# Agentenbasierte dynamische Testfallpriorisierung

Von der Fakultät Informatik, Elektrotechnik und Informationstechnik
der Universität Stuttgart zur Erlangung der Würde eines
Doktor-Ingenieurs (Dr.-Ing.) genehmigte Abhandlung

Vorgelegt von
Christoph Malz
aus Danzig

| | |
|---|---|
| Hauptberichter: | Prof. Dr.-Ing. Dr. h. c. Peter Göhner |
| Mitberichter: | Prof. Dr.-Ing. Birgit Vogel-Heuser |
| Tag der Einreichung: | 25.09.2012 |
| Tag der mündlichen Prüfung: | 13.02.2013 |

Institut für Automatisierungs- und Softwaretechnik
der Universität Stuttgart

2013

IAS-Forschungsberichte

Band 2/2013

**Christoph Malz**

# Agentenbasierte dynamische Testfallpriorisierung

D 93 (Diss. Universität Stuttgart)

Shaker Verlag
Aachen 2013

**Bibliografische Information der Deutschen Nationalbibliothek**
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über http://dnb.d-nb.de abrufbar.

Zugl.: Stuttgart, Univ., Diss., 2013



ISBN 978-3-8440-1829-5
ISSN 1610-4781

Shaker Verlag GmbH • Postfach 101818 • 52018 Aachen
Telefon: 02407 / 95 96 - 0 • Telefax: 02407 / 95 96 - 9
Internet: www.shaker.de • E-Mail: info@shaker.de

Die vorliegende Arbeit entstand während meiner Tätigkeit als wissenschaftlicher Mitarbeiter am Institut für Automatisierungs- und Softwaretechnik (IAS) der Universität Stuttgart.

Mein besonderer Dank gilt meinem Doktorvater, Herrn Prof. Dr.-Ing. Dr. h. c. Peter Göhner, für die Anregungen und die fortwährende Unterstützung während der Entstehung dieser Arbeit sowie für die Übernahme des Hauptberichts.

Frau Prof. Dr.-Ing. Birgit Vogel-Heuser danke ich für das Interesse an meiner Arbeit und die Übernahme des Mitberichts.

Allen ehemaligen Kolleginnen und Kollegen am IAS gilt mein Dank für die offene und kollegiale Atmosphäre und die gute Zusammenarbeit. Ganz besonders bedanke ich mich bei Dr.-Ing. Nasser Jazdi für die regelmäßigen und hilfreichen Diskussionen über mein Forschungsthema sowie die kritische und gründliche Durchsicht des Manuskripts.

Erwähnen möchte ich auch die vielen Studierenden, die im Rahmen ihrer Diplom-, Master-, Studien- und Bachelorarbeiten einen wichtigen Beitrag zum Gelingen der Arbeit geleistet haben.

Schließlich möchte ich meinen Eltern für die langjährige Unterstützung während meiner gesamten Ausbildungszeit und meiner Frau Susanne für ihr Verständnis, ihre Geduld und ihre Unterstützung zum Gelingen dieser Arbeit herzlich danken.

Stuttgart, im März 2013

Christoph Malz

# Inhaltsverzeichnis

# Abbildungsverzeichnis

# Tabellenverzeichnis

# Abkürzungsverzeichnis

| | |
|---|---|
| CASE | **C**omputer **A**ided **S**oftware **E**ngineering |
| CAST | **C**omputer **A**ided **S**oftware **T**esting |
| DB | **D**aten**b**ank |
| DOM | **D**ocument **O**bject **M**ode |
| F | **F**unktion |
| FFW | **F**ehler**f**indungs**w**ahrscheinlichkeit |
| FW | **F**ehler**w**ahrscheinlichkeit |
| GP | **G**lobale **P**riorität |
| IAS | **I**nstitut für **A**utomatisierungs- und **S**oftwaretechnik |
| JADE | **J**ava **A**gent **DE**velopment Framework |
| LP | **L**okale **P**riorität |
| RPC | **R**emote **P**rocedure **C**all |
| SA | **S**chnittstellen-**A**gent |
| SM | **S**oftware**m**odul |
| SQL | **S**tructured **Q**uery **L**anguage |
| SVN | **S**ub**v**ersio**n** |
| TF | **T**est**f**all |
| TW | **T**est**w**ichtigkeit |
| TZP | **T**est**z**yklus**p**lanung |
| TZS | **T**est**z**yklus**s**teuerung |
| UML | **U**nified **M**odeling **L**anguage |
| XML | E**x**tensible **M**arkup **L**anguage |

# Begriffsverzeichnis

**Automatisierungssoftware:** Menge an Programmen zum Betrieb eines rechnerbasierten automatisierten Systems inklusive Dokumentation [Lind08].

