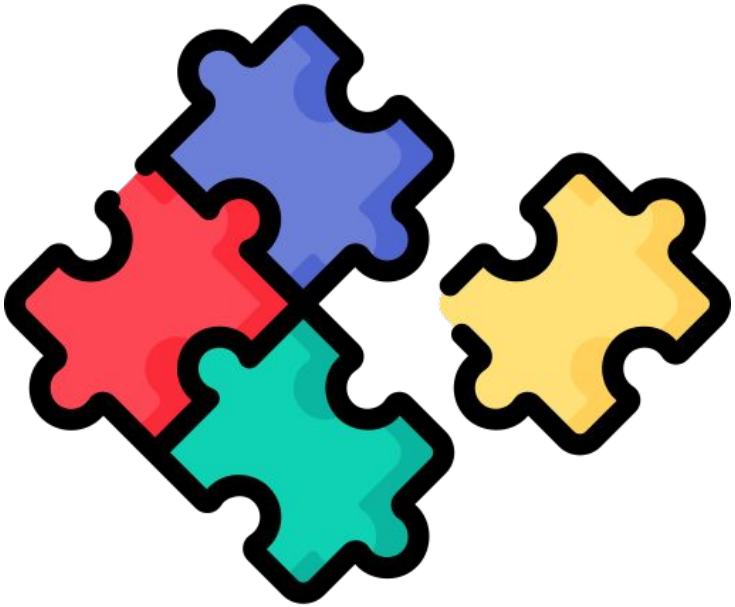
Reutilización de Componentes

Fausto Panello 2025

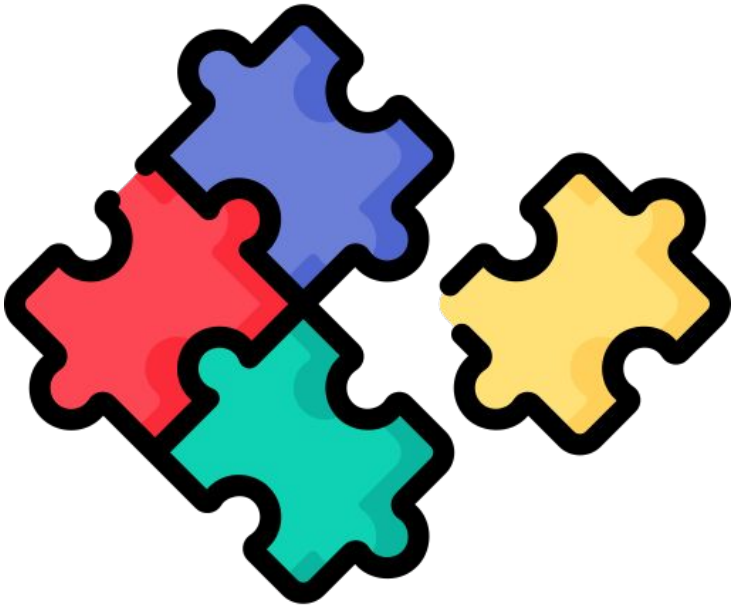
Modelo de Reutilización de Componentes

Es el proceso de creación de sistemas de software **a partir de un software ya existente**, para de ese modo evitar un diseño de sistema desde el principio. En los años 60, se construyeron bibliotecas de subrutinas científicas reutilizables con un dominio de aplicación limitado, en la actualidad se crean componentes comunes a varios procesos con el fin de poder utilizarlos en la construcción de software.



