
Verifysoft
TECHNOLOGY
SAD21
Softwaremetriken: Ein bewährtes Entwicklungswerkzeug oder nur ungeliebte Magic Numbers?
Offenburg, 11.05.2021
J. D. Baltzer, M. Sc.
Verifysoft Technology GmbH
baltzer@verifysoft.com
+49 781 127 8118-88

1


Agenda
Verifysoft
TECHNOLOGY
1. Einleitung
2. Historie
3. Theorie
4. Komplexitätsmetriken in der Praxis
5. Werkzeuge zur Generierung von Software-Produkt-Metriken
6. Quellen
7. Fragen
Jan-David Baltzer
Support & Training
© 2021 Verifysoft Technology GmbH
www.verifysoft.com
2

2

Einleitung

Zurück zum Studium:

-> Eine kleine Softwarefirma stellt sich vor.
-> Ein vollautomatisierter Entwicklungs/Build-Prozess wird vorgeführt.
-> Check in and Go…

- ✓ All targets built.
- ✓ Test 1 passed.
- ✓ Test 2 passed.
- ✓ Test 3 passed.
- ✓ …
- ✓ All tests passed.

Your Overall Number is 7!

3

Eine Frage

4

Was Nehmen wir mit?

verifysoft TECHNOLOGY

- Das Team war überschaubar, 3 Person.
- Die Firma war jung und ausgelastet.
- Die Zahl wurde durch die Toolchain automatisch generiert und eine Analyse des Features wäre mit Aufwand verbunden, den man lieber in andere Bereiche investiert.
- Es wurde direkt und offen kommuniziert, die Zahl ist halt „Magic“ und ohne Bedeutung.

Folgerung:

1. Die Handhabung in diesem Kontext ist nachvollziehbar und zweckmäßig.
2. Zahlen, deren Zweck und Grundlage nicht nachvollziehbar sind, stoßen auf wenig Akzeptanz.

5

Teamwechsel

verifysoft TECHNOLOGY

- Wir kommen in ein neues Team.
- Die Magic-Einstellung nehmen wir mit.
- Ansonsten hängen wir uns voll rein und leisten gute Arbeit.

6

Krisensitzung

Verifysoft TECHNOLOGY

Wir sind genervt, der Chef ist genervt und das alles wegen irgendwelcher Magic Numbers, auf die wir keinen Einfluss haben – oder etwa doch?

Kein Einfluss

Einfluss

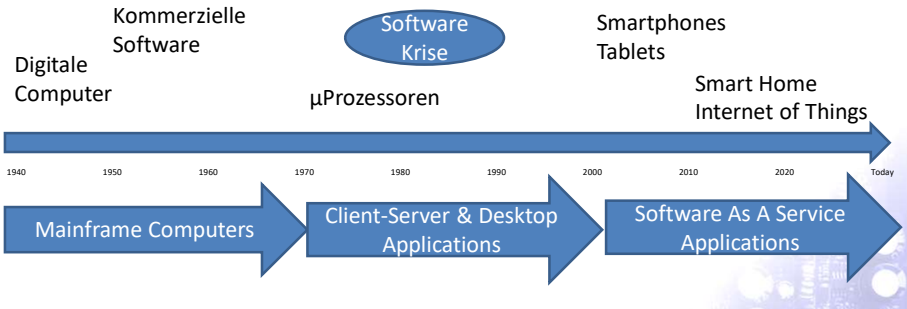
Irgendwas stimmt in dem Team nicht.

Es ist Aufklärung erforderlich.

7

Historie - Software

Verifysoft TECHNOLOGY



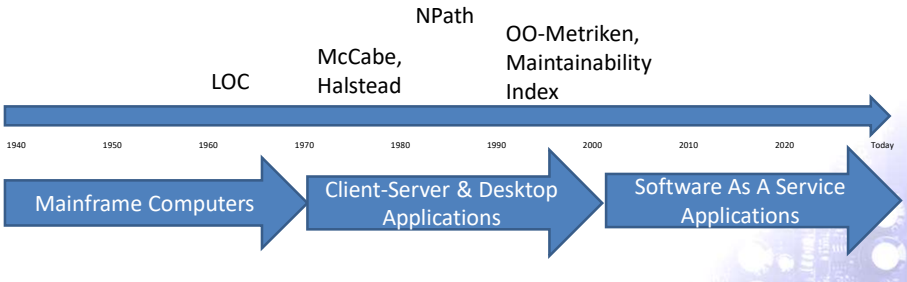
Fazit:
Es wurde immer mehr und mehr Software!