**Black-Box-Verfahren:** Ein Verfahren zur Herleitung und Auswahl von Testfällen. Es basiert auf einer Analyse der funktionalen oder nicht-funktionalen Anforderungen (Spezifikationen) eines Softwaremoduls oder Softwaresystems ohne die Berücksichtigung ihrer internen Struktur [ISTQB09].

**Defekt:** Zustand eines Softwaresystems oder einer seiner Softwaremodule, der unter spezifischen Bedingungen eine geforderte Funktion des Softwaresystems beeinträchtigen kann bzw. zu einer Fehlerwirkung führt [ISTQB09].

**Fehlerfindungsrate:** Anzahl der Fehler gewichtetet mit der Fehlerschwere, die in einer Teststufe gefunden wurden, dividiert durch die Gesamtzahl der Fehler, die in dieser Teststufe und danach mit jeglichen Mitteln gefunden wurden [SRWL08].

**Fehlerschwere:** Der Grad der Auswirkung, den ein Fehler auf Entwicklung oder Betrieb eines Softwaremoduls oder eines Softwaresystems hat [ISTQB09].

**Fehlerwirkung:** Abweichung eines Softwaremoduls oder eines Softwaresystems von der erwarteten Lieferung, Leistung oder dem Ergebnis [ISTQB09].

**Fehlerzustand:** Zustand eines Softwaresystems oder einer seiner Softwaremodule, der unter spezifischen Bedingungen eine geforderte Funktion des Softwaresystems beeinträchtigen kann bzw. zu einer Fehlerwirkung führt [ISTQB09].

**Integrationstest:** Testen mit dem Ziel, Fehler in den Schnittstellen und im Zusammenspiel zwischen integrierten Softwaremodulen aufzudecken [ISTQB09].

**Modultest:** Testen eines einzelnen Softwaremoduls [ISTQB09].

**Regressionstest:** Erneutes Testen eines bereits getesteten Testobjekts bzw. einer Teilfunktionalität nach deren Modifikation. Ziel ist es nachzuweisen, dass durch die vorgenommenen Änderungen keine Fehlerzustände eingebaut oder (bisher maskierte Fehlerzustände) freigelegt wurden [ISTQB09].

**Softwareagent:** In der agentenorientierten Softwareentwicklung ist ein Softwareagent das Konzept einer abgrenzbaren Softwareeinheit mit einem definierten Ziel. Ein Softwareagent versucht, dieses Ziel durch autonomes Verhalten zu erreichen und interagiert dabei kontinuierlich mit seiner Umgebung und anderen Softwareagenten [VDI2653-1].

**Systemtest:** Testen eines integrierten Softwaresystems um sicherzustellen, dass es spezifizierte Anforderungen erfüllt [ISTQB09].

**Testaktivität:** Aktivität bzw. Teilaufgabe, die innerhalb des Testprozesses ausgeführt wird [SRWL08].

**Testdaten:** Daten, die vor der Ausführung eines Testfalls existieren, und die die Ausführung des Softwaremoduls bzw. des Softwaresystems beeinflussen bzw. dadurch beeinflusst werden [ISTQB09].

**Testeffektivität:** Anzahl gefundener Fehler gewichtetet mit der Fehlerschwere pro Testfall [SRWL08].

**Testeffizienz:** Anzahl gefundener Fehler gewichtetet mit der Fehlerschwere pro Zeiteinheit [SRWL08].

**Testendekriterien:** Kriterien, die vorab festgelegt werden und erfüllt sein müssen, um eine (Test-) Aktivität als abgeschlossen bewerten zu können [SpLi05].

**Testfall:** Ein Testfall umfasst folgende Angaben: Die für die Durchführung notwendigen Vorbedingungen, die Menge der Eingabewerte, die Menge der vorausgesagten Ergebnisse sowie die erwarteten Nachbedingungen [SRWL08].

**Testmanagement:** Planung, Aufwandsschätzung, Überwachung und Kontrolle von Testaktivitäten, die üblicherweise durch einen Testmanager erfolgen [ISTQB09].

