A black and white photograph of a man with glasses, wearing a suit and tie, sitting at a desk and looking down at a laptop computer. The image is slightly blurred, suggesting a professional or academic setting.

Modelo de software y arquitectura de software

Modelo de software y arquitectura de software

Contenidos

- › Modelado de software.
- › Calidad de software
- › Arquitectura de software.
- › Características de la arquitectura de software.
- › Objetivos de la arquitectura de software.
- › Importancia de la arquitectura de software.
- › El arquitecto de software.
- › Evolución de la arquitectura de software.
- › Desafíos de la arquitectura de software.
- › Proceso de modelado de arquitectura de software.

Modelado de software.



Y la calidad del producto?

Si lo único que importa es la funcionalidad, cualquier software monolítico serviría

Modelado de software.

Gestión de requerimientos



Modelado de software.

Qué pasa con:

- la modificabilidad
- la interoperabilidad
- la disponibilidad
- la seguridad
- la predictibilidad
- la portabilidad ….

Los atributos de calidad del software y su caracterización son esenciales.

Modelado de software.

Si no se desarrolla explícitamente la arquitectura, se obtendrá una de todas formas, pero puede que lo que se obtenga no satisfaga las necesidades!



- Descripciones operacionales
- Requisitos funcionales de alto nivel
- Sistemas legados



- Diseño detallado
- Implementación

Modelado de software.

Atributos de calidad:

- ➊ Rara vez se capturan como parte de la especificación de requisitos.
- ➋ Generalmente son sólo vagamente comprendidos.
- ➌ Frecuentemente poblemente articulados.
- ➍ La funcionalidad es en gran medida ortogonal a los requisitos de calidad.
- ➎ Los sistemas se descomponen en elementos para lograr variados propósitos, más allá de la funcionalidad.

Calidad de software

Es uno de los problemas fundamentales del software y la computación.

Se puede definir como:

- “conjunto de cualidades que caracterizan el software y que determinan su utilidad y existencia.”
- “sinónimo de eficiencia, flexibilidad, corrección, confiabilidad, mantenibilidad, portabilidad, usabilidad, seguridad e integridad.”

La calidad del software es medible y varía de un sistema a otro o de un programa a otro.

Puede evaluarse:

- al nivel de "cero fallas".
- sólo a nivel de funcionalidad.
- a nivel de su confiabilidad, mantenibilidad y flexibilidad.

Debe controlarse durante todas las etapas del ciclo de vida del software.

Arquitectura de software

- Define la ***estructura general del sistema***, entendiendo estructura como los componentes que nacen de la abstracción del sistema, que cumple funciones específicas, e interactúan entre sí con un comportamiento definido.
- Puede ser vista como la ***estructura del sistema en función de la definición de los componentes y sus interacciones***, considerando los requerimientos y restricciones del sistema, junto a los argumentos que justifiquen que la estructura definida satisface los requerimientos del sistema.
- Una estructura compuesta por ***componentes de software*** y ***reglas que caracterizan la interacción entre estos componentes***.
- Un ***conjunto de elementos arquitecturales*** (de procesamiento, de datos y de conexión) que tienen una forma particular.
- Una ***colección de componentes*** en conjunto con una descripción de las interacciones entre estos componentes, es decir, de los conectores.

Arquitectura de software

- Una *estructura organizacional de un sistema de software* que incluye componentes, conexiones, restricciones y una exposición razonada de los requerimientos que ella satisface.
- Se refiere a grandes rasgos, a una *vista del sistema* que incluye los componentes principales del mismo, la conducta de esos componentes según se le percibe desde el resto del sistema y las formas en que los componentes interactúan y se coordinan para alcanzar la misión del sistema.
- Tiene que ver con el diseño y la implementación de estructuras de software de *alto nivel*.
- Es el resultado de *ensamblar un cierto número de elementos arquitectónicos* de forma adecuada para *satisfacer los requerimientos funcionales y no funcionales*.

