

INGENIERÍA DEL SOFTWARE II

TEMA 8 REUTILIZACIÓN

REUTILIZACIÓN

- El Campo de la Reutilización
- Marcos de Trabajo de Aplicación (Frameworks)
- Líneas de Productos Software
- Reutilización de Productos COTS
(Commercially available Off-The-Shelf, sistemas comerciales)

Reutilización de Software

- En la mayoría de las disciplinas de la ingeniería los sistemas se diseñan por acoplando **componentes existentes** que han sido usados en otros sistemas
- La Ingeniería del Software se ha centrado más en el **desarrollo original** pero ahora se reconoce que para **conseguir mejor software**, más rápidamente y a menos coste, necesitamos un proceso de diseño basado en la **reutilización sistemática** de software
- Se ha producido un **importante cambio** hacia el desarrollo basado en la reutilización en los últimos 15 años

Ingeniería de Software Basada en la Reutilización

- **Reutilización del sistema de aplicación**
 - El total de una sistema de aplicación puede ser reutilizado incorporándolo sin cambios en otros sistemas (reutilización de COTS) o desarrollando familias de aplicaciones
- **Reutilización de componentes**
 - Componente de una aplicación desde subsistemas hasta objetos simples pueden ser reutilizados
- **Reutilización de objetos y funciones**
 - Componentes software que implementen un objeto único o función bien definidos

Beneficios de la Reutilización de Software

- **Incremento de la confiabilidad:** el software reutilizado ha sido probado y usado en sistemas en funcionamiento, sus fallos en el diseño y la implementación ya habrán sido encontrados y reparados
- **Reducción del riesgo del proceso:** el coste del software existente ya es conocido, mientras que los costes de desarrollo son siempre cuestionables; reduce el margen de error, particularmente cuando se trata de reutilización de componentes relativamente grandes
- **Uso efectivo de especialistas:** en vez de hacer lo mismo una y otra vez, los especialistas de aplicación pueden desarrollar software reusable que encapsula su conocimiento

Beneficios de la Reutilización de Software

- **Cumplimiento de estándares:** pueden ser implementados como un conjunto de componentes reusables, como los estándares de interfaz, si los menús presentan los mismos formatos a los usuarios se cometerán menos errores en su manejo
- **Desarrollo acelerado:** sacar un sistema al mercado cuanto antes es a menudo más importante que los costes totales de desarrollo; se puede sacar antes el producto porque tanto el tiempo de desarrollo como el de validación se reduce

Problemas con la Reutilización

- **Incremento de los costes de mantenimiento:** si el código fuente de los componentes reutilizados no está disponible, dichos componentes pueden ser cada vez más incompatibles con los cambios del sistema
- **Falta de soporte de herramientas:** el proceso software asumido por estas herramientas puede no tener en cuenta la reutilización, pueden ser difíciles o imposibles de integrar con un sistema de librerías de componentes
- **Síndrome “reinventar la rueda”:** algunos ingenieros prefieren reescribir componentes porque creen que pueden mejorarlos, esto tiene que ver con la confianza y también con el hecho de que escribir software original es más desafiante que reutilizar el software de otros

