Workshop "Applied Program Analysis" (APA 2007)

Jochen Quante, Thilo Mende
AG Softwaretechnik, FB 03, Universität Bremen
{quante,tmende}@informatik.uni-bremen.de

im Rahmen der INFORMATIK 2007 in Bremen
27. September 2007

Aktuelle Softwaresysteme sind aus Entwicklersicht auf Grund ihrer Größe und Komplexität manuell kaum mehr handhabbar. Dieses Problem betrifft den gesamten Lebenszyklus einer Software – neben der Entwicklung und dem Test auch die Wartung und Weiterentwicklung. Der Quelltext als verlässlichste Informationsquelle hat dabei für viele Aufgaben einen zu hohen Detaillierungsgrad. Durch automatisierte Programmanalysen kann das Abstraktionsniveau erhöht bzw. genau die jeweils benötigte Information extrahiert werden. Programmanalysen werden damit in der Softwareentwicklung zunehmend relevanter.

Ein wichtiges Anwendungsgebiet der Programmanalyse ist das Programmverstehen. Empirische Untersuchungen haben gezeigt, dass Wartungsprogrammierer ca. 50% ihrer Zeit nur mit der Analyse der Implementierung verbringen, bevor sie eine Änderung tatsächlich durchführen und testen können. Techniken und Werkzeuge, die diese Analysetätigkeit geeignet unterstützen, können die dafür benötigte Zeit deutlich reduzieren. Entwicklungsbegleitender Einsatz entsprechender Werkzeuge kann auch dazu führen, dass die Software von vornherein besser wartbar ist, weil etwa die korrekte Umsetzung der Architektur sichergestellt wird.

Auch in anderen Bereichen können Programmanalysen Unterstützung bieten. So beispielsweise bei der Fehlersuche: Es existieren Ansätze zur Erkennung oder Vermeidung von bestimmten Arten von Fehlern sowie zur Lokalisierung von Fehlern. Andere Analysen werden zur automatischen Erzeugung von Testdaten eingesetzt, um so die Testabdeckung zu erhöhen und damit die Qualität der Software zu verbessern. Auch für die Messung und Sicherstellung von Softwarequalität werden Programmanalysen genutzt.

Programmanalysen werden also in der Softwaretechnik zu ganz unterschiedlichen Zwecken eingesetzt. Sie basieren auf den gleichen Grundlagen und stehen vor ähnlichen Herausforderungen. Dazu gehört beispielsweise die Aufgabe, genau die gewünschten Informationen aus den entstehenden großen Datenmengen zu extrahieren, um so eine effektive Nutzung zu ermöglichen. Diese Gemeinsamkeiten und der Gedanke, Erfahrungen aus bestimmten Anwendungen der Programmanalyse in andere Bereiche zu übertragen, bildeten die Grundidee für diesen Workshop.

Am 27. September 2007 fand in Bremen im Rahmen der INFORMATIK 2007 der erste "Workshop Applied Program Analysis" statt. Ziel dieses Workshops war es, industrielle Anwender und Forscher aus verschiedenen Gebieten der Programmanalyse im Kontext von Softwaretechnik zusammenzubringen, um einen Erfahrungsaustausch zu ermöglichen. Dabei stand neben einem breiten Spektrum von Einsatzzwecken der Aspekt der Praxistauglichkeit im Vordergrund.

Im ersten Teil des Workshops wurden sechs ausgewählte Arbeiten präsentiert und diskutiert. Die Beiträge befassten sich im Wesentlichen mit den Themen Reengineering, Softwarequalität und Programmverifikation.

Stefan Staiger (Universität Stuttgart): *Statische Analyse von graphischen Oberflächen*

In seinem Beitrag beschreibt Stefan Staiger, wie Programme mit graphischer Oberfläche statisch analysiert werden können. Die Analyse extrahiert aus dem Quellcode die Widget-Hierarchien der GUI sowie GUI-Ereignisse und Reaktionen darauf. Mit Hilfe dieses Ansatzes kann beispielsweise die Migration einer handgeschriebenen GUI zu einem GUI-Builder unterstützt werden.

Marcel Bennicke, Frank Steinbrückner, Mathias Radicke (BTU Cottbus), **Jan-Peter Richter** (sd&m Research): *Das sd&m Software-Cockpit: Architektur und Erfahrungen*

Ein Software-Leitstand analysiert eine Codebasis bezüglich verschiedener Qualitätskriterien und stellt darauf aufbauend eine individuell konfigurierbare Sicht auf den Qualitätszustand eines Software-Projekts bereit. Die Autoren um Marcel Bennicke untersuchen in ihrem Beitrag, welchen Zusatznutzen eine solche Integrationsumgebung im Vergleich zum Einzeleinsatz bestehender Analysewerkzeuge generieren kann.