Arquitectura de software

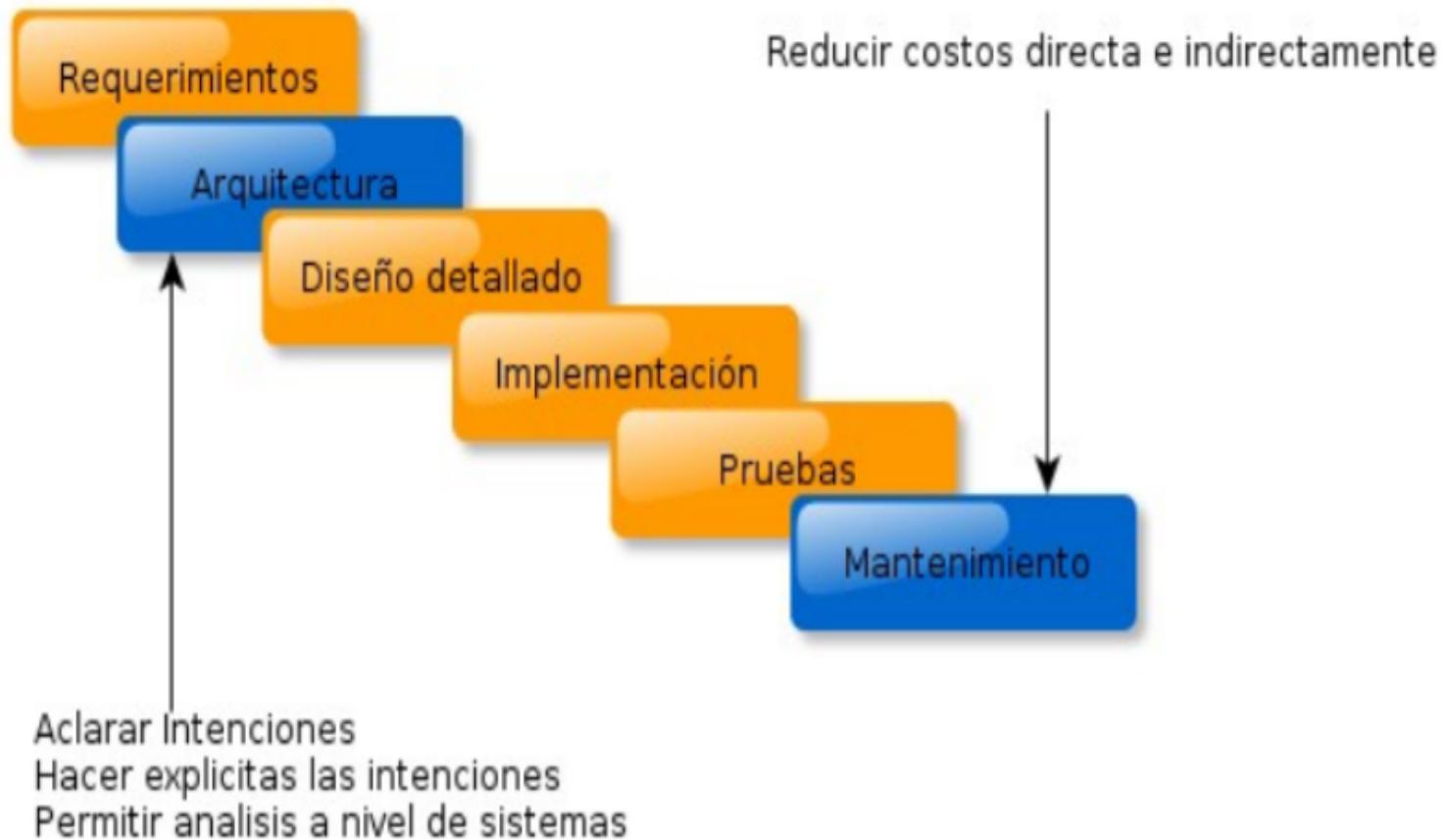
- La IEEE 1471-2000 la define como la *organización fundamental de un sistema* encarnada en sus componentes, las relaciones entre ellos, el ambiente y los principios que orientan su diseño y evolución.
- Se refiere a la *especificación de la estructura del sistema*, entendida como la organización de *componentes* y *relaciones entre ellos*, los requerimientos que debe satisfacer el sistema y las restricciones a las que está sujeto, así como las propiedades no funcionales del sistema y su impacto sobre la calidad del mismo; las reglas y decisiones de diseño que gobiernan esta estructura y los argumentos que justifican las decisiones tomadas.

Arquitectura de software



- La arquitectura de software puede considerarse como el “**puente**” entre los requerimientos del sistema y la implementación.
- La Arquitectura de software asocia las capacidades del sistema especificadas en el requerimiento con los componentes del sistema que habrán de implementarla.

Arquitectura de software



Características de la arquitectura de software

- Parte del diseño de software.
- Nivel del diseño de software donde se definen la estructura y propiedades globales del sistema.
- Incluye sus componentes, las propiedades observables de dichos componentes y las relaciones que se establecen entre ellos.
- Representación de alto nivel de la estructura del sistema describiendo las partes que lo integran.
- Puede incluir los patrones que supervisan la composición de sus componentes y las restricciones al aplicar los patrones.
- Trata aspectos del diseño y desarrollo que no pueden tratarse adecuadamente dentro de los módulos que forman el sistema.
- Toda aplicación tiene una arquitectura, aunque no sea explícita.

