Rainer Typke

Nützlichkeit von Zusicherungen als Hilfsmittel beim Programmieren

Ein kontrolliertes Experiment

Diplomarbeit

Diplom.de

Bibliografische Information der Deutschen Nationalbibliothek:

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

ISBN: 9783832423957

Rainer Typke

Nützlichkeit von Zusicherungen als Hilfsmittel beim Programmieren

Ein kontrolliertes Experiment

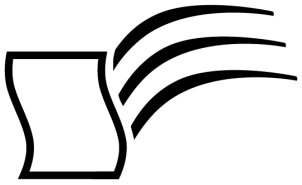
Diplom.de

Rainer Typke

Nützlichkeit von Zusicherungen als Hilfsmittel beim Programmieren

Ein kontrolliertes Experiment

Diplomarbeit
an der Universität Fridericiana Karlsruhe (TH)
Fachbereich Informatik
Prüfer Prof. W. Tichy
Institut für Programmstrukturen und Datenorganisation,
April 1999 Abgabe



ID 2395
Typke, Rainer: Nützlichkeit von Zusicherungen als Hilfsmittel beim Programmieren: Ein kontrolliertes Experiment / Rainer Typke - Hamburg: Diplomarbeiten Agentur, 2000
Zugl.: Karlsruhe, Technische Universität, Diplom, 1999

Dipl. Kfm. Dipl. Hdl. Björn Bedey, Dipl. Wi.-Ing. Martin Haschke & Guido Meyer GbR
Diplomarbeiten Agentur, http://www.diplom.de, Hamburg 2000
Printed in Germany

Diplomarbeiten Agentur

Wissensquellen gewinnbringend nutzen

Qualität, Praxisrelevanz und Aktualität zeichnen unsere Studien aus. Wir bieten Ihnen im Auftrag unserer Autorinnen und Autoren Wirtschaftsstudien und wissenschaftliche Abschlussarbeiten – Dissertationen, Diplomarbeiten, Magisterarbeiten, Staatsexamensarbeiten und Studienarbeiten zum Kauf. Sie wurden an deutschen Universitäten, Fachhochschulen, Akademien oder vergleichbaren Institutionen der Europäischen Union geschrieben. Der Notendurchschnitt liegt bei 1,5.

Wettbewerbsvorteile verschaffen – Vergleichen Sie den Preis unserer Studien mit den Honoraren externer Berater. Um dieses Wissen selbst zusammenzutragen, müssten Sie viel Zeit und Geld aufbringen.

http://www.diplom.de bietet Ihnen unser vollständiges Lieferprogramm mit mehreren tausend Studien im Internet. Neben dem Online-Katalog und der Online-Suchmaschine für Ihre Recherche steht Ihnen auch eine Online-Bestellfunktion zur Verfügung. Inhaltliche Zusammenfassungen und Inhaltsverzeichnisse zu jeder Studie sind im Internet einsehbar.

Individueller Service – Gerne senden wir Ihnen auch unseren Papierkatalog zu. Bitte fordern Sie Ihr individuelles Exemplar bei uns an. Für Fragen, Anregungen und individuelle Anfragen stehen wir Ihnen gerne zur Verfügung. Wir freuen uns auf eine gute Zusammenarbeit

Ihr Team der *Diplomarbeiten* Agentur

Dipl. Kfm. Dipl. Hdl. Björn Bedey
Dipl. Wi.-Ing. Martin Haschke
und Guido Meyer GbR

Hermannstal 119 k
22119 Hamburg

Fon: 040 / 655 99 20
Fax: 040 / 655 99 222

agentur@diplom.de
www.diplom.de

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Hiermit versichere ich, die vorliegende Arbeit selbständig verfaßt und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet zu haben.

