Princípios e Práticas de Arquitetura de Software

Capítulo 9. Estilos e Padrões Arquiteturais

Prof. Paulo Nascimento

Aula 9.1. Estilos Arquiteturais

Nesta aula

- O que é um estilo arquitetural?
- Visão Geral dos estilos
  - Layered.
  - Message Bus.
  - N-Tier.
  - Service Oriented.

O que é um estilo arquitetural?

"Um estilo de arquitetura define uma família de tais sistemas em função de um padrão de organização estrutural. Mais especificamente, um estilo arquitetural determina o vocabulário de componentes e conectores que podem ser utiliados em instâncias desse estilo, juntamente com um conjunto de restrições sobre como elas são combinadas." [Shaw, 1994]"

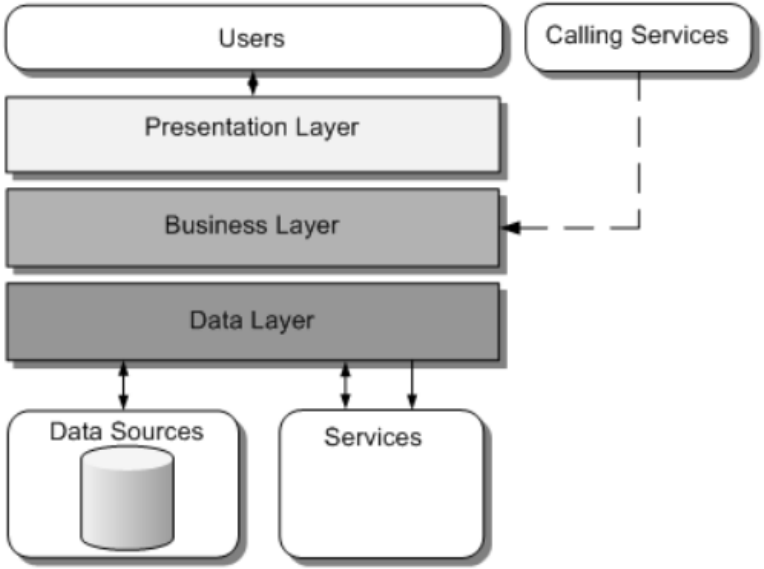
FONTE: David Garlan and Mary Shaw, January 1994, CMU-CS-94-166, see "*An Introduction to Software Architecture*"

- Benefícios
  - Fornecem uma linguagem comum.
  - Oportunidade de conversa independente de tecnologia.

Layered

- Foca em agrupar funcionalidades relacionadas dentro de uma aplicação em camadas distintas, as quais são empilhadas verticalmente.

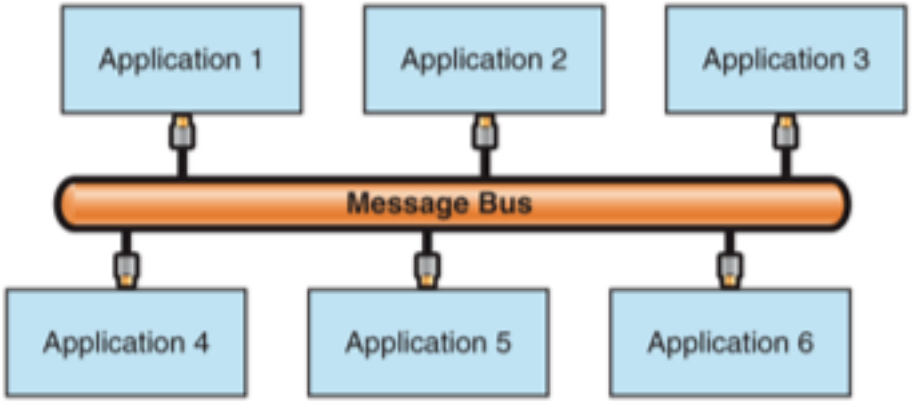
- BENEFÍCIOS:
  - Abstração.
  - Gerenciamento.
  - Desempenho.
  - Reuso.
  - Isolamento.



