Freie Universität Berlin

Defektabschätzung

- Malgorzata Wojciechowska
- Institut für Informatik
- FU Berlin

- 03.04.2006

Erste Untersuchung:

1. “Quantitative Analysis of Faults and Failures in a Complex Software System”

Niclas Ohlsson, PhD
Dep. of Computer and Information Science
at Linköping University
in Sweden

Norman E. Fenton
Professor of Computer Science
Computer Science Department,
Faculty of Informatics and Mathematical Sciences
Queen Mary University of London

Methoden

Verteilung der Module nach Größe

LOC	Release n	Release n+1
<1000	23	26
1001-2000	58	85
2001-3000	37	73
3001-4000	15	38
4001-5000	6	16
5001-6000	0	6
>6000	1	2
Total	140	246

Methoden

Freie Universität Berlin

- Die Defektdaten wurden während vier unterschiedlicher Phasen gesammelt:
  - Funktionstest (FT)
  - Systemtest (ST)

  } Testdefekte

  - Die ersten 26 Wochen mit einer bestimmten Anzahl an Betriebstests (SI)
  - Das erste Jahr des Betriebs (OP)

  } Betriebsdefekte

Methoden

Zusammenfassung der entdeckten Anzahl von Defekten in jeder Testphase:

	pre-release faults		post-release faults	
Release	Function test	System test	Site test	Operation
n (sample size 140 modules)	916	682	19	52
n+1 (sample size 246 modules)	2292	1008	238	108

Freie Universität Berlin

2. "The Top Ten List: Dynamic Fault Prediction"

Richard C. “Ric” Holt
Professor,
Software Engineering,
School of Computer Science,
University of Waterloo

Ahmed E. Hassan
PhD Thesis,
School of Computer Science,
Faculty of Mathematics,
University of Waterloo

Übersicht der untersuchten Systeme

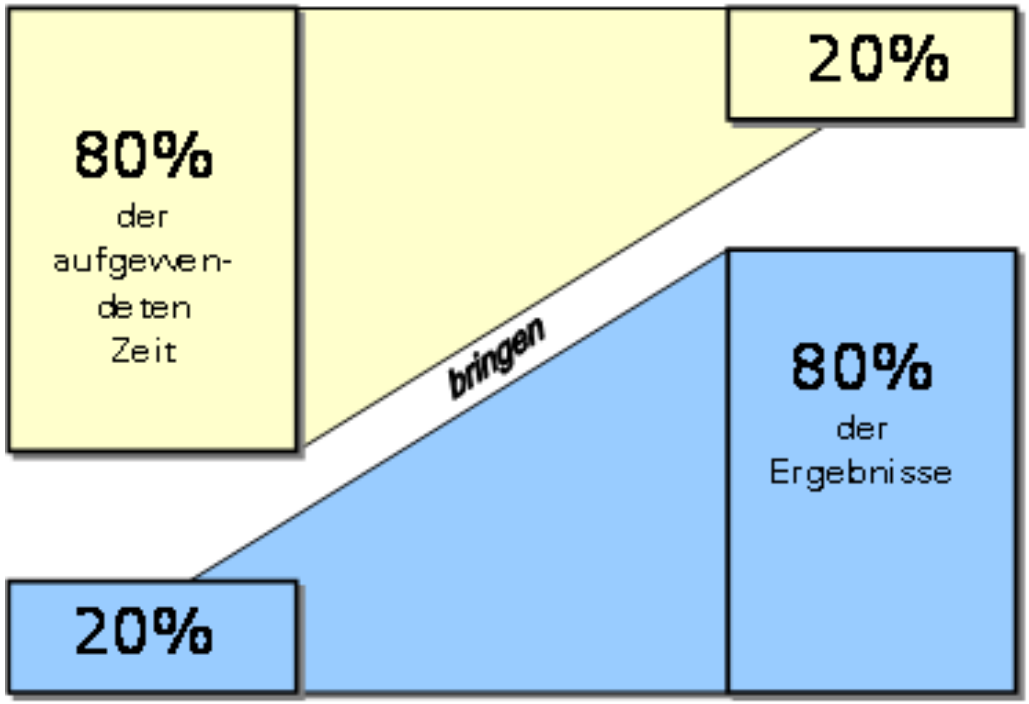
Application Name	Application Type	Start Date	Subsys. Count	Faults	Prog. Lang.
NetBSD	OS	21 March 1993	393	2451	C
FreeBSD	OS	12 June 1993	182	3264	C
OpenBSD	OS	18 Oct 1995	401	1015	C
Postgres	DBMS	9 July 1996	104	1401	C
KDE	Windowing System	13 April 1997	167	6665	C++
Koffice	Productivity Suite	18 April 1998	259	5223	C++

- Hypothesen in Bezug auf das Pareto-Prinzip über Defekt - und Versagensverteilung

Was ist das Pareto-Prinzip?

