Diplomarbeit

Studiengang Geoinformatik

Thema:

Statistische Analyse von Softwarequalitätsmetriken
basierend auf Softwareprojektarchivdaten

URN:
urn:nbn:de:gbv:519-thesis2008-0273-8

von: Henrike Barkmann
Matrikelnummer: 282404

Betreuer:
Rüdiger Lincke, Phil. Lic., Växjö Universität
Prof. Dr.-Ing. Andreas Wehrenpfennig, Hochschule Neubrandenburg

Neubrandenburg, 23. Oktober 2008

Erklärung

Hiermit erkläre ich, dass ich die vorliegende Diplomarbeit selbständig angefertigt habe. Es wurden nur die in der Arbeit ausdrücklich benannten Quellen und Hilfsmittel benutzt. Wörtlich oder sinngemäß übernommenes Gedankengut habe ich als solches kenntlich gemacht.

_____ _____

Ort, Datum Unterschrift

Kurzfassung

In dieser Diplomarbeit wird die Notwendigkeit von Metriken eines Softwarequalitätsmodelles untersucht und eine Schwellenwertbestimmung für diese Metriken hinsichtlich der Ausreißereleminierung gemacht.
Es werden Grundlagen über Softwarequalität und Metriken, genutzte Tools und statistische Verfahren erläutert.
Der zentrale Punkt dieser Arbeit ist die Gewinnung, Aufbereitung und Auswertung von Daten aus Open Source Projekten mit Hilfe des VizzAnalyzer-Tools, um gestellte Hypothesen bezüglich der Schwellenwerte und Korrelationen zu prüfen und aufgeworfene Fragestellungen zu beantworten.

Abstract

In this diploma thesis the need of metrics of a software quality model is verified and a threshold value determining for these metrics concerning the outlier elimination is made. Basics over software quality and metrics, used tools and statistic procedures are described. The central point of this work is the mining, the processing and analysing of data from open source projects with the help of the VizzAnalyzer tool, in order to check hypotheses and to answer posed questions concerning the threshold values and correlations.

Inhaltsverzeichnis

1 Einleitung **1**

2 Grundlagen **3**

2.1 Qualität von Software 3

2.2 Metriken 3

2.3 VizzAnalyzer und sein Export-Tool 4

2.4 Softwareprojektarchive und Open Source 4

2.5 Projekte und ihre Metadaten 5

2.6 Statistische Grundlagen 6

2.6.1 Verteilungskurven 6

2.6.2 Schwellenwerte 7

2.6.3 Korrelation 8

3 Hypothesen **11**

3.1 Hypothese 1 11

3.2 Hypothese 2 12

4 Datenerfassung **13**

4.1 Überblick über die Datenerfassung 13

4.2 Erfassen und Vorbereiten des Projektsourcecodes 14

4.3 Erfassen der Metadaten 15

4.4 Vermessen der Software und Export der Rohdaten 15

5 Auswertung der Hypothesen **17**

5.1 Hypothese 1: Schwellenwerte 17

5.1.1 Datenaufbereitung 18

5.1.2 Ergebnisse 19

5.1.3 Analyse 34

5.2 Hypothese 2: Zusammenhänge zwischen Metriken 37

5.2.1 Datenaufbereitung 37

5.2.2 Ergebnisse 38

5.2.3 Analyse 43

6 Evaluierung **45**

6.1 Schwellenwerte 45

6.2 Korrelationen 47

6.3 Fazit der Evaluierung 50

7 Abschlussbetrachtung **51**

A Anhang **53**

A.1 Metriken 53

A.2 Projektliste 57

A.3 Inhalt der DVD 61

Abbildungsverzeichnis **VI**

Tabellenverzeichnis **VII**

Verzeichnis der Literatur- und Internetquellen **XI**

1 Einleitung

Die heute entwickelte und eingesetzte Software wird immer komplexer und verursacht damit steigende Wartungskosten. Um diese zu verringern bzw. in Grenzen zu halten werden Qualitätsbewertungen notwendig, durch welche eine Abschätzung der Kosten und Problemfelder möglich wird. Dieses geschieht derzeit vorallem manuell.
Ein Forschungsprojekt der Universität Växjö befasst sich mit dieser Problematik und untersucht unter anderem die Möglichkeiten der automatischen Bewertung von Software basierend auf der Nutzung von Metriken. Dieses erfolgt aktuell mit Hilfe von Softwarequalitätsmodellen. [Lin07]
Aus diesen Bemühungen heraus ergeben sich u.a. folgende Fragestellungen:

- Was ist ein angemessener Schwellenwert für Metriken?
- Werden alle Metriken des Softwarequalitätmodells benötigt?

Ein dabei auftretendes Problem ist das Fehlen von größeren Datenmengen, um aussagekräftige Analysen durchführen zu können.

Problemstellung

Zur Gewinnung einer aussagekräftigen Menge an Daten ist das Definieren und Anwenden eines Prozesses notwendig, mit dessen Hilfe eine große Menge an Softwareprojektdaten gesammelt und archiviert werden kann.
Diese zusammengetragenen Daten sollen zur Beantwortung noch offener Fragen beitragen. Hierfür muß geklärt werden, woher die Daten bezogen werden können. Von diesen Projekten muß dann sowohl der Sourcecode zur Verfügung stehen, als auch die Metadaten. Außerdem ist das Kompilieren dieser Daten eine Voraussetzung zu deren Nutzung.

Motivation

Um das Verständnis für die Softwarequalitätsbewertung mit Hilfe von Metriken zu erhöhen, sollen verschiedene Fragen beantwortet werden. Zum einen soll der Sinn der bisher gültigen Schwellenwerte hinsichtlich der Ausreißerbeurteilung betrachtet werden. Zum anderen ergibt sich die Frage, ob alle Metriken des Softwarequalitätsmodells, wie es zur Zeit vorliegt,

benötigt werden, oder ob durch Ausschließen einiger dieser Maßzahlen der Aufwand reduziert werden kann.

Ziele

Um die Fragestellungen beantworten zu können und somit die Ziele dieser Arbeit zu erreichen, sind mehrere Arbeitsschritte notwendig. Dazu gehört das Sammeln von Daten zur Analyse und zum Testen von Hypothesen. Diese Daten sollen repräsentativ sein, in ausreichender Zahl vorliegen und müssen einfach abrufbar und analysierbar sein. Dieser Vorgang des Datensammelns muß auf jeden Fall wiederholbar sein.
Ebenso wichtig ist die Beschreibung der gesammelten Daten, was mit Hilfe von statistischen Hilfsmitteln, wie Mittelwert, Verteilungskurven und Häufigkeiten geschieht.
Die Beantwortung der Fragen mittels der Hypothesen und Analysen ist ein weiterer Schritt, welcher mit sorgfältiger Bearbeitung der Daten und Dokumentation der Analysen und Ergebnisse einhergeht.
Die Grenzen für die Genauigkeit und Möglichkeiten werden gesetzt durch die Nutzung von Standardtechnologien wie Excel, Eclipse, Java Programmen und dem VizzAnalyzer.

Inhalt

Die Arbeit gliedert sich in sieben Kapitel. In diesem Kapitel wurden die Problemstellung und Ziele dieser Arbeit beschrieben. Kapitel 2 beschreibt Hintergrund und Grundlagen von Softwarequalitätsanalysen, der Datenerfassung und statistischen Analysen inklusive der verwendeten Werkzeuge. In Kapitel 3 werden die Fragestellungen und Hypothesen benannt und erläutert. Kapitel 4 zeigt den Weg der Datensammlung vom Erfassen der Daten und Metadaten über das Vermessen der Rohdaten bis hin zum Export. Kapitel 5 stellt speziel die Datenaufbereitung für die einzelnen Hypothesen dar, sowie die Präsentation der Ergebnisse und deren Analyse. Auf das Bewerten der Ergebnisse wird in Kapitel 6 eingegangen und in Kapitel 7 wird ein Fazit gezogen und der Ausblick auf Zukünftiges gegeben.

2 Grundlagen

Dieses Kapitel erläutert die Grundlagen, welche für das Verständnis der Arbeit benötigt werden. Zuerst wird Softwarequalität, so wie sie im Rahmen der vorliegenden Arbeit verstanden wird, erklärt. Dann wird auf den Begriff Metriken näher eingegangen. Anschließend wird das Tool näher erläutert, mit welchem die Analysen gemacht werden, wobei analysieren hier das Ermitteln der Masszahlen bedeutet und erklärt, worum es sich bei Softwareprojektarchiven und Open Source handelt. Im Anschluss wird auf die gesammelten Projekte und ihre Metadaten eingegangen.
Abschließend betrachtet dieses Kapitel die statistischen Grundlagen wie Verteilungskurven, Schwellenwerte und Korrelationen.

2.1 Qualität von Software

Diese Arbeit stützt sich auf die Definitionen von Softwarequalität, die dem VizzAnalyzer-Projekt zugrunde liegen.
Hier ist festgelegt, welche der im Qualitätsmodell festgelegten Bedingungen (basierend auf ISO 9126-3 [ISO03]) ein Softwaresystem erfüllen muss, um eine gute bzw. hohe Qualität zu besitzen. Zu diesen Bedingungen gehören bestimmte Faktoren und Kriterien, wie die Funktionalität, Effizienz, Wartbarkeit, Wiederverwendbarkeit, Verlässlichkeit und Übertragbarkeit, sowie die Komplexität, Architektur und Struktur und das Design. Auch die Metriken und deren Beziehungen zueinander zählen dazu. [Lin07]

2.2 Metriken

In diesem Zusammenhang werden Metriken benötigt, um die Qualität von Software einschätzen zu können.
Sie werden mit Hilfe von Eigenschaften einer Software berechnet, zu welchen u.a der Zusammenhang, die Komplexität, die Größe und auch der Grad der Dokumentation gehören. Diese Eigenschaften werden durch die Analyse des Sourcecodes eines Systems mit dem VizzAnalyzer zusammengetragen.
Das heißt, mit Hilfe der vom VizzAnalyzer berechneten Metriken werden aus den Daten der Softwareprojekte analysierbare Maßzahlen ermittelt.

Es gibt mehrere Arten von Metriken, die sich im Wesentlichen in Design-Metriken und Komplexitäts-Metriken unterscheiden.
Die wohl bekannteste Metrik ist die Lines of Code (LOC) Metrik, die die Größe eines Programmes mit der Anzahl von Codezeilen in den einzelnen Dateien abstrahiert. Allgemein sind Metriken Maßzahlen, die den Zusammenhang von Methoden und Feldern innerhalb einer Klasse und die Verbindungen zwischen Klassen beschreiben[1]. [Lin07, LL06]

2.3 VizzAnalyzer und sein Export-Tool

Der VizzAnalyzer ist ein Werkzeug zur Berechnung von Metrikwerten und zur Analyse von Softwarequalität. Es liest Softwarecode, andere Designspezifikationen und Dokumentationen ein und trägt dazu bei Qualitätsanalysen durchführen zu können.
Um die Daten aus den Softwarearchiven analysieren und einschätzen zu können, wird das „Express Multi DB Export“-Tool des VizzAnalyzers genutzt, welches die berechneten Ergebnisse aus dem VizzAnalyzer in eine Datenbank exportiert.
Berechnet werden verschiedene Metrikwerte, die anschließend über eine Open Database Connectivity (ODBC) Schnittstelle in eine Datenbank exportiert werden. Verwendet wird eine Microsoft Access-Datenbank, auf die unter anderem auch von Excel aus direkt zugegriffen werden kann.
In die Datenbank wird unter anderem Folgendes geschrieben:

- der Name des Projektes und das Downloaddatum,
- berechnete Klassen- und Packagemetriken,
- weitere Statistiken[2]

  und

- Metadaten.

2.4 Softwareprojektarchive und Open Source

Softwareprojektarchive - oder auch Software Repositorys - werden genutzt, um Dateien, Programme und Metadaten zu verwalten.

[1] Eine Liste der hier verwendeten Metriken samt Erläuterungen ist im Anhang A.1 zu finden.
[2] Diese sollen im Rahmen dieser Arbeit nicht näher betrachtet werden.

Mit Hilfe von verschiedenen Managementsystemen werden Installationen, zeitliche Entwicklungen, die Kommunikation zwischen Projektmitwirkenden und unterschiedliche Versionen verwaltet, protokolliert und archiviert. So ist es relativ einfach möglich die Systementwicklung zu dokumentieren und frühere Zustände zu rekonstruieren. Dieses hilft dabei, den Fortschritt und Ablauf eines Softwareprojektes zu steuern.
Mögliche Versionsmanagementsysteme sind CVS (Concurrent Versions System) [PXI08] und SVN (Subversion) [Tig08], die zur Versionsverwaltung von Dateien und Verzeichnissen eingesetzt werden, so dass Konflikte vermieden werden können.
Über diese wird z.B. der zu bearbeitende bzw. bearbeitete Quellcode aus- und eingecheckt, dass heißt neue Versionen und Dateien können zur Verfügung gestellt oder bezogen werden.
Diese Versionskontrolle und die Verwendung von Softwareprojektarchiven sind in der Industrie immer häufiger anzutreffen, da sie dort eingesetzt wird, wo ein Team von Personen zusammenarbeitet und Versionen verwaltet werden. Vermehrt sind diese Systeme auch bei nicht kommerziellen Projekten und im privaten Bereich zu finden, so auch in der Open Source Gemeinde.
Hier ist SourceForge die größte Entwicklung zum Austausch von Informationen und zur Verbreitung von Open Source Projekten. Auch ist es das größte Projektarchiv für Open Source Code und Applikationen [Wik08]. Alle hier archivierten Projekte gehören in den Open Source Bereich und können damit frei heruntergeladen, verwendet und bearbeitet werden. Dieses setzt voraus, dass veränderter Quellcode und Funktionsweisen ebenso der Öffentlichkeit zur Verfügung gestellt werden.

2.5 Projekte und ihre Metadaten

Für die statistische Analyse wird eine große Menge an Daten benötigt. Diese Daten werden aus Projekten gewonnen, welche sowohl auf SourceForge zur Verfügung gestellt werden, als auch unter www.java-source.net aufgelistet sind.
Dabei wird darauf geachtet, an Projekte zu gelangen, die in Java geschrieben wurden, ohne subjektive Einschränkungen zu machen [3].
Neben den rein projektbezogenen Daten wie dem Sourcecode, werden häufig auch Metadaten gespeichert, die ebenfalls für die statistische Analyse von Interesse sind. Diese gehen

[3] Eine Liste aller verwendeten Projekte befindet sich im Anhang A.2.

vom Anwendernamen, über Kommentare und Revisionsnummer, bis hin zu komplexen Kategorien. Speziell in Source Forge gehören unter anderem die folgenden Metadaten zu den Projekten: Development Status, die Anzahl der Bugs, der Unixname, die Programmiersprache und die Kategorie. Unter www.java-source.net sind diese nicht direkt detailliert aufgeführt.

2.6 Statistische Grundlagen

Die für die Beschreibung und Auswertung der gesammelten Daten herangezogenen statistische Verfahren werden im Nachfolgenden kurz erläutert.

2.6.1 Verteilungskurven

Verteilungskurven stellen graphisch die Häufigkeiten von Messwerten dar, wobei die Werte der Größe nach sortiert sind und die Skala über den Bereich der Stichprobe in Klassen aufgeteilt wird. Diese werden verwendet, um einen Überblick über die Daten und Messwerte zu schaffen, ohne diese einzeln betrachten zu müssen.
Die Ausprägung und somit die Form von Verteilungskurven können sehr unterschiedlich sein (Abb. 1). Möglich sind zum Beispiel

- ein oder mehrere Gipfel,
- Symmetrie oder Asymmetrie,
- Rechtsschiefe oder Linksschiefe.

Um die ungefähre Ausprägung einer Kurve festzustellen, kann ein einfacher Wertevergleich herangezogen werden. Hierfür werden der Modus, der häufigste Wert der Häufigkeitsverteilung, der Median, welcher die Verteilung halbiert und der Mittelwert oder arithmetrisches Mittel benötigt [Die01]:

- Modus < Median < Mittelwert => rechtsschief
- Modus = Median = Mittelwert => unimodal und symmetrisch
- Modus > Median > Mittelwert => linksschief

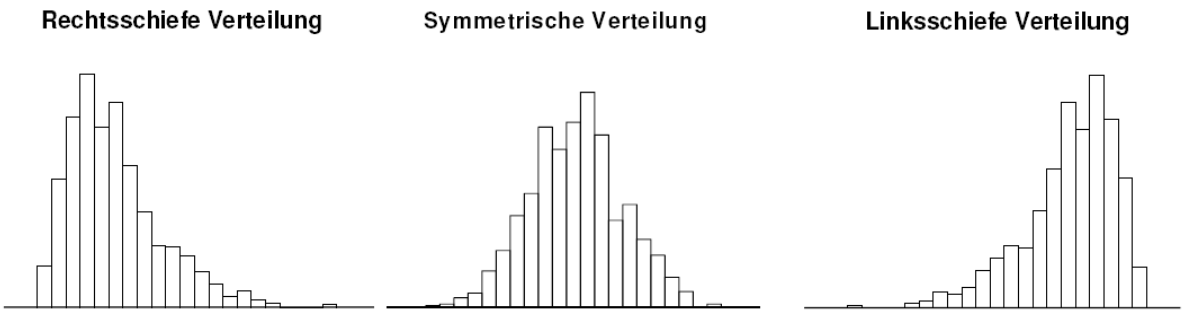


Abbildung 1: Formen der Verteilungsausprägung [Mue08]

Die Kurve einer Normalverteilung, die als Idealform einer Häufigkeitsverteilung gilt und den Namen „Gauß-Kurve“ oder „Gauß-Glocke“ trägt, hat unter anderem folgende Eigenschaften:

- Sie hat die Form einer Glocke,
- ist unimodal und symmetrisch,
- der Modus, der Median und der Mittelwert fallen zusammen,
- die Schiefe ist 0,
- sie nähert sich asymptotisch der x-Achse

  und
- zwischen den Wendepunkten befindet sich ca. 2/3 der Fläche.

Es gibt unendlich viele Normalverteilungungen, wobei die Verteilung mit einem Mittelwert von 0 und einer Streuung von 1 als Standardnormalverteilung betrachtet wird.
Ein Test auf Normalverteilung ist unter anderem dann sinnvoll, wenn über dieser Verteilung statistische Verfahren angewendet werden sollen, da Verfahren existieren, die eine normalverteilte Wertemenge voraussetzen. Sie kann also in manchen Fällen die Grundlage für statistische Tests und Rechenanalysen bilden.

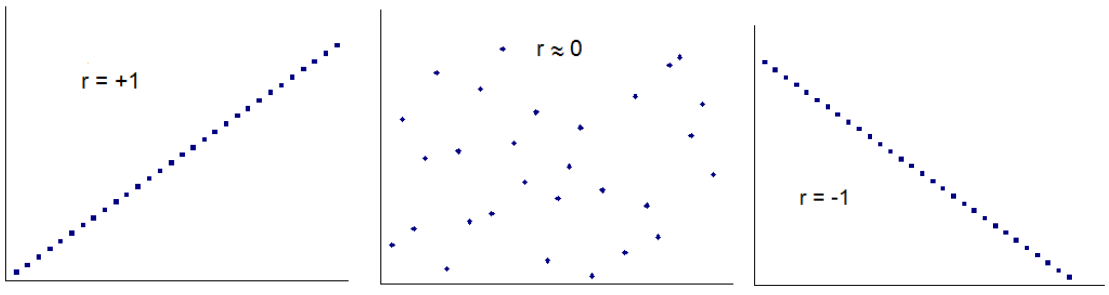
2.6.2 Schwellenwerte

Schwellenwerte werden eingesetzt, um Ausreißer zu erkennen, wobei als Ausreißer Werte betrachtet werden, die nicht in den Erwartungraum bzw. in die erwartete Messreihe passen. Es sind also Werte, die sich stark von anderen Werten einer Stichprobe abheben und

einen meist ungewollten Einfluss auf Berechnungen und Verteilungen haben, was dazu führt, dass diese eleminiert werden.
Für die Analyse von Projekten hinsichtlich ihrer Qualität bedeutet dieses, dass die Klassen und Projekte, welche diese festgelegten Schwellen überschreiten bzw. unterschreiten, wahrscheinlich keine gute Qualität im Bezug auf die im Qualitätsmodell festgelegten Faktoren und Kriterien haben. (Kap. 2.1)
Durch die Schwellen kann also eingegrenzt werden, in welchem Bereich ein Analyseergebnis liegen sollte, damit dieses als annehmbar gelten kann.

2.6.3 Korrelation

Ein Korrelationskoeffizient ist ein Maß für die Stärke einer stochastischen Abhängigkeit zweier Werte und kann einen beliebigen Wert zwischen +1 und -1 annehmen.
Ergibt sich ein positiver Wert, existiert ein gleichläufiger Zusammenhang, bei einem negativen besteht ein gegenläufiger.
Ist das Ergebnis 0, sind die Zufallsvariablen nicht korreliert.
Bei einem Wert von ± 1 zeigt dieser einen absoluten Zusammenhang an.
Bildet man die Variablen mit Hilfe eines Punktdiagrammes ab, kann durch die Verteilung der Punkte ebenfalls darauf geschlossen werden, wie hoch die Korrelation in etwa ist.
Bei einer unstrukturierten Punktwolke geht der Korrelationskoeffizient gegen 0 (Abb. 2b).
Erstreckt sich diese entlang einer Linie, geht der Korrelationskoeffizent gegen ± 1(Abb. 2a und c).