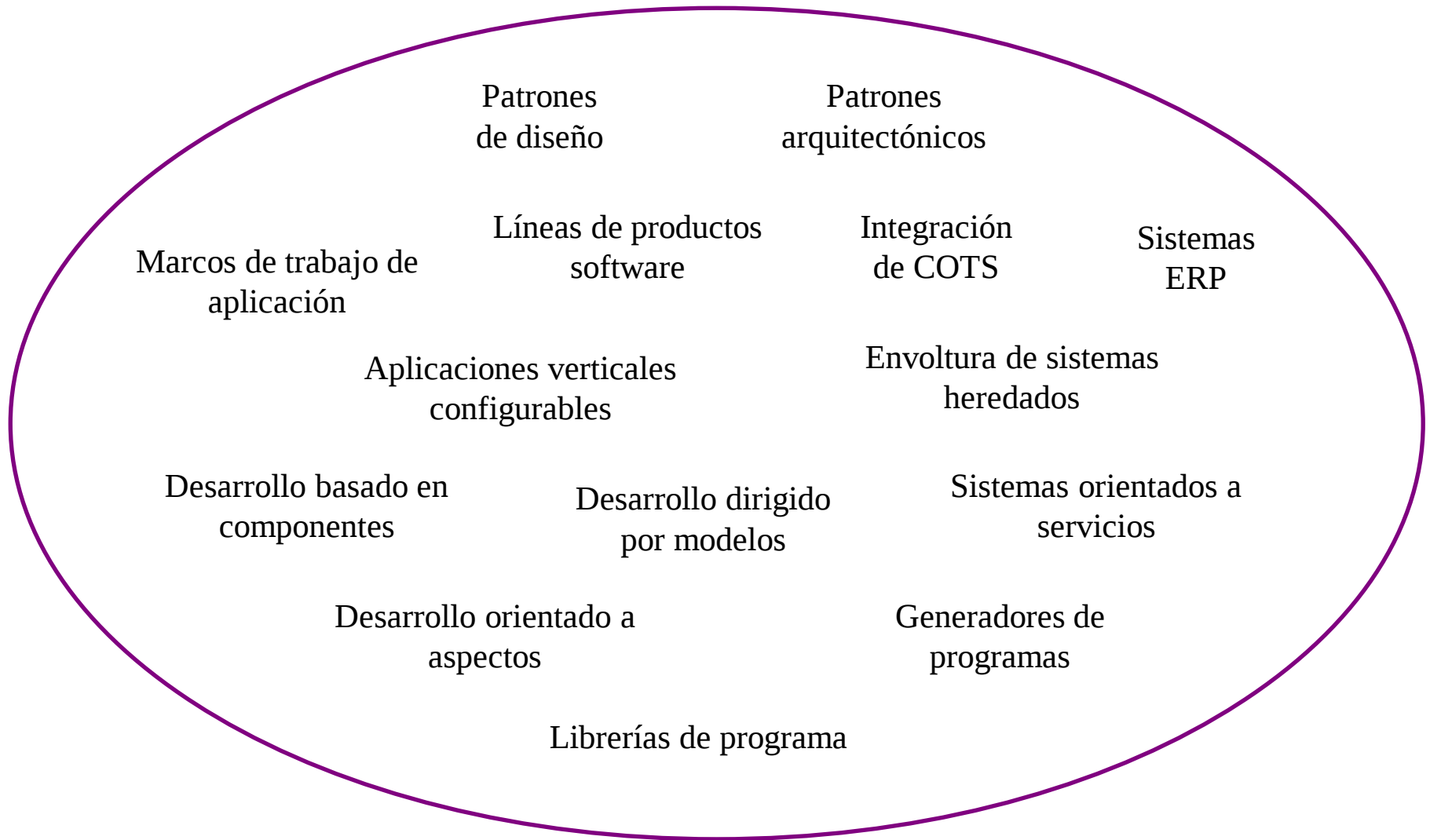
Problemas con la Reutilización

- **Coste de la creación, mantenimiento y utilización de una librería de componentes:** poblar dicha librería y asegurar que puede ser usada por los desarrolladores puede ser **caro**, los procesos software deben ser adaptados
- **Confianza en encontrar, comprender y adaptar componentes reutilizables:** hay que encontrar los componentes que nos pueden ser útiles, comprenderlos y a veces adaptarlos al trabajo en un nuevo entorno, los ingenieros deben estar razonablemente confiados de encontrar el componente antes de incluir dicha búsqueda como una parte de su **proceso normal** de desarrollo

El Campo de la Reutilización

- Aunque la reutilización es a menudo pensada solamente como la reutilización de componentes de sistema, **hay muchas aproximaciones diferentes**
- La reutilización es posible a un rango de niveles desde funciones simples a aplicaciones de sistemas completas
- El campo de la reutilización cubre el **rango** de las técnicas de reutilización posibles

