Teilprojekt B1

Generische Modellierung von Prozessen und Experimenten

Leiter

Prof. Dr. Dieter Rombach
geb. 6.6.1953
Universität Kaiserslautern
Postfach 3049
67653 Kaiserslautern
Tel.: 0631/205-2895
Fax: 0631/205-3331
rombach@informatik.uni-kl.de

Mitarbeiter

Ergänzungsausstattung:

Dipl.-Inform. Jürgen Münch

Grundausstattung:

Dipl.-Inform. Marcus Ciolkowski (seit 15.7.99)
Dipl.-Inform. Christiane Differding (bis 30.6.99)
Dipl.-Inform. Karin Drabe (bis 15.08.98)
Prof. Dr. Victor Basili (externer Berater)

1 Kurzfassung der Aufgabenstellung

Langfristiges Ziel dieses Teilprojekts ist die Entwicklung einer generisch ausgelegten Experimentiermethodik für die Gewinnung detaillierter Erfahrungswerte bezüglich der Eignung von Entwicklungstechniken und -prozessen in Bezug auf die relevanten Projektziele in der Domäne Gebäudeautomation unter Berücksichtigung der gegebenen Randbedingungen. Dies soll durch leichte Instantiierung generischer Experimentpläne bzw. generischer Prozessmodelle unterstützt werden. Die Aufgabenstellung umfasst die wissenschaftliche Auseinandersetzung mit folgenden Themenbereichen:

a) Entwicklung einer werkzeuggestützten Methodik zur Generierung von Messplänen, die die Spezifikation und Analyse experimenteller Messziele unterstützen;
b) Untersuchung, Realisierung und Erprobung von Generierungstechniken zur effektiven und effizienten Bereitstellung maßgeschneiderter Prozessmodelle;
c) Entwicklung einer Methodik zur Bereitstellung maßgeschneiderter, umfassender Projektpläne, die verschiedene Sichten (z. B.: Messsicht, Prozesssicht) konsistent integrieren;
d) Definition eines Schemas zur Ablage experimentell gewonnener Erfahrungen in einer Erfahrungsdatenbank.

In der ersten Förderungsperiode wurden Methoden, Techniken und Prototypwerkzeuge zur Beschreibung von Messplänen und Prozessmodellen sowie Schemata für die Ablage experimentspezifischer Ergebnisse bereitgestellt.

In der zweiten Förderungsperiode sollten diese Methoden, Techniken und Prototypwerkzeuge erprobt und optimiert werden. Die Optimierung sollte einerseits hinsichtlich der Erhöhung der Wiederverwendbarkeit von Messplänen und Prozessmodellen durch deren generische Auslegung für das Anwendungsgebiet „Gebäudeautomation" erfolgen. Andererseits sollte die Wiederverwendbarkeit von Ergebnissen vergangener Experimente durch deren Ablage im experimentübergreifenden Datenbereich der Erfahrungsdatenbank des SFB erhöht werden.

2 Entwicklung des Teilprojekts in der Förderperiode

Auf die im Berichtszeitraum erzielten Ergebnisse wird in Abschnitt 3 eingegangen.

Im Berichtszeitraum wurden zwei Dissertationen verfasst:

In Rahmen der von C. Differding verfassten Dissertation *Supporting Development, Modification, and Reuse of Measurement Plans* [17] wurde eine Methodik zur Erstellung und Wartung von Messplänen unter Wiederverwendung existierender Projektinformationen (insbesondere Prozess- und Qualitätsmodelle) entwickelt. Diese Methodik ist werkzeuggestützt. Des Weiteren wurde ein Algorithmus zur Verfeinerung von Messplänen aus Messzielen entwickelt. Dieser Algorithmus ermöglicht eine

Teilautomatisierung des Verfeinerungsprozesses von Messplänen und bietet somit wesentliche Unterstützung bei der Generierung von Messplänen basierend auf Projektinformation.

Im Rahmen der von J. Münch verfassten Dissertation *Zielführende, wiederverwendungsorientierte Erstellung von Software-Projektplänen* [18] wurde eine werkzeuggestützte Planungsmethodik zur Erstellung von Software-Projektplänen entwickelt. Der in dieser Arbeit entwickelte Ansatz ist umfassend in dem Sinne, dass er einen Planungsprozess ausgehend von einer Definition der Ziele und der Charakterisierung eines Software-Entwicklungsprojekts bis zur Erstellung eines maßgeschneiderten Projektplans beschreibt. Die entwickelte Methodik ist auf die Vorgehensplanung beschränkt, d. h. die Beschreibung der technischen Einzelprozesse und deren Zusammenspiel. Techniken, die im Rahmen dieses Ansatzes neu entwickelt wurden, betreffen a) die Komposition wiederverwendbarer Artefakte, die Prozesswissen beinhalten, b) die Verifikation von Plänen bzw. Teilplänen gegen formal definierte Ziele sowie c) die projektspezifische Anpassung von Prozessmodellen, die in einer Prozessmodellierungssprache formuliert sind. Der Ansatz basiert auf einem im Rahmen dieser Arbeit entwickelten konzeptuellen Modell für Software-Entwicklungsprozesse.

Die Ergebnisse des Teilprojekts haben von der Kooperation mit und von der begleitenden Begutachtung durch externe Experten aus dem Fraunhofer-Institut für Experimentelles Software Engineering IESE sowie aus dem Ausland (insbesondere Prof. Basili von der University of Maryland und Prof. Iida, Nara Institute of Science and Technology, Japan) profitiert. Prof. Basili hat sich mehrere Male zu Projektbegutachtungen in Kaiserslautern aufgehalten. Prof. Iida erarbeitete im Rahmen eines halbjährigen Forschungsaufenthalts (1. Juni 1998 bis 30. November 1998) einen Ansatz zur Kapselung von Wissen über Software-Entwicklungsaktivitäten mittels Prozessmustern. Hierbei identifizierte er basierend auf im SFB angewandten Vorgehensmodellen Prozessmuster und entwickelte eine Vorgehensweise zur Kombination dieser Muster [57]. Über die Kooperation mit dem Fraunhofer-Institut IESE konnten die entwickelten Techniken und Methoden unter industriellen Randbedingungen (z. B.: bei der Allianz Lebensversicherungs AG, bei der Bosch Telecom GmbH [jetzt Tenovis]) erprobt werden.

Im Berichtszeitraum fand in Kaiserslautern die internationale Konferenz SEKE'99 (11th International Conference on Software Engineering and Knowledge Engineering) statt. Der Teilprojektleiter war General Chair der Konferenz.

Im Berichtszeitraum wurde der 6. Workshop der GI-Fachgruppe 5.1.1 zum Thema "Vorgehensmodelle, Prozessverbesserung und Qualitätsmanagement" durchgeführt (in Kooperation mit dem Fraunhofer-Institut IESE).

Die Ergebnisse des Teilprojekts wurden bereits weltweit veröffentlicht (z. B.: International Journal of Empirical Software Engineering, International Conference on Software Engineering and Knowledge Engineering) und sind auf positive Resonanz in Fachkreisen gestoßen.

3 Durchgeführte Arbeiten und Ergebnisse

3.1 Arbeitsschwerpunkt A: Generische Modellierung von GQM-Plänen

Identifikation relevanter Parameter

Ziel dieses Arbeitspakets war die Identifikation von Parametern, die Struktur und Inhalt von GQM-Plänen wesentlich beeinflussen, sowie die Untersuchung der Wirkung der identifizierten Parameter auf GQM-Pläne. In der zweiten Förderungsperiode wurde der Einfluss ausgewählter GQM-Zielparameter auf GQM-Pläne untersucht. Ziele von GQM-Plänen variieren in Abhängigkeit von Objekt und Zweck der Messungen, vom gewählten Qualitätsaspekt, von der Sicht sowie vom Kontext. Speziell untersucht wurden Variationen der Parameter Objekt und Qualitätsmerkmal. Der Parameter Objekt beschreibt Informationen zur Beantwortung der Frage: Was wird gemessen? Sei beispielsweise ein Vorgehensmodell das Objekt in einem Messprogramm zur Charakterisierung des Aufwandes. Die Hinzufügung eines Prozessschrittes führt dann im GQM-Plan zu einer Hinzufügung einer entsprechenden GQM-Frage bzw. eines entsprechenden Maßes. Der Parameter Qualitätsmerkmal beschreibt Informationen zur Beantwortung der Frage: Welche Qualitätseigenschaft wird gemessen? Sei beispielsweise der Defect Slippage (d. h., die Verteilung der Fehler hinsichtlich ihres Entdeckungszeitpunkts) das zu untersuchende Qualitätsmerkmal und man verändere die Fehlerklassifikation. Die Wirkung einer solchen Parameteränderung auf den GQM-Plan besteht in der Veränderung des unmittelbar zugehörigen Maßes sowie der Anpassung hiervon abhängiger Maße. Des Weiteren müssen unter Umständen neue mit GQM-Fragen assoziierte Berechnungsvorschriften zur Interpretation der Messwerte in Bezug auf den Defect Slippage in den GQM-Plan eingefügt werden. Der Parameter Zweck beschreibt Informationen zur Beantwortung der Frage: Wozu wird gemessen? Im Rahmen der in diesem Arbeitspaket durchgeführten detaillierten Untersuchung der Parameter wurde der Parameter Zweck zunächst auf den Zweck des Charakterisierens fixiert. Dies ist unter anderem damit begründet, dass bei den im SFB durchgeführten Baselining-Entwicklungen eine Charakterisierung wichtiger Qualitätsmerkmale im Vordergrund steht. Die beiden variierten Parameter Objekt und Qualitätsmerkmal werden repräsentiert durch Prozessmodelle beziehungsweise Qualitätsmodelle. Der Einfluss dieser Parameter auf GQM-Pläne wurde bei der Entwicklung der Methodik zur Erstellung und Wartung von Messplänen [17] maßgeblich berücksichtigt.