Características de la arquitectura de software

- Hace explícito con rigor el repertorio de técnicas, patrones y expresiones para estructurar sistemas de software complejos.
- Incluye modelos, lenguajes y herramientas para la descripción y desarrollo práctico de arquitecturas de software.
- La arquitectura de software no se ocupa de:
 - *Diseño detallado.*
 - *Diseño de algoritmos.*
 - *Diseño de estructuras de datos.*

Objetivos de la arquitectura de software

- ▶ Comprender (abstracción) y mejorar la estructura de las aplicaciones complejas.
- ▶ Reutilizar dicha estructura (o partes de ella) para resolver problemas similares.
- ▶ Analizar la corrección de la aplicación y su grado de cumplimiento respecto a los requisitos iniciales.
- ▶ Permitir el estudio de algunas propiedades específicas del dominio.
- ▶ Planificar la evolución de la aplicación, identificando las partes mutables e inmutables de la misma, así como los costos de los posibles cambios.
- ▶ Permitir la adaptación al cambio por composición, reconfiguración, reutilización, escalabilidad, mantenibilidad, etc.
- ▶ Presentar la organización a alto nivel del sistema, incluyendo aspectos como: la descripción, el análisis de propiedades relativas a su estructura y control global, los protocolos de comunicación, los protocolos de sincronización utilizados, la distribución física del sistema y sus componentes, etc.

Importancia de la arquitectura de software



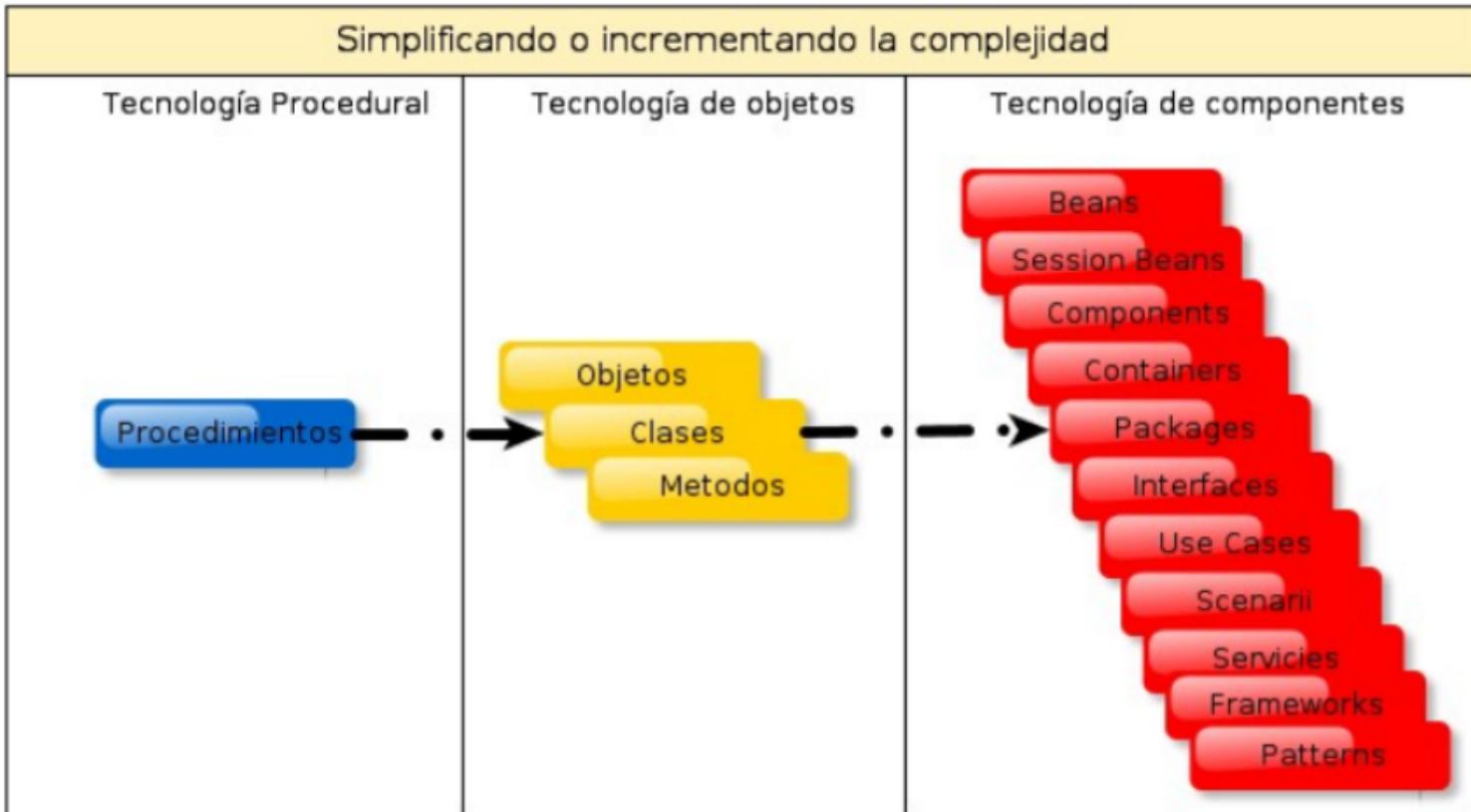
Importancia de la arquitectura de software