El Campo de la Reutilización



Factores Clave de la Reutilización

- El **calendario** de desarrollo para el software
- La **vida** esperada del software
- Los antecedentes, habilidades y experiencia del **equipo** de desarrollo
- La **criticidad** del software y sus requerimientos no funcionales
- El **dominio** de aplicación
- La **plataforma** en la que operará el sistema

Frameworks de Aplicación

- Los **frameworks** son entidades moderadamente grandes que pueden ser reutilizados, están a medias entre la reutilización de sistema y componentes
- Son un diseño de un subsistema formado por una **colección** de clases abstractas y concretas y las interfaces entre ellas
- El subsistema **se implementa** añadiendo componentes para rellenar partes del diseño e instanciando las clases abstractas del framework

Clases de Framework

- De **infraestructura** de sistema
 - Apoyan el desarrollo de infraestructuras de sistema como comunicaciones, interfaces de usuario y compiladores
- De integración de **middleware**
 - Estándares y clases que soportan comunicación de componentes e intercambio de información
- De aplicación **empresarial**
 - Se ocupan de dominios específicos de aplicaciones como los sistemas de telecomunicaciones o financieros

Frameworks de Aplicación Web (WAF)

- Apoyan la construcción de páginas web dinámicas como una **plantilla para la GUI** para aplicaciones web
- Están disponibles para **todos** los lenguajes de programación de uso común como Java, Python, Ruby, etc.
- El modelo de interacción está basado en el **patrón MVC** (Modelo-Vista-Controlador)

Modelo-Vista-Controlador

- Framework de infraestructura de sistema para diseño de la **GUI**
- Permite las **presentaciones múltiples** de información de un objeto y separar las interacciones con estas presentaciones
- El framework MVC incluye la instanciación de un número de patrones (Observador, Composite, Estrategia...)

Características de los WAF

- **Seguridad:** pueden incluir clases para ayudar a implementar la autenticación de usuario (login) y el control de acceso
- **Páginas Web dinámicas:** se ofrecen clases para ayudar a definir las plantillas de la página web y alimentar a éstas dinámicamente a partir de la base de datos del sistema
- **Soporte de bases de datos:** puede proporcionar clases que ofrezcan una interfaz abstracta a diferentes bases de datos
- **Gestión de sesión:** por lo general son parte de un WAF clases para crear y administrar sesiones (un conjunto de interacciones con el sistema por parte de un usuario)
- **Interacción de Usuarios:** la mayoría brindan soporte AJAX, que permite la creación de páginas web más interactivas

Extender un Framework

- Los frameworks son genéricos y se extienden para crear una aplicación o subsistema más específicos, proporcionan una arquitectura **esqueleto** para el sistema
- Extender implica
 - **Añadir clases concretas** que heredan operaciones de las clases abstractas del framework
 - **Añadir métodos** que son llamadas como respuesta a eventos reconocidos por el framework
- Los problemas de los frameworks son su **complejidad** que supone necesitar mucho **tiempo** para usarlos efectivamente

Líneas de productos software

- Las líneas de productos software o familias de aplicación son aplicaciones con **funcionalidad genérica** que puede ser adaptada y configurada para usarse en un contexto específico
- Una línea de productos software es un conjunto de aplicaciones con una **arquitectura común** y componentes compartidos, con cada aplicación se especializa para reflejar diferentes requisitos
- La **adaptación** puede implicar
 - Configuración de componentes y del sistema
 - Añadir nuevos componentes al sistema
 - Seleccionar en una librería de componentes existentes
 - Modificar componentes para cumplir con nuevos requisitos

Frameworks de Aplicación y las Líneas de Productos

- Los frameworks se apoyan en características **orientadas a objetos** como polimorfismo y herencia para implementar extensiones, las líneas de productos **no requieren** un enfoque orientado a objetos
- Los frameworks se centran en proporcionar **soporte técnico** más que específico del dominio, las líneas de productos lleva **incrustada información** del dominio y la plataforma
- Las líneas de producto a menudo son aplicaciones de **control** para equipos
- Las líneas de producto software están constituidas por una familia de aplicaciones, normalmente propiedad de una **misma organización**

