FACVLDADE MAURÍCIO DE NASSAU
FAZENDO PARTE DA SUA HISTÓRIA

Arquitetura de Software

Conceitos Arquiteturais, componentes e conectores.

- **Qual o propósito da Arquitetura de Software?**
  - Criar soluções (simples, rápidas e eficientes);
  - Não somente resolver problemas, mas prevenir a ocorrência deles, e otimizar ainda mais soluções já existentes.

Conceitos Arquiteturais

FACVLDADE MAURÍCIO DE NASSAU
FAZENDO PARTE DA SUA HISTÓRIA

- **Software**
  - Interface homem-máquina;
  - Trabalha e opera executando algoritmos;
  - Função e desempenho;
  - Fases do Desenvolvimento:
    - Definição;
    - Desenvolvimento;
    - Verificação, liberação e manutenção.

Conceitos Arquiteturais

- **Fases do Desenvolvimento**
  1. **Definição**
     - Planejamento (plano de projeto de software);
     - Identificar quais riscos e fazer uma análise deles;
     - Finalidade do Software (o que se propõe e como);
     - Quais recursos serão necessários (humanos, materiais, custos e prazos);

- **OBS:** O objetivo geral do plano de projeto de software é fornecer uma perspectiva geral que permita analisar a viabilidade da sua execução (relação custo/benefício).

Conceitos Arquiteturais

- **Fases do Desenvolvimento**
  1. **Definição**
     - Detalhamento dos requisitos de software (protótipo, diagramas e/ou pseudocódigo);
     - Especificação dos requisitos de software;
     - Revisão da especificação dos requisitos;
     - Revisão do plano de projeto de software.

Conceitos Arquiteturais

- **Fases do Desenvolvimento**
  2. **Desenvolvimento**
     - Análise dos documentos gerados;
     - Criação da estrutura modular (interfaces e estrutura de dados);
     - Especificação do projeto (configurações do software e procedimentos que serão utilizados);
     - Codificação (linguagem de programação ou ferramentas CASE);
     - Listagem da linguagem-fonte.

Conceitos Arquiteturais

- **Fases do Desenvolvimento**

  3. **Verificação, liberação de manutenção**
     - Plano e procedimento de testes (relatos - função e desempenho). Verificação individual dos módulos;
     - Testes de integração (análise dos módulos atuando em conjunto);
     - Teste de validação (todos os requisitos foram atendidos?) - *Debugging*;
     - Controle de qualidade (controles de configuração bem estabelecidos, manual de qualidade para o usuário, etc.);
     - Manutenção e atualização.

Conceitos Arquiteturais

- **Arquitetura de Software**
  - **Estrutura interna de um determinado sistema**. Basicamente, ela explica a forma como um software se organiza e funciona, além do seu modo de implementação.
  - A evolução de um projeto torna o software:
    - Flexível (aprimoramento e adaptação);
    - Extensível (incorporação de novos elementos);
    - Portável (diferentes plataformas);
    - Reutilizável (padrões de arquitetura).

Componentes e Conectores

FACVLDADE MAURÍCIO DE NASSAU
FAZENDO PARTE DA SUA HISTÓRIA

- **Elementos da Arquitetura de Software**
  - Segundo Perry e Wolf a fórmula que define a arquitetura de software é:

    Arquitetura = (Elementos + Organização + Decisões)
  - Os elementos podem ser classificados em três tipos:
    - **Processamento** (operam os dados);
    - **Dados** (matéria-prima);
    - **Conexão** (conectam os elementos na estrutura).

Componente e Conectores

- **Componente de Software:**

  É uma entidade arquitetural que: 1) encapsula um subconjunto das funcionalidades e/ou dados do sistema; 2) restringe o acesso a este subconjunto através de interfaces explicitamente definidas; e 3) possui dependências - explicitamente definidas – em relação ao seu contexto de execução.

Componente e Conectores

- **Componente de Software:**
  - Pode ser simples como uma única operação ou complexo como um sistema inteiro;
  - É visto pelo usuário (humano ou outro software) somente através da sua interface pública;
  - São aplicações dos princípios de encapsulamento, abstração e modularidade;
  - Geralmente são aplicações específicas, mas não é sempre o caso (ex: servidores web, front-ends, back-ends, toolkits para GUI, componentes COTS, etc).

Componente e Conectores

- O tratamento explícito do contexto de execução do qual o componente depende pode informar**:**
  - **Interfaces requeridas pelo componente (*required interfaces*)**: serviços disponibilizados por outros componentes e dos quais o componente em questão depende para o seu correto funcionamento;
  - **Recursos necessários**: arquivos de dados ou diretórios necessários ao componente;
  - **Softwares do sistema requeridos**: ambientes de *runtime* plataformas de middleware, sistemas operacionais, protocolos de rede, drivers de dispositivos, etc;
  - **Configurações de hardware** necessárias para executar o componente.