(a) (b) (c)

Abbildung 2: Korrelationsausprägungen

Eine Berechnung dieser Zusammenhänge ist dann sinnvoll, wenn zwischen den Variablen ein sachlogischer Zusammenhang zu finden ist, da sonst die Gefahr der „Nonsenskorrela-

tion“ besteht.[Nie08]
Die bekannteste Variante der Berechnung nutzt die Kovarianz und ist nach Karl Pearson benannt.

$$r = \frac{Cov(X, Y)}{\sigma_x \cdot \sigma_y} \tag{1}$$

Diese Formel (1) ist dann einzusetzen, wenn folgende Bedingungen erfüllt sind:

- Skalierung: Intervallskala oder dichotome Daten
- Normalverteilung
- Linearität
- Signifikanzbedingung

Ist auch nur eine dieser Bedingungen nicht erfüllt, muß auf die Rangkorrelation (z.B. von Spearman) zurückgegriffen werden.[PDKR08]
Diese beruht nicht, wie bei Pearson, auf den Originalmessdaten, sondern auf den Rangzahlen dieser Werte und berechnet sich wie folgt [Erl08]:

- Ränge zuordnen
- Differenz zwischen den Rängen bilden
- Quadrate der Differenz bilden und aufsummieren (je höher die Summe der Quadrate, desto niedriger die Rangkorrelation)
- Berechnung der Koeffizienten mit Hilfe der Formel (2), wobei D_i die Differenz der Rangzahlen darstellt [www08]

$$r_s = 1 - \frac{6 \sum_{i=1}^{n} D_i^2}{n(n^2 - 1)} \tag{2}$$

Voraussetzungen für diese Art der Berechnung:

- ein monotoner Zusammenhang, es ist keine Linearität notwendig
- es liegen mindestens Ordinalskalen vor

Die Einstufung der Ergebnisse der Korrelationskoeffizientenberechnung ist nur als ungefähre Angabe zu finden, da diese verschieden ausgelegt werden können. Diese Unterteilung richtet sich nach den Anforderungen und dem Fachgebiet eines Problems und variiert daher bei jeder statistischen Untersuchung.
Eine Variante ist folgende [Köl08]:

0 bis $<$ 0,1 kein Zusammenhang

0,2 bis $<$ 0,4 geringfügiger/schwacher Zusammenhang

0,4 bis $<$ 0,8 mittlerer/mittelstarker Zusammenhang

0,8 bis $<$ 0,9 starker Zusammenhang

0,9 bis $<$ 1 sehr starker Zusammenhang

1 absoluter Zusammenhang

3 Hypothesen

In der Einleitung wurde als Kontext für diese Arbeit ein Forschungsprojekt benannt, welches als Ziel hat, eine Korrelation zwischen Softwarqualitätsmetriken und Softwarequalität nachzuweisen. Fragen, die im Verlauf dieses Projekts aufgekommen sind, wurden dargelegt.
In diesem Kapitel werden nun jeweils die Nullhypothesen (H_0) und Alternativhypothesen (H_1) dieser Fragestellungen diskutiert.

3.1 Hypothese 1

In der ersten Fragestellung geht es um einen angemessenen Schwellenwert für Metriken bzw. um den Aussagegehalt und die Anpassung der bisher gewählten Schwellenwerte.
Des Weiteren ergibt sich die Frage nach dem Sinn eines absoluten Wertes bezüglich der bisher relativ zum Projekt berechneten Schwellen, wenn diese über eine große Anzahl von Projekten bestimmt, also Schwellenwerte Projekt übergreifend berechnet werden.
Schwellenwerte spielen beim Eleminieren von Ausreißern eine große Rolle. (Kap. 2.6.2) Auch kann durch diese der Wertebereich eingeschränkt werden, in welchem ein Großteil aller Werte liegt und welcher daher vermutlich als relativ optimal eingestuft werden kann. Bisher wurde mit einem Wert gearbeitet, welcher willkürlich festgelegt wurde. Dieser Schwellenwert von 15% war für alle Metriken zu nutzen und zum System relativ. Da die Ergebnisse der Metriken nicht immer normalverteilt sein müssen, kann es bei diesem festen Wert von 15% der Werteskala passieren, dass Werte aus der Betrachtung genommen werden, die evtl. als ideal gelten könnten. 15% der Werteskala entsprechen 15% der Spannweite, also vom kleinsten bis zum größten vorkommenden Wert einer Verteilung. Die Schwellen werden beidseitig gemessen und somit bleiben 70% der Werteskala erhalten. Es soll für jede Metrik getestet werden, ob es optimalere Schwellenwerte gibt, also Werte, welche besser auf die Ausreißer und die Klassenanzahl eines Projektes eingehen als der bisher genutzte.

H_0: 15% der Werteskala ist der optimale Schwellenwert.

H_1: Mindestens für eine Metrik existieren optimalere Schwellenwerte.

3.2 Hypothese 2

Die zweite Fragestellung, welche sich aus den bisherigen Betrachtungen des Forschungsprojektes ergibt, betrifft die Notwendigkeit aller Metriken des Qualitätsmodells. Hierbei wird der Ansatz verfolgt, den Aufwand zu verringern, indem einzelne Metriken weggelassen werden.
Die zweite Hypothese bezieht sich dementsprechend auf die Zusammenhänge von Metriken untereinander.
Es soll aufgezeigt werden, ob die vom VizzAnalyzer berechneten und exportierten Metriken voneinander abhängig sind.
Hierzu muß beantwortet werden, welche Metriken einen Zusammenhang aufweisen. Dieses geschieht über die Messung der Korrelation zweier Metriken.

H_0: Alle Metriken sind voneinander unabhängig.

H_1: Mindestens ein Paar ist voneinander abhängig.

4 Datenerfassung

In diesem Kapitel wird die Erfassung der Daten beschrieben, welche zur Auswertung der genannten Hypothesen notwendig sind. Hierzu gehört ein Überblick über die Daten, deren Herkunft und die an die Daten gestellten Anforderungen sowie die Erfassung der Daten und ihrer Metadaten. Auch das Vermessen der Software und der Export der Rohdaten in eine Datenbank wird beschrieben. Der Ablauf der Datenerfassung ist in Abbildung 3 verdeutlicht.

Abbildung 3: Ablauf der Datenerfassung

4.1 Überblick über die Datenerfassung

Die Projekte für die Analyse müssen in erster Linie zwei Voraussetzungen erfüllen. Zum einen müssen sie in Java geschrieben worden sein, zum anderen wird der Sourcecode für die Vermessung mittels VizzAnalyzer benötigt. Die zweite Bedingung wird durch Projekte aus dem Open Source Bereich erfüllt, weswegen diese entweder von www.sourceforge.net oder www.java-source.net bezogen werden. Ansonsten werden die Projekte ohne Blick auf

Größe, Kategorie, Status oder andere Eigenschaften ausgewählt.
Da es allein auf sourceforge.net mehr als 16.000 Projekte auf Javabasis gibt und nicht alle Projekte in dem gesetzten zeitlichen Rahmen der Arbeit analysiert werden können, wird sich auf eine Stichprobe beschränkt. Mit den vorhandenen Resourcen ist es möglich 70 verschiedene Projekte mit insgesamt über 30000 Klassen zu analysieren und für die Datensammlung zu verwenden. Unter Zuhilfenahme von Daten, die in einer Bachelor-Thesis [DOG08], welche am gleichen Lehrstuhl betreut wurde, gesammelt wurden, kann die Stichprobe auf ungefähr den doppelten Umfang gebracht werden, wobei die endgültig verwendete Datenmenge auf eine Stichprobe von 145 Projekten mit über 54000 Klassen zurückgreift.

4.2 Erfassen und Vorbereiten des Projektsourcecodes

Die Daten werden auf drei Arten heruntergeladen:

1. per SVN,
2. per CVS

   oder

3. über die bereitgestellten Downloads auf sourceforge.net oder von den Hersteller-/Entwickler-Seiten

Heruntergeladen wird jeweils die neueste zur Verfügung stehende Version des Sourcecodes, welcher anschließend oder direkt in Eclipse hineingeladen wird.
Hier ist es notwendig, dass die Projekte in Eclipse keine Fehlermeldungen hervorrufen, kompilieren und somit vom VizzAnalyzer verwendet und vermessen werden können.
Ein Teil der Programme ist direkt nach dem Download für die weitere Verwendung nutzbar, ein anderer Teil ist zuvor mit Hilfe von Libraries und verschiedenen jdk- und JUnit-Versionen aufzubereiten.
Um eine Ordnung und Eindeutigkeit in die Liste der Programme zu bringen, setzt sich der Name dieser Projekte zusammen aus:

- dem Unix-Namen, welcher in sourceforge.net angegeben ist

  oder

- einem selbst gewählten Namen [4]

  und

- dem Download-/Versions-Datum, in dem Format jjmmtt.

Durch diese Namenskonvention können verschiedene Versionen eines Programmes unterschieden werden und die Metadaten, welche mit Hilfe eines Tools, welches das Laden der Metadaten von „Sourceforge-Projekten“ erleichert, können automatisch den Programmdaten zugeordnet werden.

4.3 Erfassen der Metadaten

Die zu den Projekten gehörenden Metadaten werden benötigt, um die Hypothesen und Fragestellungen beantworten zu können.
Zum Teil können die Informationen direkt von www.sourceforge.net heruntergeladen werden. Dieses geschieht mit Hilfe eines Programmes, das im Rahmen einer Bachelor-Thesis geschrieben worden ist [Yue08] und die Metadaten direkt von www.sourceforge.net herunterlädt, wofür der oben erwähnte Unixname benötigt wird.
Die Metadaten der nicht von www.sourceforge.net stammenden Programme werden manuell in der Datenbank ergänzt.

4.4 Vermessen der Software und Export der Rohdaten

Nachdem der Sourcecode der Projekte heruntergeladen und zum kompilieren gebracht worden ist, kann das Express Multi DB Export Tool des VizzAnalyzers auf diese angewendet werden.
Nacheinander wird angegeben:

- welche Projekte analysiert werden sollen [5],

- wohin die .log und die .gml Dateien gespeichert werden sollen,

- die Verwendung des Treibers (sun.jdbc.odbc.JdbcOdbcDrivers),

[4] Im Namen darf nicht das $-Zeichen enthalten sein, da dieses als Trennzeichen im VizzAnalyzer-Tool genutzt wird.
[5] Es steht eine beschränkte Menge an Arbeitsspeicher zur Verfügung, was als Folge hat, dass große Programme einzeln analysiert werden sollten.

- der Name der Datenbank, in welche die Ergebnisse exportiert werden

  und

- der Username und das Passwort.

Für jedes Projekt werden 27 Metriken berechnet und in die Datenbank exportiert, wobei für die Auswertungen später nicht alle dieser Metriken gebraucht werden, weshalb auf die nicht verwendeten in dieser Arbeit nicht näher eingegangen wird. (Kap. 2.2 und A.1)
Die weiteren Verarbeitungen und Analysen der Exportdaten sind für jede Hypothese individuell und werden im Kapitel 5 näher erläutert.

5 Auswertung der Hypothesen

Dieses Kapitel befasst sich mit der Prüfung der im Kapitel 3 genannten Hypothesen, der Beantwortung der Fragestellungen und mit den in Kapitel 4 gesammelten Daten. Es wird auf jede Hypothese einzeln eingegangen und die Aufbereitung der Daten, die Darlegung der Ergebnisse und die anschließende Analyse dieser Ergebnisse beschrieben.

5.1 Hypothese 1: Schwellenwerte

Die erste Hypothese befasst sich mit der Frage nach angemessenen und sinnvoll gewählten Schwellenwerten für Metriken und ob es sinnvoll ist, einen absoluten Wert über eine große Anzahl von Projekten zu bestimmen.
Benötigt werden diese Schwellenwerte, um eventuelle Ausreißer erkennen und eleminieren zu könnnen.
Die bisher genutzten Schwellenwerte von willkürlich gewählten 15% der Werteskala haben den Nachteil, dass nicht bekannt ist, wie viele Werte in dem ausgeschlossenen Wertebereich liegen werden. Daher kann es vorkommen, dass 90 % oder mehr der Werte als Ausreißer angesehen werden.

H_0: 15 Prozent der Werteskala ist der optimale Schwellenwert.

H_1: Mindestens für eine Metrik existieren optimalere Schwellenwerte.

Es werden drei Schwellenwerte gewählt um abschätzen zu können, welcher Schwellenwert Ausreißer am ehesten eleminiert und um zu beobachten, welchen Einfluss die Größe des Bereichs auf die Anzahl der Ausreisser hat. Diese neuen Schwellenwerte liegen bei jeweils 10, 15 und 20 % der Werte. Es bleiben also jeweils 80, 70 und 60 % aller Werte zwischen den Schwellenwerten erhalten. Dieses gilt für jede einzelne Metrik. Genutzt werden alle gesammelten Projekte und ihre Daten.
Die neu berechneten Schwellenwerte und der bisher genutzte Wert werden anhand von Normalverteilungszahlen verglichen. Hierzu gehört die Messung der Schiefe, des Mittelwertes, des Modalwertes und des Modus.
H_0 ist dann widerlegt, wenn bei mindestens einer Metrik der Schwellenwert von 15 Prozent der Werteskala nicht das beste Ergebnis im Vergleich mit den anderen drei Schwellen

darstellt. Dieses ist der Fall, wenn z.B. sehr wenig Werte im angegebenen Wertebereich verbleiben, da über 50% als Ausreißer angesehen werden würden.
Im folgenden werden die Datenaufbereitung, die erlangten Ergebnisse und die Analyse der Ergebnisse für die erste Hypothese beschrieben. Ermittlet werden die Schwellenwerte für alle Klassen- und Package-Metriken [6].

5.1.1 Datenaufbereitung

Um die benötigten Berechnungen machen zu können, werden mit Hilfe des Excelabfrageassistenten und SQL-Abfragen die Datensätze aus der Datenbank ausgelesen. Zu diesen Daten gehören die Metrikwerte der gesammelten Projekte, jeweils betrachtet für jede einzelne Metrik.
Über diese Daten wird anschließend eine Pivottabelle erstellt, welches eine einfache Möglichkeit ist, an die Häufigkeit jedes einzelnen Wertes zu kommen.
Berechnet werden nacheinander und zum Teil mit Hilfe selbst erstellter Makros:

- der Median,
- der Modus,
- der Mittelwert,
- die kumulierte Häufigkeit, um zu zeigen, wieviele der Werte in einem bestimmten Wertebereich liegen,
- die absoluten Werte für 10, 15 und 20 % (jeweils der obere und untere Schwellenwert),
- die Grenzen für 15 Prozent der Werteskala

  und

- die Schiefe der Verteilung mit Beachtung der Schwellenwerte

Sowohl der Vergleich des Median mit dem Modus und dem Mittelwert, als auch die Betrachtung der Schiefe und der Verteilungskurven lassen einen Schluss dahingehend zu, ob es sich um Normalverteilungen handelt. (Kap. 2.6.1) Dieses ist der Fall, wenn der Mittelwert, der Median und der Modus übereinstimmen und die Schiefe 0 ist.

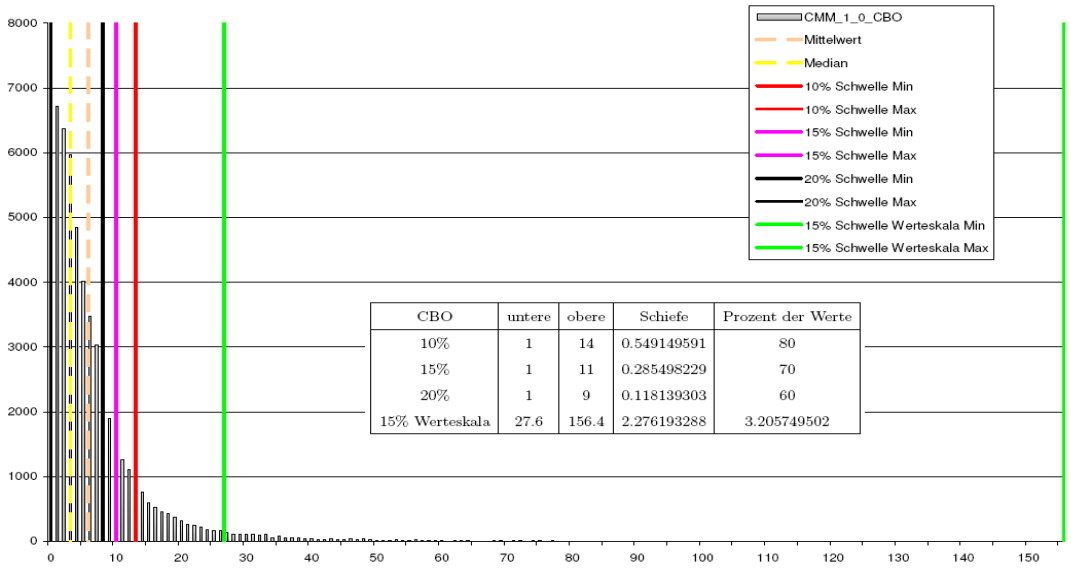
[6] Eine Liste aller Metriken ist im Anhang in den Tabellen 9, 10 und 11 zu finden

Zur Veranschaulichung der Daten werden diese zusätzlich als Verteilungskurven dargestellt.

5.1.2 Ergebnisse

Dieses Unterkapitel zeigt für jede Metrik die Verteilungskurve in den Abbildungen 4-29, sowie eine Interpretation der beobachteten Verteilungen der Metrikwerte.
Die Abbildungen bzw. Diagramme sind wie folgt aufgebaut:

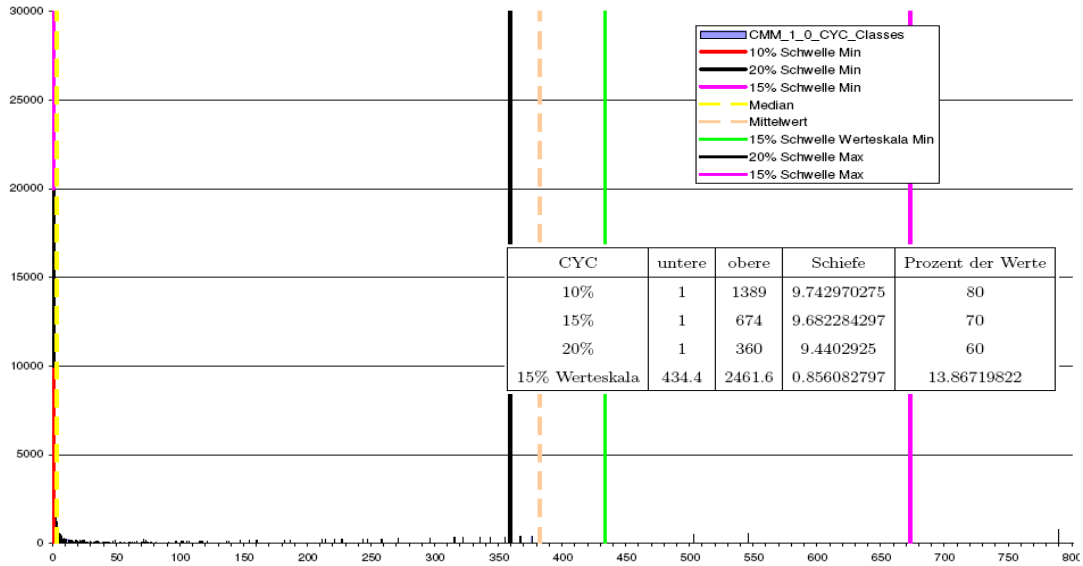
- Die x-Achse zeigt die Werteskala der Metrik und deren Ausdehnung (soweit nicht anders benannt).
- Die y-Achse stellt die Häufigkeit dieser einzelnen Werte dar.
- Aufgezeigt sind der Mittelwert und der Median (gestrichelt),
- sowie die unteren und oberen Schwellenwerte der vier betrachteten Varianten.
- Zur Verdeutlichung der Schwellen und der Art der Verteilung jeder einzelnen Kurve sind ebenfalls die Schwellenwerte und die Schiefe mittels einer Tabelle mit angezeigt,
- ebenso der prozentuale Anteil der zwischen den Schwellen liegenden Werte im Bezug auf alle aus der Analyse erhaltenen Werte.



CBO	untere	obere	Schiefe	Prozent der Werte
10%	1	14	0.549149591	80
15%	1	11	0.285498229	70
20%	1	9	0.118139303	60
15% Werteskala	27.6	156.4	2.276193288	3.205749502

Abbildung 4: CBO Schwellen

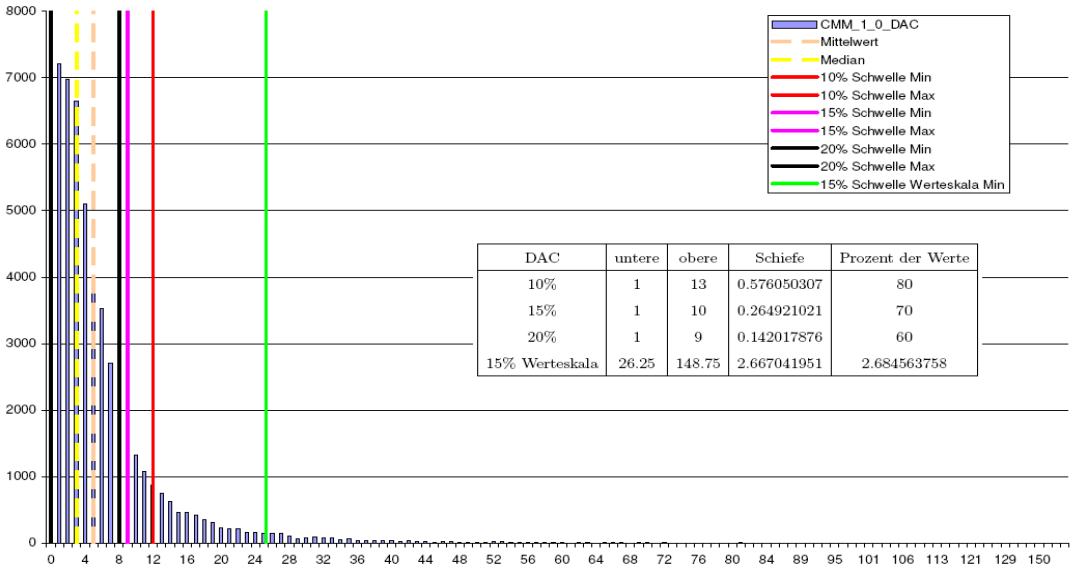
Abbildung 4 zeigt die Schwellenwerte für die CBO Metrik (Anzahl der Klassen die mit einer Klasse in Verbindung stehen). Die gesammelten Messwerte sind nicht normalverteilt, sondern weisen eine Rechtsschiefe auf. Die 15% Schwelle der Werteskala umfasst nur ca. 3% aller Werte. Die 20% Schwelle hat die geringste Schiefe mit 0.12.