**Testmethode:** Planmäßiges, auf einem Regelwerk aufbauendes Vorgehen zur Herleitung und/oder Auswahl von Testfällen [SpLi05].

**Testobjekt:** Die Softwaremodule oder das Softwaresystem, welches getestet wird [ISTQB09].

**Testphase:** Eine abgegrenzte Menge von Testaktivitäten, die einer Projektphase zugeordnet sind [ISTQB09].

**Testplan:** Eine Liste von Aktivitäten, Aufgaben oder Ereignissen des Testprozesses, mit Angabe ihrer geplanten Anfangs- und Endtermine sowie ihrer gegenseitigen Abhängigkeiten [ISTQB09].

**Testprojekt:** Im Wesentlichen ein einmaliges Vorhaben, bei dem innerhalb einer definierten Zeitspanne definierte Testziele erreicht werden sollen; üblicherweise Teil eines Software- oder Systementwicklungsprojekts [SRWL08].

**Testprozess:** Der fundamentale Testprozess umfasst die folgenden Aktivitäten: Testplanung und Teststeuerung, Testspezifikation, Testausführung und Testauswertung [ISTQB09].

**Testressourcen:** Die Elemente, die für die Durchführung des Testfalls benötigt werden, bestehend aus Testumgebung, Testwerkzeugen und gegebenenfalls Personal [SRWL08].

**Testskript:** Anweisungen zur automatischen Ausführung eines Testfalls in einer geeigneten Programmiersprache [ISTQB09].

**Teststeuerung:** Als Teststeuerung bezeichnet man die Managementaufgaben zur Entwicklung und Anwendung von Korrekturmaßnahmen, um in einem Testprojekt eine Abweichung vom geplanten Vorgehen zu beherrschen [ISTQB09].

**Teststrategie:** Abstrakte Beschreibung der vorgesehenen Teststufen und der Art und Weise, wie innerhalb dieser Teststufe vorzugehen ist, für eine Organisation oder ein Testobjekt – gültig für ein oder mehrere Projekte [ISTQB09].

**Teststufe:** Eine Teststufe ist eine Gruppe von Testaktivitäten, die gemeinsam ausgeführt und verwaltet werden. Teststufen sind mit Zuständigkeiten in einem Projekt verknüpft. Beispiele für Teststufen sind der Modultest, der Integrationstest und der Systemtest [ISTQB09].

**Test Suite:** Die Zusammenstellung mehrerer Testfälle für den Test eines Softwaremoduls oder eines Softwaresystems [ISTQB09].

**Testumgebung:** Eine Umgebung, die benötigt wird, um Testfälle auszuführen. Sie umfasst Hardware, Instrumentierung, Simulatoren, Softwarewerkzeuge und andere unterstützende Hilfsmittel [ISTQB09].

**Testziel:** Ein Grund oder Zweck für den Entwurf und die Ausführung von Testfällen [ISTQB09].

**Testzyklus:** Durchlauf durch den Testprozess auf Basis genau einer Version des Testobjekts, an dessen Ende Fehlerkorrektur- und Änderungsaufträge für die Entwickler vorliegen [SpLi05].

**White-Box-Verfahren:** Ein dokumentiertes Verfahren zur Herleitung und Auswahl von Testfällen, basierend auf der internen Struktur eines Softwaremoduls oder eines Softwaresystems [ISTQB09].

# Zusammenfassung

Die zunehmende Komplexität von Automatisierungssoftware erfordert einen immer größer werdenden Testaufwand dieser Softwaresysteme, um sicherzustellen, dass die automatisierten Systeme zuverlässig funktionieren. Auf der einen Seite verfügt der Testmanager für die Durchführung dieser Systemtests über eine große Menge an Testfällen. Auf der anderen Seite sind jedoch die Testzeit und die Anzahl der Testressourcen, die dem Testmanager zur Disposition stehen, begrenzt. Er ist gezwungen, mit hohem Aufwand sowohl von Testzyklus zu Testzyklus als auch gegebenenfalls während eines Testzyklus des Systemtests eine Priorisierung der zur Verfügung stehenden Testfälle durchzuführen, um trotz der begrenzten Testzeit und der begrenzten Testressourcen verlässliche Testresultate abzuliefern und sicherzustellen, dass die Softwaresysteme ausreichend geprüft werden.

