UNIP
UNIVERSIDADE PAULISTA

UNICLUSTER

MVC

Carlos Arruda Baltazar
UNIP – Cidade Universitária

Padrões de projeto

Padrões de projetos são soluções para problemas que foram resolvidos através da criação de um modelo que foi documentado e que pode ser adaptado integralmente ou de acordo com necessidade.

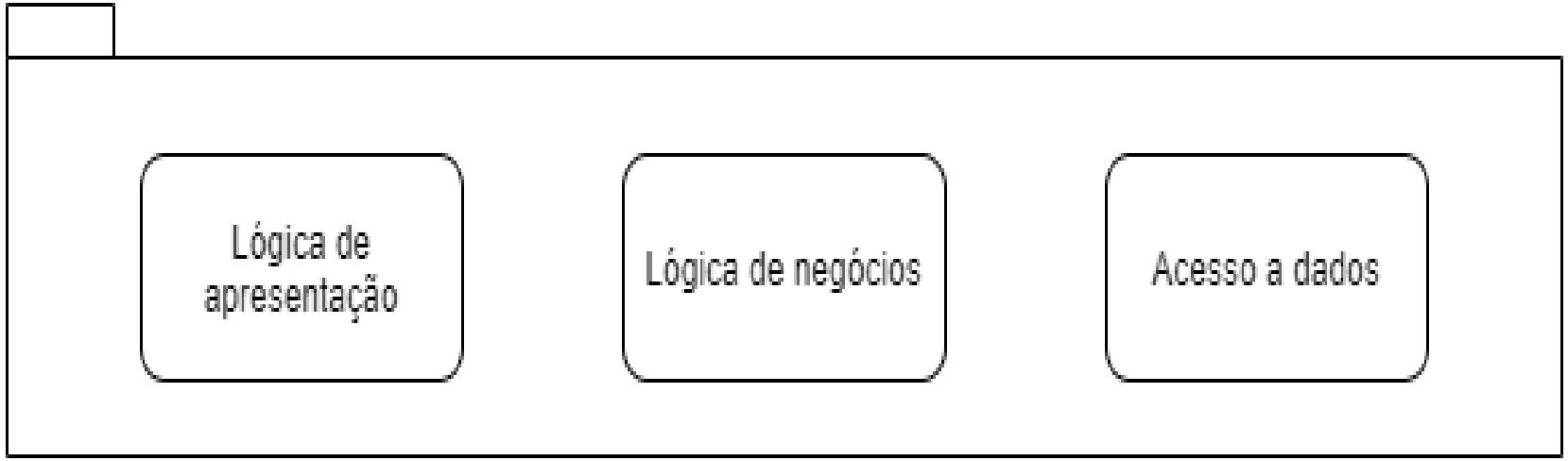
Padrões de projeto

O sucesso para o desenvolvimento de aplicações orientadas a objetos está diretamente associado a arquitetura que será adotada. A tendência indica que esta arquitetura se baseia na organização da aplicação em camadas e na observação dos padrões utilizados pelo mercado.

Aplicações monolíticas

Na época em que predominavam os computadores de grande porte com processamento centralizado e os computadores pessoais independentes, um aplicativo era desenvolvido para ser usado em uma única máquina. Geralmente este aplicativo contém todas as funcionalidades em um único módulo composto por uma grande quantidade de linhas de código e de manutenção nada fácil. A entrada do usuário, verificação, lógica de negócio e acesso a banco de dados estava presente em um mesmo lugar, como esquematizada a seguir:

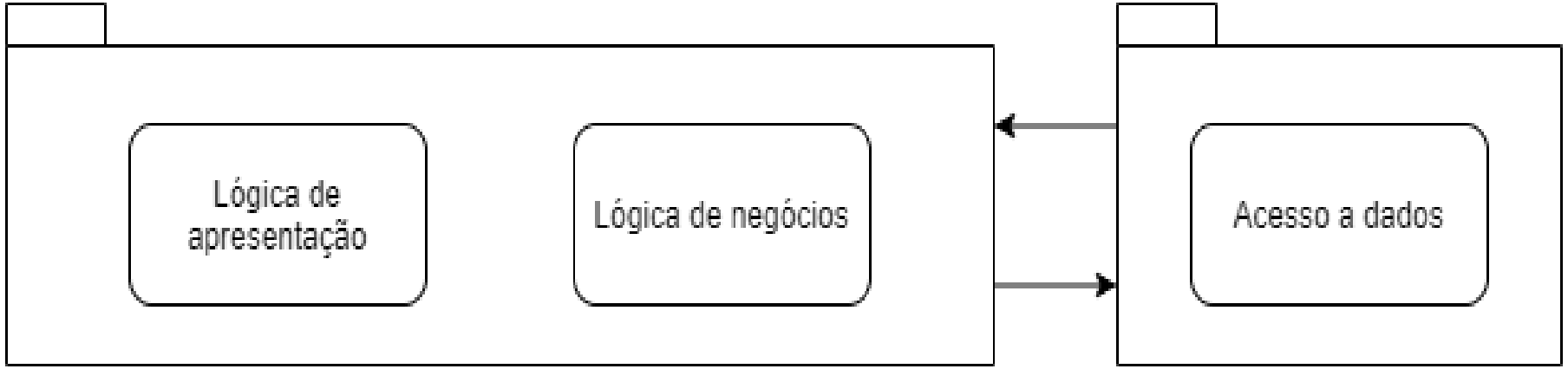
Aplicações monolíticas



Aplicações em 2 camadas

A necessidade de compartilhar a lógica de acesso a dados entre vários usuários simultâneos fez surgir as aplicações em duas camadas. Nesta estrutura, a base de dados é alocada em uma máquina específica, apartada das máquinas que executam a aplicação. Nesta abordagem, os aplicativos instalados em estações clientes contém toda a lógica da aplicação. Um grande problema neste modelo é o gerenciamento de versões pois para cada alteração os aplicativos precisam ser atualizados em todas as máquinas clientes.

Aplicações em 2 camadas



Aplicações em 3 camadas

Com o aumento da complexidade dos softwares e o surgimento das aplicações web, houve um movimento para separar a lógica de negócio da interface com o usuário. A ideia é que os usuários possam acessar as mesmas aplicações sem ter que instalar estas aplicações em suas máquinas locais. Deste modo, a lógica do aplicativo, que inicialmente era hospedada na máquina do usuário, passou a ser hospedada em um servidor. Então, somente a interface do usuário é apresentado localmente para o usuário.

Aplicações em 3 camadas

Aplicações em 3 camadas

No modelo de 3 camadas, a lógica de apresentação está separada em sua própria camada lógica e física. A separação em camadas lógicas torna os sistemas mais flexíveis permitindo que as partes possam ser alteradas de forma independente. As funcionalidades da camada de negócio podem ser divididas em classes e essas classes podem ser agrupadas em pacotes ou componentes reduzindo as dependências entre as classes e pacotes; podendo ser reutilizadas por diferentes partes do aplicativo e até por aplicativos diferentes. O modelo de 3 camadas tornou-se a arquitetura padrão para sistemas corporativos.

Aplicações em 3 camadas

A modelagem orientada a objetos ajuda a promover a modularidade e oferecem funcionalidades através de seus métodos. Projetando-se de forma adequada os objetos podem ter reduzidas as dependências entre si ficando assim fracamente acoplados e serão mais fáceis de manter e evoluir.

MVC

O modelo de três camadas físicas divide um aplicativo de modo que a lógica de negócio resida no meio das três camadas físicas. Isto é chamado de camada física intermediária ou camada física de negócios. A maior parte do código escrito reside na camada de apresentação e de negócio.

MVC

A arquitetura MVC (Modelo Visualização e Controle do inglês: Model, View and Controller) fornece uma maneira de dividir a funcionalidade envolvida na manutenção e apresentação dos dados de uma aplicação. A arquitetura MVC não é nova e foi originalmente desenvolvida para mapear as tarefas tradicionais de entrada, processamento e saída para o modelo de interação com o usuário. Usando o padrão MVC fica fácil mapear esses conceitos no domínio de aplicações multicamadas.

MVC

Na arquitetura MVC o modelo representa os dados da aplicação e as regras do negócio que governam o acesso e a modificação dos dados. O modelo mantém o estado persistente do negócio e fornece ao controlador a capacidade de acessar as funcionalidades da aplicação encapsuladas pelo próprio modelo.

MVC

Um componente de visualização renderiza o conteúdo de uma parte particular do modelo e encaminha para o controlador as ações do usuário; acessa também os dados do modelo via controlador e define como esses dados devem ser apresentados.