CYC	untere	obere	Schiefe	Prozent der Werte
10%	1	1389	9.742970275	80
15%	1	674	9.682284297	70
20%	1	360	9.4402925	60
15% Werteskala	434.4	2461.6	0.856082797	13.86719822

Abbildung 5: CYC Schwellen

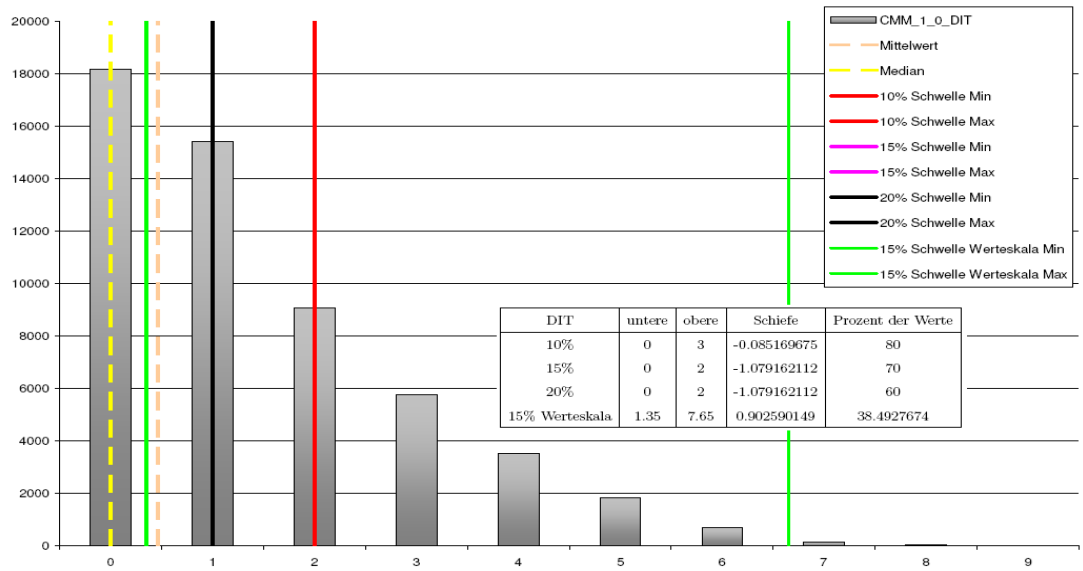
Abbildung 5 zeigt die Schwellenwerte für die CYC Metrik (Anzahl der Klassen eines Systems, die sich mit einer Klasse in einer zyklischen Beziehung befinden). Die Verteilung ist rechtsschief und die Werte liegen weit auseinander, wodurch nicht alle abgebildet sind. Zwischen den 15% Schwellen der Werteskala liegen nur ca. 13.8% aller Werte. Mittelwert und Median weichen stark von einander ab.



DAC	untere	obere	Schiefe	Prozent der Werte
10%	1	13	0.576050307	80
15%	1	10	0.264921021	70
20%	1	9	0.142017876	60
15% Werteskala	26.25	148.75	2.667041951	2.684563758

Abbildung 6: DAC Schwellen

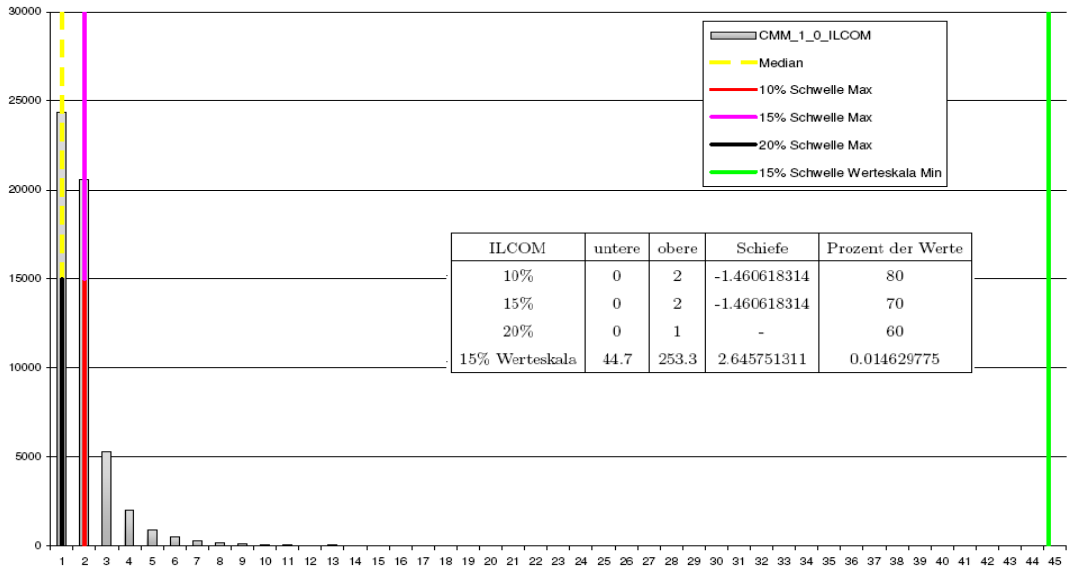
Abbildung 6 zeigt die Schwellenwerte für die DAC Metrik (Anzahl der Verbindungen, welche durch abstrakte Datentypen verursacht werden). Es ist eine Rechtsschiefe zu erkennen. Die 15% Schwelle der Werteskala umfasst hier keine 3%. Der Wertebereich ist langgestreckt, daher können aufgrund der Anschaulichkeit viele Werte und die Max-Schwelle der 15% Schwelle der Werteskala nicht abgebildet werden.



DIT	untere	obere	Schiefe	Prozent der Werte
10%	0	3	-0.085169675	80
15%	0	2	-1.079162112	70
20%	0	2	-1.079162112	60
15% Werteskala	1.35	7.65	0.902590149	38.4927674

Abbildung 7: DIT Schwellen

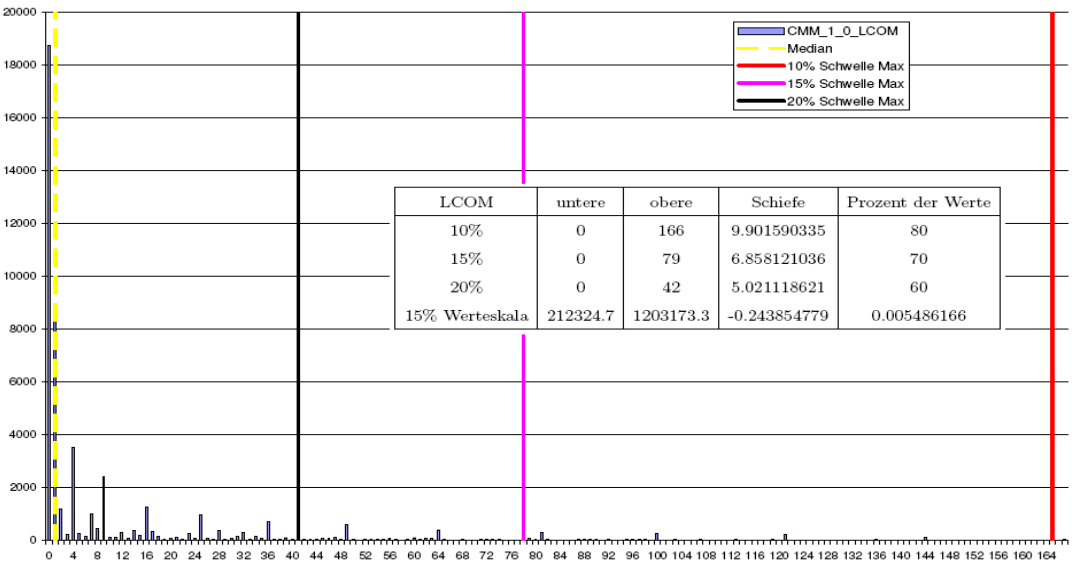
Abbildung 7 zeigt die Schwellenwerte für die DIT Metrik (Tiefe von Klassen oder Interfaces in der Vererbungshierarchie). Die Verteilung ist rechtsschief und zeigt bei der 10% Schwelle die geringste Schiefe.



ILCOM	untere	obere	Schiefe	Prozent der Werte
10%	0	2	-1.460618314	80
15%	0	2	-1.460618314	70
20%	0	1	-	60
15% Werteskala	44.7	253.3	2.645751311	0.014629775

Abbildung 8: ILCOM Schwellen

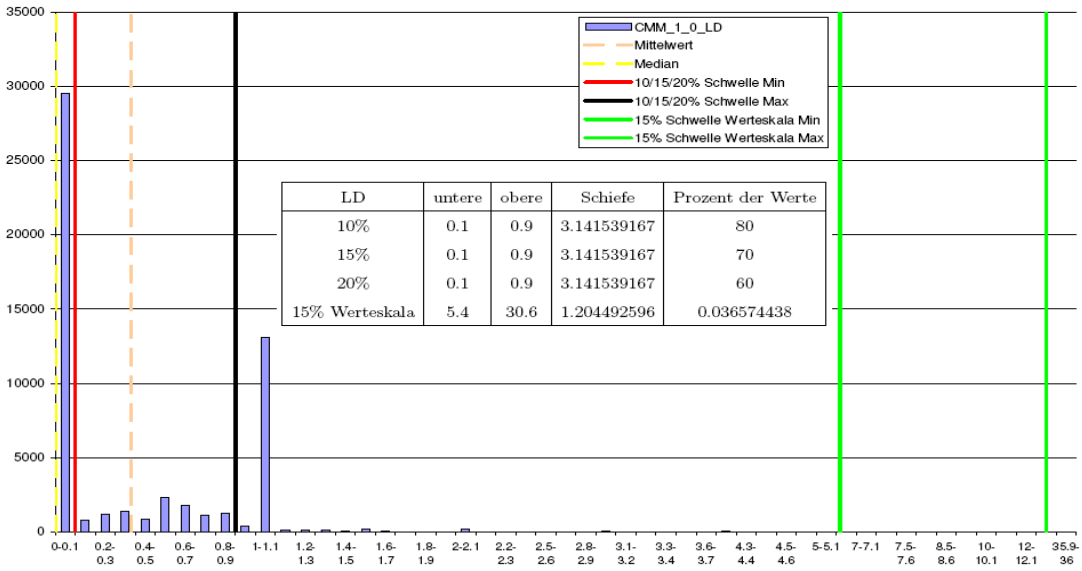
Abbildung 8 zeigt die Schwellenwerte für die ILCOM Metrik (Anzahl der verbundenen Methoden einer Klasse, die sich Instanzvariablen einer Klasse teilen). Die Werte bilden eine rechtschiefe Verteilung und sind stark auf die kleinsten Werte verteilt. Der Wertebereich ist stark ausgedehnt, daher können aufgrund der Anschaulichkeit nicht alle Werte abgebildet werden, ebenso wie die 15% Schwelle der Werteskala.



LCOM	untere	obere	Schiefe	Prozent der Werte
10%	0	166	9.901590335	80
15%	0	79	6.858121036	70
20%	0	42	5.021118621	60
15% Werteskala	212324.7	1203173.3	-0.243854779	0.005486166

Abbildung 9: LCOM Schwellen

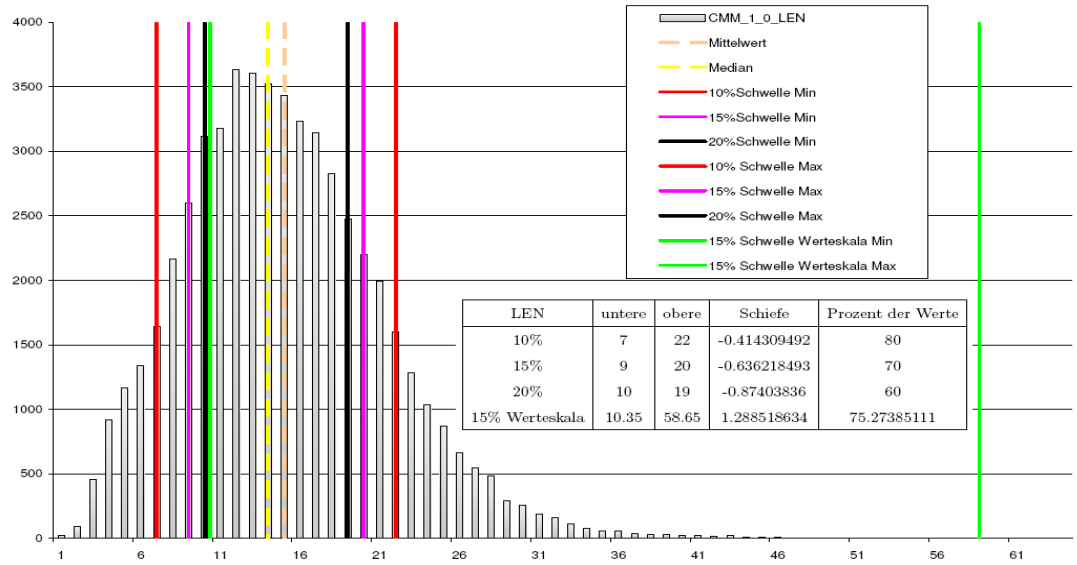
Die Abbildung 9 zeigt die Schwellenwerte der LCOM Metrik (Anzahl der nicht verbundenen Methodenpaare einer Klasse). Der Bereich der 15% Schwelle der Werteskala liegt in den hohen Zahlen und umfasst nur ca. 0.005%. Da sich der Wertebereich bis 1415498 erstreckt, ist dieser nicht vollständig abgebildet, wobei auch der Mittelwert (210.1094673) und die 15% Schwelle der Werteskala aus dem Sichtfeld entrücken.



LD	untere	obere	Schiefe	Prozent der Werte
10%	0.1	0.9	3.141539167	80
15%	0.1	0.9	3.141539167	70
20%	0.1	0.9	3.141539167	60
15% Werteskala	5.4	30.6	1.204492596	0.036574438

Abbildung 10: LD Schwellen

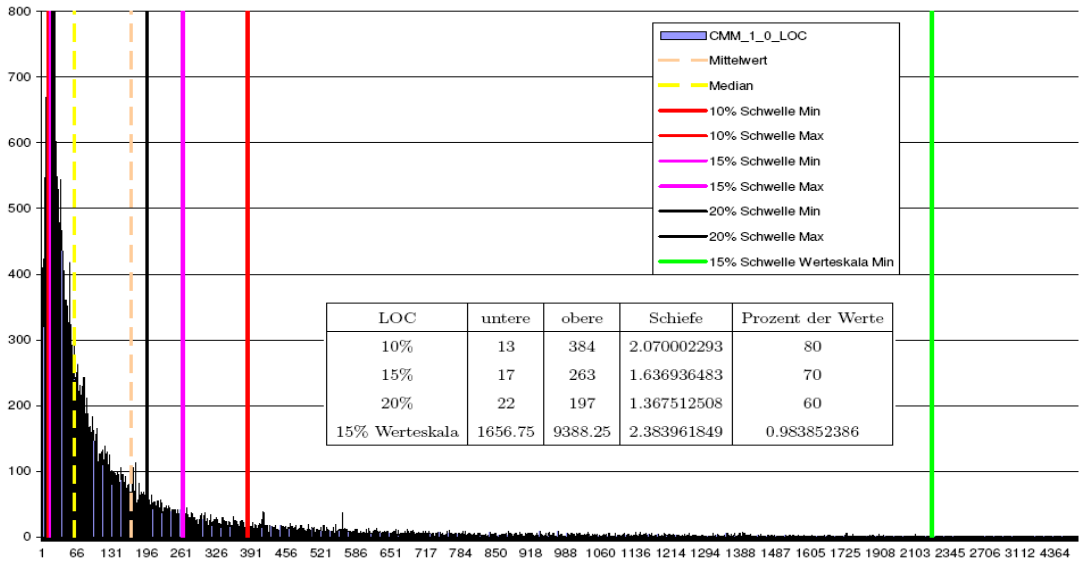
Die Abbildung 10 zeigt die Schwellenwerte der LD Metrik (Menge der Daten, die direkt einer Klasse angehören, im Vergleich zu der Menge von Daten, die von dieser Klasse verwendet werden).Der Bereich der 15% Schwelle der Werteskala umfasst nur ca. 0.03%. Der größte Teil der Werte liegt im unteren Wertebereich bei 0 und 1.



LEN	untere	obere	Schiefe	Prozent der Werte
10%	7	22	-0.414309492	80
15%	9	20	-0.636218493	70
20%	10	19	-0.87403836	60
15% Werteskala	10.35	58.65	1.288518634	75.27385111

Abbildung 11: LEN Schwellen

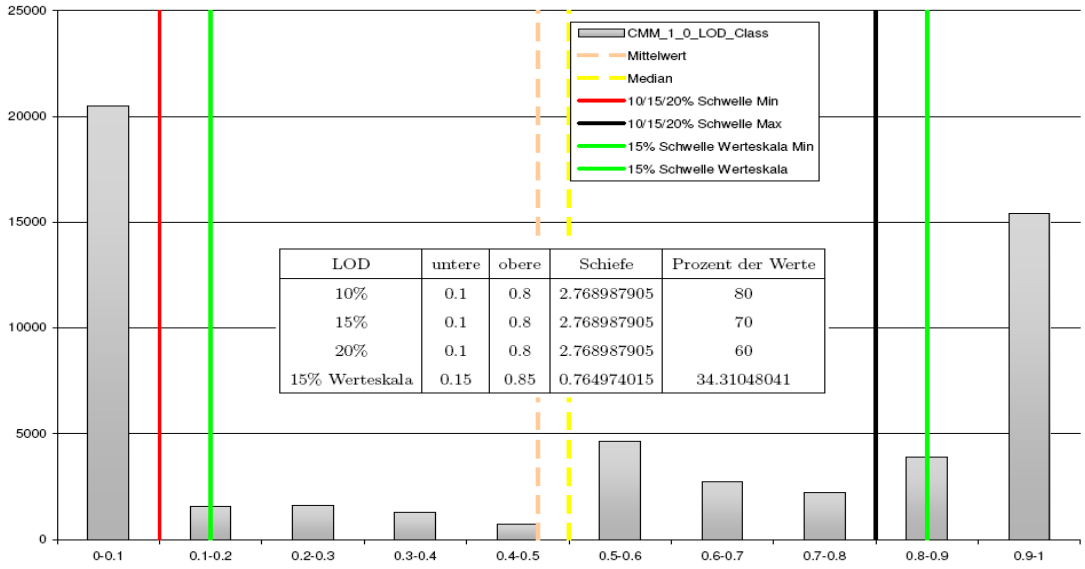
Die Abbildung 11 zeigt die Schwellenwerte der LEN Metrik (Anzahl der Buchstaben in Klassen-, Package-, Methoden- und Feld-Namen). Diese Verteilung kommt einer Normalverteilung recht nahe, weist aber eine leichte Schiefe auf. Die Schiefe ist bei der 10% Schwelle am geringsten, bei welcher auch die meisten Werte mit berücksichtigt werden.



LOC	untere	obere	Schiefe	Prozent der Werte
10%	13	384	2.070002293	80
15%	17	263	1.636936483	70
20%	22	197	1.367512508	60
15% Werteskala	1656.75	9388.25	2.383961849	0.983852386

Abbildung 12: LOC Schwellen

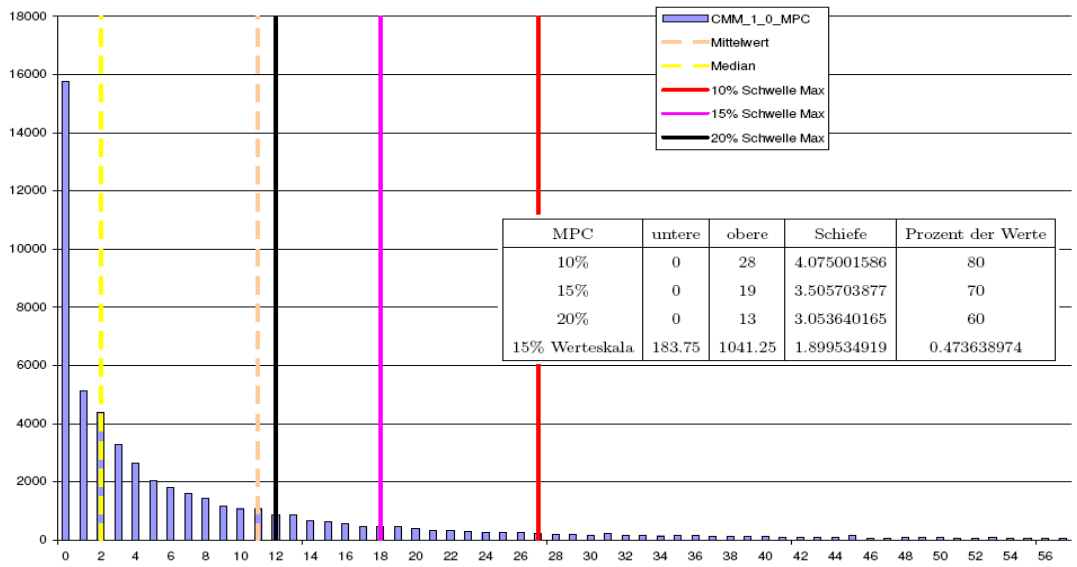
Die Abbildung 12 zeigt die Schwellenwerte der LOC Metrik (Anzahl der Codezeilen). Die Verteilung weist eine Rechtsschiefe auf und erstreckt sich über einen Wertebereich von 1 bis 11045. Zur besseren Veranschaulichung ist dieser nicht komplett abgebildet, womit auch die 15% Schwelle der Werteskala verschwindet.



LOD	untere	obere	Schiefe	Prozent der Werte
10%	0.1	0.8	2.768987905	80
15%	0.1	0.8	2.768987905	70
20%	0.1	0.8	2.768987905	60
15% Werteskala	0.15	0.85	0.764974015	34.31048041

Abbildung 13: LOD Schwellen

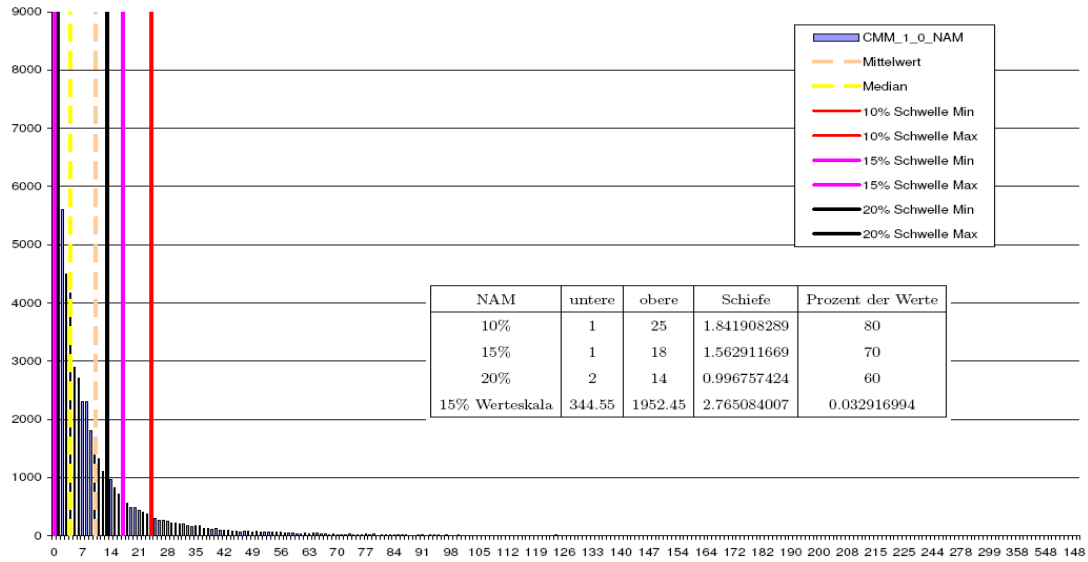
Die Abbildung 13 zeigt die Schwellenwerte der LOD Metrik (Anzahl der fehlenden Kommentare). Die Verteilung weist bei 0 und 1 zwei Spitzen auf.



