Padrões de Arquitetura

Prof. Marum Simão Filho

Agenda

- Padrões Arquiteturais - Motivação
- Definição
- Classificação
- Descrevendo Padrões
- Padrão Camadas
- Padrão MVC
- Padrão Broker
- Padrão Pipes e Filtros

Motivações para o estudo de Arquiteturas de Software

- Sistemas cada vez maiores;
- Sistemas cada vez mais complexos;
- Mais requisitos em termos de confiabilidade;
- Mais requisitos em termos de desempenho;
- Mais requisitos em termos de economia;
- Necessidade de manutenibilidade – facilidade de reparo e evolução
- Tendência à componentização;
- Busca pela reusabilidade.

Definição

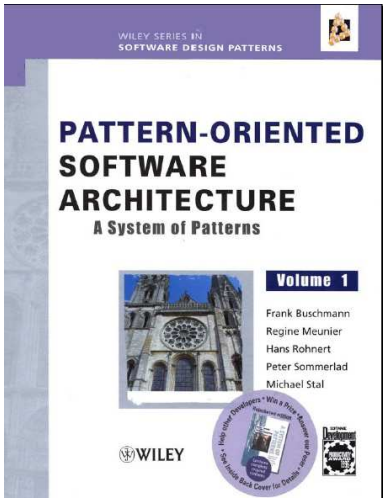
- David Garlan e Dewayne Perry, do Software Engeneering Institute, da Carnegie Mellon University, definiram arquitetura de software como:
- *A estrutura dos componentes de um programa/sistema, seus inter-relacionamentos, princípios e diretrizes guiando o projeto e evolução ao longo do tempo.*

Questões estruturais da Arquitetura de Software

- Seleção de alternativas de projeto;
- Escalabilidade e desempenho;
- Organização e estrutura geral de controle;
- Protocolos de comunicação, sincronização;
- Atribuições de funcionalidade a componentes de projeto.

Padrões Arquiteturais POSA

- Padrões **POSA** - **P**attern-**O**riented **S**oftware **A**rchitecture: A System of Patterns.
- Buschmann, Frank; Meunier Regine; Rohnert, Hans; Sommerlad, Peter; Stal, Michael.
- Classifica os padrões em 3 categorias:
  - Padrões Arquiteturais
  - Padrões de Projeto
  - Idiomas



Padrões Arquiteturais POSA

- Padrões de Arquiteturais
  - Expressam um esquema de organização estrutural para sistemas de software.
  - Oferecem um conjunto de subsistemas pré-definidos, especifica suas respectivas responsabilidades e inclui regras e diretrizes para organizar as relações entre eles.
  - São considerados padrões de alto nível.

Principais Padrões Arquiteturais

- Estruturais *(From Mud to Structure)* – padrões nesta categoria ajudam você a evitar um "mar" de componentes ou objetos. Em particular, eles suportam uma decomposição controlada de uma tarefa global do sistema em sub-tarefas cooperativas.
- A categoria inclui os seguintes padrões:
  - Camadas
  - Pipes e Filtros
  - Quadro-Negro

Principais Padrões Arquiteturais

- Sistemas Distribuídos – esta categoria compreende macro-soluções para sistemas baseados em distribuição.
- O padrão abaixo se enquadra nessa categoria:
  - Broker

Principais Padrões Arquiteturais

- Sistemas Interativos – esta categoria compreende dois padrões que suportam a estruturação de sistemas de software que trabalham a interação homem-máquina.
- Os padrões dessa categoria são:
  - Model-View-Controller (MVC)
  - Presentation-Abstraction-Control (PAC)

Principais Padrões Arquiteturais

- Sistemas Adaptativos – os padrões nessa categoria suportam a extensão de aplicações e suas adaptações às tecnologias em evolução, bem como às mudanças dos requisitos funcionais.
- Os padrões dessa categoria são:
  - Microkernel
  - Reflection

Descrevendo Padrões Arquiteturais

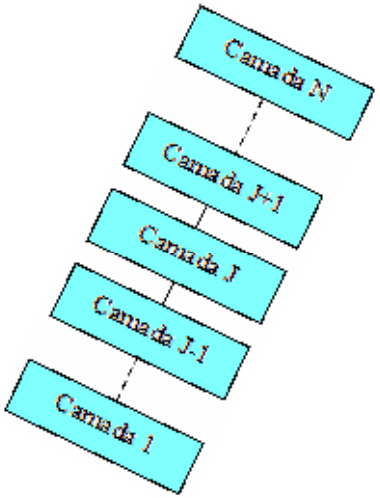
- Nome
  - O nome e um curto resumo do padrão.
- Exemplo
  - Um exemplo do mundo real demonstrando a existência do problema e a necessidade do padrão.
- Contexto
  - As situações nas quais o padrão se aplica.
- Problema
  - Descreve o problema que o padrão soluciona, incluindo uma discussão das forças associadas (restrições, imposições e demais questões associadas ao problema).

Descrevendo Padrões Arquiteturais

- Solução
  - O princípio básico da solução associado ao padrão.
- Estrutura
  - Uma especificação detalhada dos aspectos estruturais do padrão, usando diagramas OMT ou UML.
- Usos conhecidos
  - Fornece exemplos de aplicação do padrão encontrados em sistemas reais.

Padrão

Camadas



Padrão de Camadas

- Exemplo
  - Modelo OSI em 7 Camadas de Protocolo de Redes

Padrão de Camadas

- Contexto
  - Um grande sistema que requer composição.
- Problema
  - Imagine um sistema cuja característica dominante é uma mistura de questões de alto e baixo nível, onde operações de alto nível confiam nas operações de baixo nível.
  - Algumas partes do sistema tratam questões de baixo nível, tais como aspectos de hardware, entrada por sensores, leitura de bits de um arquivo sinais elétricos de um fio.
  - No final do espectro deve existir a funcionalidade visível ao usuário, tal como interface de um jogo multi-usuário ou políticas de alto nível tais como tarifas de bilhetagem telefônica.