Definition einer Notation zur generischen Formulierung von GQM-Plänen

Ziel dieses Arbeitspakets war die Entwicklung einer Notation, die generische GQM-Pläne beschreibt. In der zweiten Förderungsperiode wurde ein formales Modell für die Darstellung von GQM-Plänen entwickelt [17]. Dieses Modell wurde mit einem (zusätzlich entwickelten) Modell für Prozess- und Qualitätsmodelle integriert.

Entwicklung einer Methodik zur generischen Erstellung und Wartung von GQM-Plänen

Ziel dieses Arbeitspakets war die Entwicklung einer Methodik zur Erstellung und Wartung von generischen GQM-Plänen. Insbesondere sollten Vorgehensweisen und Richtlinien bereitgestellt werden, die die schrittweise Erstellung von GQM-Plänen unter Ausnutzung wiederverwendbaren Erfahrungswissens sowie die Durchführung von Änderungen an GQM-Plänen unterstützen. In der zweiten Förderungsperiode wurde ein Erstellungs- und Wartungsprozess für Messpläne (die GQM-Pläne enthalten) entwickelt [17]. Dieser Prozess enthält unter anderem einen auf dem integrierten Modell basierenden Algorithmus zur Verfeinerung von GQM-Plänen und bildet damit die Grundlage zur Generierung von GQM-Plänen aus Messzielen, Prozess- und Qualitätsmodellen.

Realisierung von Werkzeugen zur Erstellung und Wartung von GQM-Plänen

Ziel dieses Arbeitspakets war die Entwicklung einer Werkzeugungerstützung zur Bearbeitung von GQM-Plänen. In der zweiten Förderungsperiode wurde ein Werkzeug erstellt [6] und weiterentwickelt [2][4], mit dem GQM-Pläne erstellt und exportiert werden können. Zur Unterstützung der Wartung von GQM-Plänen wurde das Werkzeug erweitert um eine Schnittstelle zum Einlesen von geänderten Prozessmodellen [4]. Basierend auf dieser Information wird die Wartung von GQM-Plänen bei Änderungen des Prozessmodells unterstützt [2].

3.2 Arbeitsschwerpunkt B: Generische Modellierung von Prozessmodellen

Identifikation relevanter Parameter

Ziel dieses Arbeitspakets war die Identifikation geeigneter Parameter für Software-Entwicklungsprozesse des Anwendungsgebiets “Gebäudeautomation”. In der zweiten Förderungsperiode wurden existierende Methoden, die zur Entwicklung eingebetteter Software geeignet oder bestimmt sind, sowie ausgewählte Varianten dieser Methoden in der formalen Prozessmodellierungssprache MVP-L formuliert. Eine vergleichende Analyse der Varianten führte zur Identifikation wesentlicher Einflussfaktoren und einer Entwicklung konkreter Charakterisierungsschemata. Im Einzelnen wurden folgende Methoden modelliert (die Beschreibung der Modellierung findet sich in den angegebenen Referenzen): 1) SLD-oriented Object Modeling Technique (SOMT) [19], 2) Naval Research Laboratory / Software Cost Reduction (NRL / SCR) [12], 3) Object Modeling Technique (OMT) [12], 4) Der Pss-05-Prozess der European Space Agency [5]. Wesentliche Einflussfaktoren wie Echtzeitfähigkeit und Sicherheitskritikalität sowie ermittelte Charakterisierungsschemata (starr, facetten- und modellorientiert) sind in [5] und [12] beschrieben.

Definition einer Notation zur Formulierung generischer Prozessmodelle

Ziel dieses Arbeitspakets war die Entwicklung einer Notation zur Formulierung generischer Prozessmodelle. In der zweiten Förderungsperiode wurde ein konzeptuelles Modell entwickelt, dass die wesentlichen Konzepte von Software-Entwicklungsprozessen umfasst und deren Semantik mittels Zusicherungen definiert [8]. Das Modell orientiert sich am Vier-Schichten-Modell zur Metamodellierung von Information (Notationsschicht, Konzeptschicht, Modellschicht und Projektschicht) und ist im Wesentlichen auf der Konzeptschicht angesiedelt. Das konzeptuelle Modell ist spezifiziert unter Verwendung folgender Beschreibungstechniken: a) Unified Modeling Language (UML, Version 1.3). Hierbei werden Klassendiagramme zur Beschreibung des konzeptuellen Modells verwendet. Des Weiteren wird das UML-Metamodell (meta-metamodel, metamodel, model, user objects) zur Einordnung der Modellelemente benutzt; b) Object Constraint Language (OCL, Version 1.3). OCL wird hierbei benutzt zur Beschreibung von Constraints von UML-Modellen. OCL wurde zur Beschreibung der Constraints des konzeptuellen Modells um benutzerdefinierte Funktionen erweitert. Das konzeptuelle Modell ist konform mit den Konzepten der Prozessmodellierungssprache MVP-L, umfasst jedoch Konstrukte (z. B.: charakterisierende Attribute, Gültigkeitsbereich, Anpassungsregeln) zur Darstellung von Generik in Prozessmodellen.

Entwicklung einer Methodik zur Erstellung maßgeschneiderter Prozessvarianten in MVP-L

Ziel dieses Arbeitspakets war die Entwicklung einer Methodik zur effektiven und effizienten Bereitstellung maßgeschneiderter Prozessmodelle in MVP-L. In der zweiten Förderungsperiode wurde basierend auf dem konzeptuellen Modell ein werkzeuggestützter Ansatz zur Erstellung von maßgeschneiderten Prozessmodellen entwickelt. Dieser Ansatz ist integriert in eine zielführende, wiederverwendungsorientierte Planungsmethodik zur Erstellung von Projektplänen [18]. Techniken, die im Rahmen dieses Ansatzes neu entwickelt wurden, betreffen a) die Komposition wiederverwendbarer Artefakte, die Prozesswissen beinhalten (in Form von Prozessmustern [57]), b) die Verifikation von Plänen bzw. Teilplänen gegen formal definierte Ziele (in Analogie zur Verifikation von Programmen nach [59]) sowie c) die projektspezifische Anpassung von Prozessmodellen, die in MVP-L formuliert sind [46]. Die Methodik ist fünfschrittig (Selektion, Komposition, Evaluierung, Anpassung, Integration) und kann iterativ durchgeführt werden.

Implementierung von Werkzeugen zum Maßschneidern von Prozessmodellen

Ziel dieses Arbeitspakets war die Realisierung einer prototypischen Werkzeugunterstützung zur (Teil-)Automatisierung von notwendigen Generierungsschritten bei der Erstellung maßgeschneiderter Prozessmodelle. In der zweiten Förderungsperiode wurde ein unterstützendes Werkzeug entwickelt, dass die Anpassung von Prozessmodellen an Projektziele und -charakteristika weitgehend automatisiert. Dieses Werkeug, das sogenannte Process Tailoring Tool (ProTail), basiert auf einem transformationsbasierten Ansatz. Die Anpassung der Prozessmodelle erfolgt hierbei durch eine Folge von Transformationsschritten, die mit der sequentiellen Anwendung von Anpassungsregeln und Anpassungsprimitiva (hier in Form von Transformationsregeln) verglichen werden können. Der transformationsbasierte Ansatz bietet die Möglichkeit, Anpassungsregeln explizit zu erfassen, sodass sie einfach an neue Erkenntnisse angepasst werden können. Der Anpassungsprozess wird hierbei unterteilt in die Teilschritte

Konfigurierung, Charakterisierung, Transformation und Dokumentation. Abb. 1 gibt einen Überblick über die Eingangs- und Ausgangsdokumente des Werkzeugs. Das Werkzeug enthält zusätzlich eine Komponente zur Verwaltung von Qualitätsmodellen, sodass bekannte Relationen zwischen relevanten Qualitätsmerkmalen und zugehörigen Einflussfaktoren in den Anpassungsprozess einbezogen werden können [13].

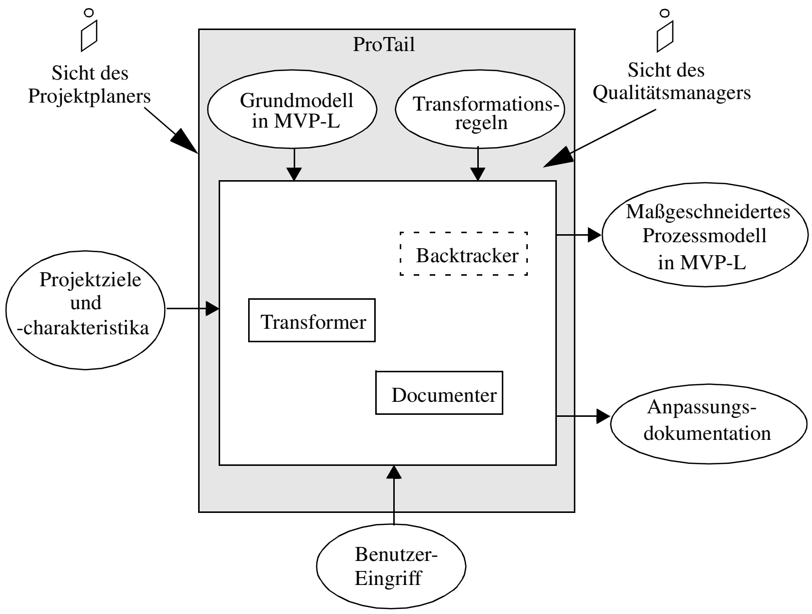

Abbildung 1: Werkzeuggestützte Anpassung von Projektmodellen

3.3 Arbeitsschwerpunkt C: Generierung instrumentierter Projektpläne

Ziel dieses Arbeitsschwerpunkts war die Entwicklung einer systematischen Methodik zur Erstellung umfassender Projektpläne, die die Mess- und Projektsicht integrieren. Die in der zweiten Förderungsperiode entwickelte Methodik zur generischen Modellierung von GQM-Plänen [17] und das zugehörige Werkzeug GQMPlanner [4] ermöglichen die Entwicklung von GQM-Plänen, die konsistent zu einem existierenden Prozessmodell sind. Dies wird durch die Erstellungs- und Wartungsmethode gesichert. Somit können Prozessmodelle und GQM-Pläne separat und koordiniert erstellt werden, müssen jedoch anschließend integriert werden. Eine solche Vorgehensweise hat zunächst den Vorteil, dass Prozess- und Messaspekte separat untersucht und Variationen innerhalb der jeweiligen Sicht identifiziert werden können. Es ergibt sich jedoch das Problem der Fragmentierung von Sichten: Zusammengehörige Aspekte wie z. B. die Aggregation von Entwicklungsprozessen und die zugehörige Verdichtung von Messgrößen werden getrennt behandelt und teilweise redundant beschrieben. Hierbei können Inkonsistenzen entstehen, die behoben werden müssen. Die Fragmentierung von Sichten bereitet bei der Projektplanung insbesondere dann Schwierigkeiten, wenn Teile des Plans generiert werden. Wechselseitige Abhängigkeiten zwischen verschiedenen Sichten (insbesondere zwischen der Mess- und der Prozesssicht) werden in diesem Falle nicht systematisch berücksichtigt. Um dieses