MPC	untere	obere	Schiefe	Prozent der Werte
10%	0	28	4.075001586	80
15%	0	19	3.505703877	70
20%	0	13	3.053640165	60
15% Werteskala	183.75	1041.25	1.899534919	0.473638974

Abbildung 14: MPC Schwellen

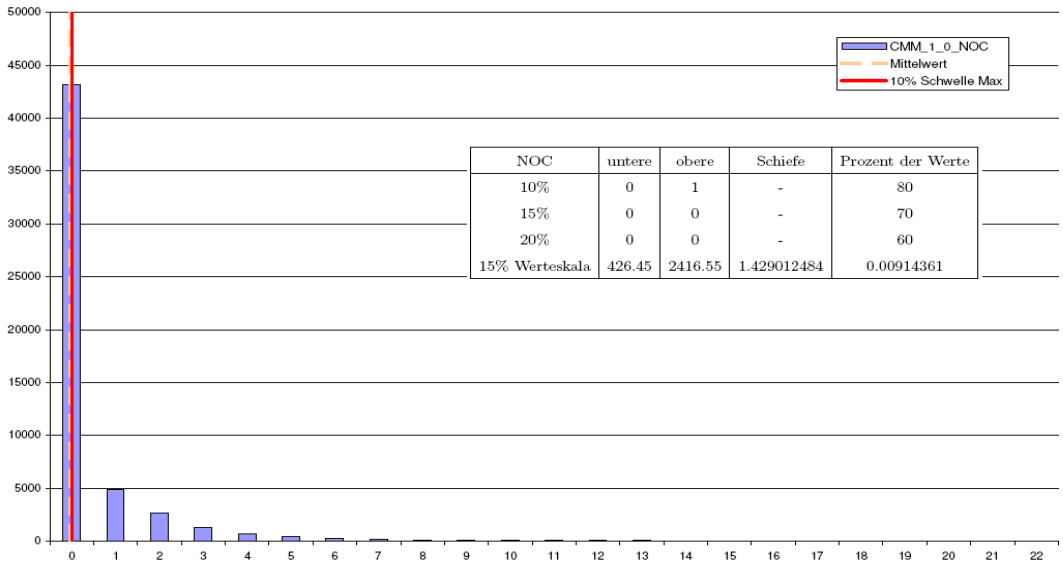
Die Abbildung 14 zeigt die Schwellenwerte der MPC Metrik (Anzahl der Aufrufe, die in Methoden einer Klasse festgelegt sind). Die Verteilungskurve ist rechtsschief und wird nicht komplett abgebildet, wodurch die 15% Schwelle der Werteskala nicht mehr zu sehen ist.



NAM	untere	obere	Schiefe	Prozent der Werte
10%	1	25	1.841908289	80
15%	1	18	1.562911669	70
20%	2	14	0.996757424	60
15% Werteskala	344.55	1952.45	2.765084007	0.032916994

Abbildung 15: NAM Schwellen

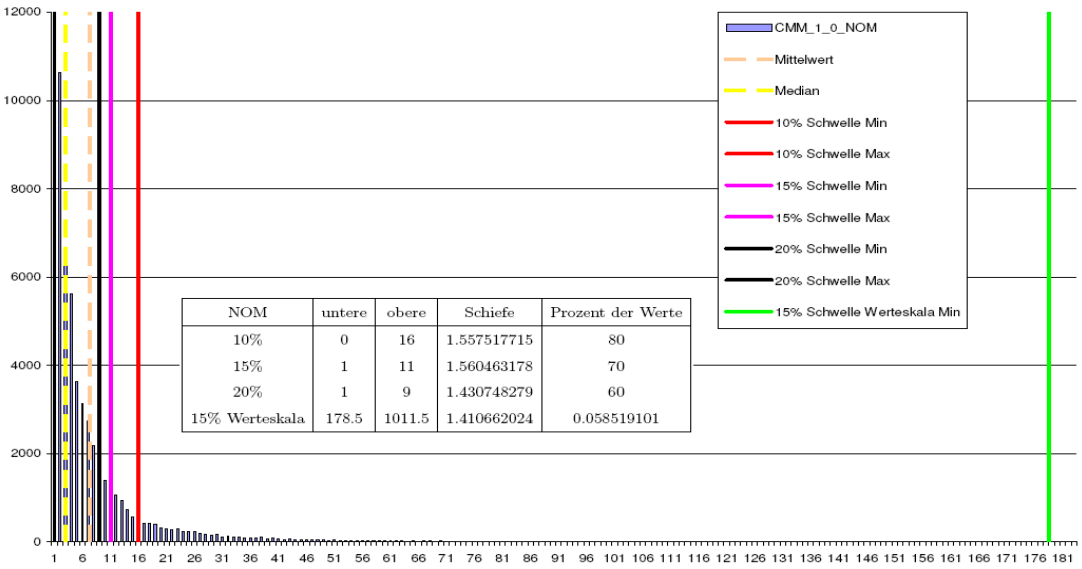
Die Abbildung 15 zeigt die Schwellenwerte der NAM Metrik (Anzahl der Attribute (Felder) und Methoden, die in einer Klasse definiert sind). Die Verteilung ist rechtsschief und hat einen Wertebereich bis 2297 aufzuweisen, welcher nicht vollständig abgebildet ist, was auch für die 15% Schwellen der Werteskala zu trifft.



NOC	untere	obere	Schiefe	Prozent der Werte
10%	0	1	-	80
15%	0	0	-	70
20%	0	0	-	60
15% Werteskala	426.45	2416.55	1.429012484	0.00914361

Abbildung 16: NOC Schwellen

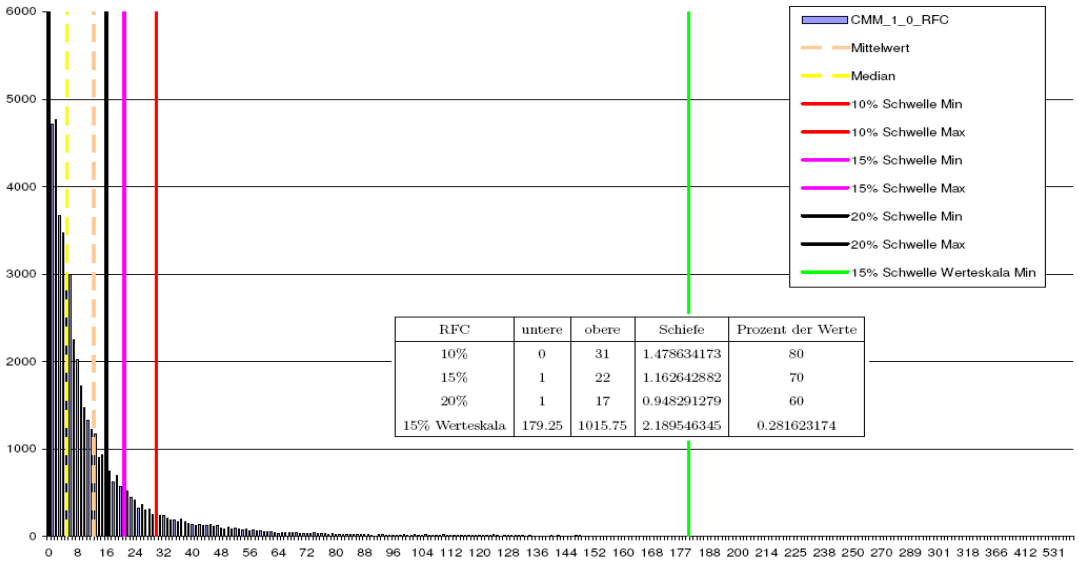
Die Abbildung 16 zeigt die Schwellenwerte der NOC Metrik (Anzahl der Klassen, welche direkt von Methoden einer Super-Klasse erben). Die Werte liegen hauptsächlich bei 0, was zeigt, dass ca. 80% der Klassen keine Kinder haben. Die Ermittlung der Schiefe und das Darstellen der Schwellen ist durch die Werteverteilung nicht vollständig möglich ist. Auch ist nicht der gesamte Wertebereich zu sehen.



NOM	untere	obere	Schiefe	Prozent der Werte
10%	0	16	1.557517715	80
15%	1	11	1.560463178	70
20%	1	9	1.430748279	60
15% Werteskala	178.5	1011.5	1.410662024	0.058519101

Abbildung 17: NOM Schwellen

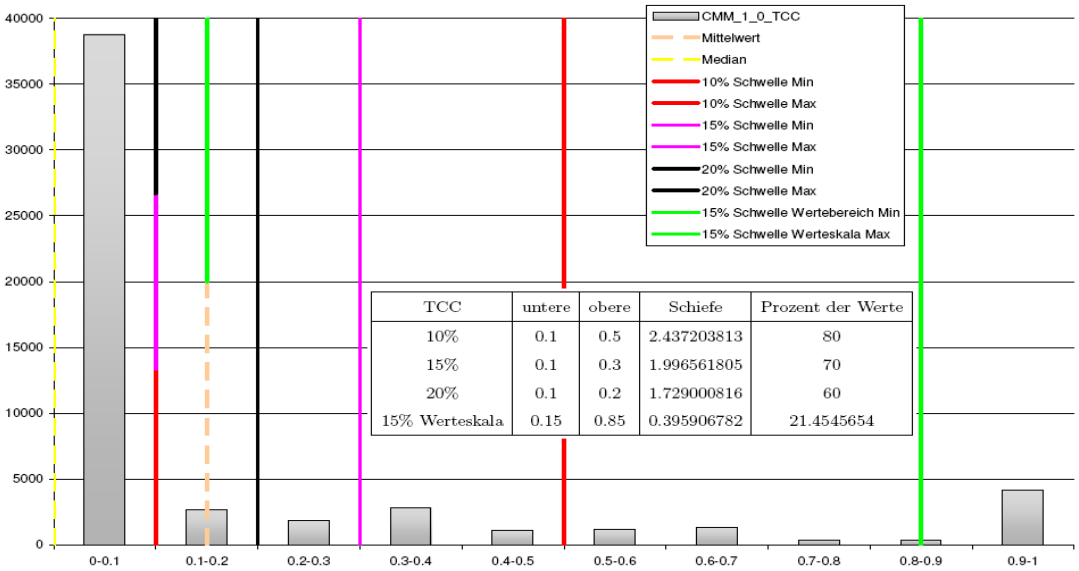
Die Abbildung 17 zeigt die Schwellenwerte der NOM Metrik (Anzahl der Methoden, die in einer Klasse lokal deklariert sind). Die Verteilung ist rechtsschief. Der Wertebereich und die 15% Schwellen der Werteskala sind nicht vollständig abgebildet.



RFC	untere	obere	Schiefe	Prozent der Werte
10%	0	31	1.478634173	80
15%	1	22	1.162642882	70
20%	1	17	0.948291279	60
15% Werteskala	179.25	1015.75	2.189546345	0.281623174

Abbildung 18: RFC Schwellen

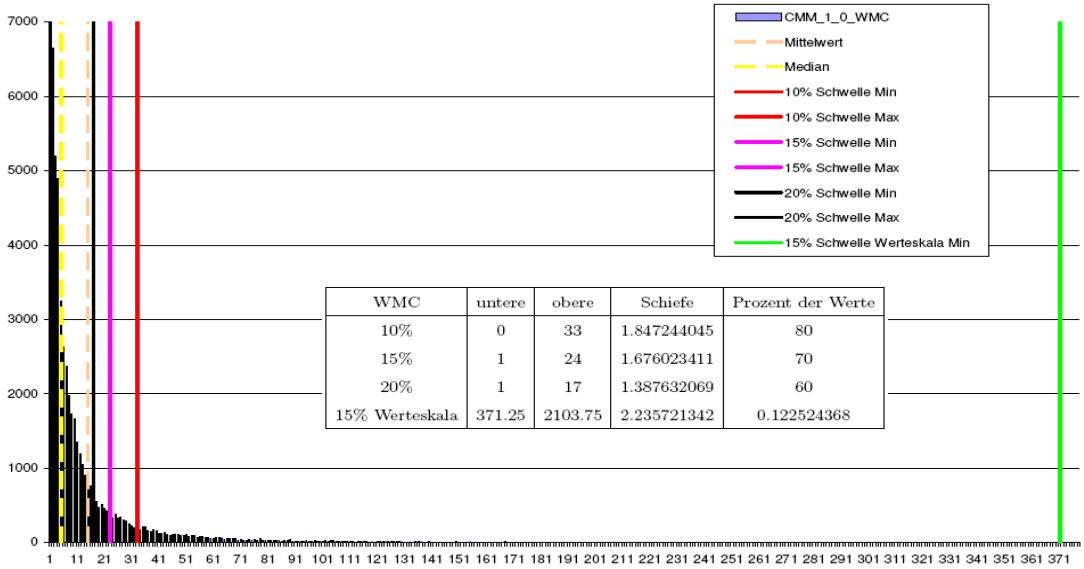
Die Abbildung 18 zeigt die Schwellenwerte der RFC Metrik (Anzahl der Methoden einer Klasse, sowie der Methoden, die von diesen aufgerufen werden). Die Verteilung ist rechtsschief. Der Wertebereich und die 15% Schwellen der Werteskala sind nicht vollständig abgebildet.



TCC	untere	obere	Schiefe	Prozent der Werte
10%	0.1	0.5	2.437203813	80
15%	0.1	0.3	1.996561805	70
20%	0.1	0.2	1.729000816	60
15% Werteskala	0.15	0.85	0.395906782	21.4545654

Abbildung 19: TCC Schwellen

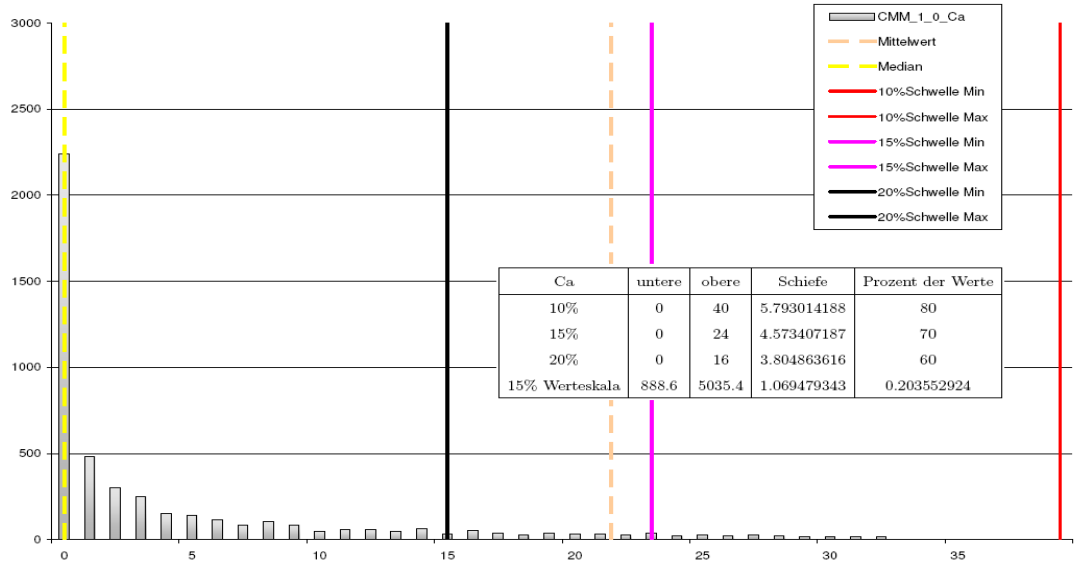
Die Abbildung 19 zeigt die Schwellenwerte der TCC Metrik (Zusammenhang zwischen den Methoden einer Klasse). Die Verteilung weist über 70% der Werte zwischen 0 und 0.1 auf.



WMC	untere	obere	Schiefe	Prozent der Werte
10%	0	33	1.847244045	80
15%	1	24	1.676023411	70
20%	1	17	1.387632069	60
15% Werteskala	371.25	2103.75	2.235721342	0.122524368

Abbildung 20: WMC Schwellen

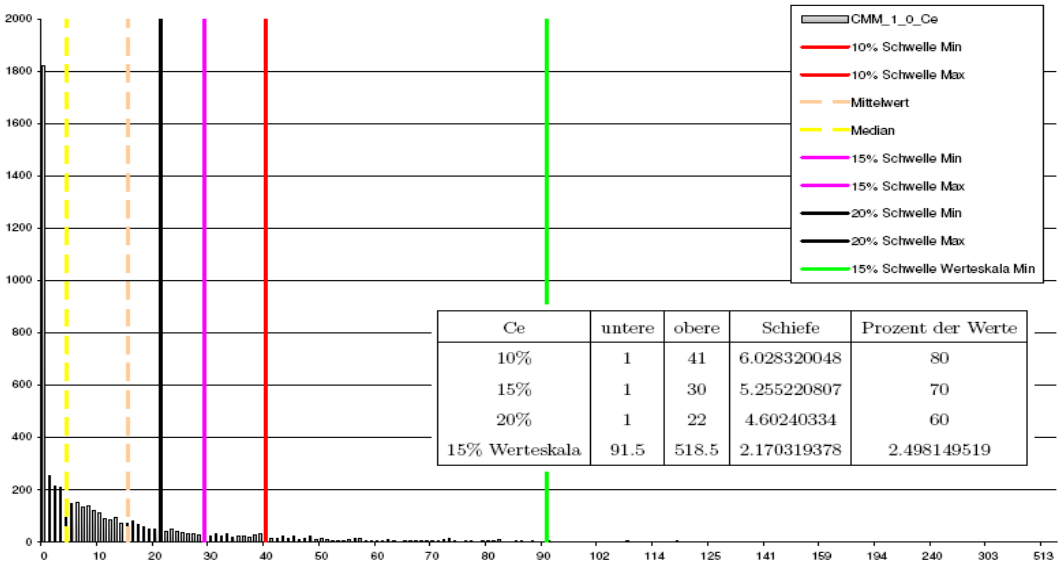
Die Abbildung 20 zeigt die Schwellenwerte der WMC Metrik (gewichtete Summe der Methoden einer Klasse). Die Verteilung ist rechtsschief und hat einen weit gefächerten Wertebereich, welcher nicht komplett dargestellt ist, was zur Folge hat, dass die 15% Schwelle der Werteskala nicht abgebildet ist.



Ca	untere	obere	Schiefe	Prozent der Werte
10%	0	40	5.793014188	80
15%	0	24	4.573407187	70
20%	0	16	3.804863616	60
15% Werteskala	888.6	5035.4	1.069479343	0.203552924

Abbildung 21: Ca Schwellen

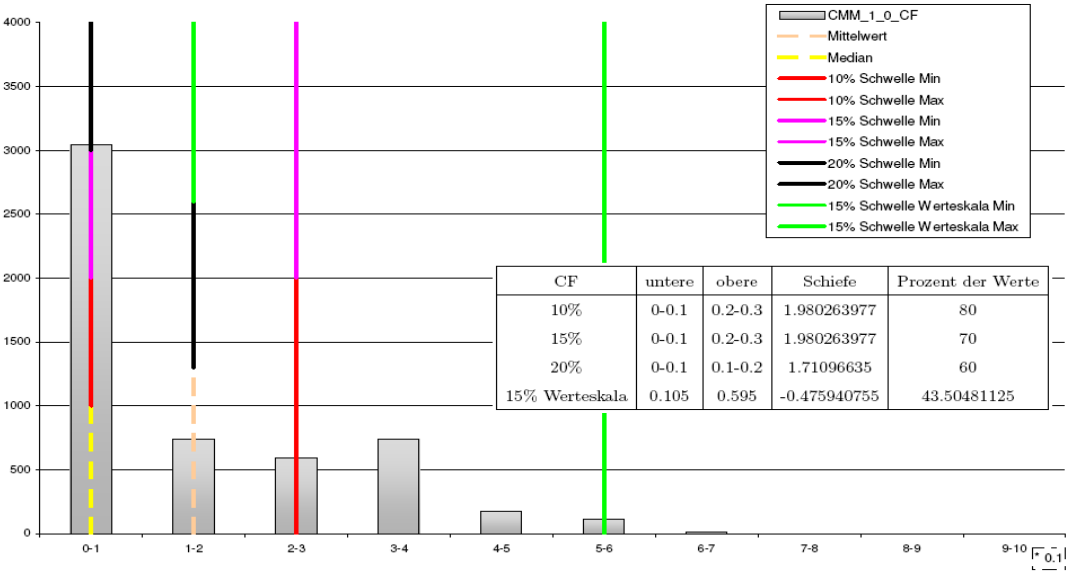
Die Abbildung 21 zeigt die Schwellenwerte der Ca Metrik (Anzahl der externen Klassen, welche mit den Klassen eines Packages verbunden sind). Der Wertebereich dieser Verteilung erstreckt sich bis 5924, welcher nicht ganz abgebildet ist und in welchem die Werte der 15% Schwelle der Werteskala liegen.



Ce	untere	obere	Schiefe	Prozent der Werte
10%	1	41	6.028320048	80
15%	1	30	5.255220807	70
20%	1	22	4.60240334	60
15% Werteskala	91.5	518.5	2.170319378	2.498149519

Abbildung 22: Ce Schwellen

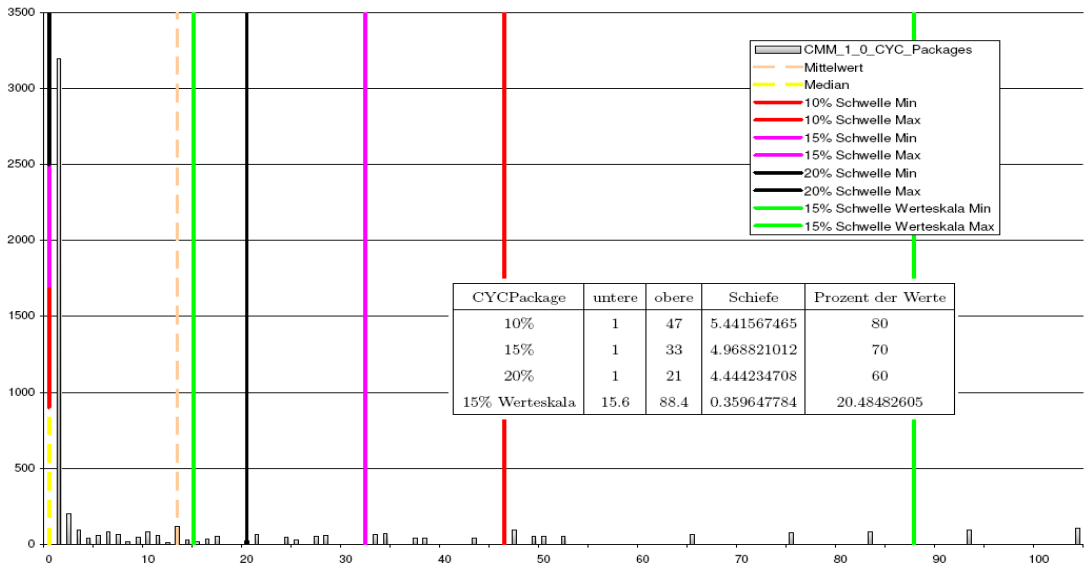
Die Abbildung 22 zeigt die Schwellenwerte der Ce Metrik (Anzahl der externen Klassen, welche mit den Klassen eines Packages verbunden sind). Der Wertebereich erstreckt sich bis 610 und ist nicht komplett abgebildet. Die 15% Schwelle der Werteskala ist ebenfalls nicht vollständig dargestellt.