Message Bus

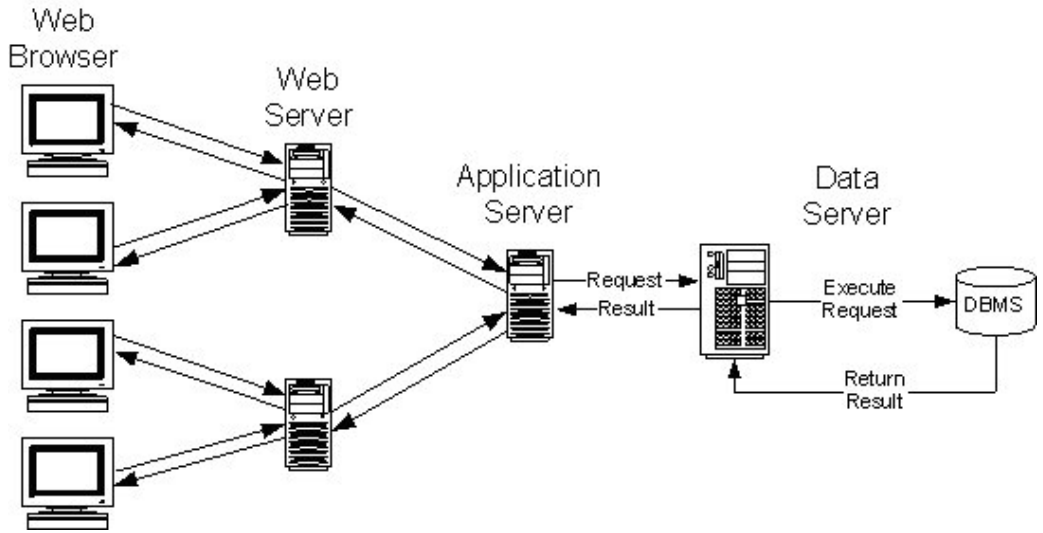
- Descreve o princípio de usar um sistema para receber e enviar mensagens usando um ou mais canais de comunicação, de forma que os sistema adjacentes possam trocar dados sem saber detalhes específicos uns dos outros.

- BENEFÍCIOS:
  - Flexibilidade.
  - Baixa complexidade.
  - Escalabilidade.
  - Baixo acoplamento.



N-Tier

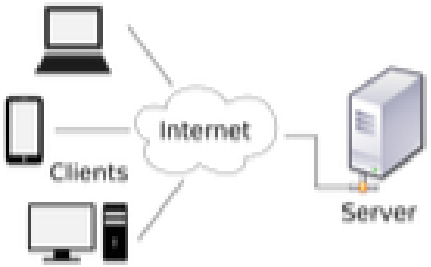
- Descreve um estilo de implantação em que funcionalidades são separadas em segmentos semelhante ao estilo Layered, mas cada segmento (Tier) pode ser uma máquina fisicamente separada.
- BENEFÍCIOS:
  - Flexibilidade.
  - Manutenção.
  - Escalabilidade.
  - Disponibilidade.

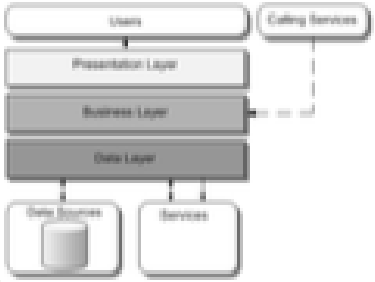


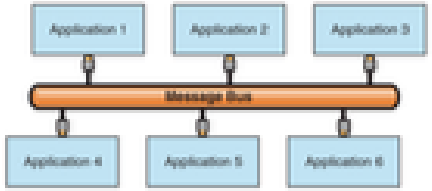
Service Oriented

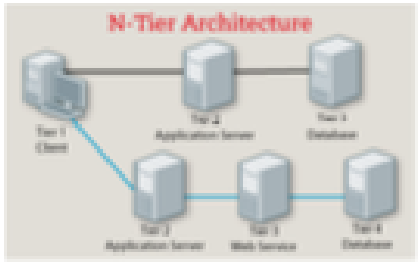
- Arquiteturas orientadas a serviço permitem que as funcionalidades da aplicação sejam fornecidas como um conjunto de serviços.
- BENEFÍCIOS
  - Alinhamento de domínio: reutilização e oportunidades.
  - Abstração: serviços são autônomos e contratuais.
  - Interoperabilidade: padrões de acesso.
  - Racionalização: remoção de duplicações.
  - Publicação: exposição das suas descrições.

