Software Quality and Measurement Lecture 12

Review to 1st Exam

Eduardo Figueiredo
http://www.dcc.ufmg.br/~figueiredo
ese.dcc@gmail.com

13 April 2015

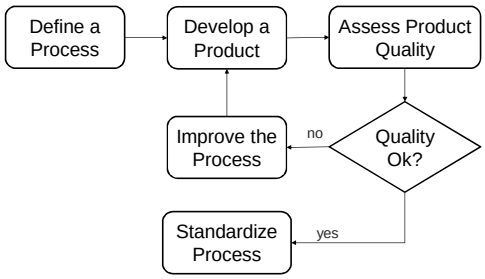
1: Course Overview

- Course introduction
- Books and papers
- Assessment criteria
- Final project, etc.

2: Introduction to SQM

- Introduction to software quality
- Software standards
- Introduction to software measurement
- Examples of Product Metrics
- Metrics for Object-Oriented Programs

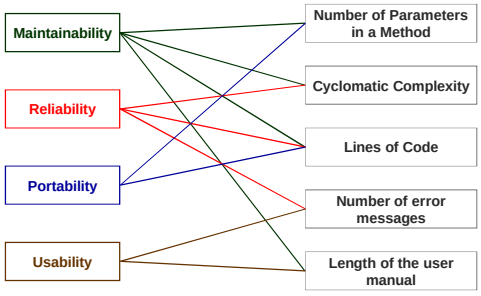
Process-based Software Quality



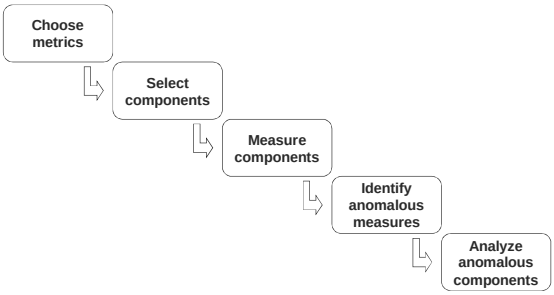
Software Standards

- Software standards play an important role in quality management
  - They define the required attributes of a product or process
- Types of standards
  - International standards
  - National standards
  - Organizational standards
  - Project standards

Example of Quality Model



Measurement Process Model



Static Product Metrics

- Fan-in / Fan-out
- Length of Code
- Cyclomatic Complexity
- Vocabulary Size
- Depth of Conditional Nesting

POO Metrics

- Chidamber-Kemerer (CK) Suite
  - Depth of Inheritance Tree (DIT)
  - Number of Children (NOC)
  - Coupling between Object Classes (CBO)
  - Lack of Cohesion of Methods (LCOM)
  - Weighted Methods per Class (WMC)
- Number of Methods Overridden (NMO)

Bibliography

- Ian Sommerville. Software Engineering, 9th Edtion. Pearson Education, 2010.
  - Chapter 24

3: Experimentation

- Introduction to Empirical Software Engineering
- Empirical Strategies (Survey, Case Study, and Experiment)
- Systematic Literature Reviews

Research Methods

- Scientific research in software engineering
- There are four main research methods in software engineering
  - Analytical
  - Scientific
  - Engineering
  - Empirical

Empirical Strategies

- The main empirical strategies are
  - Survey
  - Case Study
  - Experiment / Quasi-Experiment
- Empirical strategies are not orthogonal
  - Some studies may be viewed as a combination of strategies

Systematic Literature Reviews

- SLR are conducted to identify, analyze and interpret all available evidence related to a specific research question
- It is structured in three steps
  - Planning
  - Conducting
  - Reporting

Bibliography

- C. Wohlin et al. **Experimentation in Software Engineering**, Springer. 2012.
  - Chapter 1 - Introduction
  - Chapter 2 - Empirical Strategies
  - Chapter 4 - Systematic Literature Reviews

4: Measurement and GQM

- Concepts of Measurement
  - Scale and Scale Type
  - Types of Measures
  - Measurement in Software Engineering
- The Goal-Question-Metric Method

Scale Types

- Scales belonging to the same scale type share the same properties
- The most common scale types are
  - Nominal
  - Ordinal
  - Interval
  - Ratio

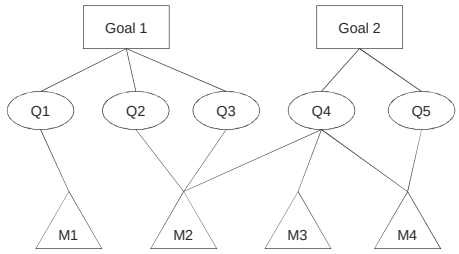
Types of Measures

- Objective Measure
- Subjective Measure
- Direct Measure
- Indirect Measure

The GQM Method

- Goals
  - They define what the organization wants to improve
- Questions
  - They refine each goal to a more quantifiable way
- Metrics
  - They indicate the metrics required to answer a question

GQM Representation



Bibliography

- C. Wohlin et al. **Experimentation in Software Engineering**, Springer. 2012.
  - Chapter 3 - Measurement
- Ian Sommerville. Software Engineering, 9th Edtion. Pearson Education, 2010.
  - Sections 24.4 and 26.2
- R. Pressman, B. Maxim. **Software Engineering: A Practitioner's Approach**, 8th Edition. McGraw-Hill Education, 2014.
  - Section 30.1.4

5: Software Product Lines

- Introduction to Software Product Lines
- Feature Modeling
- Tools for Software Product Lines

