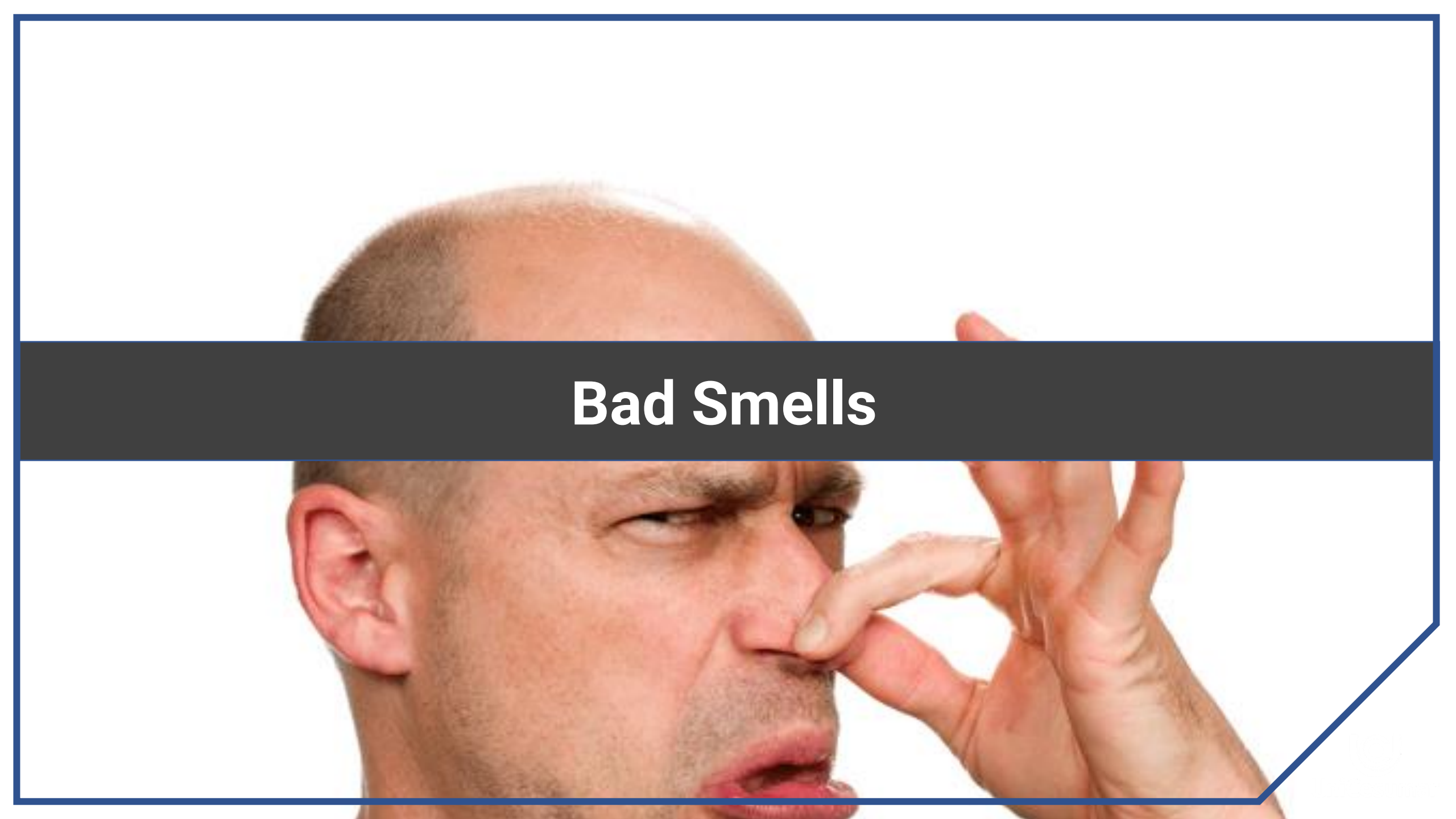
Engenharia de Software I

Profª. Me. Cynara Leão Garcia

cynara.garcia@unicesumar.edu.br

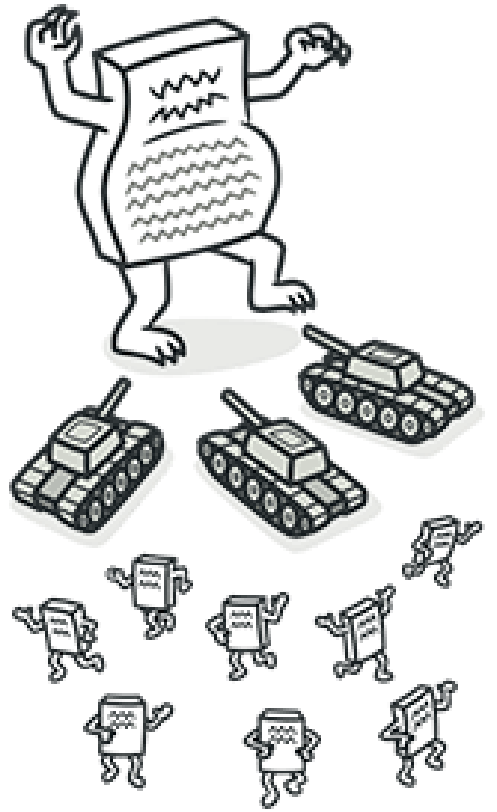
Bad Smells



Bad Smell e Refatoração

- Bad smell é uma situação no qual a estrutura do programa pode ser melhorada com refatoração

