Tema 1

Arquitectura de Software

Nuevo vino en botellas viejas? (i.e., arquitectura de software ≅ diseño global?, architecto ≅ deseñador)



Contenido

- 1 Definición
- 2 Historia
- El Rol del Arquitecto
- Importancia de la arquitectura
- 3



Arquitectura de software, definición

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

(IEEE Estándar sobre la práctica recomendada para descripciones arquitectónicas, 2000.)



Arquitectura de software, definición

La arquitectura de un sistema de software define ese sistema en términos de componentes computacionales e interacciones entre esos componentes.

(Saw and Garlan, Software Architecture)

(Saw and Garlan, Software Architecture, Perspectives on an Emerging Discipline, Prentice-Hall, 1996.)



Arquitectura de Software

instrucciones procedimiento modulo (diseño) patron arquitectura



Arquitectura de software, definición

La arquitectura del software de un sistema es la estructura o estructuras del sistema, que comprende elementos de software, las propiedades visibles externamente de esos elementos y las relaciones entre ellos.

(from Bass, Clements, and Kazman, *Software Architecture in Practice*, SEI Series in Software Engineering. Addison-Wesley, 2003.)



Otros puntos de vista

- La arquitectura es un diseño de alto nivel
- La arquitectura es la estructura general del sistema
- La arquitectura es la estructura, incluidos los principios y las directrices que rigen su diseño y evolución en el tiempo
- La arquitectura son componentes y conectores

Historia

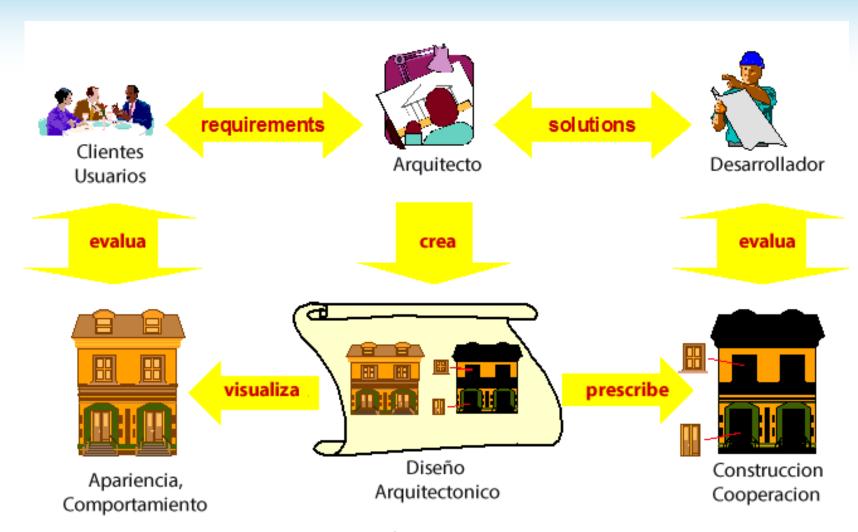
- ❖ 1992: Perry & Wolf
- ❖ 1987: J.A. Zachman; 1989: M. Shaw
- ❖ 1978/79: David parnas, program families
- ❖ 1972 (1969): Edsger Dijkstra, program families
- 1969: I.P. Sharp @ NATO Software Engineering conference:

"Creo que tenemos algo más que ingeniería de software [..] Este es el tema de la arquitectura de software. La arquitectura es diferente de la ingeniería."



