Bericht über die ICSE Workshops SESS and WADS 2005

Timo Warns
Carl von Ossietzky Universität Oldenburg,
Department für Informatik, Abt. Software Engineering,
Graduiertenkolleg *Vertrauenswürdige Software**
timo.warns@informatik.uni-oldenburg.de

Der Aspekt der Vertrauenswürdigkeit nimmt eine zunehmend größere Rolle bei der Entwicklung von Software ein. Um die dabei auftretenden Herausforderungen zu bewältigen, ist die Zusammenarbeit der Communities der Fehlertoleranz, Sicherheit und Softwaretechnik gefragt. Dieser Bericht fasst die Ergebnisse der ICSE Workshops *Software Engineering for Secure Systems* (SESS) und *Workshop on Architecting Dependable Systems* (WADS) 2005 zusammen, in denen diese Communities zusammentrafen.

1 Einleitung

Die 27. *International Conference on Software Engineering* (ICSE 2005) fand in den Vereinigten Staaten in St. Louis, Missouri, vom 15. bis zum 21. Mai 2005 statt. Als bedeutendste und größte Konferenz zur Softwaretechnik bietet sie ein umfassendes Forum für Forschung, Industrie und Lehre. Neben der eigentlichen Hauptkonferenz bündelte sie in 19 Workshops und 15 Tutorials verschiedene Strömungen zur (Weiter-)Entwicklung von Techniken, Praktiken und Werkzeugen der Softwaretechnik.

Software durchdringt immer mehr alle Bereiche des täglichen Lebens. Daraus resultiert eine zunehmende Abhängigkeit der Benutzer vom erfolgreichen Einsatz der Software. Die ICSE 2005 adressierte unter dem Oberthema *Software Everywhere* die Herausforderungen der Entwicklung solch allgegenwärtiger Systeme.

Eine Voraussetzung für den erfolgreichen Einsatz von Software Systemen ist das Vertrauen der Benutzer in die Systeme. Der Aspekt der Entwicklung vertrauenswürdiger Systeme wurde im Rahmen der ICSE 2005 in den Workshops *Software Engineering for Secure Systems* und *Workshop on Architecting Dependable Systems* behandelt. Der vorliegende Bericht fasst die Vorträge und Diskussionen dieser beiden Veranstaltungen zusammen. Weitere Informationen zur ICSE 2005 lassen sich unter `http://www.cs.wustl.edu/icse05/` finden.

2 SESS

Der erstmalig stattgefundene Workshop *Software Engineering for Secure Systems* war ein Rahmen für Diskussionen zu aktuellen Ansätzen der Entwicklung und Analyse sicherer Systeme. Unter dem Motto *Building Trustworthy Applications* wurden die internationalen Communities der Sicherheit und Softwaretechnik zusammen gebracht, um Synergie-Effekte für vertrauenswürdigere Applikationen zu nutzen.

Der Workshop erstreckte sich zeitlich in drei Sitzungen vom Morgen des 15. Mai bis zum Mittag des 16. Mai. Jede Sitzung beinhaltete zwei

*Diese Arbeit wurde durch das Graduiertenkolleg GRK 1076/1 der Deutschen Forschungsgemeinschaft (DFG) unterstützt.

bis vier lange Vorträge (20 min Präsentation und 5 min Fragen) und ein bis zwei kurze Vorträge (10 min Präsentation und 5 min Fragen). Darüber hinaus war vor den Pausen ausreichend Zeit für allgemeine Diskussionen über die eigentlichen Themen der Vorträge hinaus.

Die Organisatoren Danilo Bruschi, Bart De Win und Mattia Monga freuten sich über 27 eingereichte Beiträge von denen 16 akzeptiert wurden. Die Vorträge deckten vom Requirements Engineering bis hin zum systematischen, automatisierten Testen ein weites Spektrum der Softwaretechnik ab. Innerhalb dieser thematischen Breite lagen Schwerpunkte bei Weiterentwicklungen von komponentenbasierten Techniken und der Zugriffskontrolle.

Die erste Sitzung, die von Danilo Bruschi geleitet wurde, begann mit einem Vortrag *DISCOA: Architectural Adaptations for Security and QoS* von Omer E. Demir. Er erläuterte einen Ansatz für Adaptionen in verteilten Architekturen, die insbesondere Lösungen von Sicherheitsanforderungen adressieren, die quer zu üblichen komponentenbasierten Sichten liegen.

Maarten Rits stellte in *XacT: A Bridge between Resource Management and Access Control in Multi-layered Applications* ein Werkzeug vor, das mit Aspekt-orientierten Techniken Zugriffskontrollinformationen in Mehrschichtsystemen ermittelt und durchsetzt.