Software Product Line

- "A software product line (SPL) is a set of software systems that share a common set of features satisfying the specific needs of a particular market segment..." [SEI]
- Software product line engineering (SPLE) is a paradigm to develop software applications using platforms and mass customization

Feature Modeling

- A feature usually has a simple name
  - It facilitates communication among stakeholders
- Features in a feature model can be classified as
  - Mandatory features
  - Optional features
  - Or-Features
  - Alternative features (XOR)

Tools

- Feature Modeling Plug-in (FMP)
- XFeature
- Pure::Variants
- SPLOT
- FeatureIDE

Bibliography

- K. Pohl, G. Bockle, F. Linden. **Software Product Line Engineering: Foundations, Principles and Techniques**, 1st ed, Springer, 2005.
  - Chapter 1
- Websites of the Tools

6: Exercise about SPL

- Pure:: Variants
  - Create a feature model
  - Change the feature model
  - Analyze the feature model
  - Configure products
  - Export and import models, etc.

7: Bad Smells and Refactoring

- Bad Smells in Code
- Refactoring

List of Bad Smells

Refused Bequest	**Large Class**	**Long Method**	**Comments**
Divergent Change	**Shotgun Surgery**	**Feature Envy**	**Long Parameter List**
Primitive Obsession	**Switch Statements**	**Lazy Class**	Speculative Generality
Temporary Field	Message Chains	Middle Man	Inappropriate Intimacy
Duplicated Code	Data Clumps	Data Class	Incomplete Library Class
Parallel Inheritance Hierarchies		Alternative Classes with Different Interfaces	

Refactoring Classification

- Composing Methods
- Moving Features Between Objects
- Organizing Data
- Simplifying Conditional Expressions
- Making Method Calls Simpler
- Dealing with Generalization
- Big Refactorings

Bibliography

- Martin Fowler. **Refactoring: Improving the Design of Existing Code**. Addison-Wesley Professional, 1st edition, 1999.
  - Chapter 3 - Bad Smell in Code
  - Chapters 2, 6, 7, 8, 9, 10, 11, and 12

8: Detection Strategies

- Metric Thresholds
- Definition of Detection Strategies
- Examples of Detection Strategies
- Tools for Bad Smell Detection

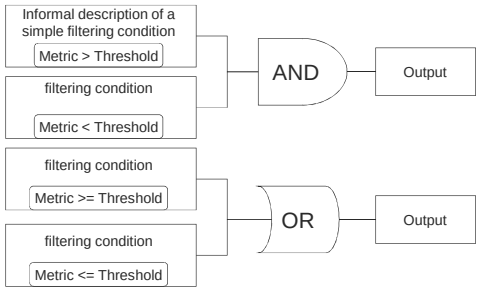
Definition of Thresholds

- A crucial factor in working with metrics is to interpret values correctly
  - What is too high or too low?
- Thresholds divide the metric values into two groups
  - Above the threshold
  - Below the threshold

Detection Strategies

- A detection strategy is a composed logical condition, based on metrics, by which code fragments with specific properties are detected
- It relies on two mechanisms
  - Filtering
  - Composition

Abstract Example



Detection Strategies

- God Class
- God Method
- Feature Envy
- Shotgun Surgery
- Refused Bequest

Tools

- JDeodorant
- inFusion
- Stench Blossom
- PMD

Bibliography

- M. Lanza e R. Marinescu. **Object-Oriented Metrics in Practice**. Springer, 2006.
  - Sections 2.1, 4.1, 5.3, 5.4, 5.6, 6.5, 7.3
- Websites of the Tools

9: Concern Metrics and Strategies

- Separation of Concerns
- Concern Metrics
- Concern-Sensitive Detection Strategies

Crosscutting Concerns

- They are concerns that are implemented by several modules of the system and mix up with other concerns
- Two typical problems
  - *Scattering*
  - *Tangling*
- It is hard to change, remove or evolve a crosscutting concern of the system

Examples of Concern Metrics

- Concern Diffusion over Components (CDC)
- Concern Diffusion over Operations (CDO)
- Concern Diffusion over Lines of Code (CDLOC)
- Number of Concerns per Component (NCC)
- Lines of Concern Code (LOCC)

Concern Detectable Bad Smells

- God Class
- God Method
- Divergent Change
- Shotgun Surgery
- Feature Envy

Bibliography

- C. Sant'Anna et al. On the Reuse and Maintenance of. Aspect-Oriented Software: an Assessment Framework. SBES 2003.
- E. Figueiredo et al. On the Maintainability of Aspect-Oriented Software: A Concern-Oriented Measurement Framework. CSMR 2008.
- M. Eaddy et al. Do Crosscuting Concerns Cause Defects? TSE 2008.
- E. Figueiredo et al. Applying and Evaluating Concern-Sensitive Design Heuristics. Journal of Systems and Software, pp. 227 - 243, 2012.
- J. Padilha. Detecção de Anomalias de Código usando Métricas de Software. Dissertação de Mestrado, DCC/UFMG, 2013. (Capítulo 5)

10: Paper Presentation

- Evidence-based Software Engineering
- Preliminary Guidelines for Empirical Research in Software Engineering

11: Paper Presentation

- When and Why Your Code Starts to Smell Bad
- How we refactor, and how we know it

13: 1st Exam

- All lectures, exercises and papers from Lecture 1 to Lecture 12