Arquitetura de Software

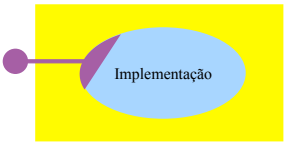
Texto complementar para este assunto:

Len Bass, Paul Clements, Rick Kazman
Software Architecture in Practice, 2nd Edition
Capítulos 1, 2 e 3

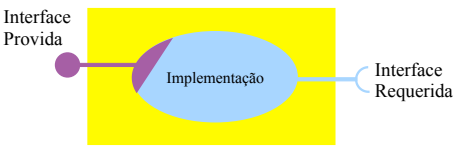
Programação Modular

Programação Modular

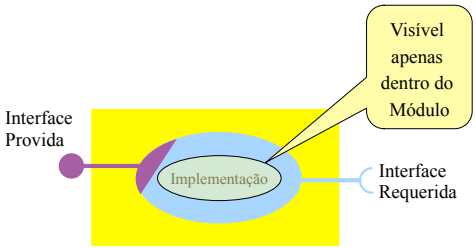
Programação Modular



Programação Modular



Programação Modular



Benefícios Esperados da Programação Modular [Parnas, 1972]

(1) Tempo de desenvolvimento encurtado, já que grupos de desenvolvimento separados podem trabalhar em módulos distintos, com pouca necessidade de comunicação

(2) Possibilidade de aplicar mudanças drásticas a um módulo sem a necessidade de mudar outros

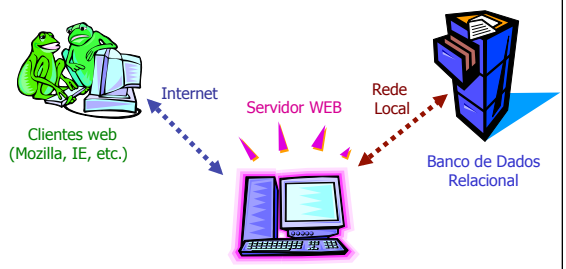
(3) Possibilidade de estudar o sistema olhando para um módulo de cada vez

- ✓ Interações entre módulos

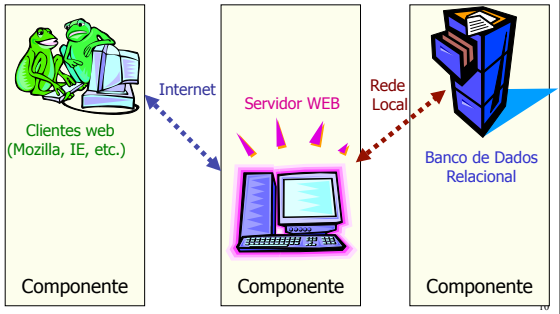
Arquitetura de Software

- A **estrutura** de um sistema de software, que engloba
  - componentes de software;
  - suas propriedades visíveis externamente;
  - e os relacionamentos e interações entre eles
- As primeiras decisões tomadas no projeto de um sistema
  - **As mais importantes!**
- Uma arquitetura de *software* é composta por **componentes** e **conectores**

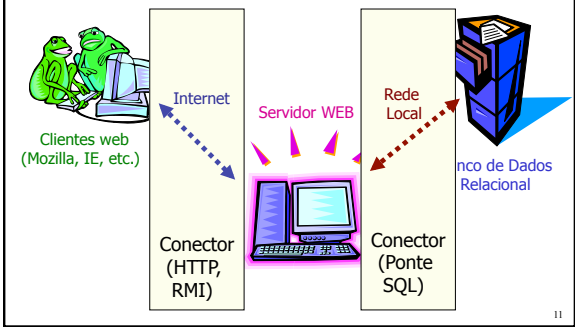
Uma Arquitetura em Camadas



Uma Arquitetura em Camadas



Uma Arquitetura em Camadas



Projeto Arquitetural

- O processo de projeto que estabelece
  - Os subsistemas que constituem um sistema
  - A maneira como esses componentes interagem
- Incluindo algumas decisões tecnológicas
  - Ex. Plataforma de componentes, SGBD
- A saída desse processo de projeto é uma descrição da arquitetura de software.
- A arquitetura de software lida com os requisitos não-funcionais do sistema

