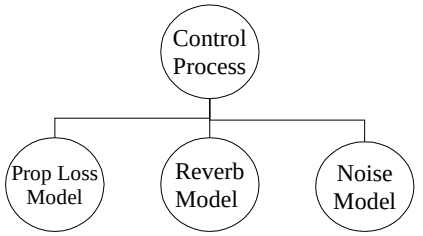
Unidad I

Arquitectura de software

Vocabulario,

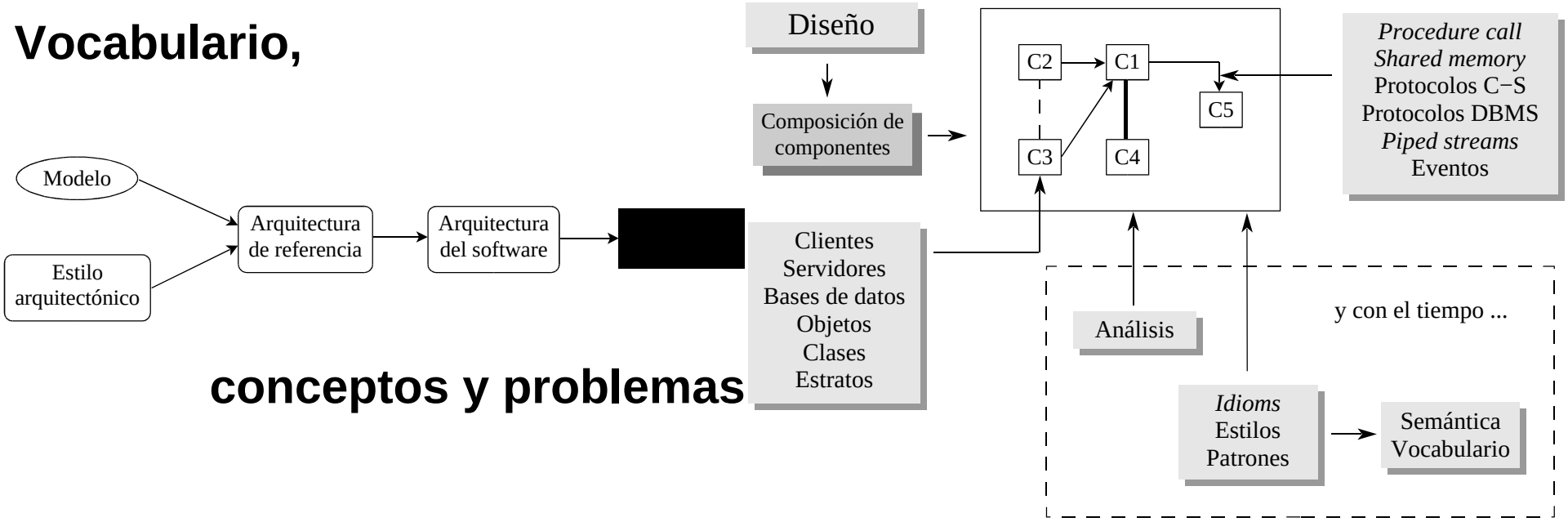
conceptos y problemas

- Consiste de cuatro componentes
- Tres de ellos pueden tener más en común
- Todos parecen estar relacionados



Típica, aunque no informativa, presentación de una arquitectura de alto nivel

- ¿Cuál es la naturaleza de los componentes?
- ¿Cuál es el significado de los arcos?
- ¿Cuál es el significado de la disposición?
- ¿Cómo opera la arquitectura en tiempo de ejecución?



Definición

Shaw Garlan

- Los elementos estructurales incluyen la organización de un sistema como la composición de componentes; estructuras de control global; protocolos de comunicación; [···] la asignación de funcionalidad a los elementos de diseño, distribución física, desempeño [···] Este es el nivel de arquitectura de software del diseño.

Bass Clements Kazman

- La arquitectura de software de un sistema de cómputo es la estructura o estructuras del sistema, que comprenden componentes de software, las propiedades de esos componentes visibles externamente, y sus relaciones.

