PUC Minas

Projeto de Software

Prof. Dr. João Paulo Aramuni

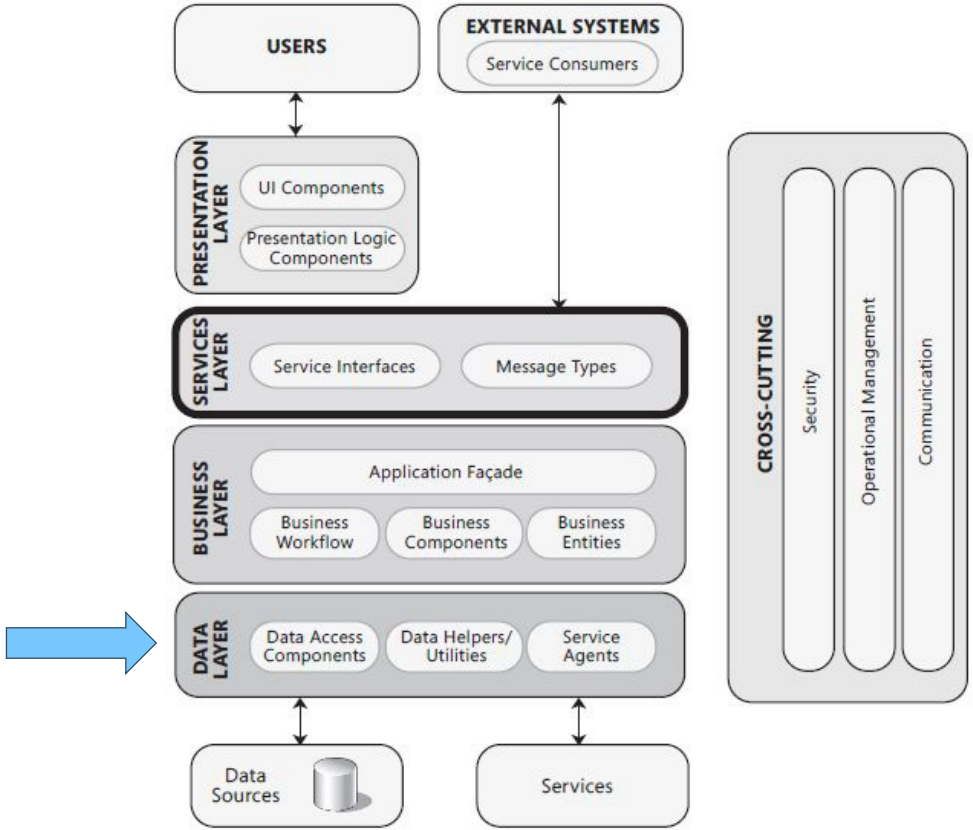
Unidade 3

Persistência

PDS - Manhã

Persistência

Projeto camada de dados



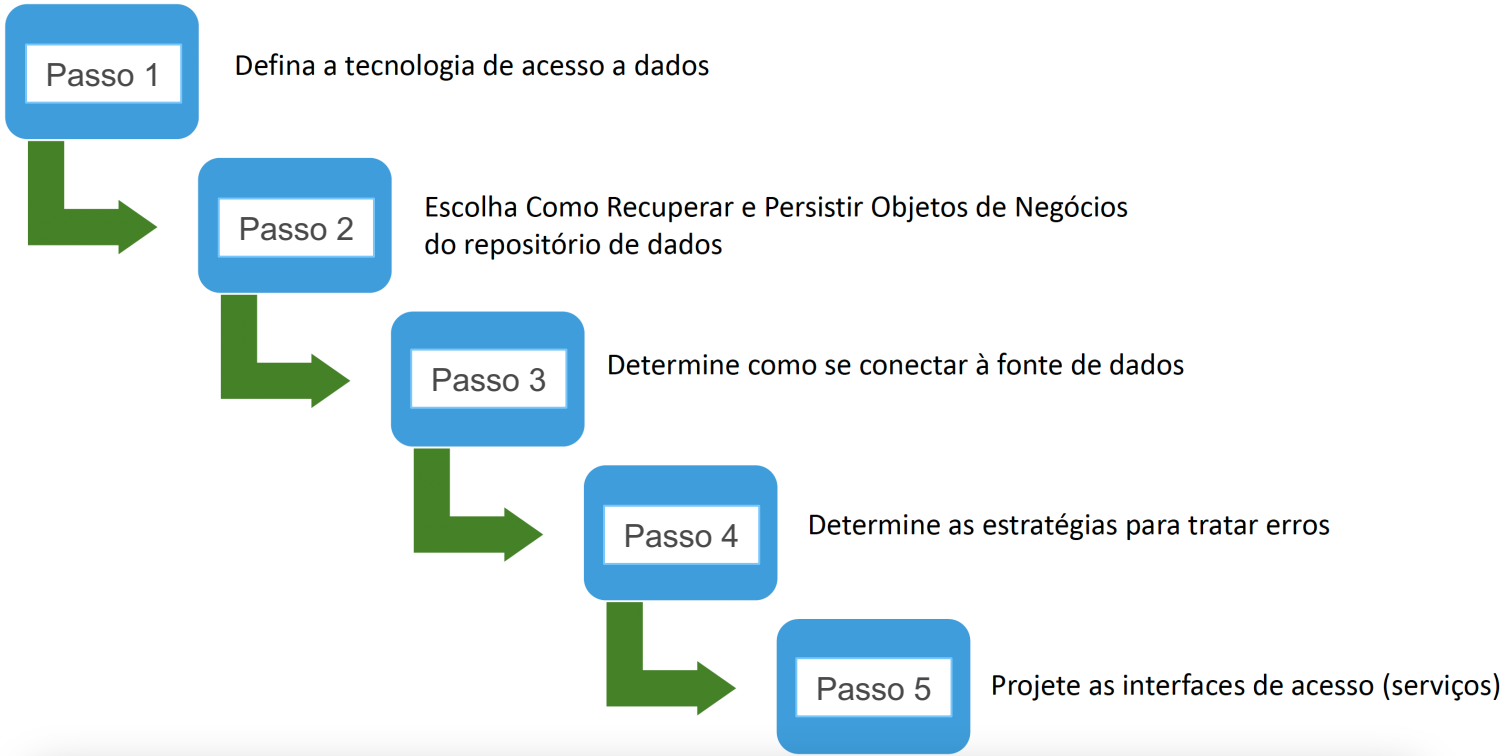
Persistência

Projeto camada de acesso aos dados

- A Camada de dados agrupa classes que têm por finalidade prover criação, remoção, alteração e recuperação de dados persistentes.
- Não é o próprio mecanismo de persistência (banco de dados ou arquivo), mas um front-end que empacota o acesso a ele.
- O fluxo de mensagem é da Camada de Negócio para a Camada de dados.
- Benefício:
  - torna possível realizar alterações na forma de persistência de dados sem impacto para o restante do sistema.

Persistência

Passos no design de acesso aos dados



Persistência

Passos no design de acesso aos dados

Defina a tecnologia de acesso a dados

- Uso de um SGBDOO ou de um SGBDOR
- Acesso direto ao banco de dados
- Uso do padrão DAO (Data Access Object)
- Uso padrão Repository
- Uso de um framework ORM
- Variações ORM (Dapper)
- Uso do padrão Active Record
- Acesso NoSQL