Problem zu überwinden, soll die Methodik in der dritten Förderungsperiode dahingehend weiterentwickelt werden, dass Erstellungsprozesse für Mess- und Prozesspläne miteinander verzahnt werden. Die Verzahnung soll schrittweise und parallel erfolgen, sodass stufenweise konsistente Resultate entstehen. Dies ermöglicht einerseits eine zielführende Auswahl geeigneter Entwicklungsansätze auf unterschiedlichen Abstraktionsebenen des Software-Entwicklungsprozesses und andererseits eine konsistente Instrumentierung der Prozesspläne.

3.4 Arbeitsschwerpunkt D: Organisation der Erfahrungsdatenbank

Ziel dieses Arbeitsschwerpunkts war die Entwicklung eines experimentübergreifenden Erfahrungsdatenbankschemas basierend auf einer vergleichenden Analyse der im SE-Kern des SFB existierenden Erfahrungen. In der zweiten Förderungsperiode wurde in enger Kooperation mit Teilprojekt A1 die existierenden Erfahrungen im SE-Kern analysiert und ein grobgranulares Datenbankschema für den experimentübergreifenden Bereich der Erfahrungsdatenbank ermittelt [44]. Dieses Schema wurde von Teilprojekt A1 verfeinert und in Zusammenarbeit mit Teilprojekt A3 realisiert.

3.5 Arbeistschwerpunkt E: Evaluierung entwickelter Techniken und Werkzeuge

Ziel dieses Arbeitsschwerpunkts war eine Evaluierung der entwickelten Methoden, Techniken und Werkzeuge. Dazu sollte insbesondere das SFB-Praktikum herangezogen werden. In der zweiten Förderungsperiode wurde die Methodik und das Werkzeug zur Erstellung von GQM-Plänen in einer initialen Anwendung [1] und weiter in einer Serie von Fallstudien evaluiert [9]. Die Effektivität und Effizienz des Prozesses zur Wartung von GQM-Plänen wurde in einem kontrollierten Experiment mit Studenten untersucht. Dabei wurde das Prozessmodell verändert, und der GQM-Plan wurde von den Studenten angepasst. Die Wartung wurde in diesem Experiment auf den initialen Schritt der Auswirkungsanalyse reduziert. Es wurde gezeigt, dass die Wartung durch ein integriertes Modell von GQM-Plan und Prozessmodell effektiver (höhere Korrektheit der identifizierten Auswirkungen) und effizienter (weniger Aufwand zum Identifizieren der Auswirkungen) ist als die Wartung ohne ein solch integriertes Modell. Außerdem ist den Studenten die Wartung mit dem integrierten Modell als Basis subjektiv leichter gefallen. Die Methodik und das Werkzeug zur Erstellung maßgeschneiderter Prozessmodelle wurden in zwei Fallstudien [18] hinsichtlich der erwarteten Reduktion des Erstellungsaufwandes und der erwarteten Erhöhung der Konsistenz der erstellten Prozessmodelle untersucht. Dabei konnte die Zeit, die zur werkzeuggestützten Generierung eines Prozessmodells benötigt wurde, gegenüber einer manuellen Erstellung signifikant reduziert werden (Faktor >65). Komplett eliminiert werden konnten Zeiten zur Identifizierung von Folgeänderungen und zur Erstellung der Dokumentation, da dies automatisch durch das Werkzeug geschieht. Sowohl bei der manuellen Erstellung als auch bei der automatischen Generierung maßgeschneiderter Prozessmodelle konnte die Konsistenz durch den Einsatz eines automatischen Konsistenzcheckers gewährleistet werden. Hierbei ist jedoch zu beachten, dass bei der manuellen Erstellung das Prozessmodell solange auf Inkonsistenzen geprüft wurde, bis das Ergebnis konsistent wurde. Bei der werkzeuggestützten Generierung war nur eine Konsistenzprüfung am Ende des Generierungsvorgangs nötig. Weiterhin ist zu bemerken, dass diese Verwendung eines Konsistenzcheckers zur Erleichterung der manuellen Erstellung nur deshalb möglich wurde, da auch die manuelle Erstellung auf der Ebene von MVP-L, d. h. einer formalen Prozessmodellierungssprache, stattfand. Bei der Verwendung einer informellen Beschreibung in Form von Textbausteinen wäre dies nicht möglich gewesen. Es ist zu vermuten, dass ohne die Verwendung eines Konsistenzcheckers (also einer tatsächlich 100%-igen manuellen Anpassung) Inkonsistenzen wesentlich häufiger auftreten würden, da Folgeänderungen übersehen würden. Abschließend sei bemerkt, dass die in Teilprojekt B1 entwikkelten Techniken, Methoden und Werkzeuge zur Unterstützung der Querschnittsprojekte Q1 und Q2 eingesetzt wurden und hierbei gemachte Erfahrungen bei der Weiterentwicklung berücksichtigt wurden. Querschnittsprojekt Q2 wurde im Rahmen des SFB-Praktikums 1999 durchgeführt.

4 Zusammenarbeit mit anderen Teilprojekten

Die Durchführung von Anwendungsentwicklungen und Experimenten erfolgte unter Anleitung des SE-Labors (Teilprojekt A1). Hierfür wurden die Ergebnisse des Teilprojekts B1 in Form von Techniken und Prototypwerkzeugen in das SE-Labor integriert und damit dem gesamten SFB zur Verfügung gestellt. Zusätzlich wurde in Zusammenarbeit mit Teilprojekt A1 das grobgranulare Schema des experimentübergreifenden Teils der Erfahrungsdatenbank definiert und von A1 und A3 realisiert.

In Zusammenarbeit mit den Teilprojekten A2 und B2 wurde der MILOS-Ansatz, der MVP-L mit plan-basierten KI-Techniken integriert, um Konzepte der verteilten Planung erweitert [50][52][55].

In Zusammenarbeit mit Teilprojekt B4 wurde die SOMT-Entwicklungsmethode formal modelliert [19].

Der Mitarbeiter der Ergänzungsausstattung arbeitete im Querschnittsprojekt Q1 bei der Mess-und Projektplanung mit [21].

Das Querschnittsprojekt Q2 (SFB-Praktikum im Sommersemester 1999) wurde von Mitarbeitern des Teilprojekts mitbetreut [23][24].

5 Eigene Veröffentlichungen im Berichtszeitraum

Projektarbeiten

[1] Eleni Chaitidou: *Erstellung von Projekt- und GQM-Plänen für ein Software-Entwicklungsprojekt.* Projektarbeit, Fachbereich Informatik, Universität Kaiserslautern, 67653 Kaiserslautern, 1998.

[2] Matthias Gutheil: *Wartbare Schema-Repräsentation für GQM-Pläne.* Projektarbeit, Fachbereich Informatik, Universität Kaiserslautern, 67653 Kaiserslautern, 1998.

[3] Sascha Werner: *Entwurf und Realisierung einer ergonomischen Benutzungsschnittstelle.* Projektarbeit, Fachbereich Informatik, Universität Kaiserslautern, 67653 Kaiserslautern, 1998.

[4] Eric Ras: *Export-/ Importschnittstelle für das Werkzeug GQMPlanner.* Projektarbeit, Fachbereich Informatik, Universität Kaiserslautern, 67653 Kaiserslautern, 1999.

[5] Björn Schmidt: *Modellierung eines adaptierbaren Software-Entwicklungsprozesses.* Projektarbeit, Fachbereich Informatik, Universität Kaiserslautern, 67653 Kaiserslautern, 1999.

[6] Christian Voigtländer: *GQMaspect II: Entwicklung eines Werkzeugs zur Erstellung von GQM-Plänen.* Projektarbeit, Fachbereich Informatik (mitbetreut durch Fraunhofer IESE), Universität Kaiserslautern, 67653 Kaiserslautern, 1999.

[7] Wolfgang Wagenbichler: *Erstellung einer SW-Systemdokumentation mit StP.* Projektarbeit, Fachbereich Informatik, Universität Kaiserslautern, 67653 Kaiserslautern, 1999.

[8] Jens Heidrich: *Conceptual Modeling of Software Development Processes.* Projektarbeit, Fachbereich Informatik, Universität Kaiserslautern, 2000.

[9] Jürgen Kortgen: *Validierung der systematischen Entwicklung von GQM-Plänen.* Projektarbeit, Fachbereich Informatik, Universität Kaiserslautern, 2000.

Diplomarbeiten

[10] Kerstin Brockhage: *Suche nach einem Verfolgbarkeitsansatz zur Verbesserung von Änderbarkeit (Search for a traceability approach to improve changeability).* Diplomarbeit, Fachbereich Informatik, Universität Kaiserslautern, 67653 Kaiserslautern, 1998.