- La necesidad del manejo de la arquitectura de un sistema de software nace con los sistemas de mediana o gran envergadura y con la tendencia al crecimiento de los sistemas en cuanto al volumen de datos, códigos y aspectos funcionales y no funcionales.
- En la medida que los sistemas de software crecen en complejidad, bien sea por número de requerimientos o por el impacto de los mismos, se hace necesario establecer medios para el manejo de esta complejidad.
- Incrementos en la evolutividad de los sistemas de software en cuanto a partes o componentes del negocio (globalizaciones, concentraciones, reorganizaciones, competencia, etc) y de la plataformas de ejecución.
- Incremento de la heterogeneidad de los sistemas en cuanto a lenguajes y paradigmas, manejadores de datos y protocolos de acceso, sistemas operativos, plataformas intermedias y sistemas operativos.

Importancia de la arquitectura de software

- Permite descomponer el sistema en piezas.
- Los componentes definidos agrupan aspectos específicos del sistema.
- Es producto de un proceso de abstracción.
- Organiza y constituye la base de la solución de un problema.

Importancia de la arquitectura de software



Importancia de la arquitectura de software

Evolución acelerada del Hardware:

- ▶ Incremento constante de la capacidad de operación (Ordenadores potentes).
- ▶ Miniaturización (Equipos pueden llevarse en el bolsillo)
- ▶ Reducción de costes para la producción de hardware.
- ▶ Avance de las comunicaciones entre sistemas. (hiperconectados)
- ▶ Incremento de tasa de surgimiento de tecnologías (coexistencia de nuevas y viejas tecnologías)

Importancia de la arquitectura de software

La medida en que un sistema alcanza sus requisitos de calidad depende de las decisiones de arquitectura:

- la arquitectura es crítica para alcanzar los atributos de calidad;
- las cualidades del producto deben diseñarse como parte de la arquitectura;
- un cambio en la estructura que mejora una calidad suele afectar las otras cualidades;
- la arquitectura sólo puede permitir, no garantizar, que cualquier requisito de calidad se alcance.

El arquitecto de software

- Es el responsable por la arquitectura de software lo que incluye las decisiones técnicas que rigen sobre el diseño e implementación del proyecto.
- Esto típicamente incluye identificar y documentar los aspectos arquitecturalmente significantes del sistema, incluidos los requerimientos, diseño, implementación y vistas del despliegue del sistema.
- Es responsable por proveer razones fundamentales por las decisiones técnicas tomadas, balancear los intereses de varios stakeholders (interesados), manejar los riesgos técnicos del proyecto y asegurar que las decisiones sean efectivamente comunicadas, validadas y adoptadas.

El arquitecto de software

Gerente de la compañía



Gerente de Producto



Usuario final



Ingeniero de Soporte



Cliente



Bajos costos,
ocupar personal,
aumentar el valor
de los activos
corporativos

Elementos
atraíentes, terminar
rápido, comparable
a la competencia

Comportamiento,
performance,
seguridad,
confiabilidad,
usabilidad

Modificabilidad

Bajos costos,
terminar rápido,
sin muchos
cambios

¿Cómo puedo hacer para que
el sistema tenga todo esto?

Arquitecto

El arquitecto de software

Architecture

Architecture definition, system structure, logical view, physical view, architectural principles, security, etc.

Contributed To | Defined

Software Selection

Application stack, databases, libraries, frameworks, technology standards, etc.

Greenfield Project | Existing System

Infrastructure Selection

Operating systems, hardware, networks, disaster recovery, etc.

Greenfield Project | Existing System

Non-functional Requirements

Performance, scalability, security, etc.

Delivered Against | Justified | Tested

Leadership

Technical leadership, responsibility and authority, steering the team, etc.

Contributed To | Performed

Coaching and Mentoring

Helping people with technical problems, helping people move into new roles, etc.

Design and Code | Architecture

Project Methodology

Project structure and use of methodology such as waterfall, RUP, XP, Scrum, etc.

Contributed To | Defined

Development Processes

Source code control, build process, continuous integration, automated testing and other development processes/tools.

