

Arquitectura de Software

Arquitectura de Software

- La arquitectura de un sistema computacional es la estructura de ese sistema, y comprende la definición de las componentes del software, sus propiedades externamente visibles, y las relaciones entre las mismas.

2

Arquitectura de Software es:

- Un diseño de alto nivel.
- La estructura del sistema.
- Las componentes de un programa o sistema, sus relaciones, y principios que gobiernan su diseño y su evolución en el tiempo.
- Componentes y conectores.
- Componentes, conectores, configuración y restricciones.

➡ No hay una definición única

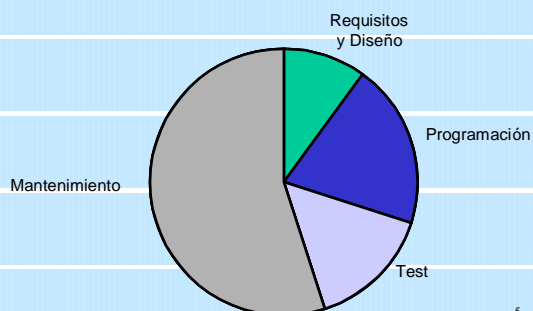
3

Importancia

- Comunicación entre las personas involucradas.
- Documentación temprana de las decisiones de diseño.
- Restricción de la implementación.
- La arquitectura dicta la estructura organizacional.
- Facilita o inhibe propiedades del sistema.
- Permite predecir cualidades del sistema.
- Facilita la administración de la evolución.
- Una abstracción transferible del sistema.
- Las líneas de productos comparten arquitectura.
- Pueden usarse COTS.
- Base para el entrenamiento de nuevo personal.

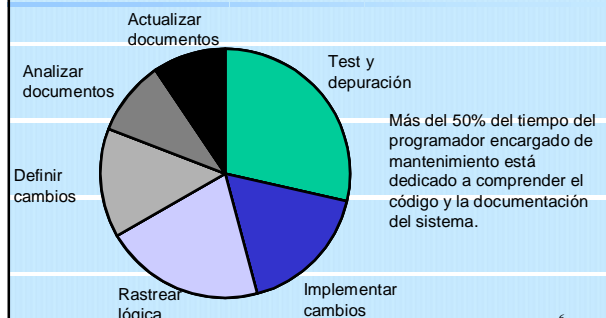
4

Distribución del Costo del Software



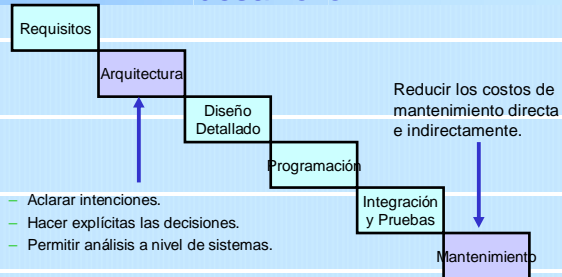
5

Costos de Mantenimiento del Software



6

Arquitectura en el proceso de desarrollo



7

Diseño de Arquitectura

- Sistemas de software grandes pueden (¿deben?) descomponerse en sub-sistemas más pequeños.
- El diseño de la arquitectura del software es el proceso que identifica estos sub-sistemas y establece la infraestructura para su control y su comunicación.

8

Paralelismo con la Arquitectura

- El rol del arquitecto:
 - es el interlocutor entre el cliente y el contratista responsable de construir el edificio;
 - un mal diseño de arquitectura no puede rescatarse con una buena construcción; una mala arquitectura de software no puede enmendarse con una buena implementación;
 - existen especialistas en ciertos tipos de construcciones, así como existen diseñadores de arquitecturas de software;
 - existen distintos estilos de arquitecturas urbanas, así como distintos patrones de arquitecturas de software.



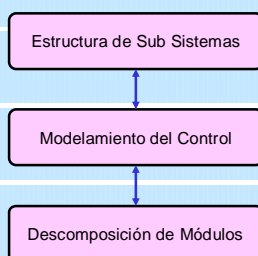
9

Diferencias con la Arquitectura

- | | |
|--|--|
| <ul style="list-style-type: none"> • Los edificios: <ul style="list-style-type: none"> - son tangibles, - sólo pequeños cambios, - la estética es importante, - un estilo de arquitectura, • El arquitecto: <ul style="list-style-type: none"> - es contratado por el cliente para diseñar y supervisar la construcción del edificio. | <ul style="list-style-type: none"> • El software: <ul style="list-style-type: none"> - es intangible, - puede modificarse, - la estructura es esencial, - mezcla de patrones. • El arquitecto de software: <ul style="list-style-type: none"> - es un empleado de la empresa desarrolladora del software. |
|--|--|

10

Proceso de Diseño



11

Sub Sistemas y Módulos

- Un sub sistema:
 - es un sistema independiente que no requiere servicios de otro sub sistema;
 - está compuesto por módulos que interactúan entre sí y con módulos de otros sub sistemas.
- Un módulo:
 - es una componente de un sistema que provee servicios a otros módulos;
 - hace uso de servicios de otros módulos;
 - no son sistemas independientes;
 - pueden tener sub componentes más simples.

