



Análisis y Diseño de Sistemas 2

2015

Ing. Luis Alberto Arias Solórzano

Unidad 2

Arquitectura de software

Definiciones

Existen muchas definiciones debido a los diferentes puntos de vista sobre la arquitectura de un software.

- ▶ Punto de vista de IEEE:

“Organización fundamental de un sistema incorporado a sus componentes, la relación entre ellos y el ambiente, y los principios gobernando su diseño y evolución”

La arquitectura captura la estructura de un sistema en términos de componentes y como éstos interactúan, define un diseño de reglas amplio del sistema y considera como éste puede cambiar

- ▶ **Componente:** Parte modular de un sistema que encapsula su contenido, su manifestación es reemplazable dentro de su ambiente. Un componente define su comportamiento en términos de interfaces

Arquitectura de software

- ▶ **Punto de vista de un Arquitecto:**

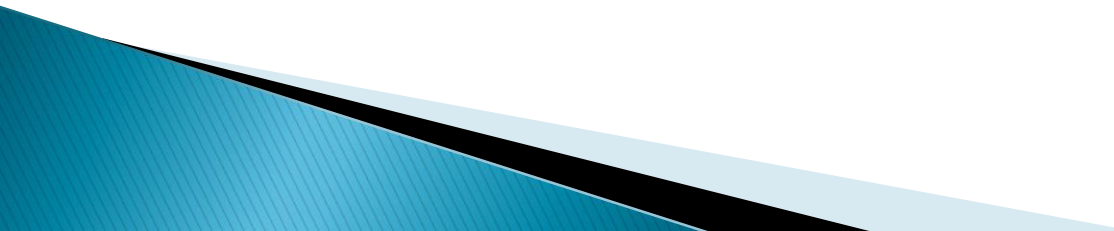
“La arquitectura de software de un programa o sistema de computación es la estructura o estructuras del sistema, que abarca elementos de software, las propiedades externas visibles de éstos elementos y las relaciones entre ellos”

La Arquitectura del Software es el diseño de más alto nivel de la estructura de un sistema. Una Arquitectura de Software, también denominada Arquitectura lógica, consiste en un conjunto de patrones y abstracciones coherentes que proporcionan el marco

¿Para qué sirve definir una arquitectura?

Establecer los fundamentos para que analistas, diseñadores, programadores, etc. trabajen en una línea común que permita alcanzar los objetivos del sistema de información, cubriendo todas las necesidades

Arquitectura de software

- ▶ Una arquitectura de software se selecciona y diseña con base en objetivos (requerimientos) y restricciones.
 - ▶ Los objetivos son aquellos prefijados para el sistema de información, pero no solamente los de tipo funcional, también otros objetivos como la mantenibilidad, auditabilidad, flexibilidad e interacción con otros sistemas de información.
 - ▶ Las restricciones son aquellas limitaciones derivadas de las tecnologías disponibles para implementar sistemas de información.
 - ▶ Unas arquitecturas son más recomendables de implementar con ciertas tecnologías mientras que otras tecnologías no son aptas para determinadas arquitecturas. Por ejemplo, no es viable emplear una arquitectura de software de tres capas para implementar sistemas en tiempo real.
- 

Arquitectura...


- ▶ La arquitectura de software no sólo se ocupa de la estructura y comportamiento, sino también del uso, funcionalidad, rendimiento, capacidad de recuperación, reutilización, comprensión, limitaciones económicas y tecnológicas así como las compensaciones, y la estética de un sistema. Estos son los atributos de calidad.