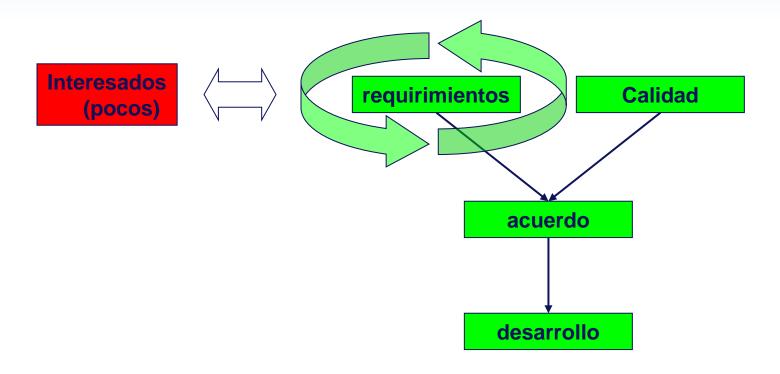


El Rol de un arquitecto





Ciclo de Vida antes de la arquitectura





Característica

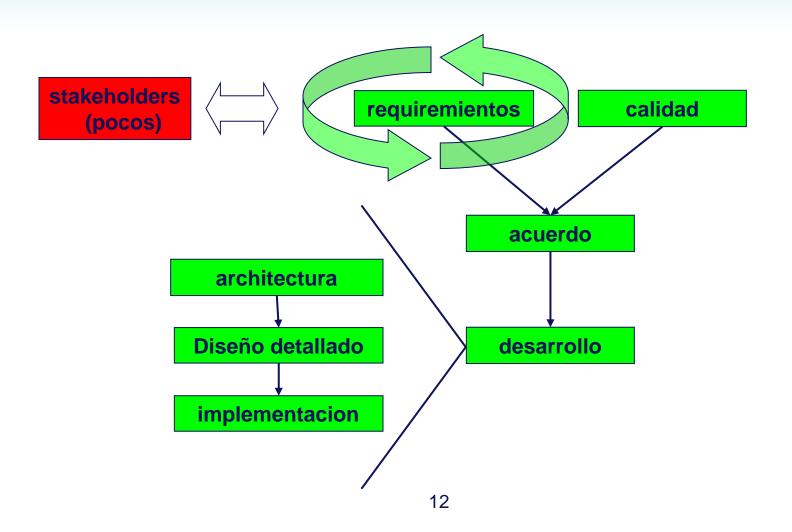
Ilteración principalmente en requisitos funcionales

Pocos interesados (stakeholders) están involucrados

Sin balance entre los requerimientos funcionales y de calidad

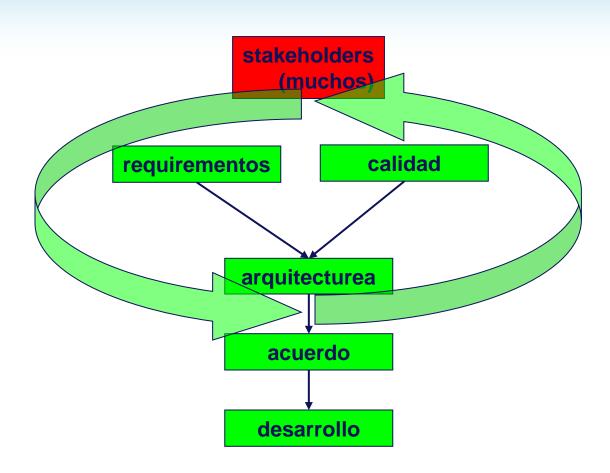


Añadiendo arquitectua, el camino facil





Architectura en el ciclo de vida



toctura?

Importancia de la Arquitectura?

Arquitectura es el vehiculo para comunicarse con los interesados (stakeholder)



- La arquitectura manifiesta el primer conjunto de decisiones de diseño
 - Restricciones en la implementación
 - Dicta estructura organizacional
 - Inhibe o habilita atributos de calidad
- La arquitectura es una abstracción transferible de un sistema
 - Las líneas de productos comparten una arquitectura común
 - Permite el desarrollo basado en plantillas
 - Bases para el entrenamiento



Caracteristicas

- Iteración de los requisitos funcionales y de calidad
- Muchos interesados(stakeholders) involucrados

Equilibrio de requisitos funcionales y de calidad



Arquitectura y calidad del software

- La noción de calidad es fundamental en la creación de software: una arquitectura de software se concibe para obtener una visión de las cualidades de un sistema en la etapa más temprana posible.
- Algunas cualidades son observables a través de la ejecución: performance, seguridad, disponibilidad, funcionalidad, usabilidad
- Y algunos no son observables a través de la ejecución: modificabilidad, portabilidad, reutilización, integrabilidad, capacidad de prueba

16



Diseño Arquitectonico





Diseño Impulsado por atributos

- Eliga un modulo para descomponer
- Refinar este modulo :
 - Elija conductores arquitectónicos (la calidad es la fuerza motriz)
 - Elija un patrón que satisfaga a los conductores
 - Aplique el patron
- Repetir los pasos



Ejemplo de interaciones ADD

❖Alto Nivel: usabilidad ⇒ separar la interfaz de usuario ⇒ Ver que es /arquitectura tres capas(niveles)