Contributed To | Defined

Practices and Standards

Coding standards and guidelines, project practices, tool selection, etc.

Contributed To | Defined | Enforced

Hands-on Design, Development & Testing

UML diagrams, code, unit tests, etc.

Yes | No

Breadth of Experience

Knowledge of many technologies and architectures.

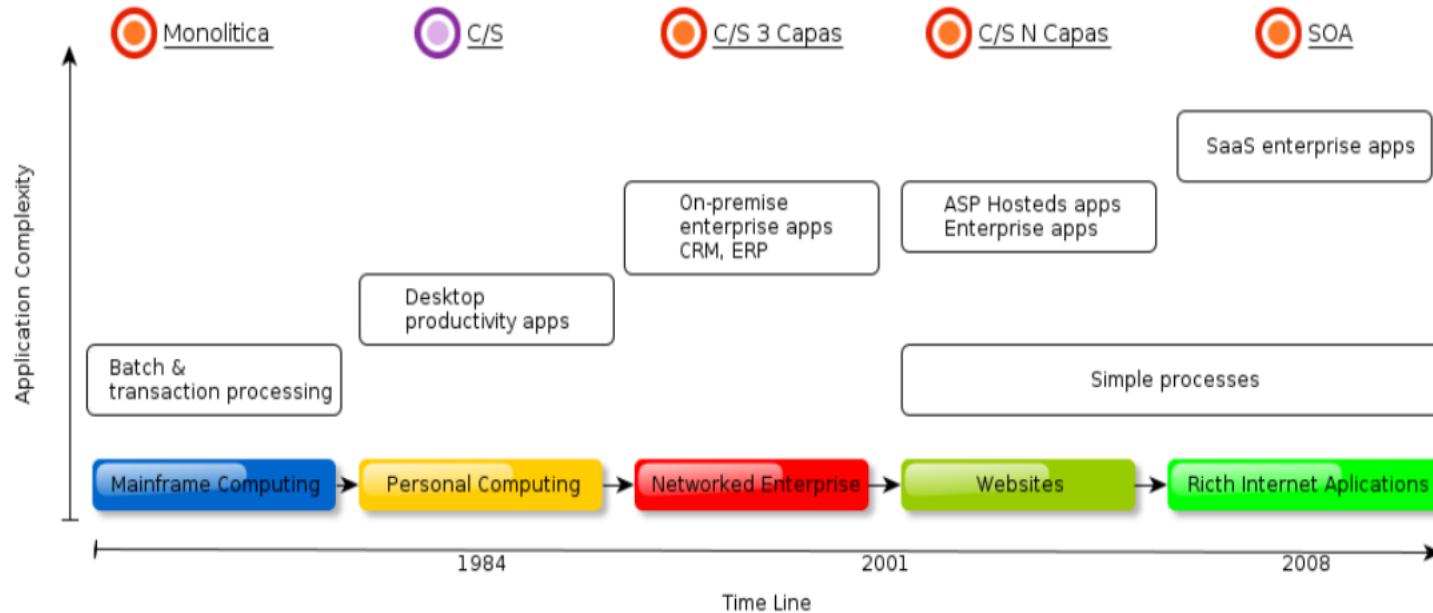
Yes | No

Software Development And Technology Trends

Agile, Web 2.0, SOA, lightweight Java EE...

