Steffen Herbold
(herbold@cs.uni-goettingen.de)
ist Postdoktorand an der Georg-August-Universität Göttingen, wo er auch promoviert hat. Er beschäftigt sich im Rahmen seiner Forschung unter anderem mit Fehlervorhersagemodellen, wie man sie anwenden kann und warum diese häufig nicht das tun, was man von ihnen erwartet.

Jens Grabowski
(grabowski@informatik.uni-goettingen.de)
ist Professor für Softwaretechnik. Er forscht und lehrt am Institut für Informatik der Georg-August-Universität Göttingen. Seine Forschungsarbeiten konzentrieren sich zurzeit auf Fragestellungen zur Qualitätssicherung und Evolution von Software.

Software-Fehlervorhersage
Intelligente Qualitätssicherung durch statistische Methoden

Ein wichtiger Kostenfaktor der Softwareentwicklung ist die Qualitätssicherung. Obwohl unentdeckte Fehler hohe Kosten verursachen können, stehen häufig nur begrenzte Ressourcen für die Qualitätssicherung zur Verfügung, die daher effizient genutzt werden müssen. Hier kann die Software-Fehlervorhersage helfen: Mittels statistischer Methoden werden Prognosen erstellt, in welchen Softwaremodulen sich wahrscheinlich Fehler befinden. Die so identifizierten Module können dann besonders intensiv getestet werden.

Das Ziel der Qualitätssicherung ist es sicherzustellen, dass ein System im Produktivbetrieb möglichst fehlerfrei arbeitet. Mit unbegrenzten Ressourcen ist es möglich, das komplette System intensiv zu testen. Da dies jedoch nicht möglich ist, muss der Aufwand für die Qualitätssicherung gezielt eingesetzt werden, sodass nicht jeder Teil eines Systems mit gleicher Intensität getestet wird. Die Vorhersage von Fehlern in Software ist ein mächtiges Werkzeug zur Steuerung des Testaufwands. Wenn man bereits vor der Priorisierung wüsste, in welchen Modulen sich mit hoher Wahrscheinlichkeit Fehler befinden, kann man dieses Wissen berücksichtigen und Ressourcen effizient nutzen.

Fehlervorhersage im Qualitätsmanagement

Qualitätsmanagement umfasst nach DIN EN ISO 8402 „Tätigkeiten der Gesamtführungsaufgabe, welche die Qualitätspolitik, Ziele und Verantwortung festlegen sowie diese durch Mittel wie Qualitätsplanung, Qualitätslenkung, Qualitätssicherung und Qualitätsverbesserung im Rahmen des Qualitätsmanagementsystems verwirklichen“. Unter Qualitätslenkung versteht man nach EN ISO 9000 Arbeitstechniken und Tätigkeiten sowohl zur Überwachung eines Prozesses als auch zur Beseitigung von Ursachen nicht zufriedenstellender Ergebnisse.

Bei begrenzten Ressourcen muss die Qualitätslenkung dafür sorgen, dass sich Qualitätssicherungsmaßnahmen auf kritische Systemteile konzentrieren. Zu den kritischen Systemteilen gehören naturgemäß Module oder Funktionen, die mit hoher Wahrscheinlichkeit Fehler beinhalten, sodass sich zur Erkennung kritischer Sys-