Componente e Conectores

- **Conector de Software**:
  - Sistemas modernos são formados por um grande número de componentes complexos, distribuídos em múltiplos hosts (possivelmente móveis) e atualizados dinamicamente sem interrupção do serviço;
  - Nestes sistemas garantir uma interação apropriada pode ser mais importante e desafiador do que a implementação dos componentes;

Componente e Conectores

- **Conector de Software**:
  - **Elemento arquitetural responsável por efetivar e regular as interações entre componentes**;
  - Em sistemas desktop convencionais os conectores são geralmente representados por simples chamadas de procedimento (*procedure call*) ou acesso a dados compartilhados (*Shared Data Access*);
  - São constantemente não representados nas arquiteturas e se resumem a um meio de permitir a interação entre pares de componentes;
  - Entretanto, em sistemas complexos os conectores passam a ter identidades, papéis e artefatos de implementação únicos;
  - São elementos críticos, ricos e subapreciados.

Componente e Conectores

- **Tipos de Conectores**:
  - O tipo mais simples e mais amplamente utilizado de conector é o ***Procedure Call***, que permitem troca síncrona (bloqueante) de dados e controle entre pares de componentes;
  - Outro tipo comum de conector é o ***Shared Data Access***, representado por alguma forma de variável não-local ou segmentos de memória compartilhada
    - Permite que múltiplos componentes se comuniquem assincronamente através da escrita e leitura de dados na área compartilhada

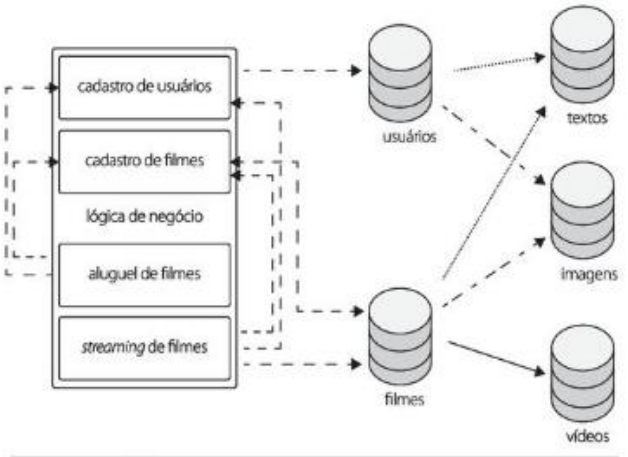
Componente e Conectores

- **Tipos de Conectores**:
  - Conectores do tipo **Distributor** encapsulam APIs para comunicação em rede, em sistemas distribuídos. Geralmente encapsulam outros conectores mais simples, como *Procedure Call*;
  - Conectores do tipo **Adaptor** são utilizados para integrar e permitir a interação entre componentes com interfaces e comportamentos incompatíveis;
  - Conectores são geralmente **aplicações independentes**;
    - Representam comportamentos já amplamente conhecidos, tais como: "*publish/subscribe*", "notificação assíncrona de eventos" e "*remote procedure call*".

Exemplos

- Na Figura abaixo podemos identificar um sistema de informação com dois grande objetivos:
  - Gerenciar o processo de locação via WEB de vídeos;
  - Proporcionar a infraestrutura de software para realizar streaming de vídeo também via WEB.

Figura 1.1 Exemplo da organização dos elementos dentro de uma estrutura.

