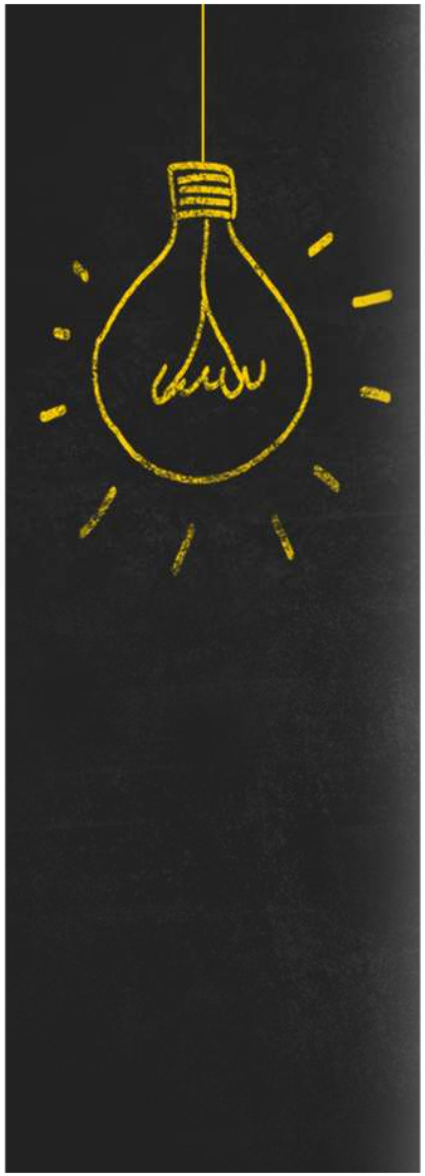
Programación 4
Guía Semana 8 (13/5)

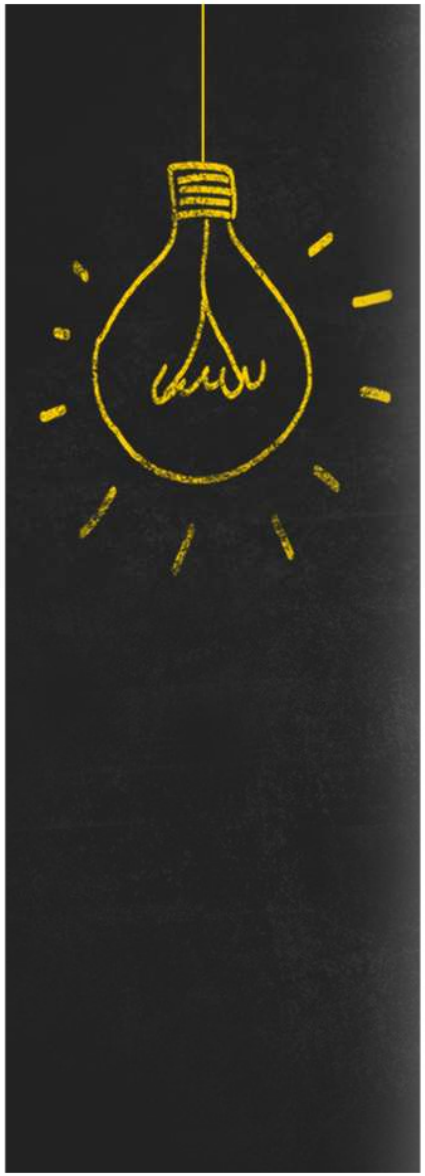
InCo, FING, UdelaR



Objetivo

Los objetivos de esta semana son:

1. definir nuevos **conceptos básicos de OO** y cómo éstos se implementan en **C++**;
2. ilustrar un posible uso que se le puede dar a los artefactos construidos en Análisis con la **generación parcial de código**;
3. introducir al **Diseño OO**, presentando el concepto de arquitectura y de diseño de bajo nivel;
4. presentar el **diseño de interacciones y de estructura** para la realización de los Casos de Uso;
5. introducir los **Diagramas de Comunicación** como herramienta para el diseño de interacciones.



