** de sistemas complejos. Su aplicación en proyectos reales ayuda a reducir malentendidos, mejorar la documentación y dar soporte tanto a fases de análisis como de diseño y mantenimiento.

# **Resumen UML – Visual Paradigm (Apuntes rápidos)**

| **Tema** | **Conceptos clave** |
| --- | --- |
| **UML** | Lenguaje estándar de modelado visual, no es una metodología. Facilita comunicación y documentación. |
| **Actores** | Roles externos (usuarios, sistemas). Siempre fuera del límite del sistema. |
| **Casos de uso** | Funcionalidades/objetivos del sistema. Óvalos dentro del sistema. Nombre: verbo + sustantivo. |
| **Relaciones** | - **Asociación**: línea simple actor ↔ caso de uso. - **<<include>>**: caso de uso obligatorio. - **<<extend>>**: caso de uso opcional. - **Generalización**: herencia entre actores o casos. |
| **Diagramas UML usados** | - Casos de uso (requerimientos). - Clases (estructura estática). - Secuencia (interacción en el tiempo). - Actividades (flujos de procesos). |
| **Buenas prácticas** | - Diagramas simples y legibles. - Actores fuera, casos dentro. - Documentar escenarios. - Usar UML de forma ligera en proyectos ágiles. |
| **Aplicación actual** | - Complementa historias de usuario. - Documenta sistemas complejos. - Compatible con herramientas modernas (Visual Paradigm, PlantUML). |

## 2. Considere el ejercicio 2 del teórico 2.1:

*Se pide:*

c) Identifique los casos de uso del sistema y elabore un diagrama de casos de uso.

### Actores del Sistema:

* **Cliente/Usuario**: Usuario final que navega y compra libros
* **Administrador**: Gestiona el catálogo y el sistema
* **Responsable de Marketing**: Analiza datos y estadísticas
* **Encargado Atención Cliente**: Gestiona reclamos y consultas
* **Sistema de Pago**: Actor externo para procesar pagos

### Casos de Uso Identificados:

#### Para Cliente/Usuario:

1. Gestionar carrito
2. Comprar libro
3. Consultar estado de pedido
4. Buscar libros
5. Ver detalle del libro
6. Consultar disponibilidad
7. Gestionar dirección de envío
8. Recibir recomendaciones
9. Registrar opinión/reseña
10. Confirmar compra

#### Para Administrador:

1. Administrar catálogo
2. Generar estadísticas

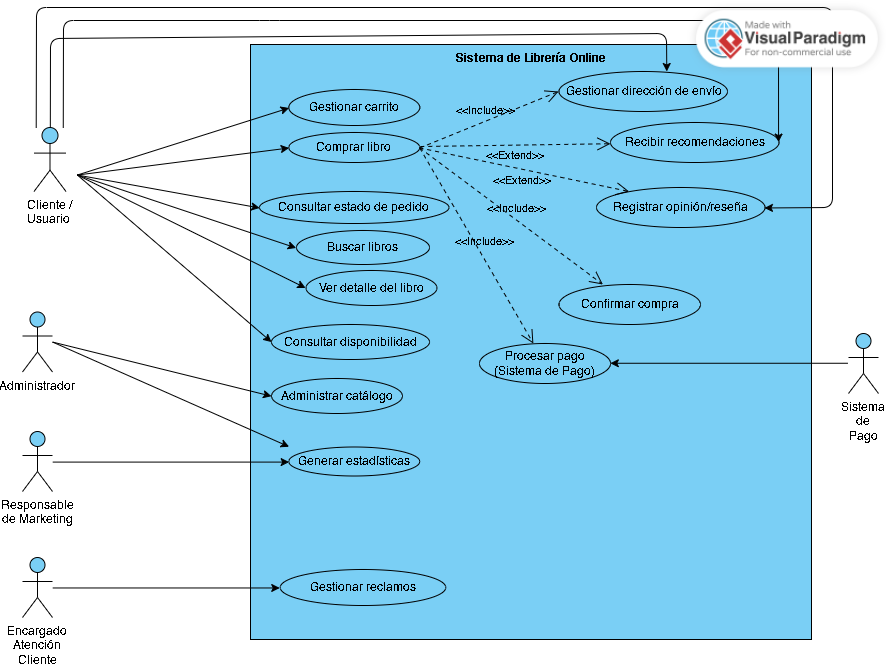
#### Para Responsable de Marketing:

1. Generar estadísticas (mismo caso de uso que Administrador)

#### Para Encargado Atención Cliente:

1. Gestionar reclamos

#### Casos de Uso del Sistema:

1. Procesar pago (con Sistema de Pago)

d) Escriba el escenario principal correspondiente a cada uno de ellos.

