Calculation and Optimization of Thresholds for Sets of Software Metrics

Steffen Herbold, **Jens Grabowski**, Stephan Waack

Georg-August-Universität Göttingen
Institute of Computer Science

Contents

What is Software Engineering?

- Software Engineering is
  - development,
  - maintenance, and
  - deployment

  of high-quality Software in consideration of
  - scientific methods,
  - economic principles,
  - planned development models, and
  - quantifiable goals.

    - B. Kahlbrandt: *Software-Engineering: Objektorientierte Software-Entwicklung mit der Unified Modeling Language*, Springer Verlag (1998)

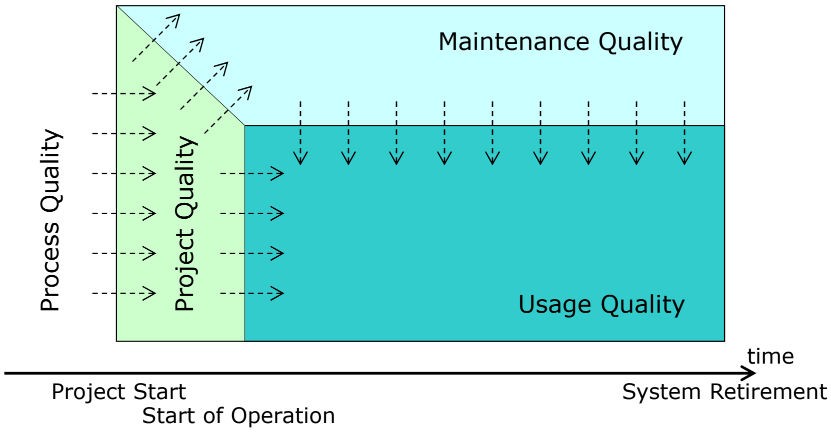
What is Software Engineering?

- Software Engineering is
  - development,
  - maintenance, and
  - deployment

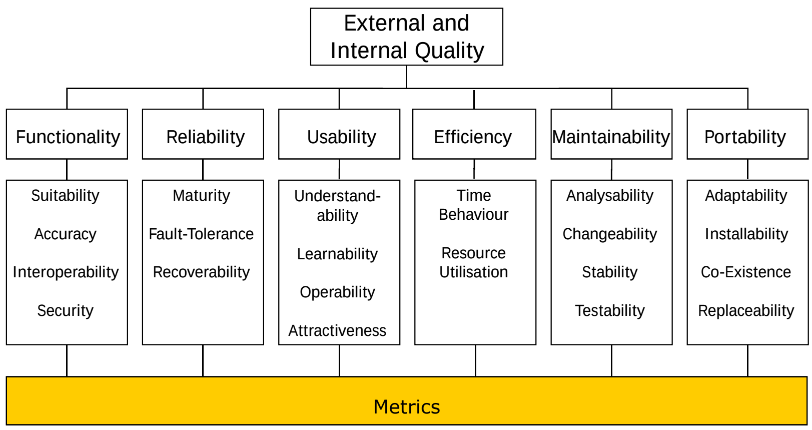
  of **high-quality Software** in consideration of
  - scientific methods,
  - economic principles,
  - planned development models, and
  - **quantifiable goals**.

    - B. Kahlbrandt: *Software-Engineering: Objektorientierte Software-Entwicklung mit der Unified Modeling Language*, Springer Verlag (1998)

What is Software Quality?



Quality Assessment using ISO 9126



Contents

What do Software Metrics Measure?

- "You cannot control what you cannot measure" (Tom DeMarco)
- "To measure is to know" (Clerk Maxwell)

Engine Power	100 PS
Fuel usage	5,8 l
Max. speed	176 km/h
Weight	1458 kg

???

Properties of Software Metrics

- Modes of measurement
  - **internal**
  - external
- Objects of measurement
  - **products**
  - processes
  - resources
- **We perform an internal measurement of products by means of static analysis of source code.**

Metrics for Methods and Classes

Modules, Files

Classes

Coupling Between Objects (CBO)
Response For a Class (RFC)
Weighted Method per Class (WMC)
Number of Overriden Methods (NORM)
Lines Of Code (LOC)
Number Of Methods (NOM)
Number of Static Methods (NSM)

Methods

Number of Statements (NST)
McCabe's Cyclomatic Number (VG)
Nested Block Depth (NBD)
Number of Function Calls (NFC)

Software Metrics for Methods

- Number of Statements (NST)
- McCabe’s Cyclomatic Number (VG)
  - Number of branches in the control flow
- Nested Block Depth (NBD)
  - Max. depth of nested statement blocks
- Number of Function Calls (NFC)
  - Number of methods invoked by the method under investigation