Karlsruhe, den 29. April 1999 _____

Zusammenfassung

Im Rahmen dieser Diplomarbeit wurde ein kontrolliertes Experiment durchgeführt, mit dem die Nützlichkeit von Zusicherungen als Hilfsmittel beim Programmieren untersucht wurde. Zusicherungen sind Bedingungen, die mitten im Programmtext, als Vor- oder Nachbedingung einer Funktion zugeordnet, oder als Invariante auf eine Klasse bezogen auftreten können. Ihre Einhaltung kann während der Laufzeit überwacht werden. In diesem Experiment wurden APP für C und `jContract` für Java als Zusicherungswerkzeuge eingesetzt. Das Ergebnis legt nahe, daß der Zeitbedarf beim Programmieren durch die Verwendung von Zusicherungen nicht steigt, aber die Qualität der entstehenden Software.

1 Einleitung

1.1 Zusicherungen

Ein bekanntes Problem bei der Softwareentwicklung ist der Konflikt zwischen den Zielen, einerseits möglichst korrekte Programme zu schreiben und andererseits den Aufwand in einem vernünftigen finanziellen und zeitlichen Rahmen zu halten. Es gibt viele Arbeiten (Beispiel: der „Karlsruhe Interactive Verifier“, KIV [Reif 92]) auf dem Gebiet der Softwareverifikation, die darauf abzielen, zu beweisen, daß ein gegebenes Programm korrekt ist, also einer Implementierung einer gegebenen Spezifikation entspricht. Derzeit sind vollständige formale Korrektheitsbeweise nur für kleine Programmteile und unter erheblichem zeitlichem Aufwand zu erreichen. Nicht genügend auf die Softwarekorrektheit zu achten, kann aber unangenehme Folgen haben. Hierzu gibt es viele bekannte Beispiele (in [Prechelt 97] ist eine lange Liste aufgeführt). Es wäre also wünschenswert, einen Mittelweg zu finden, der mit vertretbarem Aufwand einen Gewinn an Softwarequalität verspricht, der dem möglichst nahekommt, was sich mit teuren formalen Verifikationsmethoden erreichen läßt.

Ein solcher Mittelweg, der schon vor mehr als zwei Jahrzehnten von verschiedenen Autoren beschrieben wurde (z. B. [Stucki, Foshee 75] und [Yau, Cheung 75]), beruht auf dem Einfügen von Zusicherungen in den Programmtext, die während der Laufzeit geprüft werden können. Der Programmierer kann also im Programmtext festhalten, daß zu einer bestimmten Zeit eine bestimmte Bedingung gelten muß. In die Programmiersprache C hat diese Idee in Form des in `assert.h` definierten Makros `assert` Eingang gefunden.[1]

David S. Rosenblum stellte vor sechs Jahren in [Rosenblum 92] fest, daß diese Grundidee und darauf aufbauende hochentwickelte Systeme wie *Anna* (ANNotated Ada) [Luckham, Henke 85] zwar theoretisch überzeugen, sich aber nicht allgemein durchgesetzt haben. Er sieht vor allem zwei Gründe dafür: mangelhafte Integration der Zusicherungs-Werkzeuge in bestehende Programmierumgebungen und fehlendes Wissen darüber, wie Zusicherungen aussehen müssen, um Fehler möglichst effektiv aufzudecken.

Auch für Eiffel gibt es eine in die Sprache integrierte Möglichkeit, Zusicherungen zu formulieren. Bertrand Meyer beschreibt das Konzept des „Entwurfs per Vertrag“ in seinem Buch „Object-oriented Software Construction“ [Meyer 97]. Dabei wird die Beziehung zwischen einer Klasse und deren Aufrufern als formeller Vertrag betrachtet, in dem für jede Seite Verpflichtungen und Rechte festgelegt werden. Verpflichtungen einer Methode schlagen sich in einer Nachbedingung nieder, die nach der Ausführung gelten muß. Im Gegenzug hat die Methode das Recht, davon auszugehen, daß die Vorbedingung gilt, die ihr zugeordnet ist. Sollte diese Voraussetzung nicht erfüllt sein, ist sie auch nicht an die Nachbedingung gebunden. Ein drittes Instrument ist die Möglich-

[1] Das „Linux Programmer’s Manual“ enthält eine nett formulierte Beschreibung dieses Makros. Man findet dort die Warnung: *assert() is implemented as a macro; if the expression tested has side-effects, program behaviour will be different depending on whether NDEBUG is defined. This may create Heisenbugs which go away when debugging is turned on.*