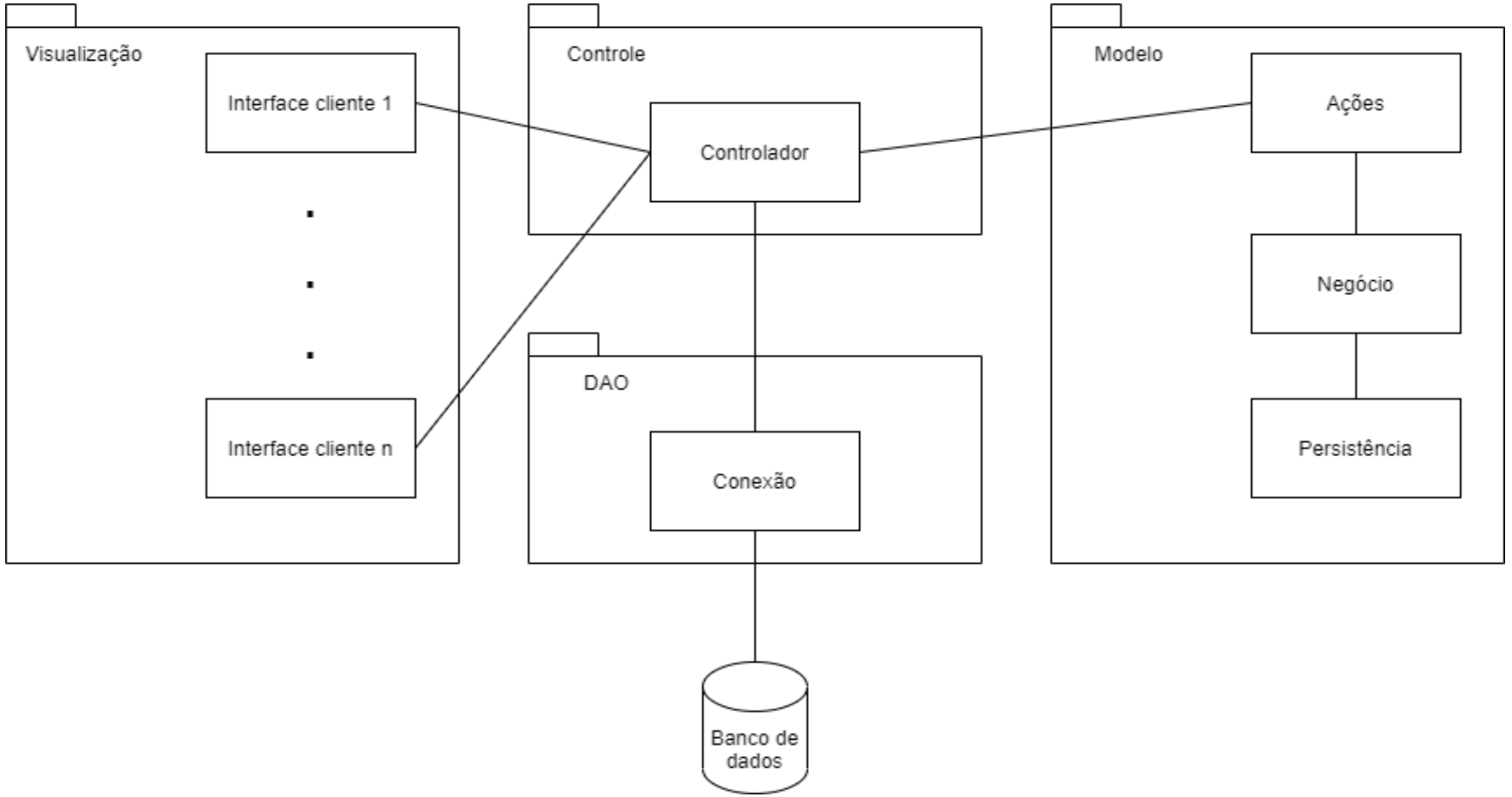
MVC

Um controlador define o comportamento da aplicação, é ele que interpreta as ações do usuário e as mapeia para chamadas do modelo. Essas ações do usuário poderiam ser um clique de botões ou seleções de menus. As ações realizadas pelo modelo incluem ativar processos de negócio ou alterar o estado do modelo. Com base na ação do usuário e no resultado do processamento do modelo, o controlador seleciona uma visualização a ser exibida como parte da resposta a solicitação do usuário. Há normalmente um controlador para cada conjunto de funcionalidades relacionadas.

MVC

A arquitetura de 3 camadas que está representada abaixo é uma implementação do modelo MVC. O modelo MVC está preocupado em separar a informação de sua apresentação.

MVC


Visualização
Interface cliente 1
Interface cliente n
Controle
Controlador
DAO
Conexão
Banco de dados
Modelo
Ações
Negócio
Persistência

MVC

- **Camada de apresentação ou visualização:** Não está preocupada em como a informação foi obtida ou onde ela foi obtida apenas exibe a informação:
  - Inclui os elementos de exibição.
  - É a camada de interface com o usuário.
  - É usada para receber a entrada de dados e apresentar o resultado

MVC

- **Camada de modelo:** É o coração da aplicação. Responsável por tudo que a aplicação vai fazer:
  - Modela os dados e o comportamento por atrás do processo de negócios.
  - Se preocupa apenas com o armazenamento, manipulação e geração de dados.
  - É um encapsulamento de dados e de comportamento independente da apresentação.

MVC

- **Camada de Controle:** determina o fluxo da apresentação servindo como uma camada intermediária entre a camada de apresentação e a lógica.
  - Controla e mapeia as ações

MVC

- **Camada DAO:** A camada DAO (Objeto de Acesso a Dados, do inglês: Data Access Object) é responsável por estabelecer a conexão entre a aplicação e o SGBD (Sistema Gerenciador de Banco de Dados), bem como manipular os dados do banco de dados.

Vantagens do MVC

- Como o modelo MVC gerencia múltiplos visualizadores usando o mesmo modelo é fácil manter, testar e atualizar sistemas múltiplos.
- É muito simples incluir novos clientes apenas incluindo seus visualizadores e controles.
- Torna a aplicação escalável.
- É possível ter desenvolvimento em paralelo para o modelo, visualizador e controle pois são independentes.

Desvantagens do MVC

- Requer uma quantidade maior de tempo para analisar e modelar o sistema.
- Requer pessoal especializado.
- Não é aconselhável para pequenas aplicações.

OBRIGADO