Kurzfassungen der Vorträge der

GI-Fachtagung
Software Engineering 2005

Universität Duisburg-Essen
Campus Essen

8. – 11. März 2005

Die "Software Engineering 2005" zielt auf die Zusammenführung und Stärkung der deutschsprachigen Softwaretechnik ab. Die Tagung bietet ein Forum zum intensiven Austausch über praktische Erfahrungen, wissenschaftliche Erkenntnisse sowie zukünftige Herausforderungen bei der Entwicklung von Softwareprodukten bzw. softwareintensiven Systemen. Sie richtet sich gleichermaßen an Experten aus Industrie und Wissenschaft.

Programmkomitee-Vorsitzende:
Peter Liggesmeyer, TU Kaiserslautern / IESE
Klaus Pohl, Uni Duisburg-Essen

Organisation:
Michael Goedicke, Uni Duisburg-Essen

Programmkomitee:
M. Broy, TU München
E. Denert, IVU
E.-E. Doberkat, Uni Dortmund
J. Ebert, Uni Koblenz
C. Ebert, Alcatel
G. Engels, Uni Paderborn
M. Goedicke, Uni Duisburg-Essen
V. Gruhn, Uni Leipzig
S. Jähnichen, TU Berlin
M. Jarke, RWTH Aachen
U. Kelter, Uni Siegen
J. Ludewig, Uni Stuttgart
M. Nagl, RWTH Aachen
F. Matthes, TU München
A. Oberweis, Uni Karlsruhe
B. Paech, Uni Heidelberg
F. Saglietti, Uni Erlangen-Nürnberg
W. Schäfer, Uni Paderborn
A. Schürr, TU Darmstadt
H.-W. Six, Fernuni Hagen
G. Snelting, Uni Passau
M. Winter, FH Köln
A. Zeller, Uni Saarbrücken
H. Züllighoven, Uni Hamburg

Eingeladene wissenschaftliche Vorträge

- W. F. Tichy: Die Bedeutung der Empirie für die Softwaretechnik
- St. Jähnichen: Modellierung gemischt kontinuierlich-diskreter Systeme

Eingeladene Praxispräsentationen

- R. E. Achatz: Optimierung der Softwareentwicklung in einem Industrieunternehmen am Beispiel Siemens
- Dr. F. Paulisch: Metriken zur effektiven Steuerung von software-basierten Systementwicklungen
- K. Beetz: SW-Qualität - aktuelle Herausforderungen in industriellen SW-Projekten
- K. Grimm: Software-Technologie in der Automobil-Industrie
- Dr. H. Dörr: Abhängigkeitsmanagement in der Systementwicklung
- Dr. F. Houdek: Erfahungen zum Requirement-Engineering-Prozess in der Steuergeräte-Entwicklung bei DaimlerChrysler

Forschungsarbeiten

Architektur

- P. Sochos, I. Philippow, M. Riebisch: Featuregesteuerte Architekturgestaltung zwecks Wartbarkeit und Evolution von Produktlinien
- J. Lankes, F. Matthes, A. Wittenburg: Architekturbeschreibung von Anwendungslandschaften: Softwarekartographie und IEEE Std 1471-2000
- R. Reussner, V. Firus, H. Koziolek, St. Becker, W. Hasselbring: Empirische Bewertung von Performanz-Vorhersageverfahren für Software-Architekturen

Testen

- J. Mayer: On Testing Image Processing Applications with Statistical Methods
- F. Belli, Ch. Budnik: Towards Minimization of Test Sets for Coverage Testing of Interactive Systems
- D. Sokenou: Ein UML-basierter Testansatz zum Klassen- und Integrationstest objektorientierter Systeme

UML

- J. Niere, U. Kelter, J. Wehren: A Generic Difference Algorithm for UML Models
- H. Störle, J. H. Hausmann: Towards a Formal Semantics of UML 2.0 Activities

Formale Methoden

- S. Glesner, O. Blech: Logische und softwaretechnische Herausforderungen bei der Verifikation optimierender Compiler
- R. Heckel, G. Lajios, S. Menge: Modulare Analyse stochastischer Graphtransformationssysteme

Spezifikationen

- P. Langendörfer, M. Lehmann: Implementation Independent Profiling of SDL Specifications
- St. Hanenberg, D. Stein, R. Unland: Eine Taxonomie für aspektorientierte Systeme
- M. Book, V. Gruhn: Modulare Spezifikation und Steuerung von Dialogen in Web-basierten Anwendungen

Erfahrungsberichte

Entwicklungsmethoden

- St. Löper, G. Schmidt:Softwareprozessmetriken und agile Methoden — eine industrielle Fallstudie
- M. Trapp, B. Schürmann, P. Wörz: Modellierung adaptiver eingebetteter Systeme
- M. Müller, F. Padberg, W. Tichy: Ist XP etwas für mich? Empirische Studien zur Einschätzung von XP

Architektur

- J.P. Richter: Wann liefert eine Serviceorientierte Architektur echten Nutzen?
- M. Verlage, J.-F. Girard, D. Ganesan, Th. Kiesgen: Überwachung der Qualität der Architektur einer Software-Produktlinie am Beispiel eines web-basierten Wertpapierinformationssystems
- R. Schäftlein, T. Mahler:MDA in der Praxis — Erfahrungen mit modellgetriebener Softwareentwicklung in Großprojekten bei der Karstadt Warenhaus AG

Praxisvorträge

- Ch. Thiel: Entwurf und Implementierung eines modernen industriellen Messsystems
- A. Möller, M. Schwarz: Geschäftsprozess-Monitoring in heterogenen IT-Landschaften
- J. Schlattmann:Von der Strategie zur Anwendung - Entwicklung langfristiger strategischer Rahmenbedingungen für die IT einer großen Versicherung
- F. Mang: Software-Entwicklung im Umbruch - Industrialisierung und Globalisierung
- G. Höfner: Aufbau eines Offshore Development Centers (OCD) — Herausforderungen und Erfolgsfaktoren
- A. Schmidt, R. Schönwald: Iterativ-inkrementelles Software-Engineering in komplexen IT-Projekten
- U. Schmedt: Vorgehensmodelle - Aufbau, Umsetzung und Weiterentwicklung in der Postbank Systems AG
- L. Wieske: Service-Orientierte Architektur in Applikations- und Integrations-Projekten
- G. Starke: Praktische Bewertung von Software-Architekturen
- M. Verlage: Aufbau und Wartung einer Software-Produktlinie in einem kleinen Unternehmen
- Ch. Ebert: Process Improvement — When the Fog Clears, Business Success Counts