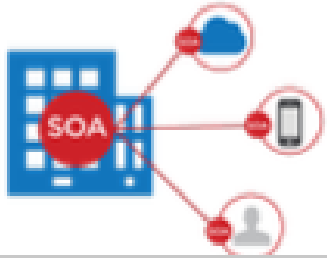
Conclusão


Internet
Clients
Server


Presentation Layer
Business Layer
Data Layer
Services


Application 1
Application 2
Application 3
Message Bus
Application 4
Application 5
Application 6


N-Tier Architecture


SOA

Próxima aula

- Padrões de Aplicação e Integração (EAP e EAI).

Aula 9.2. Padrões de Aplicação e Integração

Nesta aula

- EAP e EAI.
- Catálogo de Padrões EAP
  - Exemplos.
- Catálogo de Padrões EAI
  - Exemplos.

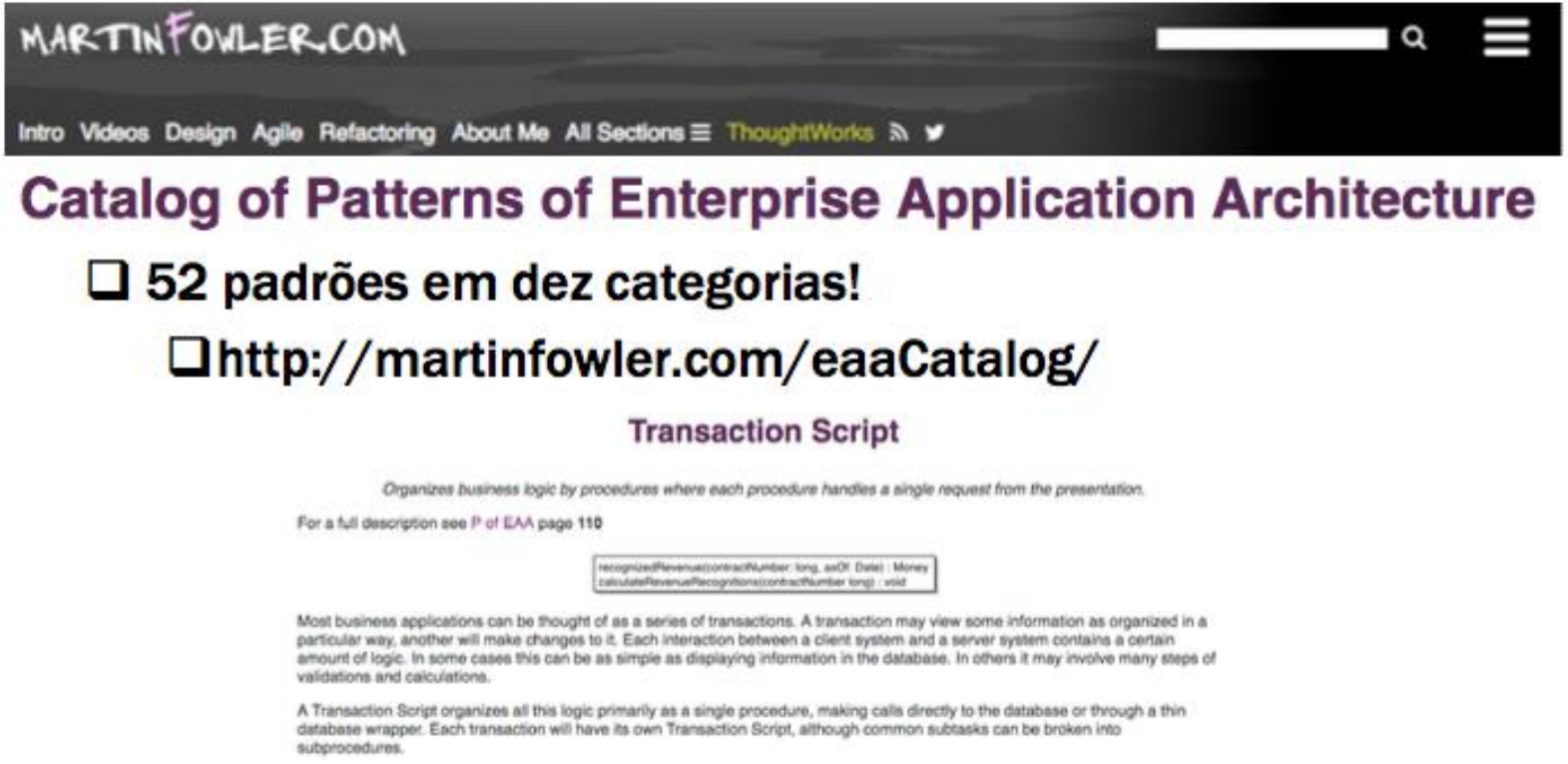
EAP e EAI

- Aplicações corporativas possuem características comuns:
  - Dados persistentes.
  - Integrações com outras aplicações (diferentes tecnologias).
  - Inúmeras interfaces de acesso.
  - Acesso concorrente.

EAP e EAI

- EAP
  - Enterprise Application Patterns (EAP) ou Patterns of Enterprise Application Architecture (P of EAA).
- EAI
  - Enterprise Application Integration são procedimentos, padrões e ferramentas que viabilizam integrações entre sistemas.
  - Não possuem “dono”.
  - Soluções para problemas comuns.
  - Seguem um formato: nome, problema e diagrama.

EAP – Catálogo de Padrões



EAP – Catálogo de Padrões

Domain Logic Patterns: Transaction Script (110), Domain Model (116), Table Module (125), Service Layer (133).

