Universidade de São Paulo
Instituto de Ciências Matemáticas e de Computação

Arquitetura de Software: Documentação

SCE 526 – Análise e Projeto Orientados a Objeto

Profa. Elisa Yumi Nakagawa

2. Semestre de 2013

Conteúdo

- Introdução
- Princípios de Boa Documentação
- Visões arquiteturais
- Padrão IEEE 1471
- Conjuntos de visões arquiteturais
- Bibliografia

Introdução

- Muitas vezes, arquiteturas de software são criadas e não são documentadas (e, consequentemente, comunicadas) de forma efetiva, isso é, desenvolvedores e outros com interesse no sistema (*stakeholders*) não têm acesso a uma representação adequada da arquitetura.

Introdução

- Perguntado sobre: “Como vocês documentam arquitetura de software??”
- Na prática, costuma-se ouvir:
  - O que mais precisamos além do diagrama de classes?
  - Desenhamos caixas e linhas (*boxes and lines*).
  - Não nos preocupamos com isso.
  - Usamos UML.
  - Como nós documentamos o quê??!!

Introdução

- Necessidade de documentação de arquitetura:
  - Define as demais atividades que deverão ser realizadas
  - É o primeiro artefato a agregar atributos de qualidade
  - É o melhor artefato nas primeiras fases do desenvolvimento
  - Elemento chave para posterior manutenção

Introdução

- Assim, a documentação de arquitetura de software torna-se o artefato principal em todas as fases do desenvolvimento em que a arquitetura é usada.
- "*Software architecture documentation speaks for the architect, today, tomorrow, and 20 years from now.* " [SEI]

Princípios de uma Boa Documentação

- Princípios para se criar uma boa documentação:
  - **Princípio 1**: Documentar sob o ponto de vista de quem irá utilizar a documentação
  - **Princípio 2**: Evitar repetições desnecessárias. Entretanto, redundância é aceitável ou desejável se:
    - Uma informação está definida em um local X da documentação e uma elaboração ou refinamento da mesma informação aparece num local Y. Neste caso, repetir a informação (ou parte dela) no local Y é comum.
    - Duas informações são mapeadas uma para outra. É difícil fazer isso sem repetir parte delas.
    - A repetição é conveniente por motivos práticos, quando trocar de página constantemente para entender o texto fica inviável (hyperlinks, quando disponíveis, por exemplo).

Princípios de uma Boa Documentação

- Princípios para se criar uma boa documentação:
  - **Princípio 3**: Evitar ambiguidade.
  - **Princípio 4**: Usar uma organização padrão para o documento a ser criado, isto é, um modelo ou *template*.
  - **Princípio 5:** Documentar a razão para as decisões tomadas
    - As mais importantes incluem aquelas que resultaram de uma discussão longa, ou que a mudança seria onerosa, ou aquelas que são cruciais para atingir requisitos chave.
    - Deve-se documentar tanto a razão para as decisões tomadas quanto alternativas rejeitadas que sejam importantes.

Princípios de uma Boa Documentação

- Princípios para se criar uma boa documentação:
  - **Princípio 6:** manter a documentação atualizada, mas não atualizada demais.
  - **Princípio 7:** revisar a documentação criada.
    - Usuários da documentação são os melhores candidatos a revisores.

ADL (*Architecture Description Language*)

- Para a especificação de projetos arquiteturais, ADL têm sido propostas.
- Uma ADL é uma linguagem usada para representar a arquitetura de um sistema de software.
- Pode-se identificar uma diversidade de linguagens:
  - C2 [Medvidovic, 1996]
  - Unicon [Shaw, 1995]
  - Meta-H [Binn, 1993]
  - Rapide [Luckham, 1995]
  - Wright [Allen, 1997]
  - Darwin
  - ACME
  - SADL
  - Aesopa

