Desarrollo de Software

CC-3S2

César Lara Avila

Universidad Nacional de Ingeniería

Bienvenidos

Temario

- Arquitectura de Software
- ¿Por qué es importante la arquitectura?
- Diseño arquitectónico
- Elección de tecnologías

Arquitectura de Software

Para crear un producto confiable, seguro y eficiente, debes prestar atención al diseño arquitectónico que incluye:

- Su organización general,
- Cómo se descompone el software en componentes,
- La organización del servidor
- Las tecnologías que utiliza para construir el software.

Hay muchas interpretaciones diferentes del término *arquitectura de software*.

- Algunos se enfocan en **arquitectura** como un sustantivo - la estructura de un sistema

- Otros consideran que **arquitectura** es un verbo - el proceso de definir estas estructuras.

La definición IEEE de arquitectura de software

La arquitectura es la organización fundamental de un sistema de software incorporada en sus componentes, sus relaciones entre sí y con el entorno y los principios que guían su diseño y evolución.

Arquitectura de software y componentes

- Un componente es un elemento que implementa un conjunto coherente de funcionalidades o características.
- El componente de software se puede considerar como una colección de uno o más servicios que pueden ser utilizados por otros componentes.
- Al diseñar la arquitectura de software, no tienes que decidir cómo se implementará un elemento o componente arquitectónico.
- Tu diseñas la interfaz del componente y deja la implementación de esa interfaz para una etapa posterior del proceso de desarrollo.

Atributos no funcionales de calidad

Capacidad de respuesta: ¿el sistema devuelve resultados a los usuarios en un tiempo razonable?

Confiabilidad:¿las funciones del sistema se comportan como esperan tanto los desarrolladores como los usuarios?

Disponibilidad: ¿puede el sistema prestar sus servicios cuando lo solicitan los usuarios?

Seguridad:¿el sistema se protege a sí mismo y a los datos de los usuarios de ataques e intrusiones no autorizadas?

Usabilidad:¿pueden los usuarios del sistema acceder a las funciones que necesitan y usarlas rápidamente y sin errores?

Mantenibilidad: ¿se puede actualizar fácilmente el sistema y agregar nuevas características sin costos indebidos?

Resiliencia: ¿puede el sistema continuar prestando servicios al usuario en caso de falla parcial o ataque externo?.

¿Por qué es importante la arquitectura?

La arquitectura es importante porque tiene una influencia fundamental en propiedades no funcionales.

El diseño arquitectónico implica comprender los problemas que afectan la arquitectura de tu producto y crear una descripción arquitectónica que muestre los componentes críticos y sus relaciones.

Minimizar la complejidad debería ser un objetivo importante para los diseñadores de arquitectura.

- Cuanto más complejo es un sistema, más difícil y costoso es entenderlo y cambiarlo.
- Los programadores son más propensos a cometer errores e introducir errores y vulnerabilidades de seguridad cuando modifican o amplían un sistema complejo.

La influencia en la arquitectura de la seguridad del sistema.

Una arquitectura de seguridad centralizada

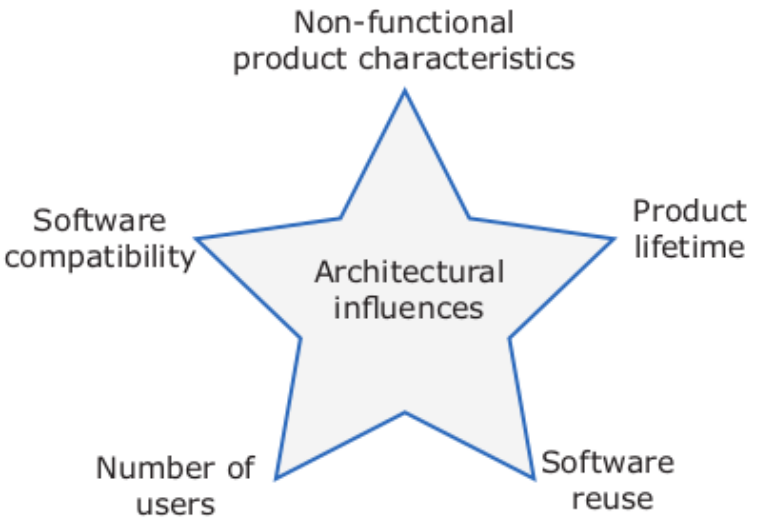
En la precuela de Star Wars, Rogue One (https://en.wikipedia.org/wiki/Rogue_One), el malvado Imperio ha almacenado los planos de todo su equipo en una sola ubicación remota, altamente segura y bien protegida. Esto se denomina arquitectura de seguridad centralizada. Se basa en el principio de que si mantienes toda tu información en un solo lugar, puedes aplicar muchos recursos para proteger esa información y asegurarse de que los intrusos no puedan obtenerla.