❖Bajo Nivel, sin la interfaz de usuario : seguridad ⇒ usuarios autenticados

❖Bajo Nivel, son la capa de datos : disponibilidad ⇒ redundancia activa



Modelo generalizado

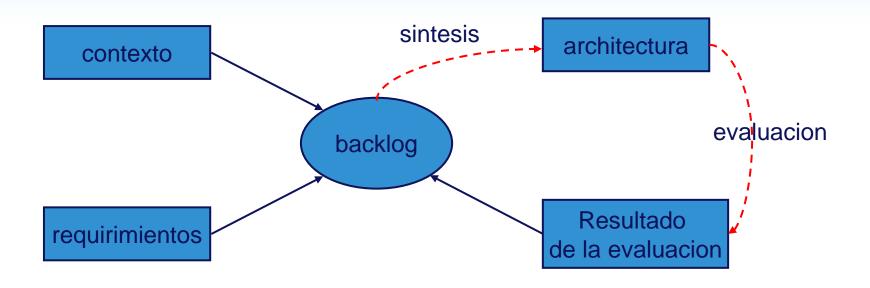
Entender El Poblema

Resolverlo

Evaluar la solucion

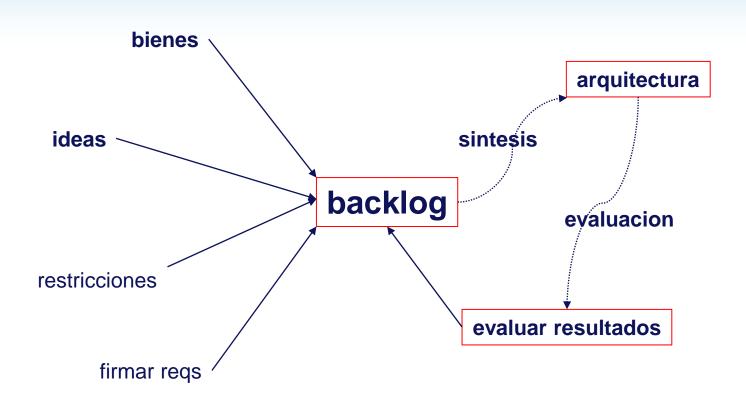
Flujo de trabajo global en diseño de arquitectura







Generalized model (cont'd)



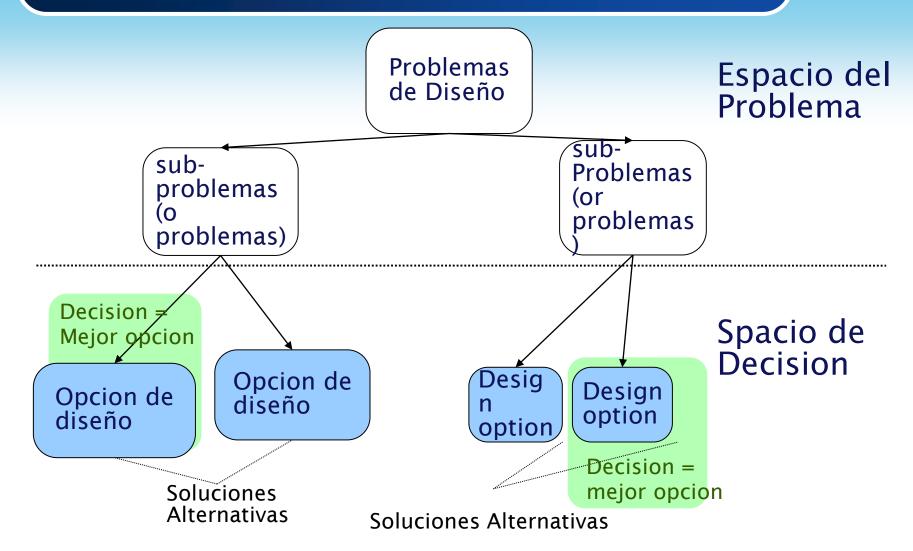
Problemas de diseño, opciones y decisiones



- Un diseñador se enfrenta a una serie de problemas de diseño
 - Estos son sub-problemas del problema general de diseño..
 - Cada problema normalmente tiene varias soluciones alternativas (u opciones de diseño)
 - El diseñador toma una decisión de diseño para resolver cada problema.
 - Este proceso implica elegir la mejor opción entre las alternativas.