Padrão de Camadas

- Problema (cont.)
  - Um padrão típico de fluxo de comunicação consiste de requisições movendo-se do nível superior para o inferior, e respostas às requisições, dados de entrada ou notificações sobre eventos viajando na direção oposta.
  - Portabilidade para outras plataformas é desejada.
  - O trabalho tem que ser subdividido em equipes com especificidades distintas.

Padrão de Camadas

- Solução
  - De um ponto de vista de alto nível, a solução é extremamente simples.
  - O sistema deve ser estruturado em um número apropriado de camadas, colocadas umas sobre as outras.
  - Inicie no nível mais baixo de abstração – chame-a de "Camada 1". Esta é a base de seu sistema.
  - Trabalhe seguindo para cima no nível de abstração colocando a Camada J sobre a Camada J-1 até alcançar o nível mais elevado de funcionalidade – chame-a de "Camada N".
  - Os serviços da Camada J são usados somente pela Camada J+1.

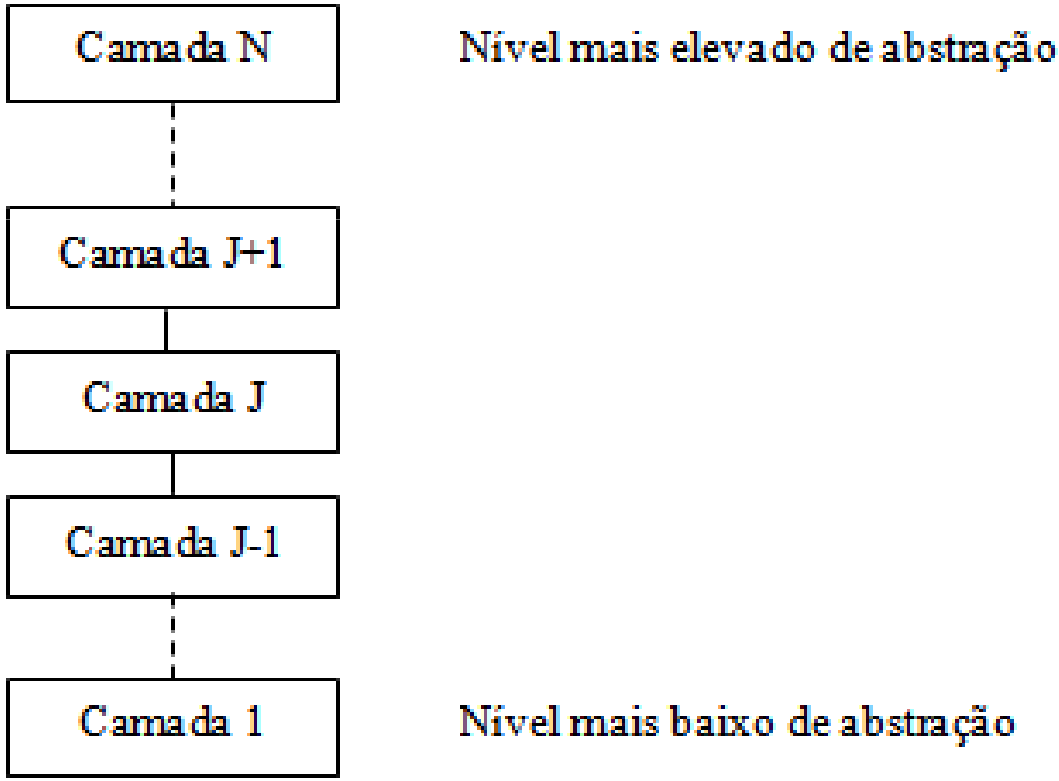
Padrão de Camadas

- Estrutura

Classe:	Colaborador:
• Camada J	• Camada J-1
Responsabilidade: • Provê serviços usados pela Camada J+1. • Delega sub-tarefas para a Camada J-1.	

Padrão de Camadas

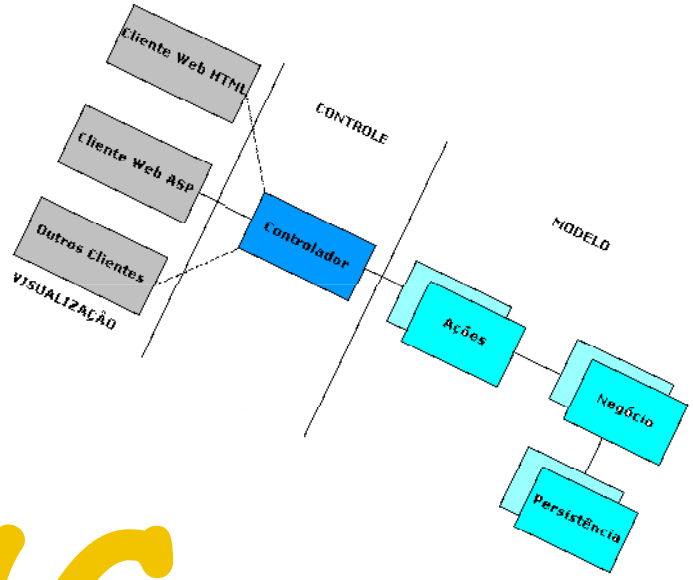
- Estrutura (cont.)



