ARCHITECTURAL DESIGN

Ian Sommerville, 8° edição – Capítulo 11

Aula de Luiz Eduardo Guarino de Vasconcelos

Objetivos

- Introduzir o projeto de arquitetura e discutir sua importância
- Explicar porque múltiplos modelos podem ser requeridos para documentar uma arquitetura de sistema
- Descrever os tipos de modelos de arquitetura que podem ser utilizados
- Discutir como modelos de referência de domínio específico podem ser utilizados como base para arquiteturas tipo produto de linha e para comparar implementações de arquitetura

Tópicos abordados

- Estruturação do sistema
- Modelos de controle
- Decomposição em módulos
- Arquiteturas de domínio específico

Arquitetura de Software

- O processo inicial de projeto para identificar subsistemas e estabelecer um *framework* para o controle e a comunicação de subsistemas
- A saída desse processo de projeto é uma descrição da *arquitetura do software*

Projeto de arquitetura

- Um estágio inicial do processo de projeto de sistema
- Representa a ligação entre os processos de especificação e projeto
- Frequentemente conduzido em paralelo com algumas atividades de especificação
- Envolve identificar os componentes de sistema principais e suas comunicações

Vantagens de arquitetura explícita

- Comunicação com os *stakeholders*
  - Arquitetura pode ser utilizada como um ponto de discussão pelos *stakeholders* do sistema
- Análise de sistema
  - Meio que auxilia a análise se o sistema pode cumprir seus requisitos não funcionais
- Reutilização em larga escala
  - A arquitetura pode ser reutilizável através de uma série de sistemas com requisitos similares

Atributos da arquitetura

- Desempenho
  - Restringir operações dentro de um pequeno número de subsistemas para reduzir a comunicação entre subsistemas
- Proteção
  - Utilizar uma arquitetura em camadas com os itens mais importantes protegidos nas camadas mais internas
- Segurança
  - Isolar componentes de segurança críticos
- Disponibilidade
  - Incluir componentes redundantes na arquitetura
- Facilidade de manutenção
  - Utilizar componentes encapsulados de menor granularidade

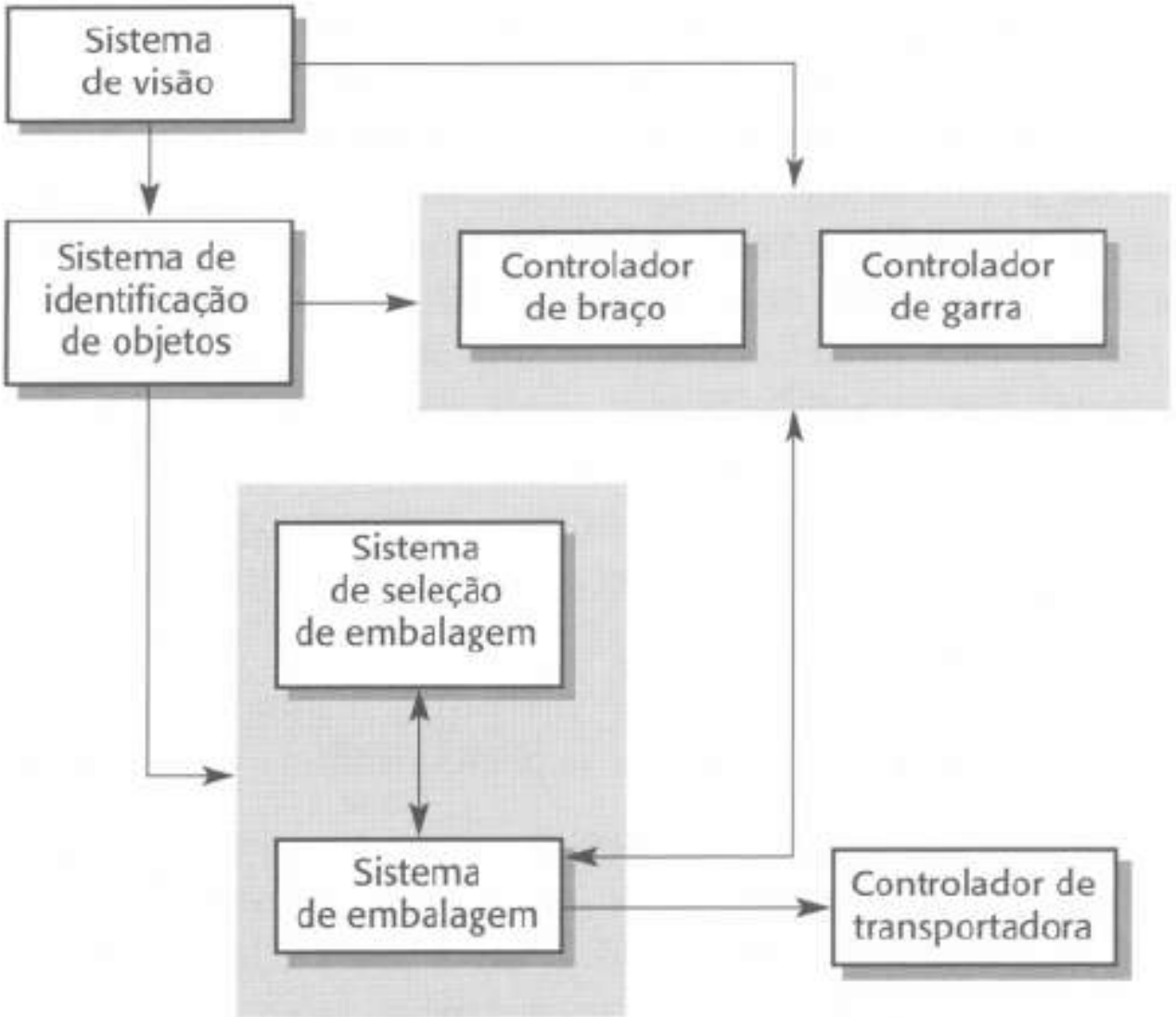
Conflitos de arquitetura

- Alta garnularidade melhora a performance mas reduz facilidade de manutenção
- Baixa granularidade facilita a manutenção mas piora desempenho
- Introduzir redundância de dados prove disponibilidade mas segurança fica dificultada

Estruturando o sistema

- Lida com a decomposição do sistema em um conjunto de subsistemas que interagem
- O projeto de arquitetura pode ser representado como um diagrama de blocos que apresenta uma visão geral da estrutura do sistema
- Modelos mais específicos, mostrando como subsistemas compartilham dados, como estão distribuídos e como atuam com interface entre si, podem ser desenvolvidos