8

Historie - Metriken

Verifysoft TECHNOLOGY

Mit der Software erwuchs das Bedürfnis, diese zu messen…



Was für Metriken sind hier gelistet? (Paradigmen, Ebenen, Bestandteile)

9

Was für Softwaremetriken sind noch bekannt?

Verifysoft TECHNOLOGY

Anzahl Testfälle, total, passed, failed,

Entwicklerstunden, Anzahl Requirements, Anzahl Sprints,

Statische Analyse, TP, FP, TN, FN und Kompositionen,

Code Coverage, CC, MCC, PC,

Budget, Gewinn, Risiko,

Speicherbedarf, Laufzeit,

Fehlerdichte, pre-release, post-release, fixed,

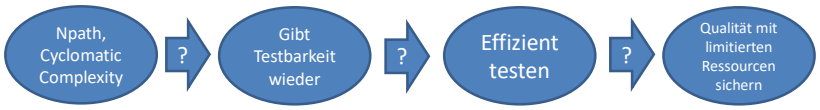
und viele, viele mehr. Kompositionen sind auch möglich.

10

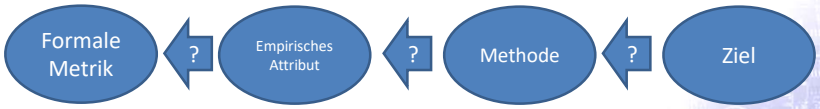
Warum Softwaremetriken?

verifysoft TECHNOLOGY

Beispiel:



Wir drehen den Spieß um und verallgemeinern:



Wir haben Model um Themen zu behandeln, welche uns wirklich interessieren.

11

Attribute und Metriken

verifysoft TECHNOLOGY

Metriken sind formale Messungen, welche vorgenommen werden, um Aussagen über empirische Attribute von Entitäten der realen Welt zu treffen.

Beispiel:
A() und B() seien Funktionen mit dem empirischen Attribut Testbarkeit.
Ca und Cb seien Komplexitätsmetriken dieser Funktionen.
Es gilt nun für:

Ca < Cb	A() ist testbarer als B()
Cb < Ca	B() ist testbarer als A()
Ca = Cb	A() und B() sind gleich testbar

Nach Chidamber & Kemerer 1994

12

Attribute und Metriken

Falls nun eine untestbare Funktion U() beobachtet wird, kann die repräsentative Komplexitätsmetrik Cu dieser Funktion als Konstante verwendet werden um beliebige weitere Funktionen als untestbar zu kategorisieren:

Für Konstante <= Cbeliebig Beliebig() ist untestbar

Problem:

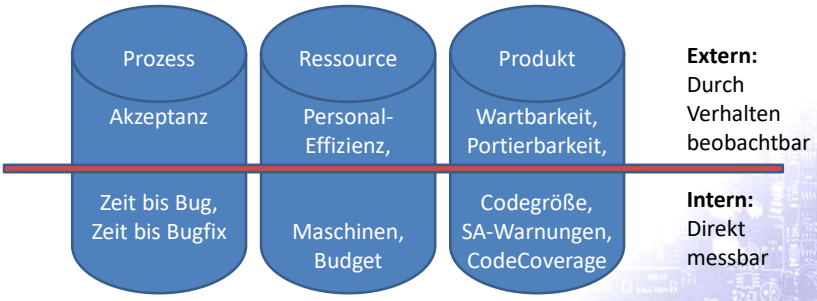
- Wie nachweisen, welche Metrik, wie zuverlässig ein Attribut wiedergibt?
- Es gibt einige Studien, welche sich daran versuchen.
  - Eine Feststellung ist, dass die Aussage von Metriken softwarespezifisch ist und von Parametern wie Entwicklungskultur, Kritikalität, Projektgröße, Programmiersprache, Targets und verwendeten Standards abhängt.
  - Ähnliche Projekte können mit dem selben Satz Metriken analysiert werden.
  - Ein bekanntes Beispiel für einen Satz Metriken, entwickelt für eine spezifische Domäne sind die HIS-Metriken im Automobilbereich.
  - Erfahrungswerte sind das wichtigste um passende Metriken zu finden!