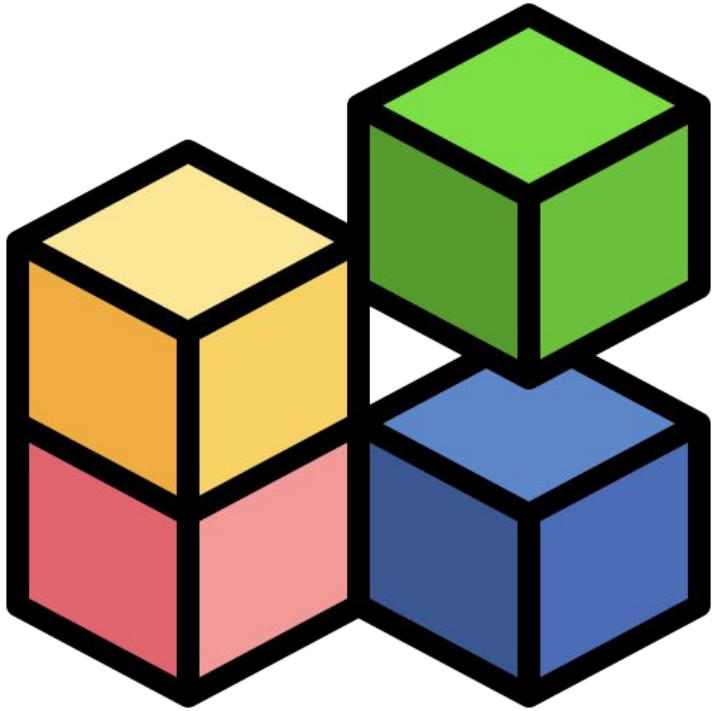
Modelo de Reutilización de Componentes

Podemos definirla como el empleo de elementos de software u otros de nivel superior, creados en desarrollos anteriores, para de este modo reducir los tiempos y simplificar el desarrollo del software, mejorando la calidad y reduciendo su costo. Un componente de software individual es un paquete de software, un servicio web API por ejemplo, o un módulo que encapsula un conjunto de funciones relacionadas (o de datos).



Modelo de Reutilización de Componentes

Todos los procesos del sistema son colocados en componentes separados de tal manera que todos los datos y funciones dentro de cada componente están semánticamente relacionados. Debido a este principio, con frecuencia se dice que los componentes son **modulares** y **cohesivos**.

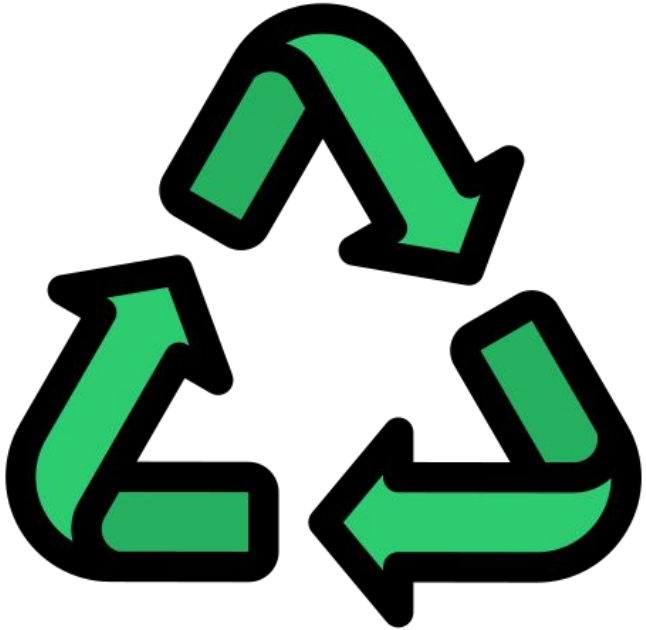


Modelo de Reutilización de Componentes

Los componentes se comunican uno con el otro por medio de interfaces (de entrada y salida). Cuando un componente ofrece servicios al resto del sistema, este adopta una **interfaz** proporcionada que especifica los servicios que otros componentes pueden utilizar, y cómo pueden hacerlo. Esta interfaz puede ser vista como una **firma** del componente. El cliente no necesita saber sobre los funcionamientos internos del componente (su implementación) para hacer uso de ella. Este principio resulta en componentes referidos como **encapsulados**.

Elementos que intervienen en la reutilización

- Especificaciones de requerimientos previamente concebidas
- Diseños previamente definidos (Estructuras de datos, algoritmos, etc.)
- Código probado y depurado con anterioridad
- Planes y casos de prueba previamente utilizados
- Personal calificado (aprovechamiento de la experiencia de los ingenieros de un proyecto a otro)
- Paquetes de software de propósito general.