Sistema de embalagem controlado por robô



Diagramas de caixa e linhas

- Diagramas com caixas e linhas não são representações úteis de arquitetura, pois não mostram a natureza dos relacionamentos entre os componentes do sistema nem as propriedades externamente visíveis dos componentes.
- Porém, esse tipo de modelo é eficiente para comunicação com stakeholders e para o planejamento do projeto, pois não está cheio de detalhes.
- Estes modelos de caixas e linhas não devem ser os únicos a serem utilizados, mas são úteis.
- Como decompor um sistema em subsistemas é complicado, principalmente pelo fato dos relacionamentos, mudanças de requisitos, etc

Decisões de projeto de arquitetura

- Projeto de arquitetura é um processo criativo que tenta organizar o sistema baseado nos requisitos funcionais e não funcionais, porém, as atividades do processo se diferem dependendo do tipo de sistema, origem e experiência do arquiteto do sistema e os requisitos específicos do sistema.
- Entretanto, um número de decisões comuns faz parte do projeto dos processos

Decisões de projeto de arquitetura

- Existe uma arquitetura genérica de aplicação que possa funcionar como um modelo para o sistema que está sendo projetado?
- Como o sistema será distribuído ao longo de vários processadores?
- Qual ou quais estilos de arquitetura são apropriados para o sistema?
- Qual será a abordagem fundamental usada para estruturar o sistema?
- Como as unidades estruturais de um sistema serão decompostas em módulos?
- Qual estratégia será utilizada para controlar a operação das unidades do sistema?
- Como o projeto de arquitetura será avaliado?
- Como a arquitetura do sistema será documentada?

Estilos de arquitetura

- O modelo de arquitetura de um sistema pode ser conforme um modelo de arquitetura genérica ou estilo.
- Qual estrutura mais apropriada, client-server ou em camadas.
- Decompor as unidades do sistema em componentes ou módulos.
- Sistemas grandes são heterogêneos e não seguem um único estilo de arquitetura

Modelos de arquitetura

- Usados para documentar o projeto de arquitetura
- Modelo estrutural estático que mostra os componentes do sistema principal
- Modelo de processo dinâmico que mostrar como o sistema é organizado em processos run-time (em tempo de execução)
- Modelo de interface que define as interfaces dos sub-sistemas
- Modelo de Relacionamento como o fluxo de dados entre os sub-sistemas.
- Modelo de distribuição que mostra como os sub-sistemas está distribuído nos computadores

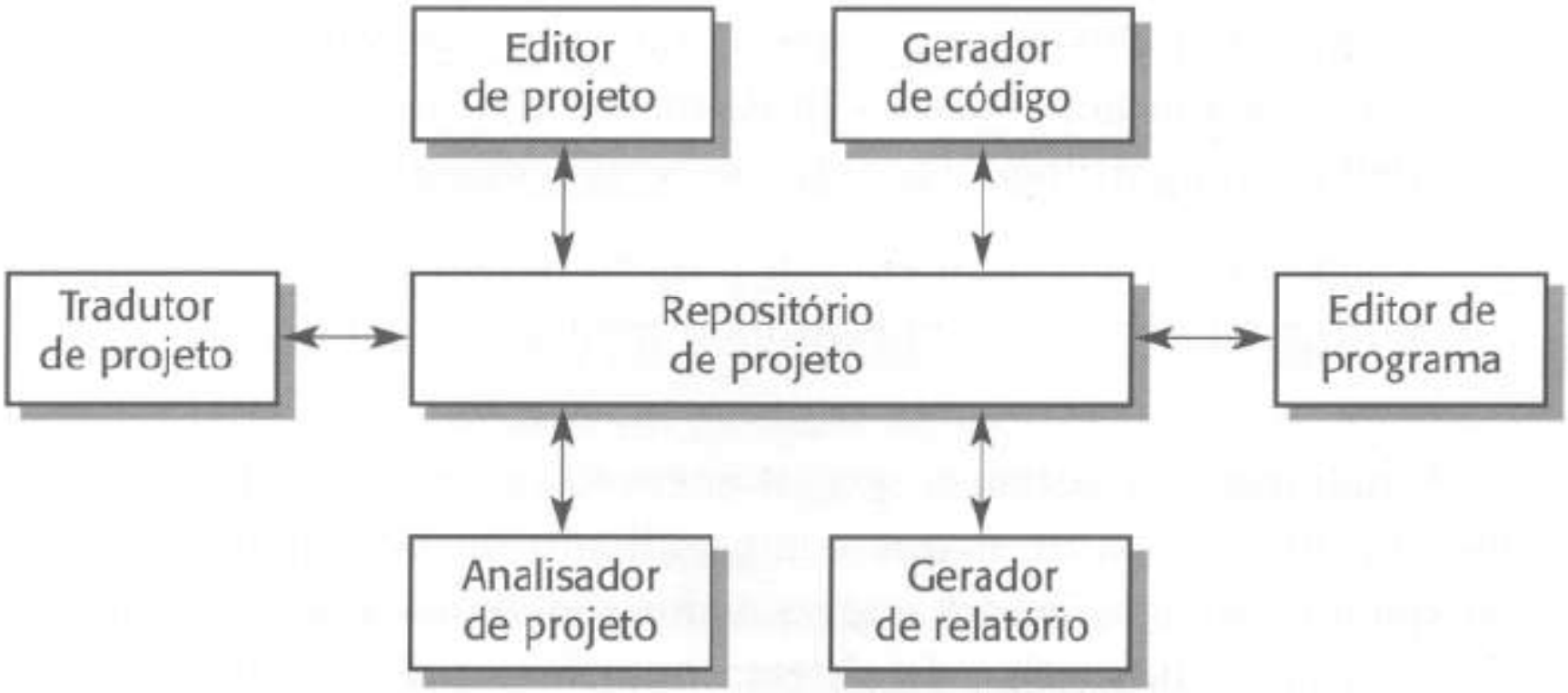
Organização do sistema

- Refelte a estratégia básica que é usada na estrutura do sistema.
- Três estilos são os mais usados:
  - Estilo de dados compartilhados em repositórios;
  - Estilo de serviços e servidores compartilhados;
  - Estilo de máquina abstrata ou em camadas.