Das Pareto-Prinzip (80%-20%-Prinzip) besagt, dass:

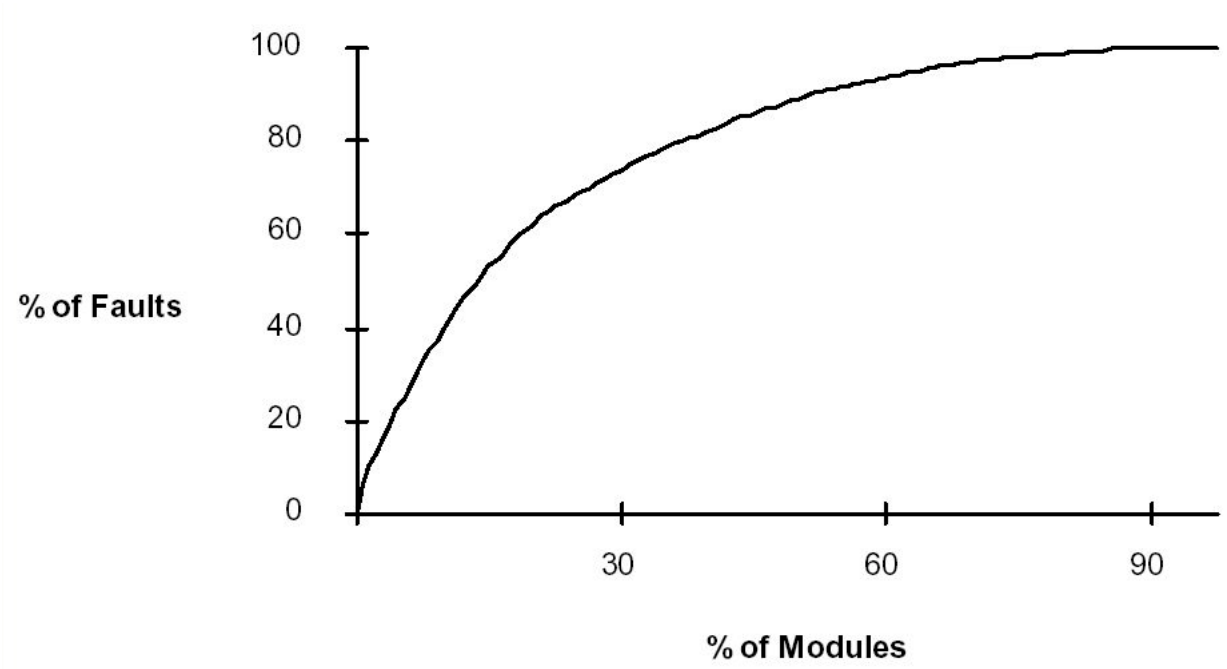
- mit 20% des Aufwandes bereits 80% der Leistung erbracht werden,
- für die letzten 20% der Leistung 80% des Aufwandes erforderlich sind.