### 1. Gestionar carrito

**Actor Principal**: Cliente/Usuario

**Escenario**: El usuario puede agregar libros a su carrito, modificar cantidades, eliminar items y ver el total de su compra antes de proceder al checkout.

### 2. Comprar libro

**Actor Principal**: Cliente/Usuario

**Escenario**: El usuario selecciona libros, los agrega al carrito, proporciona información de envío y pago, y completa la transacción.

### 3. Consultar estado de pedido

**Actor Principal**: Cliente/Usuario

**Escenario**: El usuario ingresa su número de pedido o accede a su cuenta para ver el estado actual de sus compras (procesando, enviado, entregado).

### 4. Buscar libros

**Actor Principal**: Cliente/Usuario

**Escenario**: El usuario ingresa criterios de búsqueda (título, autor, género) y el sistema muestra los resultados coincidentes.

### 5. Ver detalle del libro

**Actor Principal**: Cliente/Usuario

**Escenario**: El usuario selecciona un libro específico para ver información detallada como descripción, precio, autor, reseñas y disponibilidad.

### 6. Consultar disponibilidad

**Actor Principal**: Cliente/Usuario

**Escenario**: El usuario verifica si un libro específico está en stock antes de realizar la compra.

### 7. Gestionar dirección de envío

**Actor Principal**: Cliente/Usuario

**Escenario**: El usuario registra, modifica o selecciona direcciones de envío para sus pedidos.

### 8. Recibir recomendaciones

**Actor Principal**: Cliente/Usuario

**Escenario**: El sistema sugiere libros basándose en el historial de compras y preferencias del usuario.

### 9. Registrar opinión/reseña

**Actor Principal**: Cliente/Usuario

**Escenario**: El usuario deja comentarios y calificaciones sobre libros que ha comprado.

### 10. Confirmar compra

**Actor Principal**: Cliente/Usuario

**Escenario**: El usuario revisa su pedido final y

confirma la transacción.

### 11. Administrar catálogo

**Actor Principal**: Administrador

**Escenario**: El administrador puede agregar nuevos libros, editar información existente, actualizar precios y gestionar el inventario.

### 12. Generar estadísticas

**Actor Principal**: Administrador, Responsable de Marketing

**Escenario**: El sistema genera reportes sobre ventas, búsquedas populares, libros más vendidos y métricas del negocio.

### 13. Gestionar reclamos

**Actor Principal**: Encargado Atención Cliente

**Escenario**: El encargado recibe, procesa y da seguimiento a las consultas y reclamos de los clientes.

### 14. Procesar pago

**Actor Principal**: Sistema de Pago

**Escenario**: El sistema externo procesa la información de pago y confirma o rechaza la transacción.

e) Escriba el desarrollo expandido para cada caso de uso del sistema (o sea, para cada uno de ellos, escriba una ficha que detalle actores, puntos de extensión y de inclusión, precondiciones, poscondiciones, flujo principal y flujos alternativos).

### **1-Gestionar carrito**

* **Actor**: Cliente/Usuario
* **Precondiciones**: El usuario debe haber iniciado sesión.
* **Postcondiciones**: El carrito queda actualizado con los ítems seleccionados.
* **Flujo Principal**:

1. El usuario agrega un libro al carrito.
2. El sistema actualiza cantidad y total.
3. El usuario puede modificar cantidades o eliminar ítems.
4. El sistema refleja los cambios.

* **Flujos Alternativos**:

2a. Libro sin stock → el sistema informa.

* **Inclusiones**: <<include>> Ver detalle del libro.
* **Extensiones**: <<extend>> Consultar disponibilidad.

### 2. **Comprar Libro**

* **Actor**: Cliente/Usuario
* **Precondiciones**: Carrito con al menos un libro, usuario autenticado.
* **Postcondiciones**: Pedido registrado en el sistema.
* **Flujo Principal**:
  1. El usuario confirma carrito.
  2. Selecciona dirección de envío.
  3. Ingresa datos de pago.
  4. El sistema envía transacción al Sistema de Pago.
  5. Pago aprobado.
  6. Se genera la orden.
* **Flujos Alternativos**:

4a. Pago rechazado → informar y permitir reintento.

* **Inclusiones**: <<include>> Procesar pago, <<include>> Gestionar dirección de envío.
* **Extensiones**: <<extend>> Confirmar compra.

### 3. **Consultar Estado de Pedido**

* **Actor**: Cliente/Usuario
* **Precondiciones**: Usuario con compras registradas.
* **Postcondiciones**: Estado del pedido mostrado.
* **Flujo Principal**:
  1. Usuario ingresa ID de pedido o historial.
  2. Sistema muestra estado (procesando, enviado, entregado).
* **Flujos Alternativos**:

1a. Pedido no válido → error.