Padrão de Camadas

- Usos Conhecidos
  - Máquinas Virtuais, como Java Virtual Machine – JVM.
  - Applicatton Programming Interface – API.
  - Sistemas de Informação em geral: Apresentação, Lógica da Aplicação, Camada de Domínio e Banco de Dados.
  - Sistemas Cliente-Servidor.
  - Windows NT: Serviços de Sistema, Camada de Gerenciamento de Recursos, Kernel, Camada de Abstração do Hardware e o próprio Hardware.

Padrão MVC

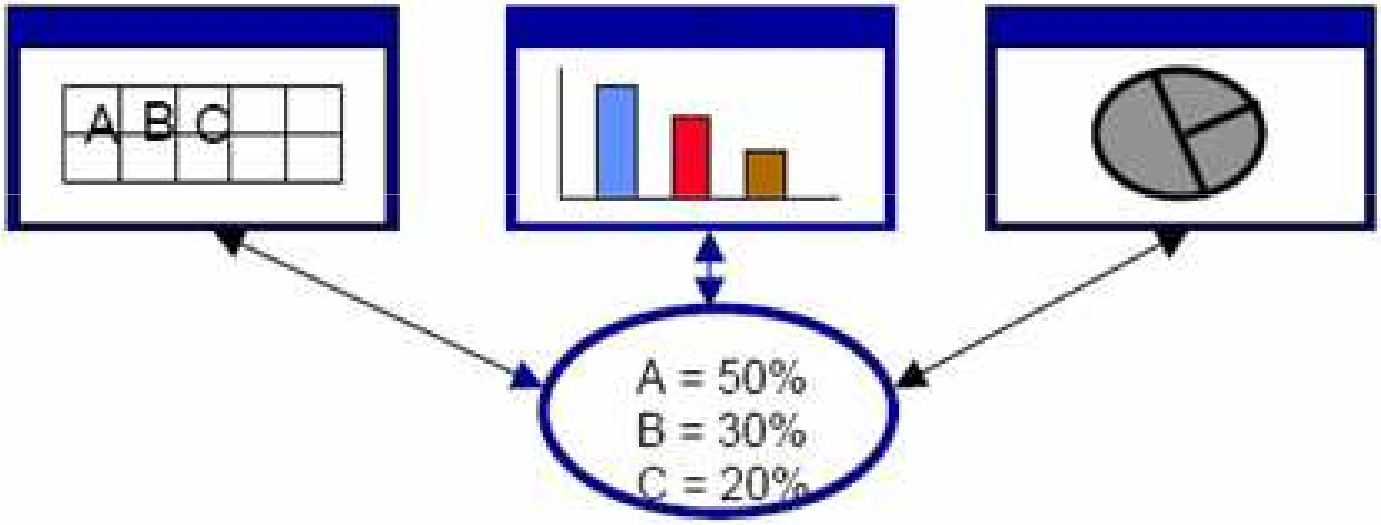

Cliente Web HTML
Cliente Web ASP
Outros Clientes
VISUALIZAÇÃO
CONTROLE
Controlador
MODELO
Ações
Negócio
Persistência

Padrão MVC

- Exemplo
  - Considere um sistema de informação simples para eleições políticas com representação proporcional.
  - Ele oferece uma planilha para entrada de dados e vários tipos de tabelas e gráficos para apresentar os resultados atuais.
  - Usuários podem interagir com o sistema através de uma interface gráfica.
  - Todo mostrador de informação deve refletir mudanças que ocorrerem nos dados de votação imediatamente.

Padrão MVC

- Exemplo



Padrão MVC

- Contexto
  - Aplicações interativas com uma interface homem-computador flexível.

Padrão MVC

- Problema
  - A mesma informação é apresentada diferentemente em várias janelas, por exemplo, em gráficos de barra e gráficos de pizza.
  - A visualização e o comportamento da aplicação devem refletir as manipulações de dados imediatamente.
  - Mudanças na interface ao usuário devem ser fáceis, e sempre possíveis em tempo de execução.
  - Suportar diferentes padrões "look and feel" e portar a interface ao usuário não devem afetar o código do núcleo (do negócio) da aplicação.

Padrão MVC

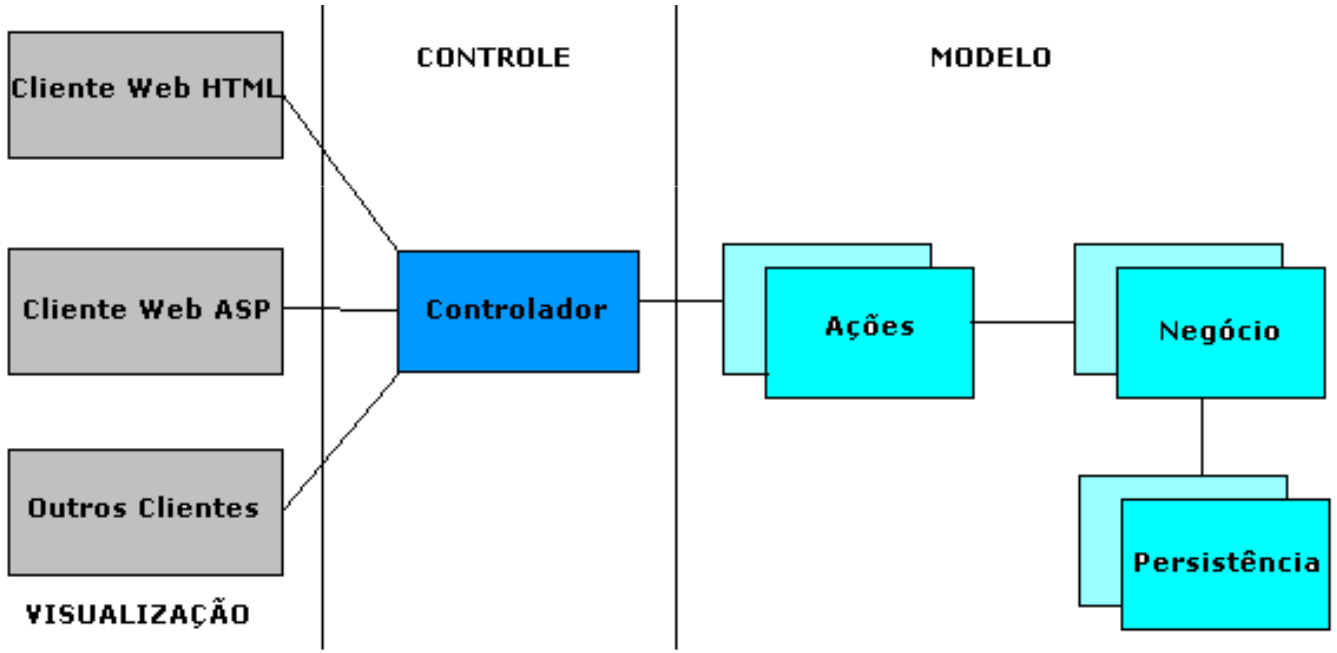
- Solução
  - MVC (Model-View-Controller) divide uma aplicação interativa em três áreas: entrada, processamento e saída.
  - O componente de modelo (*model*) encapsula dados e funcionalidade centrais.
  - O modelo é independente de representações de saída específicas ou de comportamento de entrada.
  - Componentes de visão (*view*) mostram informação ao usuário.