Alexander Schmidt, Michael Schöbel (Universität Potsdam): *Analyzing System Behavior: How the Operating System Can Help*

Alexander Schmidt und Michael Schöbel berich-

ten von ihren Erfahrungen mit der Instrumentierung des Windows Kernels. Die resultierende feingranulare Logging-Infrastruktur kann helfen, das Verhalten von Windows-Anwendungen – insbesondere Abhängigkeiten von auftretenden Ereignissen – zu verstehen.

Gunther Vogel (Universität Stuttgart): *Transformation und Vergleich von endlichen Automaten zur Analyse von Software-Protokollen*

Gunther Vogel beschreibt, wie endliche Automaten für Software-Protokolle aus dem Quelltext gewonnen und durch Transformationen für die Weiterverarbeitung und den Vergleich aufbereitet werden können. Diese Technik wird zur Prüfung von Protokollen oder zur Herleitung von Protokollspezifikationen eingesetzt.

Klaus Wissing (PolySpace Technologies GmbH): *Static Analysis of Dynamic Properties - Automatic Program Verification to Prove the Absence of Dynamic Runtime Errors*

Der Nutzen von formaler Verifikation zur Messung von Software-Qualitätsattributen wird in Klaus Wissings Beitrag thematisiert. Die zugrunde liegende Technik, die abstrakte Interpretation, kann die Abwesenheit von Laufzeitfehlern beweisen. Deren Ergebnisse werden dann genutzt, um qualitätssichernde Maßnahmen gezielt und somit effektiver einzusetzen.

Jan Peleska, Helge Löding (Universität Bremen), **Tatiana Kotas** (Verified Systems International GmbH): *Test Automation Meets Static Analysis*

In diesem Beitrag wird die Anwendung der statischen Analyse im Modul- oder Integrationstest beschrieben. Dabei wird abstrakte Interpretation genutzt, um Testdaten zu generieren, die eine möglichst hohe Verzweigungsabdeckung erreichen. Unvermeidlich auftretende Falsch-Positive können anschließend durch einen automatisierten Test identifiziert werden.

Langfassungen dieser Beiträge sind in den Proceedings zur INFORMATIK 2007 (R. Koschke, O. Herzog, K.-H. Rödiger, M. Ronthaler (Hrsg.): INFORMATIK 2007 – Informatik trifft Logistik, Band 2, LNI 110, Bremen, 2007, S. 243–286) erschienen.

Der zweite Teil des Workshops startete mit einer Reihe von Tooldemos. Es wurden fünf Werkzeuge aus dem Bereich Programmanalyse vorgestellt. Dabei wurden eindrucksvoll die Potentiale von Programmanalysen für die Praxis demonstriert. Das Anwendungsspektrum der Werkzeuge reichte von der Architekturanalyse und -visualisierung (SotoArc, Bauhaus Suite) über Programmverifikation (Verified RT-Tester) und die Wartung von Mainframe-Programmen (IBM Websphere Developer) bis zur Erkennung von Abhängigkeiten verschiedener Anwendungen (IBM Tivoli Application Dependency Discovery Manager). Zwei der Tools standen dabei in unmittelbarem Zusammenhang zu entsprechenden Vorträgen aus dem ersten Teil. Auf die Verfügbarkeit und Praxistauglichkeit der vorgestellten Tools wurde bei der Auswahl besonderer Wert gelegt. Im Anschluss an die Tooldemos wurde noch im kleineren Kreis weiterdiskutiert und -demonstriert sowie neue Kontakte geknüpft.

Insgesamt war der Workshop mit knapp 40 Teilnehmern einer der drei bestbesuchten Workshops der INFORMATIK-Tagung und damit ein großer Erfolg. Von vielen Teilnehmern wurde der Wunsch nach einer regelmäßigen Fortführung des Workshops geäußert. Dies verdeutlicht den Bedarf und den Nutzen des Austausches zwischen den verschiedenen Anwendern von Programmanalysen.

Der Workshop “Applied Program Analysis” wurde in Kooperation mit der Fachgruppe *Software-Reengineering* (`http://www.uni-koblenz.de/sre/`) durchgeführt. Die Organisatoren danken herzlich den Referenten und Autoren für die Ausarbeitung und Präsentation ihrer Beiträge, den Workshopteilnehmern für das Interesse und die anregenden Diskussionen, den Mitgliedern des Programmkomitees für ihre Gutachten sowie den Organisatoren der INFORMATIK 2007 für die Bereitstellung des organisatorischen Rahmens.