Kurzfassungen der Vorträge[1]

R. E. Achatz: Optimierung der Softwareentwicklung in einem Industrieunternehmen am Beispiel Siemens

Mit mehr als 30.000 Software-Entwicklern und des starken Einflusses von Software auf den Geschäftserfolg — 60 % des Geschäftes — spielt Siemens in der Champions League der Software-Companies. Da ist natürlich eine möglichst optimale Softwareentwicklung von großer Bedeutung. Die wichtigsten Hebel um dies zu erreichen sind: Angemessene Prozesse, die richtige Struktur und die Nutzung von Synergien. Das Vorgehen bei Siemens wird in dem Vortrag beschrieben. Wichtig ist es auch, die Erfolge der Verbesserung messbar zu machen. Hierzu werden Beispiele vorgestellt. Da der größte Teil der Software bei Siemens 'embedded Software' ist, spielt das Thema Software-Qualität eine besonders große Rolle.

K. Beetz: SW-Qualität - aktuelle Herausforderungen in industriellen SW-Projekten

F. Belli, Ch. Budnik: Towards Minimization of Test Sets for Coverage Testing of Interactive Systems

A model-based approach for minimization of test sets for interactive systems is introduced. Test cases are efficiently generated and selected to cover the behavioral model and, complementarily, the fault model of the system under test (SUT). Structural features of the SUT are exploited in order to decrease the complexity of the introduced algorithms. The approach is extended to UML statecharts to increase its applicability.

M. Book, V. Gruhn: Modulare Spezifikation und Steuerung von Dialogen in Web-basierten Anwendungen

Die Benutzerfreundlichkeit von Web-Anwendungen leidet häufig unter der fehlenden Unterstützung hierarchischer Dialogsequenzen, an die Benutzer sich während der Arbeit mit fensterbasierten Benutzeroberflächen und ihren hierarchisch strukturierten Dialogfenstern bereits gewöhnt haben. Für Multikanal-Anwendungen liegt eine zusätzliche Herausforderung darin, die geräteunabhängige Geschäftslogik mit den gerätespezifischen Interaktionsmustern zu verbinden, die die unterschiedlichen Ein-/Ausgabefähigkeiten verschiedener Clients erforderlich machen. Um diese Probleme anzusprechen, stellen wir eine grafische Dialogfluss-Notation vor, die die Spezifikation hierarchischer Dialogsequenzen für verschiedene Präsentationskanäle ermöglicht. Nach Übersetzung in eine XML-basierte Dialogbeschreibungssprache eignet sich diese Spezifikation als Eingabe für ein Framework, das die Dialogflüsse komplexer Web-Anwendungen steuert.

Dr. H. Dörr: Abhängigkeitsmanagement in der Systementwicklung

Ch. Ebert: Process Improvement — When the Fog Clears, Business Success Counts

Simultaneously improving project management, product development and engineering processes is for many companies crucial to surviving in a fast changing environment. However, the integration of these activities often falls back due to methodology wilderness, lack of vision or organizational misalignment. To stay competitive with its systems and software development, Alcatel has put in place an orchestrated improvement program of its processes and the underlying engineering tools environment. We will in this overview present Alcatel s engineering process improvement initiatives over the past years. Results are impressive. For instance, with moving to a CMM(I) maturity level 3, we reduced delays of projects to half. We also pinpoint some challenges, such as the need of appraisal model stability in order to facilitate timeline benchmarking.

S. Glesner, O. Blech: Logische und softwaretechnische Herausforderungen bei der Verifikation optimierender Compiler

Korrektheit von Compilern ist notwendige Voraussetzung für die Korrektheit der damit übersetzten Software. Insbesondere optimierende Compiler sind oft fehlerhaft. In diesem Papier stellen wir nach einem Überblick über den Stand der Forschung unsere neuen Arbeiten zur Verifikation optimierender Compiler vor. Dabei diskutieren wir zum einen, welche logischen Probleme sich bei der formalen Verifikation von Übersetzungsalgorithmen in Compilern mittels Theorembeweisern ergeben und welche Lösungen wir dafür entwickelt haben. Zum anderen zeigen wir, wie man die Korrektheit auch realer optimierender Compiler mit beträchtlichem Implementierungsumfang sicherstellen kann. Damit tragen unsere Ergebnisse zur Korrektheit von Compilern, einem wichtigen

[1]Mit freundlicher Genehmigung der Gesellschaft für Informatik, Bonn, entnommen aus: Lecture Notes in Informatics (LNI) - Proceedings; Volumne P-64; ISBN 3-88579-393-8; ISSN 1617-5468

Werkzeug in der Softwaretechnik, bei. Außerdem entwickeln wir auf diese Weise Methoden, die auch in anderen Anwendungsbereichen zur Verifikation von Software eingesetzt werden können.

K. Grimm: Software-Technologie in der Automobil-Industrie

Die Automobilindustrie ist einer der Bereiche, die in den letzten Jahren besonders stark durch die industrielle Software-Revolution beeinflusst wurden. Immer mehr wettbewerbsrelevante Innovationen im Fahrzeug basieren auf Software. Sie gehen einher mit einer dramatischen Zunahme an Komplexität, steigendem Zeit- und Kostendruck und hohen Qualitätsanforderungen. Aus diesen Trends und Anforderungen ergeben sich eine Reihe von software-bezogenen Herausforderungen, die für die Automobilindustrie zukünftig von wettbewerbsentscheidender Bedeutung sein werden. Unabhängig davon, ob der Automobilhersteller die Software selbst entwickelt oder mit Zulieferern kooperiert, muss jeder Hersteller mit Premium-Anspruch spezifische Software-Kompetenzen haben beziehungsweise aufbauen. Wichtige Kompetenzfelder sind die Beherrschung des Software-Entwicklungsprozesses, das Software-Qualitätsmanagement unter Einbeziehung der Kooperation mit Zulieferern, die Gesamtarchitektur der Software-Systeme im Fahrzeug sowie die Fähigkeit, die Systeme zu spezifizieren, zu integrieren und zu testen. Aufgrund der großen Bedeutung der Software-Technologie wird in der Automobilindustrie intensiv am Aufbau und Ausbau der erforderlichen Software-Kompetenzen gearbeitet. Im DaimlerChrysler-Forschungslabor Software-Technologie werden innovative, leistungsfähige Ansätze erarbeitet, die sowohl den Bereich Architektur als auch den Bereich Prozesse, Methoden und Tools und die speziellen Aspekte der Systemsicherheit umfassen.