Especialización de una Línea de Productos

- **De plataforma**
 - Se elaboran diferentes versiones para diferentes plataformas, diferentes sistemas operativos
- **De entorno**
 - Se crean diferentes versiones de la aplicación para manejar diferentes entornos operativos como diferentes tipos de equipamientos de comunicación
- **Funcional**
 - Se construyen diferentes versiones para clientes con requisitos diferentes
- **De proceso**
 - Diferentes versiones para soportar diferentes procesos de negocio

Arquitecturas de Líneas de Productos

- Deben estar estructuradas de tal modo que se **separen** los diferentes subsistemas y se permita que sean modificados
- También debería **separar entidades** y sus **descripciones** y en los niveles más altos del sistema acceder a las entidades a través de sus descripciones mejor que directamente
- Es frecuente que estas arquitecturas se estructuren **en capas**

Reutilización de productos COTS

- Un producto commercial-off-the-shelf (COTS) es un sistema software que puede adaptarse a las necesidades del cliente **sin cambiar el código fuente**
- Los sistemas COTS tienen **características genéricas** y por lo tanto pueden ser utilizados y reutilizados en diferentes entornos
- Se adaptan utilizando usando mecanismos de **configuración internos** que permiten adecuar la funcionalidad a cliente específico
 - Por ejemplo, en sistema de registro de pacientes de un hospital, pueden definirse formularios de entrada e informes de salida distintos para diferentes tipos de pacientes

Beneficios de la Reutilización de COTS

- Como con otros tipos de reutilización, puede ser posible un desarrollo **más rápido** de un sistema fiable
- Es posible ver que funcionalidad proporcionan las aplicaciones y por lo tanto es más **fácil juzgar** si es probable que sean adecuadas o no
- Los negocios pueden **centrarse** en su actividad principal sin tener que dedicar un montón de recursos al desarrollo de sistemas de TI
- Al evolucionar las plataformas operativas, pueden **simplificarse las actualizaciones** tecnológicas ya que éstas son más responsabilidad de los proveedores de los productos COTS que del cliente

Problemas de la Reutilización de COTS

- Los **requisitos** normalmente tienen que **adaptarse** para reflejar la **funcionalidad** y **modo de operación** de los productos COTS
- Los COTS pueden basarse en **suposiciones** que son prácticamente imposible de cambiar
- Elegir el COTS apropiado para una empresa puede ser **difícil**, especialmente por causa de que muchos **no** están bien documentados
- Puede faltar **experiencia local** para apoyar el desarrollo de los sistemas
- Los proveedores de COTS **controlan** el soporte y evolución del sistema

Sistemas de solución COTS

- Son aplicaciones genéricas que pueden diseñarse para dar soporte a un **tipo particular de negocio**, **actividad empresarial** o incluso a **toda una empresa**
 - Por ejemplo, se puede producir una solución COTS para dentistas que maneje citas, registros dentales, programas revisiones para pacientes, etc.
 - A mayor escala un sistema de planeación de recursos empresariales (**ERP**) puede apoyar todas las actividades de una compañía grande
- Los COTS de **dominio específico**, como los sistemas que soportan una función de negocio, proporcionan la funcionalidad que es probable que sea requerida por sus usuarios potenciales

Sistemas ERP

- Un sistema de planeación de recursos empresariales (ERP, Enterprise Resource Planning system) es un sistema genérico que apoya **procesos de negocio comunes** como fabricación, pedidos, facturación y servicio de atención al cliente
- Son ampliamente utilizados en grandes compañías, representan probablemente la forma **más común** de reutilización de software
- El núcleo genérico se adapta incluyendo módulos e incorporando conocimiento de los **procesos** y **reglas** de negocio

Bibliografía

- *Ian Sommerville*, “*Ingeniería de Software*,” *Pearson*, *Novena Edición*, 2011