Identificação de *Bad Smells* em Softwares a partir de Modelos UML

Henrique G. Nunes (DCC/UFMG)
Mariza A. S. Bigonha (DCC/UFMG)
Kecia A. M. Ferreira (DECOM/CEFET-MG)

Sumário

- Introdução e Objetivos
- Bad Smell
- Trabalhos Relacionados
- Trabalho Proposto e UMLsmell
- Experimento Preliminar
- Resultados
- Conclusão
- Continuidade do Trabalho
- Contribuições Esperadas

Introdução

- Alterar o projeto no início é mais "barato" (Girgis et al., 2009).
- O modelo mais utilizado na UML é o diagrama de classes. (Yi et al., 2004).
- Métricas de diagramas de classes são geradas a partir de número de classes, atributos, métodos e relacionamentos.
- Métricas fornecem informações de manutenibilidade e complexidade (Girgis et al., 2009).
- Inviável calcular manualmente(Girgiset al., 2009).
- *Bad Smell*: Indicador de possível problema estrutural em código-fonte, que pode ser melhorado via refatoração (Fowler, 1999).

Objetivo

- Definir um método para identificação de problemas estruturais no início do projeto de *software* .
- Objetivos específicos:

1. Identificar *bad smells* que podem ser extraídos de diagramas UML.
2. Selecionar métricas para identificar os bad smells.
3. Identificar valores referência para métricas utilizadas, propostos na literatura.
4. Projeto e implementação de uma ferramenta para automatizar o método de identificação de *bad smells*.

Bad Smell

- Foram utilizados em nosso experimento:

1. *God Class*: Classes que tendem centralizar a inteligência do sistema (Marinescu, 2002).
2. *Shotgun Surgery*: Ocorre quando a mudança em uma classe propaga para um grande número de classes (Marinescu, 2002).
3. *Indecent Exposure*: Quando uma classe revela demais seus dados internos (Hamza, 2008).

- Estratégia de Detecção é uma expressão quantificável de uma regra, que permite avaliar se fragmentos de código estão de acordo com essa regra (Marinescu, 2002).

Trabalhos Relacionados

- Ferramentas para extração de métricas em UML

1. Utilização de diagramas de classes
2. Parser de arquivos XMI
3. Geração de métricas para UML

- Fontes: Girgis et al. (2009), Soliman et al. (2010) e Nuthakki et al. (2011)

Trabalhos Relacionados

Definição de *bad smells* (Fowler, 1999)

- Definição literal de uma lista de *bad smells.*

Modelos de *bad smells* para código-fonte (Marinescu, 2002)

- Definição de estratégias de detecção de *bad smells.*
- Valores referência: média e desvio padrão

Modelos de *bad smells* para UML (Bertrán, 2009)

- Adaptação das estratégias de detecção de Marinescu (2002).
- Comparação com resultados do inCode e Together.
- Valores referência de Marinescu (2002).

Trabalho Proposto

- Identificar bad smells na literatura
- Avaliar bad smells que podem ser extraídos de diagramas de classes
- Identificar métricas que possibilitem identificar bad smells selecionados
- Identificar valores referência na literatura para as métricas selecionadas
- Desenvolver ferramenta para automatizar metodologia
- Realizar experimentos para avaliar e validar a ferramenta:

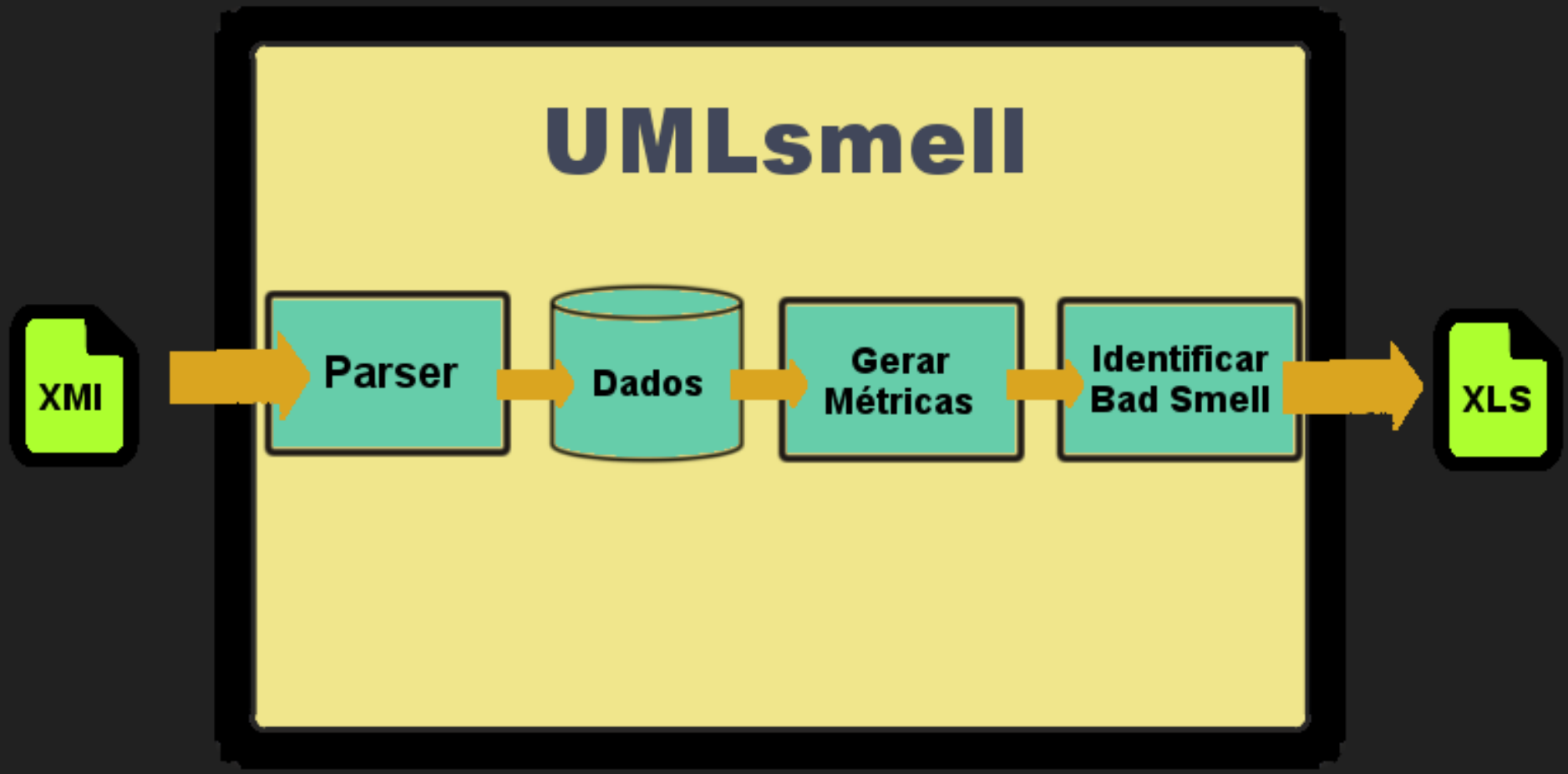
1. Automático x Manual
2. Bad Smell x Bug
3. Modelo x Código-Fonte