La Arquitectura de Software define la estructura del sistema

- ▶ La arquitectura de software define, de manera abstracta, los componentes que llevan a cabo alguna tarea de computación, sus interfaces y la comunicación entre ellos. Toda arquitectura debe ser implementable en una arquitectura física, que consiste simplemente en determinar qué computadora tendrá asignada cada tarea.
- ▶ Gran parte del tiempo es invertido en pensar como partir una aplicación en un subconjunto interrelacionado de componentes, módulos, objetos o cualquier unidad de software
- ▶ Toda arquitectura de software debe describir diversos aspectos del software. Generalmente, cada uno de estos aspectos se describe de una manera más comprensible si se utilizan distintos modelos o vistas.
 - Cada uno de ellos constituye una descripción parcial de una misma arquitectura y es deseable que exista cierto solapamiento entre ellos. Esto es así porque todas las vistas deben ser coherentes entre sí, evidente dado que describen la misma cosa.

La Arquitectura del Software define la estructura del sistema

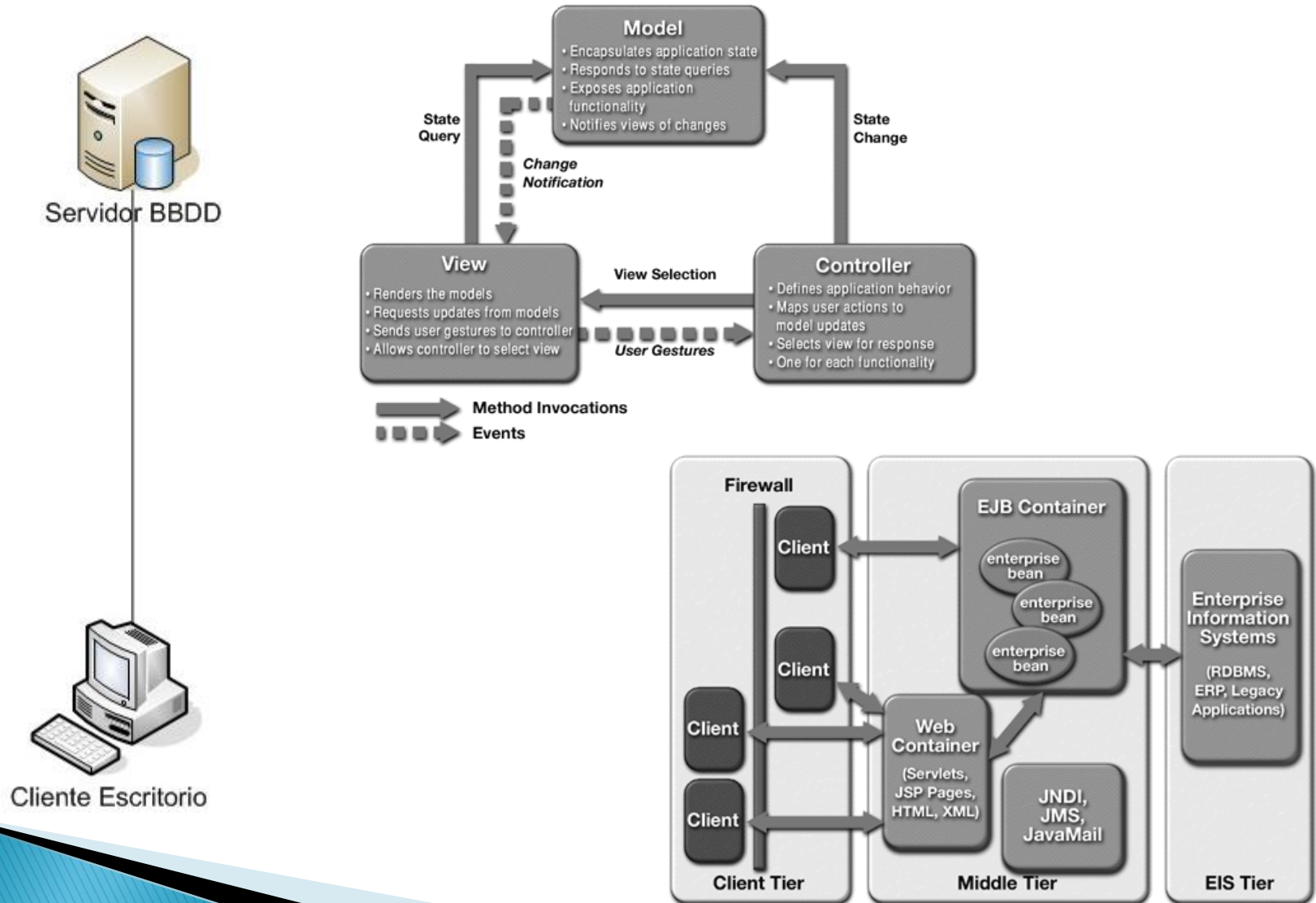
- ▶ Al partir una aplicación, el arquitecto asigna responsabilidades a cada componente, las cuales definen tareas que un componente debe de hacer para actuar en la aplicación, cada componente juega un rol específico en la aplicación y todos los componentes juntos que incluye la arquitectura, colaboran para proveer una funcionalidad requerida.
 - ▶ El exceso de dependencia hace difícil hacer cambios a los sistemas, el costo incrementa al probar los cambios, incrementa el tiempo de construcción.
 - ▶ Existe dependencia entre componentes cuando un cambio en uno fuerza un cambio en otro, al eliminar dependencias innecesarias, los cambios son localizados y no se propagan a través de la arquitectura.
- 