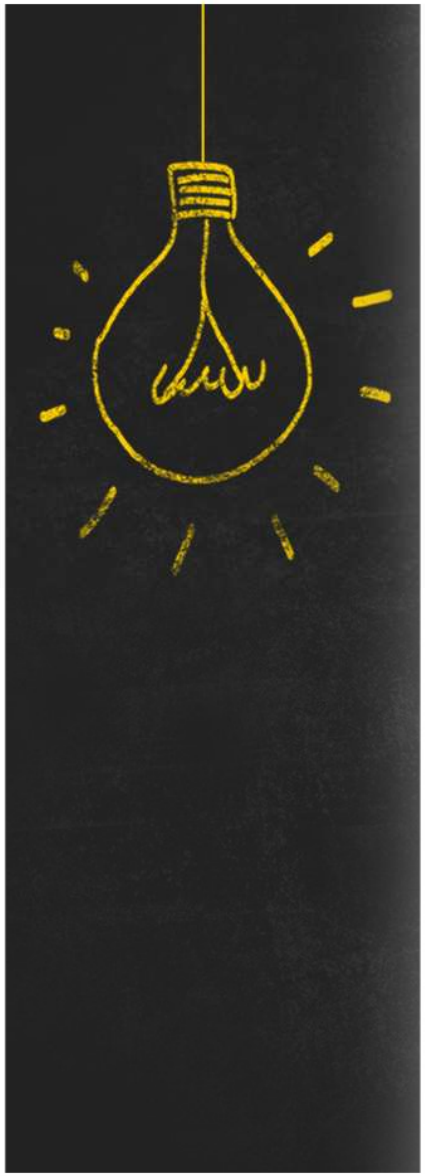
Resumen :: Conceptos OO

Una **interfaz** define un conjunto de operaciones, sin un estado ni método para las operaciones

Puede ser entendida como la especificación de un **rol** que un *objeto* debe desempeñar

Se dice que una clase ***C* realiza una interfaz *I*** si *C* provee un método para todas las operaciones declaradas en *I*

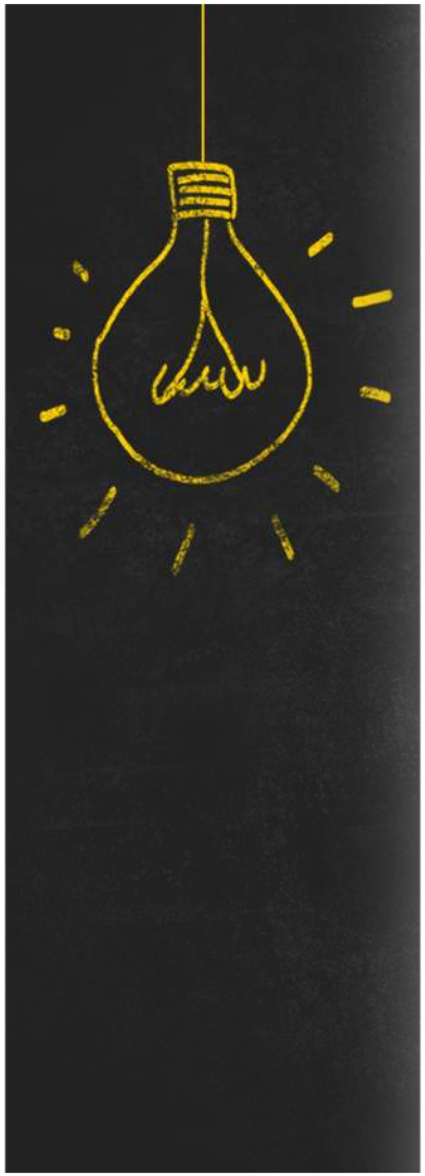
Una clase puede implementar varias interfaces



Resumen :: Conceptos OO

```
class Comparable {
public:
        // devuelve algo mayor a 0 si this > b
        // 0 si this = b, o algo negativo si this < b
        virtual int comparar(Comparable *b) = 0;
        virtual ~Comparable() {};
};

class Representable {
public:
        // devuelve una representación del objeto
        virtual string toString() = 0;
        virtual ~Representable() {};
}
```

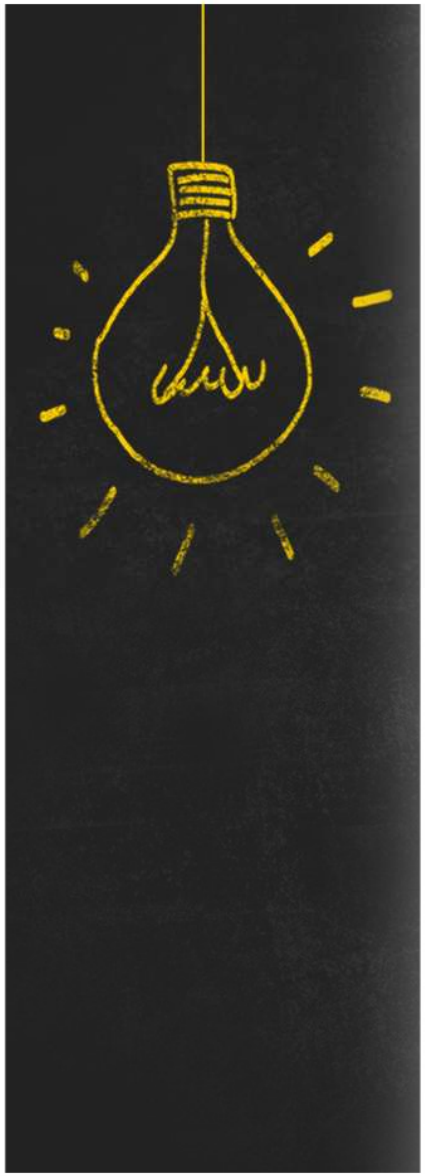


Resumen :: Conceptos OO

```
class Integer: public Comparable, public Representable {
private:
        int val;
public:
        Integer(int val);
        int getVal();
        string toString();
        int comparar(Comparable *c);
};

ostream& operator <<(ostream& o, Representable& r)
{
        return o << r.toString();
}

void Utilidades::ordenar(Comparable **arr, int largo)
{
        // ;)
}
```