- Avaliar possíveis melhorias

Bad Smells e Métricas

Bad Smell	Referências	Métricas	Valores Referência
God Class (Large Class)	Marinescu (2002) e Fowler(1999)	Nº de métodos públicos (NMP)	médio: 11 a 40 ruim: > 40
		Nº de conexões aferentes (NCA)	médio: 2 a 20 ruim: >20
Shotgun Surgery	Marinescu (2002)	Nº de conexões aferentes (NCA)	médio: 2 a 20 ruim: >20
Indecent Exposure	Hamza (2008)	Nº de atributos públicos (NAP)	médio: 1 a 10 ruim: >10

UMLsmell


UMLsmell
XMI
Parser
Dados
Gerar Métricas
Identificar Bad Smell
XLS

UMLsmell

View Bad Smells Table

Class	Shotgun Surgery	God Class	Indecent Exposure
XMLElement	Average	Bad	Bad
XMLParseException	Good	Good	Good
CDATAReader	Average	Good	Good
ContentReader	Average	Good	Good
NonValidator	Average	Good	Good
PIReader	Average	Good	Good
StdXMLBuilder	Average	Good	Good
StdXMLParser	Average	Good	Good

Experimento Preliminar

Questão de Pesquisa

A utilização das métricas e seus valores referência foram úteis para identificação de bad smells?

Experimento Preliminar

- Software avaliado: Mobile Media Versão 9.
- Classes: 60.
- Linguagem: Java.
- Domínio: Aplicativo móvel para gerenciar imagens, videos e áudios.
- O Mobile Media é um software aberto desenvolvido no meio acadêmico.

Experimento Preliminar

- Dois especialistas avaliaram o diagrama de classes e avaliaram os pontos críticos
- O XMI do Mobile Media foi analisado pelo UMLsmell
- As avaliações dos especialistas foram comparadas com a saída do UMLsmell

Resultados - *Indecent Exposure*

- 7 classes com valores regulares para o UMLsmell e todas citadas pelos especialistas como mal encapsuladas.
- 1 classe com valores ruins para o UMLsmell e também citada pelos especialistas
- A métrica NPA para valores regulares variaram de 2 a 8
- A métrica NPA para valores ruins foi de 17

Resultados - *Shotgun Surgery*

- A métrica utilizada foi a de conexões aferentes.
- Classes com valores regulares de 2 a 6 foram identificadas pelo UMLsmell, mas não pelos especialistas.
- Classes com valores regulares de 9 a 16 foram identificadas pelo UMLsmell e pelos especialistas.
- Classes com valores ruins 24 e 25 foram identificadas pelo UMLsmell e pelos especialistas.

Resultados - *God Class*

- Além das conexões aferentes, a métrica NMP foi utilizada para identificar tal bad smell
- Três classes foram identificadas pelos especialistas, uma delas obteve valores regulares para as métricas e duas delas valores ruins.

Conclusão

- Maioria das classes avaliadas como problemáticas pelos especialistas atingiram níveis médio e ruim pela avaliação da ferramenta.
- Foi possível identificar classes problemáticas utilizando os valores referência definidos na literatura.

Continuidade do trabalho

- Avaliação de mais softwares
- Utilização de mais métricas e *bad smells*
- Avaliação dos resultados:

1. Comparação de inspeção manual com os resultados do UMLsmell.
2. Comparação do UMLsmell com outras ferramentas.
3. Identificação de correlações entre *bad smells* e ocorrência de bugs.

Contribuições Esperadas

Os resultados desse trabalho gerarão as seguintes contribuições principais

1. Seleção de métricas para identificar os bad smells em modelos UML.
2. Definição de um método para identificação de bad smells a partir de diagramas de classe da UML, utilizando as métricas de software e seus valores referência.
3. Disponibilização da ferramenta UMLsmell para suporte a engenheiros de software na identificação de bad smells no início do projeto.
4. Publicação de pelo menos dois artigos em conferências na área de Engenharia de Software.

Obrigado!

Perguntas e críticas ?