Padrão MVC

- Solução
  - Uma *view* obtém os dados do modelo. Podem existir múltiplas *views* para o mesmo modelo.
  - Cada *view* tem um componente controlador (*controller*) associado.
  - *Controllers* recebem entrada geralmente como eventos de movimentos de mouse, ativações de botões ou entradas de teclado.
  - Eventos são traduzidos em requisições de serviços para o modelo ou para a visão. O usuário interage com o sistema somente através dos *controllers*.

Padrão MVC

- Solução



Padrão MVC

- Estrutura
  - Modelo (*model*)
    - Provê o núcleo funcional da aplicação.
    - Registra visões e controladores dependentes.
    - Notifica componentes dependentes sobre mudanças de dados.
  - Visão (*view*)
    - Cria e inicializa seu controlador associado.
    - Exibe informações ao usuário.
    - Implementa o procedimento de atualização.
    - Busca dados do modelo.

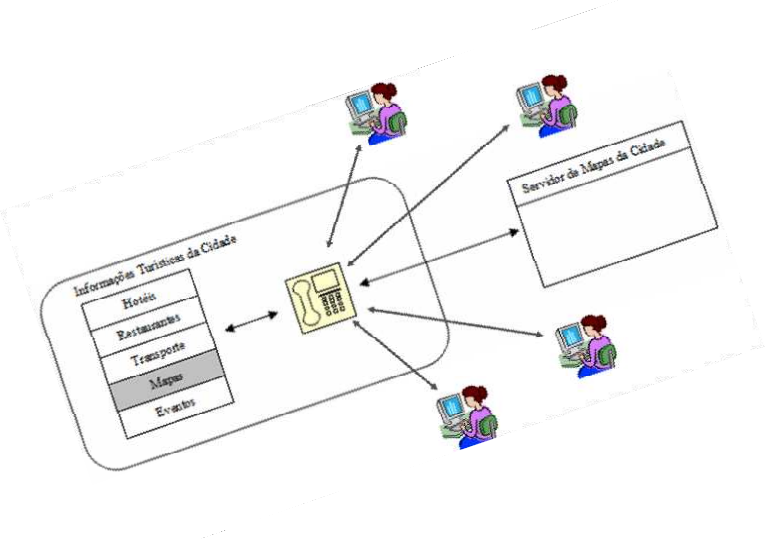
Padrão MVC

- Estrutura
  - Controlador (*controller*)
    - Recebe entrada dos usuários como eventos.
    - Traduz eventos em requisições de serviços para o modelo ou requisições de apresentação para a visão.
    - Implementa o procedimento de atualização, se solicitado.

Padrão MVC

- Usos Conhecidos
  - Smaltalk foi o primeiro exemplo padrão MVC
  - Jakarta Struts
  - WebWork
  - Ruby on Rails
  - Symfony
  - Prado
  - Diversos *frameworks* de mercado.