St. Hanenberg, D. Stein, R. Unland: Eine Taxonomie für aspektorientierte Systeme

Die aspektorientierte Softwareentwicklung entwickelte sich aus der Beobachtung heraus, dass eine Vielzahl logisch zusammenhängender Softwareelemente mit Hilfe konventioneller Techniken nicht modularisierbar ist. Eine mangelhafte Modularisierung wiederum reduziert die Verständlichkeit und Wartbarkeit der Software. Aspektorientierte Systeme bieten zusätzliche Konstrukte, um ein höheres Maß an Modularisierung zu ermöglichen. Es existiert bereits eine Reihe von Systemen, welche die aspektorientierte Softwareentwicklung in unterschiedlichsten Ausprägungen unterstützen. Es fehlt jedoch an Kriterien, anhand derer überprüft werden kann, ob ein bestimmtes System die Modularisierung eines gegebenen Anwendungsproblems ermöglicht. In diesem Papier schlagen wir eine Taxonomie aspektorientierter Systeme vor. Anhand derer können zum einen unterschiedliche aspektorientierte Systeme miteinander verglichen werden. Zum anderen ermöglicht die Taxonomie es festzustellen, ob ein System zur Lösung eines gegebenen Problems geeignet ist.

R. Heckel, G. Lajios, S. Menge: Modulare Analyse stochastischer Graphtransformationssysteme

Analog zu stochastischen Petrinetzen oder Prozesskalkülen wurden stochastische Graphtransformationssysteme entwickelt, um funktionale und nichtfunktionale (insbesondere zeitabhängige und probabilistische) Aspekte von Softwaresystemen integrieren und auf hohem Abstraktionsniveau beschreiben zu können. Diese Kombination eignet sich vor allem für die Modellierung und Analyse mobiler Anwendungen, wo dynamische Rekonfiguration von Netzen an der Tagesordnung ist. Mit Hilfe von Analysewerkzeugen wie stochastischen Modelcheckern ist es möglich, Eigenschaften dieser Modelle zu überprüfen. Um diese Technik auf Systeme interessanter Größe anwenden zu können, befasst sich die vorliegende Arbeit mit der Modularisierung der stochastischen Analyse. Insbesondere wird die Korrektheit eines entsprechenden modularen Erzeugungsverfahrens gezeigt und von Experimenten berichtet, die die Überlegenheit des modularen Ansatzes belegen.

G. Höfner: Aufbau eines Offshore Development Centers (OCD) — Herausforderungen und Erfolgsfaktoren

Siemens Information Systems Ltd. (SISL) ist das System- und Softwarehaus von Siemens in Indien mit Schwerpunkt auf Software Offshore Development. Nach einer kurzen Vorstellung von SISL wird in dem Vortrag auf Basis einer Definition der Begriffe Offshoring und Outsourcing erläutert, welche Motivation in der Regel hinter dem Aufbau eines Offshore Development Center steckt. Da das Thema Kostenersparnis oft eine wichtige Rolle spielt, erweist es sich als wichtig, neben den reinen Stundensätzen versteckte Kosten, wie beispielsweise zusätzlicher Kommunikationsaufwand oder erhöhte Reisekosten, aufzuzeigen. Die Herausforderungen beim Aufbau eines Offshore Development Centers, die sogenannten Entfernungseffekte, lassen sich allgemein in drei Kategorien einteilen: Kultur, Zeit, und Geographie. Diese Herausforderungen können durch gezielte Maßnahmen wie interkulturelles Training (Kultur), multisite-fähiges Konfigurationsmanagement (Zeit) oder frühzeitige Beschaffung der für etwaige Reisen notwendigen Dokumente (Geographie) gemeistert werden. Auf ein Thema, das häufig unterschätzt wird, legt der Vortrag besonderen Wert: Die Berücksichtigung der interkulturellen Unterschiedlichkeiten. Beim Aufbau eines Offshore Development Centers gibt es darüber hinaus noch eine Reihe weiterer Erfolgsfaktoren: Gewisse Mindestanforderungen bzgl. des Reifegrads der jeweiligen Software-Entwicklungsprozesse, die Bereitstellung einer angemessenen Infrastruktur, die Auswahl der richtigen Mitarbeiter, der Aufbau der notwendigen Domänenkompetenz, die Auswahl von für Offshoring geeigneter Arbeitspakete, etc. Um sicherzustellen, dass der Aufbau eines Offshore Development Centers erfolgreich verläuft, hat SISL eine systematische Vorgehensweise entwickelt. Diese Vorgehensweise wird allgemein und anhand konkreter Beispiele präsentiert.

Dr. F. Houdek: **Erfahungen zum Requirement-Engineering-Prozess in der Steuergeräte-Entwicklung bei DaimlerChrysler**

St. Jähnichen: **Modellierung gemischt kontinuierlich-diskreter Systeme**

Das Verhalten eingebetteter Systeme lässt sich zumeist nur als eine Mischung von Komponenten verstehen, deren Verhalten durch kontinuierliche und diskrete Anteile beschrieben ist. Im diskreten Fall sind Zustandsautomaten in unterschiedlicher Ausprägung ein adäquates Modellierungsmittel, für kontinuierliches Verhalten verwenden wir naturgemäß Differential- und algebraische Gleichungen und modellieren damit u.a. Regelungsverhalten oder physikalische Effekte. In beiden Fällen lassen sich die modellierten Verhalten mit Hilfe objektorientierter Techniken aus einfacheren Teilverhalten konstruieren. Die resultierenden Modellstrukturen repräsentieren dabei idealerweise die Strukturen des realen Systems. Der Vortrag beschreibt eine Modellierungstechnik, die beide Verhaltensarten integriert und zusätzlich in Abhängigkeit vom Systemzustand eine Verhaltenssimulation in unterschiedlicher Detaillierungstiefe unterstützt. Die Verhaltensdynamik wird damit abhängig von der Struktur des Systems und wir sprechen von Modellstrukturdynamik. Als Notation verwenden wir MOSILA, eine Erweiterung der objektorientierte Modellierungssprache Modelica.