CF	untere	obere	Schiefe	Prozent der Werte
10%	0-0.1	0.2-0.3	1.980263977	80
15%	0-0.1	0.2-0.3	1.980263977	70
20%	0-0.1	0.1-0.2	1.71096635	60
15% Werteskala	0.105	0.595	-0.475940755	43.50481125

Abbildung 23: CF Schwellen

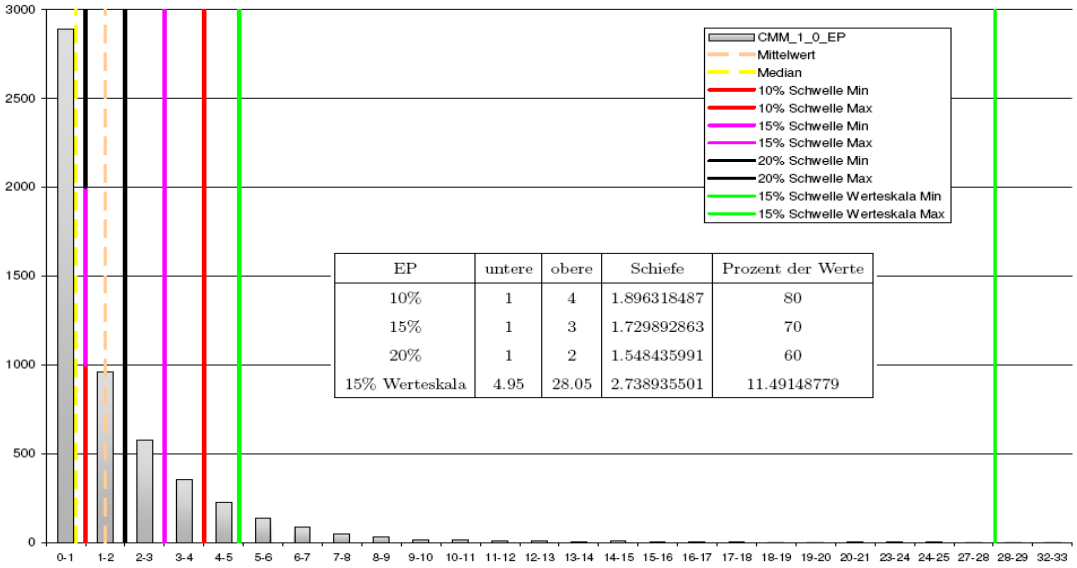
Die Abbildung 23 zeigt die Schwellenwerte der CF Metrik (Verbindungen zwischen Klassen ohne die Verbindungen zu beachten, welche auf Vererbung beruhen). Die Verteilung zeigt eine deutliche Spitze bei 0-0.1.



CYCPackage	untere	obere	Schiefe	Prozent der Werte
10%	1	47	5.441567465	80
15%	1	33	4.968821012	70
20%	1	21	4.444234708	60
15% Werteskala	15.6	88.4	0.359647784	20.48482605

Abbildung 24: CYCPackage Schwellen

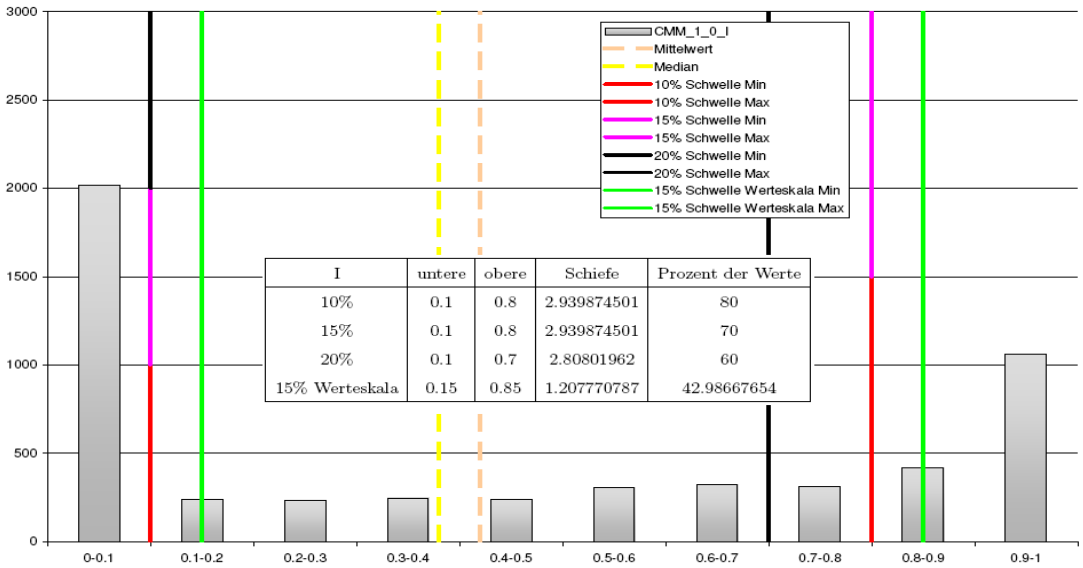
Die Abbildung 24 zeigt die Schwellenwerte der CYCPackage Metrik (Anzahl der Packages eines Systems, die sich mit einem Packages in einer zyklischen Beziehung befinden). Die Verteilung weist eine stark ausgebildete Spitze auf und der Wertebereich ist aufgrund seiner Streuung nicht vollständig abgebildet.



EP	untere	obere	Schiefe	Prozent der Werte
10%	1	4	1.896318487	80
15%	1	3	1.729892863	70
20%	1	2	1.548435991	60
15% Werteskala	4.95	28.05	2.738935501	11.49148779

Abbildung 25: EP Schwellen

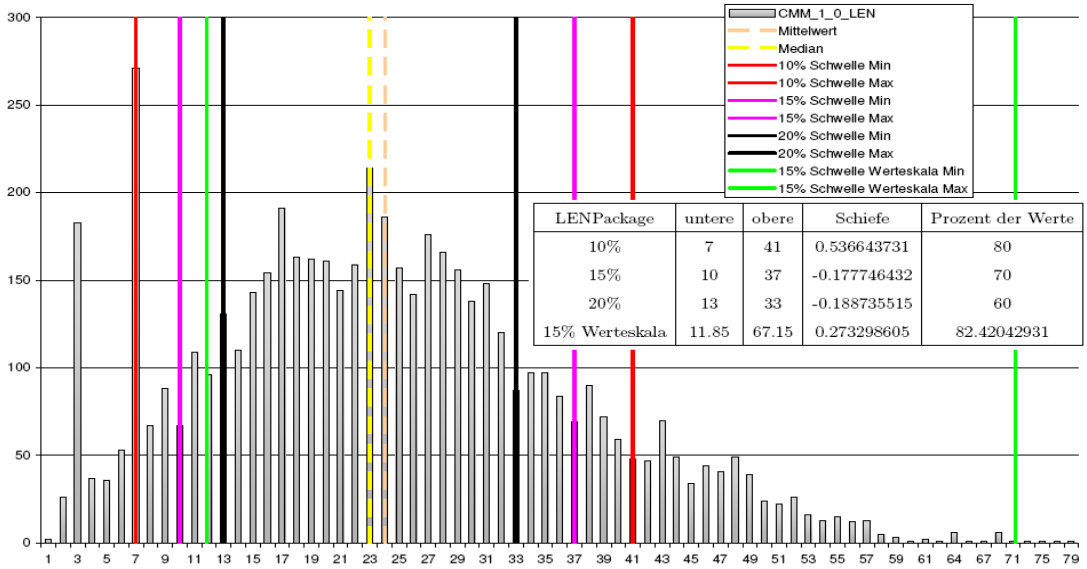
Die Abbildung 25 zeigt die Schwellenwerte der EP Metrik (Klassen außerhalb eines Packages, die von den Klassen in dem Package verwendet werden (efferent coupling, Ce) im Verhältnis zu der Anzahl der Klassen n in dem Package (Ce/n)). Die Verteilung ist rechtsschief und zwischen 0 und 10 liegen mehr als 98% der Werte.



I	untere	obere	Schiefe	Prozent der Werte
10%	0.1	0.8	2.939874501	80
15%	0.1	0.8	2.939874501	70
20%	0.1	0.7	2.80801962	60
15% Werteskala	0.15	0.85	1.207770787	42.98667654

Abbildung 26: I Schwellen

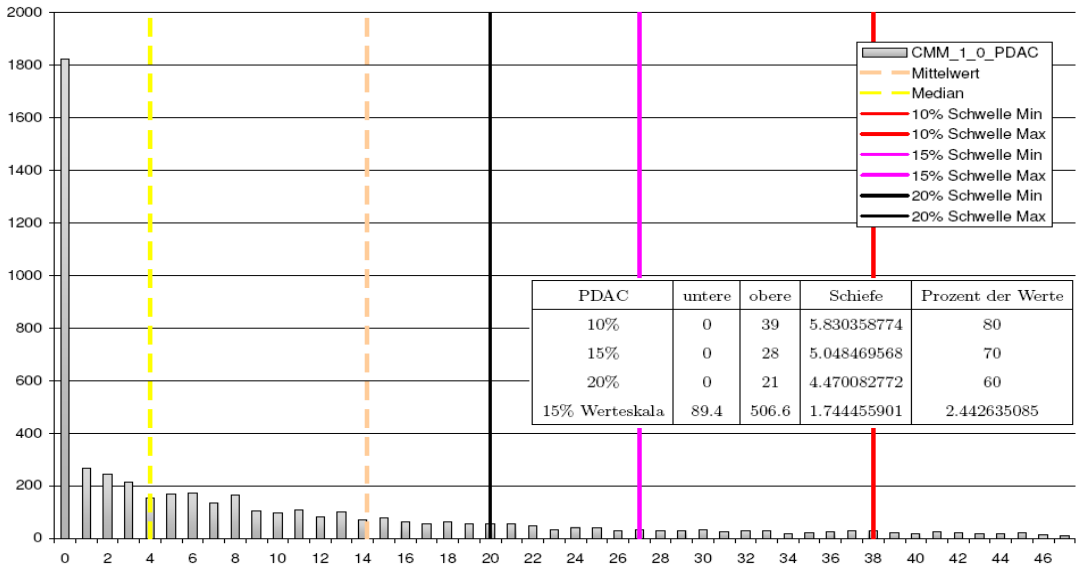
Die Abbildung 26 zeigt die Schwellenwerte der I Metrik (Verhältnis von hinausgehenden und hinaus- und hineingehenden Verbindungen von Klassen innerhalb eines Package mit Klassen anderer Packages). Die meisten Werte dieser Verteilung liegen zwischen 0 und 1.



LENPackage	untere	obere	Schiefe	Prozent der Werte
10%	7	41	0.536643731	80
15%	10	37	-0.177746432	70
20%	13	33	-0.188735515	60
15% Werteskala	11.85	67.15	0.273298605	82.42042931

Abbildung 27: LENPackage Schwellen

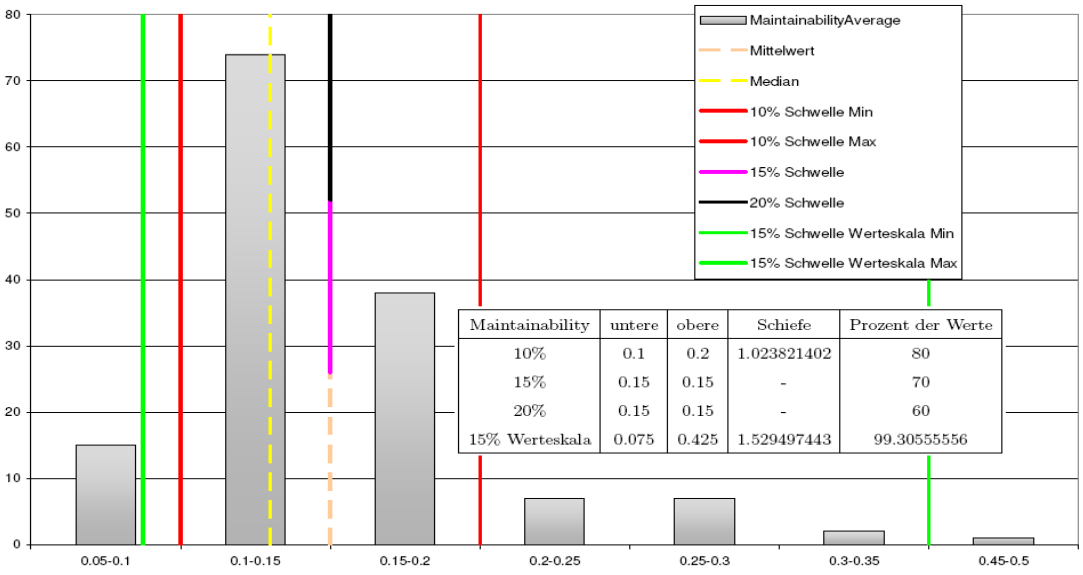
Die Abbildung 27 zeigt die Schwellenwerte der LENPackage Metrik (Anzahl der Buchstaben in Klassen-, Package-, Methoden- und Feld-Namen). Die Verteilung kommt einer Normalverteilung recht nahe. Die Schiefe ist bei der 15% Schwelle am geringsten.



PDAC	untere	obere	Schiefe	Prozent der Werte
10%	0	39	5.830358774	80
15%	0	28	5.048469568	70
20%	0	21	4.470082772	60
15% Werteskala	89.4	506.6	1.744455901	2.442635085

Abbildung 28: PDAC Schwellen

Die Abbildung 28 zeigt die Schwellenwerte der PDAC Metrik (durch abstrakte Datentypen auf dem Package-Level verursachte Verbindungskomplexität). Die Verteilung ist rechtsschief und fast 34% der Werte liegen bei 0. Der Bereich der Werte ist nicht vollständig abgebildet. Dieses umfasst die 15% Schwelle der Werteskala.



Maintainability	untere	obere	Schiefe	Prozent der Werte
10%	0.1	0.2	1.023821402	80
15%	0.15	0.15	-	70
20%	0.15	0.15	-	60
15% Werteskala	0.075	0.425	1.529497443	99.30555556

Abbildung 29: Maintainability Schwellen

Die Abbildung 29 zeigt die Schwellenwerte der Maintainability Metrik (Aussage über die Wartbarkeit einer Software). Ungefähr 51 % der Werte liegen bei dieser Verteilung zwischen 0.1 und 0.15.

Die Tabellen 1 und 2 zeigen in der Gesamtübersicht die absoluten Werte für die 10%, 15% und 20% Schwellen sowie für die 15% Werteskala Schwelle für alle Metriken.

Schwellen-	10%		15%		20%		15% Werteskala	
werte	unterer	oberer	unterer	oberer	unterer	oberer	unterer	oberer
CBO	1	14	1	11	1	9	27.6	156.4
CYC	1	1389	1	674	1	360	434.4	2461.6
DAC	1	13	1	10	1	9	26.25	148.75
DIT	0	3	0	2	0	2	1.35	7.65
ILCOM	0	2	0	2	0	1	44.7	253.3
LCOM	0	166	0	79	0	42	212324.7	1203173.3
LD	0.1	0.9	0.1	0.9	0.1	0.9	5.4	30.6
LEN	7	22	9	20	10	19	10.35	58.65
LOC	13	384	17	263	22	197	1656.75	9388.25
LOD	0.1	0.8	0.1	0.8	0.1	0.8	0.15	0.85
MPC	0	28	0	19	0	13	183.75	1041.25
NAM	1	25	1	18	2	14	344.55	1952.45
NOC	0	1	0	0	0	0	426.45	2416.55
NOM	0	16	1	11	1	9	178.5	1011.5
RFC	0	31	1	22	1	17	179.25	1015.75
TCC	0.1	0.5	0.1	0.3	0.1	0.2	0.15	0.85
WMC	0	33	1	23	1	17	371.25	2103.75

Tabelle 1: Schwellenwerte Klassenmetriken

Schwellen-	10%		15%		20%		15% Werteskala	
wert	unterer	oberer	unterer	oberer	unterer	oberer	untere	oberer
Ca	0	40	0	24	0	16	888.6	5035.4
Ce	0	41	0	30	0	22	91.5	518.5
CF	0.1	0.2	0.1	0.2	0.1	0.1	0.105	0.595
CYC Package	1	47	1	33	1	21	15.6	88.4
EP	1	4	1	3	1	2	4.95	28.05
I	0.1	0.8	0.1	0.8	0.1	0.7	0.15	0.85
LEN Package	7	41	10	37	13	33	11.85	67.15
PDAC	0	39	0	28	0	21	89.4	506.6
Maint.	0.1	0.2	0.15	0.15	0.15	0.15	0.075	0.425

Tabelle 2: Schwellenwerte Packagemetriken

5.1.3 Analyse

Wie aus den Tabellen 1 und 2 sowie aus den Diagrammen 4-29 hervorgeht, liegt bei keiner der betrachteten Metriken eine vollständige Normalverteilung vor, sondern diese weisen in den meisten Fällen eine Rechtsschiefe auf oder haben zwei Spitzen an den Rändern der Werteskalen. Dieses ist sowohl graphisch gut erkennbar, als auch über die Schiefe aufgezeigt.

Es wurden vier mögliche Schwellenwerte betrachtet und ergänzend die Schiefe berechnet. Bei einigen Metriken ist zu erkennen, dass die Analyseergebnisse sich auf sehr wenige Werte beschränken, so dass es nahezu unmöglich ist, ordentliche Grenzwerte zu ermitteln und die Schiefe zum Teil nicht berechnet werden kann.

Bei den bisher verwendeten, willkürlich festgelegten Schwellenwerten bei 15% bzw. 85% der Werteskala ergeben sich oft Ergebnisse, welche trotz guter Schiefe - also einem Wert der gegen 0 geht - weder als Normalverteilung im eigentlichen Sinne betrachtet werden können, noch als Idealwert für die Schwellensuche. Hier wird ein Bereich der Werteskalen erreicht, in welchem nur noch eine geringe Anzahl der analysierten Projektergebnisse liegt. Ein großes Problem, das die Anwendung des alten Schwellenwertes mit sich bringt, ist die Tatsache, dass zum Teil erhebliche Mengen an Projektwerten als Ausreißer be-

trachtet werden und somit nicht in Berechnungen einfließen. So bleiben oft nicht einmal 1 Prozent der Werte erhalten. Diese Grenze weist nur in den Fällen LEN, LENPackage und Maintainability mehr als 50 Prozent der Werte auf.

Insgesamt kann für die einzelnen Metriken auf der Grundlage dieser Arbeit eine Zuteilung der Schwellenwerte aufgestellt werden, wie sie in Tabelle 3 zu sehen ist. Diese zeigt bei den meisten Metriken bei der Schwelle von 20% im Vergleich mit den anderen drei Schwellenwerten die besten Ergebnisse.
Die Nullhypothese ist somit widerlegt.

Metrik	Schwelle	Schwellenwerte min - max	Metrik	Schwelle	Schwellenwerte min - max
CBO	20%	1 - 9	Ca	20%	0 - 16
CYC	20%	1 - 360	Ce	20%	1 - 22
DAC	20%	1 - 9	CF	10%	0.1 - 0.2
DIT	10%	0 - 3	CYC Package	20%	1 - 21
ILCOM	10%	0 - 2	EP	20%	1 - 2
LCOM	20%	0 - 42	I	10%	0.1 - 0.8
LD	10%	0.1 - 0.9	LEN Package	15%	10 - 37
LEN	10%	7 - 22	PDAC	20%	0 - 21
LOC	20%	22 - 197	Maintainab.	10%	0.1 - 0.2
LOD	15% Werteskala	0.15 - 0.85			
MPC	20%	0 - 13			
NAM	20%	2 - 14			
NOC	10%	0 - 1			
NOM	20%	1 - 9			
RFC	20%	1 - 17			
TCC	20%	0.1 - 0.2			
WMC	20%	1 -17			

Tabelle 3: Schwellenwerte Überblick

Da bei dieser Analyse nur die Schwellen bei 10%, 15%, 20% und 15% der Werteskala betrachtet werden, welche sowohl für die oberen als auch für die unteren Schwellen gelten, kann davon ausgegangen werden, dass bei einigen der betrachteten Metriken ein passenderer Schwellenwert zu finden wäre, wenn sowohl Werte zwischen den betrachteten Schwellen herangezogen werden würden, als auch Werte, die im oberen und unteren Wertebereich ungleiche Schwellenwerte aufweisen.
Absolute Schwellenwerte, welche über eine große Anzahl von Projekten ermittelt werden, ergeben durchaus sinnvolle Ergebnisse, allerdings wäre es wohl angebrachter, die Projekte in klein, mittel und groß zu unterteilen, um geeignetere absolute Schwellenwerte zu bekommen, da die Metrikwerte zum Teil mit der Größe des Projektes wachsen.
Wie die Tabelle 4 zeigt, sind aber nicht für alle Metriken die absoluten Schwellenwerte geeignet.
Für die Maintainability Metrik scheint die bisher genutzte relative Schwelle besser zu passen, da in dem Falle über 99% der Werte berücksichtigt werden und die Schiefe trotzdem keine wesentlich stärkere Abweichung von 0 hat, als bei den anderen Schwellen.
Bei den Metriken CYC, LOD, TCC und I ist eine eindeutige Bestimmung hinsichtlich der Art der Schwellenberechnung nicht möglich. Hier würde die Verwendung einer wesentlich größeren Stichprobe eventuell Klärung bringen.