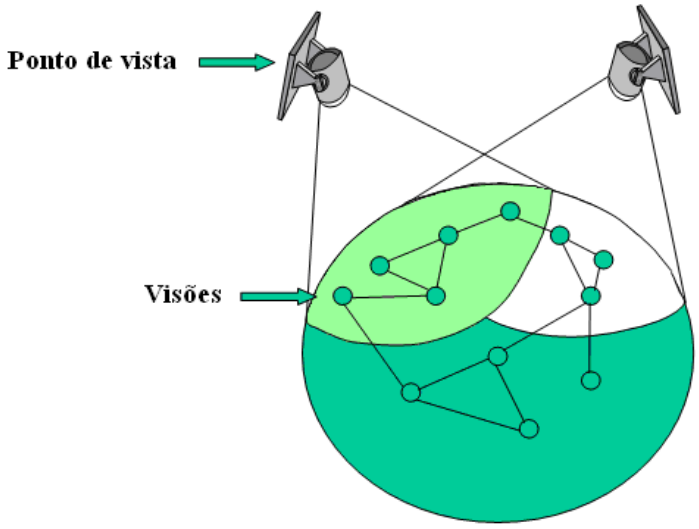
ADL (*Architecture Description Language*)

- Apesar da diversidade de ADLs:
  - ...existe pouco consenso sobre uma definição universalmente aceita pela comunidade de arquitetura de software do que seria de fato uma ADL.
- Diferentemente das linguagens de programação, as ADLs não apresentam nenhuma classificação ainda bem estabelecida na literatura

Visões Arquiteturais

- A documentação de uma arquitetura consiste de múltiplas visões arquiteturais *(architectural views)*
- Visão arquitetural é uma abstração do sistema feita a partir de um conjunto de regras estabelecidas em um determinado ponto de vista *(viewpoint).*
- Ponto de vista é a perspectiva através da qual uma dada visão do sistema é construída.