Arquitecturas Existentes

Cada vez que iniciamos el diseño muchas veces no es necesario inventar una nueva arquitectura de software para cada sistema de información. Lo habitual es adoptar una arquitectura conocida en función de sus ventajas e inconvenientes para cada caso en concreto.

- ▶ **Monolítica.** Donde el software se estructura en grupos funcionales muy acoplados.
- ▶ **Cliente-servidor.** Donde el software reparte su carga de cómputo en dos partes independientes pero sin reparto claro de funciones.
- ▶ **Arquitectura en Capas.** Especialización de la arquitectura cliente-servidor donde la carga se divide en tres partes (o capas) con un reparto claro de funciones: una capa para la presentación (interfaz de usuario), otra para el cálculo (donde se encuentra modelado el negocio) y otra para el almacenamiento (persistencia). Una capa solamente tiene relación con la siguiente.
- ▶ **Otras arquitecturas:**
 - ▶ Modelo Vista Controlador.
 - ▶ En pipeline.
 - ▶ Entre pares.
 - ▶ En pizarra.
 - ▶ Orientada a servicios.
 - ▶ Dirigida por eventos.
 - ▶ Máquinas virtuales

Ejemplos de Arquitectura



La arquitectura especifica comunicación entre componentes

- ▶ Al dividir en componentes es necesario pensar como éstos comunicarán datos e información de control.

- ▶ Arquitecturas basadas en patrones:

Los patrones de arquitectura son estructuras que facilitan la comunicación de ciertos componentes, estos son en esencia, planos de arquitectura re-usables que describen la estructura e interacción entre colecciones de componentes

“Un patrón es una idea que ha sido de utilidad en un contexto específico y probablemente lo será en otros contextos”

Arquitectura basada en patrones


- ▶ **Patrones de arquitectura:** Expresan la organización estructural fundamental para un sistema de software, provee un conjunto de subsistemas, especifica sus responsabilidades e incluye reglas y guías para organizar las relaciones entre ellos.
 - Cliente-servidor
 - N-Capas
 - MVC (Patrón Modelo-Vista-Controlador)
- ▶ **Patrones de diseño:** Provee un esquema para refinar los subsistemas o componentes de un sistema de software y las relaciones entre ellos. Describe comúnmente la estructura de la comunicación de los componentes que dan solución a un problema general de diseño dentro de un contexto particular

Patrones y vistas

- ▶ Las vistas de una arquitectura son partes específicas de uno o mas modelos que representan la arquitectura de un sistema, enfocándose en ciertos aspectos de interés para ciertos stakeholders.
- ▶ Los patrones pueden ser de ayuda en el diseño de dichos modelos y en vistas basadas en éstos

La arquitectura direcciona requerimientos NO funcionales

Los arquitectos deben entender los requerimientos funcionales y crear una plataforma que los soporte y simultáneamente satisfaga los requerimientos NO funcionales y restricciones técnicas y de negocio.