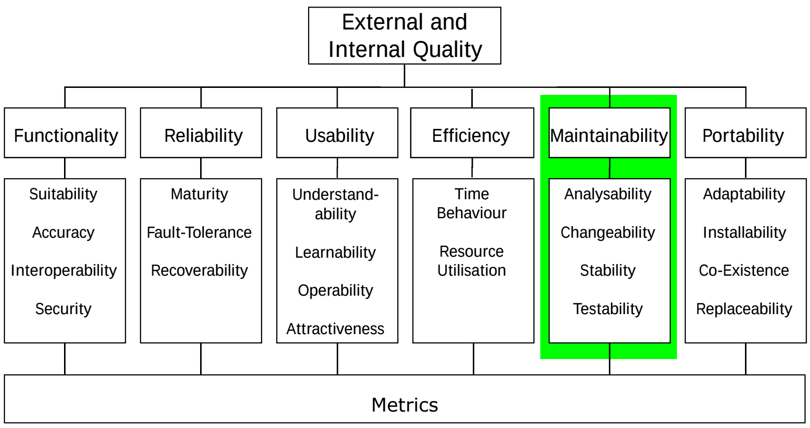
Metrics for Classes 1(2)

- Coupling Between Objects (CBO)
  - Number of associations with other classes
- Response For a Class (RFC)
  - Number of methods that can be called when the methods of the class under investigation are invoked
- Weighted Methods per Class (WMC)
  - Sum of complexities of all methods
    - Complexity: McCabe’s Cyclomatic Number (VG)

Metrics for Classes 2(2)

- Number of Overridden Methods (NORM)
  - Number of redefined methods inherited from a superclass
- Lines Of Code (LOC)
  - Lines of source code without empty lines and comments
- Number Of Methods (NOM)
- Number of Static Methods (NSM)

Quality Assessment using ISO 9126
(revisited)

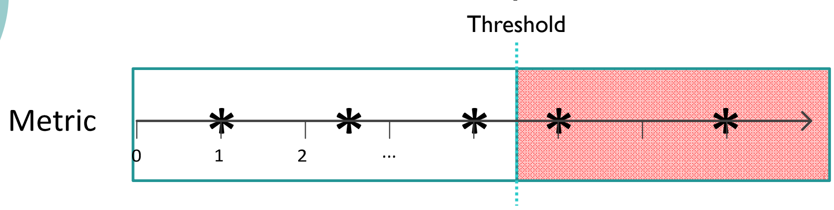


Contents

- Motivation
- Software Metrics
- **Classification with Thresholds**
- Optimization of Sets of Metrics
- Applications
- Case Studies
- Summary and Outlook

Thresholds

- Mechanism to classify values



- Metrics with upper and lower bound
  - Only upper bounds are considered

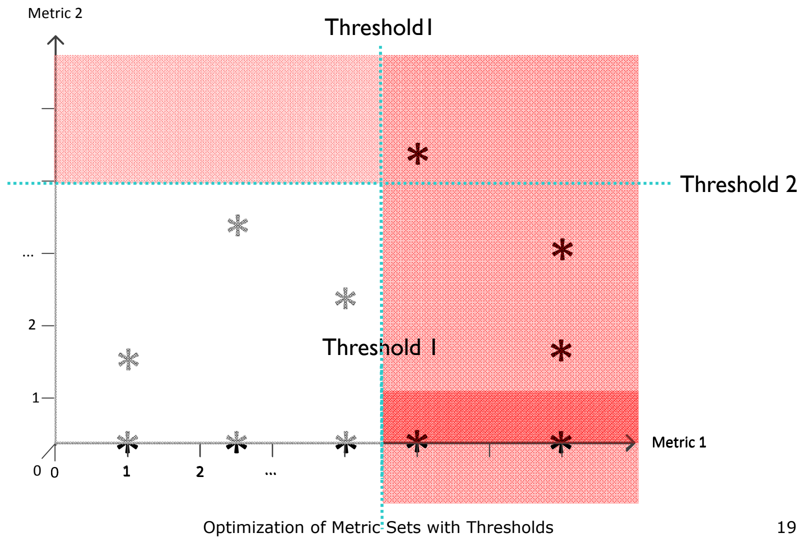
Thresholds for Methods

Name of Metric	Programming Language	Threshold
McCabe's Cyclomatic Number (VG)	C	24
	C++/C#	10
Nested Block Depth (NBD)	C/C++/C#	5
Number of Function Calls (NFC)	C/C++/C#	5
Number of Statements (NST)	C/C++/C#	50

Thresholds for Java Classes

Name of Metric	Threshold
Weighted Methods per Class (WMC)	100
Coupling Between Objects (CBO)	5
Response For a Class (RFC)	100
Number of Overriden Methods (NORM)	3
Lines of Code (LOC)	500
Number of Methods (NOM)	20
Number of Static Methods (NSM)	4