Copyrights http://www.teachsam.de/arb/zeitmanagement/zeitmanag_2_3_4.htm

Das Pareto-Prinzip

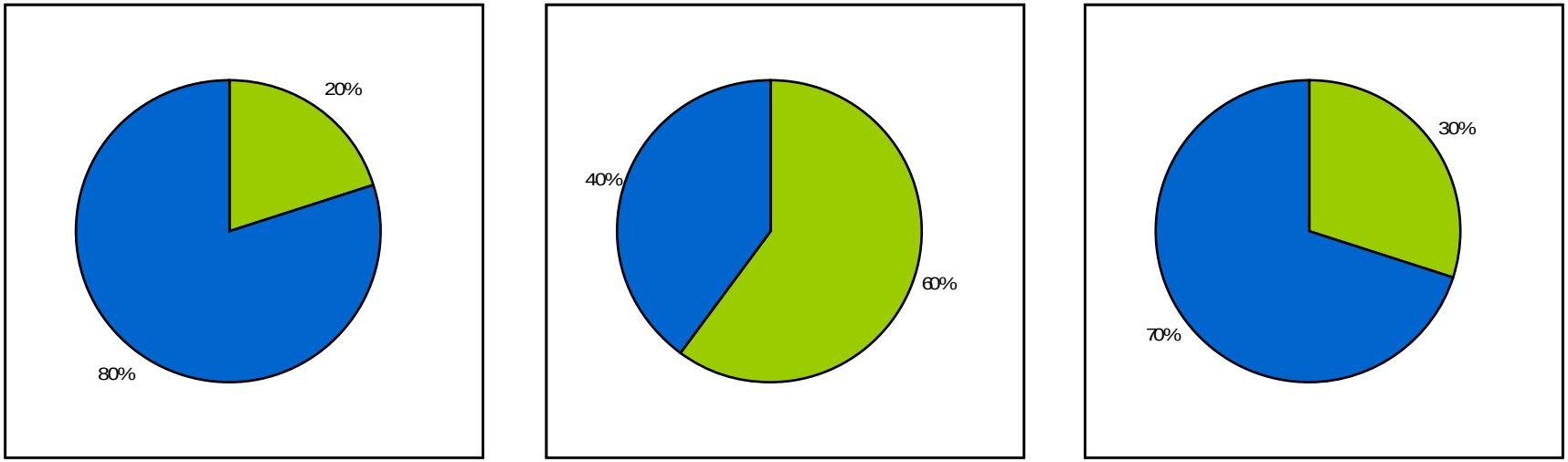
- **Hypothese 1a:** Die überwiegende Mehrheit der Defekte, die während der Vor-Veröffentlichungstests entdeckt werden, konzentriert sich in einigen wenigen Modulen.



Das Pareto-Prinzip

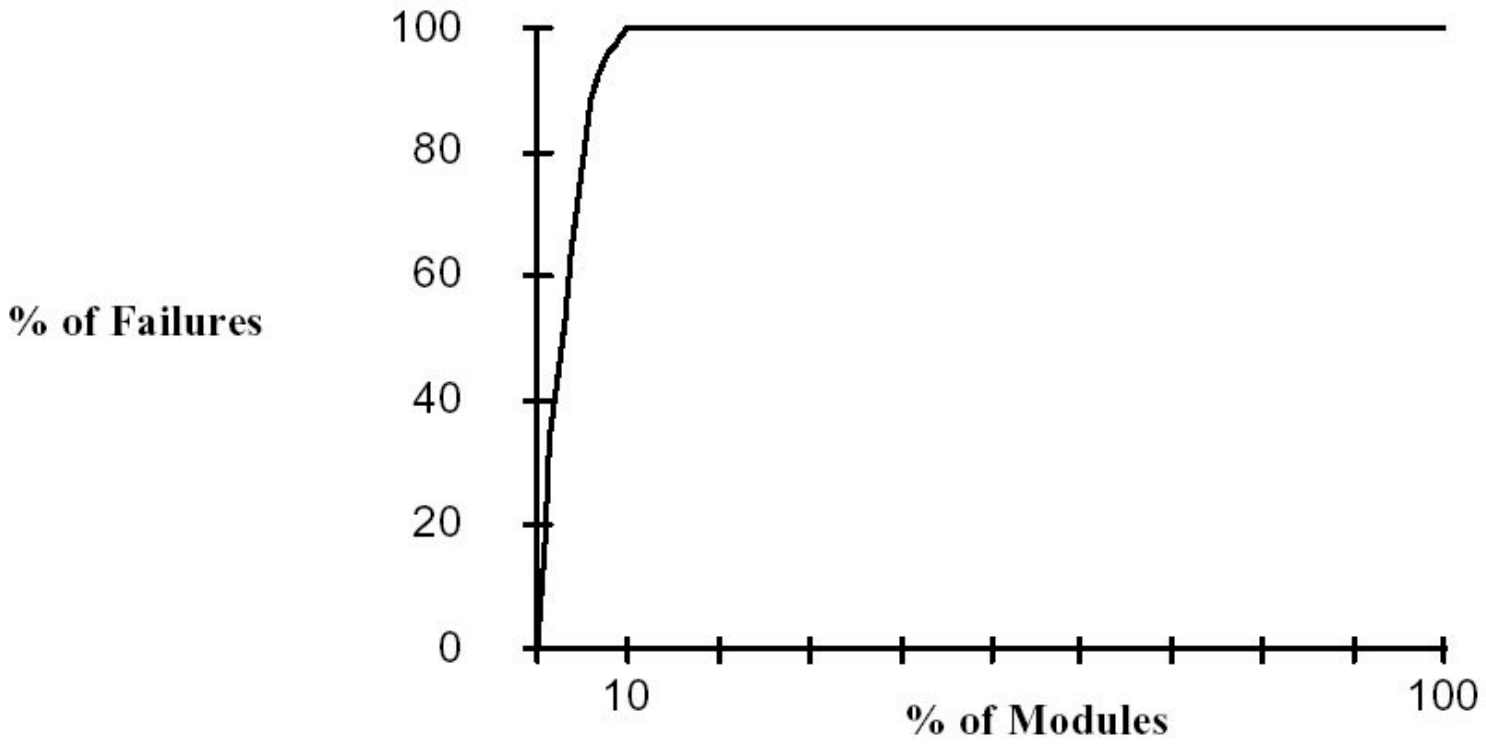
- **Hypothese 1b:** Die Module, in denen sich die Defekte, die während der Vor-Veröffentlichungstests entdeckt werden, konzentrieren, machen gleichzeitig den größten Teil des gesamten Codes aus.

