

Grado en Ingeniería Informática
Fundamentos de Programación II 25/26

Sesión 2:
**Fases de desarrollo de software
orientado a objetos**

Resumen

En esta segunda sesión de prácticas, nos centraremos en las fases del desarrollo de software orientado a objetos. En otras asignaturas del grado se tratarán más en profundidad los aspectos relacionados con las distintas etapas y metodologías de desarrollo. En esta sesión, únicamente nos centraremos en una visión general de las fases del desarrollo de software, especialmente en lo referente a la programación orientada a objetos.

1. Objetivos

- Conocer las fases del desarrollo de software orientado a objetos.
- Resolver un caso práctico completo aplicando estas fases.

2. Desarrollo de la práctica

2.1. Fases del desarrollo de software orientado a objetos

En las sesiones de teoría se han visto de forma general las fases del desarrollo de software orientado a objetos. Podemos resumirlas en tres grandes bloques:

- **Análisis:** En esta fase se identifican los requisitos del sistema. Intentamos encontrar los **objetos** y **relaciones** adecuados para el problema propuesto. Investigamos el problema centrándonos en los conceptos del dominio del problema, sin preocuparnos por la implementación. Es importante entender el problema y definir claramente los requisitos antes de pasar a la siguiente fase.
- **Diseño:** En esta fase se define la **arquitectura** del sistema. Partiendo de los objetos y relaciones identificados en el análisis, se diseña la estructura del sistema. Se definen las clases, sus atributos y métodos (aún sin pensar en la implementación, solo en aquella funcionalidad que cada clase debe tener), así como las relaciones entre ellas (herencia, composición, etc.). Es importante diseñar un sistema modular y flexible que permita cambios futuros sin afectar a toda la aplicación. En esta fase es fundamental el uso de diagramas UML (especialmente el diagrama de clases).
- **Implementación:** Únicamente cuando tenemos claro el diseño, pasamos a la **implementación**. En esta fase se escribe el código fuente de la aplicación siguiendo el diseño establecido. Es importante seguir buenas prácticas de programación y mantener el código limpio y organizado.

2.2. Caso práctico

Para ilustrar estas fases, vamos a resolver un caso práctico completo. Supongamos que queremos desarrollar un **sistema de gestión de pedidos para una librería online**.

2.2.1. Descripción del problema

Una pequeña librería online necesita un sistema sencillo para gestionar los pedidos de sus clientes. La librería dispone de un catálogo de libros, donde cada libro tiene un código único, un título, un autor y un precio. Los clientes de la librería están registrados con sus datos personales: nombre completo, dirección de email y dirección postal.

Cuando un cliente realiza un pedido, se identifica mediante un número de pedido único y se registra la fecha del pedido. Cada pedido puede contener una serie de libros, hasta un máximo de 20. El pedido también tiene que tener la información del cliente que lo realizó.

Una vez completado el pedido el sistema debe ser capaz de calcular el importe total sumando los precios de todos los libros incluidos y enviar una notificación de confirmación al cliente a través de un servicio externo (ServicioNotificaciones) que tenga únicamente una función¹ que reciba la información del pedido.

El sistema debe generar un resumen completo de cada pedido. Este resumen debe incluir toda la información relevante: los datos del cliente que realizó el pedido, la lista detallada de libros (con su código, título, autor y precio), y el importe total del pedido. De forma opcional, este resumen puede guardarse en un fichero de texto llamado `resumen_pedidos.txt` que contenga la información de todos los pedidos realizados.

Importante: En vuestro análisis y diseño, debéis identificar las entidades principales del sistema y sus responsabilidades. El sistema debe tener clases que representen estas entidades. Es fundamental que identifiquéis correctamente las relaciones entre ellas: debe haber al menos una relación de **asociación** (cuando una clase mantiene una colección o referencia permanente a objetos de otra clase como atributo) y una relación de **dependencia** (cuando una clase utiliza objetos de otra clase como parámetros de métodos, pero no los almacena como atributos).

2.2.2. Instrucciones de trabajo

Para resolver este caso práctico, sigue las fases del desarrollo de software orientado a objetos:

Fase 1: Análisis

- Identifica los objetos del sistema (¿qué entidades necesitamos representar?).
- Identifica las relaciones entre los objetos.
- Define claramente qué debe hacer cada objeto (responsabilidades).

Fase 2: Diseño

- Diseña el diagrama de clases UML del sistema.
- Para cada clase, define:
 - Atributos (con su tipo y visibilidad)
 - Métodos (con sus parámetros, tipo de retorno y visibilidad)

¹Esto se puede simplificar mostrando un mensaje por pantalla con la información del pedido y del cliente

- Constructor(es)
- Indica claramente las relaciones entre las clases (asociación, dependencia) con la notación UML adecuada.

Fase 3: Implementación

- Implementa cada clase en Java siguiendo el diseño.
- Implementa el método `main` que demuestre el funcionamiento del sistema.
- (Opcional:) Implementa la escritura del fichero de salida.
- Prueba el programa y verifica que funcione correctamente.

Importante: Es fundamental seguir las fases en orden. NO empieces a programar sin haber completado el análisis y el diseño. El diagrama UML debe estar completo antes de escribir una sola línea de código Java.

2.2.3. Ejemplo de salida esperada

Como referencia, la salida generada por vuestro programa podría tener un contenido similar al siguiente:

```
RESUMEN DE PEDIDOS
=====
```

```
Pedido 1001
```

```
Fecha: 16/02/2026
```

```
Cliente: María García López
```

```
Email: maria.garcia@email.com
```

```
Dirección: Calle Mayor 15, Toledo
```

```
Libros:
```

```
- Código: L001, Título: El Quijote, Autor: Miguel de Cervantes,
```

```
Precio: 15,50€
```

```
- Código: L002, Título: Cien años de soledad, Autor: Gabriel García Márquez,
```

```
Precio: 18,90€
```

```
- Código: L003, Título: 1984, Autor: George Orwell, Precio: 12,95€
```

```
Total del pedido: 47,35€
```

```
Pedido 1002
```

```
Fecha: 16/02/2026
```

```
Cliente: Juan Martínez Rodríguez
```

```
Email: juan.martinez@email.com
```

```
Dirección: Avenida de España 42, Ciudad Real
```

```
Libros:
```

```
- Código: L004, Título: El principito, Autor: Antoine de Saint-Exupéry,
```

```
Precio: 9,95€
```

- Código: L001, Título: El Quijote, Autor: Miguel de Cervantes,

Precio: 15,50€

Total del pedido: 25,45€

Nota: Este es solo un ejemplo orientativo. Vuestro programa debe generar una salida similar que incluya toda la información relevante de forma clara y estructurada.

3. Entrega

3.1. Contenido de la entrega

Se debe entregar en la tarea correspondiente de Campus Virtual un fichero comprimido (ZIP) que contenga:

- El código fuente de la aplicación (todos los archivos .java).
- El diagrama de clases UML en formato PDF o imagen (PNG, JPG).