Conceptos de reutilización de software

- La reutilización de software aparece como una alternativa para desarrollar aplicaciones y sistemas de una manera más eficiente, productiva y rápida.
- La idea es reutilizar elementos y componentes en lugar de tener que desarrollarlos desde un principio.
- Surge formalmente en 1968.
- La idea principal era producir componentes de software como si de componentes eléctricos se tratara.
- El objetivo es reutilizar lo existente sin tener que volver a rediseñarlo desde el principio.

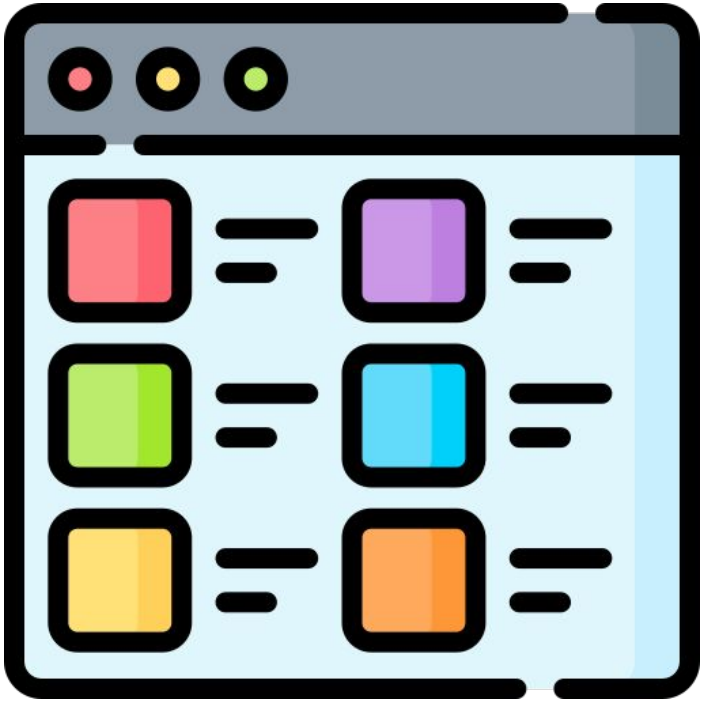
Dificultades al aplicar reutilización

- En muchas empresas no existe plan de reutilización (No se considera prioritario).
- Escasa formación (y personal técnico disponible) en dicho campo.
- Pobre soporte metodológico (los componentes son paquetes habitualmente cerrados y el soporte metodológico puede ser complejo o incompleto).
- Uso de métodos que no promueven la reutilización (Estructurados).

Categorías de recurso de software

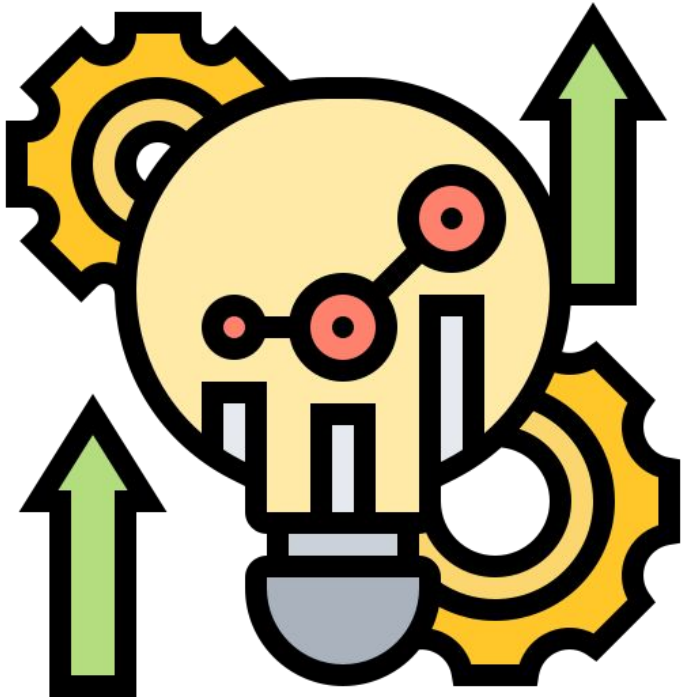
Hay diferentes categorías de recurso de software a reutilizar:

- Componentes ya desarrollados
- Componentes ya experimentados
- Componentes con experiencia parcial
- Componentes nuevos



Componentes ya desarrollados

El software existente se puede adquirir de una tercera parte o provenir de uno desarrollado internamente para un proyecto anterior. Llamados componentes CCYD (componentes comercialmente ya desarrollados), estos componentes están listos para utilizarse en el proyecto actual y se han validado totalmente.



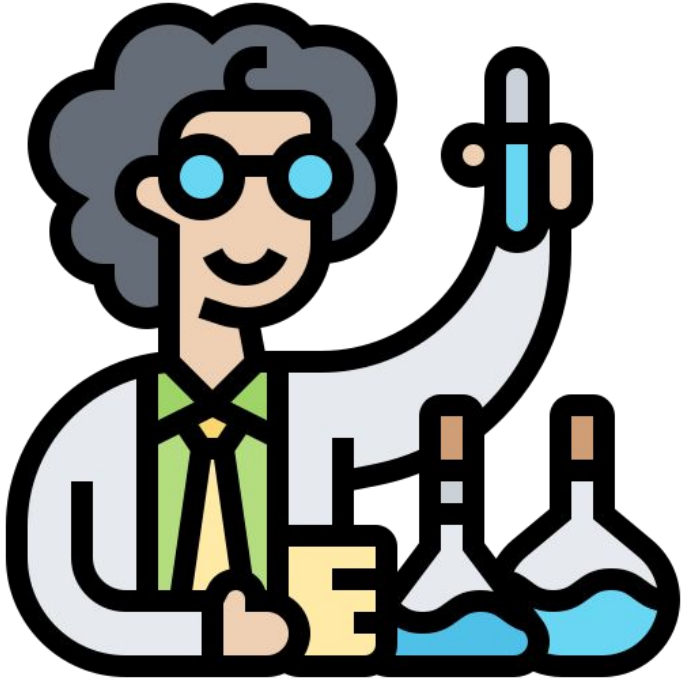
Ejemplos de componentes ya desarrollados

Biblioteca de gráficos 3D: Un proyecto de desarrollo de un videojuego puede reutilizar una biblioteca de gráficos 3D existente, como Unity o Unreal Engine, para crear los efectos visuales y la renderización del juego.

Sistema de autenticación: Un proyecto de desarrollo web puede utilizar un componente de software de autenticación ya desarrollado, como OAuth o Firebase Authentication, para gestionar la autenticación de usuarios en la aplicación.

Componentes ya experimentados

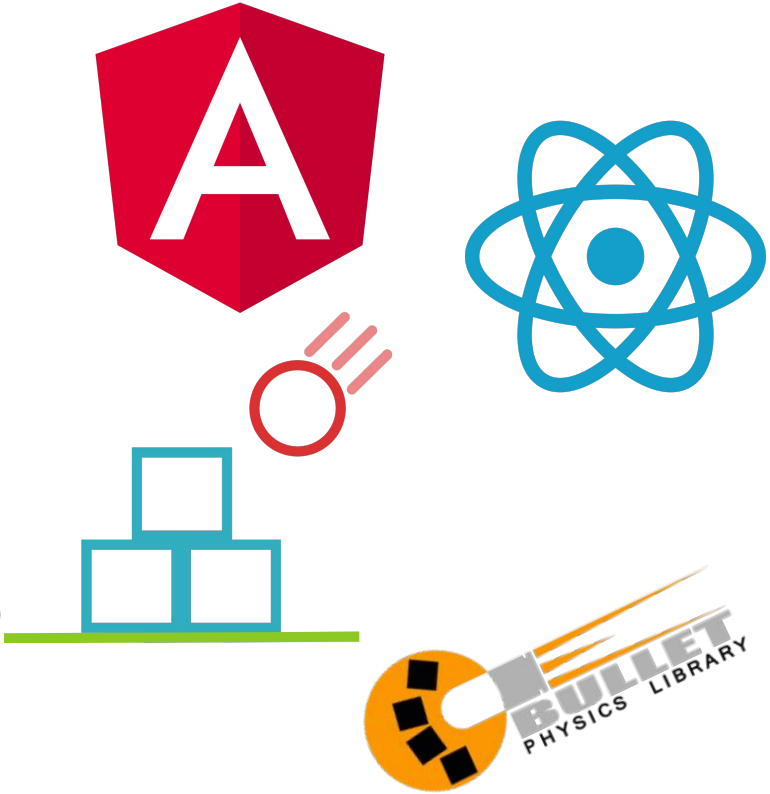
Especificaciones, diseños, código o datos de prueba existentes desarrollados para proyectos anteriores que son similares al software que se va a construir para el proyecto actual. Los miembros del equipo de software actual ya han tenido la experiencia completa en el área de la aplicación representada para estos componentes. Las modificaciones, por tanto, requeridas para componentes de total experiencia, tendrán un riesgo relativamente bajo.