Unter dem Titel *Leveraging Architectural Models to Inject Trust into Software Systems* präsentierte Chris A. Mattmann eine rudimentäre Taxonomie zum Konzept der Vertrauenswürdigkeit und bildete deren Elemente auf Architekturelemente ab. Diese als Positionspapier zu verstehende Arbeit möchte das Verständnis für Vertrauenswürdigkeit und deren Beziehung zu Software Architekturen erhöhen.

In *Towards An Architectural Treatment of Software Security: A Connector-Centric Approach* erörterte Jie Ren eine Erweiterung der Architekturbeschreibungssprache *xADL* um Komponentenverträge zur Sicherheit. Die Verträge werden dabei durch Konnektoren und Adaptionstechniken durchgesetzt. Dies wurde anhand eines Beispiels für eine P2P Dateitauschbörse veranschaulicht.

Der Mitorganisator Bart De Win stellte in *Towards a Unifying View on Security Contracts* eine initiale Studie über Komponentenverträge zur Sicherheit vor. Mit dem Ziel der Vorhersagbarkeit der Sicherheit eines Systems präsentierte er eine Klassifikation von Verträgen in *Funktionale Interaktion*, *Interaktionsprotokolle*, *Sicherheitsspezifische Interaktion* und *Infrastruktur* eingeteilt werden.

Tine Verhanneman präsentierte mit *Requirements Traceability to Support Evolution of Access Control* eine Abstraktionsschicht für die Umsetzung von Zugriffskontrolle. Die vorgestellte Abstraktionsschicht wird mit Aspekt-orientierten Techniken umgesetzt und unterstützt so insbesondere die Evolution von Anforderungen.

Die zweite Sitzung, die von Bart de Win geleitet wurde, wurde von Robin A. Gandhi mit *Establishing Trustworthiness in Services of the Critical Infrastructure through Certification and Accreditation* eingeleitet. Er beschrieb darin ein Framework zur Zertifizierung und Akkreditierung von Systemen für die Automatisierung des *Department of Defense Information Technology Security Certification and Accreditation Process* (DITSCAP).

Zaid Dwaikat schlug in *Risky Trust: Risk-Based Analysis of Software Systems* ein Modell zur Evaluation von Systemen vor, das auf dem Risiko-Konzept beruht. Komponenten werden dabei mit zusätzlichen Informationen zur Sicherheit versehen, um für Transaktionen das Risiko von Sicherheitsverletzungen zu bestimmen. Ähnlich zum Vortrag von Bart De Win wird dabei das Ziel verfolgt, die Systemqualität aus den Eigenschaften der Komponenten vorher zusagen.

Nancy R. Mead beschrieb in *Security Quality Requirements Engineering (SQUARE) Methodology* eine Methodik zum Requirements Engineering, die Rücksicht auf Besonderheiten von Sicherheitsanforderungen nimmt und in einer Fallstudie von Studenten in Zusammenarbeit mit einem Industriepartner angewendet wurde.

Karsten Sohr stellte in *Articulating and Enforcing Authorisation Policies with UML and OCL* ein Werkzeug zur Handhabung von Zugriffskontrollinformationen vor. Das Tool unterstützt die Formulierung und Durchsetzung von Regeln zur Zugriffskontrolle mit Hilfe der UML und OCL.

Die letzte Sitzung, geleitet von Mattia Monga, begann mit einem Vortrag *A Framework for Testing Security Mechanisms for Program-Based Attacks* von Ben Breech. Er diskutierte einen Ansatz zum systematischen, automatisierten Testen von Sicherheitsmechanismen, die Programme gegen bösartige Eingaben schützen sollen. Mit Hilfe von dynamischen Compilern werden dabei Attacken simuliert, die Verwundbarkeiten wie z.B. Puffer-Überläufe nutzen.

Sam Weber präsentierte mit *A Software Flaw Taxonomy: Aiming Tools At Security* eine Taxonomie von Software-Fehlern und wendete diese auf eine Liste von Verwundbarkeiten von Web-Applikationen des *Open Web Application Security Project* (OWASP) an.

Wes Masri zeigte in *Using Dynamic Information Flow Analysis to Detect Attacks against Applications* wie die Informationsfluss- zusammen mit der Cluster-Analyse genutzt werden kann, um Attacken gegen Programme zu erkennen.

In *Enabling Control over Adaptive Program Transformation for Dynamically Evolving Mobile Software Validation* schilderte Lori L. Pollock die Arbeit einer ihrer Studenten, die die Verteilung von Programmtransformationen in verteilten, mobilen Kontexten absichert.