P. Langendörfer, M. Lehmann: **Implementation Independent Profiling of SDL Specifications**

The Specification and Description Language (SDL) is a worldwide accepted standard for development of new protocols, e.g. an increasing number of IEEE protocol standards uses SDL. Especially for mobile devices it is important that the protocol implementation is as energy efficient as possible. This cannot be achieved with a straight forward code generation from SDL to a target language as for example C. Thus, the normal way is that after the specification is done, parts of the protocol are implemented by hand using C or even VHDL. In this paper we are presenting our profiling tool, profSDL, which is designed to support engineers in determining the partitioning of the SDL specification. profSDL allows to determine static effort and dynamic effort of the protocol under development. The dynamic effort can be assigned to the protocol itself as well as to the SDL runtime environment. We applied profSDL to a fully functional SDL model of IEEE802.11a. The design decisions indicated by profSDL matched exactly what the designers decided manually. The major benefit of profSDL is that it works on SDL level, so that the profiling results are independent of any potential implementation decision e.g. programming language features.

J. Lankes, F. Matthes, A. Wittenburg: **Architekturbeschreibung von Anwendungslandschaften: Softwarekartographie und IEEE Std 1471-2000**

Jenseits der Beschreibung einzelner Softwaresysteme gewinnt die Beschreibung von Anwendungslandschaften, bestehend aus hunderten oder tausenden von Informationssystemen, zunehmend an Bedeutung. In unserem Forschungsprojekt Softwarekartographie haben wir den State of the Art zur Beschreibung von Anwendungslandschaften in der Praxis aufgenommen und entwickeln in Zusammenarbeit mit unseren Projektpartnern Methoden und (graphische) Modelle zur Beschreibung, Bewertung und Gestaltung von Anwendungslandschaften [MW04a; MW04b; LMW05]. In diesem Beitrag untersuchen wir den Begriffsapparat des IEEE Std 1471-2000 "IEEE Recommended Practice for Architectural Description of Software-Intensive Systems" systematisch auf seine Eignung zur Architekturbeschreibung komplexer Anwendungslandschaften. Der Nutzen aus den standardisierten Begriffsdefinitionen des IEEE 1471 sowie den dort vorgestellten

Best-Practice-Vorgehensweisen soll auch für das Management von Anwendungslandschaften zugänglich gemacht werden. Dazu identifizieren wir Stärken und Schwächen des Standards und schlagen eine Erweiterung des Begriffsapparats vor, um insbesondere den kritischen Prozess der Informationsbeschaffung adäquat zu modellieren.

St. Löper, G. Schmidt:Softwareprozessmetriken und agile Methoden — eine industrielle Fallstudie

Vor einigen Jahren sind im Bereich der Softwareentwicklung agile Softwareentwicklungsprozesse entworfen worden, welche u.a. für den Entwicklungsprozess schnellere Entwicklungszeiten und Flexibilität als Vorteile versprechen und damit einen Ansatz darstellen, die Nachteile traditioneller Entwicklungsmethoden zu überwinden. Softwaremetriken definieren Kenngrößen eines Software-Produktes und/oder -Prozesses, die gemessen werden. Sie werden als integraler Bestandteil und Stärke der traditionellen Softwareentwicklungsmethoden angesehen. Die Anwendung von Prozessmetriken auf agile Softwareentwicklungsmethoden ist bis zum heutigen Zeitpunkt nur geringfügig untersucht. Es existieren einige Prozessmetriken für agile Methoden. Inwieweit dies Metriken Aussagen über die Effzient agiler Softwareentwicklungsmethoden treffen und sich daraus Optimierungspotenziale bei der Methode ableiten lassen, wird in einer industriellen Fallstudie untetrsucht. Dabei werden empirische Daren präsentiert und analysiert, die auf der Grundlage von ausgewählten Prozessmetriken gesammelt wurden.

F. Mang: Software-Entwicklung im Umbruch - Industrialisierung und Globalisierung

Die Softwareindustrie wie auch die IT-Abteilungen von Unternehmen befinden sich nach dem Platzen der eCommerce-Blase in einem für diese Branche noch nicht dagewesenen Umbruch. Wie bisher nur in anderen Branchen wie der Automobilindustrie beobachtet, wird das "Erwachsenwerden" der IT von drei wesentlichen Faktoren bestimmt: Konsolidierung, Industrialisierung und Globalisierung. Der Faktor Konsolidierung war in den vergangenen Jahren und ist noch jeden Tag ein sehr aktuelles Thema in der Presse. Übernahmeschlachten wie zwischen Oracle und Peoplesoft und die regelmäßige Bekanntgabe von Aufkäufen der großen Softwareanbieter wie Microsoft, IBM, SAP und anderen lassen die Anzahl der Marktteilnehmer in verschiedenen Teilen der Softwareindustrie deutlich schrumpfen. Aber auch im IT-Dienstleistungsbereich sind ähnliche Entwicklungen zu beobachten. Konzerne lagern ihre IT-Bereiche in einem ersten Schritt in eigenständige Firmen aus und verkaufen diese zu einem späteren Zeitpunkt häufig. Das alles geht einher mit der lange ersehnten und erwarteten Industrialisierung der IT. Nach langen Jahren sind die Prozesse und die Tools für eine professionelle Entwicklung von Software endlich in einem Zustand, in dem sie für ingenieurmäßiges Vorgehen und eine Professionalisierung des gesamten Produkt- (und Lösungs-) Lebenszyklus eingesetzt werden können. Unterstützt wird dies noch durch zahlreiche Standards, die in diesen Bereichen inzwischen existieren. Das alles ist mit erheblichen Investitionen auch in die Mitarbeiter verbunden, was wiederum die Konsolidierung weiter vorantreibt.

J. Mayer: On Testing Image Processing Applications with Statistical Methods

Testing image processing applications is very ressource consuming. Many complex images have to be generated as test inputs and the expected resulting outputs have to be determined to complete the test cases. The present paper deals with this challenge in testing implementations of image operations, namely dilation. It applies random testing using models from stochastic geometry for random input generation. The Statistical Oracle, a modification of the well-known Heuristic resp. Parametric Oracle, is used to compare the results. Therefore, reliability predictions are also possible.