Awareness | Opinions

Evolución de la arquitectura de software



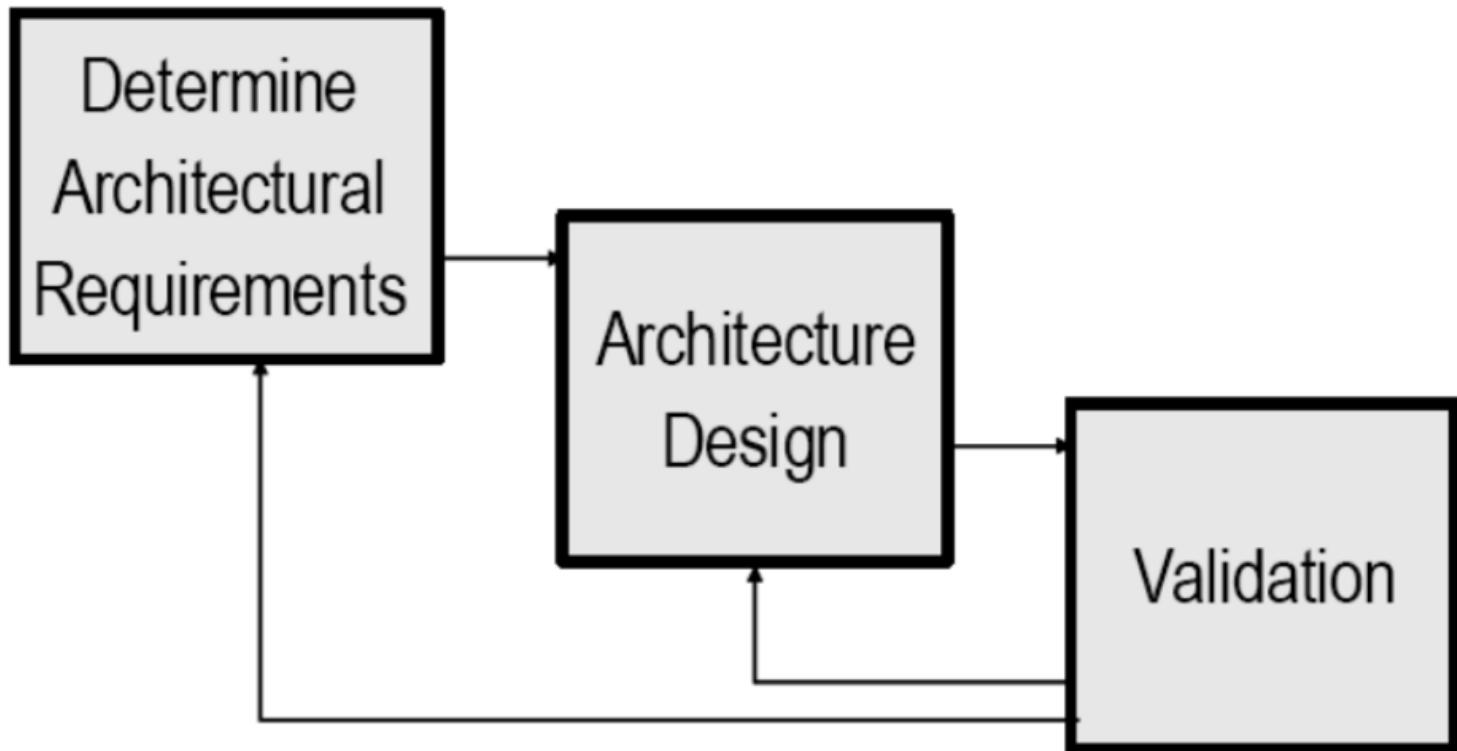
Desafíos de la arquitectura de software

- ▶ ¿Qué significan con precisión atributos de calidad tales como modificabilidad, seguridad, performance y confiabilidad?
- ▶ ¿Cómo se estructura el sistema de modo que tenga estas cualidades deseadas?
- ▶ ¿Se puede analizar el sistema para determinar si tiene estas cualidades?
- ▶ ¿Cuán temprano puede realizarse este análisis?
- ▶ ¿Cómo se sabe si una arquitectura de software es apropiada para un sistema sin tener que construir el sistema primero?
- ▶ ¿Están realmente comprendidos los objetivos organizacionales y las propiedades del sistema requeridas por el negocio y estos están articulados entre sí?

Desafíos de la arquitectura de software

- Los arquitectos deben identificar e involucrar activamente a los interesados de modo de:
 - *comprender las restricciones reales del sistema.*
 - *administrar las expectativas de los interesados.*
 - *negociar las prioridades del sistema.*
 - *tomar decisiones de compromiso.*