Projeto Arquitetural

- É o primeiro estágio do projeto do sistema
- Representa a ligação entre os processos de especificação e de projeto
- É freqüentemente conduzido em paralelo com algumas atividades de especificação
  - Às vezes junto com a elicitação de requisitos
- Envolve a identificação dos componentes principais do sistema e sua interação
  - Componentes => unidades de **modularidade**

Vantagens de uma Arquitetura Explícita

- Comunicação com os *stakeholders*
  - A arquitetura pode ser usada como um foco de discussão pelos *stakeholders* do sistema.
- Análise de sistema
  - Se há possibilidade de o sistema atender a seus requisitos de qualidade (não-funcionais)
- Reuso em larga escala
  - A arquitetura pode ser reusável em uma variedade de sistemas
    - Suas partes também!

Conflitos de arquitetura

- O uso de componentes de alta granularidade aprimora o desempenho mas diminui a facilidade de manutenção
- A introdução de dados redundantes aprimora a disponibilidade, mas torna a proteção mais difícil
  - E cria dificuldades para tornar o sistema confiável em outras partes
- Localizar as funcionalidades críticas de segurança em poucos locais pode criar gargalos de desempenho
- **Decisões de projeto**

Decisões de projeto

- Projeto de arquitetura é um processo criativo
  - Cada sistema envolve diferentes decisões/ requisitos/conflitos/restrições
  - Envolve solucionar os problemas representados pelos requisitos
- Decisões de projeto:
  - Escolhas feitas durante o projeto de um sistema
  - Afetam sua capacidade de fornecer seu serviço
  - Normalmente resultam em compromissos
  - É importante avaliar as opções existentes
  - Não estão restritas ao projeto arquitetural!

Exemplos de Decisões de Projeto

- Como representar o mapa em um sistema que traça rotas percorridas por ônibus de modo a minimizar o trabalho da equipe?
- Como garantir a confiabilidade de um servidor a um baixo custo?
- Qual a maneira mais eficiente de se construir uma grade de horários levando-se em conta as várias restrições impostas por professores, diretores e regras departamentais?
- Qual a melhor tecnologia para se construir uma ferramenta de análise de programas?
- Como a arquitetura do sistema deve ser documentada?
- **E nos projetos de vocês?**

Características de um Sistema que decorrem de sua Arquitetura

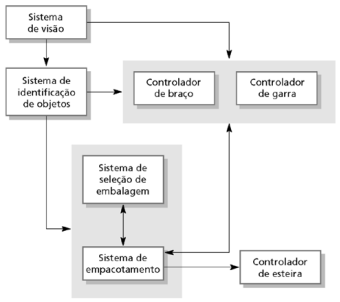
- Desempenho
  - Localizar operações críticas e minimizar comunicações. Usar componentes de alta ao invés de baixa granularidade.
- Proteção (*security*)
  - Usar uma arquitetura em camadas com itens críticos nas camadas mais internas.
- Segurança (*safety*)
  - Localizar características críticas de segurança em um pequeno número de subsistemas.
- Disponibilidade
  - Incluir componentes redundantes e mecanismos para tolerância à falhas.
- Facilidade de manutenção
  - Usar componentes facilmente trocáveis

Representação de Arquiteturas

- Arquiteturas são um ativo importante no desenvolvimento
  - Podem ser a diferença entre o sucesso e o fracasso
- Representá-las é importante
  - Torna possível "falar" sobre ela
  - O projeto de arquitetura é normalmente expresso como um diagrama de blocos
- Modelos mais específicos também podem ser desenvolvidos.

Sistema de controle robotizado de empacotamento

Figura 11.1
Diagrama de blocos de um sistema de controle robotizado de empacotamento.