[11] Ralph Eiselein: *Definition of measurement points and measurement procedures by using process models and GQM plans.* Diplomarbeit, Fachbereich Informatik, Universität Kaiserslautern, 67653 Kaiserslautern, 1998.

[12] Erik Petersen: *Formalisierung ausgewählter Software-Entwicklungsmethoden für reaktive Systeme in MVP-L (Formalization of selected software development methods for reactive systems).* Diplomarbeit, Fachbereich Informatik, Universität Kaiserslautern, 67653 Kaiserslautern, 1998.

[13] Thomas Ruf: *Erweiterung eines Werkzeugs zur interaktiven Manipulation von Software-Projektplänen (Extension of a tool for the interactive manipulation of software project plans).* Diplomarbeit, Fachbereich Informatik, Universität Kaiserslautern, 67653 Kaiserslautern, 1998.

[14] Marcus Ciolkowski: *Evaluating the Effectiveness of Different Inspection Techniques on Informal Requirements Documents.* Diplomarbeit, Fachbereich Informatik (mitbetreut durch Fraunhofer IESE), Universität Kaiserslautern, 67653 Kaiserslautern, 1999.

[15] Marco Heide: *Objektorientierter Entwurfsprozess zur Gewährleistung von Verfolgbarkeit im Bereich technischer Systeme.* Diplomarbeit, Fachbereich Informatik, Universität Kaiserslautern, 67653 Kaiserslautern, 1999.

[16] Christian Voigtländer: *Werkzeugunterstützung zur Aufbereitung von Auswertungen im Rahmen von Messprogrammen.* Diplomarbeit, Fachbereich Informatik, Universität Kaiserslautern, 67653 Kaiserslautern, 2000.

Dissertationen

[17] Christiane Differding: *Supporting Development, Modification, and Reuse of Measurement Plans*. Dissertation (zur Einreichung im Berichtszeitraum vorgesehen), Fachbereich Informatik, Universität Kaiserslautern, 67653 Kaiserslautern, 2000.

[18] Jürgen Münch: *Zielführende, wiederverwendungsorientierte Erstellung von Software-Projektplänen.* Dissertation (zur Einreichung im Berichtszeitraum vorgesehen), Fachbereich Informatik, Universität Kaiserslautern, 67653 Kaiserslautern, 2000.

SFB-Berichte

[19] Birgit Geppert, Jürgen Münch und Erik Petersen: *Ein formales Prozessmodell für die Software-Entwicklungsmethode SOMT (A Formal Process Model for the Software Development Method SOMT).* Technischer Bericht Nr. 03/98, Sonderforschungsbereich 501, Fachbereich Informatik, Universität Kaiserslautern, 67653 Kaiserslautern, 1998.

[20] Kerstin Brockhage, Christiane Differding und Antje von Knethen: *Bessere Änderbarkeit durch Verfolgbarkeit: Auswahl eines Ansatzes.* Technischer Bericht Nr. 301/99, Fachbereich Informatik, Universität Kaiserslautern, 67653 Kaiserslautern, 1999.

[21] Raimund L. Feldmann, Jürgen Münch, Stefan Queins, Stefan Vorwieger und Gerhard Zimmermann: *Baselining a Domain-Specific Software Development Process.* Technischer Bericht Nr. 02/97, Sonderforschungsbereich 501, Fachbereich Informatik, Universität Kaiserslautern, 67653 Kaiserslautern, 1999.

[22] Jürgen Münch: *Reuse of Process Patterns.* In Tagungsband des Workshops “Software Reuse”, Technischer Bericht Nr. 5/99, Sonderforschungsbereich 501, Fachbereich Informatik,Universität Kaiserslautern, 67653 Kaiserslautern, 22.-23. Februar 1999.

[23] Antje von Knethen, Raimund Feldmann, Jürgen Münch, Stefan Vorwieger: *A Baseline for the Development of Embedded Software*. Technischer Bericht Nr. 4/00, Sonderforschungsbereich 501, Fachbereich Informatik, Universität Kaiserslautern, 67653 Kaiserslautern, 2000.

[24] Antje von Knethen, Jürgen Münch: *Entwicklung eingebetteter Software mit UML: Der Do-it-Prozess V 1.0*, Technischer Bericht Nr. 05/00, Sonderforschungsbereich 501, Fachbereich Informatik, Universitt Kaiserslautern, 67653 Kaiserslautern, 2000.

Vorträge

[25] H. D. Rombach: *Continuous Improvement of Software Development Competence: Prerequisite for Future Competitiveness*, Espoo & Oulu, Februar 1998.

[26] H. D. Rombach: *Kontinuierliche Qualitätsverbesserung*, ESA, Amsterdam, Niederlande, März 1998.

[27] H. D. Rombach: *Software Experience Center Project*, Ericsson, Irland, März 1998.

[28] H. D. Rombach: *Software-Prozessverbesserung: Strategische Notwendigkeit für alle Branchen*, SQM-Tagung, 1998, Köln, April 1998.

[29] H. D. Rombach: *Experimentation as a Vechicle for Software Technology Transfer: A Family of Software Reading Techniques*, „FESMA ’98“, Konferenz, (Keynote), Antwerpen, Mai 1998.

[30] H. D. Rombach: *Die Bedeutung von Experimenten im Software Engineering*, Kolloquium, Graduiertenkolleg „Parallele & Verteilte Systeme“, Universität Stuttgart, Juni 1998.

[31] H. D. Rombach: *Experimentieren: Prototyping bei der Prozessverbesserung*, GI-Konferenz „Softwaretechnik '98“, Paderborn, August 1998.

[32] H. D. Rombach: *Softwaretechnologie: Entscheidender Faktor für Wirtschaftswachstum*, Jahrestagung der FhG, Darmstadt, Oktober 1998.

[33] H. D. Rombach: *Empirische Methoden: Voraussetzung für ingenieurmäßige Softwareentwicklung*, Eingeladener Festvortrag, 10-Jahresfeier Fakultät für Informatik, Universität Stuttgart, November 1998.

[34] H. D. Rombach: *Testing and Inspections: A Historical Perspective*, „EuroStar98“, Konferenz, (Keynote) München, Dezember 1998.

[35] H. D. Rombach: *Software Experience Factory: Basis für beschleunigte Prozessverbesserung*, GI-Workshop, Kaiserslautern, April 1999.

[36] H. D. Rombach: *Innovationen im Software-Entwicklungsprozess*, Robert-Bosch Kolleg, Stuttgart, Juni 1999.

[37] H. D. Rombach: *Software Process Improvement: Prerequisite for Future Competitiveness*, SEC Meeting, Baden, Schweiz, September 1999.

[38] H. D. Rombach: *Continuous Improvement of Software Development Competence: Prerequisite for Future Competitiveness*, University of New South Wales, New South Wales, Australia, November 1999.

[39] H. D. Rombach: *Experimental Software Engineering*, Colloquium, Nara Institute of Science and Technology (NAIST), School of Information Science, Nara, Japan, November 1999.

[40] H. D. Rombach: *Experimental Software Engineering*, Colloquium, Osaka University, Graduate School of Engineering Science, Department of Computer Science & Mathematics, Osaka, Japan, November 1999.

[41] H. D. Rombach: *GERMANY: Activities in Software Process Improvement, Country Report for Germany* (invited presentation), Indian SEPG Conference, Bangalore, India, Februar 2000.

[42] H. D. Rombach: *Capitalising on Experience – What is missing in CMM & PSP/TSP?* -, (keynote address), Indian SEPG Conference, Bangalore, India, Februar 2000.

[43] H. D. Rombach: *Software Experience Factory „how to do it“* Guide, (tutorial), Indian SEPG Conference, Bangalore, India, Februar 2000.

Tagungsbeiträge

[44] Raimund L. Feldmann, Jürgen Münch und Stefan Vorwieger: *Towards Goal-Oriented Organizational Learning: Representing and Maintaining Knowledge in an Experience Base*. In Proceedings of the 10th International Conference on Software Engineering and Knowledge Engineering (SEKE'98), San Francisco, USA, 18.-20. Juni 1998.

[45] Antje von Knethen, Erik Kamsties, Ralf Reussner und Bin Shen: *A Comparative Case Study with Industrial Requirements Engineering Methods.* 11th International Conference on Software Engineering and its Applications, Paris, 8.-10. Dezember 1998.

[46] Jürgen Münch: *Anpassung von Vorgehensmodellen im Rahmen ingenieurmäßiger Softwarequalitätssicherung.* In Tagungsband des 6. Workshops der Fachgruppe 5.1.1 (GI): Vorgehensmodelle, Prozessverbesserung und Qualitätsmanagement, Kaiserslautern, 19.-20. April, 1999.

[47] Sigrid Goldmann, Jürgen Münch und Harald Holz: *MILOS: A Model of Interleaved Planning, Scheduling, and Enactment.* In Web-Proceedings of the 2nd Workshop on Software Engineering over the Internet, Los Angeles, CA, USA, 17. Mai 1999.

[48] Erik Kamsties, Antje von Knethen, Jan Philipps und Bernhard Schätz: *Eine vergleichende Fallstudie mit CASE-Werkzeugen für objektorientierte und funktionale Modellierungstechniken.* OMER Workshop, 28.-29. Mai 1999, P. Hofmann, A. Schürr (EDS.), OMER Workshop Proceedings Bericht Nr. 1999-01, Mai 1999.

[49] Erik Kamsties, Antje von Knethen, Jan Philipps und Bernhard Schätz: *Eine vergleichende Fallstudie mit CASE-Werkzeugen für formale und semi-formale Beschreibungstechniken.* GI/ITG-Fachgespräch formale Beschreibungstechniken für verteilte Systeme, 1999.

[50] Sigrid Goldmann, Jürgen Münch und Harald Holz: *A Meta-Model for Distributed Software Development.* In Proceedings of the 8th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WET-ICE'99), Stanford, California, USA, 16.-18. Juni 1999.