A. Möller, M. Schwarz: Geschäftsprozess-Monitoring in heterogenen IT-Landschaften

Der wirtschaftliche Erfolg eines Unternehmens hängt nicht zuletzt von einem umfassenden Geschäftsprozessmanagement ab. Daher besteht für viele Unternehmen die Notwendigkeit, ihre IT-Systeme auf Geschäftsprozessebene miteinander zu verbinden. Da jedoch die heutigen IT-Landschaften in der Regel über Jahre gewachsen sind und die einzelnen Anwendungssysteme auf verschiedenen Softwaretechnologien basieren, erweist sich das Fehlerhandling in den Geschäftsprozessen als Herausforderung. Dieser Praxisvortrag stellt das Konzept eines fachlichen Transaktionsmonitors vor, der es ermöglicht, Geschäftsprozesse über Systemgrenzen hinweg zu verfolgen und im Fehlerfall eine aussagekräftige Fehlermeldung an den Initiator des Geschäftsprozesses zurückzumelden. Folgende Ziele und Randbedingungen spielten bei der Realisierung eine Rolle: Es galt eine technische und fachliche

Möglichkeit zu finden, die beteiligten Systeme und die auf ihnen implementierten Teil-Geschäftsprozesse möglichst einheitlich zu überwachen, zeitlich voneinander entkoppelte Systeme (Batchbetrieb) sollten berücksichtigt werden, das neue System sollte Schritt für Schritt integrierbar sein. Als Lösung wurde ein Transaktionsmonitor integriert, welcher als Dienstleister verschiedene Services anbietet. Die wichtigsten geschäftsprozessinitiierenden Systeme wurden insoweit angepasst, dass fachliche Transaktionen mit Hilfe des Monitors eröffnet werden und dass im Falle eines Fehlers der Transaktionsmonitor benachrichtigt wird. Über geeignete Auswertungen können fehlerhafte Geschäftsprozesse identifiziert und an Korrekturteams übergeben werden. Diese Teams identifizieren anhand der gesammelten Transaktionsmonitor-Informationen den Initiator des Geschäftsprozesses, ermitteln die zugehörigen Daten und den Fehlerkontext zum Fehlerfall. Auf diese Weise lassen sich kurzfristig fehlerhafte Prozesse korrigieren und langfristig Fehlerquellen identifizieren und beseitigen.

M. Müller, F. Padberg, W. Tichy: Ist XP etwas für mich? Empirische Studien zur Einschätzung von XP

Agile Software-Entwicklungsmethoden, und speziell Extreme Programming (XP), haben große Aufmerksamkeit unter Praktikern und Forschern auf sich gezogen. Dieser Artikel bietet eine Gesamtschau der bis jetzt bekannten, empirischen Erkenntnisse zu XP, mit dem Ziel, dem Praktiker fundierte Entscheidungshilfen zu bieten sowie die Lücken in der wissenschaftlichen Untersuchung aufzuzeigen. Verhältnismäßig gut untersucht sind die Praktiken der Paarprogrammierung und der testgetriebenen Entwicklung; weitere Techniken von XP wurden bis jetzt nur in Erfahrungsberichten evaluiert. Langzeituntersuchungen über die Nachhaltigkeit der XP-Praktiken stehen ebenfalls noch aus.

J. Niere, U. Kelter, J. Wehren: A Generic Difference Algorithm for UML Models

It is state-of-the-art to use the Uni ed Modelling Language (UML) to describe software system models. In order to support cooperative team work a version management system which supports UML models is absolutely necessary. The essential part of such systems is the ability to calculate differences and present them to the developer. This paper presents an approach for computing differences between UML models encoded as XMI les. In contrast to our previous work, we present a generic approach, with which we are able to cover a broad range of UML diagram types. It also does not require persistent identi ers of diagram elements. Our prototype implementation shows, that our difference algorithm used leads to good runtimes in the case of small documents and acceptable runtimes in the case of large documents. Overall, it has a very low error rate, i.e. the quality of the differences is almost optimal.

Dr. F. Paulisch: Metriken zur effektiven Steuerung von software-basierten Systementwicklungen

R. Reussner, V. Firus, H. Koziolek, St. Becker, W. Hasselbring: Empirische Bewertung von Performanz-Vorhersageverfahren für Software-Architekturen

Die Architektur eines Software-Systems beeinflusst maßgeblich seine Qualitätseigenschaften wie Performanz oder Zuverlässigkeit. Daher sind Architekturänderungen oft die einzige Möglichkeit, Mängel bei diesen Qualitätseigenschaften zu beheben. Je später diese Änderungen an der Architektur während des Software-Entwicklungsprozesses vorgenommen werden, desto teurer und riskanter sind sie. Aus diesem Grund ist eine frühzeitige Analyse verschiedener Architektur-Entwurfsalternativen bezüglich ihrer Auswirkungen auf Qualitätseigenschaften vorteilhaft. Dieser Artikel beschreibt die Evaluation dreier verschiedener Performanz-Vorhersageverfahren für Software-Architekturen hinsichtlich ihrer Eignung, korrekte Empfehlungen für frühzeitige Entwurfsentscheidungen zu geben. Zusäatzlich sollen diese Vorhersageverfahren prüfen, ob extern vorgegebene Performanz-Anforderungen realisierbar sind. Die Performanz- Vorhersageverfahren SPE , Capacity Planning und umlPSI wurden empirisch durch 31 Teilnehmer untersucht, die eine Menge vorgegebener Alternativen beim Entwurf der Architektur eines Webservers zu bewerten hatten. Die Ergebnisse zeigen, dass Entwurfsalternativen mit allen Verfahren richtig bewertet wurden, sofern deutliche Auswirkungen auf die Performanz vorhanden waren. Ohne den Einsatz der Performanz-Vorhersageverfahren wurden häufiger weniger performante Entwurfsalternativen vorgeschlagen. Darüber hinaus konnte das Verfahren Capacity Planning die absoluten Werte bei den meisten Entwurfsalternativen relativ genau vorhersagen.

J.P. Richter: Wann liefert eine Serviceorien-

tierte Architektur echten Nutzen?