13

Kategorisierung von Attributen

- Einige Modelle, welche eine gewisse Ähnlichkeit aufweisen.
- Zum Teil werden die Begriffe Attribut und Metrik nicht getrennt.

- Modell nach Fenton und Pfleeger 1996:



14

Komplexitätsmetriken in der Praxis

verifysoft TECHNOLOGY

Fallbeispiel aus Grundler et al 2018, über den Einsatz von Software-Produkt-Metriken in der Robert Bosch GmbH.

Problemstellung:
Wie kann Wartbarkeit gemessen und gesteuert werden?
Welche Produkt-Metriken geben wieder, welche Software-Komponenten schwer zu verstehen und zu entwickeln sind?

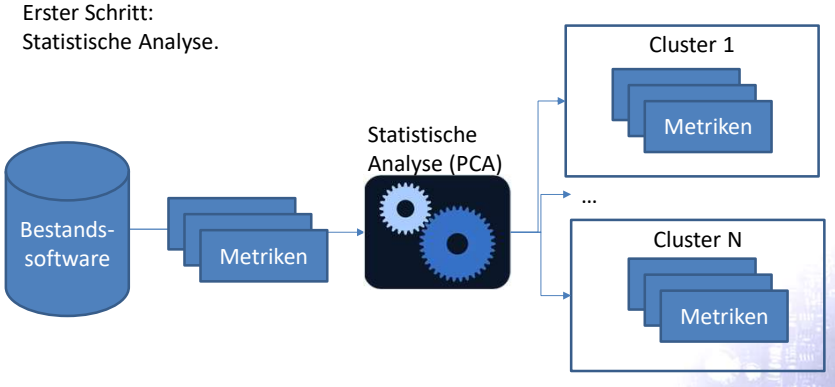
Durch Recherche wurde festgestellt, dass es zahlreiche Metriken und Werkzeuge gibt, doch deren Aussagekraft ist fragwürdig.

15

Komplexitätsmetriken in der Praxis

verifysoft TECHNOLOGY

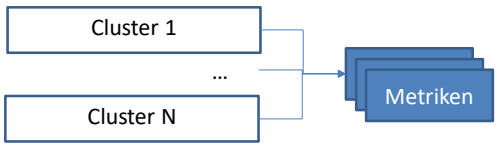
Erster Schritt:
Statistische Analyse.



16

Komplexitätsmetriken in der Praxis

Zweiter Schritt:
Selektion aussagekräftiger Repräsentanten.



Selektionskriterien:

- Jedes Cluster muss vertreten sein.
- Eine Verbesserung der Metrik muss eine Verbesserung der Software entsprechen (Steuerwirkung).
- Die Metrik muss leicht verständlich und damit nachvollziehbar/abschätzbar sein.
- Grenzwerte müssen selektierbar sein um beispielsweise nicht ganze Projekte als Kritisch zu markieren.

17

Komplexitätsmetriken in der Praxis

Dritter Schritt:

Aus den vorselektierten Metriken wurde in Zusammenarbeit mit der Entwicklung ein Subset von 6 HIS-Metriken mit Grenzwerten ausgewählt.

Damit hatte man einen Satz von Design-Metriken zur Unterstützung der tägliche Entwicklungsarbeit.

Die Validierung erfolgte statistisch als auch durch Erfahrungswerte der Entwickler selbst.

Eine überschaubare Sammlung von transparenten, praxisbewährten Metriken fördert zudem die Akzeptanz in der Entwicklung.

18

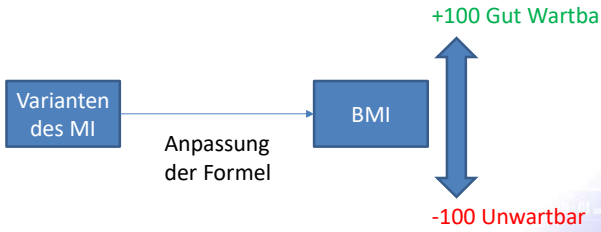
Komplexitätsmetriken in der Praxis

verifysoft TECHNOLOGY

Nächster Schritt:
Wartbarkeitsindex nach Oman und Hagemeister 1992.

Grund:
Das Projektmanagement benötigt eine einfache Metrik zur Steuerung/Priorisierung.

