# Fehlerprognosen zur Qualitätssteigerung in frühen Entwicklungsphasen der Automobilindustrie

Matthias Wieman, Jörg Gericke

Siemens AG
Corporate Research & Technologies
CT PP 6
Otto-Hahn-Ring 6
81730 München
joerg.gericke@siemens.com

**Abstract:** Die strukturelle Komplexität hybrider Systeme wuchs in den letzten Jahren stark an und führte zu einem deutlichen Anstieg der Fehlerzahl in der implementierten Software. Besonders evident ist dies in der Automobilindustrie. Qualitätssicherungsmaßnahmen in frühen Phasen des Entwicklungszyklus sollen dieser Entwicklung kosteneffektiv entgegenwirken. Fehlerprognosen können die komplexitätsabhängig zu erwartende Fehleranzahl von Softwaremodulen abschätzen. Anhand der Prognosen kann der Entwurf besonders komplexer und somit stärker fehlerbehafteter Softwaremodule überdacht werden, mit dem Ziel die Komplexität und dadurch die zu erwartende Fehleranzahl zu verringern. Es wird ein zielführendes Fehlerprognosemodell vorgestellt, das unter Verwendung von Softwaremetriken und statistischen Verfahren aufgestellt worden ist.

## 1 Einleitung und Motivation

Der Markt eingebetteter Systeme in der Automobilindustrie ist in den letzten Jahren stark gewachsen [Be06]. Ein Grund hierfür ist, dass Innovationen hauptsächlich (80-90%) durch Software realisiert werden [Be06]. Dieser intensive Einsatz von Software birgt jedoch Probleme in der Überschaubarkeit und Beherrschbarkeit von Systemen [Em05, Eq07, Li02]: Fahrzeugherstellerübergreifend soll dadurch die Zahl der durch Software verursachten Autopannen auf zwei drittel ansteigen [Dp03]. Dieser prognostizierten Entwicklung soll durch qualitätssteigernde Verfahren basierend auf Fehlerprognosen kosteneffektiv begegnet werden.

Im Folgenden wird daher ein statistischer Fehlerprognoseansatz empirisch im Automobilumfeld untersucht. Die Fehlerprognose kann frühzeitig anhand von Designdokumenten, später auch anhand von Quelltext Anzahl und Art der Fehler eines Systems abschätzen. Die Anzahl der Fehler kann für Testaufwandsplanungen zur Ableitung von Zeitaufwand und Anzahl benötigter Tester herangezogen werden. Die Fehlerart – beispielsweise 17 Fehler aufgrund von Vererbungsbeziehungen – kann zur Identifikation von fehlerverursachenden Systemeigenschaften herangezogen werden, die im Anschluss zur Fehlervermeidung entsprechend optimiert werden sollten.

## 2 Das Fehlerprognosemodell

Die untersuchte Fehlerprognose basiert auf der sogenannten *strukturellen und kognitiven Komplexität* [MH78], die aus einzelnen *Komplexitätsaspekten* wie Anzahl geerbter Methoden oder Anzahl öffentlicher Attribute abgeleitet wird. Sogenannte *(Software-) Metriken* sollen möglichst alle Komplexitätsaspekte vollständig vermessen. Besondere Sorgfalt wurde auf die Überdeckung aller Komplexitätsaspekte objekt-orientierter Software gelegt, da die Wahl der Metriken großen Einfluss auf die Genauigkeit der Fehlerabschätzung hat [Ne04]. Ne92 zu überdeckende Komplexitätsaspekte für objektorientierte Software wurden von Neumann [Ne04] unter Anwendung der *GQM-Methode* (Goal-Question-Metric) [GHW95] identifiziert. Diese Komplexitätsaspekte werden nahezu vollständig von einer von Lionel Brand vorgeschlagenen und in der Praxis bewährten Menge von 18 Metriken überdeckt [BDM97]. Vervollständigt wurde die Überdeckung aller Komplexitätsaspekte durch die Hinzunahme von vier weiteren Metriken. Insgesamt werden sechs quantitative Metriken, drei Komplexitätsmetriken sowie dreizehn Kopplungsmetriken verwendet.

**Principal Component Analysis:** Die 22 ausgewählten Metriken können korrelieren. Daher werden aus den abhängigen Metriken mit Hilfe der sogenannten *Hauptkomponentenanalyse (engl. Principal Component Analysis*, *PCA)* [Du89] unabhängige *virtuelle Metriken* berechnet. Eine virtuelle Metrik kann eine individuelle Interpretation wie „Vererbungskomplexität“ besitzen und wird auch als *Principal Component (PC)* [Jo02] bezeichnet. Die multivariate Hauptkomponentenanalyse wurde in der Literatur [FN05, MK90, MK92, Ne92] kritisiert, da sich in Untersuchungen eine Vielzahl von Metriken oft auf wenige (im Allgemeinen zwischen drei und sechs) virtuelle Metriken reduzieren ließ [MK90, MK92, Ne92]. Unserer Meinung nach sollte diese Tatsache jedoch nicht zur Interpretation führen, dass eine derart geringe Anzahl von Metriken ausreichend sei, das System und somit sämtliche Komplexitätsaspekte vollständig zu vermessen.