O modelo de repositório

- Subsistemas precisam trocar dados. Isso pode ser feito de duas maneiras:
  - Dados compartilhados são mantidos em um banco de dados central, que pode ser acessado por todos os subsistemas (modelo de repositório)
  - Cada subsistema mantém seu próprio banco de dados e passa dados explicitamente para outros subsistemas
- Quando grandes quantidades de dados devem ser compartilhados, o modelo de repositório é mais comumente utilizado

Arquitetura de um conjunto de ferramentas CASE



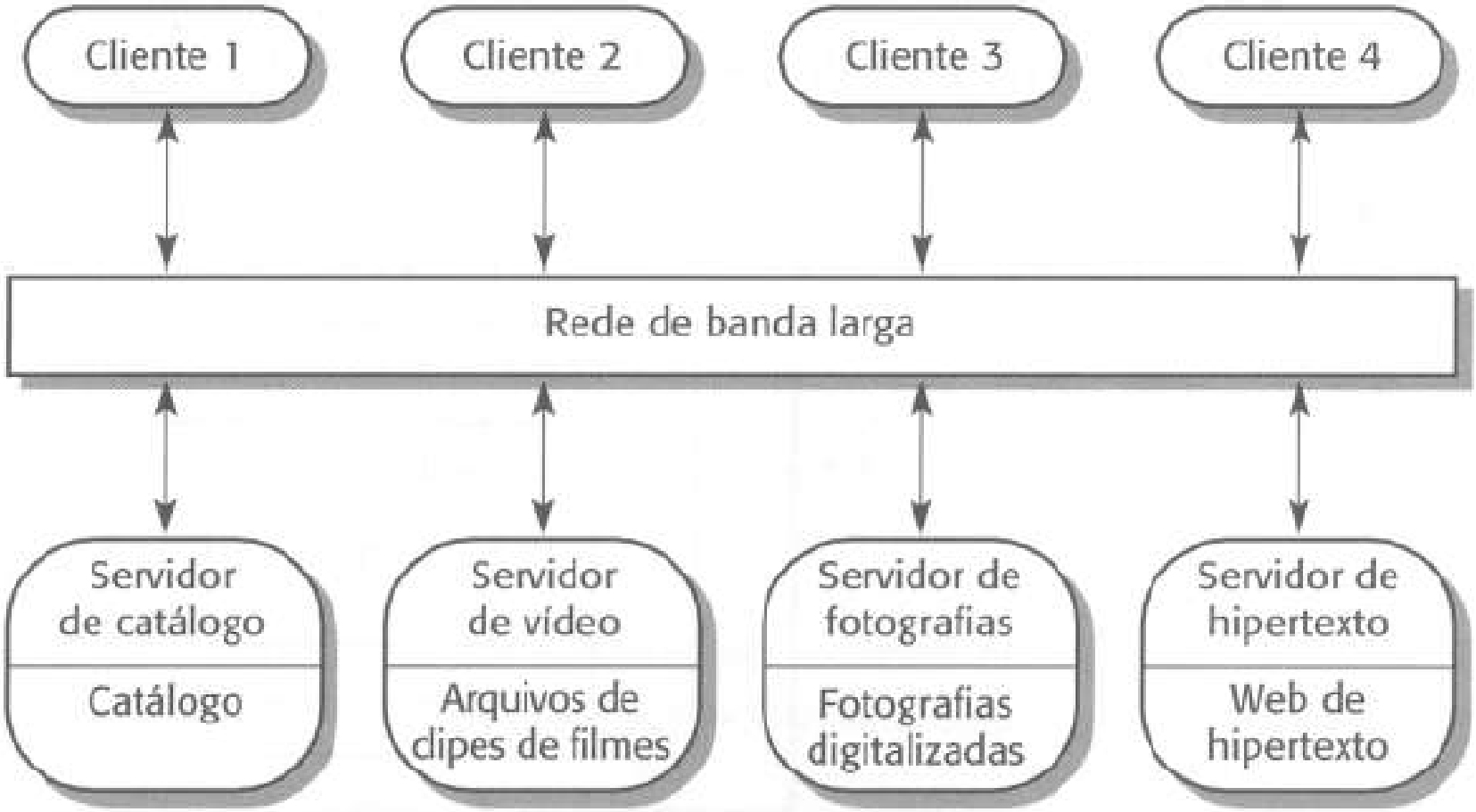
Características do modelo de repositório

- Vantagens
  - Maneira eficiente de compartilhar grandes quantidades de dados
  - Subsistemas que produzem dados não precisam saber como os dados são utilizados pelos outros subsistemas
  - Gerenciamento centralizado de backup, segurança etc
  - Modelo de compartilhamento é visível por meio do esquema de repositório
- Desvantagens
  - Subsistemas devem concordar com o modelo de dados de repositório
  - Evolução de dados é difícil e dispendiosa
  - Não há escopo para políticas específicas de gerenciamento
  - Difícil distribuir o repositório eficientemente

Modelo Client-server

- Modelo de sistema distribuído que mostra como os dados e o processamento são distribuídos em uma série de processadores
- Conjunto de servidores em *stand-alone* que oferece serviços específicos como impressão, gerenciamento de dados etc
- Conjunto de clientes que solicitam os serviços oferecidos pelos servidores
- Rede que permite aos cliente acessarem os servidores

