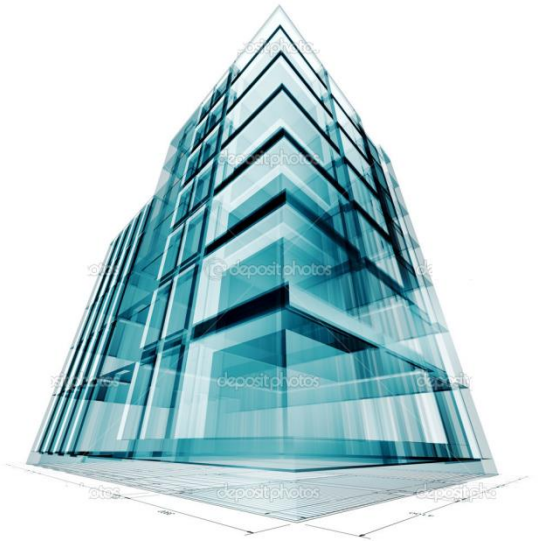
Arquitectura de Software

Clase 1:

Introducción a la Arquitectura de Software

Hugo R. Cordero



Objetivos

- Conocer lo que involucra la Arquitectura de Software y las preocupaciones que persigue organizar
- Entender la importancia de la Arquitectura de Software en el desarrollo de los sistemas actuales
- Conocer el perfil del profesional Arquitecto de Software

Temas

1. Antecedentes
2. ¿Qué es la Arquitectura de Software?
3. Diferencia entre Arquitectura y tecnología
4. El rol del Arquitecto de Software

Antecedentes

- Las Tecnologías de la Información (TI) son la aplicación de computadores y equipos de telecomunicación para almacenar, recuperar, transmitir y manipular datos, en el contexto de las negocios.
- Los sistemas de TI están en todos lados:
  - Bancos
  - Comercios
  - Servicios
  - Militares, Juegos, …
- Amplio uso comercial de plataformas de base de datos, servidores web, paquetes de aplicaciones, etc.
- Diferentes tipos: ERP, CRM, SCM, DSS, GIS, Field Service, etc. pueden ser grandes, complejos, heterogéneos, distribuidos
- Los mayores problemas están en el diseño, selección de la tecnología, integración con otras aplicaciones y rápida adecuación al negocio

Antecedentes

- También están los problemas identificados por los profesionales e investigadores de software:
  - Características particulares para la programación en conjunto
  - Necesidad de reutilización del software
  - Dificultades propias de la Ingeniería de software

  Muchas ideas se originaron en otros dominios (de negocio), no informáticos.

- Surge la Arquitectura de Software, pero en realidad los ingenieros de software siempre han empleado arquitecturas de software, muy a menudo sin darse cuenta
- La arquitectura se enfoca en los problemas que serán difíciles/imposibles de cambiar cuando el sistema esté construido
  - Equivalencia: Arquitecto de edificaciones

¿Qué es la Arquitectura de Software?

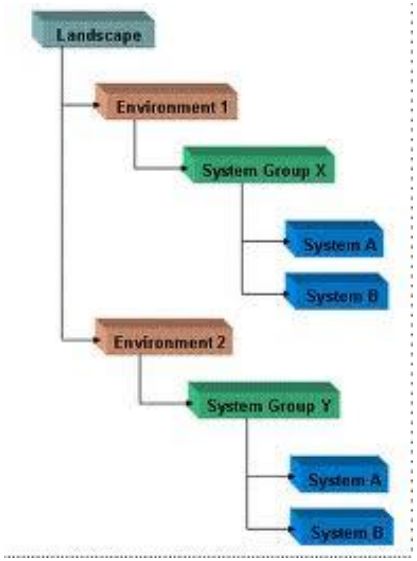
- Es hablar de diseño de software
  - Toda arquitectura es diseño de software, pero no todo diseño es una arquitectura de software
  - Es parte del proceso de diseño de software

