

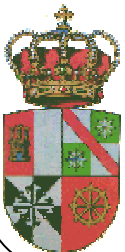
# Capítulo 1

## El diseño de software

---

Diseño y Programación Orientado a Objetos  
Ingeniería Informática  
Ingeniería Técnica de Informática de Sistemas y Gestión  
Optativa (6 créditos)

<http://www.info-ab.uclm.es/asignaturas/42579>



# Palabras clave del capítulo

---

- Extensibilidad
- Reutilización
- Modularidad
- Facilidad de mantenimiento
- Factores de calidad del software internos y externos
- Paradigmas de programación



# Bibliografía de consulta

---

- Construcción de software orientado a objetos (Segunda edición)  
Bertrand Meyer.  
Prentice Hall.
- Ingeniería del software. Un enfoque práctico (Quinta edición)  
Roger S. Pressman.  
McGraw Hill.
- Piensa en Java (Segunda edición)  
Bruce Eckel.  
Prentice Hall.



# Objetivos del capítulo

---

- Resaltar la importancia del diseño
- Presentar factores de calidad del software
- Presentar directrices y conceptos de diseño
- Características del software
- Pasos hacia la programación orientada a objetos



# Introducción (i)

---

- El objetivo general de la Ingeniería del Software es producir **software de calidad**
- Por **calidad** se entiende la adecuación del software a los requisitos exigidos
- El camino para obtener software de calidad es mediante un planteamiento riguroso del problema
- El **proceso de desarrollo** de software es aquel en el que las necesidades del usuario son traducidas en requisitos de software, estos transformados en **diseño** y el diseño implementado en código



# Introducción (y ii)

---

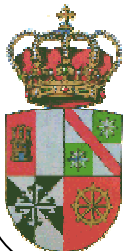
- El proceso de desarrollo de software requiere por un lado un conjunto de conceptos, una metodología y un lenguaje propio. A este proceso también se le denomina **ciclo de vida del software**

Análisis

Diseño

Implementación

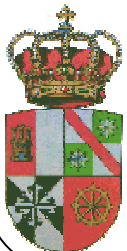
Instalación



# La calidad y sus factores

---

- Pueden distinguirse dos tipos de factores de calidad:
  - Los **factores de calidad externos** son aquellos que son perceptibles por los usuarios
    - Corrección
    - Robustez
    - Extensibilidad
    - Reutilización
    - Compatibilidad
    - Eficiencia
    - Portabilidad
    - Facilidad de uso
    - Funcionalidad
  - Los **factores de calidad internos** son los perceptibles por los profesionales en computación
    - Modularidad
    - Legibilidad



# Factores de calidad. Definiciones (i)

---

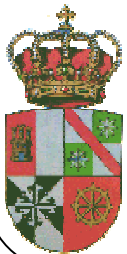
- **Corrección**
  - Habilidad de un sistema o producto software para desempeñar las funciones, exactamente como le fueron definidas en los requisitos y especificaciones
- **Solidez o robustez**
  - Habilidad para funcionar aún en condiciones anormales, es decir, con aquellos casos no explicitados en las especificaciones. Si se presentan el sistema termina “limpiamente”
- **Confiabilidad** = Corrección + Robustez



# Factores de calidad. Definiciones (ii)

---

- **Extensibilidad**
  - Facilidad para adaptarse a los cambios en las especificaciones. Se logra haciendo simples los diseños de los módulos autónomos
- **Reutilización**
  - Habilidad para utilizar de nuevo productos de software completos o partes de ellos en nuevas aplicaciones
- **Compatibilidad**
  - Facilidad con la que un producto software puede combinarse con otros. Se logra homogeneidad en el diseño y estandarización en la comunicación entre programas



# Factores de calidad. Definiciones (iii)

---

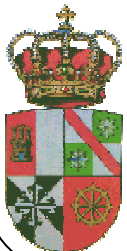
- **Eficiencia**
  - Facilidad de utilizar el mínimo de recursos de cómputo para conseguir mayor rapidez y menor necesidad de almacenamiento
- **Portabilidad**
  - Facilidad de transferir productos a diferentes plataformas
- **Facilidad de uso**
  - Facilidad con la que personas con diferentes niveles de experiencia pueden aprender a usar los productos software y aplicarlos a resolver problemas



# Factores de calidad. Definiciones (y iv)

---

- **Funcionalidad**
  - Conjunto de posibilidades ofrecido por un sistema
- Otros factores
  - Oportunidad
  - Economía
  - Integridad
  - Facilidad de reparación
  - Facilidad de verificación
  - ...