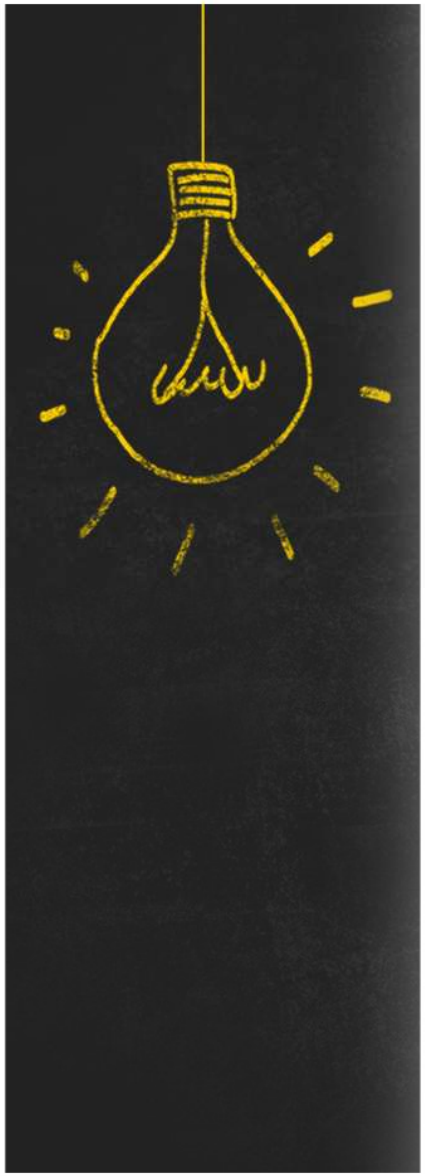
Resumen :: Generación de Código

Durante el **Análisis** se producen el Modelo de Dominio, los Diagrama de Secuencia del Sistema y los Contratos

Se pueden utilizar estos artefactos para **generar código**

Es necesario realizar ciertas hipótesis, por lo que **no es posible generar una implementación completa**.

Por lo tanto, este esqueleto de código debe considerarse como **modificable y no final**

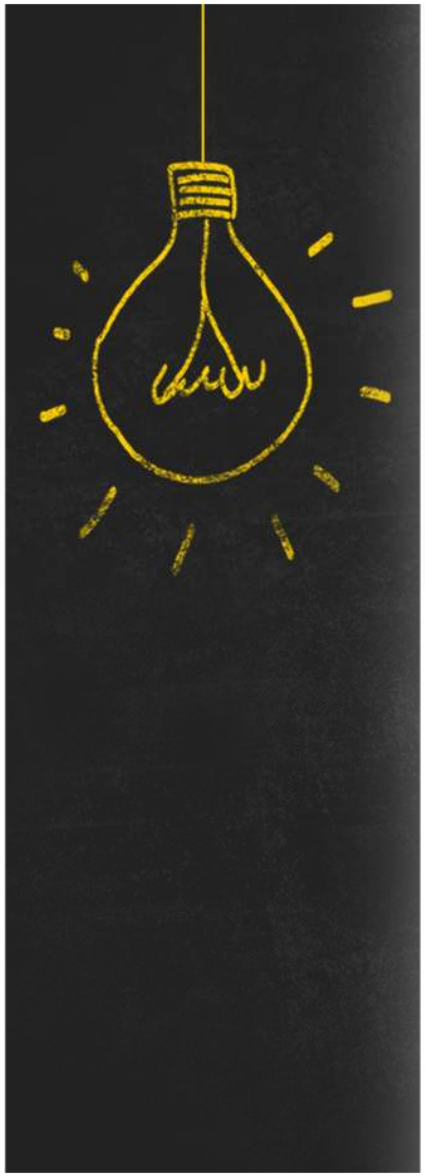


Resumen :: Generación de Código

Un **Modelo de Dominio** presenta los conceptos y relaciones más relevantes del problema

Se puede generar un esqueleto de código, particularmente de su estructura (no de su comportamiento) asumiendo que los **conceptos** serán **clases adecuadas** en el Diseño

Hay decisiones que se toman en Diseño, por ejemplo, cómo se representan los Tipos Asociativos