12

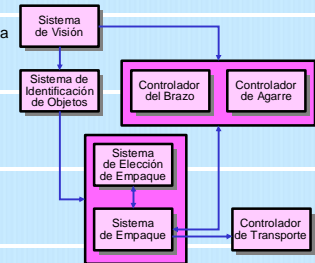
Estructura del Sistema

- La primera actividad del diseño de la arquitectura de un sistema de software es definir un conjunto de sub sistemas que componen el sistema completo.
- Una primera aproximación es un diagrama de cajas y flechas donde:
 - las cajas son sub sistemas,
 - las flechas son:
 - flujos de datos,
 - flujos de control.

13

Estructura de un Robot de Empaque

- El sistema :
 - detecta objetos en una cinta transportadora,
 - identifica el tipo de objeto,
 - selecciona el tipo de empaque apropiado,
 - transporta los objetos al lugar de empaque,
 - los objetos empacados se colocan en otra cinta transportadora.



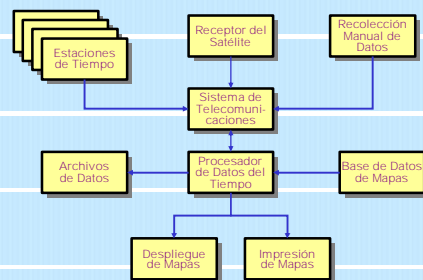
14

Sistema de Mapas del Tiempo

- Las estaciones de tiempo recogen información y la transmiten para ser procesada.
- La base de datos de mapas proporciona mapas básicos para agregar datos del tiempo.
- Un mapa de tiempo puede desplegarse o imprimirse.
- Los datos del tiempo son usados para producir los mapas y también son archivados.

15

Sistema de Mapas del Tiempo



16

Problemas a Considerar

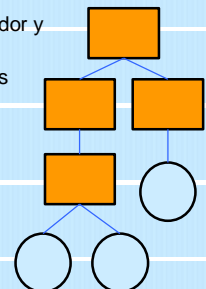
- ¿Cuál es la diferencia entre las distintas flechas?
¿Qué significa cada una de ellas?
- ¿Qué datos se archivan? ¿Qué se hace con los datos erróneos que pueden recibirse de las estaciones de tiempo?
- ¿Cuántas estaciones de tiempo hay en el sistema?
¿Son todas iguales o tienen diferentes particularidades?

➡ Formalidad

17

Ventajas de los Formalismos

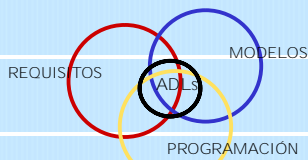
- Mejor comunicación entre el diseñador y el lector.
- Posibilidad de análisis de decisiones tempranas de diseño.
- Posibilidad de generar código a partir de la especificación.
- Mayor probabilidad de mantener la especificación de la arquitectura actualizada con respecto a la evolución del sistema.



18

Architecture Description Languages: ADLs

- Solución tecnológica emergente para representar y analizar arquitecturas de software.
- Casi todos los ADLs están en etapa de investigación, y sólo unos pocos son productos comerciales.



19

Propiedades de un ADL ideal

- Composición - describir un sistema como componentes y conectores independientes.
- Abstracción - describir las componentes y sus interacciones indicando clara y explícitamente su rol abstracto en el sistema.
- Reutilización - reutilizar componentes, conectores y patrones de arquitectura en distintos sistemas.
- Configuración - describir la estructura del sistema, independientemente de los detalles de los elementos que lo componen.
- Heterogeneidad - combinar descripciones de arquitecturas variadas y heterogéneas.
- Análisis - poder aplicar análisis a las descripciones de arquitecturas.

20

Estado del Arte en ADLs

- Sintaxis y semántica formales.
- Notación gráfica y textual.
- Elementos típicos de modelamiento de sistemas distribuidos.
- Algunas facilidades para registrar la motivación de las decisiones de diseño.
- Flujo de datos y control como los principales mecanismos de conexión; pocos representan otros conectores.
- Varios niveles de abstracción.
- Instanciación múltiple de elementos definidos una vez.