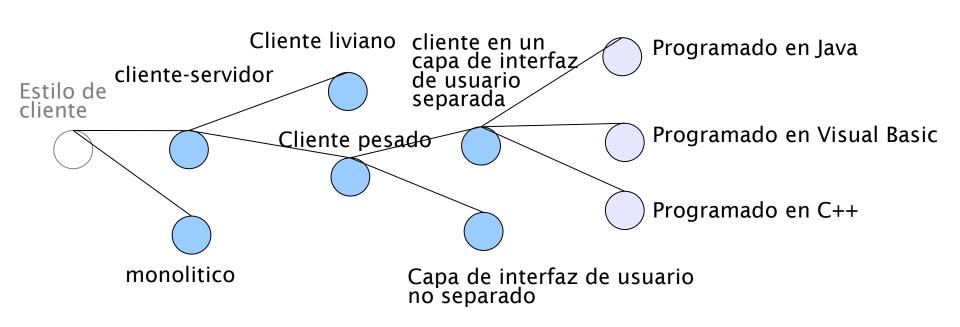
Tomando Desiciones





Espacio de Decisiones

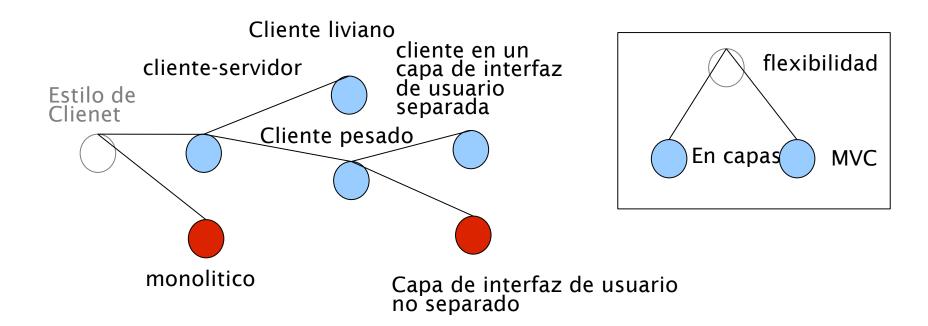
El espacio de posibles diseños que se puede lograr eligiendo diferentes conjuntos de alternativas.





Arbol o grafo?

Los problemas y las opciones no son independientes ...





Más que solo TI

- Las cuestiones y opciones técnicas y no técnicas se entrelazan
- Arquitectos decidiendo sobre el tipo de base de datos

versus

 La dirección decide sobre una nueva asociación estratégica

O

La dirección decide sobre el presupuesto

Algunas decisiones (tácitas) están relacionadas con normas y





Tipos de decisiones

- Implicita, no documentada
 - Desconocido, tácito, por supuesto conocimiento
- Explicita, no documentada
 - DEsaparece con el tiempo
- Explicita, explícitamente indocumentado
 - Razones tácticas y personales
- Explicita, documentada
 - Situación preferida y excepcional

¿Por que es importante



documentar las decisiones de

diseño?

- Evita repetir (costosos) pasos pasados
- Explica por qué esta es una buena arquitectura
- Hace hincapié en las cualidades y la criticidad para los requisitos / objetivos
- Proporciona contexto y fondo



Usos de las decisiones de diseño

- Identificar decisiones clave para un interesado(stakeholder)
 - Tome las decisiones clave rápidamente disponibles. Por ejemplo, presentar personas nuevas y actualizarlas2.
 - ..., Obtenga una justificación, Valide las decisiones en contra de los requisitos
- Evaluar el impacto
 - Si queremos cambiar un elemento, ¿cuáles son los elementos impactados (decisiones, diseño, problemas)?

Limniar la arquitectura identificar los

Elementos de una decisión de



- Problemas: problemas de diseño que se están abordando
- Decision
- Stado: e.g., pendiente, aprobado
- Suposiciones: suposiciones subyacentes
- Alternativas
- Razón fundamental; el por qué de la decisión tomada
- ❖ Implicaciones: p. neesidad de más

Apuntes sobre las decisiones de diseño



- Hofmeister et al, Generalizing a Model of Software Architecture Design from Five Industrial Approaches, Journal of Systems and Software, 2007
- ❖ Tyree and Ackerman, Architecture decisions: demystifying architecture, IEEE Software, vol. 22(2), 2005.
- Kruchten, Lago and van Vliet, Building up and exploiting architectural knowledge, WICSA, 2005.
- Lago and van Vliet, Explicit assumptions enrich architectural models, ICSE, 2005.







Diseño de software en UML

Diagramas de clase, diagramas de estado, diagrama de secuencia, etc.

¿Quién puede leer esos diagramas?

❖¿Qué tipo de preguntas responden?

¿Proporcionan suficiente información?

¿Quién puede leer esos diagramas?

Diseñador, programador, probador, mantenedor, etc.

- Cliente?
- ❖Usuario?

¿Qué tipo de preguntas responden?

- ❖¿Cuanto costara?
- ❖¿Qué tan seguro será el sistema?
- ❖¿Funcionará?
- ❖¿Qué tal el costo de mantenimiento?
- ❖¿Qué pasa si el requisito A es reemplazado por el requisito B?