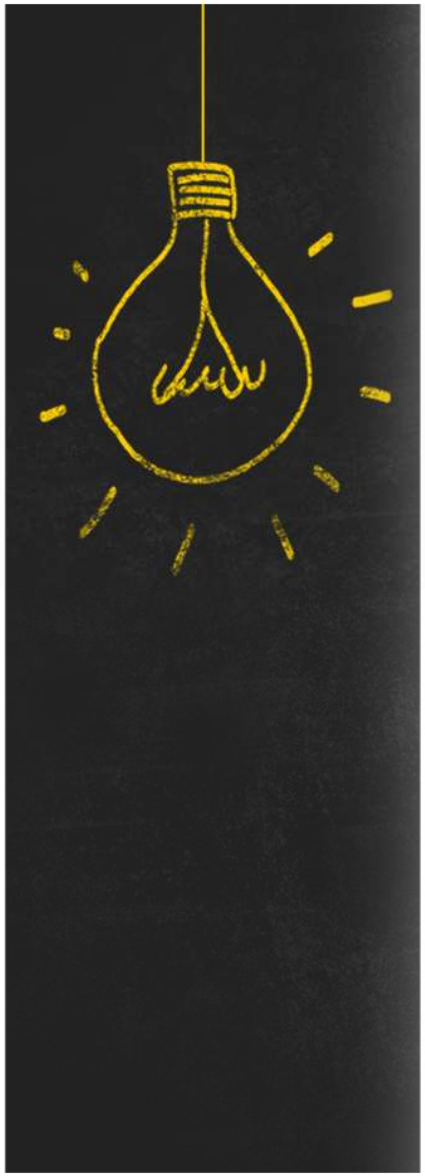


Resumen :: Generación de Código

Los **Diagramas de Secuencia del Sistema** ilustran cómo los usuarios interactúan con el Sistema (como caja negra) a través de operaciones

Se puede generar un esqueleto del código del método `main()` a partir de esa información

Hay decisiones que se toman en Diseño, por ejemplo, el comportamiento concreto de las operaciones (definidos en los **Contratos**)

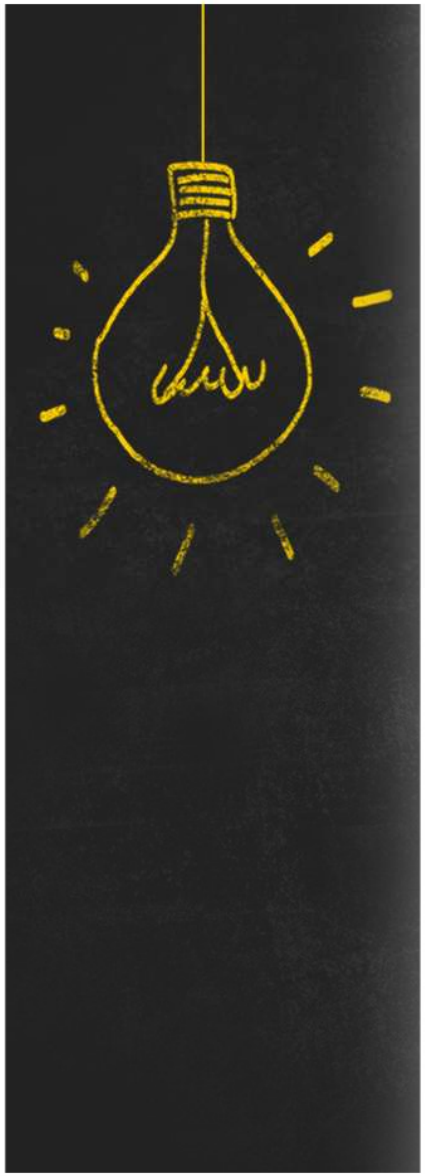


Resumen :: Diseño OO

Durante el análisis un caso de uso fue reformulado en términos de interacciones entre los actores y el sistema (**DSS**)

El efecto de cada mensaje fue especificado en forma precisa (**Contrato**)

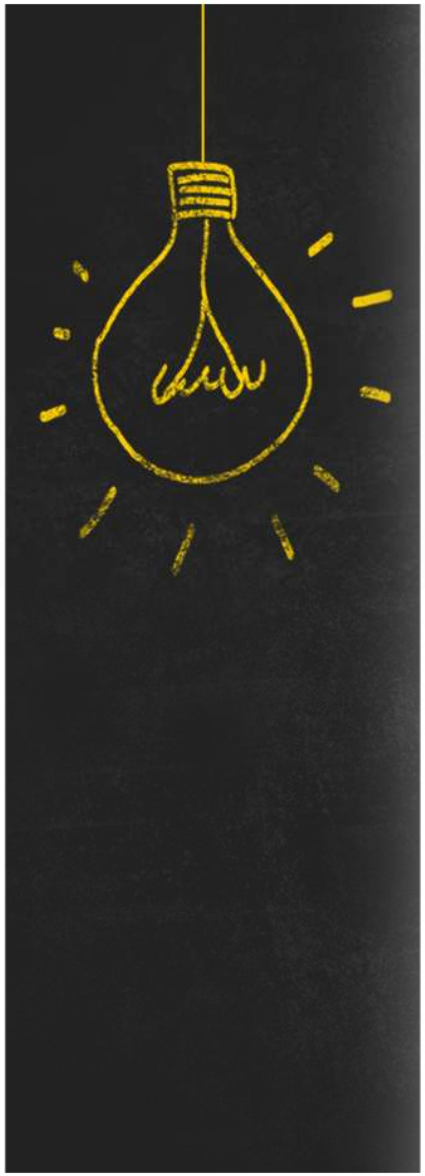
Es el momento de definir **cómo** hace el sistema internamente para resolver cada una de las operaciones del sistema