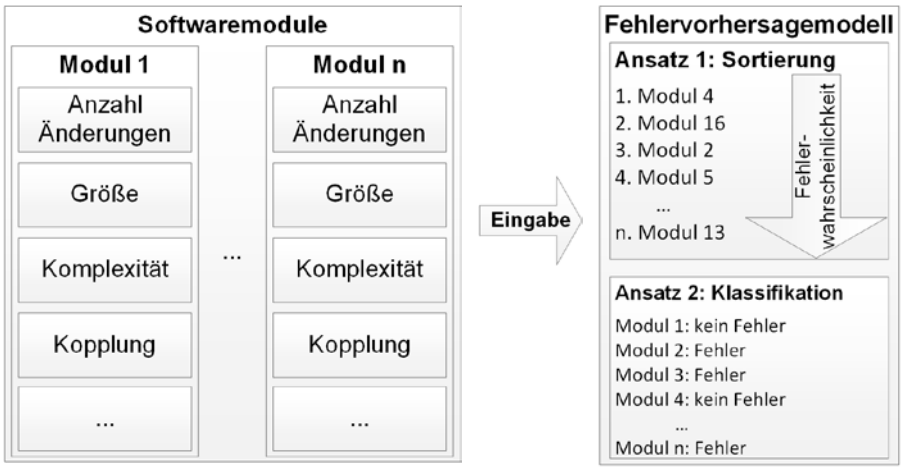

Abb. 1: Fehlervorhersage für Module

temteile der systematische Einsatz von Methoden zur Fehlervorhersage anbietet.

Insbesondere der Softwaretest kann als wichtigste Qualitätssicherungsmaßnahme von den Ergebnissen der Fehlervorhersage stark profitieren. Durch eine geeignete Planung lassen sich Ressourcen so verschieben, dass kritische Systemteile besonders intensiv und nach strengeren oder zusätzlichen Kriterien getestet werden können. Bei Regressionstests vor der Veröffentlichung neuer Releases oder nach der kurzfristig notwendigen Beseitigung schwerwiegender Fehler hat man zudem häufig zu wenig Zeit, um alle vorhandenen Tests durchzuführen. Die Fehlervorhersage kann hier ein zusätzliches Kriterium für die Auswahl beziehungsweise Priorisierung von Regressionstests liefern.

Der Einsatz von Methoden zur Fehlervorhersage sollte nicht spontan erfolgen, sondern muss in der Projektplanung berücksichtigt werden. Es bietet sich an, Zweck und Einsatz der Fehlervorhersage im Qualitätssicherungsplan zu definieren. Üblicherweise findet man in einem Qualitätssicherungsplan (z. B. nach IEEE Standard 730) auch die zur Anwendung kommenden Standards, Methoden, Konventionen und Metriken, notwendige Software Reviews und Tests sowie Verfahren zur Meldung von Problemen und die zugehörigen Korrekturmaßnahmen. Der testspezifische Einsatz der Fehlervorhersage sollte zudem im Testplan (z. B. nach IEEE Standard 829) beschrieben werden. Ein Testplan beinhaltet unter anderem die zu testenden Objekte, Test-Strategien, Kriterien für das Bestehen und Nicht-Bestehen von Tests, Kriterien für den Abbruch und die Wiederaufnahme von Tests, einen Testablaufplan und eine Risikoanalyse.

Fehlervorhersage für Entwickler

Fehlervorhersage ist nicht nur für das Qualitätsmanagement wichtig, sondern kann dem Softwareentwickler auch bei seiner täglichen Arbeit helfen. Insbesondere in Umgebungen, wo mehrere Entwickler an den gleichen Modulen Änderungen durchführen dürfen (engl. Collective Code Ownership), oder in der Softwarewartung, wo Module in der Vergangenheit mehrfach und von mehreren Entwicklern geändert worden sind, kann die Fehlervorhersage einem Entwickler wertvolle Hinweise für seine Arbeit geben. Je häufiger das von ihm bearbeitete Modul geändert worden ist, desto kritischer ist das Modul und desto mehr Aufwand muss in den Test und die Dokumentation der Änderungen investiert werden. Die Ergebnisse der Fehlervorhersage können zudem einen hohen Aufwand für die Qualitätssicherung einzelner Systemteile gegenüber einem Auftraggeber oder Vorgesetzten begründen.

Vorhersagemodelle

Doch wie funktioniert Fehlervorhersage überhaupt? Die wichtigste Unterscheidung bei der Fehlervorhersage ist, wofür Fehler vorhergesagt werden sollen: für die Module eines kompletten Projektes oder für einzelne Änderungen.