Data Source Architectural Patterns: Table Data Gateway (144), Row Data Gateway (152), Active Record (160), Data Mapper (165).

Object-Relational Behavioral Patterns: Unit of Work (184), Identity Map (195), Lazy Load (200)

Object-Relational Structural Patterns: Identity Field (216), Foreign Key Mapping (236), Association Table Mapping (248), Dependent Mapping (262), Embedded Value (268), Serialized LOB (272), Single Table Inheritance (278), Class Table Inheritance (285), Concrete Table Inheritance (293), Inheritance Mappers (302).

Object-Relational Metadata Mapping Patterns: Metadata Mapping (306), Query Object (316), Repository (322).

Web Presentation Patterns: Model View Controller (330), Page Controller (333), Front Controller (344), Template View (350), Transform View (361), Two-Step View (365), Application Controller (379).

Distribution Patterns: Remote Facade (388), Data Transfer Object (401)

Offline Concurrency Patterns: Optimistic Offline Lock (416), Pessimistic Offline Lock (426), Coarse Grained Lock (438), Implicit Lock (449).

Session State Patterns: Client Session State (456), Server Session State (458), Database Session State (462).

Base Patterns: Gateway (466), Mapper (473), Layer Supertype (475), Separated Interface (476), Registry (480), Value Object (486), Money (488), Special Case (496), Plugin (499), Service Stub (504), Record Set (508)

