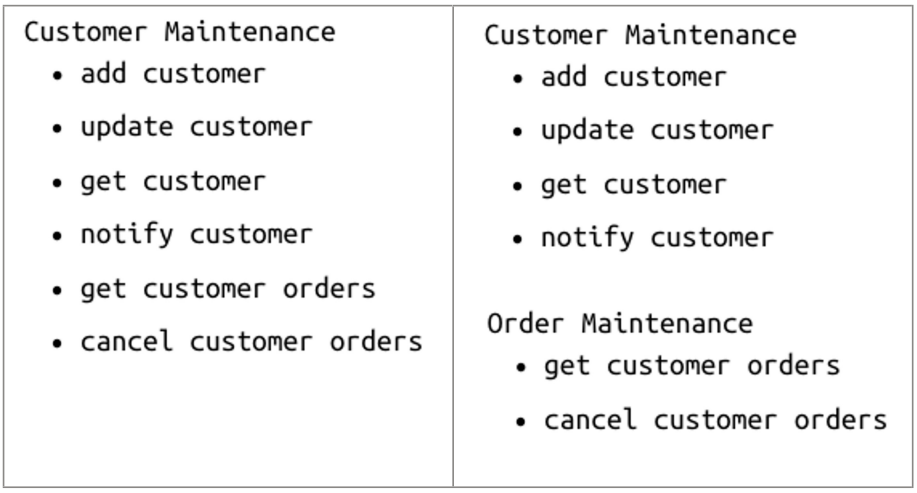
3 - Modularidade

Monday, 3 March, 2025 8:14 Manhã

- Difícil definir modularidade
- Princípio organizacional
- Agrupamento lógico de código (não físico necessariamente)
- Importantes implicações na arquitetura
- Anos 70 - Programação Modular

- **MÉTRICAS**
- Coesão -
  - Quanto que as partes precisam estar juntas
  - Mede interdependência
  - Vários tipos:
    - Funcional
    - Sequencial
    - Comunicacional
    - Procedural
    - Temporal
    - Lógica
    - Coincidental (ruim - junto mas não devia)
  - Como agrupar código - trade-off
  - 

  - Métrica disso é mais subjetiva
  - Métrica Chindambor e Kemer (CK) - > para OOP
    - LCOM -
      - Mede falta de coesão em métodos
      - Ajuda a achar métodos que deveriam estar separados
      - Tem falhas

- Acoplamento -
  - Análise por grafos - mais objetivo
  - (Aferente - entrada; Eferente - saída)
  - **ABSTRAÇÃO:**
    - Proporção de elementos abstratos para elementos concretos

Equation 3-3. Abstractness

- $$A = \frac{\sum m^a}{\sum m^c}$$

- **INSTABILIDADE:**
  - Determina volatilidade do código
  - Quanto maior a volatilidade, maior a tendência de quebrar quando mudar
  - Classe que chama muitas outras classes para delegar tarefa é mais instável

Equation 3-4. Instability

- $$I = \frac{C^e}{C^e + C^a}$$

- **DISTÂNCIA DA SEQUÊNCIA PRINCIPAL**
  - Relaciona abstração e instabilidade

Equation 3-5. Distance from the main sequence

- $$D = |A + I - 1|$$
  - Quanto menor D, mais equilibrada
  - Zona de Inutilidade:
    - Abstrato demais
  - Zona de sofrimento:
    - Sem abstração e difícil de manter