Persistência

Passos no design de acesso aos dados

Escolha como recuperar e persistir objetos de negócios do repositório de dados

- ORM
- Json
- XML

Persistência

Passos no design de acesso aos dados

Determine como se conectar à fonte de dados

- Connections
- Connection Pooling
- Transactions and Concurrency

Persistência

Passos no design de acesso aos dados: **Transações**

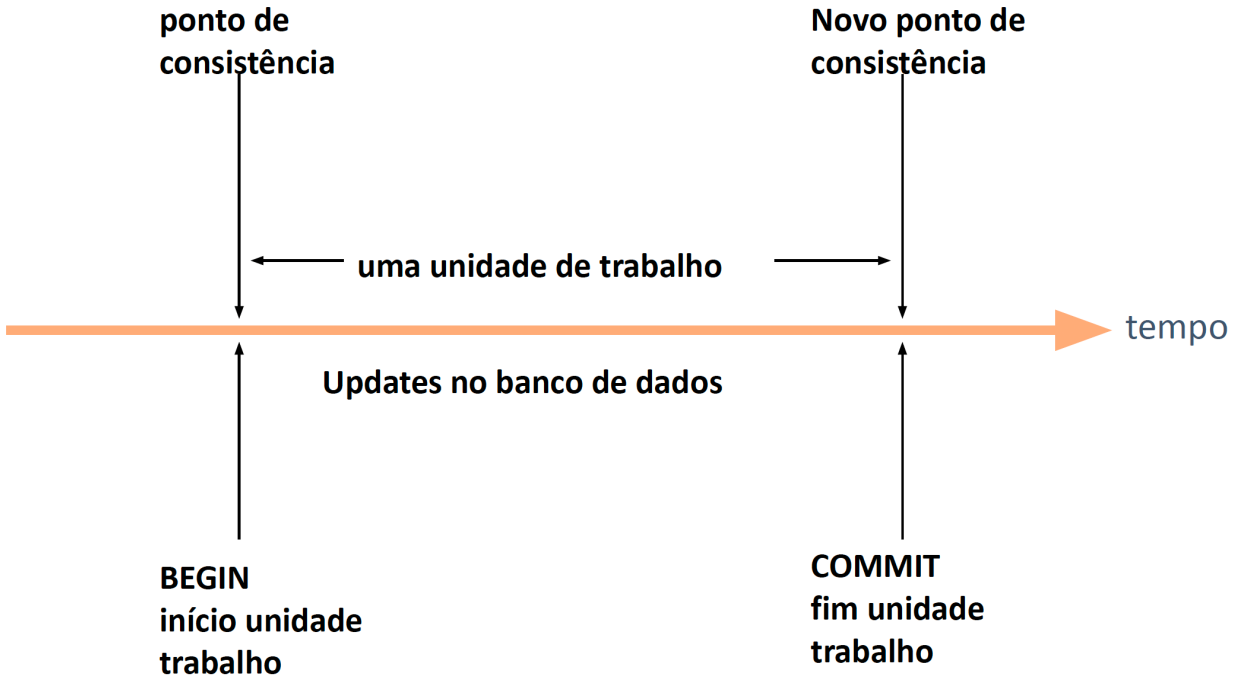
- Delimitada pelas instruções:
  - begin transaction e
  - end transaction

- A transação consiste em todas as operações executadas entre begin e end transaction.

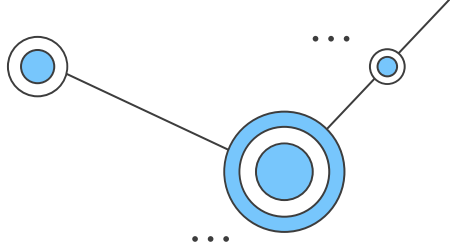
- Toda a comunicação com um banco de dados tem que ocorrer dentro de uma transação, não importa se você vai ler ou escrever dados.

Persistência

Passos no design de acesso aos dados: **Transações**

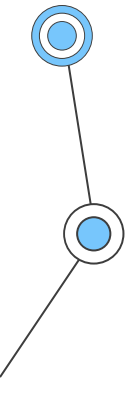


Persistência



Passos no design de acesso aos dados: **Concorrência**

- Técnicas de controle concorrência: Técnicas de Travamento em Duas Fases: bloqueios compartilhados / Exclusivos
- A trava tem três estados possíveis
  - Read-locked (compartilhado), travado para leitura
    - ✓ Outras transações podem ler o item
  - Write-locked (exclusivo), travado para gravação
    - ✓ Apenas o dono do travamento pode ler ou gravar o item
  - Unlocked, destravado
- Operações: read_lock(X), write_lock(X), unlock(X)
- A tabela de travamento (locks) terá quatro entradas
  - Nome do item de dados
  - Valor da trava
  - Número de leituras simultâneas
  - Transações responsáveis pelo travamento



Persistência

Passos no design de acesso aos dados: **Concorrência**

- Na tabela, o valor da trava é sempre read_locked ou write_locked (itens destravados não são mantidos na tabela)
- Se o valor for write_locked, então o número de transações envolvidas terá que ser 1 e a lista terá apenas a transação responsável pelo travamento
- Se o valor for read_locked, haverá um número qualquer >0 de transações e suas identificações formarão uma lista

Persistência

Passos no design de acesso aos dados

Determine as estratégias para tratar erros.

