### Modelación y Diseño de Sistemas

#### **Analista Universitario de Sistemas Informáticos**

Ing. Fernando Bono

2021





ESCMB
ESCUELA SUPERIOR
DE COMERCIO
MANUEL BELGRANO

Nombre: Fernando Bono

Mail: Fernando.Bono@unc.edu.ar

Celular: +54 - 9351 - 5122902

Skype: fer-bono

Whatsapp: <a href="http://bit.ly/2D6qZnM">http://bit.ly/2D6qZnM</a>



- Unidad I Software: Visión general
  - Introducción

- Unidad II Modelado de Software
  - Modelando casos de uso
  - Notación UML

- Unidad III Proyectos, Metodologías y Marcos de Trabajo
  - Ciclo de Vida del Software
  - Diseño de Software y Conceptos de Arquitectura
  - Metodologías Ágiles



- 1) El Modelado de Software
- 2) Modelado orientado a Objetos y el UML
- 3) Diseño de Arquitectura de Software
- 4) Método y Notación
- 5) COMET: Metodología de modelado y diseño colaborativo para el desarrollo de software
- 6) El estándar UML
- 7) Evolución del modelado de software y métodos de diseño
- 8) Evolución de los métodos de Diseño y Análisis Orientado a Objetos
- 9) Repaso de los métodos de diseño Concurrentes, distribuidos y en tiempo real.

# Unidad II - Modelado de Software Modelando casos de uso

- 1) Modelado de Requerimientos
- 2) Casos de uso
- 3) Actores
- 4) Identificación de casos de uso
- 5) Documentación de casos de uso en el Modelo de Caso de Uso
- 6) Requerimientos No funcionales
- 7) Diagramas de actividades

### **Unidad II - Modelado de Software** Notación UML

- 1) Diagramas UML
- 2) Diagramas de Casos de Uso
- 3) Clase y Objetos
- 4) Diagramas de Clases
- 5) Diagramas de Interacción
- 6) Diagramas de State Machine
- 7) Packages (paquetes)



### Unidad III - Proyectos, Metodologías y Marcos de Trabajo Ciclo de Vida de Software

- 1) Modelos de ciclos de vida de software
- 2) Diseño, verificación y validación
- 3) Testing de software



### Unidad III - Proyectos, Metodologías y Marcos de Trabajo Diseño de Software y Conceptos de Arquitectura

- 1) Conceptos de Orientación a Objetos
- 2) Ocultamiento de información
- 3) Herencia
- 4) Procesamiento concurrente
- Patrones de Diseño
- 6) Arquitectura de Software y componentes
- 7) Atributos de Calidad de Software



## Unidad III - Proyectos, Metodologías y Marcos de Trabajo Metodologías Ágiles

- 1) El Manifiesto Agile
- 2) Orígenes
- 3) <mark>Utilización</mark>
- 4) Características
- 5) Valores del Manifiesto Agile



### Unidad III - Proyectos, Metodologías y Marcos de Trabajo Metodologías Ágiles

- 6) Nivel de Ruido en el proyecto
- 7) Scrum Framework
  - 1) Ceremonias
    - 1) Sprint Planning
    - 2) Dailies Meetings
    - 3) Retrospecives
    - 4) Showcase / Demo
  - 2) Artefactos
    - 1) Product Backlog
      - 1) Backlog Item
      - 2) Gestión del Backlog
  - 3) Roles
    - 1) Scrum master
    - 2) Product owner
    - 3) Team members
    - 4) Stakeholders
  - 4) Burndown Chart

#### Modelación y Diseño de Sistemas

**Analista Universitario de Sistemas Informáticos** 



2020





ESCMB ESCUELA SUPERIOR DE COMERCIO MANUEL BELGRANO

- ¿Que es un modelo?
  - Un modelo es una abstracción de un sistema
  - La abstracción permite ocuparnos en detalles relevantes para un propósito
  - Utilidad del modelado: abordar sistemas complejos
  - Técnica muy empleada



- ¿Que ofrece el modelado?
  - Visualizar un Sistema
  - Especificar su comportamiento
  - Crear Plantillas que nos guíen durante el desarrollo
  - Documentar decisiones de diseño

- ¿Que Lenguaje utilizar para modelar software?
  - Código Fuente: difícil de entender y procesar
  - Lenguaje Natural: propenso a Errores
  - Lenguaje Visual: fácil de interpretar y procesar





• Arquitectura de Software

Conjunto de patrones que proporcionan un marco de referencia necesario para guiar la construcción de un software

Permite a los programadores, analistas y todo el conjunto de desarrolladores del software compartir una misma línea de trabajo y cubrir todos los objetivos y restricciones de la aplicación

Es considerada el nivel más alto en el diseño de la arquitectura de un sistema puesto que establecen la estructura, funcionamiento e interacción entre las partes del software



#### Arquitectura de Software

La arquitectura de software forma la **columna vertebral** para construir un sistema de software, es en gran medida responsable de permitir o no ciertos atributos de calidad del sistema entre los que se destacan la **confiabilidad** y el **rendimiento** del software. Además es un modelo abstracto reutilizable que puede transferirse de un sistema a otro y que representa un medio de comunicación y discusión entre participantes del proyecto, permitiendo así la interacción e intercambio entre los desarrolladores con el objetivo final de establecer el intercambio de conocimientos y puntos de vista entre ellos.