Metrik	Art der Schwelle	Metrik	Art der Schwelle	Metrik	Art der Schwelle
CBO	absolut	CYC	abs./rel.	DAC	absolut
DIT	absolut	ILCOM	absolut	LCOM	absolut
LD	absolut	LEN	absolut	LOC	absolut
LOD	abs./rel.	MPC	absolut	NAM	absolut
NOC	absolut	NOM	absolut	RFC	absolut
TCC	abs./rel.	WMC	absolut	Ca	absolut
Ce	absolut	CF	absolut	CYCPackage	absolut
EP	absolut	I	abs./rel.	LENPackage	absolut
PDAC	absolut	Maint.	relativ		

Tabelle 4: geeignete Art der Schwelle

5.2 Hypothese 2: Zusammenhänge zwischen Metriken

Die zweite Hypothese befaßt sich mit der Notwendigkeit, alle Metriken des Softwarequalitätsmodells zu berechnen, da evtl. durch das Ausschließen einiger Metriken Aufwand gespart werden kann.
Über die Zusammenhänge von Metriken untereinander soll geklärt werden, ob eine indirekte Gewichtung von Metriken durch mehrmaliges Berechnen gleicher Werte besteht.

H_0: Alle Metriken sind voneinander unabhängig.

H_1: Mindestens ein Paar ist voneinander abhängig.

Um diese Hypothese prüfen zu können, muß die Korrelation zwischen den Metriken ermittelt werden.
Um die richtige Berechnungsmethode zu wählen, wird geprüft, ob eine Normalverteilung vorliegt.
Ist dies der Fall, können die Korrelationskoeffizienten nach Pearson berechnet werden. Liegt ein gegenteiliges Ergebnis vor, ist auf den Rangkorrelationskoeffizienten zurückzugreifen.
Die Betrachtung der Korrelation bezieht sich auf alle Klassenmetriken [7].

5.2.1 Datenaufbereitung

Wie in den Grundlagen bereits erklärt, muß auf Normalverteilung getestet werden, bevor die Berechnung der Korrelation zwischen zwei Variablen mit Hilfe der Pearsonmethode sinnvoll durchgeführt werden kann.
Die Eigenschaften, welche eine solche Verteilung aufweist, sind in Kapitel 2.6.1 erläutert worden und bereits in der Analyse der ersten Hypothese zur Anwendung gekommen.
Hierbei hat sich ergeben, dass keine der Metriken vollständig normalverteilte Werte aufweist. Daher wird im folgenden für alle Korrelationstests der Rangkorrelationskoeffizient genutzt.
Auch hier wird die Datenbank mit Hilfe des Excelabfrageassistenten abgefragt, wobei alle Klassenmetriken genutzt und verglichen werden.

[7] Eine Liste aller Klassenmetriken ist im Anhang in Tabelle 9 zu finden

Anschließend wird der Koeffizient, wie in Kapitel 2.6.3 beschrieben, paarweise berechnet, d.h. jede Metrik wird mit jeder anderen direkt auf Korrelation geprüft und die Ergebnisse in einer Tabelle zusammengetragen.

5.2.2 Ergebnisse

Nach der Berechnung der Rangkorrelationskoeffizienten ergeben sich die in der Tabelle 5 zusammengestellten Ergebnisse.
Die auffälligsten Ergebnisse sind hervorgehoben und als Punktdiagramme dargestellt, wobei nicht alle Datenpunkte angezeigt werden können, da in Excel nur 32.000 Datenpunkte in 2D-Diagrammen angezeigt werden. Von einem Programmwechsel wird an dieser Stelle aber abgesehen, da die Menge der Punkte ausreichend ist, um einen Eindruck von der Verteilung zu bekommen. Auf die Berechnungen der Korrelationen hat die Einschränkung keinen Einfluß, da für diese alle Daten mit einbezogen werden.
Zu sehen sind die Metrikpaare CBO-DAC, NOM-WMC, NOM-RFC, RFC-WMC und NAM-NOM, wobei vor allem bei der Darstellung CBO-DAC (Abb. 30) die starke Annäherung an eine Linie zu beobachten ist.
Das Metrikpaar LOD-WMC (Abb. 35) ist ebenfalls abgebildet, um zu zeigen wie sich eine Punktwolke verhalten kann, wenn kaum oder keine Korrelation zwischen den Werten besteht.

	CBO	CYC	DAC	DIT	ILCOM	LCOM	LD	LEN	LOC	LOD	MPC	NAM	NOC	NOM	RFC	TCC	WMC
CBO	1	0.522	0.982	0.529	0.397	0.539	0.315	0.131	0.581	0.092	0.830	0.519	0.061	0.563	0.717	0.336	0.599
CYC	0.522	1	0.527	0.594	0.538	0.580	0.558	0.251	0.347	0.321	0.547	0.402	0.552	0.449	0.405	0.672	0.417
DAC	0.982	0.527	1	0.528	0.414	0.551	0.334	0.132	0.600	0.072	0.813	0.533	0.087	0.580	0.709	0.355	0.606
DIT	0.529	0.594	0.528	1	0.391	0.405	0.430	0.266	0.142	0.245	0.534	0.165	0.406	0.238	0.277	0.543	0.204
ILCOM	0.397	0.538	0.414	0.391	1	0.478	0.794	0.079	0.477	0.307	0.576	0.632	0.572	0.598	0.529	0.781	0.570
LCOM	0.539	0.580	0.551	0.405	0.478	1	0.449	0.146	0.580	0.276	0.597	0.682	0.380	0.799	0.715	0.468	0.725
LD	0.315	0.558	0.334	0.430	0.794	0.449	1	0.157	0.325	0.372	0.505	0.468	0.620	0.480	0.412	0.803	0.440
LEN	0.131	0.251	0.132	0.266	0.079	0.146	0.157	1	-0.075	0.095	0.210	-0.04	-0.046	0.002	0.019	0.206	-0.01
LOC	0.581	0.347	0.600	0.142	0.477	0.580	0.325	-0.075	1	-0.208	0.667	0.837	-0.11	0.791	0.801	0.269	0.844
LOD	0.092	0.321	0.072	0.245	0.307	0.276	0.372	0.095	-0.208	1	0.276	0.096	0.262	0.137	0.120	0.447	0.112
MPC	0.830	0.547	0.813	0.534	0.576	0.597	0.505	0.210	0.667	0.276	1	0.627	0.216	0.650	0.817	0.510	0.712
NAM	0.519	0.402	0.533	0.165	0.632	0.682	0.468	-0.045	0.837	0.096	0.627	1	0.061	0.911	0.837	0.411	0.880
NOC	0.061	0.552	0.087	0.406	0.572	0.380	0.620	-0.046	-0.11	0.262	0.216	0.061	1	0.144	0.022	0.841	0.054
NOM	0.563	0.449	0.580	0.238	0.598	0.799	0.480	0.002	0.791	0.137	0.650	0.911	0.144	1	0.907	0.455	0.939
RFC	0.717	0.405	0.709	0.277	0.529	0.715	0.412	0.019	0.801	0.120	0.817	0.837	0.022	0.907	1	0.367	0.930
TCC	0.336	0.672	0.355	0.543	0.781	0.468	0.803	0.206	0.269	0.447	0.510	0.411	0.841	0.455	0.367	1	0.405
WMC	0.599	0.417	0.606	0.204	0.570	0.725	0.440	-0.01	0.844	0.112	0.712	0.880	0.054	0.939	0.930	0.405	1

Tabelle 5: Rangkorrelation

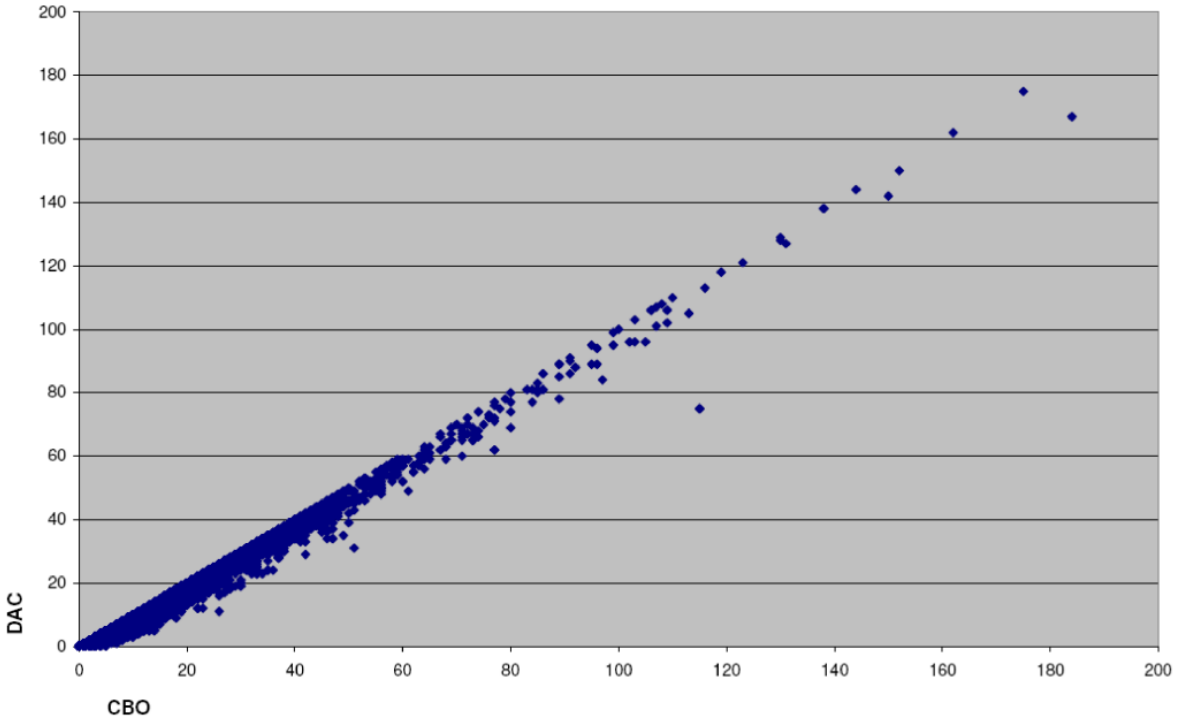


Abbildung 30: Punktwolke CBO DAC

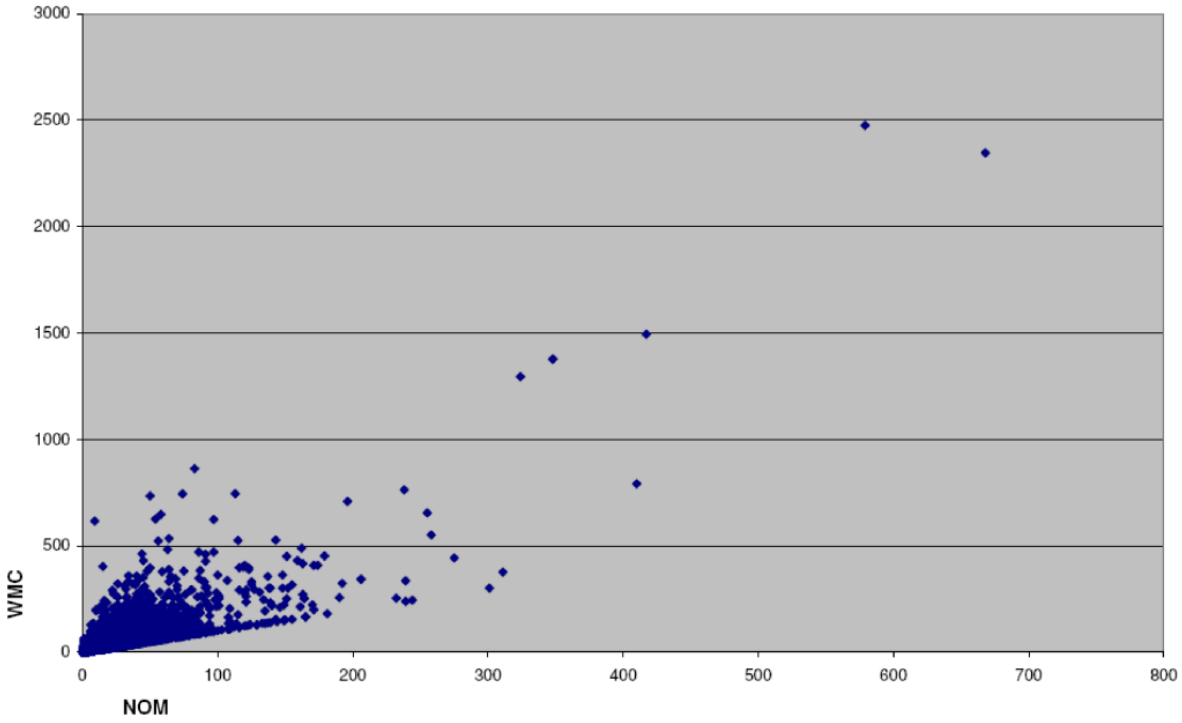


Abbildung 31: Punktwolke NOM WMC

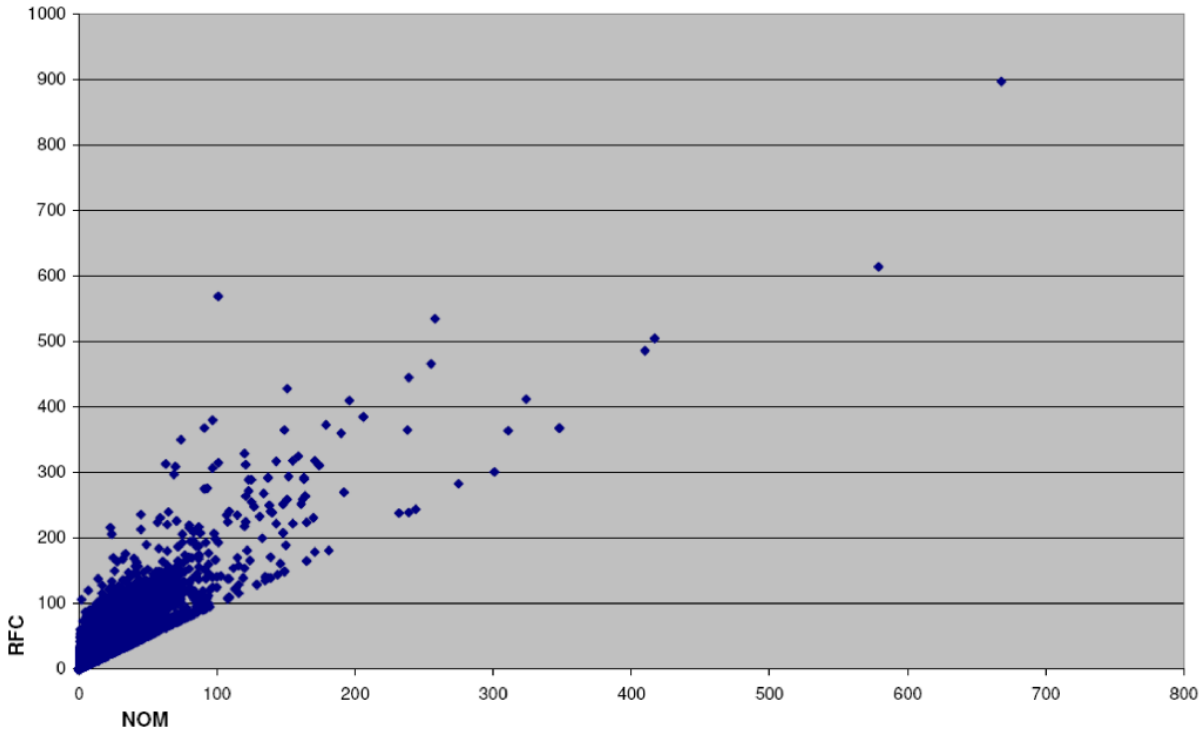


Abbildung 32: Punktwolke NOM RFC

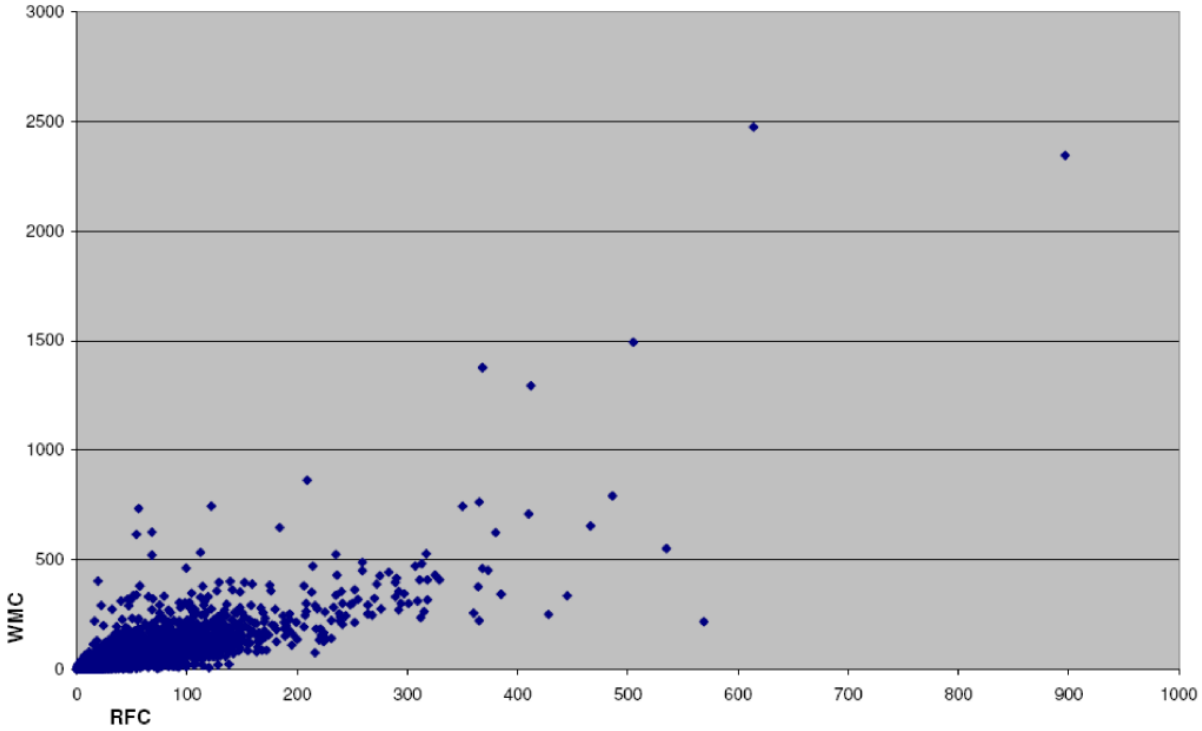


Abbildung 33: Punktwolke RFC WMC

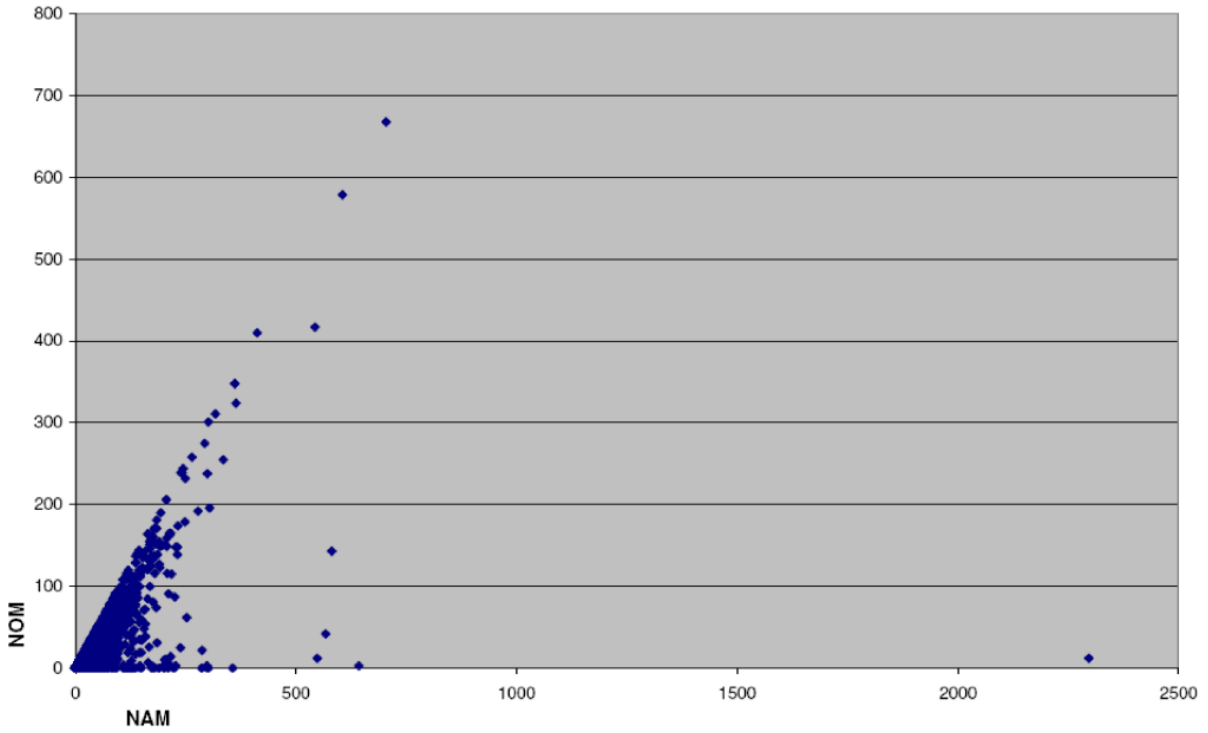


Abbildung 34: Punktwolke NAM NOM

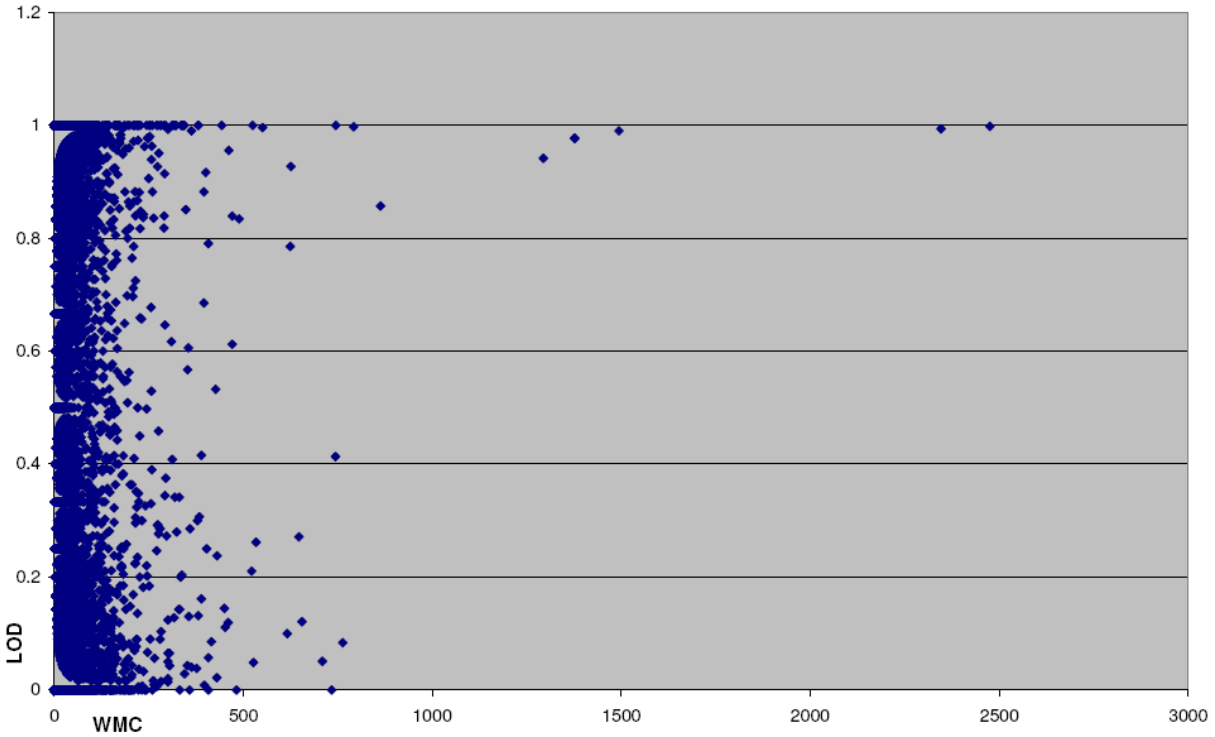


Abbildung 35: Punktwolke LOD WMC

5.2.3 Analyse

Für die Korrelationsstärken gibt es verschiedene Einstufungsvarianten. In Kapitel 2.6.3 ist eine dieser Varianten aufgezeigt.
Es ergibt sich also die Frage, wo die Grenze für einen starken Zusammenhang gesetzt werden sollte.
Da nur eine sehr starke Korrelation in Frage kommt, um eine Metrik ausschließen zu können bzw. um zeigen zu können, dass nicht alle Metriken benötigt werden, wäre ein Wert >= |0.9| angebracht.
Werden die Ergebnisse auf dieser Grundlage gesichtet, lassen sich in der Tabelle mit den zuvor berechneten Korrelationsergebnissen (Tabelle 5) folgende markante Metrikpaare entnehmen (blau):

