



# Programación 4

## Guía Semana 1 (4/3)

InCo, FING, Udelar

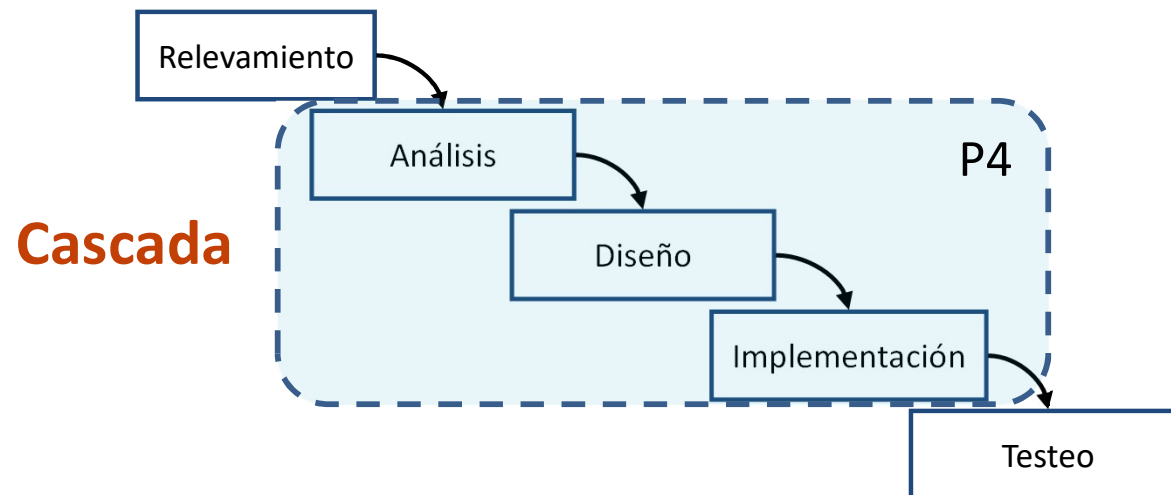
# Objetivo

El objetivo de esta semana es **contextualizar** el paradigma de **Orientación a Objetos (OO)** en el marco de la Ingeniería de Software, así como comenzar a ver sus **conceptos básicos** y cómo éstos se implementan en **C++**.



# Resumen :: OO

Un **proceso de desarrollo** de software describe un enfoque para construir, instalar y mantener sistemas de software. Hay diversos enfoques que varían las actividades y su orden.



# Resumen :: OO

Las actividades dependen del **paradigma de desarrollo** utilizado.

Utilizaremos la **Orientación a Objetos** que puede ser entendida como:

- Una forma de pensar basada en abstracciones de conceptos existentes en el mundo real.
- La organización del software como una colaboración de objetos que interactúan entre sí por medio de mensajes.



# Resumen :: OO

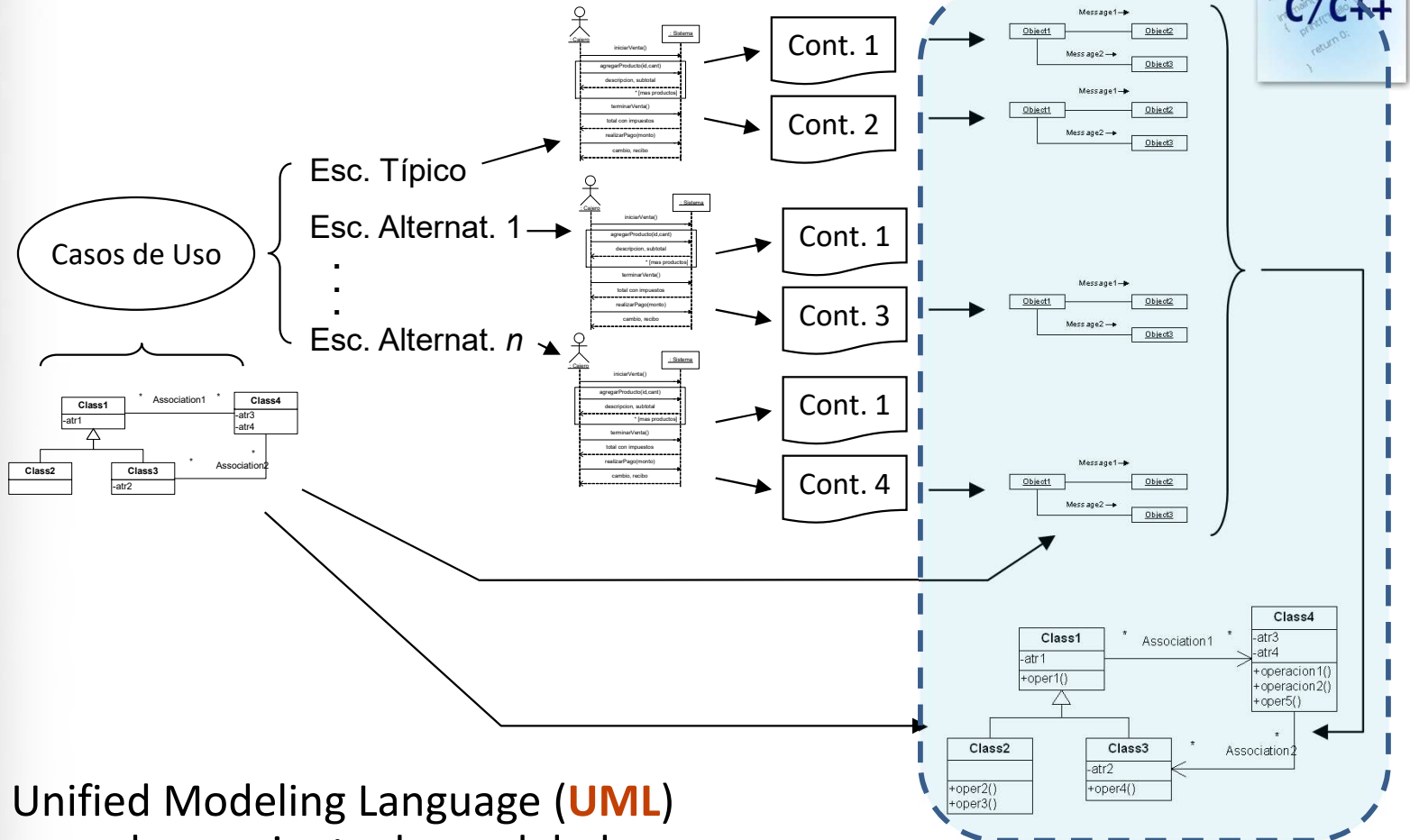
**Análisis OO:** describir los conceptos en el dominio de la aplicación y sus relaciones, para comprender mejor la realidad y el problema.