Ejemplos de componentes ya experimentados

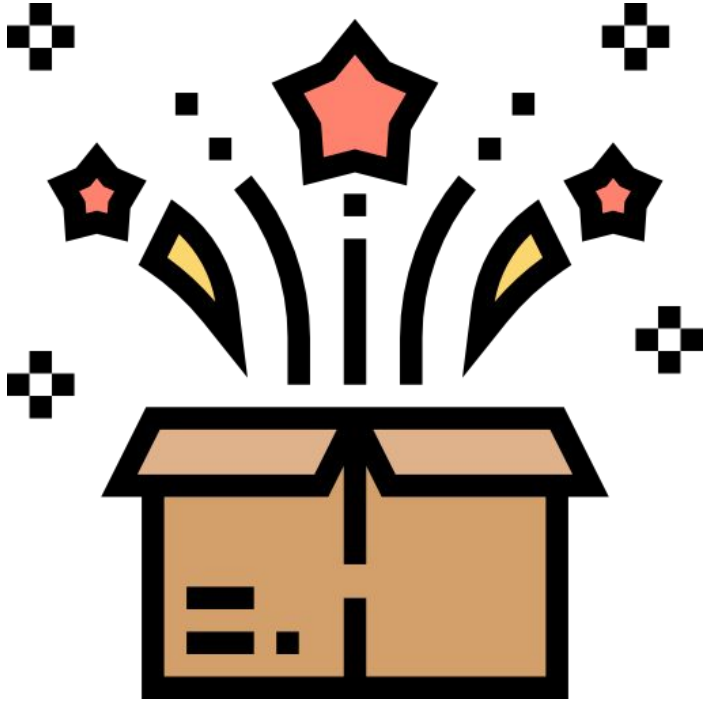
Framework de desarrollo web: Un equipo de desarrollo web que ha utilizado anteriormente un framework como Angular o React puede reutilizar el código, las especificaciones y los diseños para desarrollar un nuevo proyecto web con características similares.

Motor de física: Un proyecto de simulación de física puede reutilizar un motor de física experimentado, como Box2D o Bullet Physics, que ya ha sido utilizado por el equipo para proyectos anteriores con éxito.



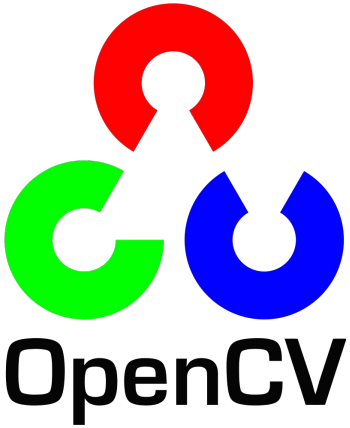
Componentes con experiencia parcial

Especificaciones, diseños, código o datos de prueba existentes desarrollados para proyectos anteriores que se relacionan con el software que se va a construir para el proyecto actual, pero que requerirán una **modificación sustancial.** Los miembros del equipo de software actual han limitado su experiencia sólo al área de aplicación representada por estos componentes. Las modificaciones, por tanto, requeridas para componentes de experiencia parcial tendrán bastante grado de riesgo.