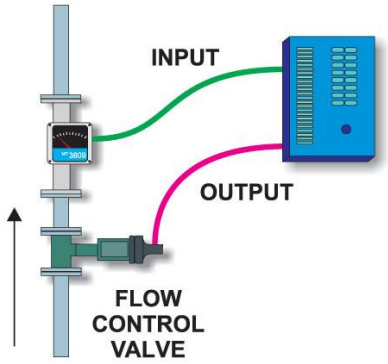
- Definiciones:
  - *"Arquitectura es la **organización fundamental** de un sistema, representado por sus **componentes**, sus **relaciones** entre ellos y con el entorno, y los principios que guían su diseño y evolución."* ANSI/IEEE Std 1471-2000
  - *"La arquitectura de software de un programa o un sistema computacional es la **estructura o estructuras** del sistema, la cual comprende **elementos de software**, las **propiedades externas visibles** de estos elementos, y las **relaciones** entre ellos."* Software Engineering Institute (SEI)

http://www.sei.cmu.edu/architecture/start/glossary/definition-form.cfm

La arquitectura define la estructura

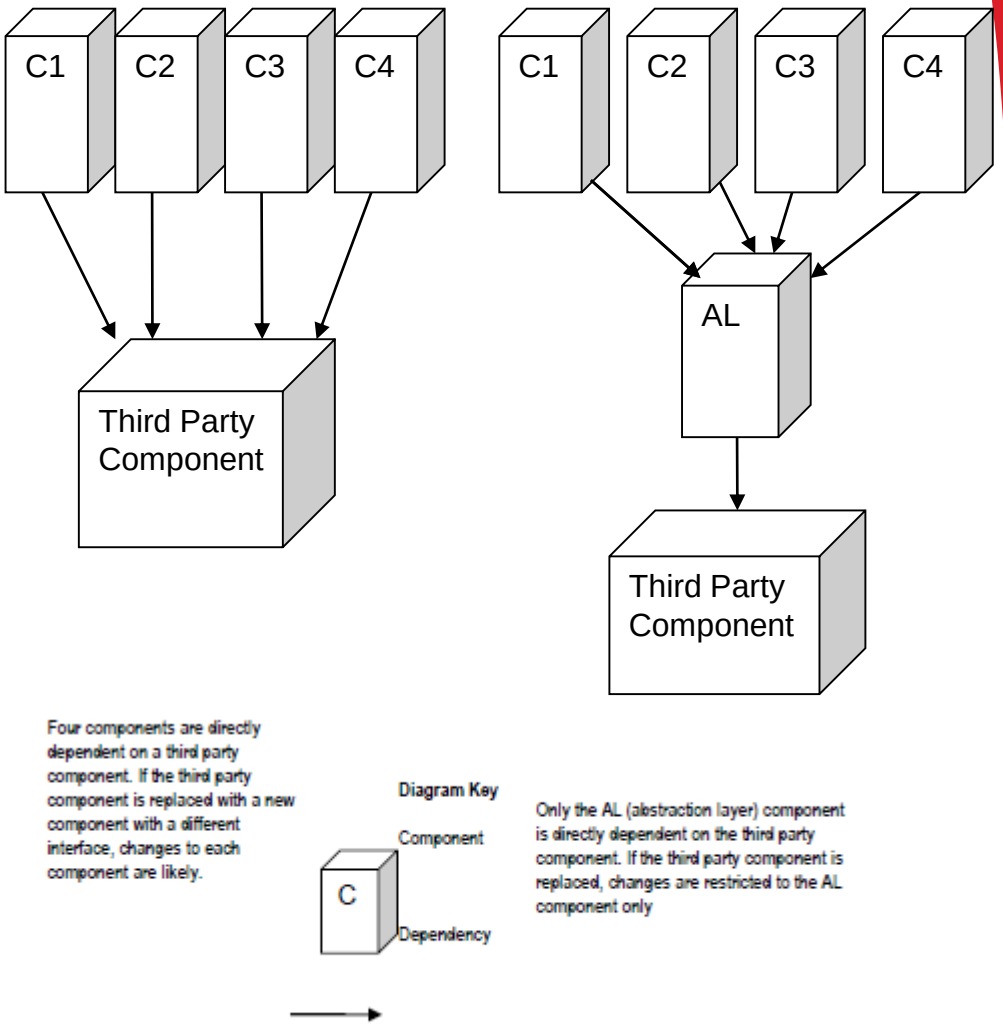
- Estructura: descomposición del sistema en: componentes/módulos/subsistemas
- La arquitectura define:
  - Interfaces de los componentes
  - Dependencias y comunicaciones de los componentes
  - Responsabilidades de los componentes
- Comunicación involucra:
  - Mecanismos de traspaso de datos:
    - Llamada de funciones
    - Invocación remota de métodos
    - Mensajes asincrónicos
  - Control del flujo:
    - Secuenciamiento
    - Concurrencia/Paralelismo
    - Sincronización