# ¿Qué es diseño? (i)

---

- El proceso de aplicar distintas técnicas y principios con el propósito de definir un producto con los suficientes detalles como para permitir su realización física.
- Con el diseño se pretende construir un sistema que:
  - Satisfaga determinada especificación del sistema
  - Se ajuste a las limitaciones impuestas por el medio de destino
  - Respete requisitos sobre forma, rendimiento utilización de recursos, coste, etc.



# ¿Qué es diseño? (y ii)

---

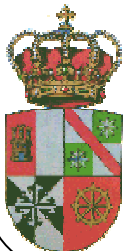
- El diseño es la primera etapa técnica del proceso de Ingeniería del Software, consiste en producir un modelo o representación técnica del software que se va a desarrollar
- El diseño es el proceso sobre el que se asienta la calidad del software
- El diseño de software es un proceso iterativo a través del cual se traducen los requisitos en una representación del software
- El diseño se representa a un alto nivel de abstracción, un nivel que se puede seguir hasta requisitos específicos de datos, funcionales y de comportamiento



# Metodologías de diseño

---

- Diseño de **datos**:
  - Modelo de información a estructuras de datos
- Diseño **arquitectónico**:
  - Define las relaciones entre los elementos estructurales de nuestro programa
- Diseño **procedimental**:
  - Se transforman los elementos estructurales de nuestro programa en una descripción procedimental del software
- Diseño de **interfaz**:
  - Describe cómo se comunica el software consigo mismo y con su entorno



# Directrices para un buen diseño

---

- El diseño debe implementar todos los **requisitos explícitos** contenidos en el modelo de análisis y debe acomodar todos los **requisitos implícitos** que desee el cliente
- El diseño debe ser una guía que puedan leer y entender los que construyan el código y los que prueban y mantienen el software
- El diseño debería proporcionar una completa idea de lo que es el software, enfocando los dominios de datos, funcional y de comportamiento desde la perspectiva de la implementación



# Principios básicos de diseño

---

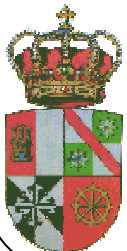
- El diseñador debe considerar **enfoques alternativos** juzgando a cada uno en base a los requisitos del problema, los resultados disponibles y los criterios de calidad interna
- Se deberían poder seguir los pasos de diseño hasta el modelo de análisis
- El diseño no va a reinventar nada que ya esté inventado
- El diseño debería presentar **uniformidad** e **integración**
- Debe estructurarse para admitir cambios
- El diseño no es escribir código y escribir código no es diseñar
- Se debería valorar la calidad del diseño mientras se crea, no después de terminado



# Conceptos del diseño (i)

---

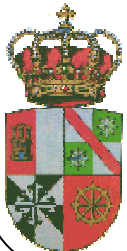
- A la hora de realizar el diseño del software debes plantearte
  - ¿Qué criterios se pueden usar para dividir el software en componentes individuales?
  - ¿Cómo se separan los detalles de una función o de la estructura de datos de la representación conceptual del software?
  - ¿Existen criterios uniformes que definan la calidad técnica de un diseño de programas?