20% der **Module** beinhalten 60% der **Defekte** und machen 30% der **Systemgröße** aus



Das Pareto-Prinzip

- **Hypothese 2a**: Eine kleine Anzahl von Modulen enthält die meisten Betriebsdefekte (gemeint sind Versagen wie man sie in Phasen SI und OP beobachtet hat).



Das Pareto-Prinzip

Freie Universität Berlin

- **Hypothese 2b**: Eine kleine Anzahl von Modulen enthält die meisten Betriebsdefekte, weil diese Module den größten Teil des Codes ausmachen.

Antithese:

Die meisten Betriebsdefekte werden durch Defekte in kleinen Teilen des Codes verursacht.

Ergebnis:

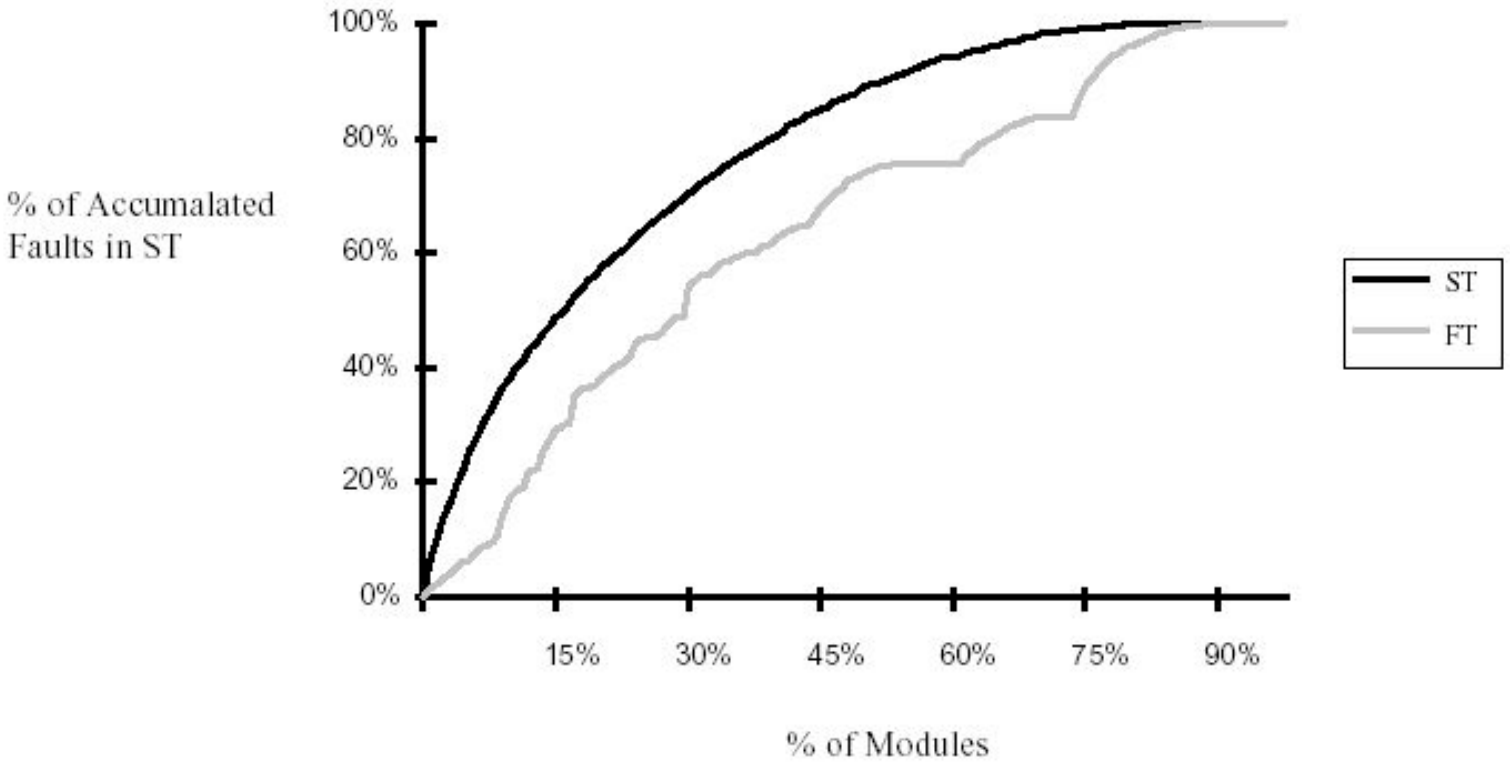
100% der Betriebsdefekte befanden sich in Modulen mit nur 12% der Systemgröße

