Ingeniería de Software
Repaso de Requerimientos y Diseño

Hernán Astudillo
Departamento de Informática
Universidad Técnica Federico Santa María
<hernan at inf.utfsm.cl>

Contenidos hasta ahora

- ✓ Introducción
- ✓ Proceso y ciclo de vida
- ✓ Manejo de proyectos
- ➡ Análisis de Requerimientos
- ➡ Arquitectura y Diseño

Requerimientos

Introducción

- Ideas-fuerza
  - "Hay que saber lo que hay que hacer antes de ponerse a hacerlo"
  - "Conocer el problema antes de inventar la solución"
- Propósito: Reducir el riesgo de construcción
  - Efectividad: riesgo de hacer algo inútil
  - Eficiencia: riesgo de sub-/sobre-estimar recursos
- Problema: "Parálisis de análisis"
  - Querer saberlo todo antes de hacer nada
- ¿Ideas revolucionarias?
  - ¿U obvias?

¿Revolucionario u obvio?

- Estudio de 8000 proyectos en EUU
  - En plazo y costo: 9%-16%
  - Retrasados/excedidos: 53%-60%
  - Cancelados (abortados): 31%
- ¿Porqué fueron cancelados?
  - Requerimientos incompletos: 13%
  - Falta de participación del usuario: 12%
  - Falta de recursos: 11%
  - Expectativas descontroladas: 10%
  - Falta de apoyo gerencial: 9%
  - Requerimientos inestables: 9%
  - Falta de planificación: 8%
  - Obsolescencia pre-parto: 7%

Tipos de requerimientos

- Requerimiento : atributo del sistema
  - Algo que el sistema debe ser capaz de hacer para cumplir su propósito
- Tipos clásicos de requerimientos
  - "Funcionales": describen tareas específicas que el sistema debe ser capaz de hacer
    - p.ej.: imprimir cheques de sueldos
  - "No-funcionales" (restricciones): limitan nuestras opciones para construir una solución
    - p.ej.: generar cheques máx. 4 hrs después de recibir listados de contabilidad
- Analogía
  - Requisitos funcionales: los "verbos" (Xear)
  - Requisitos no-funcionales: los "adverbios" (Xmente)

Proceso

1. Levantamiento y análisis
   - Análisis del problema
   - Descripción del problema
   - Prototipo y prueba
2. Definición y especificación de requerimientos
   - Documentación y validación

- Idea: determinar QUÉ hacer, no CÓMO hacerlo
- Un enfoque:
  - identificar gente, procesos y recursos que apoyan el sistema actual (socio-técnico)
  - documentar sus relaciones e interacciones
- Priorizar requerimientos
  1. Esenciales (deben ser cumplidos)
  2. Deseables pero no esenciales
  3. Opcionales pero eliminables

Técnicas de Descripción

- Estáticas
- Dinámicas
- O.O.
- Otras

Técnicas Estáticas

- Permiten razonar sobre comportamiento de sistemas que cambian poco (o nada) en el tiempo
- Referencia indirecta
- Definición axiomática
- Abstracción de datos

Técnicas dinámicas

- Permiten razonar sobre comportamiento de sistemas que cambian en el tiempo
- Tablas de decisión
- Diagramas de transición
- Redes de Petri

Prototipos

- Prototipos
  - "Throw away"
  - Evolutivo
- Prototipos son herramientas de *control de riesgo*
  - Prototipos funcionales
    - Reducen riesgo de no entender lo que el usuario quiere
  - Prototipos tecnológicos
    - Reducen riesgo de asumir que ciertas cosas son posibles

Diseño y Arquitectura

Diseño

- Diseño: proceso creativo de derivar una solución a partir de un problema
- Características:
  - En general, no hay UNA solución
  - En general, no hay una solución “mejor”
  - Evaluación y elección depende del cliente

Diseño conceptual

- Describir aspectos fundamentales del sistema
  - ámbito (límites)
  - entidades, atributos, relaciones
- Responder preguntas como:
  - ¿de dónde vienen los datos?
  - ¿qué le pasa a los datos en el sistema?
  - ¿cómo verán los usuarios el sistema?
  - ¿qué operaciones y opciones tendrán los usuarios?