[51] Dieter Rombach: *Experimentation: Engine for Applied Research and Technology Transfer in Software Engineering.* In Proceedings of the 24th Annual Software Engineering Workshop of the SEL, Greenbelt, USA, 1./2. Dezember 1999.

[52] Sigrid Goldmann, Jürgen Münch und Harald Holz: *Distributed Process Planning Support with MILOS.* In Proceedings of the 11th International Conference on Software Engineering and Knowledge Engineering (SEKE'99), Kaiserslautern, 17.-19. Juni 1999. Erweiterte Fassung in International Journal of Software Engineering and Knowledge Engineering, Oktober 2000.

Zeitschriften / Monografien

[53] F. van Latum, R. van Solingen, M. Oivo, B. Hoisl, D. Rombach und G. Ruhe: *Adopting GQM-based Measurement in an Industrial Environment.* In IEEE Software, Seiten 78-86, Januar/Februar 1998.

[54] J. Avenhaus, R. Gotzhein, T. Härder, L. Litz, K. Madlener, J. Nehmer, M. Richter, N. Ritter, H. D. Rombach, B. Schürmann und G. Zimmermann: *Entwicklung großer Systeme mit generischen Methoden - Eine Übersicht über*

den Sonderforschungsbereich 501. Informatik, Forschung und Entwicklung, 13(4):227–234, Dezember 1998.

[55] Sigrid Goldmann, Jürgen Münch und Harald Holz. *Distributed Process Planning Support with MILOS.* In International Journal of Software Engineering and Knowledge Engineering, August 2000.

6 Fremdveröffentlichungen

[56] V.R. Basili, G. Caldiera und H.D. Rombach: *Goal Question Metric Paradigm*; In John J. Marciniak, Herausgeber, Encyclopedia of Software Engineering, Band 1, Seiten 528-532, John Wiley & Sons, 1994.

[57] Hajimu Iida: *Pattern-oriented Approach to Software Process Evolution.* In Proceedings of the International Workshop on Process of Software Evolution (IWPSE'99), Fukuoka City, Japan, 16.-17. Juli 1999.

[58] S. Queins und G. Zimmermann: *A First Iteration of a Reuse-Driven, Domain-Specific System Requirements Analysis Process*. Technischer Bericht Nr. 13/99, Sonderforschungsbereich 501, Fachbereich Informatik, Universität Kaiserslautern, 67653 Kaiserslautern, 1999.

[59] Marvin V. Zelkowitz: *A Functional Correctness Model of Program Verification*. IEEE Computer, Seiten 30-39, November 1990.

Teilprojekt B2

Flexible Planung und Ausführung von Softwareentwicklungsprozessen

Leiter

Prof. Dr.Michael M. Richter
geb. 21.6.1938
Universität Kaiserslautern
Postfach 3049
67653 Kaiserslautern
Tel.: 0631/205-2800
Fax.: 0631/205-3357
richter@informatik.uni-kl.de

Mitarbeiter

Ergänzungsausstattung:

Dipl. Inform. H. Holz
Dipl. Inform. B. Dellen (bis 31.12.98)
Dipl. Inform. M. Schaaf(ab 15.2.99)

Grundausstattung:

Dipl. Inform. F. Weberskirch
Dipl. Inform. S. Schmitt

1 Kurzfassung der Aufgabenstellung

Langfristiges Ziel des Teilprojekts B2 ist die Entwicklung von Methoden und Techniken zur flexiblen Planung und Steuerung komplexer Software-Entwicklungsprozesse. Hierbei liegt das Hauptaugenmerk darauf, Aktivitäten, die sich mit dem Management des Softwareentwicklungsprozesses beschäftigen (also Meta-Aktivitäten), möglichst fließend in die Projektdurchführung zu integrieren. Dazu wurden in der ersten Förderungsperiode Grundlagen entwickelt, welche die Modellierung und Operationalisierung von Projektplänen ermöglichen. Darauf aufbauend sollten in der zweiten Förderungsperiode die folgenden Ziele erreicht werden:

- Weiterentwicklung von Methoden und Techniken zur Verwaltung von Änderungen von Projektplänen während der Ausführung
- Entwicklung von Konzepten zur Herleitung und Verwaltung von domänenspezifischen Abhängigkeiten
- Bereitstellung von Techniken zur Verwaltung von Entscheidungsbegründungen

Hierzu wurde in Zusammenarbeit mit dem Teilprojekt A2 das Prozessunterstützungssystem MILOS[1] implementiert. Es erlaubt die Modellierung und Planung von Software-Entwicklungsprojekten und unterstützt ihre Durchführung u.a. mittels To-Do-Listen und Änderungsnotifikationen. Planverfeinerungen und Planänderungen während der Projektdurchführung werden unterstützt und gehen in die dem MILOS-System zugrundeliegende Abhängigkeitsverwaltung ein.

2 Entwicklung des Teilprojekts in der Förderperiode

Die inhaltlichen Ergebnisse des Teilprojekts werden im folgende Abschnitt erläutert. Hier werden äußere Ereignisse, an denen das Teilprojekt beteiligt war, beschrieben.

Sowohl in 1998 als auch in 1999 wurde auf der Konferenz “Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises” (WETICE) ein Workshop “Coordinating Distributed Software Development Projects” organisiert. In Zusammenarbeit mit Prof. Frank Maurer (University of Calgary, assoziiertes Mitglied des SFB und vormaliger Mitleiter des B2-Projekts) wurde auf der “International Conference on Software Engineering” (ICSE) in 1999 und 2000 je ein Workshop zum Thema “Software Engineering over the Internet” veranstaltet.
In Kooperation mit der Arbeitsgruppe von Prof. Maurer wurde auch der MILOS-Prototyp entwickelt und implementiert. Das System ist 1999 sowohl auf der ICSE als auch auf der SEKE (“International Conference on Software Engineering and Knowledge Engineering”) demonstriert [33] worden.
Die Konzepte der Abhängigkeitsverwaltung in MILOS wurden in enger Zusammenarbeit mit Dr. Charles Petrie (Center for Design Research, Stanford University) sowie

1. Modeling Language and Operational Support for Software Processes

Jussi Stader und Peter Jarvis (Artificial Intelligence Application Institute, University of Edinburgh) diskutiert und weiterentwickelt. Aus der Zusammenarbeit mit Dr. Charles Petrie ging eine gemeinsam betreute Diplomarbeit hervor [15].

3 Durchgeführte Arbeiten und Ergebnisse

Ein zentraler Gegenstand des B2-Projektes war das MILOS-System [40], dessen Konzeption auf den Systemen MVP-L (AG Rombach) und CoMo-Kit (AG Richter) beruht. Die Implementierung geschah dabei in enger Zusammenarbeit mit dem Projekt A2. Die Schwerpunkte von B2 in der MILOS-Entwicklung lassen sich stichwortartig an den erzielten Ergebnissen ablesen:

- Entwicklung und Implementierung einer flexiblen Planungskomponente, die insbesondere auch den Import von Microsoft-Project-Plänen in MILOS erlaubt. Änderungen in einem MS-Project-Plan bewirken eine sofortige Aktualisierung des Zustands der MILOS-Workflowengine beim erneuten Import.
- Entwicklung und Implementierung der Event-Condition-Action- (ECA-) Regeln [20] und Erstellung eines Editors zur Erfassung und Löschung von Regeln als Ergänzung zu den automatisch vom System generierten Abhängigkeiten.
- Darauf basierend die Konzeption eines Notifikationsmechanismus für das Management von Planänderungen.
- Erstellung eines Produktmodells zur Verwaltung von domänenspezifischen Abhängigkeiten innerhalb von Produkten des Softwareentwicklungsprozesses; dies wurde am Beispiel von UML-Klassendiagrammen implementiert [12].

Der Änderungsdienst ist eine wesentliche Komponente eines industriellen Anwendungsprojekts bei der Firma Nortel in Calgary (durchgeführt von Prof. Maurer unter Mitwirkung von Gaststudenten aus Kaiserslautern).
Insgesamt wurden damit wichtige Aspekte der Verschmelzung von Planung und Ausführung realisiert. Ein wesentlicher Gesichtspunkt ist nun, dass viele Änderungsfragen die Planung selbst als Gegenstand haben. Die in dieser Periode vorgenommenen Realisierungen haben dies in die Planungsebene selbst integriert ohne eine explizite Meta-Planung zu erfordern. Für diese Vorgehensweise scheint aber jetzt im wesentlichen eine Grenze erreicht zu sein. Deshalb wird auf den Fortsetzungsantrag hingewiesen, in dem Meta-Planung zum expliziten Gegenstand der Betrachtung gemacht wird (vgl. auch [39]).

3.1 Methoden und Techniken zur Verwaltung von Planänderungen während der Projektdurchführung

Planänderungen während der Projektdurchführung wirken sich direkt auf die Ausführung aus und müssen sich unmittelbar in Aktionen umsetzen lassen. Solche Aktionen sind zunächst Notifizierungen der beteiligten Stellen, die durch ein System von ECA-

Regeln erfolgen. Dabei war die inhaltliche Schwierigkeit, die betroffenen Adressaten mit vertretbarem Aufwand möglichst genau zu ermitteln. Hier galt es, einen Kompromiss zu schließen, der hier so getroffen wurde, dass die Gruppe der von einer Änderung eventuell betroffenen (und daher zu notifizierenden) Benutzer restriktiv aus den dem System bekannten Abhängigkeiten extrahiert wurde. Die zweite wichtige Art von Aktionen betrifft die Änderungen in der Workflowengine, die ebenfalls unmittelbar zu erfolgen haben. Dabei dürfen bisherige Ergebnisse nicht verloren gehen, sie werden in der Object-Version-Management-Komponente von MILOS verwaltet und in dem OODBMS GemStone abgelegt, so dass sie jederzeit wieder zugreifbar sind.

Abbildung 1 demonstriert die Zusammenarbeit zwischen Planer und Planausführern, die vom MILOS-System integriert verwaltet wird.