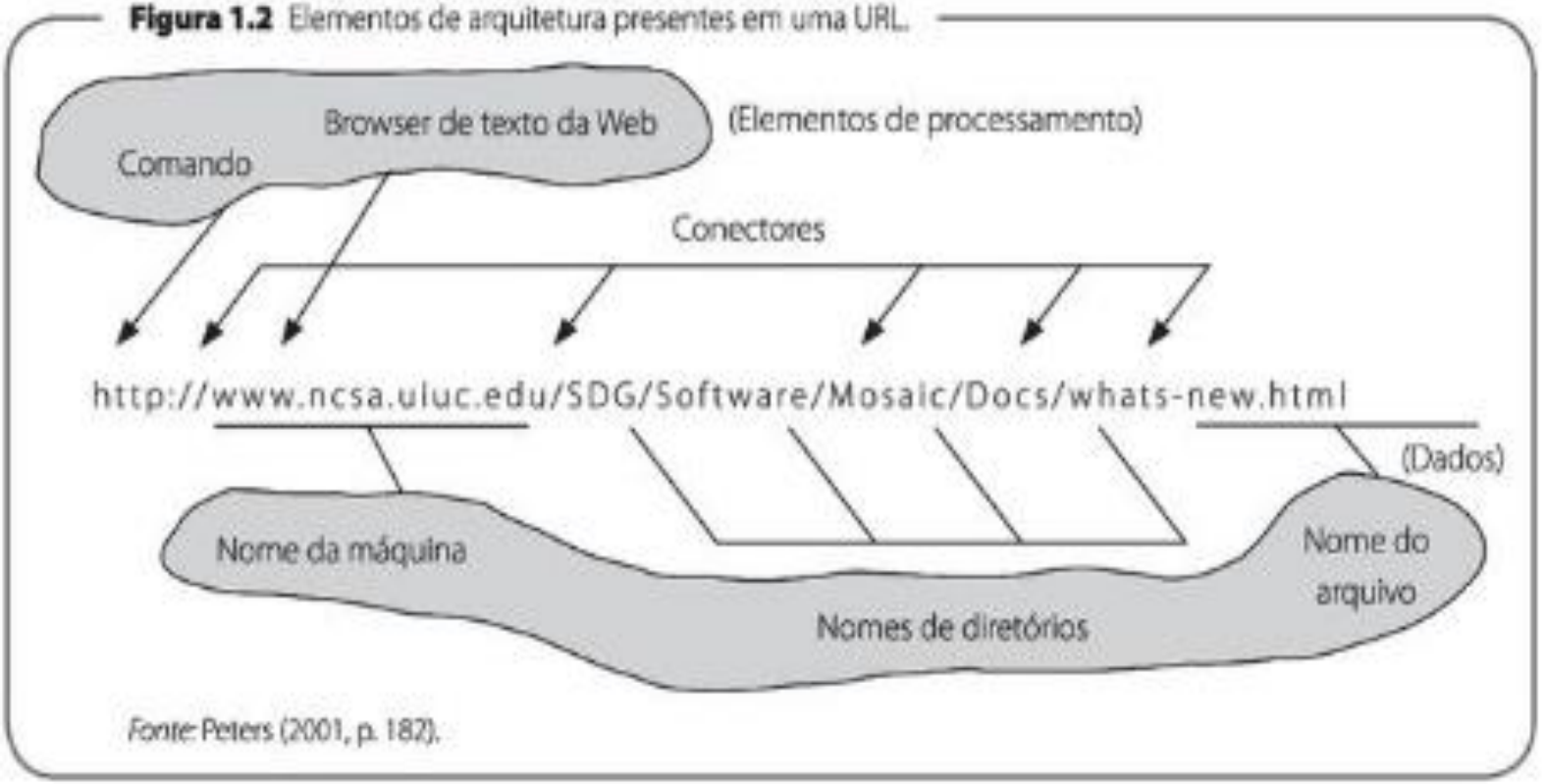


Fonte: Barbosa (2009, p. 84).

Exemplos

- **URL** (*uniform resource locator* – localizador de fonte uniforme):

Figura 1.2 Elementos de arquitetura presentes em uma URL.



Fonte: Peters (2001, p. 182).

Exemplos

- **Lista de elementos arquiteturais utilizados**:

Quadro 1.1 Exemplo de elementos comuns em arquiteturas.

Arquitetura	Exemplos de arquiteturas		
Elementos	*Uniform resource locator* (URL).	Pacote de aplicação: Microsoft Word.	Conjunto de ferramentas gráficas: Mathematica.
Componente(s) de processamento	Método de acesso: FTP (programa de transferência de arquivos); file (o mesmo que ftp); http (identifica o site da Web); telnet (conecta ao m/c).	Ortografia, gramática, dicionário de sinônimos, personalizar, contagem de palavras.	Plot [2D plot] Axeslabel Gridlines PlotRange GraphicsArray.
Dados	Nome da máquina; nome do diretório; nome do arquivo.	Nome de arquivo.	Expressão(ões). Exemplo: Seno [x(2].
Conector(es)	:// (anterior ao nome do m/c); / (anterior ao diretório ou nome do arquivo).	Botão de comando de menu suspenso.	→ PlotRange → {0 – 3,5}

Fonte: Peters (2001, p. 182).

Exercícios de Fixação

1. Explique o que é um software e qual a sua função.
2. Explique de maneira concisa o que é a fase de definição.
3. O que é o plano de projeto de software?
4. O que é a especificação de dos requisitos de software?
5. O que é especificação de projeto?
6. Explique, resumidamente o que é fase de verificação, liberação e manutenção.
7. O que é debugging?
8. Explique, resumidamente, o que significa querer que um software seja, flexível, extensível, portável e reutilizável.
9. Resuma o modelo proposto por Perry e Wolf.
10. O que são elementos de conexão?