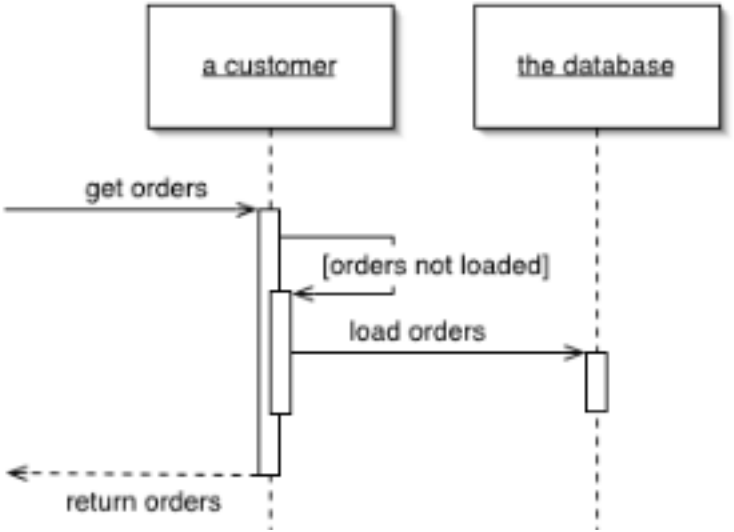
Exemplo

Suponha que voce está escrevendo uma consulta que será executada muito frequentemente pelos usuários. A entidade retornada tem um relacionamento 1:N com outra, mas você não irá precisar dessa outra entidade nesse momento.

O QUE VOCÊ FAZ?

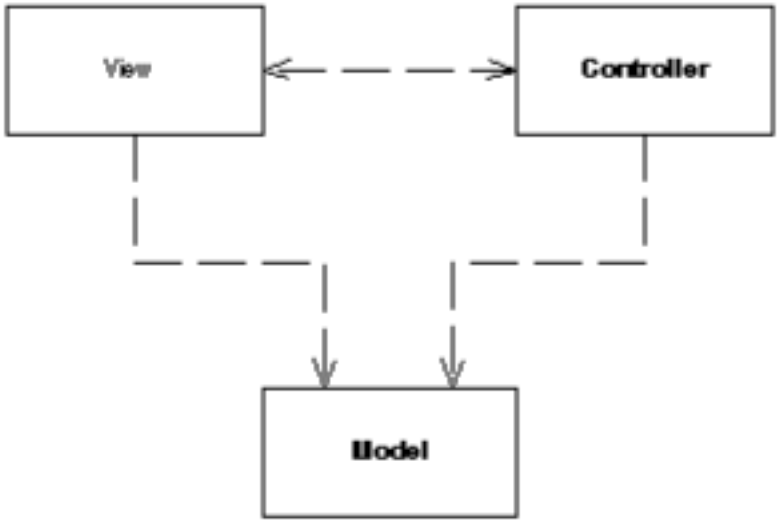
Lazy Load

An object that doesn't contain all of the data you need but knows how to get it.