# Conceptos del diseño (ii)

---

- Para ayudar a resolver las anteriores preguntas se han establecido unos conceptos fundamentales del diseño del software
  - Abstracción
  - Refinamiento
  - Modularidad
  - Arquitectura del software
  - Jerarquía de control
  - Partición estructural (horizontal y vertical)
  - Estructura de datos
  - Procedimientos
  - Ocultamiento de la información



# Conceptos del diseño (y iv)

---

- El **diseño modular efectivo** reduce la complejidad, facilita los cambios y produce como resultado una implementación más sencilla, permitiendo el desarrollo paralelo de las diferentes partes del sistema
- La **independencia funcional** se adquiere desarrollando módulos con una clara función evitando una excesiva interacción con otros módulos. Este concepto está derivado de la modularidad, la abstracción y el ocultamiento de la información



# Conceptos del diseño (iii)

---

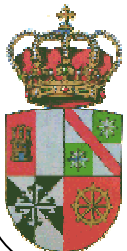
- La independencia funcional se mide con dos criterios:
  - **Cohesión**: extensión del concepto de ocultamiento de información. Un módulo cohesivo ejecuta una tarea sencilla de un procedimiento de software y requiere poca interacción con procedimientos que ejecutan otras partes de un programa
  - **Acoplamiento**: medida de la interconexión entre módulos de un programa
- Interesa una cohesión alta y un acoplamiento bajo



# Modularidad

---

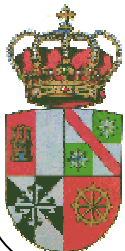
- **Criterios**
  - Descomposición modular
  - Composición modular
  - Comprensibilidad modular
  - Continuidad modular
  - Protección modular
- **Reglas**
  - Correspondencia directa
  - Pocas interfaces
  - Interfaces pequeñas
  - Interfaces explicitas
  - Ocultación de la información
- **Principios**
  - Unidades modulares lingüísticas
  - Auto-documentación
  - Acceso uniforme
  - Principio de abierto-cerrado
  - Elección única



# Programación Orientada a Objetos

---

- Como paradigma es una filosofía de la que surge una cultura nueva que incorpora técnicas y metodologías diferentes
- En ella el universo computacional está poblado por objetos, cada uno responsable de si mismo, y comunicándose con los demás por medio de mensajes. Cada objeto representa una instancia de alguna clase, y estas clases son miembros de una jerarquía de clases unidas vía relaciones de herencia
- La diferencia entre un objeto y una clase es que un objeto es una entidad concreta que existe en tiempo y espacio, mientras que una clase representa una abstracción, la esencia de un objeto.



# Principios del modelo orientado a objetos

---

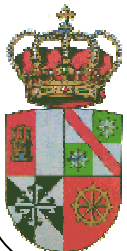
- **Abstracción**
  - Descripción simplificada del sistema que resalta unos detalles y suprime otros
- **Encapsulación**
  - Ocultar detalles de un objeto que no contribuyen a sus características esenciales
- **Modularidad**
  - Propiedad de un sistema que ha sido descompuesto en un conjunto de módulos coherentes e independientes
- **Jerarquía o herencia**
  - Abstracciones organizadas por niveles



# Evolución histórica

---

- Programación estructurada
- Programación modular
- Programación orientada a objetos
- Programación orientada a aspectos
- Programación orientada a componentes



# Principios de construcción

---

*FUNCTION* busquedaColección(x: elemento, t:colección): *BOOLEAN*

*VAR*

pos: posición;

Variación de tipos

*BEGIN*

Variación en implementación

pos = **Comenzar**(x, t);

*WHILE* not **Completa**(pos, t) and not **Encontrado**(pos, t) *DO*

pos = **Siguiente**(pos, x, t);

busquedaColección = not **Completa**(pos, t);

*END*;

Independencia de la representación

Agrupación de rutinas

Factorizar los comportamientos comunes



# Resumen

---

- El **diseño** es el proceso sobre el que se asienta la calidad del software
- Por **calidad** se entiende la adecuación del software a los requisitos exigidos
- El **diseño modular efectivo** reduce la complejidad, reduce la complejidad, facilita los cambios y produce como resultado una implementación más sencilla
- Los principios del modelo orientado a objetos son: **abstracción, encapsulación, modularidad y jerarquía**