Padrão Broker

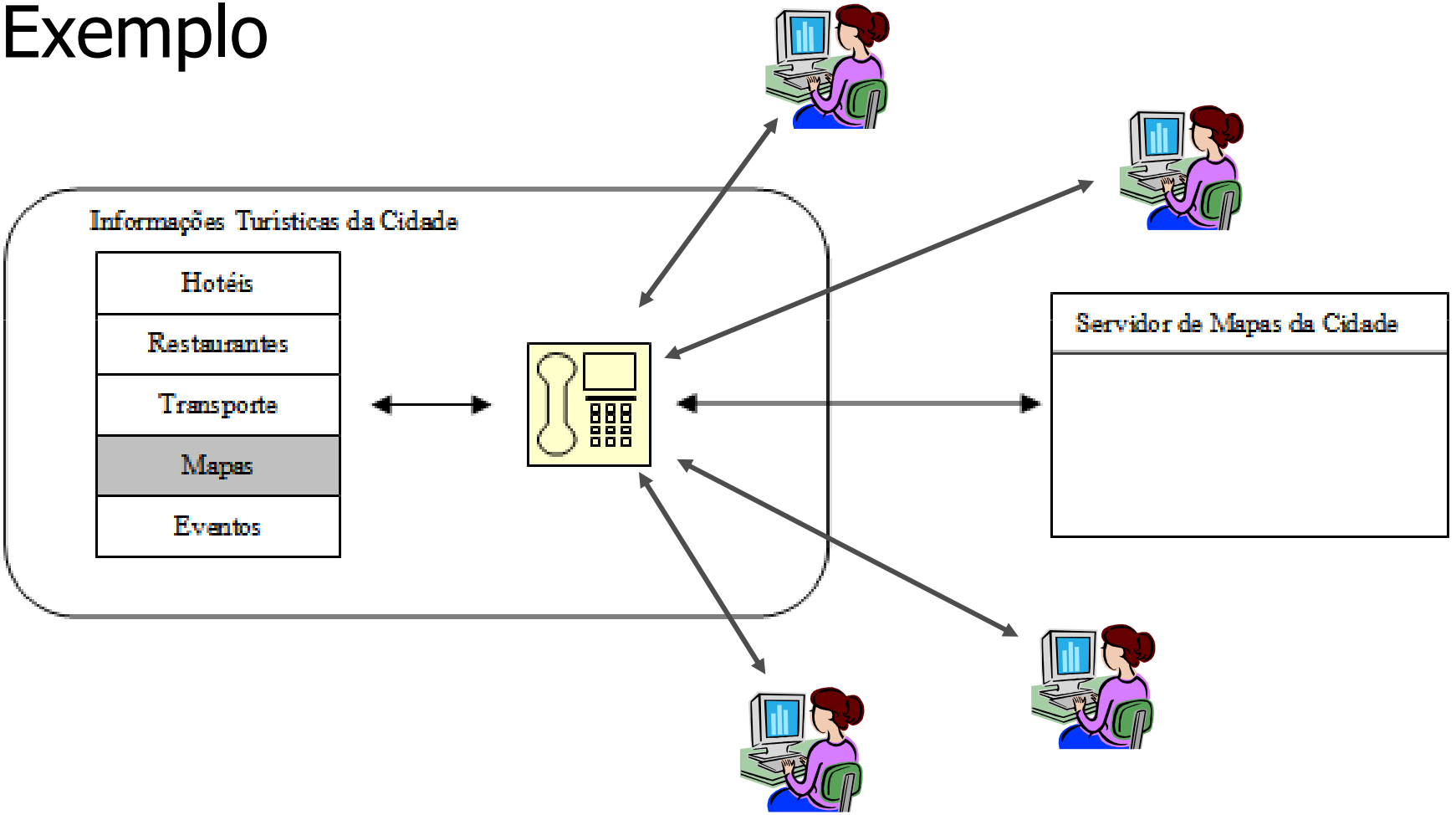


Padrão Broker

- Exemplo
  - Sistema de informações turísticas para uma cidade.
  - Alguns computadores na rede hospedam um ou mais serviços que mantêm informações sobre eventos, restaurantes, hotéis, monumentos históricos ou transporte público.
  - Terminais de computadores são conectados à rede. Turistas por toda a cidade podem acessar informações nas quais estão interessados a partir dos terminais usando navegadores de Web (Web browsers).

Padrão Broker

- Exemplo



Padrão Broker

- Contexto
  - Seu ambiente é um sistema distribuído e possivelmente heterogêneo, com componentes independentes que cooperam entre si.

Padrão Broker

- Problema
  - Construir um sistema de software complexo como um conjunto de componentes desacoplados e inter-operantes.
  - Serviços para adicionar, remover, trocar, ativar e localizar componentes são também necessários.
  - Aplicações que usam esses serviços não devem depender de detalhes específicos do sistema para garantir portabilidade e interoperabilidade, mesmo dentro de uma rede heterogênea.

Padrão Broker

- Problema
  - Componentes devem ser capazes de acessar serviços oferecidos por outros componentes através de invocações de serviços remotos com transparência da localização.
  - Pode ser necessário trocar, adicionar ou remover componentes em tempo de execução.
  - A arquitetura deve esconder detalhes específicos de implementação e do sistema dos usuários de componentes e serviços.

Padrão Broker

- Solução
  - Utilize um componente *broker* (intermediário) para alcançar melhor desacoplamento entre clientes e servidores.
  - Os servidores se registram junto ao *broker* e tornam seus serviços disponíveis aos clientes através das interfaces de seus métodos.
  - Clientes acessam a funcionalidade dos servidores enviando requisições através do *broker*. As tarefas do *broker* incluem:
    - Localizar o servidor apropriado,
    - Repassar a requisição ao servidor e
    - Transmitir resultados e exceções de volta ao cliente.

Padrão Broker

- Estrutura
  - O padrão arquitetural *Broker* compreende seis tipos de componentes participantes:
    - clientes,
    - servidores,
    - *brokers*,
    - *bridges* (pontes),
    - *proxies* (representantes) de cliente (*client-side proxies*)
    - *proxies* de servidor (*server-side proxies*).

Padrão Broker

- Estrutura
  - Servidor
    - Implementa serviços.
    - Registra a si mesmo no *broker* local.
    - Envia respostas e exceções de volta ao cliente através de um *proxy* de servidor.
  - Cliente
    - Implementa funcionalidade do usuário.
    - Envia requisições a servidores através de um *proxy* de cliente.