Das Schlagwort der Serviceorientierten Architektur (SOA) beherrscht gegenwärtig die Diskussion um die Gestaltung unternehmensweiter Anwendungslandschaften. Unternehmen erwarten sich von einer SOA die Reduktion von IT-Kosten und eine höhere Flexibilität ihrer Geschäftsprozesse. Der Begriff SOA an sich ist aber zu ungenau definiert, um konkrete Hilfestellungen bei der Fortentwicklung der Unternehmens-IT geben zu können. Der Beitrag benennt Voraussetzungen, um den erwarteten Nutzen zu erzielen, und beleuchtet dabei sowohl technische als auch fachliche Aspekte. Wichtiger als die Einführung einer technischen Infrastruktur ist der Aufbau einer fachlichen Architektur für die Anwendungslandschaft.

R. Schäftlein, T. Mahler:**MDA in der Praxis — Erfahrungen mit modellgetriebener Softwareentwicklung in Großprojekten bei der Karstadt Warenhaus AG**

J. Schlattmann:**Von der Strategie zur Anwendung - Entwicklung langfristiger strategischer Rahmenbedingungen für die IT einer großen Versicherung**

Die IT-Abteilungen werden anhand der momentan geleisteten Unterstützung der Endanwender gemessen. Deshalb ist Stabilität und Dauerhaftigkeit der Leistungserbringung das Bewertungskriterium der Nutzer der IT-Systeme. Projekte sorgen dafür, dass die notwendigen Leistungen in Form von Hardware, Infrastruktur und Anwendungen in vertretbarer Zeit entwickelt und bereitgestellt werden. Neben diesen eher "kurzfristigen" operativen Aktivitäten wird der langfristige Erfolg der IT durch die Einhaltung und zukunftsfähige Orientierung der strategischen Rahmenbedingungen in einer immer komplexer werdenden Welt sichergestellt. Dabei haben sich alle Aktivitäten und Rahmenbedingungen kompromisslos an den Zielen des Unternehmens zu orientieren. Effizienz ("die Dinge richtig tun") ist wichtig für die Akzeptanz der IT im Unternehmen. Effizienz nützt jedoch nichts ohne eine gesteuerte Fokussierung. Effektivität (im Sinne von "die richtigen Dinge tun") ist somit für die Überlebensfähigkeit der IT im Unternehmen essentiell. Grundlage einer effektiven IT ist ihre strategische Ausrichtung im Wechselspiel von Geschäftszielen, Technologieentwicklung, Marktgeschehen und Innovationsfähigkeit im Unternehmen. Wesentliche Elemente einer strategischen Ausrichtung sind die eigentliche Strategie und die daraus abgeleiteten Bebauungspläne, die Grundlage der operativen Architekturen (Facharchitektur, Anwendungsarchitektur, Systemarchitektur und Servicearchitektur) sind. Im Vortrag sollen auf der Basis der praktischen Erfahrungen in einer mehr als 10-jährigen Strategie- und Architekturentwicklung beim LVM die Vorgehensweisen, Möglichkeiten, Grenzen und mögliche Vermittlungswege aufgezeigt werden. Dabei soll insbesondere der Zusammenhang zwischen den externen Einflüssen, interner Architekturentwicklung und Umsetzung in die Praxis der Anwendungsentwicklung dargestellt werden.

U. Schmedt: **Vorgehensmodelle - Aufbau, Umsetzung und Weiterentwicklung in der Postbank Systems AG**

Die Postbank Systems AG ist der IT-Dienstleister der Deutsche Postbank AG. Das Leistungsangebot umfasst alle Produkte des IT-Betriebs sowie alle IT-Projekte. Sie beschäftigt mehr als 1000 Mitarbeiter. Gemeinsam mit SAP hat die Postbank Systems die erste Standardsoftware für große Retailbanken entwickelt und eingeführt. Neben weiteren Banking-Modulen steht die Entwicklung und Einführung einer ZV-Plattform und eines elektronischem Archivs auf der Agenda. Auf Basis der neuen Transaktionsplattform stellt die Postbank im neuen Geschäftsfeld Transaction Banking anderen Banken Zahlungsverkehrskapazitäten und Mehrwertdienste als Dienstleistung zur Verfügung. Die zunehmenden Anforderungen an die Qualität der Entwicklungsergebnisse und -prozesse machte die Entwicklung eines neuen Vorgehensmodells für die Postbank Systems erforderlich. Im Jahre 2002 wurde das Vorgehensmodell der Postbank Systems (PBVGM) erarbeitet und unternehmensweit in Kraft gesetzt. Das PBVGM basiert im Wesentlichen auf dem Industriestandard Rational Unified Process (RUP). Zusätzlich werden bewährte Regelungen anderer Vorgehensmodelle übernommen, um Entwicklungsszenarien für die Einführung von Standard-Software sowie die strukturierte Entwicklung abzudecken. Die Umsetzung und Pilotierung des Vorgehensmodells erfolgte ab dem Jahre 2003 gezielt in ausgewählten Projekten. Die Erfahrungen aus der Pilotierung wareninsgesamt positiv. Gleichzeitig machten die in diesen Projekten gesammelten Erfahrungen — in Verbindung mit der Reorganisation der Postbank Systems — eine umfassende Weiterentwicklung des Vorgehensmodells notwendig. So wurden die Themen Projektplanung, Betriebseinführung sowie Architektur in enger Abstimmung mit den am Entwicklungsprozess beteiligten Personen in der neuen Version 2.0 klarer herausgearbeitet. In dem Vortrag werden die wesentlichen Inhalte und Sichten auf das Vorgehensmodells dargestellt. Es wird die aufbauorganisatorische Verankerung beschrieben sowie über die Erfahrungen bei der Umsetzung des Prozesses in der Postbank Systems berichtet. Daraus werden kritische Erfolgsfaktoren bei der Umsetzung von Vorgehensmodellen abgeleitet sowie ein kurzer Ausblick auf die nächsten Schritte in der Postbank Systems AG gegeben.

A. Schmidt, R. Schönwald: **Iterativ-inkrementelles Software-Engineering in komplexen IT-Projekten**

Agile und iterativ-inkrementelle Vorgehensmodelle für die Software-Entwicklung sind in der Regel auf kleinere Vorhaben beschränkt. Für größere Entwicklungsprojekte müssen Prozesse und Methoden neu priorisiert und angepasst werden. Anforderungsmanagement, Planungs- und Fortschrittskontrolle, stabile Architekturvorgaben sowie die Berücksichtigung des Kommunikationsbedarfes sind in großen Projekten wesentliche Schwerpunkte für eine erfolgreiche Anwendung iteraktiv-inkrementeller Vorgehensweisen. Ein Praxisbericht demonstriert Ansatz und Realisierung der Methoden.