Die Vortragsreihe schloss mit einer Präsentation von Michael Gegick über *Matching Attack Patterns to Security Vulnerabilities in Software-Intensive System Designs*, in dem reguläre Ausdrücke genutzt wurden, um Angriffe als Muster zu beschreiben. Diese Muster sollen zur Evaluation der Sicherheit von Systemen genutzt werden.

Neben den eigentlichen Vorträgen gab es Diskussionen zu Herausforderungen der Softwaretechnik von sicheren Systemen im Allgemeinen. Als ungelöstes Problem wird das Requirements Engineering von Sicherheitsanforderungen gesehen. Gründe dafür liegen in der Natur solcher Anforderungen. Sie werden häufig als Negation formuliert und sind so schwer umzusetzen und zu validieren (z.B. „Die Information xyz darf das System nicht verlassen.“). Außerdem sind adäquate Fehlermodelle, wie sie im Bereich der Fehlertoleranz üblich sind, sehr komplex, da eine große Anzahl heterogener Aspekte zu beachten ist, wie z.B. menschliche Faktoren und verdeckte Kanäle.

Große Herausforderungen liegen auch in der Umsetzung von Sicherheitsanforderungen, da entsprechende Lösungen oft quer zu üblichen Sichten, wie Komponentenzerlegungen, liegen. Es ist zur Zeit eine offene Frage, wie bzw. ob sich der Grundsatz *Separation of Concerns* der Softwaretechnik auf Sicherheit anwenden läßt. Darüber hinaus ist unklar, ob sich die Sicherheit eines Systems aus den Qualitätseigenschaften seiner Komponenten vorhersagen lässt, wie es für andere nicht-funktionale Eigenschaften, wie Zuverlässigkeit oder Verfügbarkeit, üblich ist.

Die Organisatoren hoffen, den Workshop auch 2006 zu veranstalten. Sie verbinden dies mit dem Wunsch, dass sich die klassische Sicherheits-Community stärker einbringt, da der Workshop 2005 durch Beiträge der Softwaretechnik dominiert wurde. Weitere Informationen finden sich unter `http://homes.dico.unimi.it/~monga/sess05.html`.

3 WADS

Der bereits vierte *Workshop on Architecting Dependable Systems* bemüht sich, die Communities der Software Architektur und der Verlässlichkeit zusammenzubringen. Thematisch konzentriert er sich auf die architekturellen Prinzipien von verlässlichen Systemen und deren Evaluation.

Der Workshop war zeitlich und thematisch in vier Sitzungen geteilt, die alle am 17. Mai 2005 stattfanden. Jedem Vortrag war ein zeitlicher Rahmen von 15 min inkl. Fragen gesetzt. Im Anschluss an jede Sitzung gab es zusätzlich 30 min für weitere Fragen und Diskussionen.

Die Organisatoren Rogério de Lemos und Alexander Romanovsky erhielten 24 Einreichungen von denen 13 akzeptiert wurden. Thematisch war der Workshop breit angelegt mit Beiträgen von der grafischen Darstellung von Entwurfsabwägungen bis hin zu Optimierungen von Tests auf architektureller Ebene. Ein großer Schwerpunkt lag allerdings bei dynamischen Architekturen und deren Transformationen bzw. Rekonfigurationen.

Als eingeladener Vortrag berichtete John C. Knight in *Assured Reconfiguration: An Architectural Core For System Dependability* über die Zukunft der Entwicklung verlässlicher Systeme. Er sieht eine Explosion der Komplexität aktueller Systeme, die durch eine Steigerung des Umfangs von Hard- und Software verursacht wird. Die verbesserten Produktionstechniken steigern zwar die Zuverlässigkeit von Hardware, allerdings leidet sie durch die steigende Komplexität zunehmend an Entwurfsfehlern. Diese Fehler werden nicht durch klassische Mittel der Fehlertoleranz, wie z.B. Replikation, abgefangen, so dass in Zukunft neue Ansätze notwendig werden. Für Software sieht er einen Mangel an Methoden, die für die Entwicklung verlässlicher Systeme notwendig sind. Die meisten Systeme benötigen allerdings nicht für die komplette Funktionalität starke Zusicherungen, sondern können auch zeitweilig mit reduziertem, aber sicherem Funktionsumfang die gestellten Anforderungen erfüllen. Daher schlägt er einen Ansatz zu *gesicherten Rekonfigurationen* vor, mit dem sich die Komplexität und die Kosten von Systemen reduzieren lassen. Zentrale Idee dabei ist der Einsatz von Software-Komponenten, die *fail-stop* Verhalten haben und im Fehlerfall rekonfiguriert werden, unter der Annahme solche Komponenten seien leichter zu entwickeln als formal verifizierte.