Validierung durch statistische Methoden, wobei die Kategorisierung der Metrik mit Expertenschätzungen verglichen wurde.



19

Komplexitätsmetriken in der Praxis

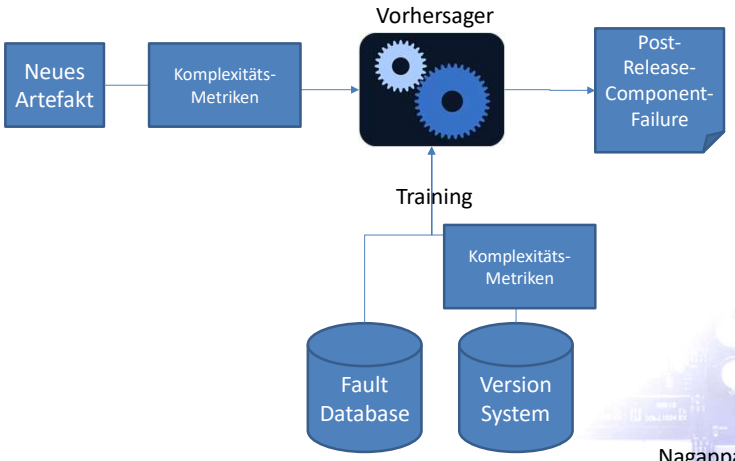
verifysoft TECHNOLOGY

Fazit:

- Metriken sind ein sinnvolles und wichtiges Entwicklungswerkzeug.
- Sie sind keine absoluten Schranken, an welche sich zum Selbstzweck gehalten werden muss.
- Sie dienen als Hilfsmittel beim Design und als Diskussionsgrundlage bez. Qualität und Wartbarkeit.
- Mehr Vertrauen in die Entwickler (Experten), als in Metriken zu haben, ist wichtig.
- Wenige, leicht verständliche Werte sind effektiver als eine Fülle von mystischen Magic Numbers.
- Metriken sollen die Entwicklung unterstützen und sie nicht ausbremsen.

20

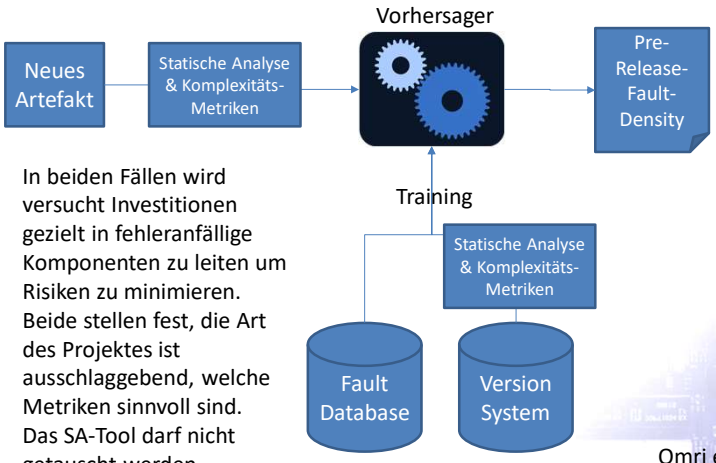
Prominente Arbeiten: Microsoft



Nagappan et al 2005

21

Prominente Arbeiten: Daimler



- In beiden Fällen wird versucht Investitionen gezielt in fehleranfällige Komponenten zu leiten um Risiken zu minimieren.
- Beide stellen fest, die Art des Projektes ist ausschlaggebend, welche Metriken sinnvoll sind.
- Das SA-Tool darf nicht getauscht werden.

Omri et al 2018

22

Metriken für kleinere Unternehmen

verifysoft TECHNOLOGY

Sie sind kein Microsoft, Daimler oder Bosch und können keine großangelegte Taskforce zur Eruierung der passendsten Metriken zu Ihrer Software stellen?

Aber:

- Sie kennen Ihre Software und Ihre Entwicklungskultur.
- Wenige, einfache Kenngrößen können enorm dabei helfen, die guten Attribute Ihrer Software zu verbessern.
- Orientierung bezüglich Metriken gibt es in der Literatur (Andere haben sich die Arbeit für Ihren Bereich vielleicht schon gemacht).
- Werkzeuge haben auch wir im Repertoire.