Resumen :: Diseño OO

Se puede decir que el diseño tiene dos niveles:

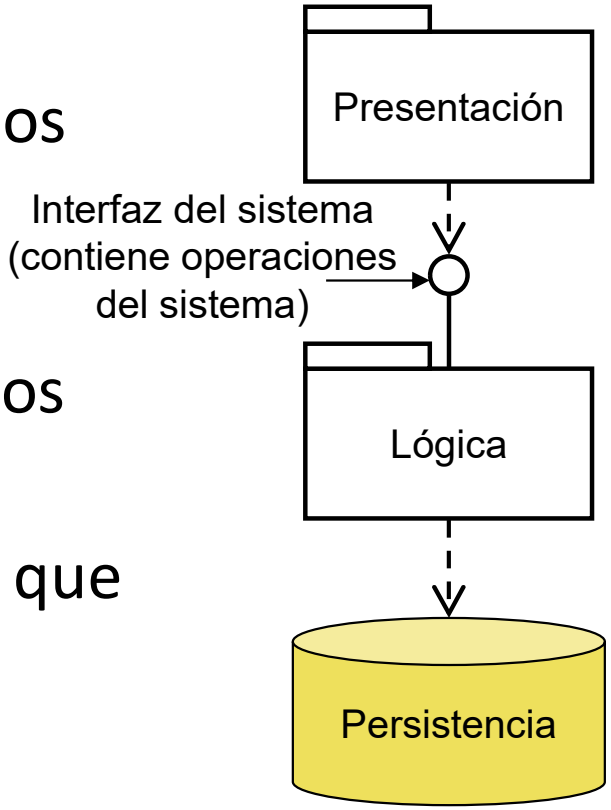
- **Arquitectura**: describe la estructura y organización del sistema de alto nivel, como conjunto de componentes relacionados entre sí, con responsabilidades específicas
- **Diseño de bajo nivel**: describe la estructura y comportamiento específico de cada componente

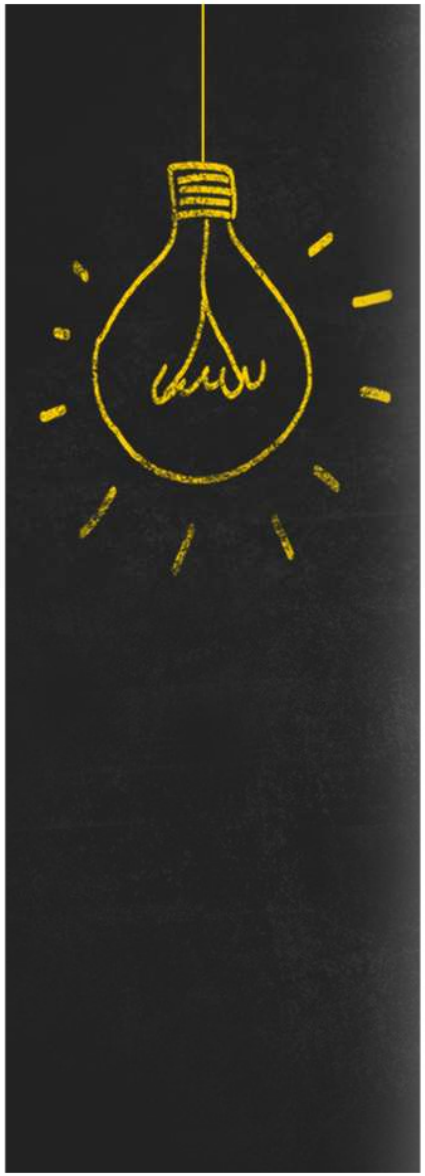


Resumen :: Diseño OO

Arquitectura en Capas

- Presentación: interacción con los usuarios
- Lógica: objetos que procesan la información para satisfacer los casos de uso del sistema
- Persistencia: datos del sistema que necesiten ser preservados