Estructura y Dependencias

- Estructurar un sistema permite identificar las relaciones y dependencias
- Excesiva dependencia de componentes no es bueno
- Principales problemas
  - Identificar componentes que pueden cambiar
  - Reducir la dependencia directa de estos componentes
- Crear sistemas más modificables
- Descomponer modularmente…



Utiliza la descomposición modular

- El módulo es una unidad funcional parte de un sistema
- Sistema

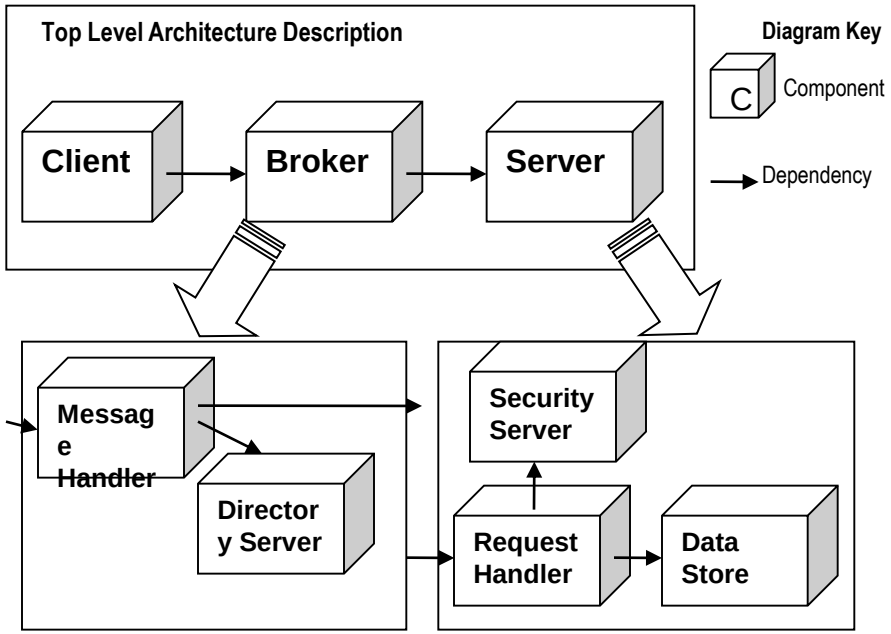
  -> Subsistema

  -> Módulo

  -> Componente

  -> Clase / Programa

  -> Método / función
- La descomposición jerárquica es un poderoso mecanismo de abstracción
  - Patrones de diseño
  - Asignar componentes para equipos de desarrollo



Se apoya en principios de diseño

- Un principio de diseño es una directriz recomendada para "dar forma" a solución con ciertas metas en mente
  - Divide y conquistarás
  - Reducir acoplamiento e incrementar la cohesión
  - Diseñar para reutilizar
  - Diseñar con reutilización
  - Diseñar pensando en portabilidad
  - Diseñar pensando en verificabilidad
- Los componentes deben:
  - No repetir funcionalidades
  - Única responsabilidad
  - Cerrado a las modificaciones y abierto a la extensibilidad
  - Trabajar con interfaces
  - Inversión de Control
  - Convenciones de configuración
  - Mantener segregadas las Interfaces