- Hypothesen in Bezug auf den Gebrauch von früheren Defektdaten dienen der Vorhersage von Defekten und Versagen

  (im Modullevel)

Gebrauch von früheren Defektdaten

- **Hypothese 3**: Ein häufigeres Auftreten von Defekten in Funktionstests (FT) ist verknüpft mit einem häufigeren Auftreten von Defekten in Systemtests (ST).



Gebrauch von früheren Defektdaten

Freie Universität Berlin

- **Hypothese 4**: Ein häufigeres Auftreten von Defekten während der Vor-Veröffentlichungstests ist verknüpft mit einem häufigeren Versagen im Betrieb.

Ergebnisse:

Für die **Veröffentlichung n**:
93% der Defekte während Vor-Veröffentlichungstests treten in Modulen, die keine Betriebsdefekte haben, auf.

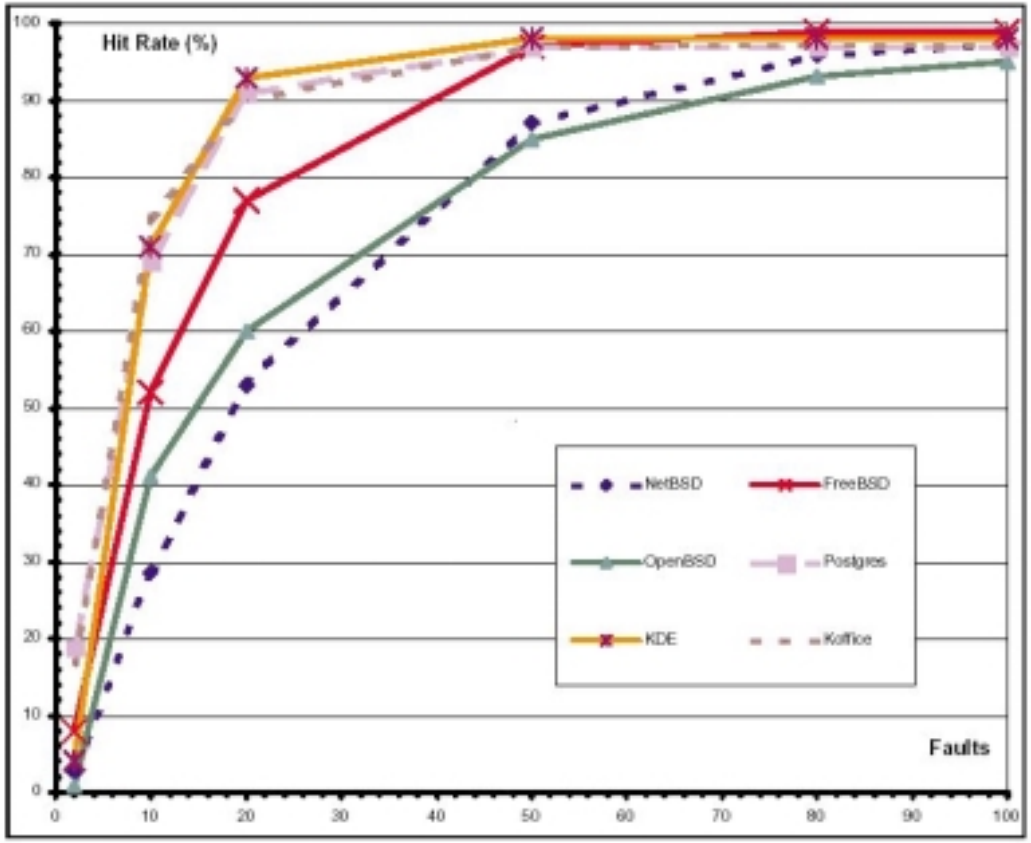
Für die **Veröffentlichung n+1**:
77% der Vor-Veröffentlichungsdefekte treten in Modulen, die keine Betriebsdefekte nachweisen, auf.