Proceso de modelado de arquitectura de software

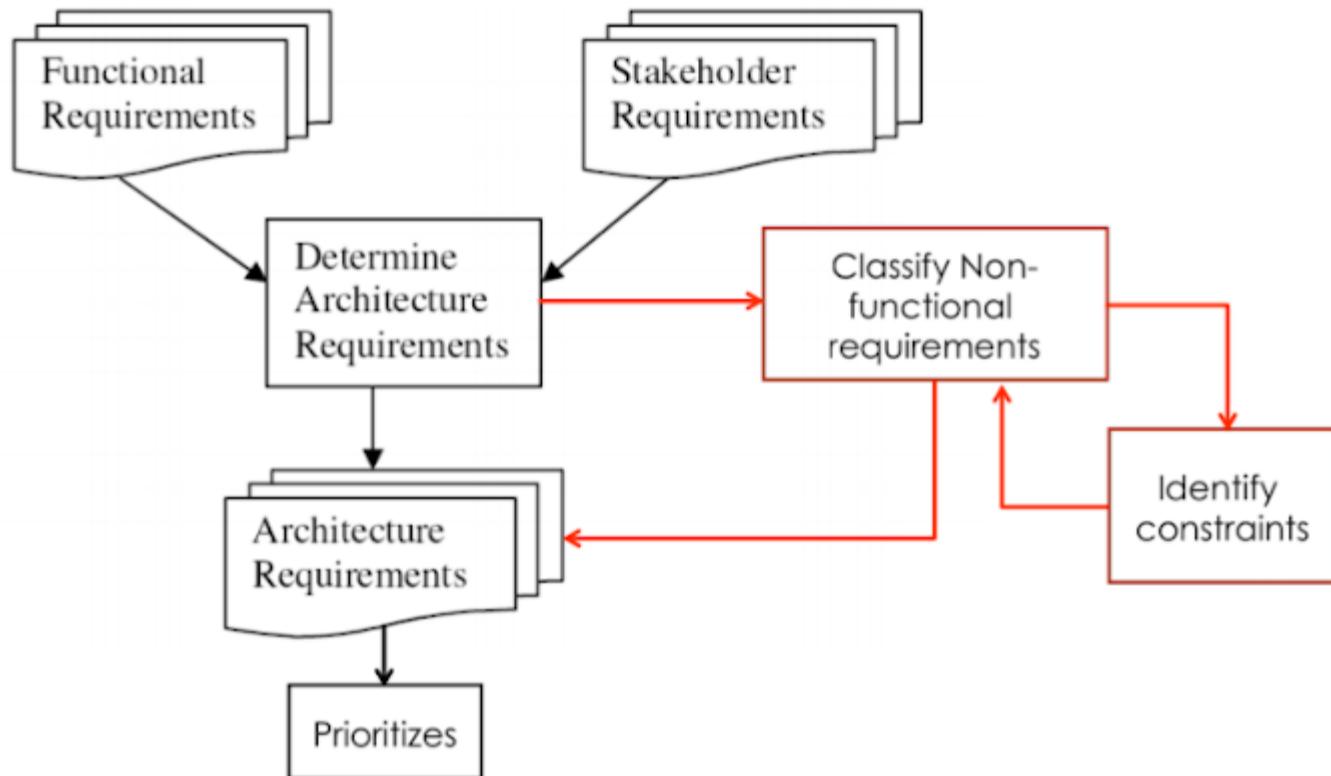


Proceso de modelado de arquitectura de software

- **Definir los requerimientos:** Involucra crear un modelo desde los requerimientos que guiarán el diseño de la arquitectura basado en los atributos de calidad esperados
- **Diseño de la Arquitectura:** Involucra definir la estructura y las responsabilidades de los componentes que comprenderán la Arquitectura de Software
- **Validación:** Significa “probar” la arquitectura, típicamente pasando a través del diseño contra los requerimientos actuales y cualquier posible requerimiento a futuro.

Proceso de modelado de arquitectura de software

- Definir los requerimientos:



Proceso de modelado de arquitectura de software

Definir los requerimientos (no funcionales):

- Describen cómo el software debe comportarse, es decir como hacer algo, no qué debe hacer
- Están relacionados con los requerimientos funcionales porque describen la forma que se espera se logren dichos requerimientos
- En algunos casos tienen restricciones de cómo hacerlo
- Se clasifican de acuerdo al atributo de calidad esperado del sistema

Proceso de modelado de arquitectura de software

Definir los requerimientos (restricciones):

- ▶ Las restricciones (constraints) imponen condiciones sobre la arquitectura que normalmente no son negociables.
- ▶ Limitan el rango de alternativas de decisión del arquitecto
- ▶ Algunas veces hace la vida más fácil para el arquitecto, en otras lo complica.
- ▶ Se pueden clasificar según su naturaleza: Negocio, Desarrollo, Tiempo, Costo, etc.

Proceso de modelado de arquitectura de software

Ejemplos de requerimientos:

Quality Attribute	Architecture Requirement
Performance	Application performance must provide sub-four second response times for 90% of requests.
Security	All communications must be authenticated and encrypted using certificates.
Resource Management	The server component must run on a low end office-based server with 512MB memory.
Usability	The user interface component must run in an Internet browser to support remote users.
Availability	The system must run 24x7x365, with overall availability of 0.99.
Reliability	No message loss is allowed, and all message delivery outcomes must be known with 30 seconds
Scalability	The application must be able to handle a peak load of 500 concurrent users during the enrollment period.
Modifiability	The architecture must support a phased migration from the current Forth Generation Language (4GL) version to a .NET systems technology solution.

Proceso de modelado de arquitectura de software

Ejemplos de restricciones:

Constraint	Architecture Requirement
Business	The technology must run as a plug-in for MS BizTalk, as we want to sell this to Microsoft.
Development	The system must be written in Java so that we can use existing development staff.
Schedule	The first version of this product must be delivered within six months.
Business	We want to work closely with and get more development funding from <i>MegaHugeTech Corp</i> , so we need to use their technology in our application.