Biblioteca de filmes e fotografias



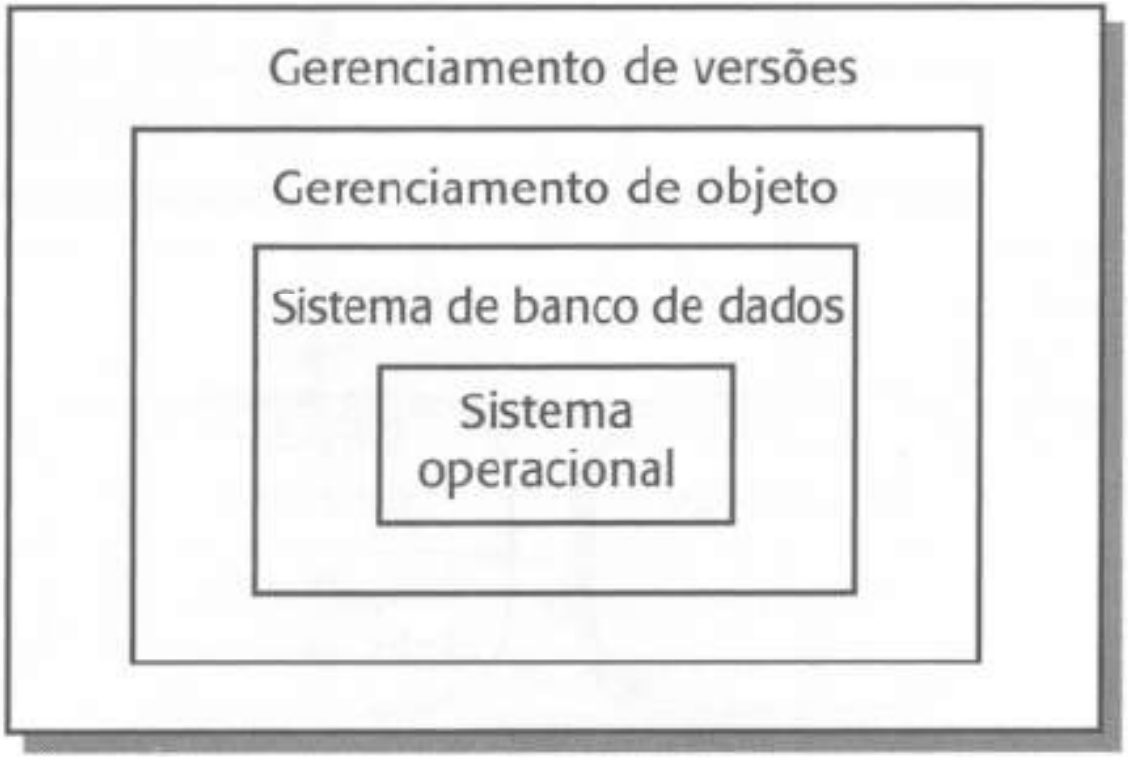
Características Client-server

- Vantagens
  - Distribuição de dados é simples
  - Permite o uso efetivo de sistemas de rede
  - É relativamente fácil adicionar novos servidores ou atualizar os servidores existentes
- Desvantagens
  - Não há um modelo de dados compartilhados de forma que cada subsistema usa uma organização de dados diferente. O intercâmbio de dados pode ser ineficiente
  - Gerenciamento de dados redundante em cada servidor
  - Não há um registro central de nomes e serviços – pode ser difícil encontrar que servidores e serviços estão disponíveis

Modelo em camadas

- Utilizado para modelar a interface de subsistemas
- Organiza o sistema em um conjunto de camadas (ou máquinas abstratas) cada uma das quais fornece um conjunto de serviços
- Apóia o desenvolvimento incremental de subsistemas em diferentes camadas. Quando a interface de uma camada muda, somente a camada adjacente é afetada
- Contudo, frequentemente é difícil estruturar sistemas dessa maneira

Sistema de gerenciamento de versões



Estilos de decomposição modular

- Qual abordagem será usada na decomposição de sub-sistemas em módulos.
- Não há distinção rígida entre organização do sistema e decomposição em módulos
- Sub-sistema é um sistema em si cuja operação não depende de serviços fornecidos por outros subsistemas. São compostos de módulos e definem interfaces para comunicar com outros sub-sistemas
- Módulo é um componente de sistema que fornece um ou mais serviços para outros módulos. Faz o uso de serviços fornecidos por outros módulos. Não é normalmente considerado como um sistema independente.

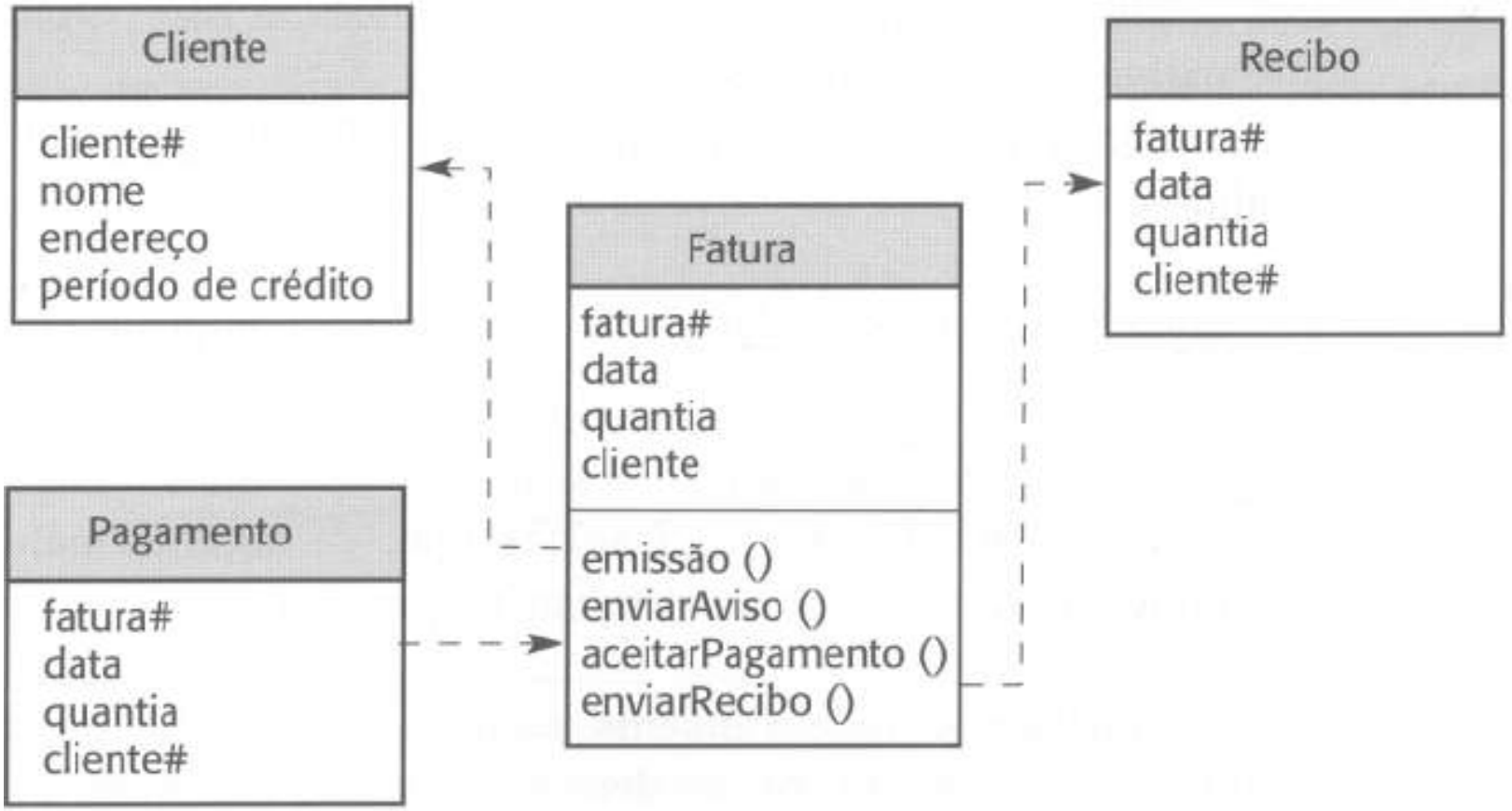
Decomposição em módulos

- Outro nível de estruturação onde subsistemas são decompostos em módulos
- Dois modelos de decomposição modular
  - Modelo orientado a objetos onde o sistema é decomposto em um conjunto de objetos que se comunicam
  - Modelo de fluxo de dados onde o sistema é decomposto em módulos funcionais que transformam entradas de dados em saídas. Também conhecido como modelo *pipeline*
- Se possível, decisões sobre concorrência devem ser postergadas até os módulos serem implementados

Modelo de Objetos

- Estruturar o sistema em um conjunto de objetos fracamente acoplados com interfaces bem definidas
- Decomposição orientada a objetos ocupa-se em identificar classes de objeto, seus atributos e operações
- Na implementação, objetos são criados a partir destas classes e algum modelo de controle pode ser utilizado para coordenar as operações dos objetos

Sistema de processamento de faturas



Vantagens do modelo de Objetos

- Objetos são fracamente acoplados e a implementação deles pode ser modificada ser afetas outros objetos.
- São representações do mundo real.
- Objetos podem ser reusados
- Linguagens de implementação OO foram desenvolvida para este fim específico.
- Entretando, a interface de objetos é a principal desvantagem e pode causar problemas complexos nos usuários dessas interfaces.