Visões Arquiteturais

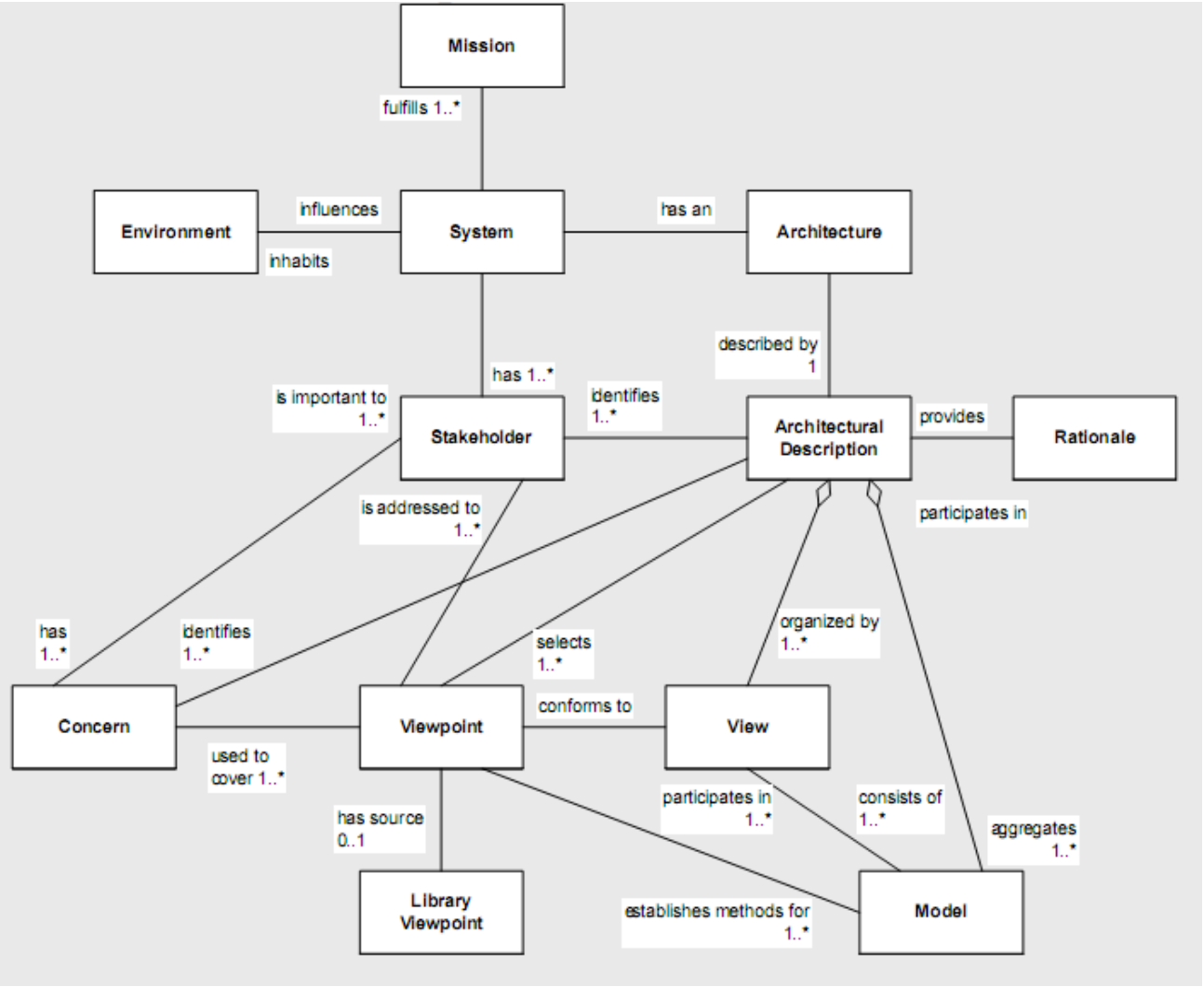


- Questões:
  - Quais visões arquiteturais são relevantes e para quê?
  - Quais notações são melhores para documentar cada visões arquiteturais?

Padrão IEEE 1471

- Motivado pela falta de um consenso sobre quais seriam as melhores visões arquiteturais, bem como as melhores técnicas para representar arquiteturas de software.
  - IEEE 1471 – *Recommended Practice for Architectural Representation of Software-Intensive Systems*
  - É atualmente a ISO/IEC 42010
  - http://www.iso-architecture.org/ieee-1471/ieee-reviewers.html

Padrão IEEE 1471



Framework conceitual da IEEE 1471 (colocar a mais recente)

Visões Arquiteturais

- Existem diversos conjuntos de visões arquiteturais propostos por diferentes autores.
- Um dos mais conhecidos é o 4+1 Views, proposto por Kruchten, 1995.
- Esse conjunto contém:
  - Visão de módulos
  - Visão em tempo de execução
  - Visão de implantação
  - Visão de implementacão
  - Visão de dados
- OBS: Nem todas as visões são relevantes para todos os sistemas.

Visões Arquiteturais

- Visão de módulos
  - Essa apresenta a estrutura do sistema em termos de unidades de implementação.
- Qual técnica usar para representar essa visão?
  - Caixas e linhas, textos ou tabelas (notação informal)
  - Diagrama de classes da UML (notação semi-formal)

Visões Arquiteturais

- Visão em tempo de execução
  - Essa visão, também chamada de visão C&C (*Component & Connector*) mostra o sistema em tempo de execução.
  - Possibilita o entendimento do funcionamento do sistema e a análise das propriedades que se manifestam em tempo de execução, tais como o desempenho.

Visões Arquiteturais

- Visão em tempo de execução
  - Possibilita apresentar:
    - os grandes componentes e seus relacionamentos
    - as bases de dados, bem como aquelas que são compartilhadas
    - os elementos replicados
    - o fluxo de dados no sistema e
    - as partes do sistema que são executadas em paralelo.
- Qual técnica usar para representar essa visão?
  - Caixas e linhas (notação informal)
  - Diagrama de componentes da UML 2.0 (notação semi-formal)

Visões Arquiteturais

- Visão de implantação
  - Mostra a estrutura de hardware (tipicamente uma rede) na qual o sistema é executado
- Qual técnica usar para representar essa visão?
  - Diagrama de redes (notação informal)
  - Diagrama de implantação (*deployment*) da UML 2.0 (notação semi-formal)

Visões Arquiteturais

- Visão de implementacão
  - Mostra a estrutura do software (como é ou como deverá ser) em termos de arquivos organizados em diretórios (pastas), tanto para o ambiente de desenvolvimento quanto de produção.
- Qual técnica usar para representar essa visão?
  - Caixas e linhas e árvores (notação informal)
  - UML 2.0 (notação semi-formal)

Visões Arquiteturais

- Visão de dados
  - É normalmente utilizada quando o sistema possui uma base de dados cuja estrutura precisa ser modelada
  - Esse modelo inicia como um modelo conceitual/lógico que vai sendo refinado até conter toda informação necessária para a criação da base de dados física.
- Qual técnica usar para representar essa visão?
  - técnicas da área de banco de dados, exemplo, MER.
  - Diagrama de classe da UML