- Método y Notación
  - Una notación de diseño de software es un medio para describir un diseño de software, ya sea Gráfica, textualmente, o ambas
  - Los diagramas de clases son una notación gráfica de diseño, y e pseudocódigo es una notación de diseño textual
  - UML es una notación gráfica para aplicaciones orientadas a objetos
  - Un concepto de diseño de software es una idea fundamental que se puede aplicar al diseño de un sistema
  - Una estrategia de diseño de software es un plan general y una dirección para desarrollar un diseño

- Método y Notación
  - Los **criterios de estructuración de software** son guías utilizadas para ayudar a un diseñador a estructurar un sistema de software en sus componentes.
  - Un método de diseño de software es un enfoque sistemático que describe la secuencia de pasos a seguir para crear un diseño, dados los requisitos de software de la aplicación. Ayuda al diseñador o al equipo de diseño a identificar las decisiones de diseño que se deben tomar, el orden en que se deben realizar y los criterios de estructuración que deben utilizarse para realizarlas
  - Un método de diseño se basa en un conjunto de conceptos de diseño, emplea una o más estrategias de diseño y documenta el diseño resultante, utilizando una notación de diseño.

 Metodología de modelado y diseño colaborativo para el desarrollo de software (COMET)

El método de modelado y diseño de objetos colaborativos, o COMET, utiliza la notación UML para describir el diseño.

COMET se basa en los conceptos de diseño de ocultar información, clases, herencia y tareas concurrentes

Utiliza una estrategia de diseño de diseño de objetos concurrentes, que aborda la estructuración de un sistema de software en objetos activos y pasivos y define las interfaces entre ellos

Proporciona criterios de estructuración para ayudar a estructurar el sistema en objetos durante el análisis y criterios adicionales para determinar los subsistemas y tareas concurrentes durante el diseño.



• El Estándar **UML** 

UML son las siglas de "Unified Modeling Language" o "Lenguaje Unificado de Modelado". Se trata de un estándar que se ha adoptado a nivel internacional por numerosos organismos y empresas para crear esquemas, diagramas y documentación relativa a los desarrollos de software (programas informáticos)



• El Estándar UML - ¿ Que es?

El término "lenguaje" ha generado bastante confusión respecto a lo que es UML. En realidad el término lenguaje quizás no es el más apropiado, ya que no es un lenguaje propiamente dicho, sino una serie de normas y estándares gráficos respecto a cómo se deben representar los esquemas relativos al software. Mucha gente piensa por confusión que UML es un lenguaje de programación y esta idea es errónea: UML no es un lenguaje de programación. Como decimos, UML son una serie de normas y estándares que dicen cómo se debe representar algo.



• El Estándar UML - ¿ Que es?

Yo contestaría que en un **set de diagramas** que nos permiten describir nuestro software con mayor o menor detalle y que sirve como herramienta de comunicación en nuestros equipos.

- El Estándar UML ¿Cuántos Diagramas hay?
  - Diagramas de estructura
    - Diagrama de clases: Describe los diferentes tipos de objetos en un sistema y las relaciones existentes entre ellos. Dentro de las clases muestra las propiedades y operaciones, así como las restricciones de las conexiones entre objetos.
    - **Diagrama de objetos**: (También llamado Diagrama de instancias) Foto de los objetos en un sistema en un momento del tiempo.
    - **Diagrama de paquetes**: Muestra la estructura y dependencia entre paquetes, los cuales permiten agrupar elementos (no solamente clases) para la descripción de grandes sistemas.
    - Diagrama de despliegue: Muestra la relación entre componentes o subsistemas software y el hardware donde se despliega o instala.
    - **Diagrama de estructura compuesta**: Descompone jerárquicamente una clase mostrando su estructura interna.
    - **Diagrama de componentes**: Muestra la jerarquía y relaciones entre componentes de un sistema software.

• El Estándar UML - ¿Cuántos Diagramas hay?



- Diagrama de casos de uso: Permite capturar los requerimientos funcionales de un sistema.
- **Diagrama de estado**: Permite mostrar el comportamiento de un objeto a lo largo de su vida.
- Diagrama de actividad: Describe la lógica de un procedimiento, un proceso de negocio o workflow.
- Diagramas de interacción: Subgrupo dentro de los diagramas de comportamiento): Describen cómo los grupos de objetos colaboran para producir un comportamiento
  - Diagrama de secuencia: Muestra los mensajes que son pasados entre objetos en un escenario.
  - Diagrama de comunicación: Muestra las interacciones entre los participantes haciendo énfasis en la secuencia de mensajes.
  - Diagrama de (visión de conjunto o resumen de) interacción: Se trata de mostrar de forma conjunta diagramas de actividad y diagramas de secuencia.
  - **Diagrama de tiempo:** Pone el foco en las restricciones temporales de un objeto o un conjunto de objetos.
  - **Diagrama de colaboración:** (Solamente en UML 1.X) Muestra las interacciones organizadas alrededor de los roles.

- Evolución del modelado de software y métodos de diseño
  - En la década de 1960, los programas se implementaban a menudo con poco o ningún análisis sistemático de requerimientos y diseño (diagramas de flujo)
  - Las subrutinas se crearon originalmente como un medio de permitir que un bloque de código sea compartido llamándolo desde diferentes partes de un programa
  - Pronto fueron reconocidos como un medio para construir sistemas modulares y fueron adoptados como una herramienta de gestión de proyectos. Un programa podría dividirse en módulos, donde cada módulo podría ser desarrollado por una persona separada e implementado como una subrutina o función
  - A mediados de la década de 1970, dos estrategias de diseño de software diferentes ganaron prominencia: flujo de datos de diseño orientado y diseño estructurado de datos.
  - Hubo una maduración general de los métodos de diseño de software en la década de 1980, y se introdujeron varios métodos de diseño de sistemas.
  - A mediados y finales de los ochenta, la popularidad y el éxito de la programación orientada a objetos condujeron a la aparición de varios métodos de diseño orientados a objetos. El énfasis en estos métodos estaba en modelar el dominio del problema, el ocultar de la información, y la herencia



## Preguntas