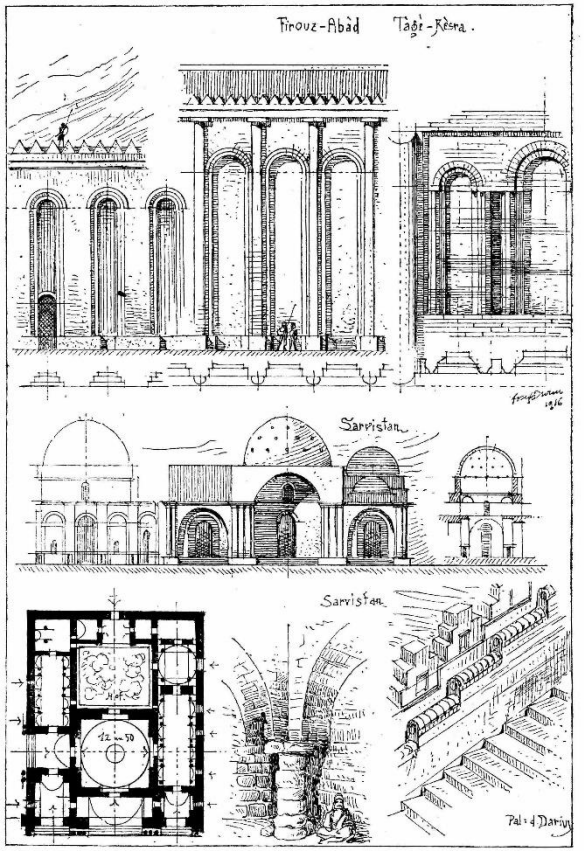
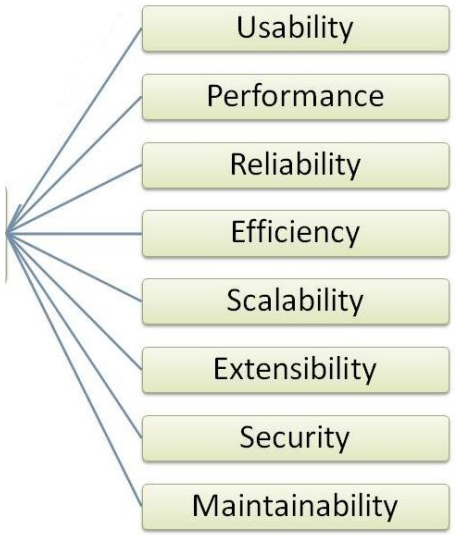


Fig. 18. – Palacios de Firouz-Abad y de Sarvistán

Direcciona a los RNFs

- Son los requerimientos no funcionales que definen “como” el sistema trabaja
- Son requerimientos normalmente “ignorados”
- RNFs son raramente incluidos en los requerimientos funcionales
  - Deben ser identificados por el arquitecto
- RNFs incluyen:
  - Restricciones técnicas:
    Plataforma pre-existentes, tecnologías específicas
  - Restricciones de negocio:
    Estándares corporativos, objetivos comerciales
  - Atributos de calidad:
    Rendimiento, escalabilidad, disponibilidad, usabilidad, eficiencia, seguridad, extensibilidad, mantenibilidad



Se complementa con Vistas

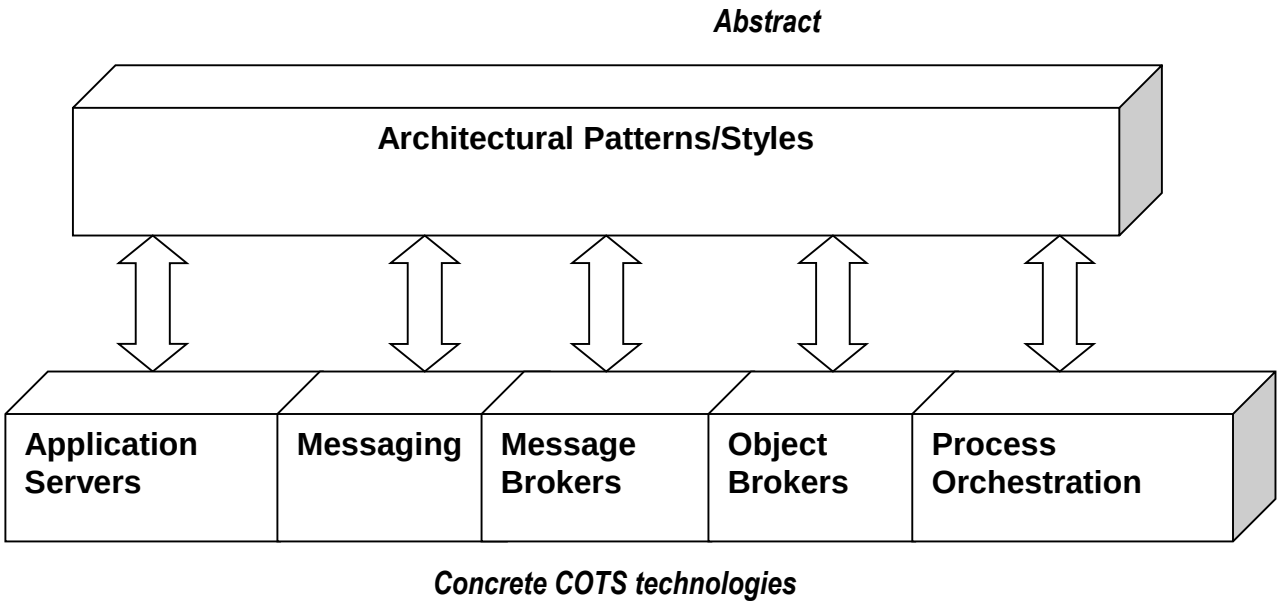
- Una arquitectura de software representa un complejo artefacto de diseño
- Hay muchas posibles vistas de la arquitectura
  - Como para la construcción de un edificio – diseño de planta, externo, plano eléctrico, plano de plomería, etc.
- Modelos de vistas 4+1, SEI, Siemens
  - Lógica, Física, de Procesos, de Despliegue o Localización, de Componentes y Conectores, de Integración, etc.
- Notaciones: UML y particulares