Padrão Broker

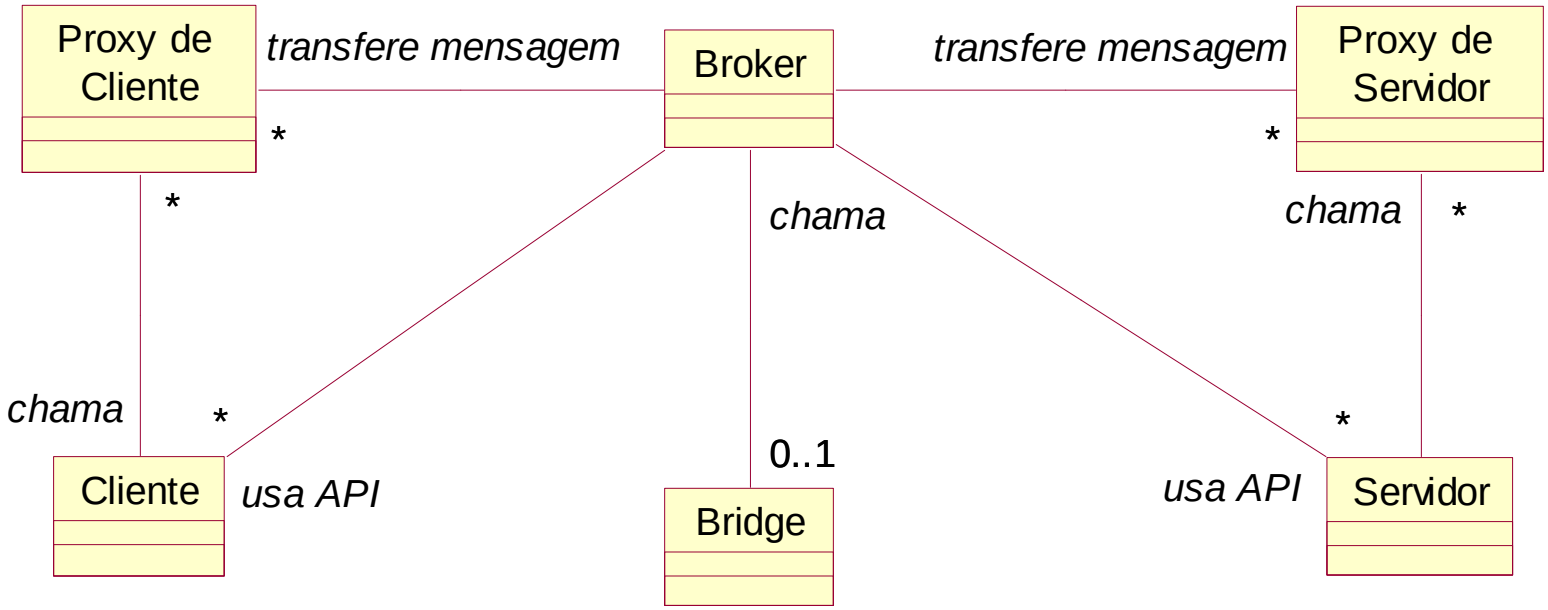
- Estrutura
  - Broker
    - Registrar e excluir registro dos servidores.
    - Oferecer APIs.
    - Transferir mensagens.
    - Recuperação de erros.
    - Inter-operar com outros *brokers* através das *bridges* (pontes).
    - Localizar servidores.

Padrão Broker

- Estrutura
  - *Proxy* de Cliente
    - Encapsula funcionalidade específica do sistema cliente.
    - Faz a mediação entre o cliente e o *broker*.
  - *Proxy* de Servidor
    - Chama serviços dentro do servidor.
    - Encapsula funcionalidade específica do sistema servidor.
    - Faz a mediação entre o servidor e o *broker*.
  - *Brigde*
    - Encapsula funcionalidade específica da rede.
    - Faz a mediação entre o *broker* local e a *bridge* de um *broker* remoto.

Padrão Broker

- Estrutura

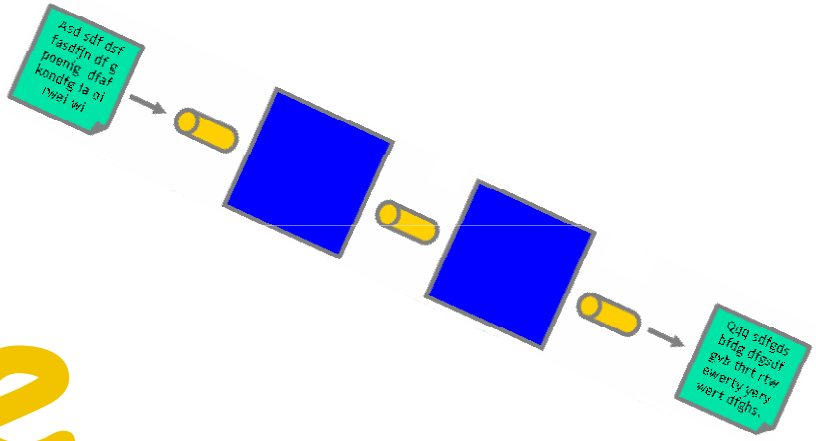


Padrão Broker

- Usos Conhecidos
  - CORBA (Common Object Request Broker Architecture) definido pelo OMG – Object Management Group.
  - O IBM SOM/DSOM (***S****ystem* ***O****bject* ***M****odel*) representa um sistema *Broker* no padrão CORBA.
  - A tecnologia OLE 2.x da Microsoft (***O****bject* ***L****inking and* ***E****mbedding*)
  - A WWW é o maior sistema de *Broker* existente no mundo.