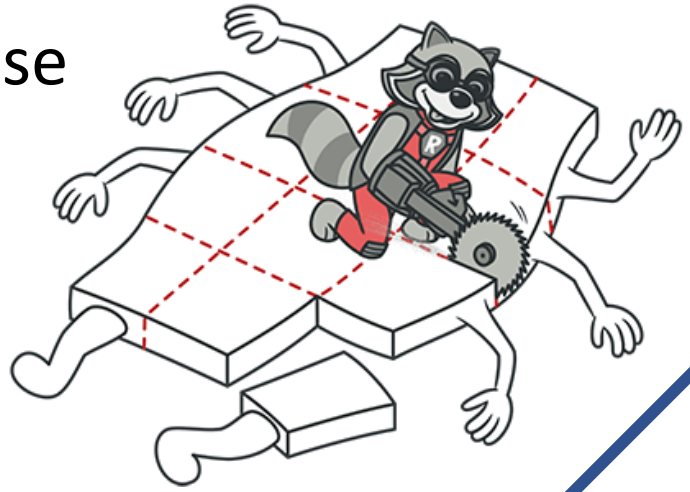
- Exemplos de bad smells
  - Código Duplicado
  - Classes ou métodos longos



Large Class / God Class

- Uma classe que faz coisa demais no sistema
  - Um sintoma pode ser o excesso de atributos

- Refatorações sugeridas
  - **Extract Class:** dividir a classe em duas
  - **Extract Subclass:** criar uma subclasse para a classe



UniCesumar

Long Method / God Method

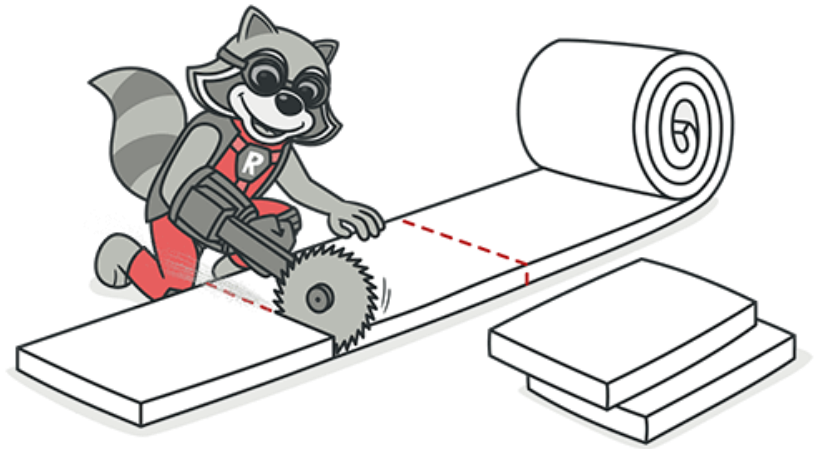
- Métodos que centralizam a funcionalidade da classe