Modelo de fluxo de dados

- Transformações funcionais processam suas entradas e produzem saídas
- Pode ser chamado de modelo de duto (*pipe*) ou modelo de filtro (terminologia do *shell* do UNIX)
- Variantes dessa abordagem são muito comuns. Quando as transformações são seqüenciais, isso é um modelo em lote *(batch)* que é intensamente utilizado em sistemas de processamento de dados
- Não muito adequado para sistemas interativos

Sistema de processamento de faturas

Vantagens do modelo pipeline

- Apóia o reuso de trasnformações
- É intuitiva, pois pessoas pensam no sentido de entrada e saída.
- A evolução do sistema pela adição de novas transformações é direta.
- Fácil de ser implementada tanto quanto um sistema concorrente ou sequencial.
- A desvantagem é que ele necessita de um formato comum para trasnferência de dados que possa ser reconhecido pelas transformações

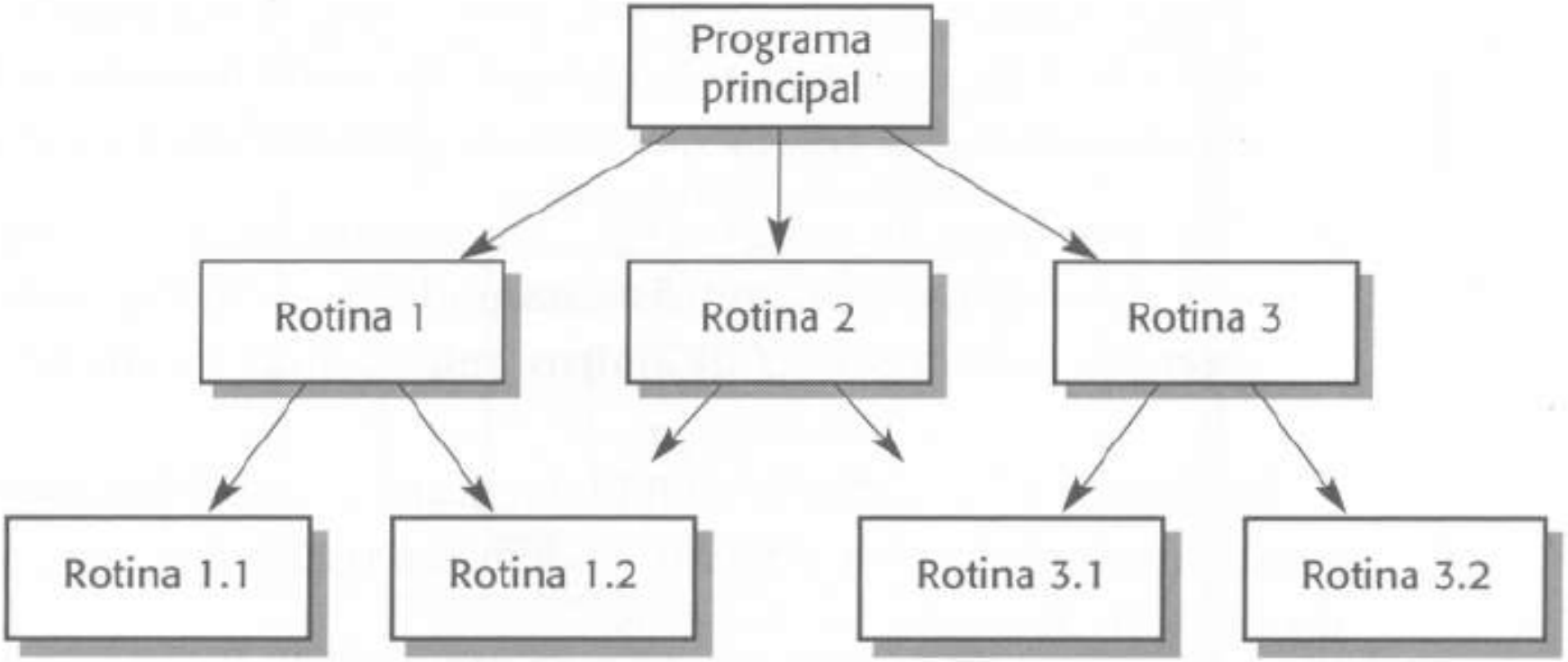
Modelos de controle

- Se ocupam com o fluxo de controle entre subsistemas, diferentemente dos modelos estruturais
- Controle centralizado
  - Um subsistema tem a responsabilidade geral pelo controle e inicia e interrompe outros subsistemas
- Controle baseado em eventos
  - Cada subsistema pode responder a eventos gerados externamente por outros subsistemas ou do ambiente do sistema