### 4. **Buscar Libros**

* **Actor**: Cliente/Usuario
* **Precondiciones**: Acceso al sistema.
* **Postcondiciones**: Resultados mostrados.
* **Flujo Principal**:
  1. Usuario ingresa criterios (título, autor, género).
  2. Sistema lista coincidencias.
* **Flujos Alternativos**:

2a. No hay resultados → mensaje “sin coincidencias”.

### 5. **Ver Detalle del Libro**

* **Actor**: Cliente/Usuario
* **Precondiciones**: Búsqueda realizada.
* **Postcondiciones**: Mostrar ficha del libro.
* **Flujo Principal**:
  1. Usuario selecciona un libro.
  2. Sistema muestra descripción, precio, autor, reseñas.
* **Extensiones**: <<extend>> Consultar disponibilidad.

### 6. **Consultar Disponibilidad**

* **Actor**: Cliente/Usuario
* **Precondiciones**: Selección de libro.
* **Postcondiciones**: Mostrar disponibilidad.
* **Flujo Principal**:
  1. Sistema valida stock del libro.
  2. Muestra cantidad disponible.
* **Flujos Alternativos**:

1a. Libro agotado → mensaje “no disponible”.

### 7. **Gestionar Dirección de Envío**

* **Actor**: Cliente/Usuario
* **Precondiciones**: Usuario autenticado.
* **Postcondiciones**: Dirección actualizada.
* **Flujo Principal**:
  1. Usuario agrega o modifica dirección.
  2. El sistema guarda cambios.

### 8. **Recibir Recomendaciones**

* **Actor**: Cliente/Usuario
* **Precondiciones**: Historial de compras o navegación.
* **Postcondiciones**: Mostrar sugerencias.
* **Flujo Principal**:
  1. Sistema analiza historial.
  2. Genera recomendaciones personalizadas.

### 9. **Registrar Opinión/Reseña**

* **Actor**: Cliente/Usuario
* **Precondiciones**: Haber comprado el libro.
* **Postcondiciones**: Reseña guardada.
* **Flujo Principal**:
  1. Usuario escribe comentario y califica.
  2. El sistema guarda y asocia al libro.
* **Flujos Alternativos**:

1a. Usuario no compró el libro → acción rechazada.

### 10. **Confirmar Compra**

* **Actor**: Cliente/Usuario
* **Precondiciones**: Pedido creado, pago aprobado.
* **Postcondiciones**: Compra confirmada.
* **Flujo Principal**:
  1. Sistema muestra resumen final.
  2. Usuario confirma operación.

### 11. **Administrar Catálogo**

* **Actor**: Administrador
* **Precondiciones**: Administrador autenticado.
* **Postcondiciones**: Catálogo actualizado.
* **Flujo Principal**:
  1. Agregar, editar o eliminar libros.
  2. El sistema actualiza base de datos.
* **Flujos Alternativos**:

2a. Libro en préstamo no puede eliminarse.

### 12. **Generar Estadísticas**

* **Actor**: Administrador / Responsable de Marketing
* **Precondiciones**: Datos históricos de ventas.
* **Postcondiciones**: Reportes generados.
* **Flujo Principal**:
  1. Usuario solicita reporte.
  2. Sistema procesa y genera estadísticas.

### 13. **Gestionar Reclamos**

* **Actor**: Encargado Atención Cliente
* **Precondiciones**: Reclamo registrado.
* **Postcondiciones**: Estado del reclamo actualizado.
* **Flujo Principal**:
  1. Encargado recibe consulta/reclamo.
  2. Registra acciones de seguimiento.
  3. El sistema actualiza estado.

### 14. **Procesar Pago**

* **Actor**: Sistema de Pago (externo)
* **Precondiciones**: Pedido en proceso de compra.
* **Postcondiciones**: Pago aprobado o rechazado.
* **Flujo Principal**:
  1. Sistema envía datos al Sistema de Pago.
  2. Validación de tarjeta y fondos.
  3. Resultado enviado (aprobado/rechazado).
* **Flujos Alternativos**:

2a. Fallo en conexión → notificar error y cancelar.

### 15. **Consultar Estado de Pedido** (ya incluido en 3, pero se mantiene para consistencia)

* **Actor**: Cliente/Usuario (a través del sistema)
* **Precondiciones**: El usuario debe tener al menos un pedido.
* **Postcondiciones**: El sistema devuelve el estado actual del pedido.
* **Flujo Principal**:

Se corresponde con el flujo detallado en el **caso de uso 3**.  
**Nota**: Este caso de uso es equivalente al Caso 3, se mantiene aquí solo por consistencia y alineación con la lista inicial de casos identificados.