- ▶ Restricciones técnicas: Restringen opciones de diseño especificando ciertas tecnologías que la aplicación debe de usar. No son negociables.
 - ▶ Restricciones de negocio: Restringen opciones de diseño para el negocio, la mayoría de veces NO son negociables.
 - ▶ Atributos de calidad: Definen los requerimientos de la aplicación en términos de escalabilidad, habilidad, facilidad de cambios, portabilidad, usabilidad, performance, etc.
- 

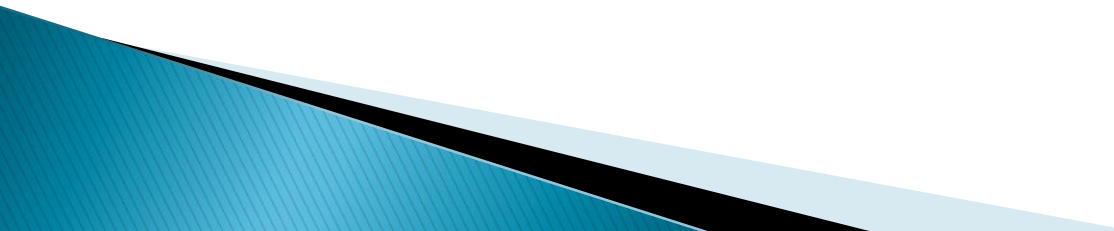
Vistas de la arquitectura

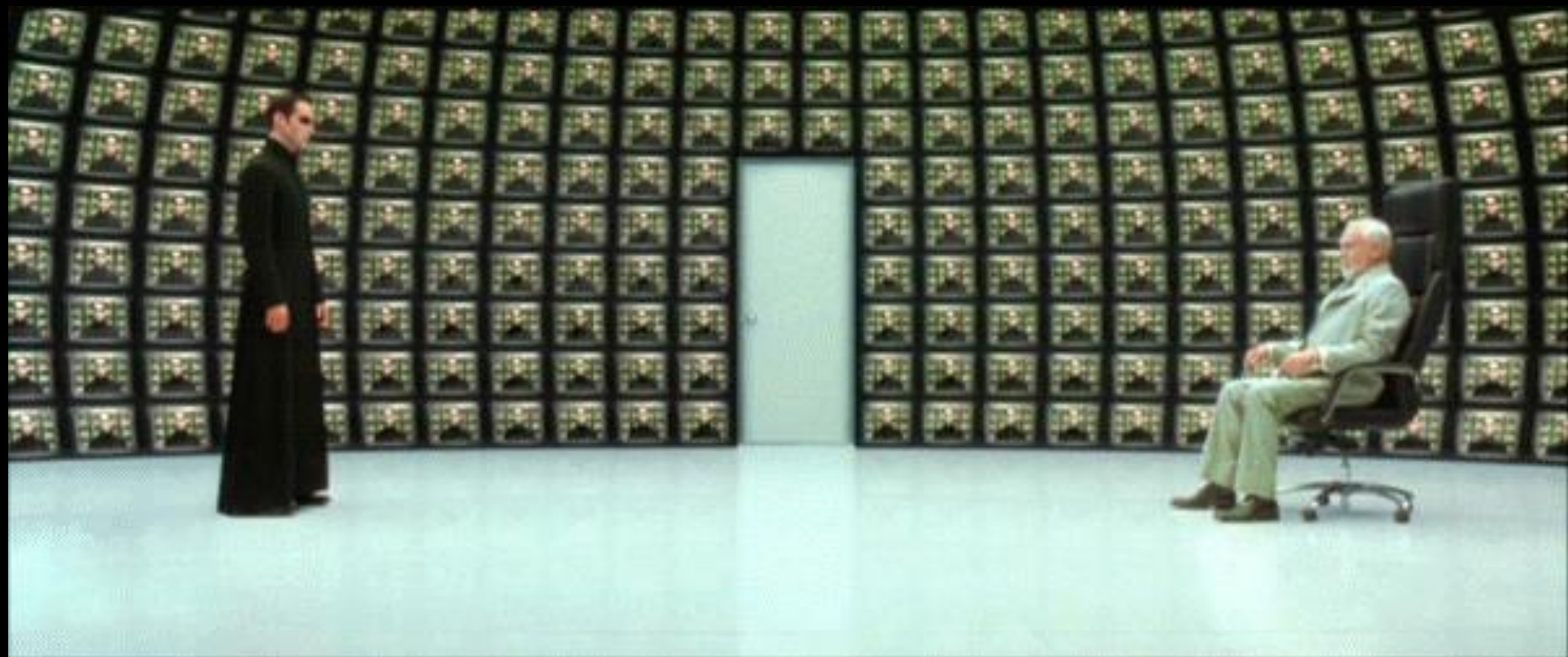
▶ Vistas fundamentales:

- Vista estática: describe qué componentes tiene la arquitectura
- Vista funcional: describe qué hace cada componente
- Vista dinámica: describe cómo se comportan los componentes a lo largo del tiempo y cómo interactúan entre sí

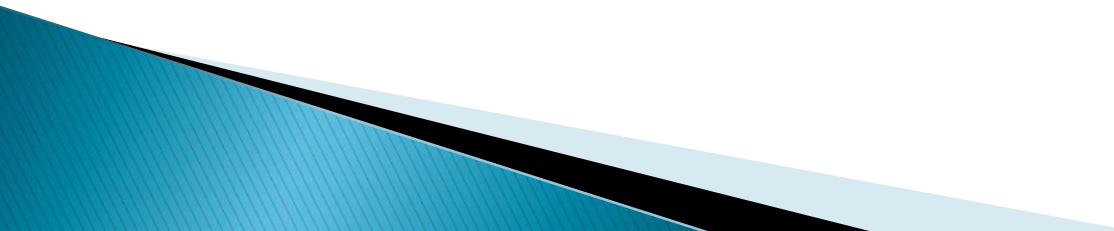
Estas son vistas generales, cada paradigma de desarrollo exige diferente número y tipo de vistas (ej. Modelo 4+1 vistas)

Tipos de arquitectura

- ▶ Según su nivel de abstracción:
 - Arquitectura de negocio
 - Arquitectura de sistemas
 - Arquitectura tecnológica
 - ▶ Según su estilo
 - Orientada a eventos
 - Orientada a servicios
 - Multinivel, etc
- 



Rol del arquitecto

- ▶ El arquitecto obtiene información del problema y diseña una solución que satisfaga los requisitos funcionales y no funcionales, manteniendo flexibilidad al cambio
 - ▶ Se apoya en patrones, modelos y frameworks
 - ▶ Dar soporte técnico y/o tecnológico a desarrolladores, clientes y expertos en negocios
 - ▶ Establece lineamientos que deben tomarse en cuenta en el desarrollo de proyectos
- 

Tipos de Arquitecto

Para definir qué es un arquitecto de software, debemos tener en cuenta un contexto y un escenario en particular:

Arquitecto técnico

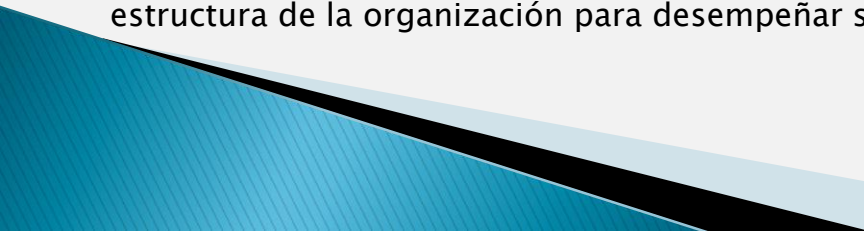
Se trata de profesionales con amplios conocimientos técnicos, conocedor del negocio de los proyectos y que, probablemente, esté asignado a uno o varios proyectos al mismo tiempo. Algunas de sus responsabilidades suelen ser: definir los lineamientos de diseño, su arquitectura y demás cuestiones técnicas de los proyectos.