Padrão

Pipes e Filtros

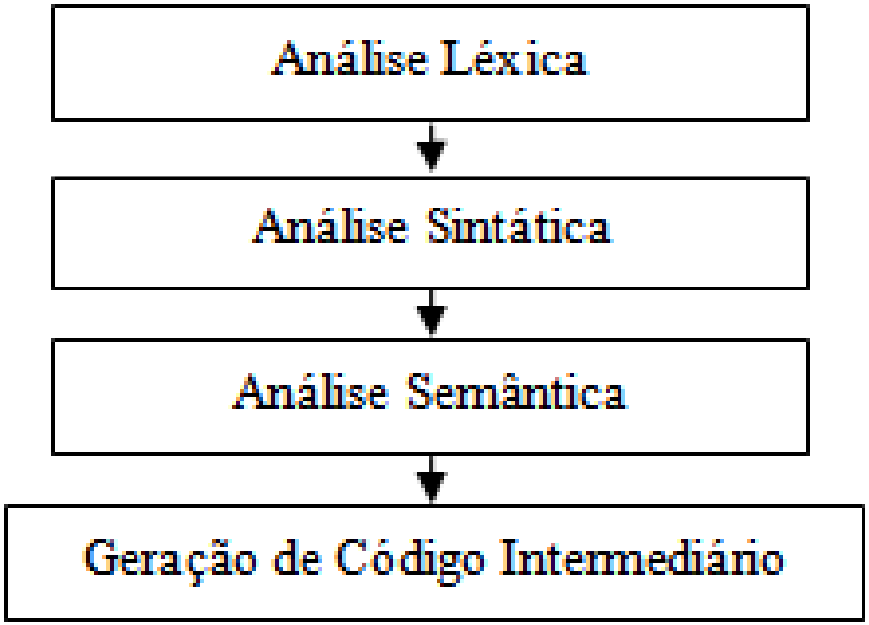


Padrão Pipes e Filtros

- Exemplo
  - Compilador de linguagens de programação.
  - Conceitualmente, a tradução do código-fonte de uma linguagem num código portável é bem definida e consiste das seguintes fases:
    - Análise Léxica,
    - Análise Sintática,
    - Análise Semântica,
    - Geração de Código Intermediário.
  - Cada fase tem sua entrada e saída muito bem definida.

Padrão Pipes e Filtros

- Exemplo



Padrão Pipes e Filtros

- Contexto
  - Processamento de cadeias (*streams*) de dados.
- Problema
  - O sistema que deve processar ou transformar uma seqüência de dados de entrada.
  - O sistema tem que ser construído por vários desenvolvedores.
  - A tarefa global do sistema naturalmente se decompõe em vários estágios de processamento.
  - Os requisitos são sujeitos a mudanças.

Padrão Pipes e Filtros

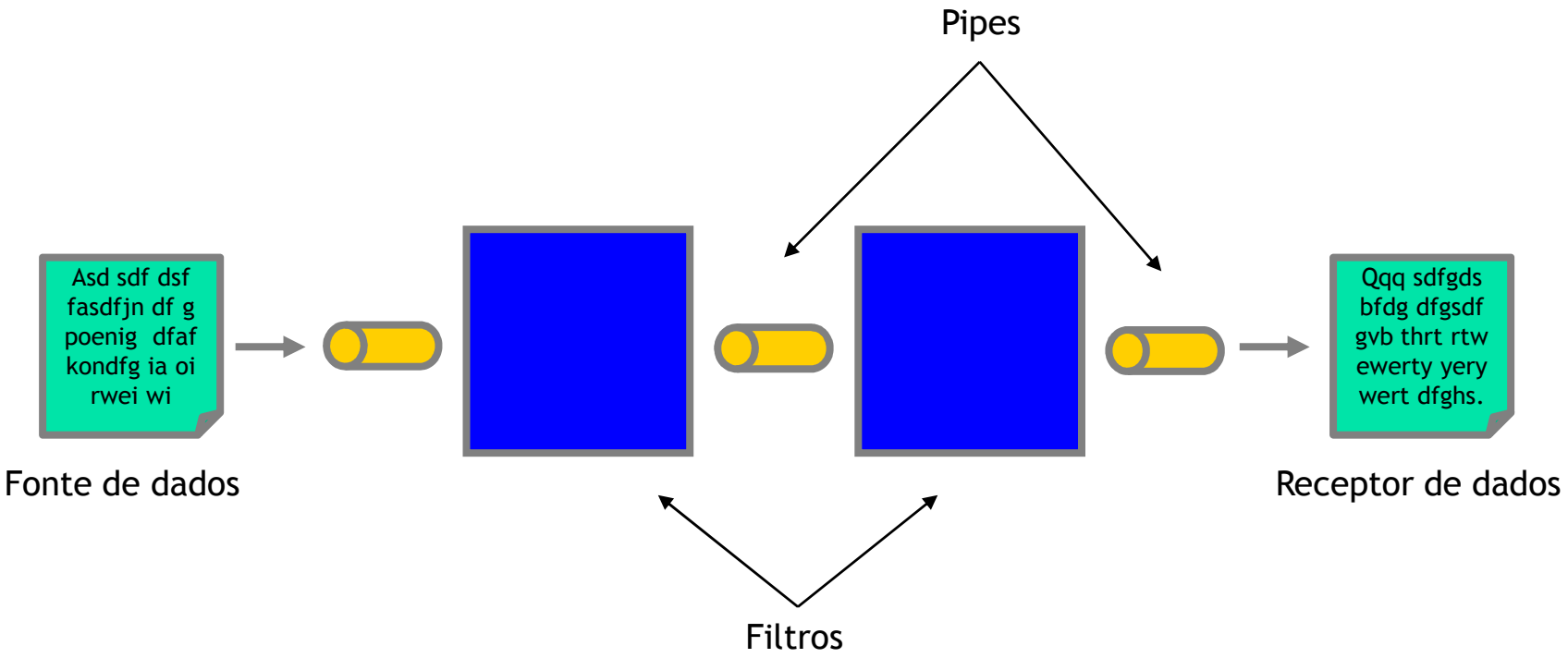
- Solução
  - Este padrão divide a tarefa de um sistema em vários estágios de processamento seqüenciais.
  - Estes estágios são conectados pelo fluxo de dados através do sistema – os dados de saída de um estágio servem de entrada para o próximo.
  - Cada estágio de processamento é implementado por um **filtro** (*filter*).
  - Um filtro consome e entrega dados incrementalmente, em contraste a consumir toda a entrada antes de produzir alguma saída, para alcançar baixa latência e permitir processamento paralelo real.