In Abbildung 1 wird der Ansatz der Fehlervorhersage für Module visualisiert. Jede Software besteht aus einer Menge von Modulen, wobei der Begriff Modul hier abstrakt gemeint ist und verschieden interpretiert werden kann. So können die Module zum Beispiel Klassen, Pakete, Quelltextdateien oder sogar Binärdateien sein. Module lassen sich durch Kennzahlen oder Metriken charakterisieren. Für die Fehlervorhersage sind jedoch nur ausgewählte Kennzahlen eines Moduls, wie die Größe, die Anzahl der vorgenommenen Änderungen, die Komplexität oder die Kopplungsstärke mit anderen Modulen, relevant. Diese ausgewählten Kennzahlen dienen als Eingabe für die Fehlervorhersage. Es gibt nun zwei Arten, wie ein Fehlervorhersagemodell für Module funktionieren kann. Der erste Ansatz sortiert die Module nach der berechneten Fehlerwahrscheinlichkeit. Im Beispiel in Abbildung 1 hat also das Modul 4 die höchste Fehlerwahrscheinlichkeit. Beim zweiten Ansatz wird jedes Modul einzeln in fehlerhaft oder fehlerfrei klassifiziert.

Bei der Fehlervorhersage für Änderungen (siehe Abbildung 2) werden Fehler für einzelne Änderungen statt für ein ganzes Modul vorhergesagt. Das grundsätzliche Vorgehen ist das gleiche wie bei der Fehlervorhersage für Module und basiert auf Kennzahlen. Die verwendeten Kennzahlen charakterisieren die vorgenommenen Änderungen an einem Modul. Typische Kennzahlen beschreiben für ein Modul die Anzahl der vorgenommenen Änderungen, die Art der Änderungen (z. B. Bugfix oder Refactoring) oder die Dauer von Änderungen. Diese Kennzahlen dienen dann als Eingabe für ein Fehlervorhersagemodell. Analog zur Vorhersage für Module gibt es auch bei der Fehlervorhersage für Änderungen zwei Ansätze: Beim Scoring wird eine Punktzahl vergeben, die die Wahrscheinlichkeit eines Fehlers in der Änderung widerspiegelt. Bei der Klassifikation wird für die Änderung direkt die Aussage getroffen, ob sie fehlerhaft oder fehlerfrei ist.

Sowohl für die Fehlervorhersage für Module wie auch für die Fehlervorhersage für Änderungen werden Vorhersagemodelle benötigt. Vorhersagemodelle können hierbei zum Beispiel in Form von einfachen Statistiken oder komplexeren Modellen (zum Beispiel Entscheidungsbäume oder Support Vector Machines) vorliegen. Techniken zur Erzeugung von Vorhersagemodellen basieren üblicherweise auf statistischen Methoden oder maschinellem Lernen. Diese Techniken verwenden dabei meist Informationen aus der Projekthistorie, wie sie zum Beispiel in Bugtracking-Systemen oder Versionsverwaltungssystemen für Quelltexte zu finden sind. Ein Überblick über Vorhersagemodelle und Techniken zur Erzeugung von Vorhersagemodellen findet sich zum Beispiel in [SBH14].

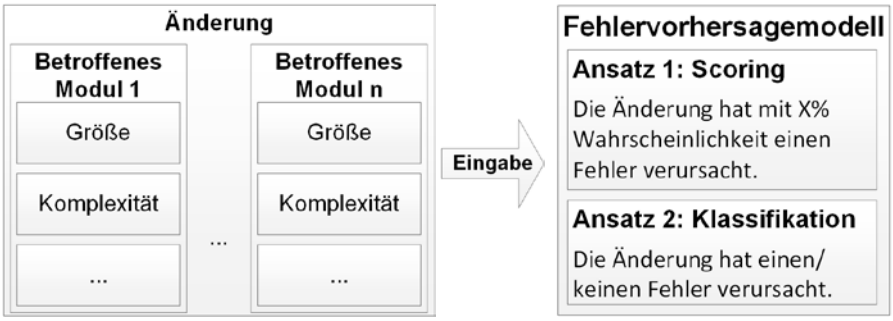

Abb. 2: Fehlervorhersage für Änderungen

Beispiel: Hotspots zur Vorhersage

Ein konkretes Beispiel für die Fehlervorhersage in der Praxis basiert auf sogenannten Hotspots und kommt bei Google zum Einsatz [CR11]. Ein Hotspot ist ein Modul einer Software, welches häufig