Arquitecto funcional

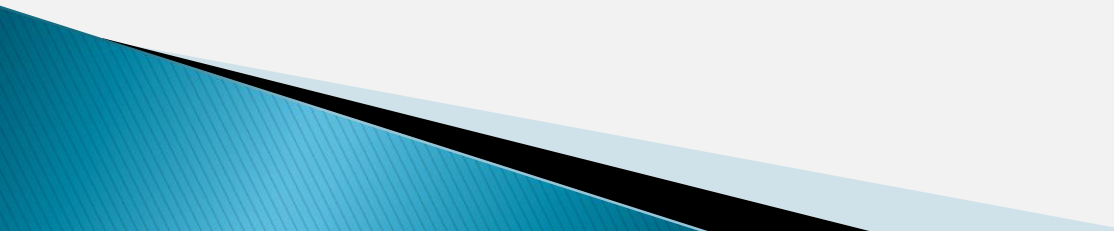
Tienden a ocupar el rol de team leader y, a su vez, de líder técnico. Manejan el project y planifican junto al PM las iteraciones. Suele representar un canal de comunicación fluida entre el PM y los equipos de desarrollo. Validan diseños; guían a los desarrolladores, para que cumplan con las expectativas de calidad tomando métricas, organizando y promoviendo la documentación y las buenas prácticas; aseguran que el proyecto no se desvíe de la arquitectura previamente definida.

Arquitecto Corporativo

Unifica los dos casos mencionados anteriormente; pero con algunos agregados. Este depende de la estructura de la organización para desempeñar sus funciones y actúa bajo políticas pre-establecidas.



Habilidades del arquitecto

- ▶ **Comunicar:** Entre clientes y equipo técnico en conjunto con los analistas de negocio y requerimientos, entre equipos de ingenieros en un proyecto, entre la gerencia, justificando diseño, tomando decisiones. Traslada y explican diferente terminología entre diferentes stakeholders.
 - ▶ **Ingeniería del software:** excelentes habilidades de diseño, deben promover buenas prácticas de ingeniería del software.
 - ▶ **Conocimiento de tecnología:** Siguen desarrollos de tecnología, entienden cómo los nuevos estándares, características y productos pueden ser de utilidad para explotar sus proyectos
 - ▶ **Administración del riesgo:** Los arquitectos están constantemente enumerando y evaluando los riesgos asociados con las decisiones de diseño y tecnología que ellos toman.
- 

Dominio del arquitecto

Existen varias tareas o dominios (más allá de las tareas propias incluidas en el ciclo de vida de un proyecto en particular) en los que suelen estar enfocados los arquitectos y que es conveniente determinar. Estos son:

- ▶ **Tecnología:** Enfocado más en los objetivos de la organización que en las decisiones técnicas, creación de modelos problema/solución, exploración de alternativas de soluciones, preparación de documentos, convencer y comunicar de la factibilidad de las decisiones técnicas a los sponsors y stakeholders.
- ▶ **Estrategia de negocios:** Claro conocimiento de la estrategia de negocio de la organización, de los ciclos de planificación, proceso de toma de decisiones; conocimiento del contexto de la organización (competencia, productos, factores principales que afectan el éxito de la organización); vender la solución, ideas y los valores de calidad transmitidos.
- ▶ **Políticas de la organización:** Conocer los principales stakeholders de la organización. Especialmente, saber lo que ellos quieren y necesitan; mantener una comunicación fluida con éstos. Generación de reportes y comunicación de resultados.
- ▶ **Liderazgo:** Tener una visión del contexto, toma de decisiones, selección y creación equipos, motivador, tener carisma y credibilidad, compromiso y dedicación, evangelización tecnológica, puente entre desarrolladores, PM's y expertos de negocio.

Gracias