Diagramas caixa e linha

- Muito abstrato – não mostram a natureza dos relacionamento de componentes, nem suas propriedades externamente visíveis
- Contudo, são úteis para comunicação com os stakeholders e para planejamento de projeto.
- Alternativas:
  - Notações formais
  - Notações informais mais organizadas

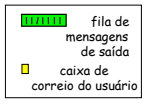
Visões Arquiteturais

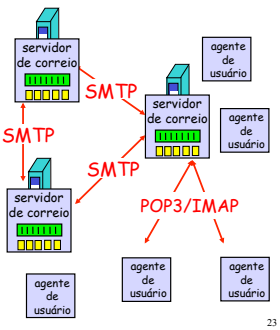
- A arquitetura de um sistema software normalmente é representada através de várias **visões**
- Visões são maneiras diversas de se enxergar uma mesma arquitetura
  - Enfocando diferentes aspectos de interesse
  - Ex.: as várias plantas de uma casa
- Arquiteturas de software são especificadas através de uma ou mais de suas visões

Correio Eletrônico – Visão 1

Três principais elementos:

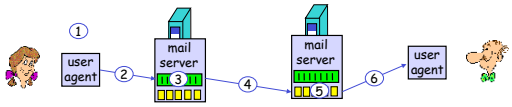
- agentes de usuário (UA).
- servidores de correio.
- simple mail transfer protocol: SMTP.



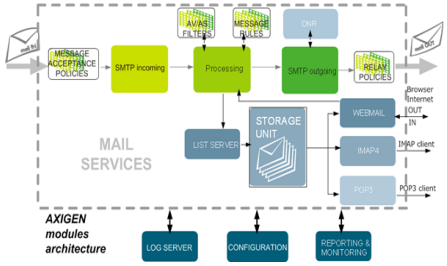


Correio Eletrônico – Visão 2

1) Alice usa o UA para compor uma mensagem "para" `bob@someschool.edu`
2) O UA de Alice envia a mensagem para o seu servidor de correio; a mensagem é colocada na fila de mensagens.
3) O lado cliente do SMTP abre uma conexão TCP com o servidor de correio de Bob.
4) O cliente SMTP envia a mensagem de Alice através da conexão TCP.
5) O servidor de correio de Bob coloca a mensagem na caixa de entrada de Bob.
6) Bob chama o seu UA para ler a mensagem.

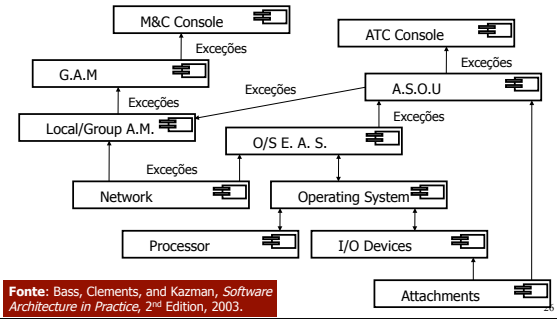


Correio Eletrônico – Visão 3



Fonte: *Axigen Mail Server Documentation - Mail Server Architecture. Consultado em 24 de março de 2008 http://www.axigen.com/docs/en/Mail-Server-Architecture_85.html*

Um Exemplo de Sistema de Controle de Tráfego Aéreo



Fonte: Bass, Clements, and Kazman, *Software Architecture in Practice*, 2nd Edition, 2003.

Sobre Visões

- Algumas são genéricas
  - Lógica
  - De interação
  - Física ou de Alocação
- As três acima deverão ser entregues no projeto da disciplina
- Outras servem a fins específicos
  - Fluxo de exceções

Reuso de arquitetura

- Sistemas do mesmo domínio freqüentemente têm arquiteturas similares que refletem os conceitos de domínio
  - Resultam em decisões de projeto similares
- Linhas do produto de software são construídas em torno de um núcleo de arquitetura
  - Variantes satisfazem requisitos de cada cliente.
- Reuso de arquiteturas é capturado através da noção de padrões ou **estilos arquiteturais**