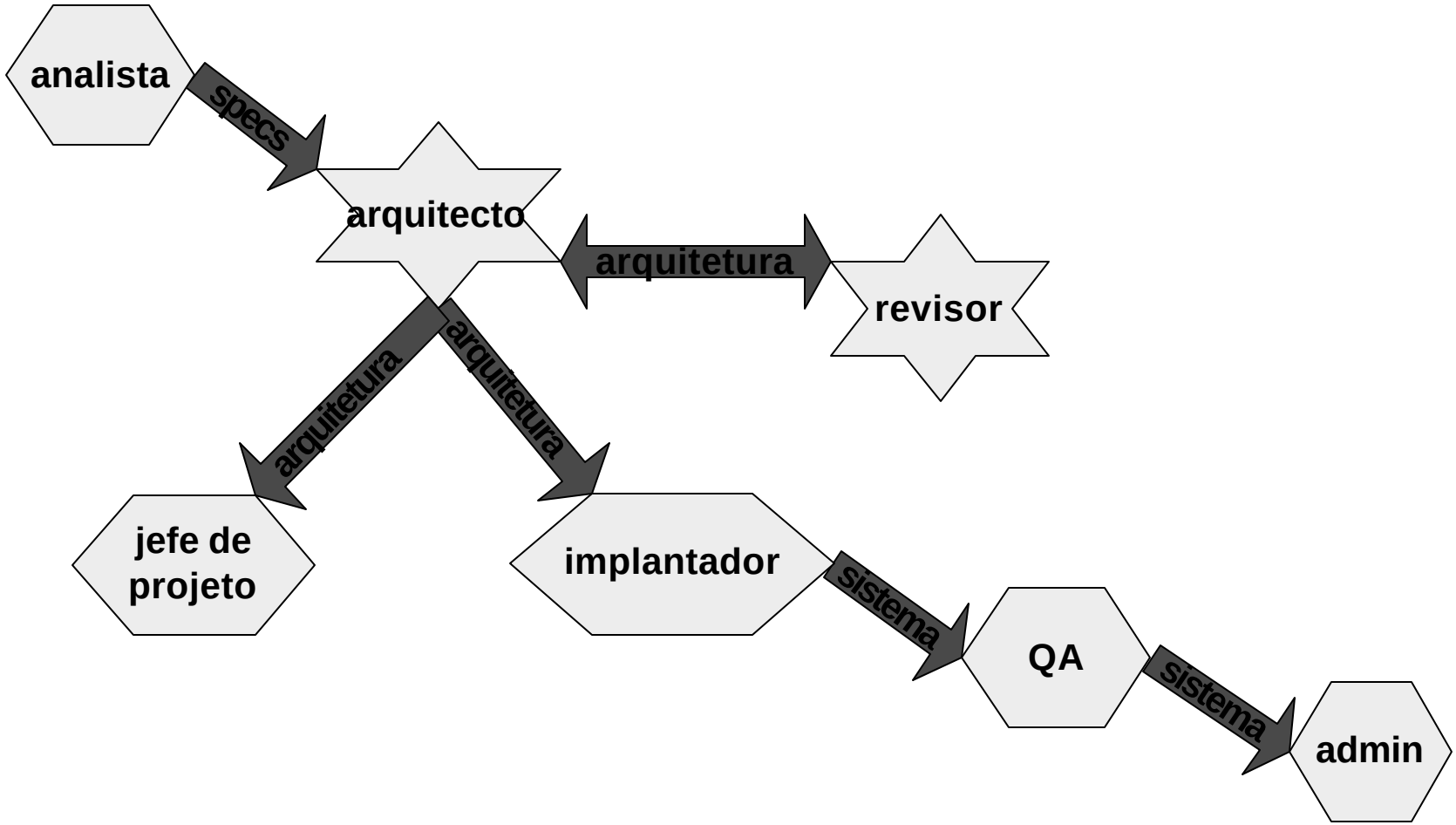
Arquitectura, diseño y programación

- "Arquitectura de Software"
  - Diseño en grande
  - En general, al construir grandes sistemas es más difícil cumplir propiedades sistémicas que las funcionales
- Arquitectura/diseño: contraste escala/propósito
  - Análisis dice *qué debe hacer* el sistema
  - Arquitectura dice *qué software* hacer (reusar/comprar)
  - Diseño dice *cómo hacer* el software