Analogía con arquitectura de

edificios

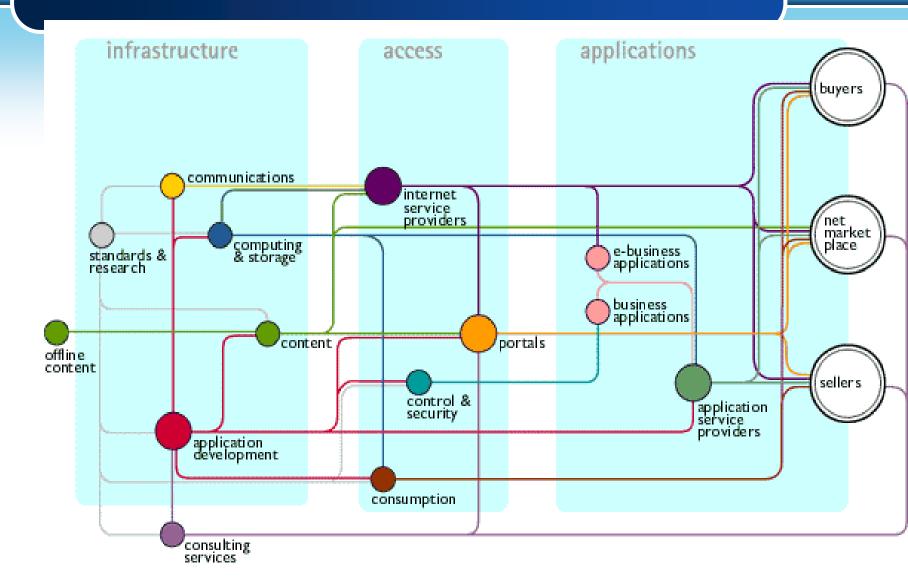
- Imagen general de la construcción (cliente)
- ❖ Vista frontal (cliente, comité de "belleza")
- Imagen separada para el suministro de agua (fontanero)
- Imagen separada para el cableado eléctrico (electricista)
- etc