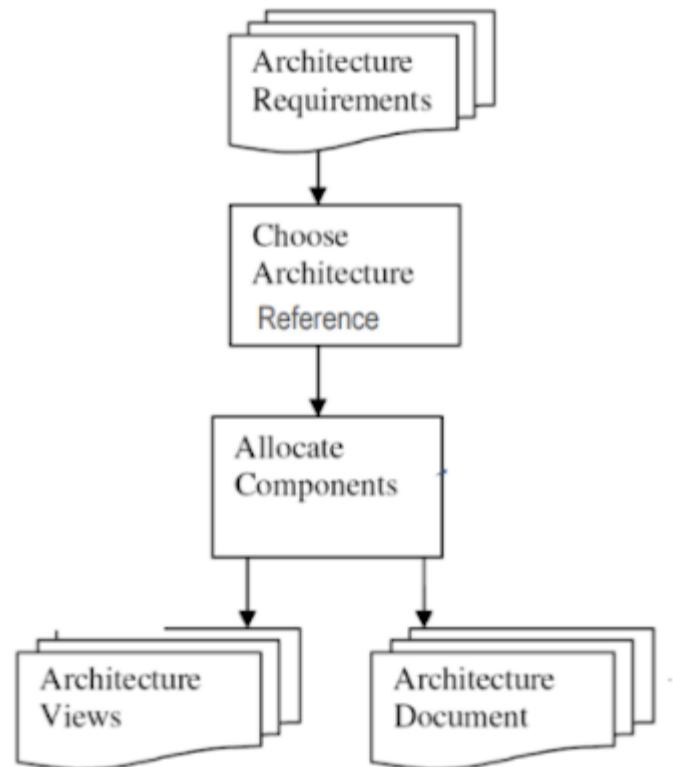
Proceso de modelado de arquitectura de software

Priorización de requerimientos

- **Alta:** La aplicación debe soportar el requerimiento. Estos requerimientos guían el diseño de la arquitectura
- **Media:** Requerimientos que necesitan ser soportados en algún momento o etapa del proyecto pero no necesariamente en esta siguiente versión.
- **Baja:** Se conoce como parte de la “wish-list”. Se pueden implementar cuando sea posible hacerlo.

Proceso de modelado de arquitectura de software

Diseño de la arquitectura:



Proceso de modelado de arquitectura de software

Escogencia de la arquitectura de Referencia :

- ▶ Discutir los posibles estilos y patrones más apropiados que den el soporte requerido para alcanzar los atributos de calidad deseados
- ▶ Basarse en Arquitecturas de Referencia reconocidas tanto por la academia como por la industria que sean implementaciones conocidas, de amplia difusión y uso y muy buena documentación
- ▶ Reconocer el tamaño de la aplicación objetivo
 - ▶ *Aplicaciones pequeñas -> Pocos patrones requeridos*
 - ▶ *Aplicaciones grandes -> Mezcla de varios patrones*

Proceso de modelado de arquitectura de software

Asignación de componentes:

- Su objetivo es definir los componentes principales que comprenderán el diseño
- La arquitectura de referencia define los patrones de comunicación en general para los componentes
- Se busca además:
 - *Identificar cómo los componentes se ajustan a los patrones*
 - *Identificar las interfaces y los servicios que cada componente soporta para así validar la asignación de responsabilidades de los componentes e Identificar dependencias entre ellos*
 - *Identificar las partes de la arquitectura candidatas a distribuirse en varios servidores.*

Proceso de modelado de arquitectura de software

Guías para diseño de componentes:

- Minimizar dependencias entre componentes evitando propagar los cambios entre muchos componentes y por ende sus pruebas.
- Diseñar componentes que encapsulan un alta cohesión del conjunto de responsabilidades.
- Aislard las dependencias con tecnologías Middleware y cualquier COTS. Esto facilita los cambios en el diseño de la AS.
- Usar la descomposición para estructurar componentes jerárquicamente: el componente más externo define la interfaz pública disponible e internamente los llamados a esa interfaz son delegados a otros componentes localmente definidos.
- Reducir al mínimo las llamadas entre componentes, ya que pueden resultar costosas si los componentes se distribuyen (métodos de grano grueso o servicios en las interfaces que hacen más trabajo por cada requerimiento-invocación).

Proceso de modelado de arquitectura de software

Validación:

- Consiste en aumentar la confianza del equipo de diseño con respecto a que la arquitectura es adecuada para cumplir con los requerimientos del sistema.
- Se puede escoger entre dos técnicas: Pruebas manuales o Prototipos.
 - *Prueba manual: Involucra la prueba de la AS usando escenarios*
 - *Prototipo: Involucra la construcción de un prototipo que crea un arquetipo de la aplicación deseada, de esta forma su capacidad para satisfacer las necesidades se pueden evaluar con más detalle a través de prototipos.*