**Diseño OO:** definir objetos lógicos (de software) y la forma de comunicación entre ellos para brindar una solución programable.

**Implementación OO:** codificar en un lenguaje de programación OO las construcciones definidas en el diseño.



# Resumen :: OO



Unified Modeling Language (UML) como herramienta de modelado.

# Resumen :: Conceptos OO

Una **clase** describe un conjunto de objetos que comparten los mismos atributos, operaciones y relaciones (entre otros).

Los **atributos** son descripciones de un compartimiento de un tipo (que tendrán valores para cada objeto).

```
class Cancion {  
    string nombre;  
    Cancion::Cancion(string titulo);  
};
```

Un **objeto** es una entidad discreta con límites e identidad bien definidos.

```
Cancion *c = new Cancion("Nombre");  
delete c;
```



# Resumen :: Conceptos OO

**Operación:** es una especificación de una transformación o consulta que un objeto puede ser llamado a ejecutar. Tiene un nombre, una lista de parámetros y un tipo de retorno.

```
class Genero {  
    string getNombre();  
};
```

**Método:** es la implementación de una operación (algoritmo) para una determinada clase.

```
string Genero::getNombre() {  
    return "el genero es" + nombre;  
}
```







# Resumen :: Conceptos OO

Una **asociación** describe una relación semántica entre clases.

```
class Genero {  
    private:  
        string nombre;           // atributo  
        Vector<Cancion> canciones; // pseudoatributo  
        ...  
};
```

Una asociación entre clases indica que es posible “conectar” entre sí instancias de dichas clases.

# Resumen :: Conceptos OO

Una **generalización** es una relación taxonómica entre un elemento (clase, data type, interfaz) más general y entre un elemento más específico

```
class Artista {  
    private: string nombre;  
};  
  
class Banda: public Artista {  
    private: Fecha *fundacion;  
};
```

El elemento más específico es consistente con el más general, y puede contener información adicional. Sus propiedades se comparten gracias a la **herencia**.



# Resumen :: Conceptos OO

El **polimorfismo** es la capacidad de asociar diferentes métodos a la misma operación.

```
class Artista {
    public: virtual string toString();
};
string Artista::toString() {
    return nombre;
}

class Banda: public Artista {
    public: virtual string toString();
};
string Banda::toString() {
    return "Banda ->" + Artista::toString();
}
```



# ¿Qué hago esta semana?

1. Estudio los materiales de [Teórico](#) y las lecturas recomendadas. Las clases correspondientes se encuentran en [OpenFing](#).
  - 01 - Introducción al Curso
  - 02 - Desarrollo OO basado en UML
  - 03 - Conceptos Básicos de OO (hasta diap. 55)
2. Estudio los [instructivos](#) sobre C++
  - Conceptos Básicos de Implementación en C++
  - Referencias Circulares y Namespaces
  - Instructivo de Compilación
  - Thinking in C++ 2nd Edition (complementario)
4. Conformo grupo para laboratorio (5 personas). **En el correr de la semana se darán las instrucciones para el procedimiento de asignación de horarios y docentes de monitoreo.**