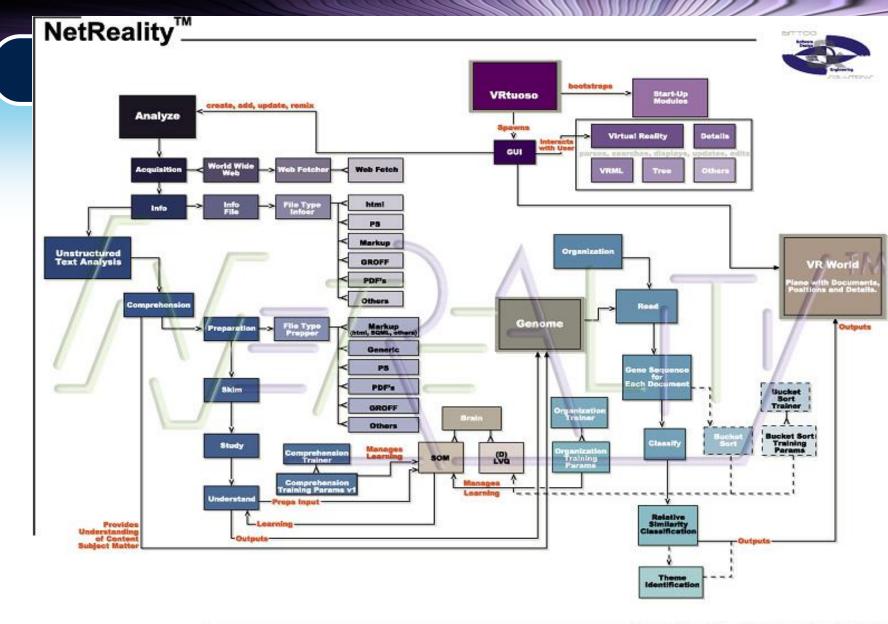


Presentaciones de arquitectura

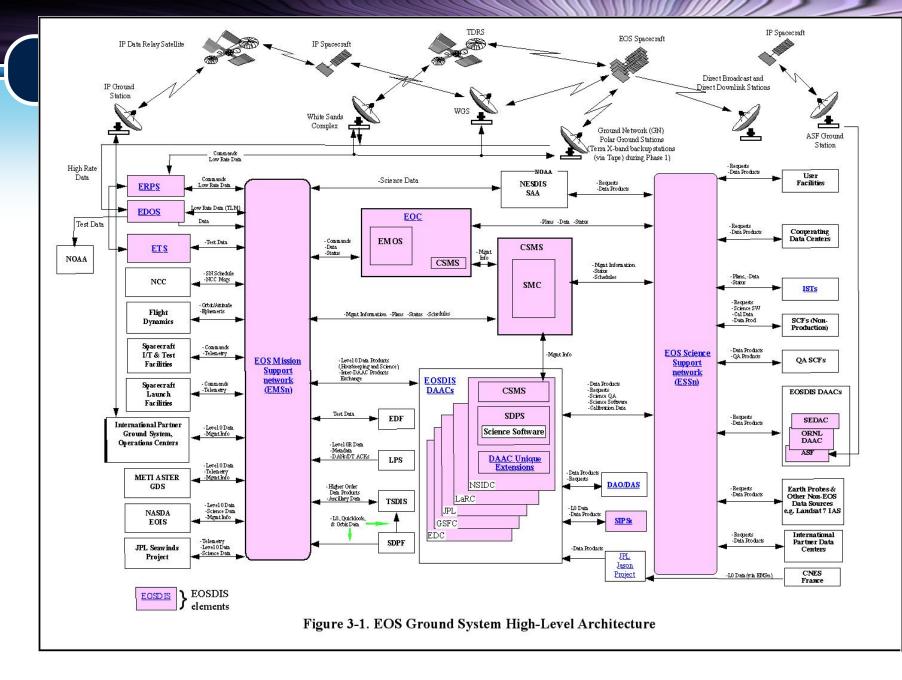
- en la práctica
 En general, dos sabores :
 - Diapositivas de Powerpoint: para administradores, usuarios, consultores, etc.
 - Diagramas UML, para técnicos