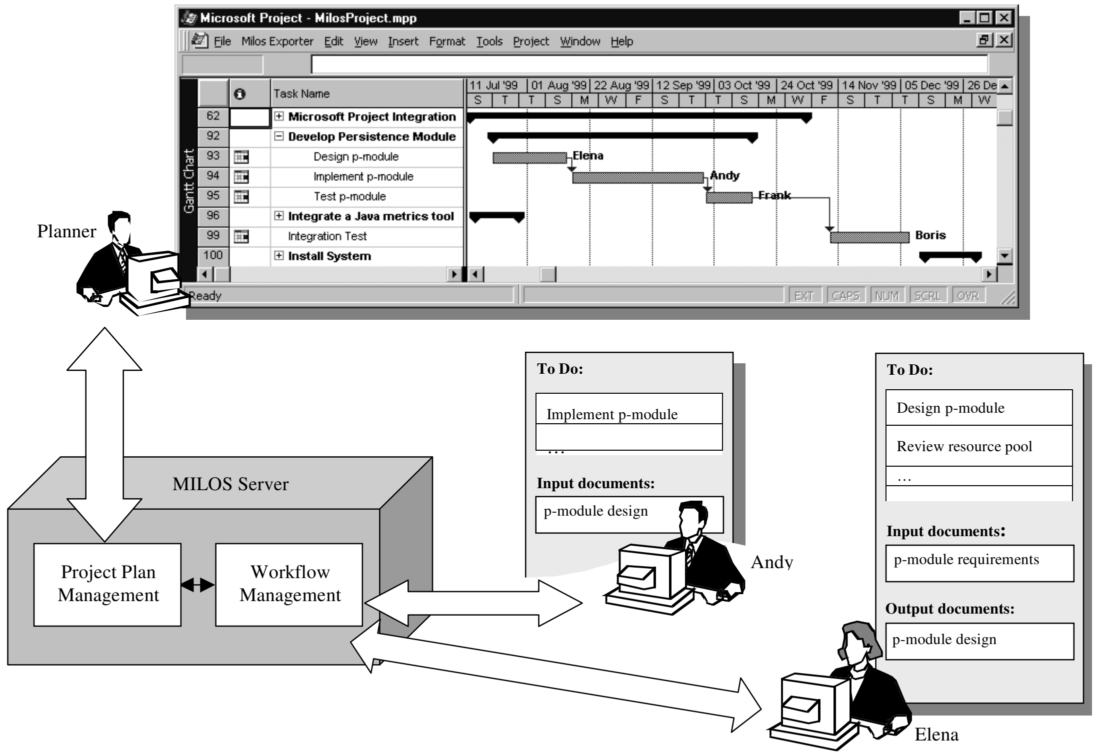

Abbildung 1: Integrierte Verwaltung von Planung und Ausführung in MILOS

3.2 Konzepte zur Herleitung und Verwaltung von domänenspezifischen Abhängigkeiten

Domänenspezifische Abhängigkeiten können auf sehr unterschiedlichen Ebenen formuliert werden. Hier wurde exemplarisch die Ebene der Klassen resp. Klassendiagramme gewählt. Die Abhängigkeiten stellen Wissen dar und betreffen Zusammenhänge, die vom Fachmann erkannt werden müssen und sich nicht aus rein formalen Bezügen (z.B. der Arbeitszuordnung von Projektbeteiligten) ergeben. Solche Abhängigkeiten sind auch in verallgemeinerter Form als Muster darstellbar. Die

Akquisition der Abhängigkeiten erfolgte mit Hilfe der Mitarbeiter der Projekte D1 und D2 und der Teams [10]. Zwei wesentliche Schwierigkeiten bestanden einerseits in der Herausarbeitung der grundsätzlichen Natur dieses Wissens und andererseits dessen Formulierung in der Form, dass ein unmittelbarer Zugriff möglich ist. Der hier erarbeitete Ansatz kann keinen Anspruch auf allgemeine Gültigkeit erheben, weil es sich um ein prinzipielles Problem des Wissensmanagements handelt.
Die identifizierten Abhängigkeiten und Abhängigkeitsmuster wurden in MILOS dargestellt. Zu diesem Zwecke wurde eine XML-Repräsentation entwickelt, welche komplexe Produkte strukturiert und Abhängigkeitsbeziehungen innerhalb von Produkten formalisiert [12].

3.3 Techniken zur Verwaltung von Entscheidungsbegründungen

Ein weiterer Forschungsschwerpunkt dieser Förderungsperiode war, Gründe für getroffene Entscheidungen (*Design Rationales*) dokumentieren und verwalten zu können. Damit werden zwei Ziele verfolgt:

1. Die kausalen Zusammenhänge zwischen getroffenen Entscheidungen einerseits sowie vorausgegangenen Entscheidungen und bereits erzeugten Produkten andererseits sollen protokolliert werden, um im Falle einer Entscheidungsrevision die betroffenen Produkte, Prozesse und Entscheidungen identifizieren zu können.
2. Begründungen für Entscheidungen und die Historie der Entscheidungsfindung sollen dokumentiert werden, um im Nachhinein Gründe für getroffene oder revidierte Entscheidungen rekonstruieren zu können.

In dieser Förderungsperiode wurde das Hauptaugenmerk unserer Forschung auf den ersten dieser Punkte gelegt. Hier war das übergeordnete Ziel, ein effizientes Änderungsmanagement zur Verfügung zu stellen, wie in 3.1 beschrieben. Es stellte sich heraus, dass die explizite Ausformulierung von Entscheidungsbegründungen dazu nur eingeschränkt notwendig war. Um von einer Änderung betroffene Personen möglichst gezielt informieren zu können genügte es vielmehr, die Abhängigkeiten zwischen einzelnen Prozessen (und ihren Lösungsalternativen), Produkten und Entscheidungen in Form von Event-Condition-Action-Regeln darzustellen: Viele Gründe für Entwurfsentscheidungen im Software Engineering betreffen vorliegende Alternativen und ihre Bewertungen, ausgewählte Ereignisse, vorliegende Bedingungen bzw. Constraints, und ihre Beziehungen untereinander.
Für unsere Zwecke konnten Design Rationales also inhaltlich auf die in 3.1 beschriebenen ECA-Regeln abgebildet werden. Eine nachträgliche Extraktion von Entscheidungsbegründungen ist nach wir vor für Dokumentationszwecke von Interesse. Sie wurde jedoch zurückgestellt, da die Dokumentation selbst nicht Gegenstand dieser Förderungsperiode war. Eine derartige Extraktion aus den vorliegenden ECA-Regeln ist jedoch leicht möglich und soll in der nächsten Förderungsperiode im Rahmen der Behandlung von Planungs- und Ausführungstraces wieder aufgegriffen werden. Hier können auch Abfolgen von Entscheidungen (*Entscheidungs-Historien*) in natürlicher Art und Weise angebunden werden.

4 Zusammenarbeit mit anderen Teilprojekten

Wie bereits erwähnt, erfolgte die Implementierung der im Teilprojekt B2 entwickelten Konzepte in MILOS in enger Zusammenarbeit mit dem Teilprojekt A2, welches auch deren Evaluierung vornahm. Dabei wurden Ergebnisse des Teilprojektes B1 zu Konzepten der Planung und Planinstrumentierung berücksichtigt.
Zur Identifikation von domänenspezifischen Abhängigkeiten wurden Ergebnisse der Teilprojekte D1 und D2 und der Teams herangezogen.
Die Ergebnisse des Teilprojekts B2 wurden in Zusammenarbeit mit dem Teilprojekt A1 in die Erfahrungsdatenbank des SFB (SFB-EB) eingebracht.

5 Eigene Veröffentlichungen im Berichtszeitraum

Projektarbeiten

[1] J. Bonnermann: *Ressourcenmanagement in MILOS; Projektarbeit Fachbereich Informatik;* Universität Kaiserslautern, 2000

[2] S. Christofidelli: *Zustandsmodelle in flexiblen Entwicklungsprozessen;* Projektarbeit Fachbereich Informatik, Universität Kaiserslautern, 1998

[3] J. Hohwiller: *Building a virtual marketplace for software development tasks;* Projektarbeit Fachbereich Informatik, Universität Kaiserslautern, 2000

[4] J. Petry: *Vor- und Nachbedingungen für Prozesse in MILOS;* Projektarbeit Fachbereich Informatik, Universität Kaiserslautern, 1999

[5] H. Pharo: *Modellierung von Meta-Aufgaben in MILOS*; Projektarbeit Fachbereich Informatik, Universität Kaiserslautern, 2000

[6] A. Raquet: *Spezifiaktion und Integration von Vor- und Nachbedingungen für Aufgaben in CoMo-Kit*; Projektarbeit Fachbereich Informatik, Universität Kaiserslautern, 1997

[7] B. Schadt: *Realisierung eines Frameworks zur graphischen Darstellung und Editierung von MILOS-Objekten in Java*; Projektarbeit Fachbereich Informatik, Universität Kaiserslautern, 2000

Diplomarbeiten

[8] C. Arenz: *Modellierung von Softwareengineering-Prozessen mit MILOS;* Diplomarbeit, Fachbereich Informatik, Universität Kaiserslautern, 1998

[9] U. Balg: *Wiederverwendung von Entwurfsmustern durch ein Workflow-Management-System;* Diplomarbeit, Fachbereich Informatik, Universität Kaiserslautern, 1998

[10] S. Christofidelli: *Verwaltung von Abhängigkeiten und Entscheidungsbegründungen bei der Planung und Ausführung von Software-Entwicklungsprozessen*; Diplomarbeit, Fachbereich Informatik, Universität Kaiserslautern, September 2000

[11] A. Herzlinger: *A Distributed Workflow Management Approach for Process-oriented Knowledge Management Systems;* Diplomarbeit, Fachbereich Informatik, Universität Kaiserslautern, 1999