  - Estes métodos são difíceis de entender e de manter

- Refatorações sugeridas

  - **Extract Method:** dividir o método em dois

  - **Replace Method with Method Object:** transformar um método em uma classe



UniCesumar

Feature Envy

- Parte do código de uma classe “inveja” outra classe

  - Por exemplo, um método de uma classe usa atributos somente da outra classe

- Refatorações sugeridas

  - **Move Method e Move Field:** mover métodos e atributos entre classes

  - **Inline Class:** juntar duas classe em uma

UniCesumar

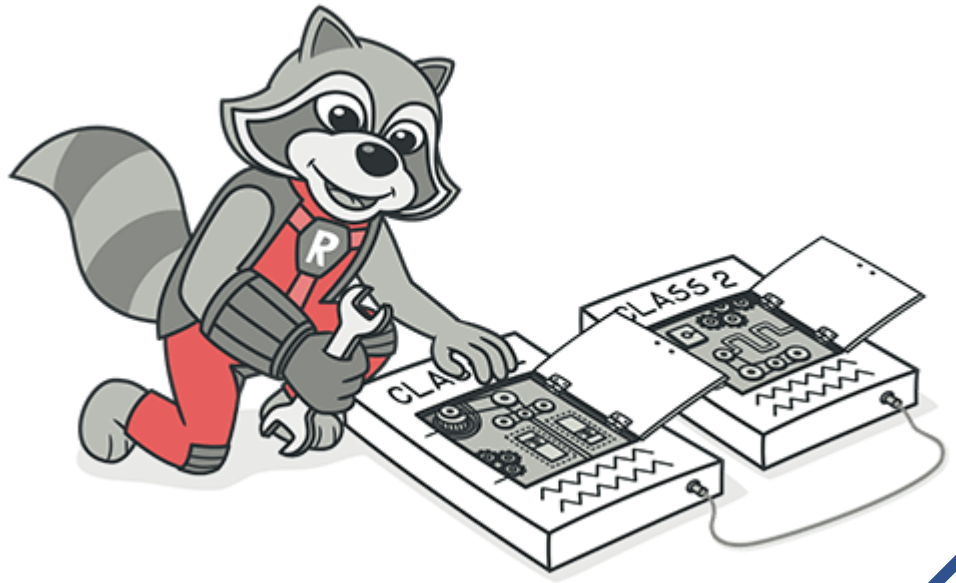
Divergent Change

- Ocorre quando uma classe pode mudar frequentemente de diferentes formas e por razões distintas

  - Idealmente, cada classe deve ser alterada apenas por um tipo de mudança

- Refatorações sugeridas

  - **Extract Class:** dividir a classe em duas



Shotgun Surgery

- Oposto do Divergent Change

  - Toda vezes que você alterar uma classe, você tem que fazer várias pequenas mudanças em outras classes diferentes

- Refatorações sugeridas

  - **Move Method e Move Field:** mover métodos e atributos entre classes

  - **Inline Class:** juntar duas classes em uma

UniCesumar

Refused Bequest

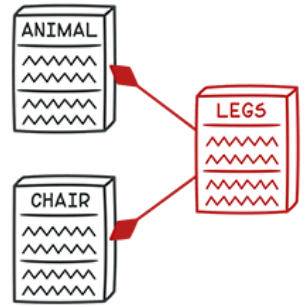
- Uma classe herda atributos e métodos de outra classe, mas não os usa

  - Algo está errado com a decomposição hierárquica (classe e subclasses)

- Refatorações sugeridas

  - **Push Down Method / Field:** mover o método ou atributo da superclasse para a subclasse

  - **Replace Inheritance with Delagation:** substituir o relacionamento de herança por associação



UniCesumar

Comments

- Comentários não é uma coisa ruim

  - Entretanto, excesso de comentários pode indicar que os nomes de métodos/atributos não estão suficientemente expressivos

- Refatorações sugeridas

  - **Extract Method:** quebrar um método em dois

  - **Rename Method/Field:** re-nomear

UniCesumar

Reutilização de Software

Reutilização de Software

- Abordagem de desenvolvimento com o objetivo de maximizar o uso de software pré-existente

- Nos últimos 20 anos, muitas técnicas foram desenvolvidas para apoiar o reuso

  - Bibliotecas, objetos, componentes, etc.

  - O movimento open source cria uma enorme base de código disponível

Tipos de Reutilização

- Objetos e Funções
  - Tipo mais comum de reutilização
  - Ocorre nos últimos 40 anos

- Componentes
  - Reuso de média granularidade. Exemplo, componente arquitetural ou sub-sistema

- Sistema
  - Um sistema pode ser reusado por incorporação à outro sistema
  - Pode ser necessário customização

Vantagens de Reutilização

- Redução de tempo e custos
  - O sistema pode ser entregue em menor prazo, que reduz os custos
- Maior confiabilidade do produto
  - O software reusado já foi testado antes
- Uso eficaz de especialistas
  - Especialistas compartilham o conhecimento
- Adequação aos padrões
  - Padrões de interface podem ser componentes reusáveis

Potenciais Problemas

- Custo de Manutenção

  - Componentes reusados podem se tornar incompatíveis em versões futuras

- Falta de Apoio de Ferramenta

  - Ambientes de desenvolvimento podem não estar preparados para reutilização

- É caro manter uma biblioteca

  - É difícil encontrar e entender o software que se pretende reusar

Planejamento para Reutilização

- Reutilização não ocorre por acaso
  - Ele deve ser planejado e incentivado em toda a organização

- Muitas empresas desenvolvem sistemas em um mesmo domínio
  - Surgem situações potenciais para reutilização

UniCesumar

Fatores do Planejamento

- Alguns fatores a considerar no planejamento de reutilização
  - Cronograma de desenvolvimento
  - Ciclo de vida do software
  - Conhecimento e experiência da equipe
  - Domínio da aplicação

Cronograma e Ciclo de Vida

- Cronograma de Desenvolvimento

  - Se o cronograma de entrega é apertado, reusar pode agilizar a entrega do produto

- Ciclo de Vida do Software

  - Reusar pode ser um problema em sistemas que sofrem manutenções frequentes

  - Componentes de terceiros (código proprietário) dificultam a manutenção

Equipe e Domínio

- Conhecimento e experiência da equipe na abordagem de reutilização

  - Muitas abordagens são difíceis de serem usadas e requerem experiência

- Domínio da aplicação

  - Em alguns domínios, é fácil encontrar componentes e bibliotecas para reusar

  - Em outros domínios, é mais complicado

Engenharia de Software I

UniCesumar