Se modela con Patrones

- Patrones catalogados como exitosos usan estructuras que facilitan ciertos tipos de comunicación de componentes:
  - Cliente-Servidor
  - Message broker
  - Pipeline
- Los patrones tienen características bien conocidas que son apropiadas para particulares tipos de requerimientos
- La arquitectura provee una vista abstracta de un diseño
  - Oculta la complejidad del diseño en detalle
  - Puede o no puede ser mapeada directamente entre elementos de la arquitectura o elementos de software
- Un Marketecture
  - Es una representación informal de la estructura e interacciones del sistema.

Se soporta en tecnologías

- La arquitectura reduce el riesgo por el uso de patrones de diseño probados
- Vendedores de software tienen creadas tecnologías que explícitamente soportan patrones ampliamente usados

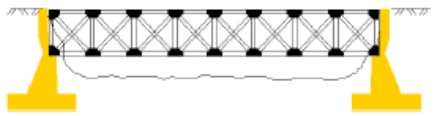


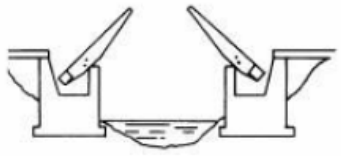
- Cada tecnología tiene múltiples vendedores/versiones open source
- Arquitectos necesitan seleccionar la tecnología sabiamente

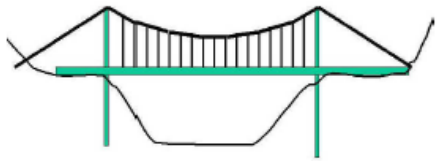
Arquitecturas y tecnologías

Diferentes estilos arquitectónicos

- De flujo de datos
  - *Tuberías y filtros (Pipeline)*
- De llamada y retorno
  - *Programa principal / subrutina*
  - *Basado en componentes*
  - *Cliente / Servidor*
  - *Arquitectura de capas*
  - *Modelo Vista Controlador*
- Entre Pares
  - *Broker*
  - *Orientado a servicios*







Arquitecturas y tecnologías

Diferentes tecnologías

- Plataformas de desarrollo: Java / .NET / PHP
- Middlewares
  - Servidores de aplicaciones, MOM, BPMs, Message Brokers
- Herramientas de desarrollo / pruebas
- Lenguajes de programación
- Gestores de versiones
- Gestores de contenidos y portales
- Base de datos
- Frameworks
- Inteligencia de Negocios / Analítica
- Etc.

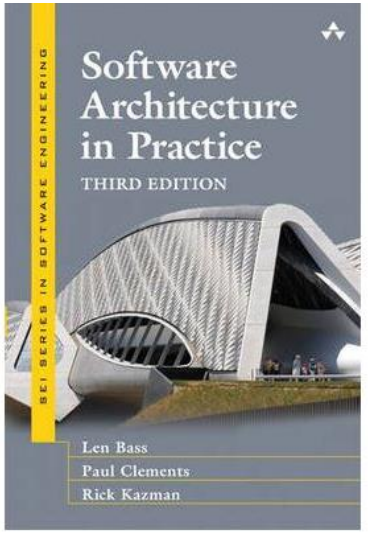
Lectura 1: Importancia de la Arquitectura

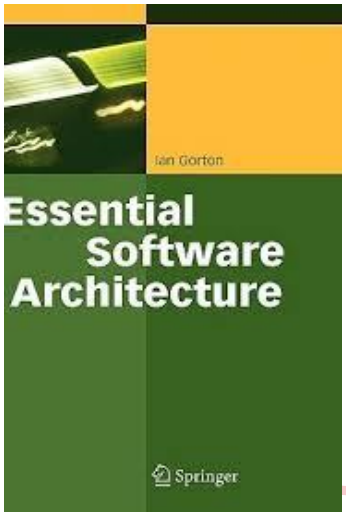
- Chapter 2: Why is Software Architecture Important?