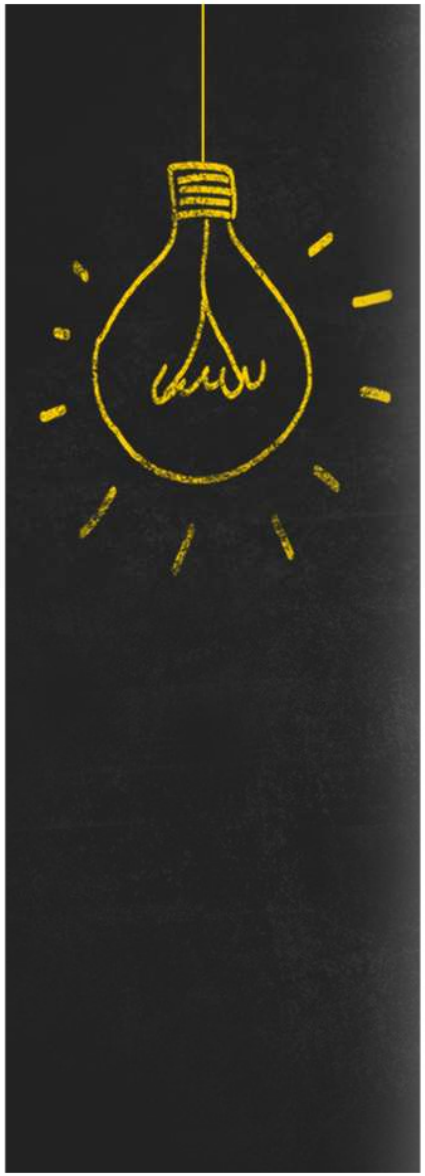


Resumen :: Diseño OO

En este curso, el **diseño de bajo nivel** estará enfocado en la **capa lógica**

Tenemos que **diseñar Colaboraciones** que realicen los Casos de Uso del sistema:

- Una **Estructura** con las clases, atributos, relaciones y operaciones que participan en la solución
- Las **Interacciones** que definen la forma en que objetos de estas clases se comunican para obtener el resultado deseado

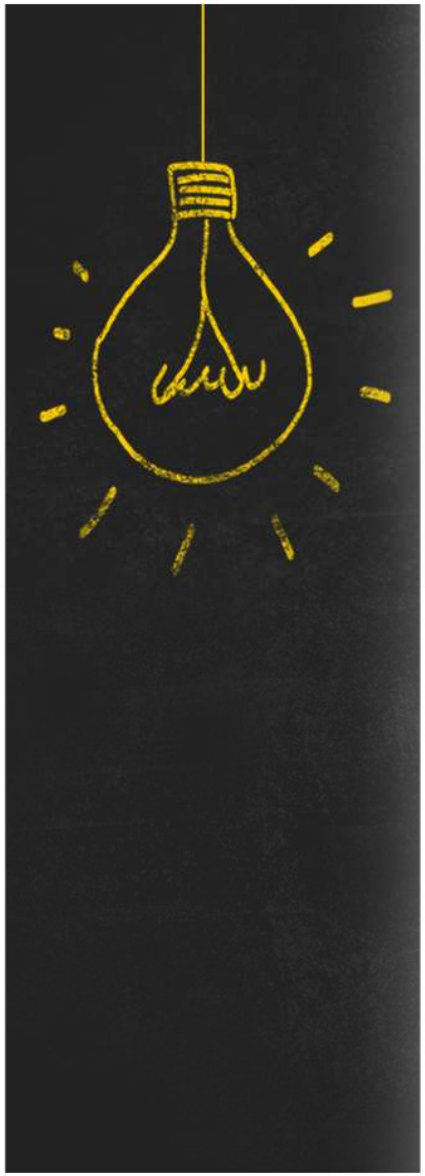


Resumen :: Diseño OO

El **Diseño de Interacciones** consiste en definir comunicaciones entre objetos que permitan resolver operaciones del sistema

- Los protagonistas aparecen “sugeridos” en el Modelo de Dominio
- El resultado es el especificado en el contrato de la operación del sistema a diseñar