[12] G. Landeck: *XML-basierte Repräsentation von Abhängigkeiten in komplexen Produkten des Software-Entwicklungsprozesses*; Diplomarbeit, Fachbereich Informatik, Universität Kaiserslautern, Oktober 2000

[13] S. Mungenas: *Entwicklung eines Ansatzes zur verteilten Abarbeitung von Workflows in einem virtuellen Unternehmen*; Diplomarbeit, Fachbereich Informatik, Universität Kaiserslautern, 1999

[14] F. Neblung: *Validierung und Erweiterung der Prozessmodellierungssprache MILOS;* Diplomarbeit, Fachbereich Informatik, Universität Kaiserslautern, 1998

[15] A. Raquet: *Dynamic Project Management for Distributed Construction Processes;* Diplomarbeit, Fachbereich Informatik, Universität Kaiserslautern, 1999

[16] T. Rüger: *Behandlung von Risiken in CoMo-Kit;* Diplomarbeit, Fachbereich Informatik, Universität Kaiserslautern, 1998

[17] M. Schaaf: *Ein System zum Kommunikationsmanagement verteilter Objekte in Java;* Diplomarbeit, Fachbereich Informatik, Universität Kaiserslautern, 1999

[18] M. Schlindwein: *A Framework Using Computer Agents to Provide Automatic Execution Support for Processes in Software Development Projects;* Diplomarbeit, Fachbereich Informatik, Universität Kaiserslautern, 1999

[19] S. Schmitt: *WWW-Based Software Process Modeling with CoMo-Kit;* Diplomarbeit, Fachbereich Informatik, Universität Kaiserslautern, 1998

Dissertationen

[20] Dellen, Barbara: *Change Impact Analysis Support for Software Development Processes;* Dissertation, Universität Kaiserslautern, Kaiserslautern, Feb. 2000.

Vorträge

[21] B. Dellen: *Internetbasiertes Wissensmanagement in Softwareprojekten*; Wissensmanagement 1999: Strategien der Gegenwart, Industriekonferenz, Frankfurt, 1998

[22] S. Goldmann: *Internetbasierte Prozessmodellierung und Projektplanung*; Wissensmanagement 1999: Strategien der Gegenwart, Industriekonferenz, Frankfurt, 1998

[23] M. M. Richter: *Flexibles Workflow-Management in Software-Entwicklungsprozessen;* Forschungsforum Wissensmanagement und schnelle Produktentwicklung, Universität Stuttgart, 1999

Tagungsbeiträge

[24] F. Bendeck, S. Goldmann, H. Holz, B. Kötting: *Coordinating Management Activities in Distributed Software Development Projects;* Proceedings of the 7 th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WET ICE '98), IEEE Computer Society Press, ISBN 0-8186-8751-7, pp. 33-38, 1998

[25] B. Dellen, F. Maurer: *Internet Based Software Process Support;* Proceedings of the 7 th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WET ICE '98), IEEE Computer Society Press, ISBN 0-8186-8751-7, pp. 27-32, 1998

[26] B. Dellen, F. Maurer: *Change Impact Analysis Support for Software Development Processes;* Journal of Applied Software Technology, International Academic Publishing, 1998

[27] B. Dellen, F. Maurer: *A Concept for an Internet-based Process-Oriented Knowledge Management Environment;* Proceedings of the KAW'98, Banff, Canada, 1998

[28] S. Goldmann, J. Münch, H. Holz: *Distributed Process Planning Support with MILOS;* Proceedings of the 11th International Software & Engineering and Knowledge Engineering Conference (SEKE '99), Knowledge Systems Institute, ISBN 1-891706-01-2, 1999

[29] S. Goldmann, J. Münch, H. Holz: *A Meta-Model for Distributed Software Development;* Proceedings of the 8th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WET ICE '99), IEEE Computer Society Press, 1999

[30] S. Goldmann, , J. Münch, H. Holz: MILOS: *A Model of Interleaved Planning, Scheduling, and Enactment;* presented at the ICSE '99 Workshop on „Software Engineering over the Internet"; Web proceedings only: http://sern.cpsc.ucalgary.ca/~maurer/ICSE99WS/Program.htm

[31] H. Holz, S. Goldmann, F. Maurer: *Working Group Report on Coordinating Distributed Software Development Projects;* Proceedings of the 7 th IEEE International Workshops on Enabling Technologies: Infrastructure for Collab-

orative Enterprises (WET ICE ’98), IEEE Computer Society Press, ISBN 0-8186-8751-7, pp. 69-72, 1998

[32] F. Maurer, H. Holz: *Process-Oriented Knowledge Management For Learning Software Organizations;* Proceedings of the 12th Knowledge Acquisition Workshop (KAW '99), Banff, Canada, 1999

[33] F. Maurer, G. Succi, H. Holz, B. Kötting, S. Goldmann, B. Dellen: *Software Process Support over the Internet;* Proceedings of the 21 th International Conference on Software Engineering (ICSE ’99), ACM Press, ISBN 1-58113-074-0, pp. 642-645, 1999

[34] F. Maurer, H. Holz: *Process-Centered Knowledge Organization for Software Engineering;* Aha, D.W.,& Munoz Avila, H. (Eds.): Exploring Synergies of Knowledge Management and Case-Based Reasoning: A 1999 AAAI Workshop (Working Notes). (Technical Report AIC-99-008). Washington, DC: Naval Research Laboratory, Navy Center for Applied Research in Artificial Intelligence, 1999

[35] F. Maurer, B. Dellen, H. Holz: *Process Support for Virtual Software Organizations;* Proceedings of the 11th International Software & Engineering and Knowledge Engineering Conference (SEKE ‘99), Knowledge Systems Institute, ISBN 1-891706-01-2, 1999

[36] Ch. Petrie, S. Goldmann, A. Raquet: *Agent-Based Project Management;* Proc. of the Intnl.Workshop on Intelligent Agents in CSCW, Deutsche Telekom, Dortmund, Sept. 1998, p. 1-1. (short version of paper of the same name in Springer LNAI 1600)

[37] M.M. Richter, B. Kötting, S. Goldmann: *Flexibles Workflow Management in Software Development Processes;* to appear in Proceedings of the “Forschungsforum Wissensmanagement und schnelle Produktentwicklung”, Stuttgart, Germany, 1999

Zeitschriften / Monografien

[38] S. Goldmann, B. Kötting: *Software Engineering over the Internet;* IEEE Internet Computing, Vol. 3, No 4, July/August 1999, pp 93 – 94, IEEE Computer Society Press, 1999

[39] S. Goldmann, J. Münch, H. Holz: *Distributed Process Planning Support with MILOS;* Int. Journal of Software Engineering and Knowledge Engineering, October 2000 (to appear)

[40] F. Maurer, B. Dellen, F. Bendeck, S. Goldmann, H. Holz, B. Kötting, M. Schaaf: *Merging project planning and web-enabled dynamic workflow for software development;* IEEE Internet Computing, Special Issue on Internet-Based Workflow, May/June 2000 (to appear)

[41] Ch. Petrie, S. Goldmann, A. Raquet: *Agent-Based Project Management;* Springer Lecture Notes on Artificial Intelligence (LNAI) 1600 special volume on “Artificial Intelligence Today”, Springer Verlag, 1998

[42] M. M. Richter: *Knowledge-Based Systems;* Intelligence and Artificial Intelligence (ed. U. Ratsch, M.M. Richter, I.-O. Stamatescu), Springer Verlag, 1998

6 Fremdveröffentlichungen

[43] G. Cugola, C. Ghezzi: *Design and Implementation of PROSYT: A Distributed Process Support System;* in IEEE Proceedings of the 8th International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, USA, 1999

[44] S. Goldmann, B. Kötting: *Working Group Report of the 2nd Workshop on Coordinating Distributed Software Development Projects;* Proceedings of the 8th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WET ICE ’99), IEEE Computer Society Press, 1999

[45] John Grundy, John Hosking, Rick Mugridge: *Coordinating Distributed Software Development Projects with Integrated Process Modelling and Enactment Environments;* in IEEE Proceedings of the 7th International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, USA, 1998

[46] P. Jarvis, J. Stader, A. Macintosh, J. Moore, P. Chung: *What Right Do You Have to Do That? Infusing Adaptive Workflow Technology with Knowledge about the Organisational and Authority Context of a Task;* Proceedings of First International Conference on Enterprise Information Systems (ICEIS-99), Setubal, Portugal, March 1999

[47] B. Kötting, F. Maurer: *Approaching Process Support for Virtual Software Corporations;* presented at the ICSE ’99 Workshop on Software Engineering over the Internet; Web proceedings only: http://sern.cpsc.ucalgary.ca/~maurer/ICSE99WS/Program.htm

[48] B. Kötting, F. Maurer: *A Concept for Supporting the Formation of Virtual Corporations through Negotiation;* Proceedings of the 8 th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WET ICE ’99), IEEE Computer Society Press, 1999

Teilprojekt B4

Generische Kommunikationssysteme

Leiter

Prof. Dr. Reinhard Gotzhein
geb. 9.10.1956
Universität Kaiserslautern
Postfach 3049
67653 Kaiserslautern
Tel.: 0631/205-3426
Fax: 0631/205-3956
gotzhein@informatik.uni-kl.de

Mitarbeiter

Ergänzungsausstattung:

Dipl. Inform. B. Geppert (bis 15.5.1999)
Dipl. Inform. C. Peper
Dipl. Inform. J. Thees (seit 1.2.2000)

Grundausstattung:

Dipl. Inform. F. Rößler (bis 15.5.1999)
Dipl. Inform. P. Schaible (seit 15.5.1999)

1 Kurzfassung der Aufgabenstellung

Langfristiges Ziel dieses Teilprojekts ist die Bereitstellung von Verfahren und Techniken zur Entwicklung anwendungsangepasster, maßgeschneiderter Kommunikationssysteme unter besonderer Berücksichtigung von Echtzeitanforderungen. Dabei wird der Vielfalt der Kommunikationsanforderungen sowie der Variabilität der Basistechnologien durch die generische Auslegung von Kommunikationssystemen und Entwicklungsprozessen Rechnung getragen. Die Qualität der Ergebnisse wird durch die Bereitstellung und den Einsatz domänenspezifischer Methoden, formaler Techniken und rechnergestützter Werkzeuge sichergestellt [53].