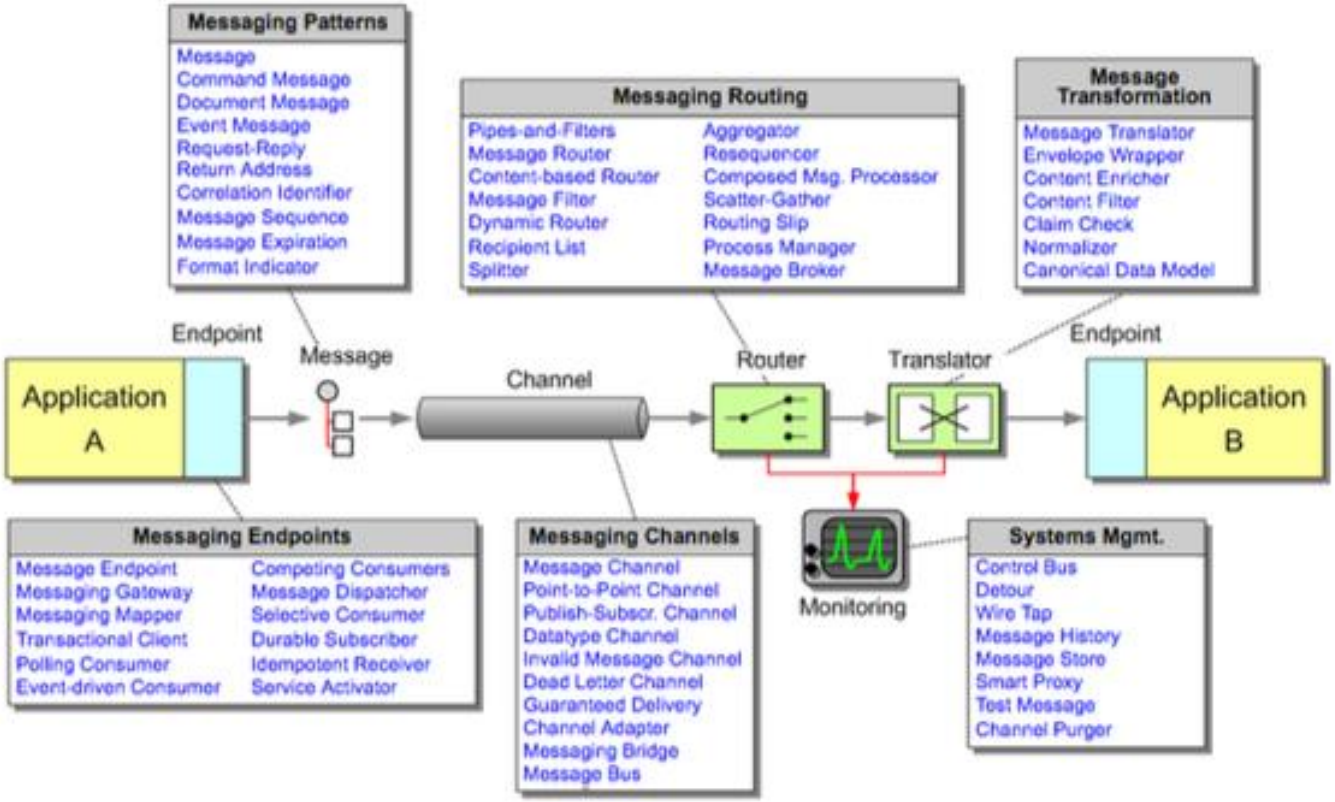


Model View Controller

Splits user interface interaction into three distinct roles.



EAI – Catálogo de Padrões



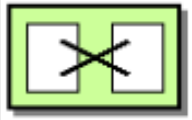
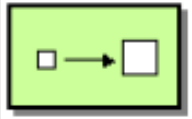
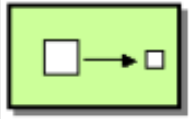
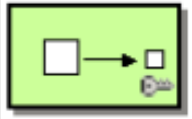
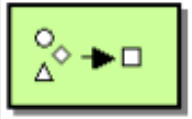
EAI – Catálogo de Padrões

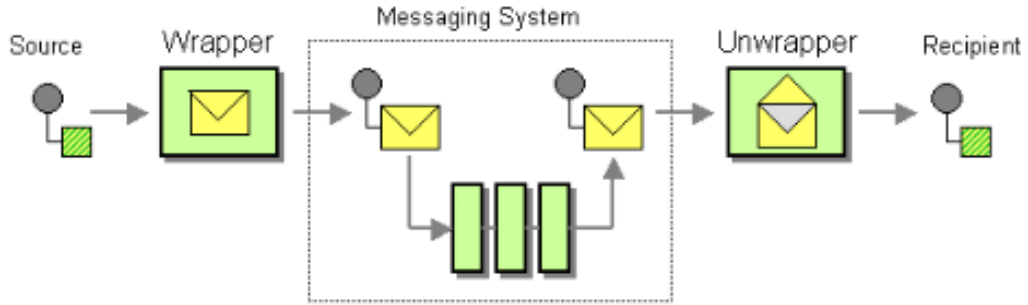
- Integration Style
  - Descrevem diferentes formas para integrar aplicações.

- Transformation Patterns
  - Altera o conteúdo da mensagem, por exemplo, adicionando, removendo ou alterando dados ou seus tipos para atender aos sistemas adjacentes.

- Channel Patterns
  - Descreve como uma mensagem pode ser transportada (esse padrão é comum em soluções de mercado).

Exemplo Transformation Patterns

Message Transformation	
	Introduction to Message Transformation
	Envelope Wrapper
	Content Enricher
	Content Filter
	Claim Check
	Normalizer



Conclusão

- ☑ Aplicações corporativas possuem características comuns:
  - ☑ Dados persistentes, diversas integrações e interfaces, além de acesso concorrente.
- ☑ EAP (Enterprise Application Patterns).
- ☑ EAI (Enterprise Application Integration).

OBRIGADO