Herramienta: **Diagrama de Comunicación**



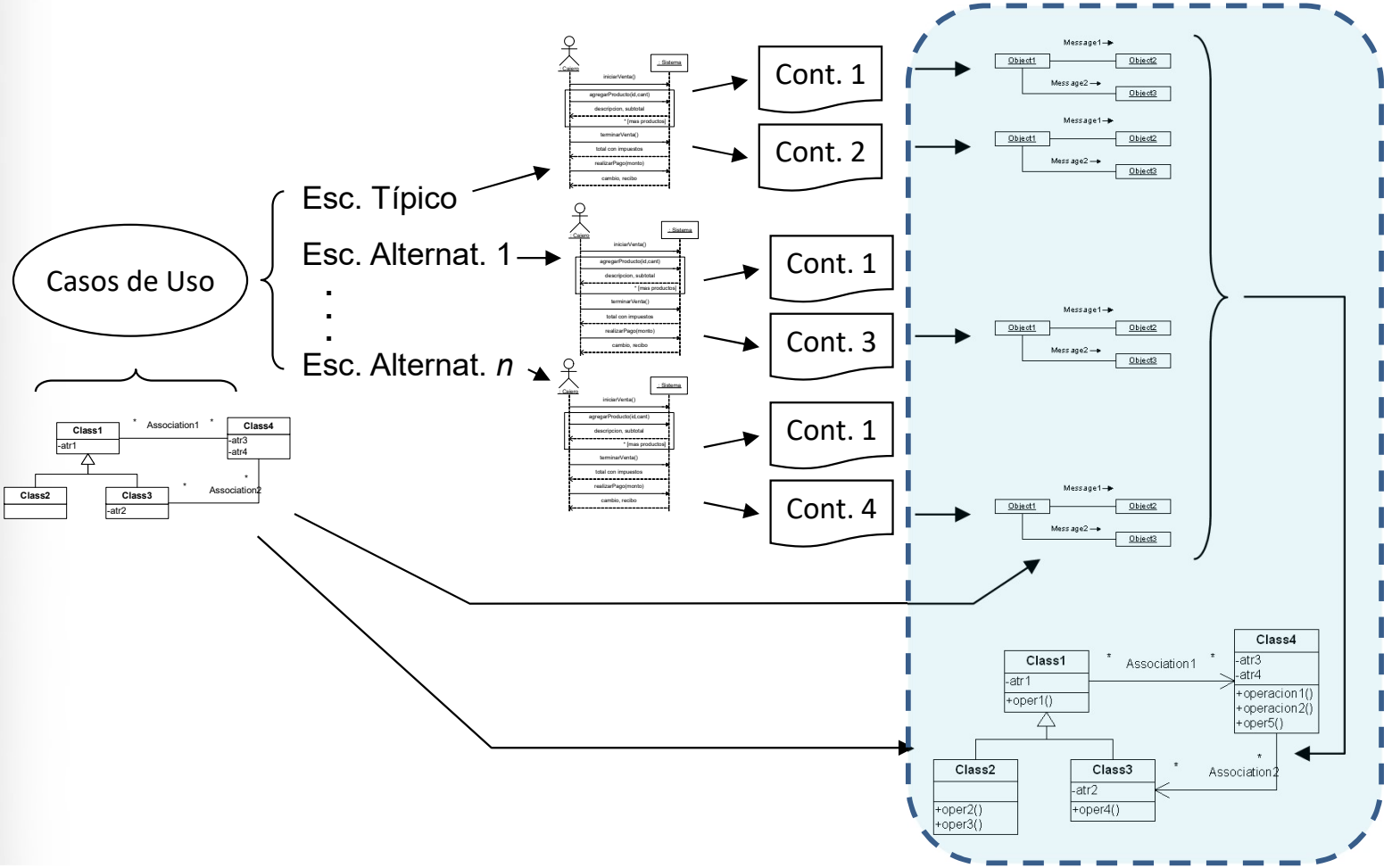
Resumen :: Diseño OO

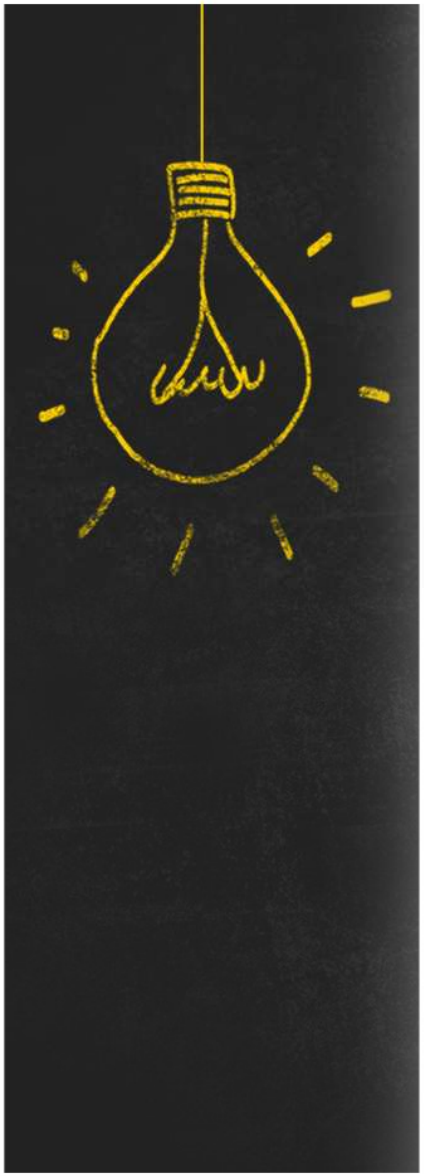
El **Diseño de Estructura** consiste en especificar completamente la estructura necesaria para que todas las interacciones puedan ocurrir