Desafortunadamente (para el Imperio), los rebeldes lograron violar su seguridad. Robaron los planos de la Estrella de la Muerte, un evento que sustenta toda la saga de Star Wars. Al tratar de detenerlos, el Imperio destruyó todo su archivo de documentación del sistema con quién sabe qué costos resultantes. Si el Imperio hubiera elegido una arquitectura de seguridad distribuida, con diferentes partes de los planos de la Estrella de la Muerte almacenados en diferentes ubicaciones, robar los planos habría sido más difícil. Los rebeldes habrían tenido que romper la seguridad en todos los lugares para robar los planos completos de la Estrella de la Muerte.

Diseño arquitectónico

El diseño arquitectónico implica comprender crear una descripción de la arquitectura que muestre los componentes críticos y algunas de sus relaciones.

Los problemas de arquitectura que son más importantes para el desarrollo de productos de software.



Debido a que es imposible optimizar todo, debes hacer una serie de concesiones al elegir una arquitectura para tu sistema.

Algunos ejemplos son:

- Mantenibilidad frente a rendimiento
- Seguridad frente a usabilidad
- Disponibilidad frente al tiempo de comercialización y costo.

Mantenibilidad vs rendimiento

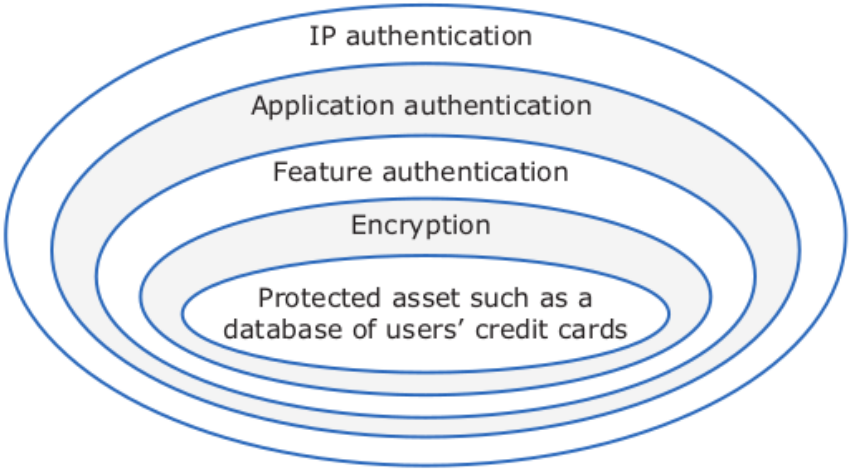
La mantenibilidad del sistema es un atributo que refleja lo difícil y costoso que es realizar cambios en un sistema después de que se haya lanzado a los clientes.

En términos arquitectónicos, esto significa que el sistema debe descomponerse en componentes de grano fino, cada uno de los cuales hace una cosa y sólo una cosa.

Si muchos componentes están involucrados en la implementación de una característica del producto, el software será más lento.

Seguridad vs usabilidad

Puedes lograr la seguridad diseñando la protección del sistema como una serie de capas.



Un atacante tiene que penetrar todas esas capas antes de que el sistema se vea comprometido.

Disponibilidad vs. tiempo de comercialización

La disponibilidad de un sistema es una medida de la cantidad de **tiempo de actividad** de ese sistema.

Arquitectónicamente, logras disponibilidad al tener componentes **redundantes** en un sistema.

Para hacer uso de la redundancia, incluye componentes que detectan fallas y componentes de conmutación que cambian la operación a un componente redundante cuando se detecta una falla.

Preguntas de diseño arquitectónico

¿Cómo debe organizarse el sistema como un conjunto de componentes arquitectónicos, donde cada uno de estos componentes proporciona un subconjunto de la funcionalidad general del sistema?.

¿Cómo deberían distribuirse y comunicarse entre sí estos componentes arquitectónicos?.

¿Qué tecnologías se debe usar para construir el sistema y qué componentes deben reutilizarse?.

Organización de componentes

Abstracción en el diseño de software significa que tu te enfocas en los elementos esenciales de un sistema o componente de software sin preocuparte por sus detalles.

A nivel arquitectónico, tu interés debe ser los componentes arquitectónicos a gran escala.

La descomposición implica analizar estos componentes a gran escala y representarlos como un conjunto de componentes de grano más fino.