- Hypothesen in Bezug auf Häufigkeit und Zeitspanne, in der Veränderung der Subsysteme vorgenommen werden

Häufigkeit und Zeitspanne der Veränderung der Subsysteme

Freie Universität Berlin

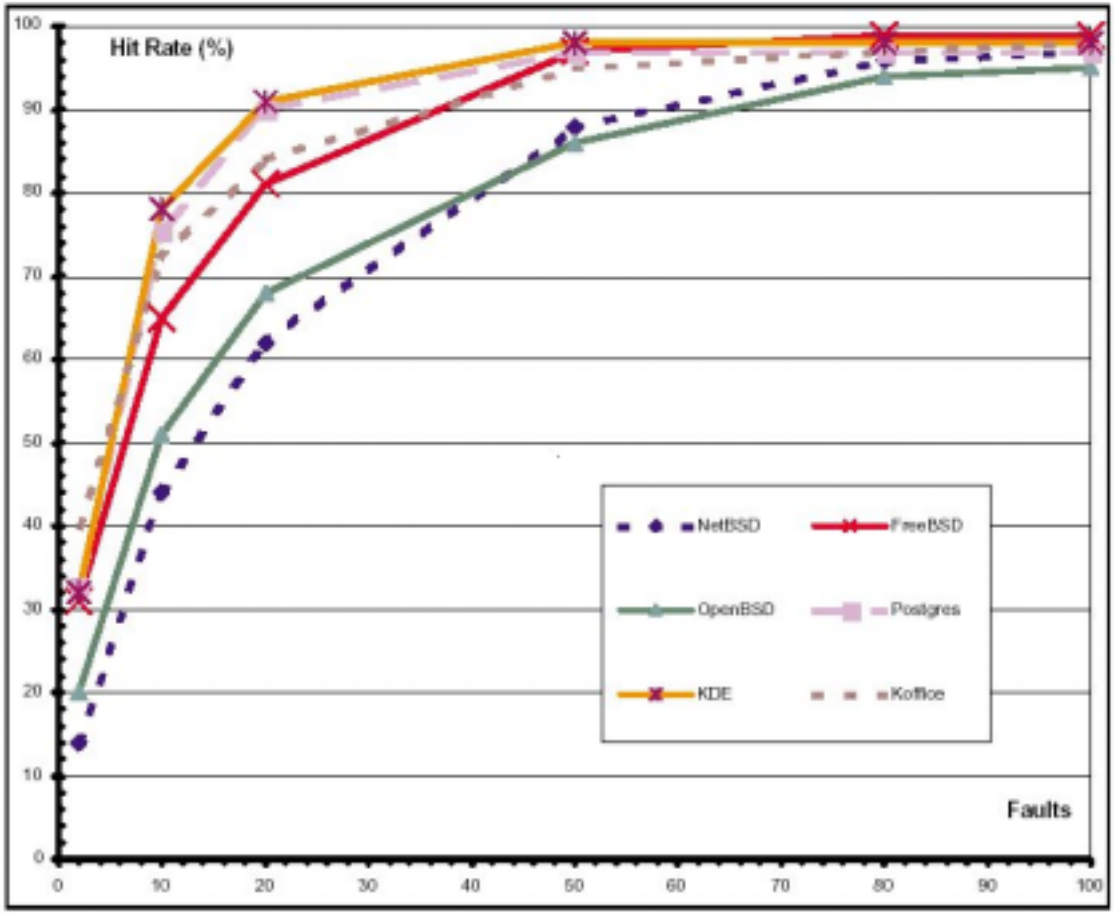
- **Hypothese 5:** Je öfter die Subsysteme verändert wurden, desto höher ist die Wahrscheinlichkeit, dass sie Defekte beinhalten.