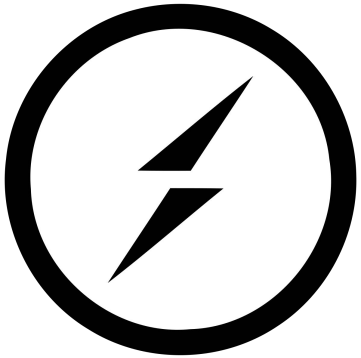


Ejemplos de componentes con experiencia parcial

Biblioteca de procesamiento de imágenes: Un equipo de desarrollo que ha trabajado previamente en el procesamiento de imágenes puede reutilizar una biblioteca existente, como OpenCV, para realizar ciertas operaciones de procesamiento de imágenes, pero necesitará modificarla para adaptarla a los requisitos específicos del proyecto actual.



Componente de comunicación en tiempo real: Un proyecto de desarrollo de aplicaciones de chat puede reutilizar un componente de comunicación en tiempo real existente, como Socket.IO, pero deberá realizar modificaciones sustanciales para adaptarlo a las necesidades específicas del proyecto.



Componentes nuevos

Los componentes de software que el equipo de software **debe construir específicamente** para las necesidades del proyecto actual. Ejemplos:

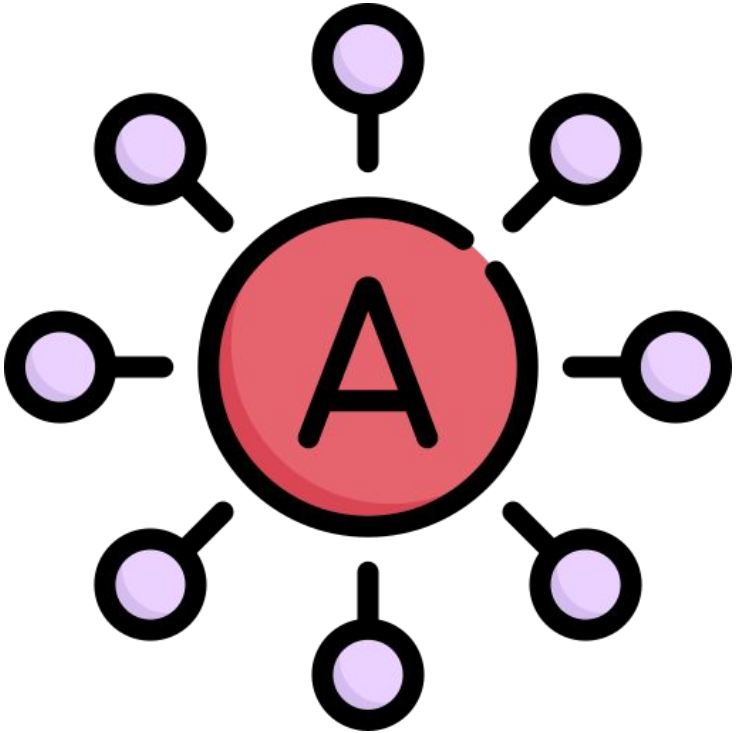
Motor de recomendaciones personalizadas: Un proyecto de desarrollo de una plataforma de comercio electrónico puede requerir la construcción de un motor de recomendaciones personalizadas que tenga en cuenta el historial de compras y preferencias del usuario para ofrecer recomendaciones relevantes.

Algoritmo de procesamiento de señales: Un proyecto de desarrollo de software de procesamiento de señales de audio puede requerir la creación de un algoritmo personalizado para el análisis y la manipulación de las señales de audio de entrada.