Die erste Sitzung *Verification and Validation* wurde von Alexander Romanovsky geleitet und bestand aus einem Vortrag *Towards Software Architecture-based Regression Testing* von Henry Muccini. Er stellte einen Ansatz zu Regressionstests sich weiterentwickelnder Architekturen vor, der die Kosten der Testdurchläufe reduzieren soll.

Die zweite Sitzung *Rigorous Design*, geleitet von Debra Richardson, begann mit einem Vortrag von Ji Zhang über *Specifying Adaptation Semantics*. Der Ansatz beschreibt die formale Semantik von dynamischen Adaptionen auf Basis einer Erweiterung der temporalen Logik. Der Fokus lag insbesondere auf Spezifikationen in frühen Phase der Systementwicklung, die in späteren Phasen korrekt implementiert werden können.

Fernando C. Filho stellte in *A Framework for Analyzing Exception Flow in Software Architectures* das Framework *Aeral* vor, das mit einem leichtgewichtigem formalen Modell den Fluss von Ausnahmen auf architektureller Ebene analysieren kann. Aeral ermöglicht die Spezifikation und Durchsetzung von Regeln für den Ausnahmefluss mit Hilfe einer relationalen Sprache erster Ordnung und der Architekturbeschreibungssprache *ACME*.

Unter dem Titel *Improving System Dependability by Enforcing Architectural Intent* beschrieb Marwan Abi-Antoun eine Abbildung zwischen verschiedenen Abstraktionsebenen einer komponentenbasierten Architektur. Die Abbildung soll beide Ebenen synchron halten, so dass sichergestellt ist, dass die niedrigere Ebene die Absichten einhält, mit denen die höhere Ebene bearbeitet wurde. Auch dieser Ansatz beruht auf der Architekturbeschreibungssprache ACME.

Lihua Xu stellte in *An Architectural Pattern for Non-functional Dependability Requirements* zunächst eine Klassifikation für Anforderungen vor, die eine Unterteilung in funktionale, operationalisierbare nicht-funktionale und überprüfbare nicht-funktionale Anforderungen vornimmt. Diese Klassifikation wird in einem vorgestellten Architekturmuster genutzt, um verschiedene Arten von Anforderungen durch verschiedene Arten von Architekturelementen umzusetzen.

Die dritte Sitzung *Fault Tolerance* wurde von Philip Koopman geleitet. Für die anschließende Diskussionsrunde bat er im Gegensatz zu den anderen Sitzungen die anwesenden Betreuer auf die Bühne, um Fragen zu den Arbeiten der präsentierenden Doktoranden bzw. PhD Studenten zu beantworten. Die Vortragsreihe wurde von Daniela Schilling mit *Computing Optimal Self-Repair Actions: Damage Minimization versus Repair Time* gestartet. Darin erläuterte sie ein Modell mit dem optimale Rekonfigurationen nach dem Auftreten von Fehlern berechnet werden. Der heuristische Ansatz adressiert das Redeployment von Software-Komponenten auf einer Menge zur Verfügung stehender Hardware-Komponenten mit Hilfe von Constraint Solvern.

John Georgas präsentierte in *Architectural Runtime Configuration Management in Support of Dependable Self-Adaptive Software* einen Ansatz zur Beobachtung, Kontrolle und Visualisierung von selbst-anpassenden Systemen. Die Visualisierung soll das Verständnis des Benutzers für die automatischen Adaptionen erhöhen und damit die Vertrauenswürdigkeit eines Systems steigern.

Der Vortrag *Architectural Support for Mode-Driven Fault Tolerance in Distributed Applications* von Deepti Srivastava beschrieb die Idee für unterschiedliche Betriebsmodi jeweils passende Fehlertoleranz-Strategien einzusetzen. Dazu präsentierte sie ein Spezifikations-Framework und eine entsprechende Architekur, die an einem System für unbemannte Fluggeräte evaluiert wurde.

Unbemannte Fluggeräte sind auch die Domäne von Osamah A. Rawashdehs Präsentation *A UAV Test and Development Environment Based on Dy-*

namic System Reconfiguration. Er beschrieb darin ein Framework für Fluggeräte, das Software-Komponenten mit Abhängigkeitsgraphen modelliert und im Fehlerfall eine Rekonfiguration bzgl. der Abbildung auf Hardware-Komponenten vornimmt.