Häufigkeit und Zeitspanne der Veränderung der Subsysteme

Freie Universität Berlin

- **Hypothese 6:** Erst vor kurzem veränderte Subsysteme weisen eine erhöhte Defektanfälligkeit auf.



- Hypothesen über Größenmaße in der Defektvorhersage

Größenmaße in der Defektvorhersage

Freie Universität Berlin

- **Hypothese 7**: Aus einfachen Größenmaßen (Lines of Code) des Moduls lässt sich eine Vorhersage über die Defektanfälligkeit ableiten.

Hypothese 7a: Kleine Module sind weniger defektanfällig als größere Module.

Hypothese 7b: Ein Größenmaß eines Moduls (Lines of Code) ist ein guter Indikator für die Anzahl von Defekten, die in Vor- (Nach) Veröffentlichungstests entdeckt werden.

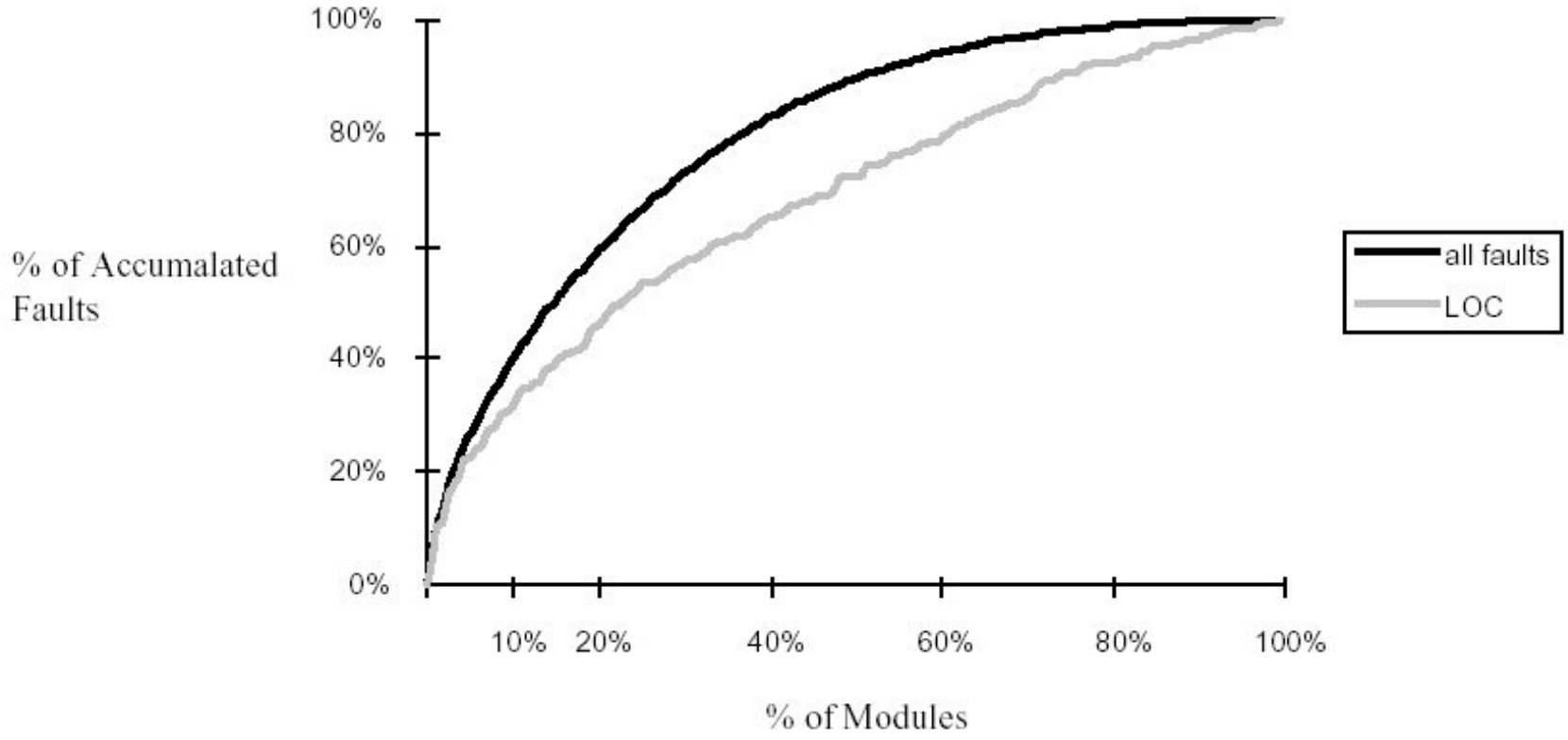
Hypothese 7c: Eine Größenmaße (Lines of Code) ist ein guter Indikator für die Defektdichte eines Moduls, die im Rahmen einer Vor- (Nach) Veröffentlichungsstudie festgestellt wird.