Sistema de análisis de sentimientos en redes sociales: Para un proyecto de análisis de datos en redes sociales, se puede construir un componente nuevo que utilice técnicas de procesamiento de lenguaje natural y aprendizaje automático para analizar los sentimientos expresados en los mensajes de los usuarios. Este componente extraería información valiosa sobre las opiniones y emociones de los usuarios en relación con ciertos temas y proporcionaría análisis detallados. Requeriría el diseño e implementación de algoritmos de análisis de sentimientos, procesamiento de texto y visualización de datos.

Tipos de reutilización

- **Oportunista**: El ingeniero de software reutiliza piezas que él sabe que se ajustan al problema.
- **Sistemática**: Esfuerzo a nivel organizacional y planificado de antemano. Todo componente reutilizado ha de ser ideado, a priori, para ser reutilizado. Implica inversiones iniciales para recoger frutos en el futuro. Diseñar componentes genéricos para que sean reutilizados con facilidad.
- **Bottom-Up**: Se desarrollan pequeños componentes para una determinada aplicación. Se incorpora a un repositorio.
- **Top-Down**: Se determinan las piezas necesarias que encajan unas con otras. Se van desarrollando poco a poco. Requiere alta inversión al comienzo. Se recogerán beneficios en el futuro. Análisis de escenarios para la reutilización.



¿Cuándo voy a requerir elementos de reutilización?

Existen al menos 4 escenarios en los que un proyecto de software requerirá elementos de reutilización:

1. El proyecto es similar a uno anterior (reutilización de un proyecto existente).
2. Mismo proyecto con configuración diferente (reutilizan productos actuales).
3. Características de uso basado en productos existentes.
4. Nueva arquitectura con capacidades o elementos existentes.

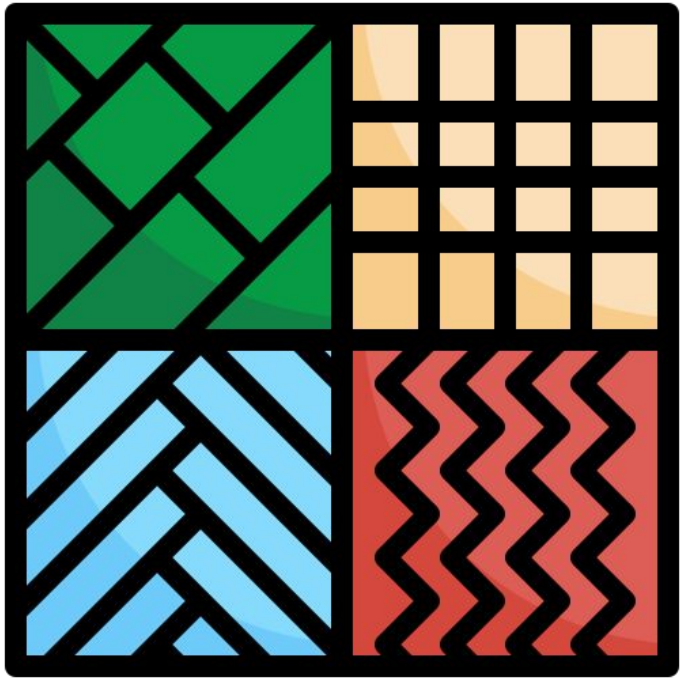
Hilton
HOTELS & RESORTS

Walmart
Save money. Live better.