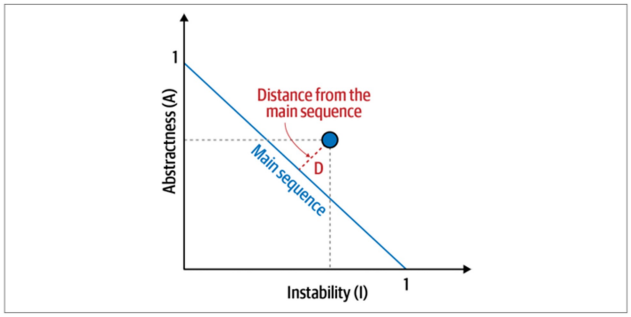


Figure 3-3. Normalized distance from the main sequence for a particular class

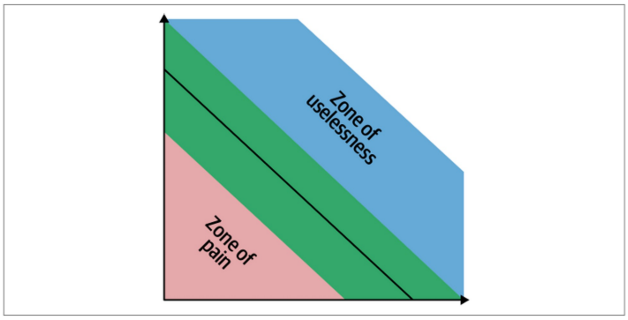


Figure 3-4. Zones of Uselessness and Pain

- <u>Conascência:</u>
  - Quanto que uma mudança em um componente requer mudanças em outros
  - Estática (a nível de código fonte)
    - De Nome
    - De Tipo
    - De Significado, convenção
    - De posição
    - De algoritmo
  - Dinâmica (a nível de execução)
    - De execução
    - De tempo
    - De valores
    - De identidade
  - Propriedades da conascência:
    - **Força**
      - Facilidade de refatorar

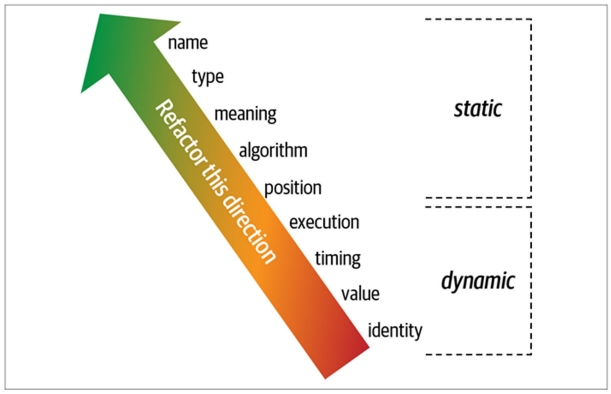
        - Conascência de nome apenas é o ideal
        - Conascência estática é melhor que dinâmica
        - 

        Figure 3-5. The strength on connascence provides a good refactoring guide

    - **Localização**
        - Proximidade lógica (e.g. mesmo módulo) entre códigos
        - Conascência em código próximo é preferível

    - **Grau**
        - Impacto da conascência
        - Quantas classes são afetadas
  - Diretrizes:
    - Encapsular partes do sistema
    - Minimizar conascência entre partes
    - Maximizar conascência intrapartes
  - Regra do Grau:
    - Converter conascência forte em fraca
  - Regra da localização:
    - Quanto maior a distância, mais fraca deve ser a forma de conascência

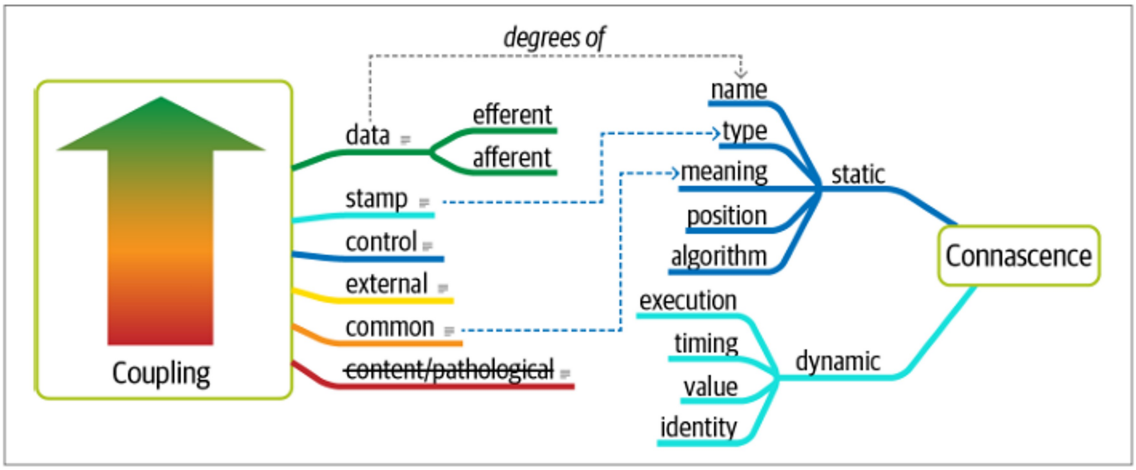
- Acoplamento é similar a conascência - overlap
- 

  Figure 3-6. Unifying coupling and connascence

- Métrica veem detalhes do código mas não estrutura da arquitetura
- Conascência não é tudo
  - Não responde decisões fundamentais a serem tomadas