23

Werkzeuge: Imagix 4D

verifysoft TECHNOLOGY

Was ist Imagix 4D:

- ✓ Imagix 4D ist unsere Wahl um Quelltext zu erkunden, verstehen und zu bewerten (C++/Java).
- ✓ Es generiert eine Vielzahl (>130) von bekannten statischen Produktmetriken. Zudem können weitere Metriken hinzugefügt werden.
- ✓ Diese können direkt mit Imagix 4D analysiert und bewertet werden.
- ✓ Sie können auch dokumentiert oder exportiert und Anderweitig weiterverarbeitet werden.

Imagix

24

Werkzeuge: Testwell CMT++/Java

Was ist Testwell CMT++/Java:

- ✓ Das Code Complexity Measurement Tool ist eine leichtgewichtige Lösung zur Ermittlung von Komplexitätsmetriken aus dem Hause Verifysoft (C++/Java).
- ✓ Es ist leicht zu bedienen, effizient und generiert die gängigsten Komplexitätsmetriken zum Export in diverse Formate.

Testwell CMT++

Testwell CMTJava

25

Werkzeuge: GrammaTech CodeSonar

Was ist GrammaTech CodeSonar:

CODESONAR

- ✓ Das führende Werkzeug zur Detektion von Softwarefehlern und Sicherheitslücken (C++/Java/.NET/Python/Binary).
- ✓ Neben seiner Hauptfunktionalität generiert GrammaTech CodeSonar die gängigsten Komplexitätsmetriken.
- ✓ Analysen können über die Webschnittstelle erfolgen. Exporte und Berichte sind in diversen Formaten vorhanden.

26

Werkzeuge: Weitere

Verifysoft TECHNOLOGY

Verifysoft bietet noch folgende Werkzeuge an, welche Softwaremetriken im weiteren Sinne erzeugen:

- ✓ Testwell CTC++ — Code Coverage Data

Testwell CTC++

- ✓ GrammaTech CodeSentry — SBOM, Vulnerability Report

CODESentry GRAMMATECH

27

Quellen

Verifysoft TECHNOLOGY

Ein großes Dankeschön ergeht an die Autoren und Institutionen folgender verwendeter Quellen:

B. A. Nejmeh. NPATH: A MEASURE OF EXECUTION PATH COMPLEXITY AND ITS APPLICATIONS, Communications of the ACM, Volume 31, Number 2, 1988.

P. W. Oman & J. Hagemeister. Metrics for Assessing a Software System's Maintainability, Proceedings of the Conference on Software Maintenance, IEEE Computer Society Press, Los Alamitos, CA, 1992.

S. R. Chidamber & C. F. Kemerer. A Metrics Suite for Object Oriented Design, IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, VOL. 20, NO. 6, 1994.

N. Fenton & S. L. Pfleeger. Software Metrics – A Rigorous and Practical Approach, 2ed. International Thomson Computer Press, London, 1996.

K. D. Welker. The Software Maintainability Index Revisited, CROSSTALK - The Journal of Defense Software Engineering, 2001.

S. Alexandre. Software Metrics – An Overview – Version 1.0. CETIC asbl – University of Namur, Software Quality Lab, Belgium, 2002.

N. Nagappan & T. Ball & A. Zeller. Mining Metrics to Predict Component Failures, Microsoft Research, USA & Saarland University, Germany, 2005.

A. M. Salem & A. A. Qureshi. Analysis of Inconsistencies in Object Oriented Metrics, Journal of Software Engineering and Applications, 2011.

S. Omri & P. Montag & C. Sinz. Static Analysis and Code Complexity Metrics as Early Indicator of Software Defects, Journal of Software Engineering and Application, 2018.

T. Grundler & H. Post & J. Quante & S. Yigit. 42 Jahre Komplexitätsmetriken – Was stoppt uns? Embedded Software Engineering Kongress, Tagungsband, Sindelfingen, 2018.

K. Kandt. Software Design Principles and Practices, Jet Propulsion Labratory, Version Q1 2021.

Quality: The Missing Metric for Software Development Managers, https://www.sealights.io/learn/quality-the-missing-metric-for-software-development-managers , Version Q1 2021.

thehistroyofsoftware, https://www.capterra.com/history-of-software , Version Q1 2021.

Bilder von Openclipart.org

28

Fragen

verifysoft TECHNOLOGY

Vielen Dank für Ihre Aufmerksamkeit!

Gibt es Fragen?