CBO-DAC	NOM-WMC	RFC-WMC	NAM-NOM	NOM-RFC
0.982	0.939	0.930	0.911	0.902

Tabelle 6: Korrelationsergebnisse >= 0.9

Die Metriken CBO und DAC zeigen mit 0.982 eine sehr hohe Korrelation und somit einen absoluten Zusammenhang, daher könnte eine dieser Metriken aus mathematischer Sicht entfallen.
Auch kommen folgende Metriken einem starken Zusammenhang sehr nahe:

- NAM
- NOM
- RFC
- WMC

Diesen Korrelationspaaren zufolge sind auch zwischen den Paaren NAM-RFC und NAM-WMC starke Zusammenhänge zu erwarten, was die Korrelationsergebnisse von 0.880 und 0.837 bestätigen (Abb. 5, grün).
Also könnten 4 der 6 Metriken, welche durch diese Berechnungen aufgefallen sind, aus den Betrachtungen genommen werden und aus dem Softwarequalitätsmodell entfallen. Allerding müssen Inhalt und Aussage dieser näher betrachtet werden, um entscheiden zu können, für welche dieser Metriken dieses in Frage kommt.

Aufgrund dieser Ergebnisse ist die Nullhypothese als widerlegt zu erachten, da es mindestens ein Metrikpaar gibt, das so stark korreliert, dass eine dieser Metriken aus der Betrachtung genommen werden kann.

6 Evaluierung

In diesem Kapitel wird die Anwendung der bisherigen Ergebnisse auf den Basisdatenbestand beschrieben, d.h. die Wirkung von getroffenen Maßnahmen wird bewertet. In der vorliegenden Arbeit bezieht sich dieses auf die Schwellenwertberechnungen, die Berechnung der Korrellationen zwischen den Metriken und deren Wirkung auf die Berechnung der Maintainabilitywerte mit Hilfe von Softwarequalitätsmodellen.

6.1 Schwellenwerte

Für eine Bewertung der ermittelten absoluten Schwellenwerte der Metrikverteilungen (5.2.2) werden Projekte ausgewertet, welche in der bisherigen Analyse noch nicht betrachtet wurden. Hierbei werden der bisher willkürlich gesetzte Schwellenwert von 15% der Werteskala den anderen Schwellenwerten (10%, 15% und 20%) gegenübergestellt. Also die relative und die absoluten Schwellen verglichen.
Die Projekte sind ebenfalls aus dem Open Source Bereich, in Java geschrieben und weisen verschiedene Größen auf [8]. Sie werden wie die bisher verwendeten Projekte aufbereitet (Kap. 4) und mittels VizzAnalyzer vermessen.
Anschließend werden die im Laufe der Arbeit berechneten absoluten Werte der 10%, 15% und 20% Schwellen (Tabellen 1 und 2) auf die Verteilungungen angewandt. Ermittelt wird die Anzahl der Klassen, welche jeweils erhalten bleiben, also nicht als Ausreißer gesehen werden.
Dieses wird ebenso mit der bisher geltenden Schwellen von 15% der Werteskala wiederholt, welche in relativer Form erhalten bleibt.
Die Tabelle 7 zeigt die verbliebene Wertemenge nach der Verwendung der Schwellenwerte in relativen Zahlen, jeweils für die 10%, 15% und 20% Schwellen und die relative Schwelle von 15% der Werteskala. Auch ist der Mittelwert der verbliebenen Wertemengen zum Vergleich angegeben.
Hierbei zeigt sich, dass die Schwelle mit den 15% der Werteskala nur wenige Klassen berücksichtigt (durchschnittlich 31.147%), während zum Vergleich die Schwelle mit den absoluten Werten der 15% Schwelle weniger Werte als Ausreißer behandelt und durchschnittlich 81.197% der Klassen beibehält. Dieses ergibt sich durch die teils sehr aus-

[8] Die hier genutzten Projekte sind im Anhang in Tabelle 13 aufgelistet

gedehnten Spannweiten der Verteilungen der unterschiedlichen Metriken, auf welche die absoluten Schwellenwerte nicht, dafür die relativen Schwellen stark reagieren.
Die 10% Schwelle brücksichtigt mit einem Durchschnitt von 86.3% der Werte die meisten Klassen.
Da bei der Verwendung des relativen Schwellenwertes von 15% größtenteils weniger als 50% aller Werte verbleiben, kann hier gesagt werden, dass die absoluten Schwellenwerte geeigneter erscheinen. Auch ist der Unterschied zwischen den absoluten und der relativen Grenze durch den direkten Vergleich der absoluten und der relativen 15% Schwelle deutlich zu sehen.

Schwellen/ Metrik	10%	15%	20%	15% Werteskala
CBO	81.067	78.795	76.932	36.432
CYC	94.848	94.848	94.848	7.322
DAC	77.591	75.433	74.013	33.222
DIT	96.645	90.475	90.475	50.527
ILCOM	89.730	89.730	80.969	35.963
LCOM	88.456	83.985	79.528	4.101
LD	94.222	94.222	94.222	29.048
LEN	77.602	59.565	50.669	75.353
LOC	78.286	68.106	56.402	18.663
LOD	68.567	68.567	68.567	25.932
MPC	93.358	90.655	84.445	15.328
NAM	81.963	76.851	54.771	22.279
NOC	92.013	81.024	81.024	34.644
NOM	88.453	75.230	68.540	23.674
RFC	91.433	79.277	75.430	31.411
TCC	83.188	77.555	72.096	26.502
WMC	85.927	75.908	69.437	20.659
Ca	85.272	81.730	80.605	11.676
Ce	87.577	85.160	83.952	28.006
CF	82.090	82.090	68.436	37.130

CYCPackage	91.416	91.416	88.5	60
EP	93.101	88.559	83.809	32.132
I	77.866	77.866	64.657	27.750
LENPackage	91.481	77.752	66.749	63.015
PDAC	87.410	85.119	83.869	27.922
Mittelwert	86.382	81.197	75.718	31.147

Tabelle 7: Verbliebener Anteil Klassen

6.2 Korrelationen

Ausgehend von den Korrelationsergebnissen dieser Arbeit (5.2.2) sind innerhalb des zugrunde liegenden Forschungsprojektes neue Softwarequalitätsmodelle implementiert worden, welche zum Teil die neu festgelegten, absoluten Werte verwenden. Für diese Modelle wurden beispielhaft neue Maintainability Werte berechnet, welche im Folgenden verglichen werden, um die Wirkung der getroffenen Massnahmen zu betrachten.

- Maintainability (MA) - umfasst die original Maintainability, verwendet also alle Metriken und 15% des Wertebereichs innerhalb eines Projekts.

- MaintainabilityReduced (MRA) - verwendet die Metriken, die in der Korrelatonsanalyse (Hypothese 2) nicht ausgeschlossen worden sind, also die unabhängigen Metriken, und umfasst 15% des Wertebereichs innerhalb eines Projekts.

- MaintainabilityAbsolute15 (MA15A) - verwendet alle Metriken, aber nutzt die absolute 15% Ober-/Untergrenze, wie sie in Hypothese 1 für jede Metrik errechnet wurde.

- MaintainabilityAbsolute15Reduced (MA15RA) - verwendet die Metriken, die in der Korrelatonsanalyse nicht ausgeschlossen worden sind und nutzt die absolute 15% Ober-/Untergrenze.

Die so berechneten neuen Werte der Maintainability werden mit Hilfe der Rangkorrelation mit den alten Maintainability-Werten und untereinander verglichen.

	MA	MRA	MA15A	MA15RA
MA	1	0.974	0.697	0.750
MRA	0.974	1	0.734	0.784
MA15A	0.697	0.734	1	0.936
MA15RA	0.750	0.784	0.936	1

Tabelle 8: Maintainabilitykorrelation

Tabelle 8 zeigt sowohl zwischen MaintainabilityAverage und MaintainabilityReducedAverage, als auch zwischen MaintainabilityAbsolute15Average und MaintainabilityAbsolute15ReducedAverage hohe Zusammenhänge mit Korrelationsergebnissen größer 0.9.
Aufgrund dieser Ergebnisse kann gesagt werden, dass, zumindest die Maintainability Metrik betreffend, zwischen den Softwarequalitätsmodellen mit oder ohne die durch die Korrelationsanalyse ausgeschlossenen Metriken nur kleine Unterschiede bestehen. Das heißt, dass es möglich ist, das reduzierte Softwarequalitätsmodell zu verwenden und damit den Aufwand bei der Metrikberechnung zu reduzieren ohne die Genauigkeit des Ergebnisses wesentlich zu beeinflussen. Zur Verdeutlichung der Korrelationen, sind die auffälligen Korrelationsergebnisse zusätzlich als Punktwolken dargestellt (Abb. 36 und 37).
Die übrigen in Tabelle 8 dargestellten Werte weisen einen weniger starken Zusammenhang auf. Dieses verdeutlicht exemplarisch die Abbildung 38 anhand des Zusammenhanges zwischen MRA und MA15RA. Dieses zeigt, dass die Berechnung der Metriken mit den absoluten Schwellenwerten, ermittelt durch die Verwendung der 15% Schwelle bezogen auf die Anzahl der Klassen, andere Ergebisse erzeugt als die Berechnung mit der relativen 15% Schwelle, welche sich auch die Werteskala bezieht.

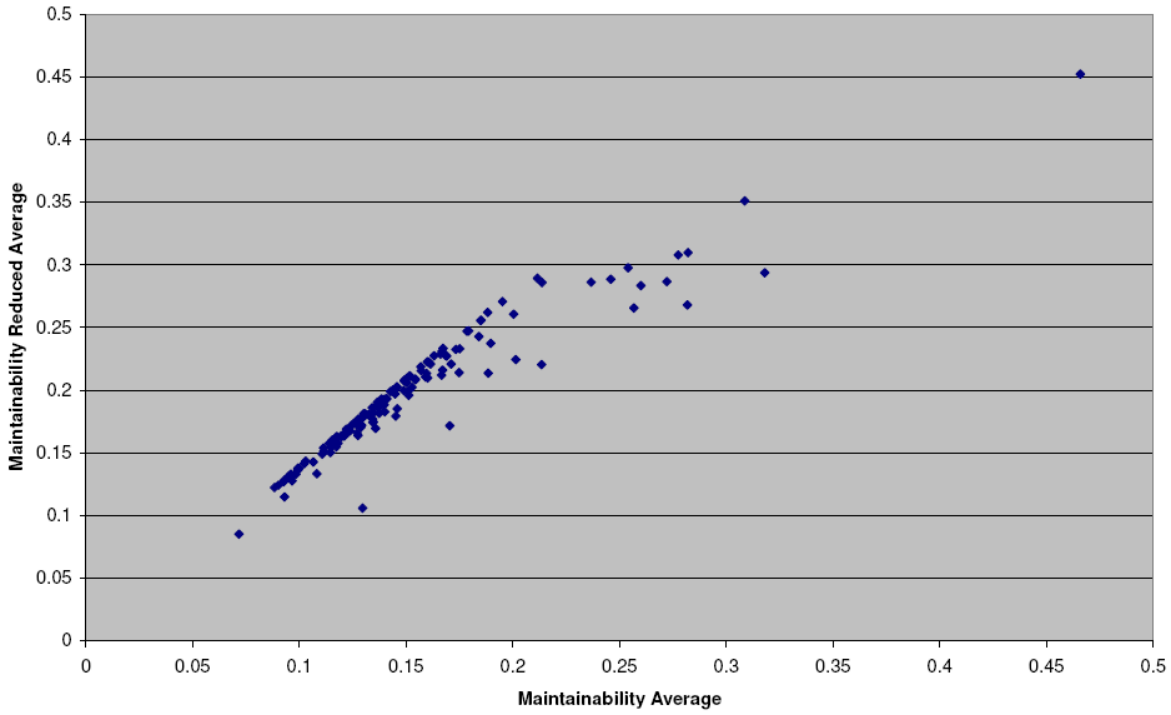


Abbildung 36: Punktwolke MA und MRA

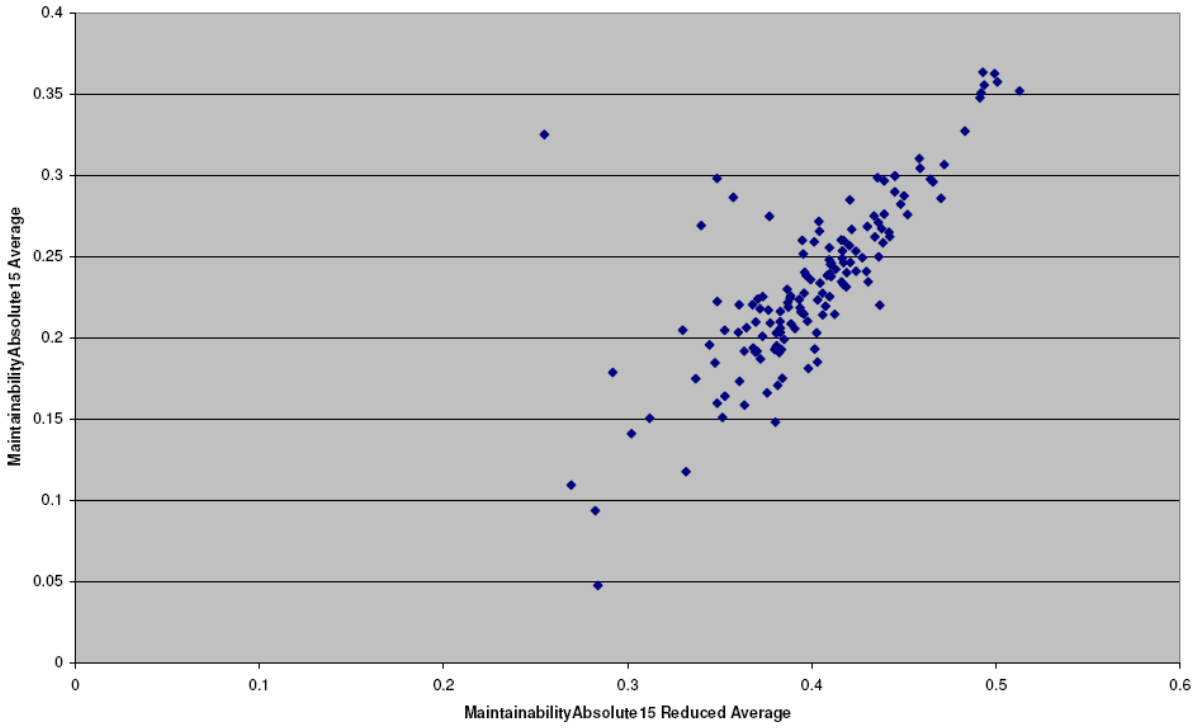


Abbildung 37: Punktwolke MA15A und MA15RA

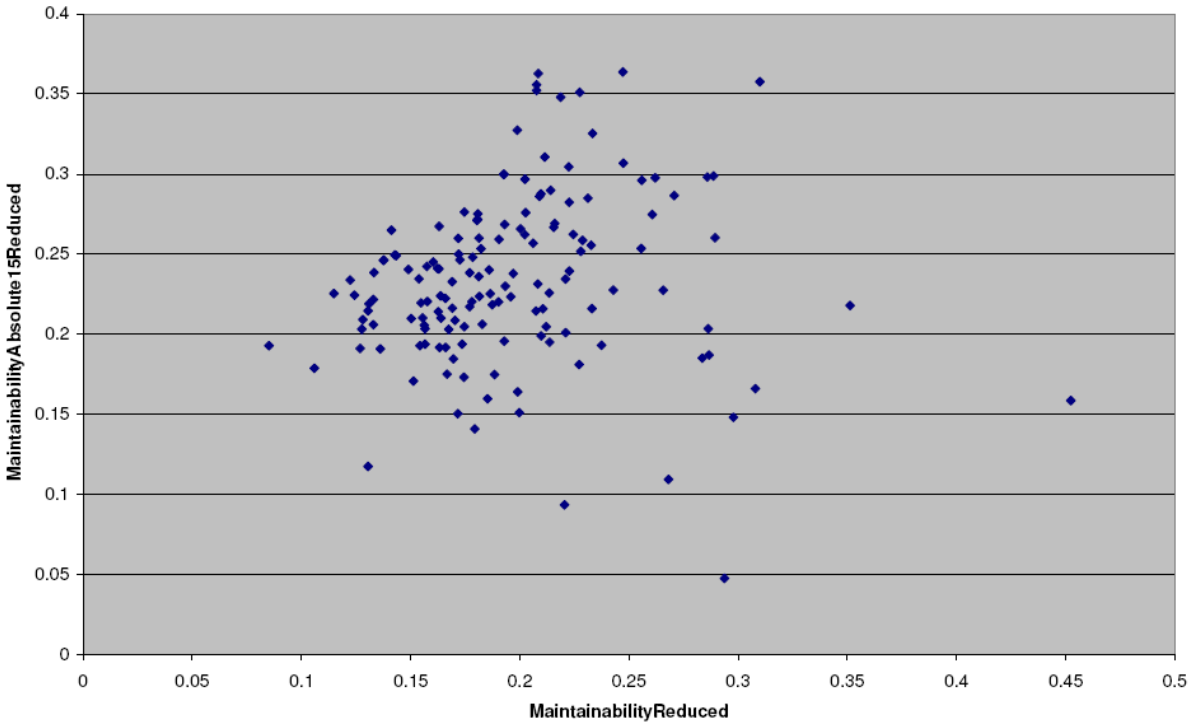


Abbildung 38: Punktwolke MRA und MA15RA

6.3 Fazit der Evaluierung

Durch die Anwendung der Ergebnisse hat sich gezeigt, dass es durch den Ausschluss der ausgewählten Metriken zu keinen wesentlichen Veränderungen der Werte, beispielhaft berechnet für die Maintainability Metrik, gekommen ist.

Die Verwendung der absoluten Schwellenwerte, exemplarisch gezeigt für die 15% Schwelle, legt offen, dass ein nicht unerheblicher Unterschied zwischen den relativen und absoluten Werten besteht. Hieraus folgt die Annahme, dass es wahrscheinlich von Vorteil wäre für die Metriken individuelle Schwellenwerte zu wählen und zu verwenden.

7 Abschlussbetrachtung

Abschließend wird noch einmal auf die Fragestellungen eingegangen, deren Test- und Analyseergebnisse zusammengefasst und ein Ausblick auf weitere mögliche Auswertungen gegeben.

Die Betrachtung der Schwellenwerte beinhaltete die Suche nach einem geeigneteren als dem bisher genutzten und willkürlich festgelegten Schwellenwert und stellte die Frage nach dem Sinn, absolute Zahlenwerte zu ermitteln und anzugeben.
Nach den Analysen kann gesagt werden, dass die neu berechneten absoluten Schwellenwerte für die Betrachtung der Metrikwerteverteilungen hinsichtlich Ausreißern besser geeignet sind, als der zuvor genutzte Wert von 15% der Werteskala und eine annehmbare Alternative darstellen, da diese den Großteil der vom VizzAnalyzer vermessenen Werte betrachten. Hierbei kann der Schwellenwert bei 10% als sinnvollster betrachtet werden, wobei auch hier Steigerungen durch das Ermitteln optimaler und individueller Schwellenwerte für die oberen und unteren Schwellenbereiche jeder Metrik möglich sind.
Eine weitere Möglichkeit der Optimierung ist das Unterteilen der zu analysierenden Projekte in Größenordnungen wie klein, mittel und groß, denn die Schwellenwerte wären dann besser angepasst an die zum Teil mit der wachsenden Größe eines Projektes steigenden Metrikwerte. Für weitere Analysen würde sich ein Programm eignen, welches im Rahmen einer Bachelor-Thesis entstanden ist [Zhe08]. Mit desses Hilfe könnten alle Projekte, die in Java geschrieben wurden und somit interessant sind, von www.sourceforge.net heruntergeladen werden, wodurch eine größere Menge an Projekten berücksichtigt werden könnte.