  Páginas 25 – 58

  Libro: Software Architecture in Practice (3rd Edition)

  Autores: Len Bass, Paul Clements, Rick Kazman





- Chapter 1: Understanding software architecture

  Páginas 1 – 12

  Libro: Essential Software Architecture (2nd Edition)

  Autor: Ian Gorton

Contexto para la Arquitectura

- Un sistema siempre estará relacionado con el contexto que lo rodea
- Conjunto de objetos exteriores al sistema, pero que influyen decididamente a éste, y a su vez el sistema influye, aunque en una menor proporción, sobre el contexto
- Es una entrada necesaria para la definición de la Arquitectura del sistema
- El sistema a diseñar es visto como un todo y se ubica normalmente en el centro:

Práctica 1: Diagramas de Contexto

- Revise los casos propuestos de la práctica en clase 1 e inicie identificando los Stakeholders, para luego elaborar el Diagrama de contexto. Considere la recomendaciones dadas en clase para desarrollar el diagrama de contexto.

- Reúnanse en grupos y elabore la práctica en una hoja borrador para entregar al profesor.

- Tiempo para la práctica: 20 minutos.

El rol del arquitecto de software

- Distintos Stakeholders en el ciclo de desarrollo de software: analistas, diseñadores, programadores, testeadores, **Arquitectos,** gestores, usuarios, clientes, vendedores
- El rol del arquitecto de edificaciones y el arquitecto de software parecen enfrentar los mismos retos
- Debe contar, con un conjunto básico de habilidades y conocimientos para ejercer este tipo de roles
- Y es que no es lo mismo

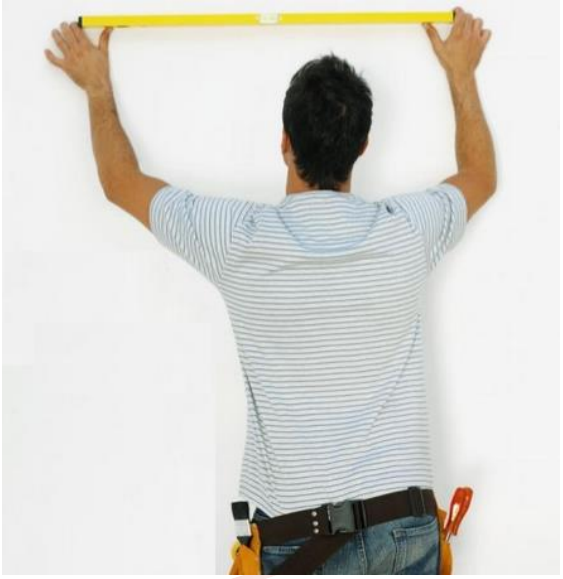
construir esto:

que esto…

El arquitecto de software es…

- El encargado de establecer a que nivel, con que estrategias, y que herramientas son necesarias para realizar una implementación que satisfaga los requisitos funcionales y no funcionales de los sistemas
- Debe ser una persona capaz de identificar las necesidades de los negocios, y conocer las habilidades del equipo de trabajo
- Muchas responsabilidades:
  - Definir, documentar y comunicar la arquitectura
  - Ser enlace con los Stakeholders
  - Asegurarse que todos la entiendan y la sigan
  - Resolver los problemas técnicos
  - Administración de riesgos relacionados con la arquitectura