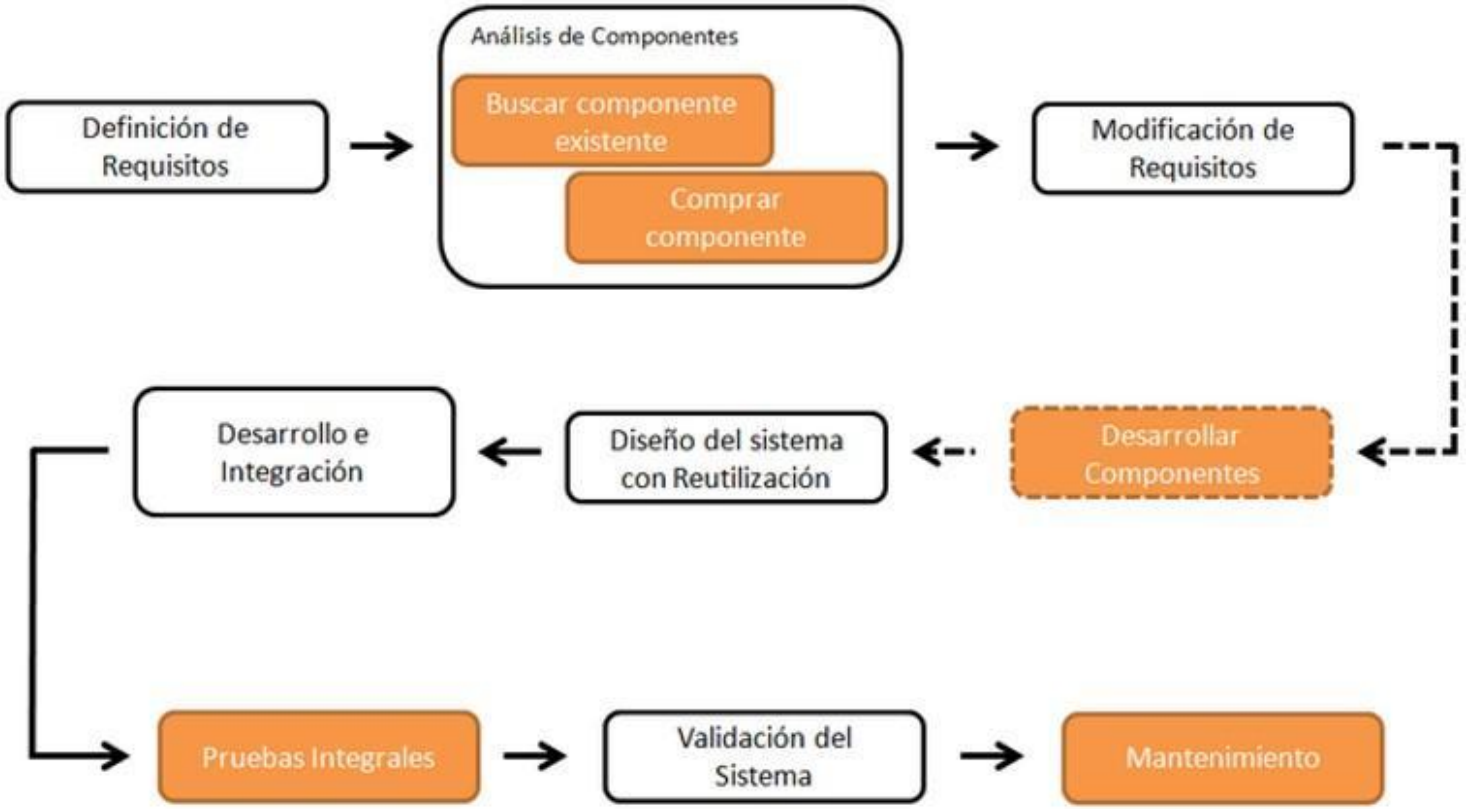
Patrones de diseño

Los patrones y los lenguajes de patrones son formas de describir las mejores prácticas, buenos diseños y encapsulan la experiencia de tal forma que es posible para otros reutilizar dicha experiencia. Algunos elementos esenciales del patrón de diseño pueden ser:

- Nombre que es una referencia significativa del patrón.
- Una descripción del área del problema que explica cuándo puede aplicarse el patrón
- Descripción de las partes de la solución del diseño, sus relaciones y sus responsabilidades.
- Una declaración de las consecuencias de aplicar el patrón.



Representación gráfica del ciclo de vida



Ventajas y desventajas

Ventajas

- Reduce el tiempo de desarrollo.
- Reduce los costos.
- No es necesario reinventar las soluciones.

Desventajas

- Necesidad de invertir antes de obtener resultados (se puede hacer entregas luego de la puesta en marcha completa del sistema).
- Carencia de métodos adecuados (dependen y se amoldan a los módulos).
- Necesidad de formar al personal por la complejidad de implementar el sistema sobre módulos preexistentes.