**Multivariate Adaptive Regression Splines:** Erfahrungswerte aus vorangegangenen Projekten werden zur Optimierung und Kalibrierung der Fehlerabschätzung herangezogen. Mit Hilfe der *Multivariate Adaptive Regression Splines (MARS)* [Fr91] werden die Werte der PCs in Bezug zu den bereits beobachteten Fehleranzahlen *und* Fehlertypen (z.B. Fehler aufgrund Vererbung, Fehler durch algorithmische Komplexität) gesetzt, wobei die Fehleranzahlen und Fehlertypen die abhängigen Variablen bilden. Die PCs der einzelnen Projekte sind die unabhängigen Variablen.

Multivariate Regressionsmethoden stehen in der Kritik [FN05], keine konkreten Optimierungsmaßnahmen aufzuzeigen, um die Anzahl prognostizierter Fehler zu senken. Da unser Fehlerprognosemodell auf detaillierten Fehlerdaten mit Fehlerursachen basiert, liefert unser Fehlerprognosemodell neben der zu erwartenden Fehleranzahl die zu erwartenden Fehlertypen – beispielsweise 30 Fehler aufgrund von Vererbungskomplexität sowie 15 Fehler aufgrund algorithmischer Komplexität.

## 3. Anwendung

**a) Anwendung während Anforderungsanalyse und Entwurf (Abbildung 3):** Die

Anwendung von Metriken auf Anforderungs- und Entwurfsdokumente wie Klassendiagramme und Zustandsautomaten ermöglicht eine Anwendung in frühen Entwicklungsphasen. Der Vergleich verschiedener Softwaremodule kann komplexe und fehlerträchtige Softwaremodule identifizieren. Diese können somit frühzeitig während Anforderungsdefinition und Entwurf hinsichtlich struktureller und kognitiver Komplexität analysiert und optimiert werden. Hierfür liefert die Fehlerprognose Hinweise zur prognostizierten Fehlerart. Die Anzahl der Entwurfs- und daraus folgenden Implementierungsfehler soll so verringert werden.

**b) Anwendung während der Testphasen (Abbildung 4):** Zur Effizienzsteigerung können im Systemtest besonders fehlerträchtige Softwaremodule gezielt getestet werden. Außerdem wird die Planung des notwendigen Testaufwands (Zeit und Personal) durch die Fehlerabschätzung unterstützt. Weiterhin ermöglicht eine realistische Abschätzung der Gesamtfehleranzahl eine präzisierte Definition eines Testabbruchkriteriums, wie z.B. „85% der Fehler sollen gefunden werden“.

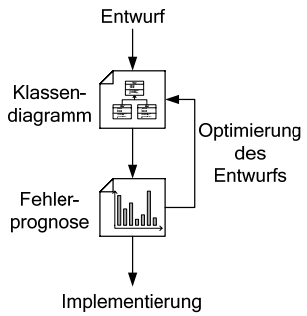


Abbildung 3. Iterative Entwurfsänderungen

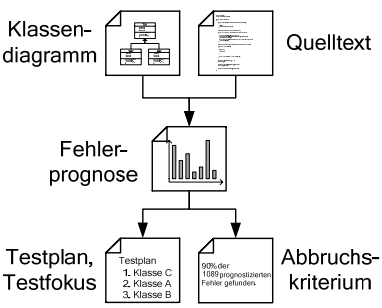


Abbildung 4. Fehlerprognose im Test

## 3. Evaluierung durch praktische Anwendung

Die beschriebene Fehlerprognose wurde als Teil der bei der Siemens AG entwickelten *Generic and Automated Test Environment (GATE)* implementiert. Das berechnete Fehlerprognosemodell fand in mehreren industriellen Projekten der Automobilindustrie Anwendung und wird anhand eines exemplarisch ausgewählten Projektes diskutiert. Im vorliegenden Softwaresystem (ca. 500 MB Quelltext) wurden nahezu 10.000 Fehler im Zeitraum von 2,5 Jahren gefunden. Hier wurde die Fehlerprognose ausschließlich zur Testplanung (s. Kapitel 2b) und nicht zur Komplexitätsreduktion (s. Kapitel 2a) eingesetzt. So ist die Bewertung der Fehlerprognosegenauigkeit ohne Seiteneffekte durch Optimierungsmaßnahmen möglich. Wenig verwunderlich ist, dass die Prognose der sich zum jeweiligen Zeitpunkt im System befindlichen Fehler wird mit zunehmender Datenbasis (Anforderungs- und Entwurfsdokumente, Quelltext) späterer Projektphasen präziser wird. In Abbildung 5 sind die über alle Projekte gemittelten Prognosefehler angetragen. Im konkreten Projekt liegt der Abschätzungsfehler nach Implementierung bei 6%. Der reale Abschätzungsfehler wäre geringer unter der Annahme, dass nicht alle Fehler im System gefunden wurden und die reale Fehleranzahl somit über 9.800 Fehlern läge. Diese Annahme kann erfahrungsgemäß als realistisch bewertet werden.