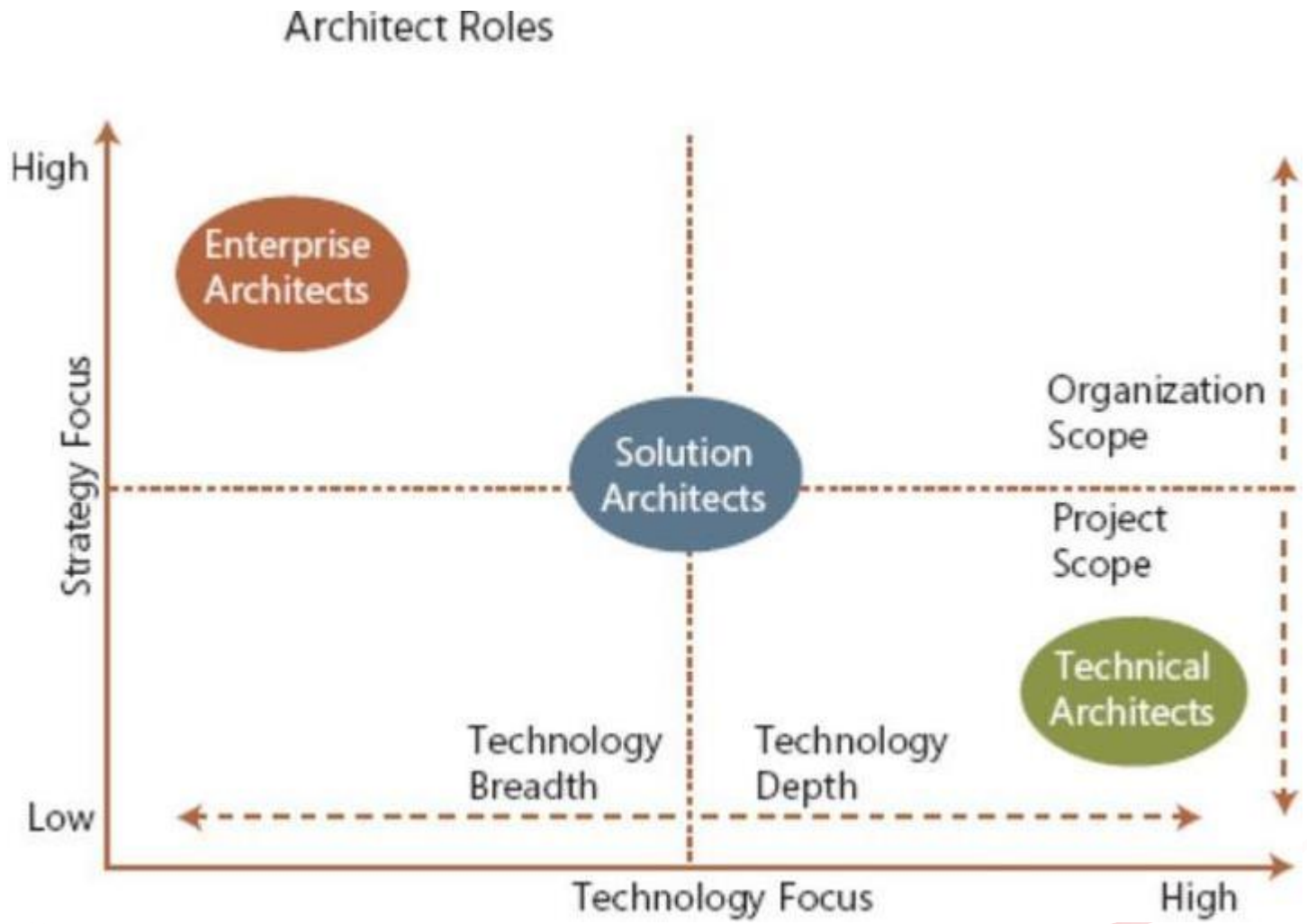
  http://www.sei.cmu.edu/architecture/research/previousresearch/duties.cfm



Habilidades del arquitecto de software

- Tener buen conocimiento de los procesos del ciclo de vida del desarrollo del software
- Poseer conocimiento de arquitecturas de TI y metodologías de diseño sobre diferentes plataformas
- Tener dominio sobre áreas no funcionales, como rendimiento, escalabilidad, interacción con usuarios
- Tiene claridad sobre lo que el negocio necesita y la capacidad de transformarlo en resultados
- Habilidades para razonamiento crítico y toma de decisiones
- Conocimiento de la tecnología y actualización constante
- Pensamiento líder proactivo
- Comprometido con la calidad