En el curso haremos una estructura que considere todas las interacciones

Herramienta: **Diagrama de Clases de Diseño**

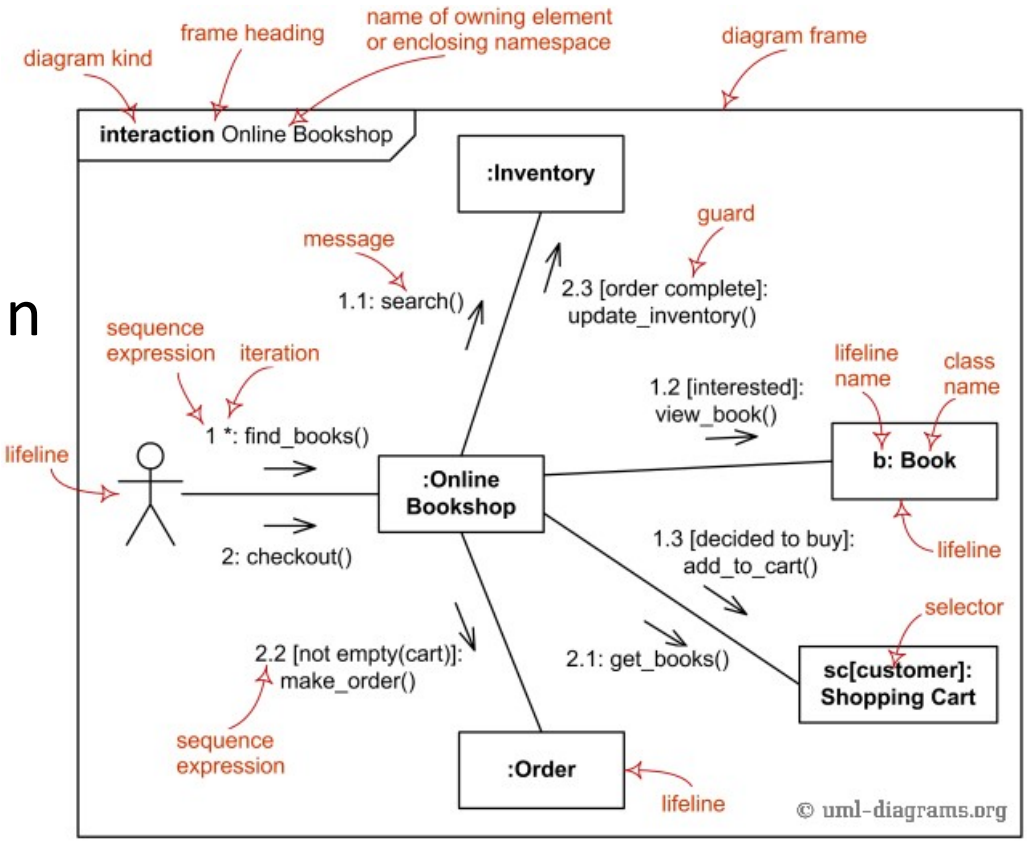
Resumen :: Diseño OO



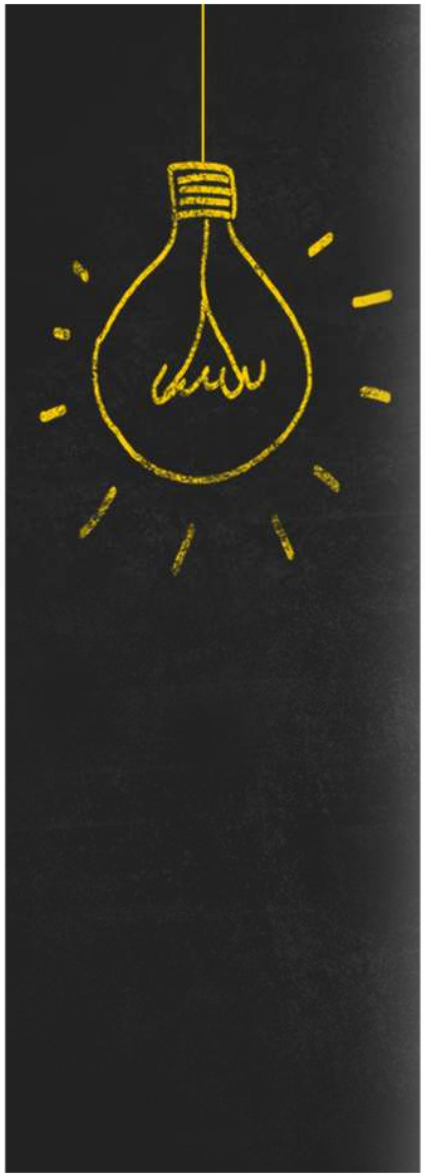


Resumen :: Diag. de Comunicación

Un **Diagrama de Comunicación** muestra cómo objetos interactúan a través de mensajes para la realización de tareas.



UML 2.5 Diagrams Overview
https://www.uml-diagrams.org



¿Qué hago esta semana?

1. Estudio los materiales de Teórico y las lecturas recomendadas. Las clases correspondientes se encuentran en OpenFing.
    - 08 - Conceptos Básicos de OO (2da Parte)
    - 07 - Generación Parcial de Código
    - 09 - Diseño: Introducción al Diseño
    - 10 - Diseño: Diagramas de Comunicación
2. Comienzo el Práctico 4 “Diseño, Diagramas de Comunicación”. Están publicadas las notas de resolución de los Ejercicios 2, 3, 8 y 12
3. Comienzo el Laboratorio 3 “Diseño”.
   Plazo de entrega: lunes 3/06, 15hs