P. Sochos, I. Philippow, M. Riebisch: **Featuregesteuerte Architekturgestaltung zwecks Wartbarkeit und Evolution von Produktlinien**

Die Wartung und Weiterentwicklung von Software-Produktlinien stellen wegen des Umfangs und der angestrebten Nutzungsdauer herausfordernde Aufgaben dar. Die Zerlegung gemäß Separation of Concerns bildet ein grundlegendes Prinzip zur Verbesserung der Wartbarkeit eines Systems, weil es zu starker Kapselung und zu geringer Kopplung von Komponenten führt. Features stehen bei der Entwicklung von Produktlinien wichtige Abstraktionen zur Verfügung. Ihre direkte Verbindung mit der Architektur trägt zu vereinfachter Wartung und Evolution bei. In der hier vorgestellten Methode Feature-Architecture Mapping (FArM) werden Features zur Steuerung der Architekturentwicklung verwendet. Dieser Beitrag stellt die Umsetzung von (nicht-funktionalen) Qualitäts-Features durch die Methode FArM vor. Solche Qualitäts-bezogenen Features lassen sich üblicherweise nicht direkt in Architekturmerkmale abbilden; Defizite der Wartbarkeit sind eine häufige Folge. Das Vorgehen bei der Umsetzung wird anhand einer Produktlinie für mobile Kommunikationsgeräte erläutert, die Komponenten mit Plug-In-Schnittstellen aufweist.

D. Sokenou: **Ein UML-basierter Testansatz zum Klassen- und Integrationstest objektorientierter Systeme**

Vorgestellt wird eine Technik für den Klassen- und Integrationstest objektorientierter Systeme. Es werden Sequenzdiagramme und UML-Statecharts zur Testfallgenerierung und Statecharts und OCL-Constraints zur Generierung des Testorakels kombiniert. Zur Einbindung des Testorakels in das zu testende System werden aspektorientierte Programmiertechniken verwendet.

G. Starke: **Praktische Bewertung von Software-Architekturen**

Viele IT-Projekte haben Schwierigkeiten, die Güte oder Qualität ihrer Software- und Systemarchitekturen zu bewerten. Damit fällt es solchen Projekten schwer, die Erfüllung nichtfunktionaler Anforderungen wie Flexibilität, Verfügbarkeit und Performance a priori zu beurteilen. Der Vortrag zeigt, wie Sie bei der Bewertung von Architekturen methodisch vorgehen können. Er beantwortet aus praktisch/pragmatischer Sicht, was hierfür geeignete Kriterien und Werkzeuge sind. Kontrastiert wird dieser Ansatz zu anderen Bewertungsmethoden, unter anderem ATAM und SAAM, insbesondere werden deren Probleme bei der praktischen Umsetzung dargestellt. Als illustrierende Beispiele dienen einige große betriebswirtschaftliche Informationssysteme sowie Framework- und EAI-Projekte.

H. Störle, J. H. Hausmann: **Towards a Formal Semantics of UML 2.0 Activities**

The new version 2.0 of the Unified Modeling Language (UML) was targeted at improving expressiveness and semantic precision. These developments are particularly evident in activity diagrams which have not only acquired many new features, but a completely new metamodel and semantic foundation. The UML contains some hints that Petri-nets are the inspirational source for the new semantics. In this paper we will investigate how strong the alignment of UML s activity diagrams to Petri-nets really is.We start by providing a mapping of the basic elements of activity diagrams to Petrinets and discuss the problems arising when trying to extend this approach to some of the advanced features of activity diagrams, namely exceptions, traverse-to-completion, and streaming. This examination raises several syntactic and semantic questions concerning activities. We conclude that for basic activities, the analogy works pretty well, but for higher-level constructs, no such intuitive connection exists.

Ch. Thiel: Entwurf und Implementierung eines modernen industriellen Messsystems

MEVInet ist ein flexibles Echtzeit-Messsystem, das ab ca. 1998 für die Firma IMS Messsysteme GmbH in Heiligenhaus entwickelt wurde, einem der führenden Hersteller von Röntgen-, Isotopen- und optischen Messsystemen in der Stahl- und Metallindustrie. Es wird weltweit mehr als 100-mal eingesetzt. In MEVInet liefern dezentrale Messstellen schnell repetierende Messwerte an mit Windows-CE betriebene Industrie-PCs. Die Messwertverarbeitung wird für jede Anlage individuell mit einem IEC61131-Programmierwerkzeug (logiCAD) hergestellt, die Visualisierung wird mittels Visual-Basic Komponenten erstellt. Um ein vorhandenes Messsystem zu verbessern wurden verschiedene Anforderungen neu definiert, wie z. B. kürzeste Projektierungszeiten, Aufwandsminimierung für Wartung und Pflege, verkürzte Einarbeitungszeit, unbegrenzte Skalierbarkeit. Die sehr aufwändige Anforderungsanpassung führte zu einem sehr gut zugeschnittenen System. Bis zum Abschluss der Entwicklung unterlag die Programmierung starken Anforderungsänderungen, da die Fertigstellung der Pflicht-Funktionen erst langsam das Bewusstsein für die geplante Funktionalität entstehen ließ. Entwickelt wurde u.a. mit Visual Age C++, MS (embedded)VC++, VB, MS Platform Builder (Windows CE), Visio, Together C++, VMWare. Der Vortrag stellt das moderne Messsystem mit seinen komplexen Herausforderungen an das Software-Engineering vor.

W. F. Tichy: Die Bedeutung der Empirie für die Softwaretechnik

Empirische Untersuchungen sind aus der Softwareforschung nicht mehr wegzudenken. Explorative empirische Studien einschließlich kontrollierter Experimente vergleichen verschiedene Techniken, um festzustellen, ob bei Verwendung der Techniken Unterschiede in der Qualität von Software, der Bearbeitungszeit oder anderen Eigenschaften tatsächlich zu beobachten sind. Diese Studien bestätigen oder verwerfen vermutete oder postulierte Eigenschaften von Werkzeugen und Methoden. Korrelationsstudien untersuchen Zusammenhänge zwischen Variablen bei der Softwarebearbeitung. Vorhersage- und Optimierungsmodelle schließlich versuchen die entdeckten Zusammenhänge für die Praxis der Softwareentwicklung nutzbar zu machen. Letztlich sucht der Wissenschaftler nach Erklärungen für die beobachteten Zusammenhänge, also nach Theorien über die Software-Erstellung, aus denen praktisch nutzbare Modelle abgeleitet werden können. Anhand von Ergebnissen aus den Bereichen Kostenschätzung, Inspektionen, Software-Entwurf, agile Methoden, u.a. illustriert dieser Vortrag den von der Empirie getriebenen Wissenschaftsprozess in der Softwaretechnik. Gezeigt wird, wo bereits beachtliche Fortschritte erzielt werden konnten und wo weiterer Bedarf an empirischen Untersuchungen besteht.