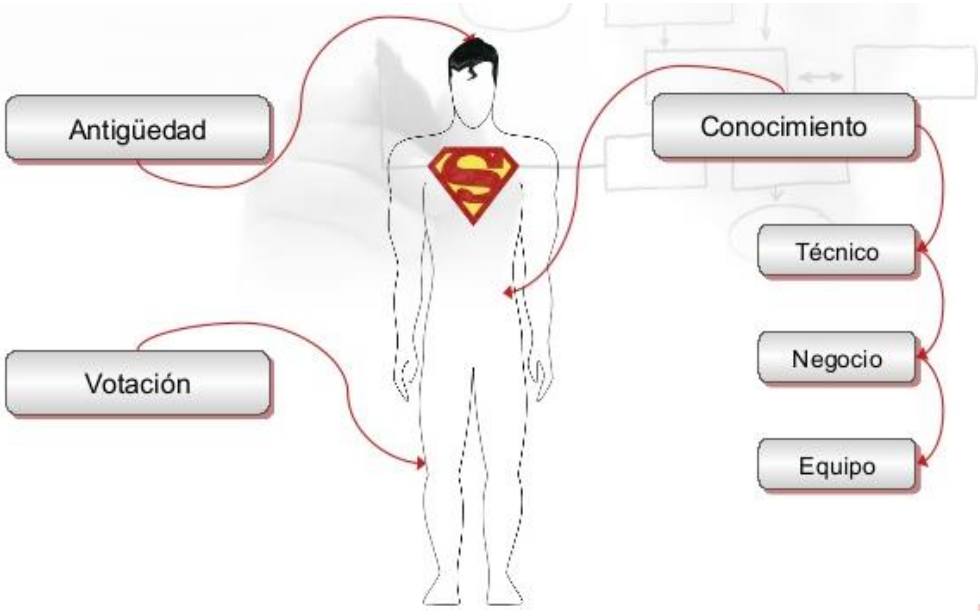
Tipos de arquitecto



Lectura 2: El rol del Arquitecto

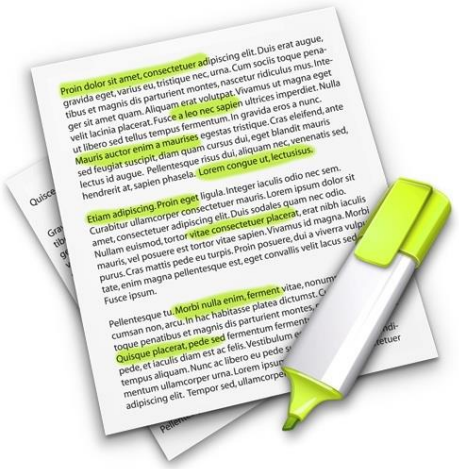
Complementar el rol del Arquitecto de Software con las lecturas en los siguientes enlaces:

- http://www.ibm.com/developerworks/rational/library/mar06/eeles/
- http://micro-howto.blogspot.com/2010/02/eres-un-arquitecto-de-software.html
- https://sg.com.mx/revista/33/el-rol-del-arquitecto-software#.WNaXhIWcHIU



Resumen

- La arquitectura involucra complejas decisiones de diseño que después serán difíciles de corregir
- La definición de la Arquitectura se hace difícil por la naturaleza temprana en el ciclo de vida del diseño
- La arquitectura reduce el riesgo por el uso de patrones de diseño probados
- Los patrones tienen características bien conocidas que son apropiadas para particulares tipos de requerimientos
- Las arquitecturas se concretan con el uso de la tecnología.
- El rol de arquitecto es mucho más que un diseñador técnico.



¿Preguntas?

- ¿Por qué es importante la arquitectura de Software?

Referencias

- Ian Gorton , *Essential Software Architecture (2nd Edition)*
  - Chapter 1, Understanding Software Architecture
- Len Bass, Paul Clements, Rick Kazman , *Software Architecture in Practice (3rd Edition)*
  - Chapter 1, What is Software Architecture?
- Humberto Cervantes, Perla Velasco, Luis Castro. *Arquitectura de Software. Conceptos y ciclo de desarrollo*
  - Capítulo 1, Introducción: La Arquitectura y el desarrollo de Software
- Humberto Cervantes, Rick Kazman. *Designing Software Achitecture (1st Edition)*
  - Chapter 1, Introduction
- Links:
  - http://www.epidataconsulting.com/tikiwiki/tiki-read_article.php?articleId=7&comments_parentId=171&comments_per_page=1&thread_style=commentStyle_plain
  - https://es.slideshare.net/RevistaSG/el-rol-de-arquitecto-de-software