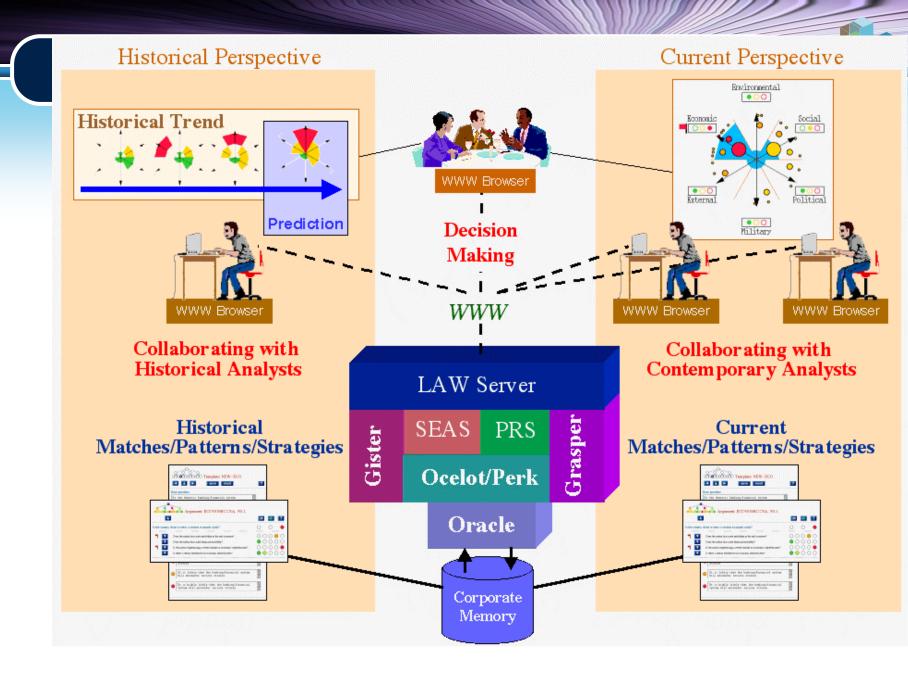


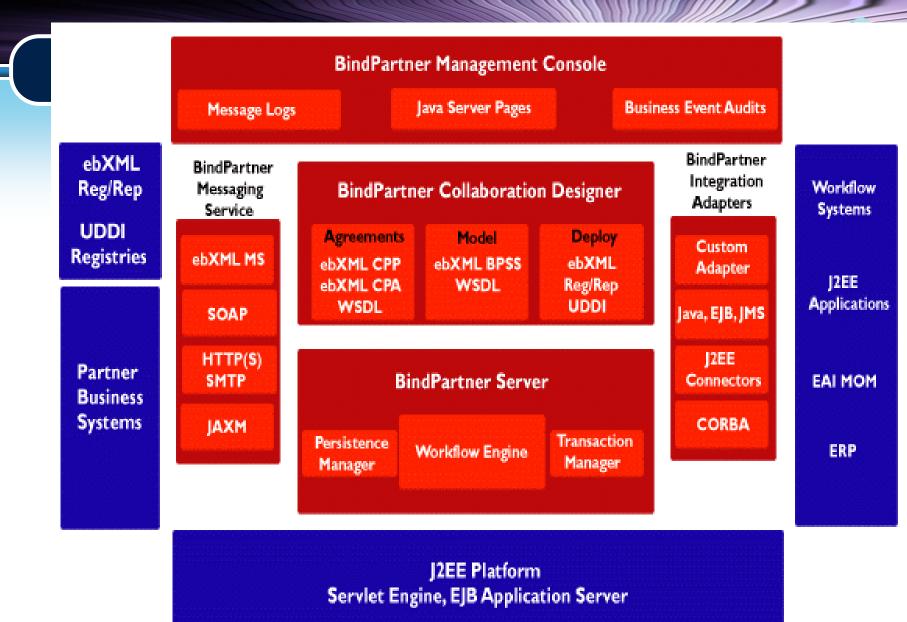




ARCHITECTURAL OVERVIEW









Asi que

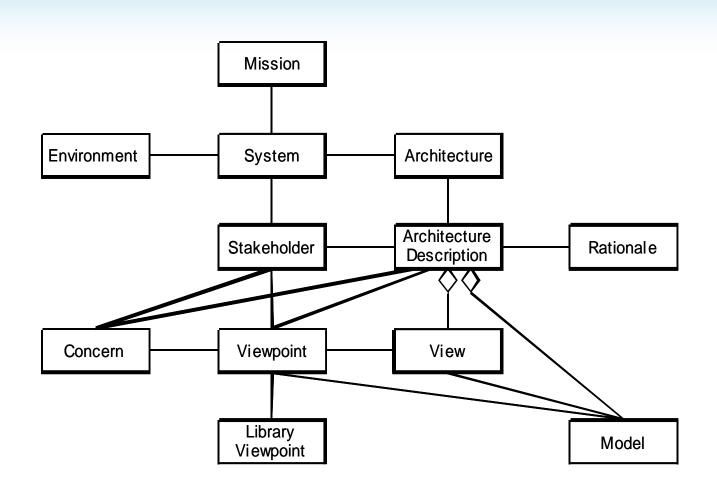
Representaciones diferentes

Para diferentes personas

Para diferentes propositos

Estas representaciones son descriptivas y prescriptivas

IEEE modelo para descripciones arquitectónicas



Algunos terminos (desde el estandar IEEE)

1

- System stakeholder: un individuo, equipo u organización (o clases de este) con intereses en, o preocupaciones relativas a, un sistema.
- View: una representación de un sistema completo desde la perspectiva de un conjunto relacionado de preocupaciones.
- ❖ Viewpoint: Un punto de vista establece los propósitos y el público para una vista y las técnicas o métodos empleados en la construcción de una⁴vista.



Stakeholders

- Arquitectos
- Ingeniero de Requirementos
- Diseñador (también de otros sistemas)
- Implementador
- Probador, Integrador
- Encargado Mantenimiento
- Gerente
- Personas de aseguramiento de calidad

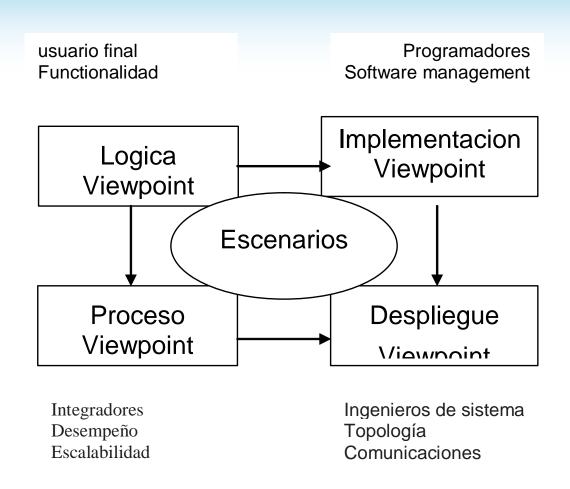


Especificacion Viewpoint

- Nombre del Viewpoint
- Stakeholders a los que esta dirigido
- Preocupaciones dirigidas
- Lenguaje, tecnicas de modelado



Kruchten's 4+1 view model





4 + 1: Viewpoint Logico

El punto de vista lógico admite los requisitos funcionales, es decir, los servicios que el sistema debería proporcionar a sus usuarios finales.

Típicamente, muestra las abstracciones clave (por ejemplo, clases e interacciones entre ellas).



4 + 1: Proceso Viewpoint

- Aborda aspectos concurrentes en el tiempo de ejecución (tareas, hilos, procesos y sus interacciones)
- Tiene en cuenta algunos requisitos no funcionales, como el rendimiento, la disponibilidad del sistema, la concurrencia y la distribución, la integridad del sistema y la tolerancia a fallas.