Padrão Pipes e Filtros

- Solução (cont.)
  - A entrada para o sistema é provida por uma fonte de dados tal como um arquivo texto.
  - A saída flui para um receptor de dados, como um arquivo texto, terminal, programa de animação, ou outros.
  - As fontes de dados, os filtros e os receptores de dados são conectados sequencialmente por **pipes** (tubos).
  - Cada *pipe* implementa um fluxo de dados entre estágios adjacentes. A seqüência de filtros combinada por *pipes* é chamada de processamento ***pipeline***, dando idéia de uma linha de montagem.

Padrão Pipes e Filtros

- Solução (cont.)



Padrão Pipes e Filtros

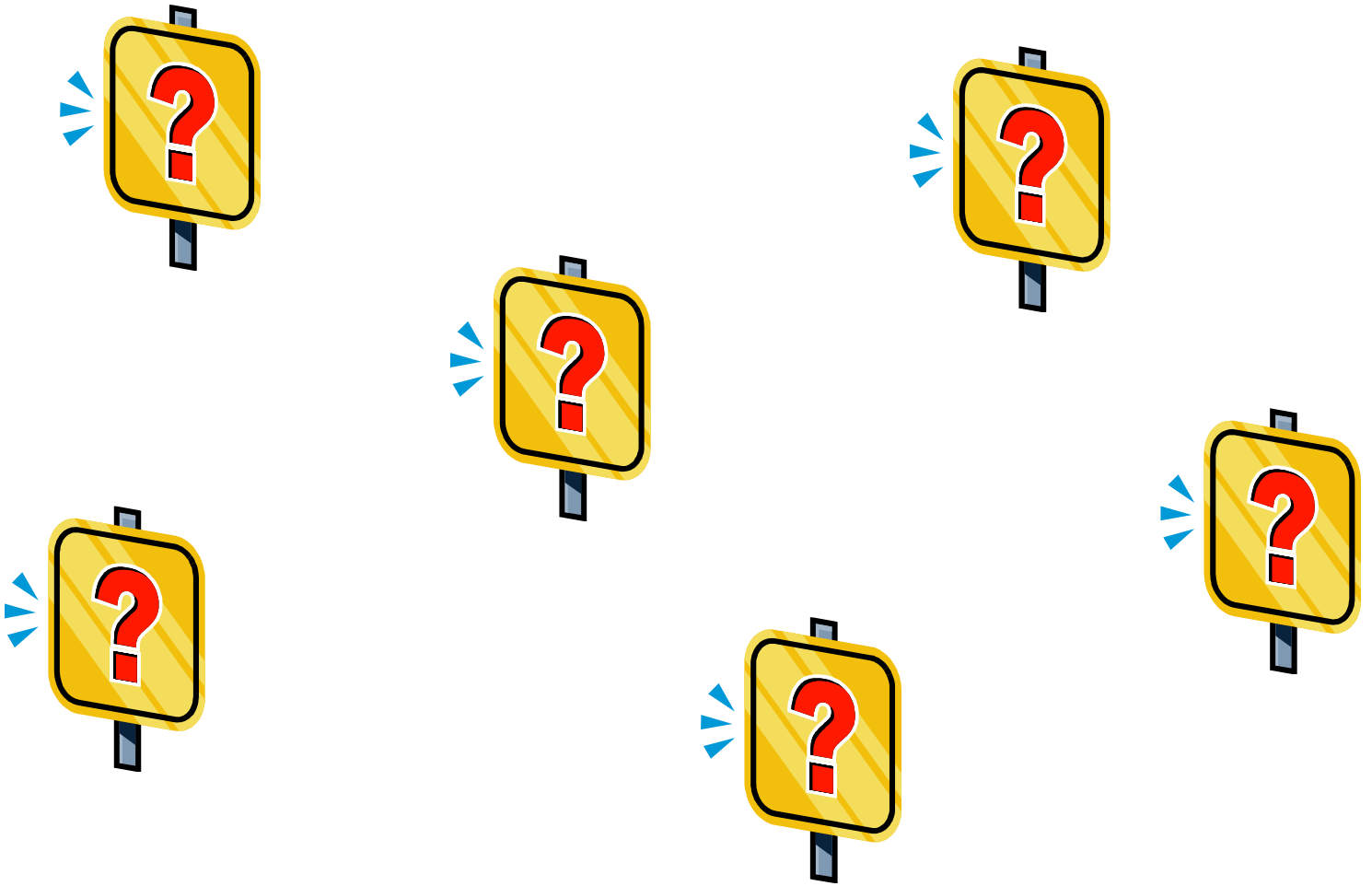
- Estrutura
  - Filtro
    - Obtém dados de entrada.
    - Executa uma função sobre os dados de entrada.
    - Fornece dados de saída.
  - Pipe
    - Transfere dados.
    - Bufferiza dados.
    - Sincroniza componentes vizinhos ativos.

Padrão Pipes e Filtros

- Estrutura (cont.)
  - Fonte de dados
    - Fornece entrada para processamento no *pipeline*.
  - Receptor de dados
    - Consome saída.

Padrão Pipes e Filtros

- Usos Conhecidos
  - UNIX popularizou o paradigma dos Pipes e Filtros.
  - CMS Pipelines é uma extensão do sistema operacional dos mainframes IBM para suportar arquitetura de Pipes e Filtros.
  - LASSPTools é um conjunto de ferramentas para suportar análise numérica e gráficos que utiliza o conceito de Pipes e Filtros.



Obrigado!!!

Agradecimentos:
Prof. Eduardo Mendes
Prof. Régis Simão

Faculdade 7 de Setembro