In der vorliegenden Arbeit wird ein Konzept vorgestellt, das mithilfe von Softwareagenten diesen manuellen Aufwand reduziert und den Fehlerentdeckungsgrad, die Effektivität und die Effizienz beim Systemtest erhöht. Softwareagenten werten die im Systemtest bereitstehenden Informationen aus der Systementwicklung, dem Systemtest und dem Systembetrieb dynamisch aus und schlagen aktiv und konstruktiv eine Testfallpriorisierung für die Testzyklen des Systemtests vor. Das Ziel der Softwareagenten ist dabei, eine Testfallpriorisierung zu erreichen, die es ermöglicht, in der zur Verfügung stehenden Testzeit möglichst viele und insbesondere schwerwiegende Fehler zu finden. Die Grundidee der agentenbasierten dynamischen Testfallpriorisierung ist, dass Softwareagenten Softwaremodule und Testfälle des zu testenden Softwaresystems vertreten und aus deren Perspektive in Zusammenarbeit die Testfallpriorisierung durchführen. Zur Testfallpriorisierung bestimmen die Softwareagenten Fehlerwahrscheinlichkeiten auf Basis von Informationen, die in bereits in der Industrie eingesetzten Entwicklungs- und Testwerkzeugen vorhanden sind, was die Anwendbarkeit im Systemtest sicherstellt. Je nach Bedarf bzw. Verfügbarkeit können die Informationen, die zur Auswertung eingesetzt werden, angepasst werden. Die Softwareagenten werten die Informationen mit Priorisierungsregeln aus, die anhand von Erfahrungswissen basierend auf Fuzzy-Regelwerken definiert wurden. Auch die Priorisierungsregeln können je nach Bedarf angeglichen werden. Die Softwareagenten arbeiten aktiv im Hintergrund. Sie registrieren die Veränderung von Informationen aufgrund dynamischer Ereignisse und reagieren auf diese mit der Anpassung der Testfallpriorisierung. Ein Testfallpriorisierungssystem basierend auf dem Konzept der agentenbasierten dynamischen Testfallpriorisierung wurde realisiert und evaluiert. Die Evaluierungsergebnisse zeigen eine deutliche Steigerung des Fehlerentdeckungsgrads, der Testeffektivität und der Testeffizienz sowie eine Reduzierung des zeitlichen Aufwands für die Testfallpriorisierung im Vergleich zu einer Testfallpriorisierung durch einen Menschen mithilfe von herkömmlichen Entwicklungs- und Testwerkzeugen.

# Abstract

The rising complexity of industrial automation software requires an increasing test effort of these software systems in order to ensure that the automated systems work reliable. On one side, a high number of test cases for the execution of these system tests are available for the test manager. However, on the other side, the test time and the extent of test resources, which are available for the test manager, are restricted. The test manager has to execute a prioritization of the available test cases with high effort for each test cycle and, if needed, during the cycle of the system test, in order to deliver reliable test results and to ensure that the software system is tested sufficiently despite the restricted test time and the restricted test resources.

In this work a concept is introduced, which with the help of software agents reduces the manual effort and increases the fault detection rate, the test effectiveness and the test efficiency at the system test. Software agents evaluate dynamically the information from the system development, the system test and the system operation, which is available in the system test, and recommend actively and constructively the test case prioritization for the test cycles of the system test. The goal of the software agents is to reach a test case prioritization, which allows finding as much as possible faults and especially severe faults in the available test time. The basic idea of the agent-based dynamic test case prioritization is that software agents represent software modules and test cases of the software system to be tested and in cooperation execute the test case prioritization from their perspective. The software agents determine fault probabilities for the test case prioritization based on information, which is available in development and test tools used already in industry. This ensures the applicability of the concept in the system test. The information which is used for evaluation can be adapted dependent on the demand and availability. The software agents evaluate the information with prioritization rules which are defined with fuzzy logic on the basis of experience knowledge. The prioritization rules can be also adapted as needed. The software agents work actively in the background. They register modifications of information due to dynamic events and react to these with modifications of the test case prioritization. The test case prioritization system based on the concept of the agent-based dynamic test case prioritization was realized and evaluated. The evaluation results show a considerable increase of the fault detection rate, test effectiveness, test efficiency and a reduction of the time effort for the test case prioritization in comparison to a test case prioritization by a human being using conventional development and test tools.