Jennifer Morris leitete die vierte und letzte Sitzung *System Evaluation* mit Ivica Crnkovic als Vorsitzendem ein. Mit *Representing Design Tradeoffs in Safety-Critical Systems* stellte sie eine Möglichkeit vor, Abwägungen während des Entwurfsprozesses grafisch mit Kiviat-Diagrammen zu visualisieren. Mit diesen Visualisierungen sollen Ähnlichkeiten von Eigenschaften verschiedener Anwendungsdomänen gefunden werden, um so zu prüfen, ob ihre Umsetzungen auf die jeweils anderen Domäne anwendbar sind. Dies wurde anhand der Domänen Schienen- und Luftverkehr illustriert.

Unter dem Titel *Sensitivity Analysis for a Scenario-Based Reliability Prediction Model* präsentierte Genaína N. Rodrigues eine Sensitivitätsanalyse für Komponenten und Transitionen in Bezug auf deren Einfluss auf die Zuverlässigkeit eines Gesamtsystems. Die Analyse wird mit Hilfe von *Message Sequence Charts* durchgeführt und beruht auf Informationen über die Komponenten und Ausführungsszenarien.

Timo Warns stellte in *Availability Simulation of Peer-to-Peer Architectural Styles* ein konzeptionelles Framework für die Evaluation von P2P-Architekturstilen vor. Dabei lag der Schwerpunkt auf der Bestimmung des Einflusses der Stile auf die Verfügbarkeit von P2P-Diensten. Die grundsätzliche Idee besteht darin, die Architekturstile auf Modelle echter System abzubilden, die per Simulation evaluiert werden können.

Tim Kelly beendete den Workshop mit dem Vortrag *Failure Modelling in Software Architecture Design for Safety*. Er stellte einen Ansatz zur Modellierung von Fehlern auf Basis der Prozessalgebra *CSP* vor. Die vorgestellte Methode erlaubt eine Fehleranalyse eines Systems anhand dessen Architektur.

Im Anschluss an die Sitzungen gab es jeweils Raum für Diskussionen über die Themen der eigentlichen Vorträge hinaus. Angeregt durch die Voraussetzung einiger vorgestellter Arbeiten, dass die jeweils eingesetzten Software-Komponenten *fail-stop* Verhalten haben, wurde diskutiert, ob diese Bedingung mit den heutigen Mitteln ohne Probleme erfüllbar sei. Insbesondere unvorhergesehene Zustände, die u.a. durch böswilliges Verhalten herbeigeführt werden könnten, werden noch nicht ausreichend berücksichtigt.

Entsprechend dem Thema des Workshops setzten sich die meisten Vorträge mit Aspekten von Architekturen auseinander. Als große Herausforderung wird aber weiterhin der Mangel an Methoden zur formalen Analyse auf architektureller Eben gesehen. In dem Zusammenhang wurde insbesondere auf die Lücken hingewiesen, die die Softwaretechnik noch zu anderen Ingenieurswissenschaften aufweist. Andere Disziplinen haben es geschafft auch komplexe Systeme mit ausreichender Präzision vorhersagbar zu machen, während die Softwaretechnik dort noch große Mängel aufweist.

Aus den früheren WADS Workshops sind bereits zwei Bücher *Architecting Dependable Systems* und *Architecting Dependable Systems II* entstanden. Die Organisatoren möchten für 2005 eine Spezialausgabe des *Journal of Systems and Software* (JSS) mit erweiterten Beiträgen aus dem Workshop herausgeben, um über den aktuellen Stand der Forschung und Industrie zu berichten.

Der WADS Workshop wurde 2004 als Zwillingsveranstaltung zu den Konferenzen ICSE 2004 und der *International Conference on Dependable Systems and Networks* (DSN) veranstaltet. Die Organisatoren hoffen, auch in Zukunft wieder auf Zwillingsveranstaltungen, so dass der Workshop vermutlich auch 2006 wieder im Rahmen der ICSE stattfinden wird. Weitere Informationen befinden sich unter `http://www.cs.kent.ac.uk/events/conf/2005/wads/`.

4 Ausblick

Die ICSE 2006 wird vom 20. bis 28. Mai 2006 in Shanghai, China, stattfinden. Die Frist für Proposals von Workshops läuft erst im Oktober 2005 aus, so dass noch keine endgültigen Aussagen getroffen werden können, ob SESS und WADS dann wieder unter den Konferenz-Workshops sind. Die Aussagen der Organisatoren lassen aber darauf hoffen. Weitere Informationen zur ICSE 2006 befinden sich unter `http://www.isr.uci.edu/icse-06/`.