Diseñador – misión

- Tarea del diseñador
  - Con una descripción de *tareas* y *propiedades sistémicas*…
  - …especificar una solución en términos de componentes
  - …que responda las preguntas de los 'stakeholders'
    - …permitir evaluar *a priori* el sistema a ser construido
    - …ser 'guía de acción' a los constructores
    - …permitir elaborar un plan de construcción
- Propósito:
  - Permitir *razonar* sobre el sistema *y* sobre el proceso

Stakeholders



Calidad según los stakeholders

- La *solución (computaciones)* debe...
  - Satisfacer los requisitos (según analistas)
- La *solución (ejecutables)* debe...
  - Ser administrable (según administradores)
- La *solución (software)* debe...
  - Ser 'construible' (según programadores & diseñadores)
  - Ser 'testeable' (según QA)
- La *descripción de la solución* debe...
  - Ser evaluable (según otros arquitectos)
- El *proceso de construcción* debe...
  - Ser manejable (según jefe de proyecto)

Estilos de arquitectura

- Estilo = definición de una familia de sistemas en términos de un patrón de organización estructural
- Un estilo define:
  - *vocabulario* de tipos de componentes y conectores
  - *restricciones* de combinación
  - uno o más *modelos semánticos* para determinar propiedades del todo a partir de las partes

Estilos

- Procesamiento descompuesto en sub-unidades
  - “Pipes and filters”
  - Tipos de datos abstractos
  - Basado en eventos (invocación implícita)
  - Sistemas en capas
  - Repositorios
  - Intérpretes
  - Control de procesos (bucles)
- Ejemplos
    - Unix: capas, pipes, event-driven

Estilo: “pipes & filters”

- Vocabulario: “filtros” y “pipes”
- Estructuras permisibles: salidas a entradas
- Invariantes: filtros son independientes
  - (no comparten estado con otros filtros)
- Especializaciones:
  - “pipelines” (topología lineal)
  - sistema secuencial en lotes (“batch”)
- Ejemplos
  - programación shell Unix

Estilo: repositorio

- Estado compartido por varios componentes
- Vocabulario:
  - repositorio (almacenamiento central)
  - componentes (almacenar, recuperar, actualizar)
- Especializaciones
  - “blackboard”: almacén crentral lanza procesamiento
  - bases de datos: transacciones (externas) lanzan procesamiento
- Ejemplos

- Nota: nada se dice sobre la implementación de la comunicación ni del repositorio

Patrones de diseño

- Patrones de diseño
  - Heurísticas sistematizadas
  - Originados en comunidad de POO [Gamma+]
- Patrón = ...
  - una solución (descripción)
  - a un problema
  - en un contexto (evaluación de "fuerzas" y comentarios)
- Proceso: patrones
  - Elaboración, aprobación y publicación
  - Comunidad fuerte y combativa ☺

Patrones de Buschman+

- Categorías de patrones
  - Estructurales (“from mud to structure”)
    - Capas (“layers”)
    - Pipes & Filtros
    - Pizarra (“blackboard”)
  - Sistemas distribuidos
    - “Broker”
  - Sistemas interactivos
    - MVC: “Model-View-Controller”
    - Presentación-Abstracción-Control
  - Sistemas adaptables
    - Reflexión
    - Micro-kernel

Patrón de arquitectura: MVC

- MVC: “Model-View-Controller”
  - Modelo: elemento de interés para observar/manipular
  - Vista: representación (gráfica u otra)
  - Controlador: manejo y control de vistas de un modelo
- Ejemplos famosos
  - Smalltalk
  - Struts (framework Web)
- Especializaciones
  - “Observer-Oberved”
  - “Publisher-Subscriber”

Patrones: Resumen

- Avance notable
  - ...desde “estilos” descriptivos
  - ...a “patrones” clásicos de sistemas
  - ...a patrones de integración
- Patrones
  - Frutos de la experiencia
  - Heurísticas sistematizadas