21

Ejemplos de ADLs

- UniCon - componentes y conectores.
 - Carnegie Melon University, 1995 †.
- Rapide - modelamiento de dinámica, especificaciones ejecutables.
 - University of Stanford, 1995†?.
- C2 - interfaces humano-computador y evolución.
 - University of California Irvine, 1995.
- Wright - conectores como elementos independientes (CSP).
 - Carnegie Melon University, 1997.
- Darwin - lenguaje de control de la configuración (π -calculus).
 - Imperial College, London, 1995.
- ACME - lenguaje de intercambio de descripción de arquitecturas.
 - Carnegie Melon University, 1998.

22

Modelamiento del Control

- Los modelos de Control organizan las componentes de acuerdo con el flujo de control entre las mismas.
- Existen dos formas básicas:
 - control centralizado:
 - una componente es responsable de iniciar y controlar la ejecución de las demás componentes;
 - control basado en eventos:
 - cada componente puede reaccionar a eventos generados externamente.

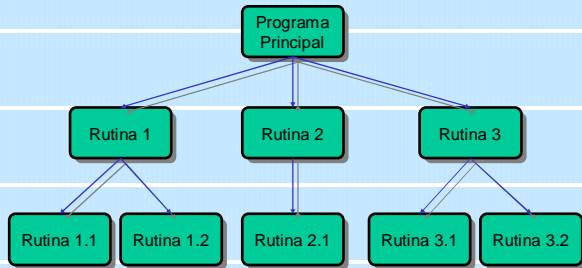
23

Control Centralizado

- Una componente central dirige la acción de las demás.
- Existen dos tipos:
 - modelo de llamada-retorno:
 - jerarquía de procedimientos que ejecutan secuencialmente;
 - modelo de administrador:
 - el control central sincroniza la ejecución paralela de las demás componentes.

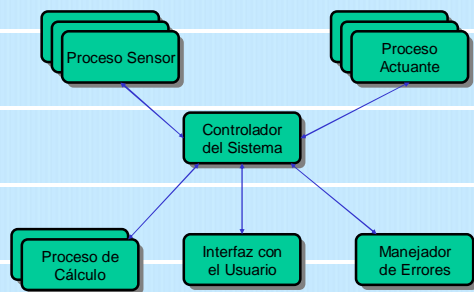
24

Llamada - Retorno



25

Administrador de Control



26

Ventajas y Desventajas

- Ventajas:
 - la rigidez del esquema hace que sea sencillo analizar el sistema y predecir su funcionamiento.
- Desventajas:
 - el acceso a datos compartidos hace que cambios en la estructura de los datos requiera ajustes en todas las componentes.

27

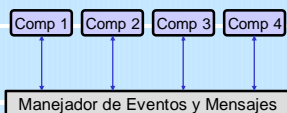
Sistemas Centrados en Eventos

- Eventos son sucesos externos que desencadenan acciones.
- Las componentes no tienen control sobre estos eventos.
- Ejemplos:
 - planillas electrónicas que se modifican al haber un cambio;
 - sistemas de reglas de producción (IA).
- Dos tipos de sistemas:
 - broadcast,
 - interrupciones.

28

Broadcast

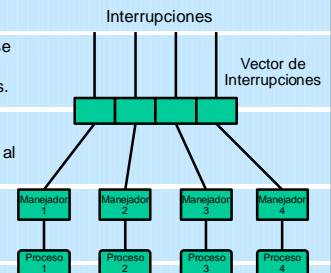
- Cada subsistema registra su interés en ciertos tipos de eventos con los cuales podrá reaccionar.
 - Cada subsistema también genera eventos que son diseminados a todos los demás subsistemas.
 - Aquellos subsistemas que reaccionen con los eventos realizarán la acción correspondiente.
 - Es fácil integrar nuevas componentes.
- Es difícil seguir el flujo de ejecución.
- El manejador puede informar de los eventos sólo a las componentes interesadas para evitar tantos mensajes.



29

Interrupciones

- Se usan en sistemas de tiempo real "hard", donde se requiere una acción inmediata a ciertos eventos.
- Cuando ocurre una interrupción particular, un switch de hardware invoca al manejador de la interrupción.



30

Ventajas y Desventajas

- Ventajas:
 - permite implementar una respuesta muy rápida a algunos eventos.
- Desventajas:
 - es difícil de programar y aún más difícil validar su corrección;
 - existe una limitación en el número de interrupciones que pueden manejarse.

31