M. Trapp, B. Schürmann, P. Wörz: Modellierung adaptiver eingebetteter Systeme

Ein elementares Problem bei der Entwicklung adaptiver Systeme besteht darin, dass das Adaptionsverhalten eines Systems durch sehr komplexe Zusammenhänge bestimmt ist. Eine zur Beherrschung dieser Zusammenhänge notwendige formale Modellierung des Adaptionsverhaltens ist allerdings nicht verfügbar. Deshalb wurden in einer Kooperation zwischen der TU Kaiserslautern und der Robert Bosch GmbH Techniken zur Modellierung adaptiver eingebetteter Systeme entwickelt. Als Fallbeispiel diente dazu das elektronische Stabilitätsprogramm ESP. Die entwickelten Modellierungstechniken haben sich in Fallstudien bewiesen und werden in Zukunft in der ESP-Entwicklung, an der mehrere Hundert Softwareentwickler beteiligt sind, eingesetzt werden. Die Entwicklung adaptiver eingebetteter Systeme gewinnt zurzeit zunehmend an Bedeutung. In diesem Beitrag möchten wir daher die bereits vorhandenen grundlegenden Erkenntnisse unserer Untersuchungen darlegen. Dazu werden wir insbesondere allgemeine Modellierungsprinzipien vorstellen, die sich im Rahmen des Projektes als sinnvoll erwiesen haben.

M. Verlage, J.-F. Girard, D. Ganesan, Th. Kiesgen: Überwachung der Qualität der Architektur einer Software-Produktlinie am Beispiel eines web-basierten Wertpapierinformationssystems

Die kontinuierliche Modifikation und Erweiterung von Software-Produktlinien führt auch zu beabsichtigten und unbeabsichtigten Änderungen an der (Referenz-)Architektur. Wichtige Architekturmerkmale müssen kontrolliert werden, damit die Software nicht degeneriert. Es wird über Erfahrungen bei der Installation und Durchführung eines Prozesses der Architekturüberwachung in einem Wertpapierinformationssystem anhand von Metriken über einen Zeitraum von mehr als drei Jahren berichtet. Klassen mit hohen Messwerten bei einzelnen Metriken und starken Veränderungen dieser Werte werden einer Analyse durch Entwickler, Designer und Manager unterzogen. So werden Probleme in der Software frühzeitig identifiziert.

M. Verlage: **Aufbau und Wartung einer Software-Produktlinie in einem kleinen Unternehmen**

Baut ein Unternehmen Kompetenz in einem technischen oder fachlichen Bereich auf, so können weitere Potentiale erschlossen und Risiken gestreut werden, indem mehrere Produkte für unterschiedliche Märkte entwickelt werden. Insbesondere für kleine und mittelständische Unternehmen kann dies ein entscheidender Faktor der Wettbewerbsfähigkeit werden. Innovative Ansätze zur Gestaltung flexibler und konfigurierbarer Software helfen auch bei Projekten mit wenigen Mannjahren zum Ausbau einer kompletten Produktline, die eine Basis für unterschiedliche Märkte, Nischenmärkte oder Individuallösungen darstellt. Die Einführung des so genannten "Product Line Engineering" hatte erhebliche Auswirkungen auf die Effizienz und Entwicklungsgeschwindigkeit bei der MARKET MAKER Software AG. Anhand der Software i*ProductLine wird über Erfahrungen bei dem Aufbau und der Erweiterung einer Produktlinie für Internet-basierte Börseninformationssysteme berichtet. Grundlegende Entscheidungen waren dabei die vorausschauende Definition potentieller Produkte, eine Konzentration auf die Architektur der gesamten Produktlinie und die Gestaltung eines Prozesses, bei dem früh zwischen Individualentwicklung und Erweiterung der Plattform unterschieden wird. Überraschende Einsichten konnten bei den Auswirkungen auf die Durchführung von Projekten für Kunden gewonnen werden. Es wird nicht eine einzelne Technik oder Methode beleuchtet, sondern eine Zusammenfassung über die Maßnahmen zur Änderung der Software-Entwicklung in einem kleinen Unternehmen gegeben.

L. Wieske: **Service-Orientierte Architektur in Applikations- und Integrations-Projekten**

Service Oriented Architecture (SOA) polarisiert die Gemüter. Während einige ein "Deja Vuformulieren, wittern andere The Next Big Thing". Im einleitenden Überblick dieses Vortrags wird SOA in das Spannungsfeld Business-/IT-Alignement, Requirements und Application/Enterprise Architecture eingeordnet. Im Hauptteil des Vortrags werden ausgehend von eigenen Erfahrungen und Untersuchungen 13 Best Practices für Projekte mit Service-Orientierten Architekturen entwickelt. Dabei wird zwischen Anwendungs- und Integrations-Projekten unterschieden, wobei für die Integrations-Projekte wieder nach Anwendungs-, Legacy- und Unternehmens-Integration gegliedert wird. In diesen Projekttypen finden sich jeweils spezifische Requirements-Profile, die sich in der jeweiligen Service-Orientierten Architektur in entsprechenden Herausforderungen und dem Einsatz charakterischer Architektur-Patterns niederschlagen. Im übergreifenden Zusammenhang werden Migrations-Strategien entwickelt und empfohlen, sowohl für das Portfolio-Management von SOA-Projekten als auch für die Planungs- und Management-Aufgaben im einzelnen SOA-Projekt. Insgesamt führt der vorgestellte Katalog von Best Practices entscheidende wirtschaftliche und strategische Erfolgsfaktoren hinsichtlich des eingangs dargestellten Spannungsfeldes (Business-/IT-Alignement, Requirements, Architecture) zusammen und ordnet sie hinsichtlich des Portfolio- und Projektmanagements für Service-Orientierte Architekturen ein.