4 + 1: Despliegue Viewpoint



El punto de vista de Despliegue define cómo los diversos elementos identificados en los puntos de vista lógicos, de proceso y de implementación (redes, procesos, tareas y objetos) deben asignarse a los diversos nodos.

❖ Tiene en cuenta los requisitos no funcionales del sistema, como la disponibilidad del sistema, la confiabilidad (tolerancia a fallas), el rendimiento (rendimiento) y la escalabilidad



4 + 1: Implementacion Viewpoint



El punto de vista de la implementación se centra en la organización de los módulos de software reales en el entorno de desarrollo de software.

❖El software está empaquetado en pequeños fragmentos, bibliotecas de programas o subsistemas, que pueden ser desarrollados por uno o más desarrolladores.



4 + 1: Escenario Viewpoint

- El punto de vista del escenario consiste en un pequeño subconjunto de escenarios importantes (por ejemplo, casos de uso) para mostrar que los elementos de los cuatro puntos de vista funcionan juntos sin problemas.
- Este punto de vista es redundante con los otros (de ahí el "+1"), pero desempeña dos funciones fundamentales:
- actúa como un controlador para ayudar a los diseñadores a descubrir elementos arquitectónicos durante el diseño de la arquitectura; valida e ilustra el diseño de la arquitectura, tanto en papel como como punto de partida para las pruebas de un prototipo arquitectónico.

Vistas arquitectónicas de al (ver = representación de una



Vistas del módulo

- Module is unit of implementation
- Descomposición, usos, capas, clase

Vistas de componentes y conectores (C &

- Estos son elementos de tiempo de ejecución
- Proceso (comunicación), concurrencia, datos compartidos (repositorio), cliente-servidor

Vistas de asignación

- Relación entre los elementos del software y el entorno
- Asignación de trabajo, despliegue, implementación



Vistas del módulo

- Descomposición: las unidades están relacionadas por "es un submódulo de", los módulos más grandes se componen de los más pequeños
- Usos: la relación es "usos" (llamadas, pasa información a, etc.). Importante para la modificabilidad
- Layer es un caso especial de usos, la capa n solo puede usar módulos de capas
 <n
- Clase: generalización, relación "hereda

Vistas de componentes y conectores

- Proceso: las unidades son procesos, conectados por comunicación o sincronización
- Concurrencia: para determinar oportunidades para el paralelismo (conector = hilo lógico)
- Datos compartidos: muestra cómo se producen y consumen los datos
- Cliente-servidor: clientes y servidores que cooperan



Vistas de asignación

- Despliegue: cómo se asigna el software a los elementos de hardware
- Implementación: cómo el software se mapea en estructuras de archivos
- Asignación de trabajo: quién hace qué

Cómo decidir en qué puntos de vista

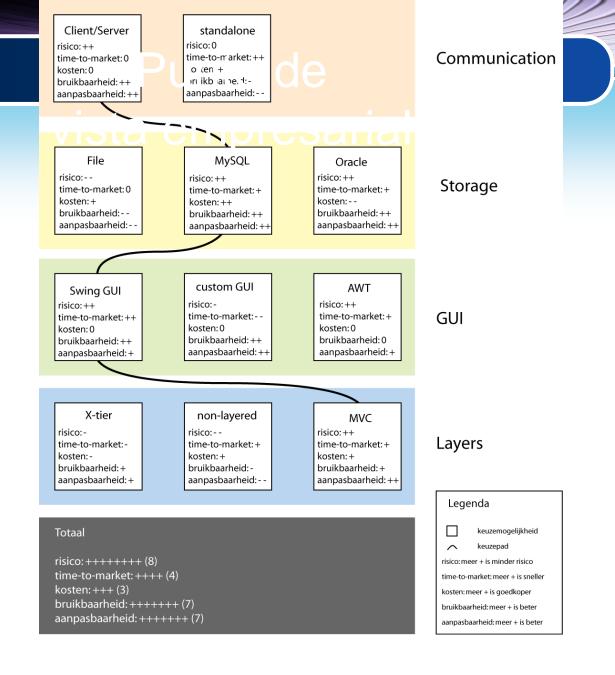
¿Cuáles son los interesados y sus preocupaciones?

- ¿Qué puntos de vista abordan estas preocupaciones?
- Priorizar y posiblemente combinar puntos de vista



Visualización de decisión







Una advertencia sobre la calidad

- Una vista se puede usar para evaluar uno o más atributos de calidad
- Por ejemplo, se puede usar algún tipo de vista de módulo para evaluar la modificabilidad
- Luego debe exponer las decisiones de diseño que afectan este atributo de calidad