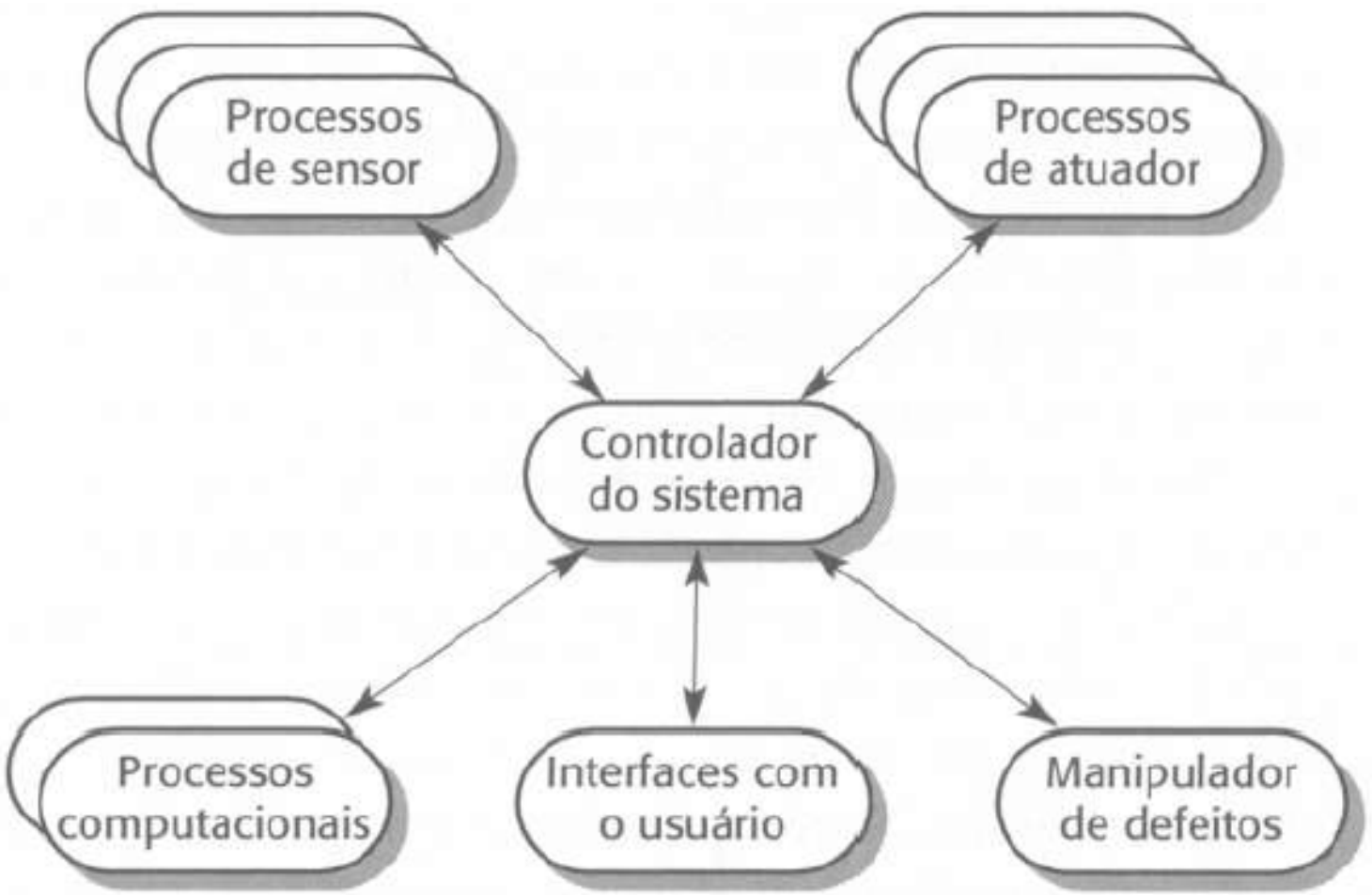
Controle centralizado

- Um subsistema de controle assume a responsabilidade de gerenciar a execução de outros subsistemas
- Modelo de retorno de chamadas
  - Modelo de subrotina onde o controle começa na parte superior da hierarquia e passa para níveis inferiores da árvore. Aplicável a sistemas seqüenciais
- Modelo gerenciador
  - Aplicável a sistemas concorrentes. Um componente de sistema controla o início, a interrupção e a coordenação de outros processos de sistema. Pode ser implementado em sistemas seqüenciais como uma declaração "CASE"

Modelo de retorno de chamada



Sistema de controle Real-time



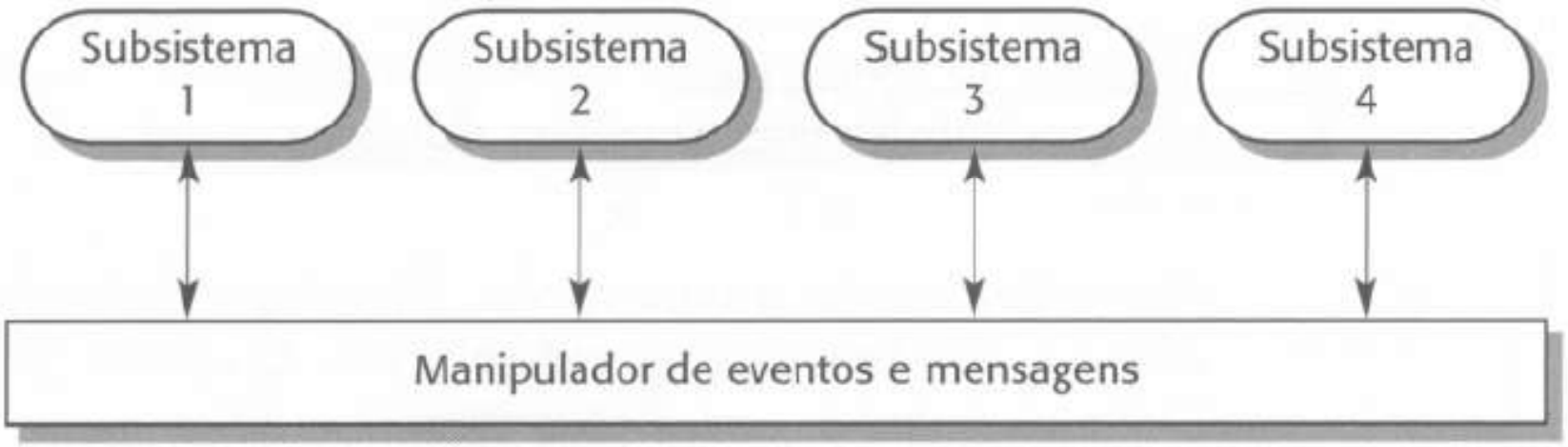
Sistema baseado em eventos

- Dirigidos por eventos gerados externamente onde a ocorrência do evento está fora do controle do processo que manipula o evento
- Dois modelos de controle orientado a eventos
  - Modelos de transmissão: um evento é transmitido para todos subsistemas. Qualquer subsistema que puder manipular esse evento responderá a ele
  - Modelos orientados a interrupções: utilizado em sistemas de tempo real em que interrupções são detectadas por um manipulador de interrupções e são passadas para algum outro componente para processamento
- Outros modelos orientados a eventos incluem planilhas eletrônicas e sistemas de produção com base em regras (AI)

Modelos de transmissão

- Eficaz na integração de subsistemas em diferentes computadores em rede
- Subsistemas registram o interesse em eventos específicos. Quando estes ocorrem, o controle é transferido para o sistema que pode manipular o evento
- Política de controle não está incluída no evento e no manipulador de mensagem. Subsistemas decidem quais eventos são requeridos
- Contudo, os subsistemas não sabem se ou quando um evento será manipulado