Probleme verursacht und daher besonders gut gewartet werden muss. Zur Erkennung von Hotspots wird hier ein sehr einfaches Verfahren angewendet, welches nur eine einzige Kennzahl verwendet: die Anzahl der bisherigen Bugfixes an einem Modul. Basierend auf den Bugfixes wird dann eine Punktzahl für jedes Modul berechnet. Bugfixes, die weit in der Vergangenheit liegen, sind jedoch weniger relevant für die Erkennung eines Hotspots als Bugfixes, die erst vor Kurzem passiert sind. Daher hat das Team von Google die Bewertung so angepasst, dass ältere Bugfixes weniger Punkte erhalten als neue Bugfixes. Der Punktabfall ist hierbei exponentiell mit der Zeit, das heißt, aktuelle Bugfixes liefern sehr viele Punkte, während alte Bugfixes nahezu ignoriert werden und nur durch ihre Masse punkten können.

Sobald für jedes Modul die Punktzahl ermittelt wurde, wird absteigend sortiert. Die 10 Prozent der Module mit der höchsten Punktzahl gelten als Hotspots. Sobald ein Modul geändert wird und in die interne Qualitätssicherung geht, wird überprüft, ob dieses Modul ein Hotspot ist. Falls ja, wird eine Warnung angezeigt, sodass der ausgewählte Bearbeiter weiß, dass es in dem Modul bereits häufig Fehler gab und er deshalb mehr Aufwand in Qualitätssicherung und Test investieren sollte.

Offene Probleme

Die Fehlervorhersage in Software ist ein sehr aktuelles Forschungsthema, an dem zunehmend intensiv geforscht wird. Auch wenn das obige Beispiel zeigt, dass es bereits industrialisierte Ansätze zur Fehlervorhersage gibt, sind viele Probleme der Fehlervorhersage ungelöst. So ist es zum Beispiel zurzeit nicht möglich, vernünftige Fehlervorhersagen für Projekte ohne historische Entwicklungsdaten durchzuführen. Nutzt man andere Daten, zum Beispiel aus anderen ähnlichen Projekten (engl. Cross-Project Defect Prediction), ist die Trefferquote bei der Fehlervorhersage noch viel zu gering. Dies verdeutlicht zum Beispiel eine aktuelle Studie, in der Fehlervorhersagemodelle für Automotive Software verglichen werden [AHGW15]. Weiterhin wird der Einsatz von Werkzeugen zur Fehlervorhersage zurzeit dadurch erschwert, dass die Vorhersagewerkzeuge aus der Forschung meist nur mangelhaft in gängige Entwicklungs-, Bugtracking-, und Quelltextmanagementsysteme integriert sind. ■

Literatur & Links

[AHGW15] H. Altinger, St. Herbold, J. Grabowski, F. Wotawa, Novel Insights on Cross Project Fault Prediction applied to Automotive Software, in: Proc. of the 27th International Conference on testing Software and Systems (ICTSS), 2015

[CR11] Ch. Lewis, Rong Ou, Bug Prediction at Google, siehe: http://google-engtools.blogspot.de/2011/12/bug-prediction-at-google.html, 2011

[SBH14] M. Shepperd, D. Bowes, T. Hall, Researcher Bias: The Use of Machine Learning in Software Defect Prediction, in: IEEE Transactions on Software Engineering 40(6): 603-616, IEEE, 2014