Größenmaße in der Defektvorhersage

Freie Universität Berlin

Prozentsatz der absoluten Defektzahl.

Ordnung der Module nach LOC-Größe für Veröffentlichung n+1.



- **Hypothese 8**: Das Komplexitätsmaß ist ein besserer Indikator für defekt- und versagensanfällige Module als ein einfaches Größenmaß.

Ergebnisse:

Für die **Veröffentlichung n+1**

• Die komplexesten Module, die defektanfälliger in der Vor-Veröffentlichung waren, haben dann in der Nach-Veröffentlichung kaum noch Defekte.

• Verantwortlich dafür ist die Art der Verteilung der Testanstrengungen über die Module.

• Die Module die komplexer zu sein scheinen, werden mit größerer Vorschicht behandelt als die einfacheren Module

- Hypothesen zur Beziehung zwischen Defektdichte, Qualität und Benchmarking-Daten

Qualität und Benchmarking

Freie Universität Berlin

- **Benchmarking** - (bedeutet übersetzt „Maßstäbe setzen“)
  - Ein Konzept, um Verbesserungsmöglichkeiten durch den Vergleich von Leistungsmerkmalen zu finden.
  - Das Ziel des Benchmarking ist die Schwächen eines Unternehmens und seiner Prozesse (oder eines Computers und seiner Programme) durch Vergleich mit anderen Unternehmen, Prozessen, (Computern oder Programmen) aufzudecken und die Leistungsfähigkeit zu erhöhen.

Freie Universität Berlin

- **Hypothese 9**: Es ist davon auszugehen, dass die Defektdichte in den Testphasen und der nachfolgenden Betriebsphase bei Hauptversionen eines Softwaresystems nahezu konstant bleibt.

Defektdichte in vier Test- und Betriebsphasen

	FT	ST	SI	OP
Rel n	3.49	2.60	0.07	0.20
Rel n+1	4.15	1.82	0.43	0.20

Freie Universität Berlin

- **Hypothese 10**: Softwaresysteme, die unter vergleichbaren Bedingungen produziert werden, weisen in der Regel eine etwa gleiche Defektdichte sowohl in der Test- als auch in der Betriebsphase auf.

Defektdichte während Vor-Veröffentlichung und Nach-Veröffentlichung

	Pre-release	Post-release	All
Rel n	6.09	0.27	6.36
Rel n+1	5.97	0.63	6.60

Fazit

Es wurde bewiesen, dass:

- die Mehrheit der Vor-Veröffentlichungsdefekte sich in einigen wenigen Modulen konzentriert,
- eine kleine Anzahl der Module die meisten Betriebsdefekte enthält,
- die Häufigkeit der Veränderung der Module Einfluss auf die Defektanfälligkeit der Module hat,
- vor kurzem veränderte Module defektanfälliger sind,

Fazit

Freie Universität Berlin

- ein Indikator für die Anzahl der Defekte die einfache Größenmaße (LOC) des Moduls ist,
- die Defektdichte bei Hauptversionen eines Softwaresystems in den Test- und Betriebsphasen konstant bleiben,
- unter vergleichbaren Bedingungen produzierte Softwaresysteme etwa gleiche Defektdichte in der Test- und Betriebsphase aufweisen.

Fazit

Es wurde nur unzureichend bewiesen, dass:

- die Häufigkeit der Defekte in Funktionstests mit der Häufigkeit der Defekte in Systemtests verknüpft ist.

Fazit

Es wurde widerlegt, dass:

- die einigen wenigen Module, in denen sich die Mehrheit der Vor-Veröffentlichungsdefekte konzentriert, den größten Teil des gesamten Codes ausmachen,

- eine kleine Anzahl von Modulen, die die meisten Betriebsdefekte enthält, den größten Teil des Codes ausmacht.

- die Häufigkeit der Defekte in Vor-Veröffentlichungstests mit der Häufigkeit der Versagen in Betriebstests verknüpft ist,

- ein Komplexitätsmaß ein besserer Indikator für defekt- und versagensanfällige Module als ein einfaches Größenmaß ist.

Vielen Dank!