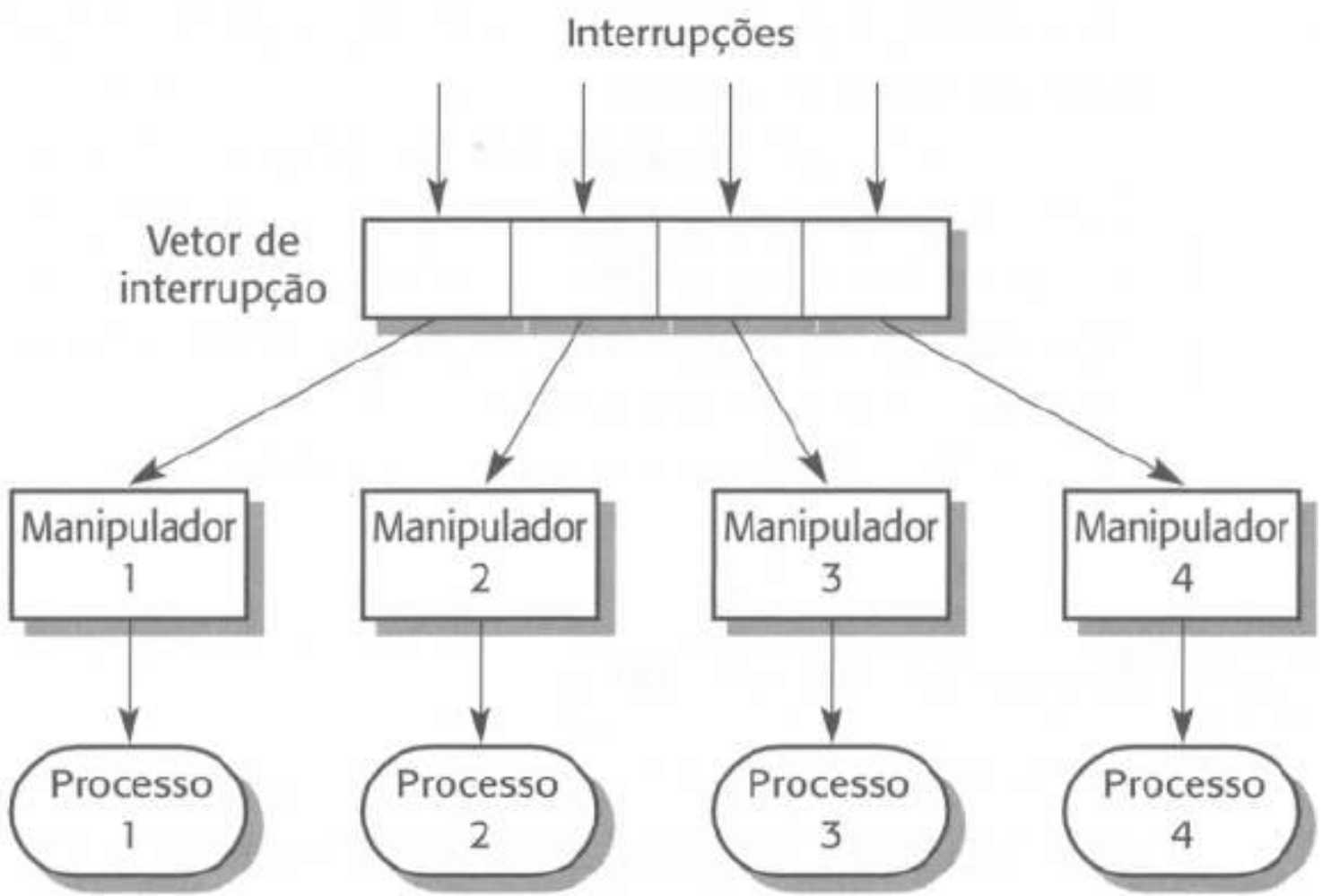
Transmissão (broadcasting) seletiva



Sistema dirigido por interrupções

- Utilizado em sistemas de tempo real em que a resposta imediata a algum evento é essencial
- Existe um número conhecido de tipos de interrupção com um manipulador definido para cada um deles
- Cada tipo é associado com uma localização de memória e um comutador de *hardware* provoca a transferência de controle imediatamente para seu manipulador
- Permite resposta rápida mas sua programação e validação são muito complexas

Controle dirigido por interrupções



Arquiteturas de domínio específico

- Modelos de arquitetura que são específicos de um determinado domínio de aplicação
- Dois tipos de modelo de domínio específico
  - Modelos genéricos que são abstrações a partir de uma série de sistemas reais e que encapsulam as características principais desses sistemas
  - Modelos de referência que são modelos mais abstratos e idealizados. Fornecem um meio de informação sobre determinada classe de sistemas e de comparação de diferentes arquiteturas
- Modelos genéricos são usualmente modelos *bottom-up*; Modelos de referência são modelos *top-down*

Arquiteturas de Referência

- Modelos de referência são derivados de um estudo do domínio da aplicação em vez dos sistemas existentes
- Pode ser utilizado como uma base para a implementação de sistema ou para comparar diferentes sistemas. Atua como um padrão em relação ao qual os sistemas podem ser avaliados
- Modelo OSI é um modelo em camadas para interconexão de sistemas

Modelo de Referência OSI

Modelo de referência Case

- Serviço de repositório de dados
  - Armazena e gerencia os dados
- Serviços de integração de dados
  - Gerenciamento de grupos.
- Serviços de gerenciamento de tarefas
  - Definição e aprovação de modelos de processos.
- Serviços de mensagem
  - Comunicações Tool-tool and tool-environment, apoiando a integração de controles.
- Serviços de interface com usuários
  - Desenvolvimento de interfaces.

The ECMA reference model (CASE)