Thresholds and Rectangles



Contents

- Motivation
- Software Metrics
- Classification with Thresholds
- **Optimization of Sets of Metrics**
- Applications
- Case Studies
- Summary and Outlook

General Idea

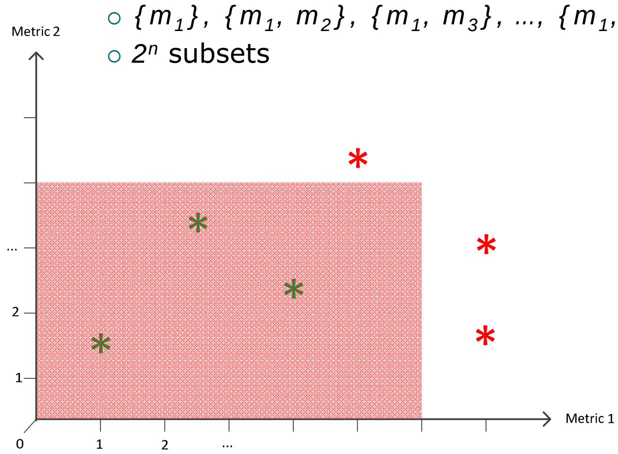
- Rectangles = sets of thresholds
- Rectangles are computed using machine learning
- Data-driven method
  - Based on previous measurements (or manual classification) of software
  - Measurements (or classification) partition the software into good and bad software

Optimization of Metric Sets

- Given:
  - Set of metrics: $M = \{m_1, \ldots, m_n\}$
  - Software system: $S = \{s_1, s_2, \ldots\}$
    - s_i = classes, methods or functions
    - Metric values $m_1(s_i), \ldots, m_n(s_i)$
    - Classification $f(s_i) \rightarrow good \vee bad$
- Sought-after:
  - Subset $M^* \subseteq M$ (including thresholds) with
    - $f_{M^*}(s_i) \approx f(s_i)$ and
    - $|M^*|$ is minimal

Calculation of Thresholds

- Calculate thresholds for all subsets
  - $\{m_1\}, \{m_1, m_2\}, \{m_1, m_3\}, \ldots, \{m_1, \ldots, m_n\}$
  - 2^n subsets



Selection of the Best Subset

- Determine classification error ε
  - deviation of metrics subset from input set
  - probability of $f_{M^*}(s_i) \neq f(s_i)$ (i.e., wrong classification)
- Select smallest subset with sufficient ε
  - ε ≤ δ for a selected error limit δ
  - δ = 1%
  - increase δ by 0,5% until a subset is found

Contents

Application Overview

- Size reduction of sets of metrics
  - Higher efficiency
- Simplification of classification
  - Better interpretation of classification
- Calculation of domain specific threshold
  - Automated quality assessment in organizations

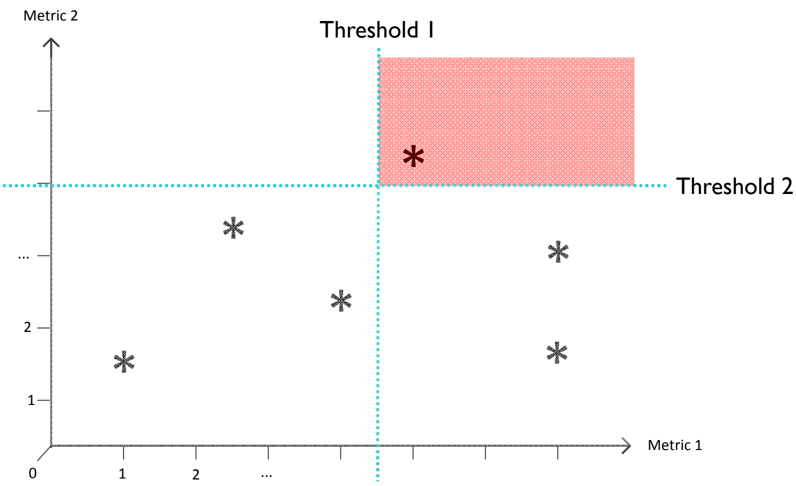
Size Reduction of Sets of Metrics

- Given
  - Set of metrics M with corresponding thresholds
- Classify software by means of M
- Calculate optimal subset $M^* \subseteq M$
  - M^* is more efficient than M