Complejidad arquitectónica

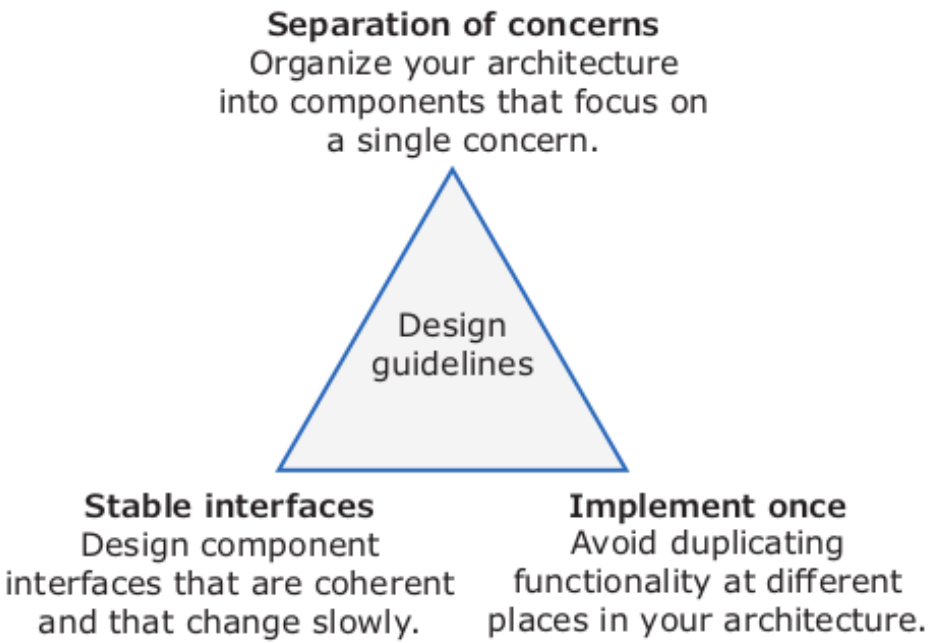
La complejidad en la arquitectura de un sistema surge debido al número y la naturaleza de las relaciones entre los componentes de ese sistema.

Al descomponer un sistema en componentes, debes intentar evitar la complejidad innecesaria del software.

- Localizar relaciones
- Reducir las dependencias compartidas

Siempre es preferible utilizar datos locales siempre que sea posible y evitar compartir datos si es posible.

Pautas de diseño arquitectónico



Referencias: Engineering Software Products Global Edition, 1st edition-Ian Sommerville.

Preocupaciones transversales

Son preocupaciones que son sistémicas, es decir, afectan a todo el sistema.

Las preocupaciones transversales son completamente diferentes de las preocupaciones funcionales representadas por capas en una arquitectura de software.

Cada capa debe tenerlos en cuenta e inevitablemente hay interacciones entre las capas debido a estas preocupaciones.

La existencia de preocupaciones transversales es la razón por la que modificar un sistema después de haber sido diseñado para mejorar su seguridad suele ser difícil.

La seguridad como preocupación transversal

Si se utilizan diferentes tecnologías en diferentes capas, como una base de datos SQL o un navegador Firefox. Los atacantes pueden intentar utilizar las vulnerabilidades de estas tecnologías para obtener acceso.

En consecuencia, necesitan protección contra ataques en cada capa, así como protección, en las capas inferiores del sistema, contra ataques exitosos que hayan ocurrido en capas de nivel superior.

Al distribuir la seguridad entre las capas, tu sistema es más resistente a los ataques y fallas de software.

Problemas en la elección arquitectónica

Tipo de datos y actualizaciones de datos

Si utilizas principalmente datos estructurados que pueden ser actualizados por diferentes características del sistema, generalmente es mejor tener una única base de datos compartida que proporcione gestión de transacciones y bloqueo.

Cambiar la frecuencia

Si anticipas que los componentes del sistema se cambiarán o reemplazarán regularmente, entonces aislar estos componentes como servicios separados simplifica esos cambios.

La plataforma de ejecución del sistema

Si planeas ejecutar tu sistema en la nube con usuarios que acceden a este a través de Internet, generalmente es mejor implementarlo como una arquitectura orientada a servicios porque escalar el sistema es más simple.

Elección de tecnologías

Base de datos

¿Debes utilizar una base de datos SQL relacional o una base de datos NOSQL no estructurada?.

Plataforma

¿Debes entregar tu producto en una aplicación móvil y/o una plataforma web?.

Servidor

¿Debes utilizar servidores internos dedicados o diseñar tu sistema para que se ejecute en una nube pública? Si es una nube pública, ¿deberías usar Amazon, Google, Microsoft o alguna otra opción?.

Código abierto

¿Existen componentes de código abierto adecuados que podrías incorporar en tus productos?.

Herramientas de desarrollo

¿Tus herramientas de desarrollo incorporan suposiciones arquitectónicas sobre el software que se está desarrollando que limitan tus opciones arquitectónicas?.

¿Preguntas?