Die Korrelationsanalysen für den Test auf die Notwendigkeit aller Metriken (Klassenmetriken) sollten eine Entscheidung möglich machen, ob Metriken aus der Betrachtung ausgeschlossen werden können und ob sich auf diese Weise Aufwand verringern ließe.
Starke Korrelationen wurden zwischen fünf Metrikpaaren festgestellt. Hierbei wurde ohne Beachtung der inhaltlichen Zusammenhänge festgestellt, dass vier Metrikpaare genauer betrachtet und evtl. ausgeschlossen werden sollten, um eine eventuelle versteckte Gewichtung einzelner Metriken auszuschließen.
Die Ergebnisse der Korrelationsbetrachtung können also aus theoretischer Sicht den Auf-

wand der Vermessungen mit dem VizzAnalyzer verringern. Diese Resultate sind evtl. auch auf andere, noch nicht betrachtete Metriken, auszuweiten, wobei auch die Nutzung eines niedrigeren Grenzwertes in Augenschein genommen werden könnte. Allerdings ist es notwendig, die inhaltlichen Zusammenhänge der Metriken zu untersuchen, um so genannte „Nonsenskorrelationen“ zu vermeiden.

Nach der Anwendung der Ergebnisse kann gesagt werden, dass die bisherigen und die auf absoluten Werten basierenden Softwarequalitätsmodelle unterschiedliche Schlussfolgerungen zulassen könnten und deshalb getrennt voneinander betrachtet werden müssen. Auch zeigt sich, dass es möglich ist, ein reduzierte Softwarequalitätsmodell zu verwenden und damit Aufwand bei der Metrikberechnung zu sparen ohne die Genauigkeit des Ergebnisses zu beeinflussen.

A Anhang

A.1 Metriken

Im Folgenden findet sich eine kurze Beschreibung der Metriken, welche in der Thesis angesprochen worden sind. Für detailierte Informationen wird auf die angegebenen Referenzen bzw. das Compendium of Software Quality Standards and Metrics (`http://www.arisa.se/compendium/`) verwiesen.

Klassenmetriken	
CBO	Coupling Between Objects ist eine Kohäsionsmetrik. Sie bezeichnet die Anzahl der Klassen eines Systems, die von einer Klasse verwendet werden, d.h. zu denen c durch jegliche Verwendungsbeziehung gekoppelt ist. [CK91, CK94, LH93, BK95, BBC+99, Bal96, CS95, HM95, HM96, HCN98b, BDW99, SK03, MH99]
CYC	Number of classes in cycle ist eine Kopplungsmetrik. Sie bezeichnet die Anzahl der Klassen eines Systems, die sich mit der Klasse in einer zyklischen Beziehung befinden.
DAC	Data Abstraction Coupling ist eine Kohäsionsmetrik. Sie bezeichnet die Anzahl der Verbindungen, welche durch abstrakte Datentypen verursacht werden. Der Wert 0 zeigt, dass keine Verbindung zu einem anderen abstrakten Datentyp besteht, während höhere Werte erkennen lassen, dass die Verbindungen einer Klasse mit anderen Klassen komplexer sind. [LH93, BBC+99, HM95, BDW99]
DIT	Depth Of Inheritance Tree ist eine Vererbungsstrukturmetrik. Sie bezeichnet die Anzahl der Klassen, die in der Vererbungshierarchie über der betrachteten Klasse liegen. Sie zeigt wie viele Super-Klassen auf die betrachtete Klasse einwirken können. [CK91, CK94, LH93, BK95, BBC+99, Bal96, CS95, HM95, HM96, HCN98b, BDW99, SK03, MH99]

ILCOM	Improvement of LCOM ist eine Kohäsionsmetrik. Sie bezeichnet die Anzahl der nicht verbundenen Methodenkluster einer Klasse. Ein Methodenkluster besteht aus Methoden einer Klasse, die sich Instanzvariablen einer Klasse teilen. Je geringer die Zahl dieser ist, desto höher ist der Zusammenhang der Methoden einer Klasse. [HM95, HM96, BBC+99]
LCOM	Lack of Cohesion in Methods ist eine Kohäsionsmetrik. Sie bezeichnet die Anzahl der nicht verbundenen Methodenpaare einer Klasse. [CK91, CK94, LH93, BK95, BBC+99, Bal96, CS95, HM95, HM96, HCN98b, BDW99, SK03, MH99]
LD	Locality of Data ist eine Datenkapselungsmetrik. Sie bezeichnet die Menge der Daten, die direkt einer Klasse angehören, im Vergleich zu der Menge von Daten, die von dieser Klasse verwendet werden. Lässt Schlüsse auf die Wiederverwendtbarkeit und die Testbarkeit zu. [HM95, HM96, BBC+99]
LEN	Length of Names bezeichnet die Anzahl der Buchstaben in Klassen-, Package-, Methoden- und Feld-Namen.
LOC	Lines of Code ist eine Komplexitätsmetrik. Sie bezeichnet die Anzahl der Codezeilen in Klassen und Interfaces. Lässt Schlüsse über die Komplexität zu. [BBC+99]
LOD	Lack Of Dokumentation ist der Prozentsatz an nicht dokumentierten Elementen einer Klasse im Vergleich zu den Elementen, die dokumentiert werden sollten. Der Aufbau und der Inhalt des Kommentars werden nicht beachtet. Der Wert 0 zeigt eine vollständige Dokumentation. Ein höherer Wert deutet auf das Fehlen von Dokumentation hin.

MPC	Message Passing Coupling ist eine Kopplungsmetrik. Sie bezeichnet die Anzahl der Aufrufe, die in Methoden einer Klasse festgelegt sind, von Methoden anderer Klassen. Sie zeigt somit die Anhängigkeiten zwischen lokaler Methoden und den Methoden anderer Klassen. Lässt Schlüsse über die Wiederverwendbarkeit, die Wartbarkeit und das Testen zu. [LH93, BBC+99, BDW99, HM95]
NAM	Number of Attributes and Methods ist eine Komplexitätsmetrik. Sie bezeichnet die Anzahl der Attribute (Felder) und Methoden, die in einer Klasse definiert sind.
NOC	Number Of Children ist eine Vererbungshierarchiemetrik. Sie bezeichnet die Anzahl der Klassen, welche direkt von einer Klasse erben. [CK91, CK94, LH93, BK95, BBC+99, Bal96, CS95, HM95, HM96, HCN98b, BDW99, SK03, MH99]
NOM	Number of local Methods ist eine (Schnittstellen-) Komplexitätsmetrik. Sie bezeichnet die Anzahl der Methoden, die in einer Klasse lokal deklariert sind. Geerbte Methoden werden nicht gezählt. Lässt Schlüsse über die Komplexität zu. [LH93, BBC+99, BDW99, HM95]
RFC	Response For a Class ist eine Komplexitätsmetrik. Sie bezeichnet die Anzahl der Methoden einer Klasse, sowie der Methoden, die von diesen aufgerufen werden. [CK91, CK94, LH93, BK95, BBC+99, Bal96, CS95, HM95, HM96, HCN98b, BDW99, SK03, MH99]
TCC	Tight Class Cohesion ist eine Kohäsionsmetrik. Sie bezeichnet den Zusammenhang zwischen Methoden einer Klasse durch das Verhältnis der Anzahl der tatsächlichen Methodenpaare zu der Anzahl der möglichen Verbindungen. [BK95, BBC+99]

WMC	Weighted Method Count ist eine Komplexitätsmetrik. Sie bezeichnet die gewichtete Summe der Methoden einer Klasse. [CK91, CK94, LH93, BK95, BBC+99, Bal96, CS95, HM95, HM96, HCN98b, BDW99, SK03, MH99]

Tabelle 9: Klassenmetriken

Packagemetriken	
Ca	Afferent Coupling ist eine Kopplungsmetrik auf Packageniveau. Sie bezeichnet die Anzahl der externen Klassen, welche verbunden sind mit den Klassen eines Packages beruhend auf hineinführenden Verbindungen. [Mar94]
Ce	Efferent Coupling ist eine Kopplungsmetrik auf Packageniveau. Sie bezeichnet die Anzahl der externen Klassen, welche verbunden sind mit den Klassen eines Packages beruhend auf hinausführenden Verbindungen. [Mar94]
CF	Coupling Factor ist eine Kopplungsmetrik auf Klassenniveau. Sie misst die Verbindung zwischen Klassen ohne auf Vererbung beruhenden Verbindungen. [eAC94, BBC+99, MH99, HCN98a]
CYC_Packages	Number of Packages in Cycle ist eine Kopplungsmetrik. Sie bezeichnet die Anzahl der Packagee eines Systems, die sich mit einem Package in einer zyklischen Beziehung befinden.
EP	Encapsulation Principle ist eine Kopplungsmetrik. Sie bezeichnet die Klassen ausserhalb eines Packages, die von den Klassen in dem Package verwendet werden (efferent coupling, Ce) im Verhältnis zu der Anzahl der Klassen n in dem Package (Ce/n).
I	Instability misst das Verhältnis von hinausgehenden und hinaus- und hineingehenden Verbindungen von Klassen innerhalb eines Package mit Klassen anderer Packages $(Ce/(Ca + Ce))$. [Mar94]
LEN	Length of Names bezeichnet die Anzahl der Buchstaben in Klassen-, Package-, Methoden- und Feld-Namen.

PDAC	Package data Abstraction Coupling misst die durch abstrakte Datentypen auf dem Package-Level verursachte Verbindungskomplexität.

Tabelle 10: Packagemetriken

sonstige Metriken	
Maintainability	Läßt Schlüsse darüber zu, wie gut eine Software gewartet werden kann.

Tabelle 11: sonstige Metriken

A.2 Projektliste

Folgende Tabellen zeigen die komplette Liste aller im Rahmen dieser Arbeit verwendeten Projekte und die dazu gehörigen Unix-Namen, welche in dieser Arbeit unter anderem zum automatischen Einlesen der Metadaten benötigt werden.

Projektname	Unix-Name
Abbot Java GUI Testing Framework	abbot
AJCT	ajct
Alster	alster
Ant task for Doxygen	ant-doxygen
AnTLR	antlr
ANTLR Testing	antlr-testing
AnTLR-works	antlr-works
Art of Illusion	aoi
Asterisk-Java Library	asterisk-java
Autofetch	autofetch
BareHTTP	barehttp
Beaver	beaver
blogunity	blogunity
Buddi	buddi
Bugtracker	bugtracker
CD Catalog	cdbrowser
Chaperon	chaperon
Class Viewer	classviewer

Projektname	Unix-Name
Coco-R	cocor
CraftyFTP	craftyftp
CroftSoft Code Library	croftsoft
CUP	cup
DbUnit	dbunit
dcm4che, a DICOM Implementation in JAVA	dcm4che
DJDoc	djdoc
DLOG4J	dlog4j
dwr20	dwr20
dynaop	dynaop
EasyJCE	ejce
EGG - Easy Generator Generator	the-egg
EJE (Everyone's Java Editor)	eje
Emonic	emonic
Essence Java Framework	essence
EXTE	exte
EZMorph	ezmorph
Firemox, Turn Based Strategy Game	firemox
FIT Decorator	fitdecorator
Free Safe	freesafe
FreeCol	freecol
FreeLords	freelords
Frinika	frinika
grammatica	grammatica
Granite Data Services	granite
HSQL Database Engine	hsqldb
HTML Form Validator	htformvalidator
Hybrid Petri Net ICSI Simulator	hisim
Invicta	invicta
j2ssh	j2ssh
Jabble	jabble
Jailer, Model-based Data Export Tool	jailer
Jaim - Java AIM Protocol library	jaimlib
Jalita	jalita
Jar Jar Links	jarjar
JasperReports - Java Reporting	jasperreports
JAud	jaud
Java Assembling Language	jasml
Java Classic RPG	javacrpg

Projektname	Unix-Name
Java compiler & operators overloading	jo2
Java Crypt Framework	jcryptfx
Java Date Picker	javadatepicker
Java library and framework: quantity	javaquantity
Java MMS Downloader	javamms
Java Online Gaming Real-time Engine	jogre
Java Simple Argument Parser	jsap
Java tokenizer and parser tools	jtopas
Java Virtual System	jvs-vfs
Javalobby Community Platform	gotjava
javavis	javavis
jConfig	jconfig
jdk118	jdk118
jdk122	jdk122
jdk131	jdk131
jdk142	jdk142
jdk150	jdk150
jEdit	jedit
Jena	jena
JFish	jfish
JFlex	jflex
JHotDraw	jhotdraw
JMario	jmario
jMemorize	jmemorize
jms	jms
JNC-API	jnc-api
joscar	joustim
Jparsec	jparsec
JPdfUnit	jpdfunit
JSch	jsch
JSDSI	jsdsi
JSettlers	jsettlers
JSkat	jskat
JSqsh	jsqsh
jTPCC	jtpcc
JUICE	juiceforge
JWebChart	jwebchart
JWhoisServer	jwhoisserver
jxarcade	jxarcade

Projektname	Unix-Name
KoLmafia	kolmafia
LaTeXDraw	latexdraw
LiquiBase	liquibase
Macad	macad
Mac-Package	macpackage
MFlow	mflow
mp3dings	mp3dings
myPod	mypod
netx	jnlp
ocwfinder	ocwfinder
On-Line Java Games	javagames
OpenICQ	openicq
Openmim	openmim
PaperClips	paperclips
PDFBox	pdfbox
Petri	sirius-petri
PicWeb	picweb
Plazma	plazma
PMD	pmd
ProGuard Java Optimizer and Obfuscator	proguard
Quick Sequence Diagram Editor	sdedit
r-ant	r-ant
RECODER	recoder
Retrotranslator	retrotranslator
Robocode	robocode
RunCC	runcc
SableCC	sablecc
SASAX	sasax
Sesame	sesame
Simple	simpleweb
SimplePDF	simplepdf
SJPT	sjpt
Software Process Dashboard	processdash
Solinger Java Utilities Project	solinger
SQuirreL SQL Client	squirrel-sql
StoryTestIQ	storytestiq
Su Doku Solver	sudoku
suprsukr	suprsukr
SyWiCo	sywico

Projektname	Unix-Name
Taste	taste
tomcat	tomcat
Transparent RMI	trmi
TrueCrypt GUI for Linux	jtcgui
uEngine BPM	uengine
WeirdX	weirdx
Wicket	wicket
WorkingCode	workingcode
XDC	xdc

Tabelle 12: Projektliste

Projektname	Unix-Name
jdk160	jdk160
CowCatcher - Java Course Development	cowcatcher
Java DjVu Viewer	javadjvu
Joj - Java Version of Java	joj
JINX - java multi-user unix-like system	jinx00
BlackJack for Java: A Java/Swing Oddessy	javablackjack
VI Toolkit for Java	vitforjava

Tabelle 13: Projektliste2

A.3 Inhalt der DVD

- Diplomarbeit als .pdf

- Kurzfassung als .pdf

- Plakat als .pdf

- Exceltabellen mit den Berechnungen (Schwellenwerte, Korrelationen)

- Projektsource

Abbildungsverzeichnis

1 Formen der Verteilungsausprägung 7
2 Korrelationsausprägungen 8
3 Ablauf der Datenerfassung 13
4 CBO Schwellen 20
5 CYC Schwellen 20
6 DAC Schwellen 21
7 DIT Schwellen 21
8 ILCOM Schwellen 22
9 LCOM Schwellen 22
10 LD Schwellen 23
11 LEN Schwellen 23
12 LOC Schwellen 24
13 LOD Schwellen 24
14 MPC Schwellen 25
15 NAM Schwellen 25
16 NOC Schwellen 26
17 NOM Schwellen 26
18 RFC Schwellen 27
19 TCC Schwellen 27
20 WMC Schwellen 28
21 Ca Schwellen 28
22 Ce Schwellen 29
23 CF Schwellen 29
24 CYCPackage Schwellen 30
25 EP Schwellen 30
26 I Schwellen 31
27 LENPackage Schwellen 31
28 PDAC Schwellen 32
29 Maintainability Schwellen 32
30 Punktwolke CBO DAC 40

31 Punktwolke NOM WMC 40
32 Punktwolke NOM RFC 41
33 Punktwolke RFC WMC 41
34 Punktwolke NAM NOM 42
35 Punktwolke LOD WMC 42
36 Punktwolke MA und MRA 49
37 Punktwolke MA15A und MA15RA 49
38 Punktwolke MRA und MA15RA 50

Tabellenverzeichnis

1 Schwellenwerte Klassenmetriken 33
2 Schwellenwerte Packagemetriken 34
3 Schwellenwerte Überblick 35
4 geeignete Art der Schwelle 36
5 Rangkorrelation 39
6 Korrelationsergebnisse >= 0.9 43
7 Verbliebener Anteil Klassen 47
8 Maintainabilitykorrelation 48
9 Klassenmetriken 56
10 Packagemetriken 57
11 sonstige Metriken 57
12 Projektliste 61
13 Projektliste2 61

Verzeichnis der Literatur- und Internetquellen

[Bal96] N. V. Balasubramanian. Object-Oriented Metrics. In APSEC '96: Proceedings of the Third Asia-Pacific Software Engineering Conference, page 30, Washington, DC, USA, 1996. IEEE Computer Society.

[BBC+99] H. Bär, M. Bauer, O. Ciupke, S. Demeyer, S. Ducasse, M. Lanza, R. Marinescu, R. Nebbe, O. Nierstrasz, M. Przybilski, T. Richner, M. Rieger, C. Riva, A. Sassen, B. Schulz, P. Steyaert, S. Tichelaar, and J. Weisbrod. The FAMOOS Object-Oriented Reengineering Handbook. `http://www.iam.unibe.ch/~famoos/handbook/`, October 1999.

[BDW99] L. C. Briand, J. W. Daly, and J. K. Wüst. A Unified Framework for Coupling Measurement in Object-Oriented Systems. IEEE Trans. Softw. Eng., 25(1):91–121, 1999.

[BK95] J. M. Bieman and B. Kang. Cohesion and Reuse in an Object-Oriented System. In SSR '95: Proceedings of the 1995 Symposium on Software reusability, pages 259–262, New York, NY, USA, 1995. ACM Press.

[CK91] S. R. Chidamber and C. F. Kemerer. Towards a Metrics Suite for Object Oriented Design. In OOPSLA '91: Conference proceedings on Object-oriented programming systems, languages, and applications, pages 197–211, New York, NY, USA, 1991. ACM Press.

[CK94] S. R. Chidamber and C. F. Kemerer. A Metrics Suite for Object Oriented Design. In IEEE Transactions on Software Engineering, volume 20 (6), pages 476–493, June 1994.

[CS95] N. Churcher and M. Shepperd. Comments on "A Metrics Suite for Object Oriented Design". In IEEE Transactions on Software Engineering, volume 21 (3), pages 263–265, 1995.

[Die01] Andreas Diekmann. Empirische Sozialforschung, Grundlagen, Methoden, Anwendungen, 7.Auflage. rowohlts enzyklopädie, 2001.

[DOG08] Oguzhan DOGAN. Defining a software analysis framework. School of Mathematics and Systems Engineering, 5 2008.

[eAC94] F. Brito e Abreu and R. Carapuça. Object-Oriented Software Engineering: Measuring and Controlling the Development Process. In 4th International Conference on Software Quality, October 1994. McLean, Virginia, USA.

[Erl08] Universität Erlangen. `http://www.imbe.med.uni-erlangen.de/lehre/Querschnittsbereich1/Unterlagen/VL_KorrelationWS0708.pdf`, 7 2008.

[HCN98a] R. Harrison, S. J. Counsell, and R. V. Nithi. An Evaluation of the MOOD Set of Object-Oriented Software Metrics. IEEE Trans. Softw. Eng., 24(6):491–496, 1998.

[HCN98b] R. Harrison, S. J. Counsell, and R. V. Nithi. An Investigation into the Applicability and Validity of Object-Oriented Design Metrics. Empirical Software Engineering, 3(3):255–273, 1998.

[HM95] M. Hitz and B. Montazeri. Measure Coupling and Cohesion in Object-Oriented Systems. In Proceedings of International Symposium on Applied Corporate Computing (ISAAC'95), October 1995.

[HM96] M. Hitz and B. Montazeri. Chidamber and Kemerer's Metrics Suite: A Measurement Theory Perspective. IEEE Trans. Softw. Eng., 22(4):267–271, 1996.

[ISO03] ISO. ISO/IEC 9126-3 "Software engineering - Product Quality - Part 3: Internal metrics", 2003.

[Köl08] Universität Köln. `http://www.uni-koeln.de/phil-fak/fs-psych/serv_pro/skripte/meth/Statistik1.pdf`, 7 2008.

[LH93] W. Li and S. Henry. Maintenance Metrics for the Object Oriented Paradigm. Software Metrics Symposium, 1993. Proceedings., First International, pages 52–60, 1993.

[Lin07] Rüdiger Lincke. Validation of a Standard- and Metric-Based Software Quality Model – Creating the Prerequisites for Experimentation. PhD thesis, School of Mathematics and Systems Engineering, Växjö University, 2007.

[LL06] R. Lincke and W. Löwe. Foundations for Defining Software Metrics. In Proceedings of the 3rd International Workshop on Metamodels, Schemas,

Grammars, and Ontologies for Reverse Engineering (ATEM), Genoa, Italy, October 2006.

[Mar94] R. Martin. OO Design Quality Metrics – An Analysis of Dependencies (position paper). In Workshop Pragmatic and Theoretical Directions in Object-Oriented Software Metrics, OOPSLA'94, oct 1994.

[MH99] T. Mayer and T. Hall. A Critical Analysis of Current OO Design Metrics. Software Quality Control, 8(2):97–110, 1999.

[Mue08] Fachhochschule Muenster. Schiefe und woelbung. `https://www.fh-muenster.de/fb12/downloads/personen/wellmann/9_schiefe%_und_w__lbung.pdf`, 9 2008.

[Nie08] Wolfgang Niemeier. Ausgleichungsrechnung, statistische Auswertemethoden, 2.Auflage. de Gruyter, 2008.

[PDKR08] Institut für Arbeits-und Umweltmedizin PD Dr. Katja Radon, Universität München. `http://arbmed.klinikum.uni-muenchen.de/sem_rad_tg5_flussdiag_statistik_0506.pdf`, 9 2008.

[PXI08] Derek Robert Price, Ximbiot, and Free Software Foundation Inc. Cvs. `http://www.nongnu.org/cvs/`, 9 2008.

[SK03] R. Subramanyam and M. S. Krishnan. Empirical Analysis of CK Metrics for Object-Oriented Design Complexity: Implications for Software Defects. IEEE Trans. Softw. Eng., 29(4):297–310, 2003.

[Tig08] Tigris.org. Svn. `https://subversion.tigris.org`, 9 2008.

[Wik08] SourceForge Wiki. `alexandria.wiki.sourceforge.net`, 7 2008.

[www08] www.statistics4u.com. Grundlagen der statistik. `http://www.statistics4u.com/fundstat_germ/cc_corr_spearman.html`, 7 2008.

[Yue08] Wang Yue. Implementation of a software extraction process. School of Mathematics and Systems Engineering, 5 2008.

[Zhe08] Yan Zheng. Sourceforge metadata export into a database. School of Mathematics and Systems Engineering, 5 2008.