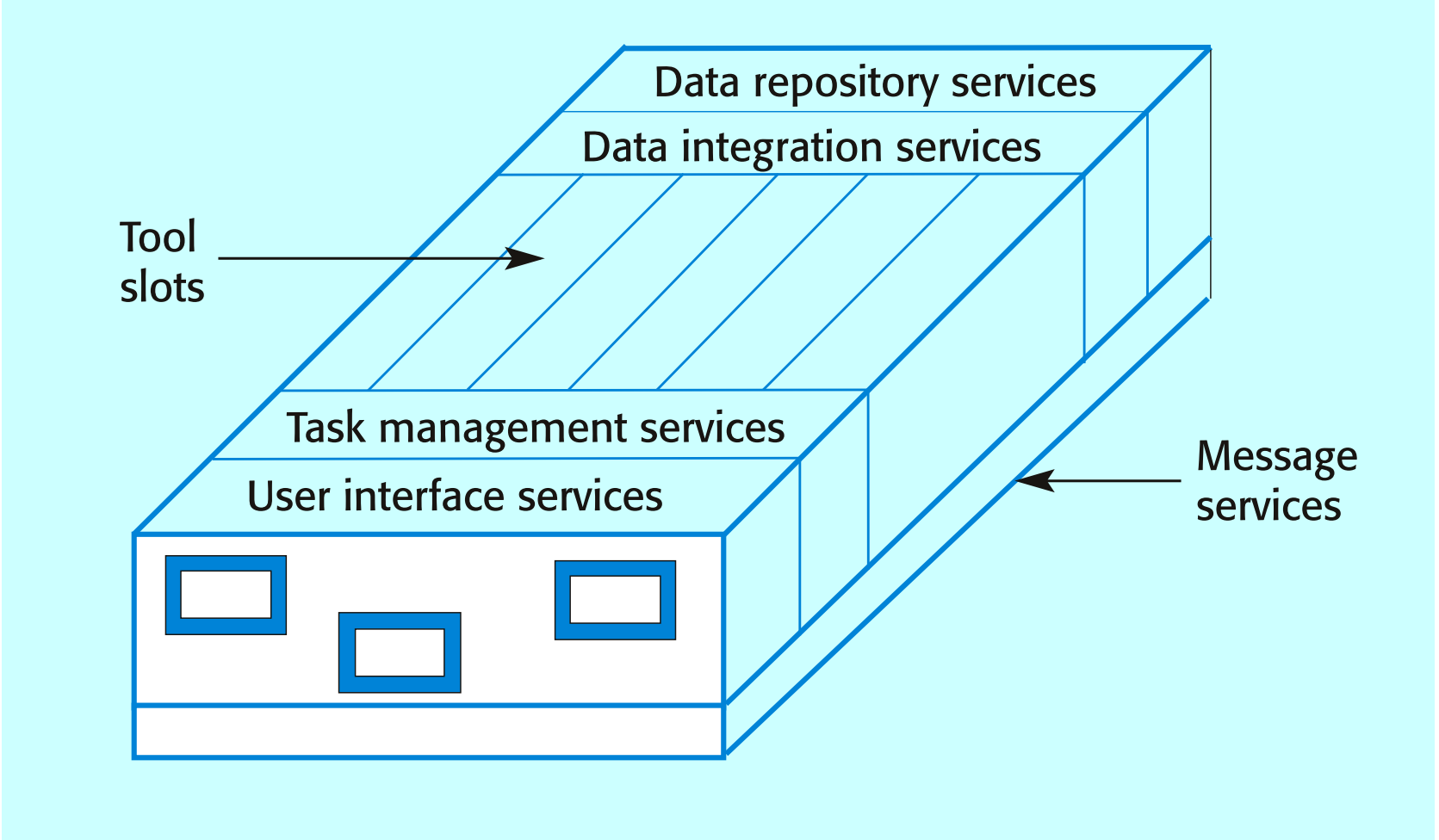
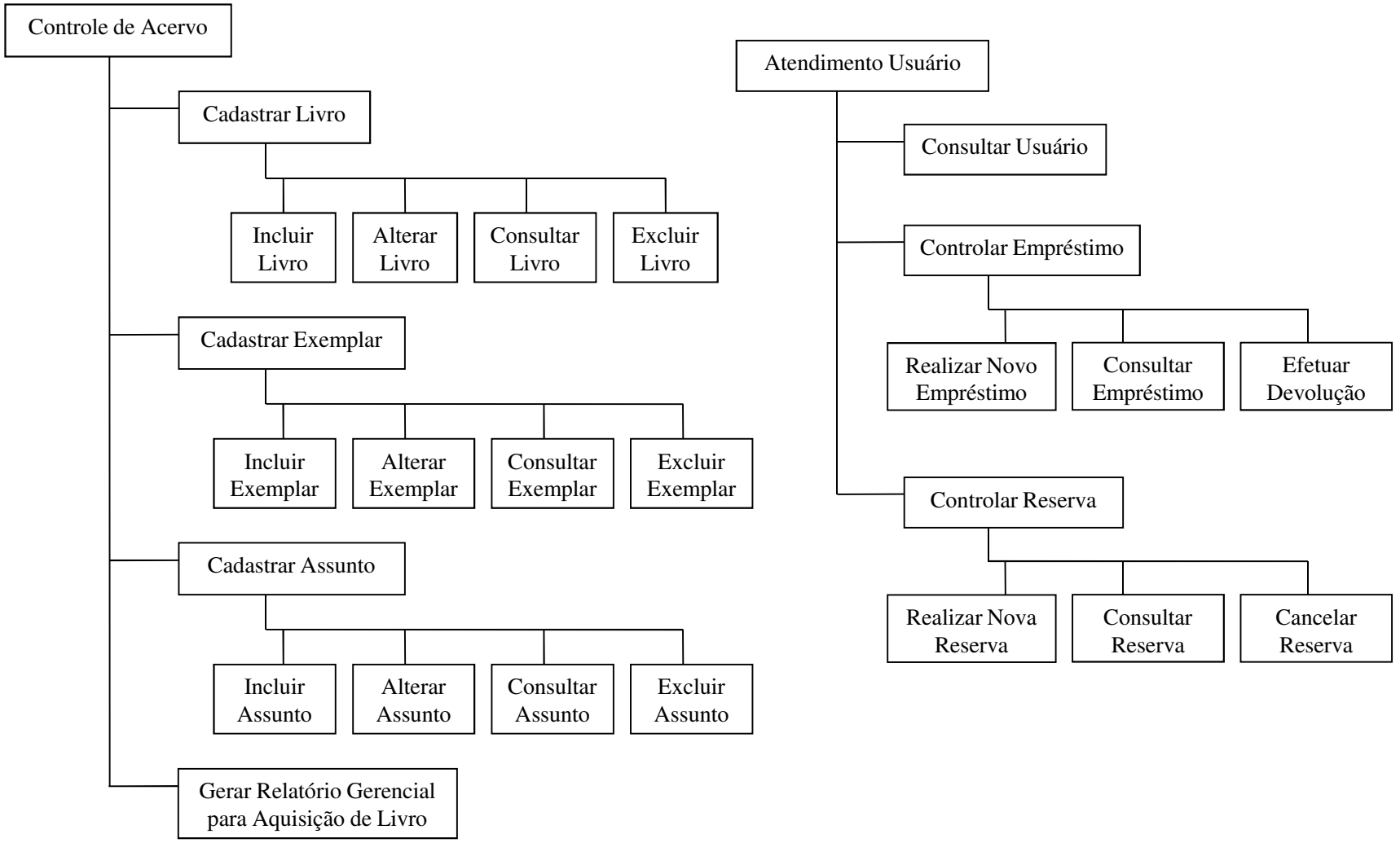


Diagrama hierárquico de Funções (DHF)



Pontos-chave

- Arquitetura de software é o framework fundamental para estruturar um sistema.
- Decisões de projeto de arquitetura incluem decisões sobre a arquitetura da aplicação, distribuição e estilos a serem usados.
- Os diferentes modelos de arquitetura, como um modelo de estrutura, de controle e de decomposição podem ser desenvolvidos.
- Modelos organizacionais de sistema incluem repositórios, client-server e modelo de camadas

Pontos-chave

- Modelo de decomposição modular inclui modelo de objeto e modelo pipelining.
- Modelos de controle incluem controle centralizado e modelos dirigidos a eventos.
- Arquiteturas de referência podem ser usadas como um meio de discutir arquiteturas de domínio específico e avaliar e comparar projetos de arquitetura