La definición de BCK

- Propiedades visibles externamente refieren a los supuestos que ciertos componentes pueden hacer sobre otro (servicios, perfomance, manejo de fallas, etc.)
- Los sistemas pueden tener más de una estructura.
- Arquitectura
  - define componentes;
  - agrupa información sobre cómo los componentes interactúan entre sí y omite la restante;
  - está del lado público de la división producida por la interfaz, el lado privado no es arquitectura;
  - todo sistema tiene una, esto no significa que alguien la conozca;
  - lo anterior revela la diferencia existente entre la arquitectura de un sistema y su descripción o especificación;
  - el comportamiento de cada componente es parte de ella en tanto es observable o deducible desde el exterior;
  - por lo tanto, la mayoría de los diagramas de líneas−y−cuadros no describen arquitecturas;
  - la definición admite buenas y malas arquitecturas, las cuales permiten o impiden que el sistema satisfaga sus requerimientos funcionales, de perfomance y de ciclo de vida.

Entonces, ¿qué se describe?

- Propiedades extra−funcionales de los componentes
  - Perfomance, capacidad, supuestos sobre el entorno, confiabilidad, seguridad; *throughput*, robustez, requerimientos de espacio, compatibilidad con estándares, etc.
- Naturaleza de las interacciones entre componentes
  - El empaquetamiento de los componentes incluye el tipo del componente y los tipos de las interacciones que soporta. La elección es en general independiente de la funcionalidad, pero deben ser empaquetados de formas compatibles si se espera cooperación entre ellos.

Componentes y conectores

- Componentes y conectores son los bloques de construcción primarios de las arquitecturas.
- Componente: entidades computacionales activas (filtros, objetos, procesos, etc.).
- Conector: mecanismo que mediatiza la comunicación, coordinación o cooperación entre componentes; a menudo no corresponden a elementos discretos del sistema en ejecución (*rpc*, protocolos, *streams*, etc.).

Estilo, patrón o *idiom* arquitectónico

Shaw Garlan

- Caracteriza una familia de sistemas que están relacionadas por compartir propiedades estructurales y semánticas.

Bass Clements Kazman

- Descripción de tipos componente y de los patrones de interacción entre ellos. Se lo puede pensar como un conjunto de restricciones sobre una arquitectura que a su vez define una familia de aquitecturas que las satisfacen.

Estilo, patrón, *idiom*...

- Un estilo puede caracterizarse respondiendo a las siguientes preguntas:
  - ¿Cuál es el vocabulario de diseño, es decir los tipos de componentes y conectores?
  - ¿Cuáles son los patrones estructurales permitidos?
  - ¿Cuál es el modelo computacional subyacente?
  - ¿Cuáles son los invariantes esenciales del estilo?
  - ¿Cuáles son algunos ejemplos comunes de su uso?
  - ¿Cuáles son los pros y contras de usar este estilo?
  - ¿Cuáles son algunas especializaciones comunes?

Ejemplo: tubos y filtros

- Componentes: filtros
  - Cada filtro tiene un conjunto de E y uno de S
  - Cada filtro lee flujos de datos en sus E y produce flujos de datos en sus S
- Conectores: tubos
  - Transmiten la salida de un filtro como entrada de otro
  - No se efectúa ningún otro procesamiento visible

Ejemplo: tubos y filtros (2)

- Invariantes: sobre los filtros
  - Deben ser independientes, en particular no deben compartir el estado con otros
  - No conocen la identidad de sus continuadores o predecesores
  - La corrección de la salida del sistema no debe depender del orden en que actúan
- Especializaciones:
  - *Pipelines* en lugar de redes *pipe−and−filter*
  - Tubos acotadas
  - Tubos tipadas

Ejemplo: tubos y filtros (3)

- Propiedades:
  - Permite entender la E/S como composición de filtros
  - Reutilización: dos filtros cualesquiera pueden unirse
  - Mantenimiento: creación y reemplazo de filtros
  - Concurrencia
- Desventajas:
  - Llevan a organizaciones *batch* no interactivas, por la forma limitada de comunicación que tienen
  - Debe forzarse un mcd de los datos transmitidos

¿Por qué la AdS es importante?

- Comunicación entre los interesados.
- Tempranas decisiones de diseño:
  - estas son las más difíciles de tomar correctamente, de corregir y las de más largo alcance
  - define restricciones sobre la implementación
  - inhibe o habilita cualidades del software, permite predecir cualidades del sistema
  - hace más fácil razonar y administrar el cambio
- Abstracción transferible de un sistema:
  - restringe posibles reemplazos de componentes
  - restringe el vocabulario de las alternativas de diseño

¿Qué estudia?

- Lenguajes para descripciones arquitectónicas
- Codificación de la experiencia arquitectónica
- Marcos generales para dominios específicos
- Soporte formal para arquitectura de software