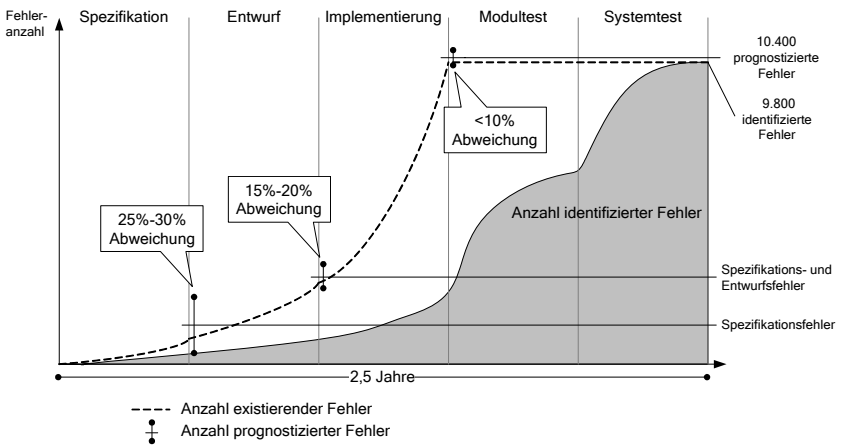


Abbildung 5. Fallstudie eines Projektes

## 4. Zusammenfassung

Ziel der Arbeit ist die Untersuchung eines multivariaten Fehlerprognoseverfahrens im Hinblick auf die praktische Anwendbarkeit in der Automobilindustrie. Untersuchungen anhand mehrerer realer industrieller Softwaresysteme ergaben, dass der durchschnittliche Prognosefehler nach der Implementierung unter 10% liegt und somit als vergleichsweise gut bewertet werden kann. Der Prognosefehler während der Entwurfsphase war mit 15% bis 20% für die Identifikation besonders kritischer Softwaremodule ausreichend. Die Reduktion des Schätzfehlers ist dennoch ein Punkt zukünftiger Arbeiten.

## Literaturverzeichnis


[BDM97] L. Briand, P. Devanbu, W. Melo. An investigation into coupling measures for C++. 19th International Conference on Software engineering. ACM, NY, USA. 1997

[Be06] K. Bender. Embedded Systems-qualitätsorientierte Entwicklung. Springer. 2006.

[Dp03] dpa Meldung vom 30.09.2003 zur 33. Jahrestagung der GI

[Du89] G. Dunteman. Principal Component Analysis. Sage Publications, 1989

[Em05] B. Emaus. Hitchhiker's Guide to the Automotive Embedded Software. 2nd International ICSE workshop on Software Engineering for Automotive Systems. St. Louis, 2005

[Eq07] Methoden zur Unterstützung der entwicklungsbegleitenden Qualitätssicherung (EQUAL). http://www.embedded-quality.de/

[FN05] N. Fenton, M. Neil. A Critique of Software Defect Prediction Models. Machine Learning Applications In Software Engineering. World Scientific. 2005

[Fr91] J. Friedman, Multivariate Adaptive Regression Splines. Annals of Statistics, 1991

[GHW95] E. Gresse, B. Hoisl, J. Wust. A Process Model for GQM-Based Measurement. Software Technology Transfer Initiative (STTI), University of Kaiserslautern. 1995

[Jo02] I. Jolliffe, Principal Component Analysis. Springer. 2002

[Li02] P. Liggesmeyer. Software Qualität. Spektrum Akademischer Verlag. 2002

[MH78] H. Mandl, G. L. Huber. Kognitive Komplexität. Göttingen. Hogrefe, 1978

[MK90] J. Munson, T. Khoshgoftaar. Regression modelling of software quality: empirical investigation. Journal of Electronic Materials, Vol 19. No. 6. pp 106-114. The Metals. 1990

[MK92] J. Munson,. T. Khoshgoftaar. The detection of fault-prone programs. SE IEEE. Vol. 18. No. 5. 1992

[Ne04] R. Neumann. A categorization for object oriented software metrics in fault prediction. Software Metrics European Forum, January 2004.

[Ne92] M. Neil. Statistical Modelling of Software Metrics. South bank University. 1992