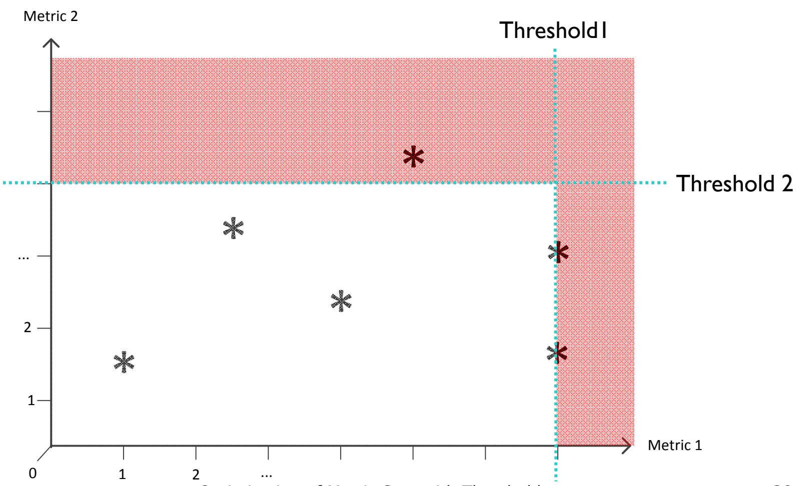
Simplification of the Classification

- Goal:
  - Using thresholds instead of a more complex classifier $f_{complex}$ such as
    - allowing certain violations of thresholds
    - decision trees
- Classify software S with classifier $f_{complex}$
- Select appropriate set of metrics M
- Calculation of an optimal subset $M^* \subseteq M$

Classification with one Violation



Approximation with Thresholds



Domainspecific Thresholds

- Assumption
  - No formal classifier available
- Expert provides base data
  - Manual classification of parts of a software product
  - Selection of metrics set *M* that may reproduce the manual classification
- Calculation of an optimal subset $M^* \subseteq M$

Contents

Data Pool

- Based on 8 open source projects

Name	Version	Programming Language	Size
Apache Webserver	2.2.10	C	6718 methods
kdebase	12/05/2008	C++	21404 methods
kdelibs	12/05/2008	C++	37444 methods
AspectDNG	1.0.3	C#	2759 methods
NetTopologieSuite	1.7.1.RC1	C#	3059 methods
SharpDevelop	2.2.1.2648	C#	15700 methods
Eclipse Java Development Tools	3.2	Java	4833 classes
Eclipse Platform Project	3.2	Java	5399 classes

Case Study: Optimization of Metric Sets 1(2)

- C functions

	VG	NBD	NFC	NST
Input	24	5	5	50
Optimized			5	

0.78% Error
75% Size Reduction!

- C++ methods and C# methods

	VG	NBD	NFC	NST
Input	10	5	5	50
Optimized			5	

0.06% Error, C++
0.59% Error, C#
75% Size Reduction!

Case Study: Optimization of Metric Sets 2(2)

- Java classes

	WMC	CBO	RFC	NORM	LOC	NOM	NSM
Input	100	5	100	3	500	20	4
Optimized		5		3			4

0.27% Error
57% Size Reduction!

Case Study: Usage of a Different Classifier 1(2)

- C functions – one violation is allowed

	VG	NBD	NFC	NST
Input	24	5	5	50
Optimized				50

0.84% Error
75% Size Reduction!

- C++ methods – one violation is allowed

	VG	NBD	NFC	NST
Input	10	5	5	50
Optimized	10			

0.87% Error
75% Size Reduction!

- C# methods – one violation is allowed

	VG	NBD	NFC	NST
Input	10	5	5	50
Optimized	9			

1.26% Error
75% Size Reduction!

Case Study:
Usage of a Different Classifier 2(2)

- Java classes – one violation is allowed

	WMC	CBO	RFC	NORM	LOC	NOM	NSM
Input	100	5	100	3	500	20	4
Optimized			98	3		20	4

1.71% Error
42% Size Reduction!

- Java classes – two violations are allowed

	WMC	CBO	RFC	NORM	LOC	NOM	NSM
Input	100	5	100	3	500	20	4
Optimized	99		110				

2.21% Error
71% Size Reduction!

Results of Case Studies

- Successful size reduction of metric sets
  - 42%-75% smaller sets
- Error in the range of statistical noise
- Complex classifications can be replaced by thresholds

Contents

Summary

- Optimization of metric sets with thresholds for quality assessment
  - Simple method with high effectiveness
- Data-driven method for the calculation of thresholds
  - Based on machine learning algorithms
- Complex classifications are replaceable by thresholds
  - Leads to a better interpretability of assessment results
- Case studies show that a small metric set is sufficient
  - Low effort for data collection

Outlook

- Disjunctive normal forms instead of simple thresholds
  - $(m_1 \wedge m_2) \vee (m_1 \wedge m_3 \wedge m_4)$
- Rating instead of classification
  - critical, suspect, unproblematic
- Metric sets on other levels of abstraction
  - modules, projects
- Inclusion of metrics for processes and resources
  - number of errors, test effort

- Thank you for your attention

Jens Grabowski
grabowski@informatik.uni-goettingen.de

For further details on the talk:
S. Herbold, J. Grabowski, S. Waack. *Calculation and Optimization of Thresholds for Sets of Software Metrics.* Accepted for publication in: Empirical Software Engineering, An International Journal. Springer, 2011.