- Todas as exceções devem ser capturadas e repassadas para outras camadas somente se as falhas afetarem responsividade ou funcionalidade da aplicação.

- Trate:
  - Exceptions
  - Retry Logic
  - Timeouts

Persistência

Passos no design de acesso aos dados

Projete as interfaces de acesso (serviços)

- São os serviços de acesso aos dados.
- Use ferramenta apropriada para adicionar uma referência de serviço.
- Determine como o serviço será usada na aplicação ou por agentes externos.

Projeto de Software

Referências básicas:

- **ACM TRANSACTIONS ON SOFTWARE ENGINEERING AND METHODOLOGY**. New York, N.Y., USA: Association for Computing Machinery, 1992-. Trimestral. ISSN 1049-331X. Disponível em: https://dl.acm.org/toc/tosem/1992/1/2. Acesso em: 19 jul. 2024. (Periódico On-line).

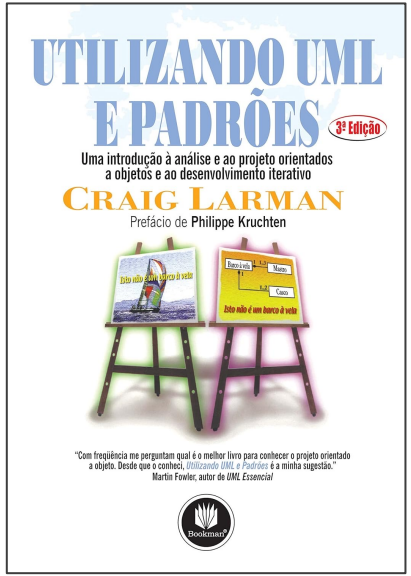
- LARMAN, Craig. **Utilizando UML e padrões**: uma introdução á análise e ao projeto orientados a objetos e desenvolvimento iterativo. 3. ed. Porto Alegre: Bookman, 2007. E-book. ISBN 9788577800476. (Livro Eletrônico).

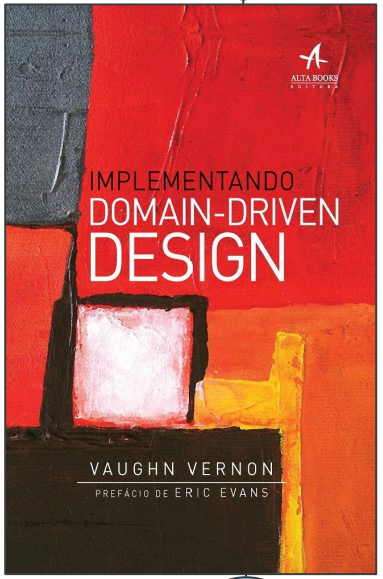
- SILVEIRA, Paulo et al. **Introdução à arquitetura e design de software**: uma visão sobre a plataforma Java. Rio de Janeiro, RJ: Elsevier, Campus, 2012. xvi, 257 p. ISBN 9788535250299. (Disponível no Acervo).

- VERNON, Vaughn. **Implementando o Domain-Driven Design**. Rio de Janeiro, RJ: Alta Books, 2016. 628 p. ISBN 9788576089520. (Disponível no Acervo).

Projeto de Software

Referências básicas:





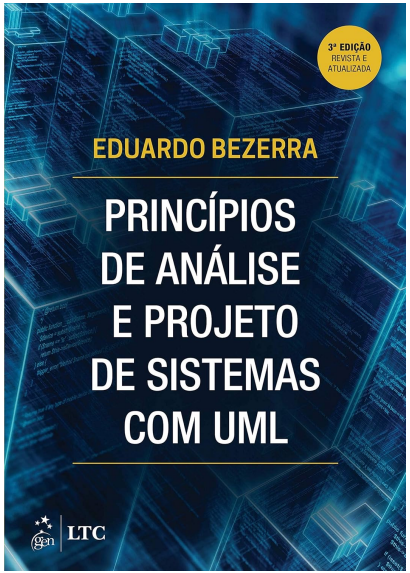
Projeto de Software

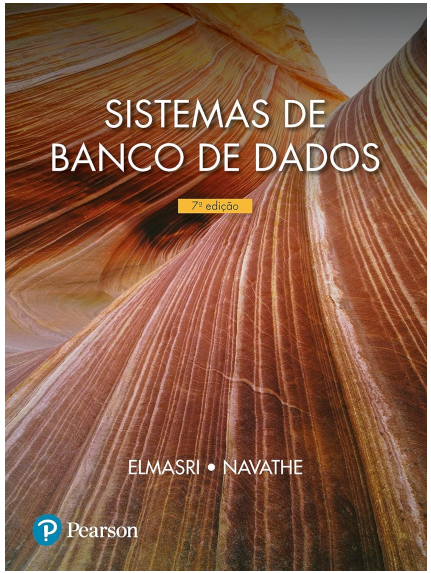
Referências complementares:

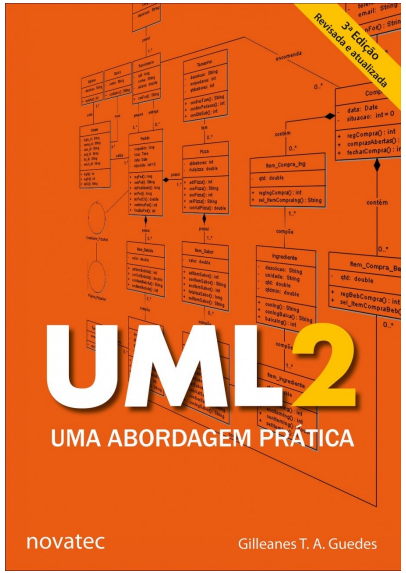
- BEZERRA, Eduardo. **Princípios de análise e projeto de sistemas com UML**. 3. ed. rev. e atual. Rio de Janeiro: Elsevier, 2015. xvii, 398 p. ISBN 9788535226263. (Disponível no Acervo).
- ELMASRI, Ramez; Navathe, Shamkant B. **Sistemas de banco de dados**, 7ª ed. Editora Pearson 1152 ISBN 9788543025001. (Livro Eletrônico).
- GUEDES, Gilleanes T. A. **UML 2**: uma abordagem prática. 2. ed. São Paulo: Novatec, c2011. 484 p. ISBN 9788575222812. (Disponível no Acervo).
- **IEEE TRANSACTIONS ON SOFTWARE ENGINEERING**. New York: IEEE Computer Society,1975-. Mensal,. ISSN 0098-5589. Disponível em: https://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=32. Acesso em: 19 jul. 2024. (Periódico On-line).
- SOMMERVILLE, Ian. **Engenharia de software**. 10. ed. São Paulo: Pearson Education do Brasil, c2019. xii, 756 p. ISBN 9788543024974. (Disponível no Acervo).
- WAZLAWICK, Raul Sidnei. **Análise e design orientados a objetos para sistemas de informação**: modelagem com UML, OCL e IFML. 3. ed. Rio de Janeiro, RJ: Elsevier, Campus, c2015. 462 p. ISBN 9788535279849. (Disponível no Acervo).

Projeto de Software

Referências complementares:

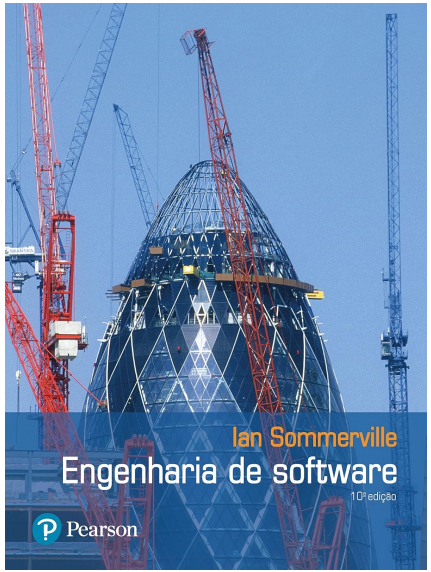






Projeto de Software

Referências complementares:



Obrigado!

Dúvidas?

joaopauloaramuni@gmail.com



GitHub LinkedIn Lattes