Zur Erreichung der langfristigen Ziele des Teilprojekts wurden in der Förderperiode die folgenden Forschungsschwerpunkte bearbeitet:

a) Bereitstellung von formalen Methoden, Techniken und Werkzeugen zur generischen Auslegung von Kommunikationssystemen unter besonderer Berücksichtigung von Echtzeitanforderungen;
b) Erfassung, Verbesserung und Wiederverwendung der relevanten Aspekte des Entwicklungsprozesses von Kommunikationssystemen;
c) Erprobung der Methoden, Techniken und Werkzeuge zur Entwicklung generischer Kommunikationssysteme im Anwendungsfeld "Gebäudeautomation".

2 Entwicklung des Teilprojekts in der Förderperiode

Exponate auf Industriemessen

Einige der wissenschaftlichen Ergebnisse des Teilprojekts wurden in Exponaten auf der CeBIT'98 [58] [59] sowie auf dem Technologietransferkongress "IKT-Anwendungen auf Netzen" im Rahmen des III. Mannheimer Unternehmerforums [60] vorgestellt.

Präsentationen auf internationalen Veranstaltungen

An exponierter Stelle konnte der im Rahmen des Teilprojekts entwickelte SDL-Pattern-Ansatz auf zwei internationalen Veranstaltungen der SDL-Community, die traditionell eine besonders starke Industriebeteiligung aufweisen, präsentiert werden:

- Während der Internationalen Konferenz "*9th SDL Forum 1999*" wurde das Tutorial "*Pattern-Based Reuse with SDL*" gehalten [37].
- Im Rahmen des Internationalen Workshops "*SDL and MSC (SAM'98)*" wurde die halbtägige Sitzung "*Patterns and SDL*" organisiert und geleitet (eingeladener Sprecher: Erich Gamma) [62].

Guest Editor "Journal of Universal Computer Science"

In diesem Jahr erscheint eine Sonderausgabe der Internationalen Zeitschrift *Journal of Universal Computer Science* (kurz: *J.UCS*), die einen besonderen thematischen

Bezug zum SFB 501 besitzt [54]. Das Thema der Sonderausgabe lautet "*Requirements Engineering: The Light Control Case Study*", Herausgeber sind Egon Börger und Reinhard Gotzhein. Der Call for Papers (URL: http://rn.informatik.uni-kl.de/~recs) fand im September 1999 statt, zum Berichtszeitpunkt sind die Begutachtungsphase und die Auswahl der akzeptierten Beiträge abgeschlossen. Der besondere SFB-Bezug besteht in der Wahl der Fallstudie, die auf einer Fallstudie aus den Querschnittprojekten Q1 und Q4 basiert. Damit werden die im Requirements Engineering erzielten Ergebnisse des SFB international vergleichbar.

Associate Rapporteur der ITU-T "Formal Semantics Definition for SDL-2000"

Während des ITU-T SG10 Meetings in Genf wurde eine revidierte Fassung der Norm Z.100, welche die im SFB in mehreren Teil- und Querschnittprojekten eingesetzte Entwurfssprache SDL definiert, verabschiedet. Die Arbeit an der formalen Semantik von SDL-2000 begann Anfang 1998 [43] und wird seither vom Projektleiter des Teilprojekts B4 in der Funktion des Associate Rapporteurs mitgetragen. Im selben Jahr fiel die Entscheidung, den Formalismus der Abstract State Machines (ASMs) zur operationalen Definition der formalen Semantik einzusetzen [42] [14]. Die Fertigstellung der Semantik ist für November 2000 vorgesehen.

3 Durchgeführte Arbeiten und Ergebnisse

3.1 Arbeitsschwerpunkt A: Generische Kommunikationsdienste

Arbeitspaket A1: Methoden und Techniken

Prozessmodell. Das in der 1. Förderperiode entwickelte erste Prozessmodell für die Problemanalyse wurde aufgrund der Erfahrungen mehrerer Fallstudien (s. Arbeitspaket A3) weiterentwickelt und konsolidiert [44] [48]. An dieser Stelle seien vier Aspekte des Prozessmodells besonders hervorgehoben:

- Die Problemanalysephase verläuft iterativ, wobei jede Iteration neben der Formalisierung ein Review mit Kundenbeteiligung auf der Basis einer schematisch übersetzten, semiformalen Fassung vorsieht.
- Ergebnis der Analysephase ist eine formale Problemspezifikation, die direkt als Eingabe für die Entwurfsphase dient. Diese Spezifikation liegt bei positivem Abschluss des Reviews mit Kundenbeteiligung bereits fertig vor.
- Zur Formalisierung der Problembeschreibung werden musterbasierte Ansätze verwendet.
- Das Prozessmodell beinhaltet ferner Schritte zur Gewinnung von Mustern durch Analyse von Entwicklungsdokumenten abgeschlossener Projekte [44].

Das Prozessmodell ist unabhängig von der eingesetzten Spezifikationstechnik. Es wurde zur Durchführung von Fallstudien mit der in Kooperation mit Teilprojekt C1 entwickelten Spezifikationstechnik FoReST [57] instanziiert.

Spezifikationstechniken. In der ersten Förderperiode wurde zur formalen Spezifikation von Anwendungsanforderungen und Kommunikationsdiensten die auf das Anwendungsfeld zugeschnittene Realzeit-Logik tRTTL (tailored Real Time Temporal Logic) eingesetzt. Die beim Einsatz von tRTTL auftretenden Defizite hinsichtlich Strukturierung und Ermittlung von Kommunikationsanforderungen konnten mit der Entwicklung der *Formal Requirements Specification Technique* (*FoReST)* [57] [56] [16] (in Kooperation mit Teilprojekt C1), die objektorientierte Konzepte und tRTTL integriert, beseitigt werden.

Referenzmodell für Problemspezifikationen. Aufbauend auf dem in [65] vorgestellten Referenzmodell für Anforderungen und Spezifikationen wurde in Kooperation mit Teilprojekt C1 ein Referenzmodell für Problemspezifikationen definiert [46] [57]. Das Referenzmodell unterscheidet zwischen beobachteten und kontrollierten Größen sowie zwischen Umgebungs- und Systemgrößen und trifft darüberhinaus Festlegungen bezüglich ihrer Sichtbarkeit.

Ermittlung der Kommunikationsanforderungen. Zur Ermittlung der Kommunikationsanforderungen wird die Problemspezifikation - unterstützt durch den Einsatz von *Service Patterns* - verfeinert. Über die Definition der Service Patterns, insbesondere das Analysemodell als Bestandteil dieser Definition, kann ein unmittelbarer Zusammenhang zu Requirement Patterns hergestellt werden. Damit ist aus dem Einsatz eines Requirement Patterns in der Problemspezifikation ein direkter Schluss auf die zur Verfeinerung anwendbaren Service Patterns möglich. Die Adaption des Service Patterns ist teilweise aus der bereits gegebenen Adaption des Requirement Patterns ermittelbar. Die Verfeinerung kann in mehreren Schritten erfolgen [47]. Die praktische Erprobung dieser Vorgehensweise sowie der Aufbau einer initialen Service-Pattern-Sammlung sind im Gange und sollen noch in dieser Förderperiode zu einem abschließenden Ergebnis führen.

Arbeitspaket A2: Verbreiterung und Verwaltung der Dienstmustersammlung

Requirement-Pattern-Sammlung. Im Berichtszeitraum wurden mehrere umfangreiche Fallstudien zur Problemspezifikation durchgeführt (s. Arbeitspaket A3), bei denen die Requirement-Pattern-Sammlung intensiv eingesetzt wurde. Im Anschluss an diese Fallstudien fand jeweils eine Überarbeitung der Mustersammlung statt. So hat sich die Sammlung gegenüber der initialen Version stark verändert, auch wenn sich die Anzahl der Muster kaum erhöht hat. Verschiedene Stadien von Requirement Patterns sind in [44], [48] und [15] dokumentiert. In [44] wird die Gewinnung von Requirement Patterns im allgemeinen und des LazyReaction-Patterns im besonderen ausführlich dargestellt.

Service-Pattern-Sammlung. Die Entwicklung der Service-Pattern-Sammlung ist wie bereits erörtert eng an die Requirement Patterns gekoppelt. Aufgrund dieser starken Abhängigkeiten und bedingt durch die neuen Resultate im Anforderungsbereich ist der Aufbau der Service-Pattern-Sammlung noch im Gange, so dass hier erst gegen Ende der laufenden Förderperiode aussagekräftige Ergebnisse vorliegen werden.

Werkzeugunterstützung. Die aktuelle Requirement-Pattern-Sammlung ist im organisatorischen Teil der Erfahrungsdatenbank SFB-EDB (s. Teilprojekt A1) abgelegt und kann bei der Durchführung von Projekten herangezogen werden. Ferner wurde das in Teilprojekt C1 entwickelte rechnergestützte Werkzeug *xforest* [64] zur Verwaltung der Requirement Patterns eingesetzt.

Arbeitspaket A3: Engineering von maßgeschneiderten Kommunikationsdiensten im SFB-Anwendungsfeld

Die in den Arbeitspaketen A1 und A2 entwickelten Methoden, Techniken, Werkzeuge und Mustersammlungen wurden in mehreren umfangreichen Fallstudien [57] [45] [18] (s. Querschnittprojekt Q4) sowie einer Variante der Problemspezifikation "Lichtsteuerung", die als Ausgangspunkt für die durchgängige Entwicklung einer verteilten Gebäudesteuerung dient, eingesetzt und erprobt. Die darin gesammelten Erfahrungen sind jeweils unmittelbar in die Weiterentwicklung und Konsolidierung dieser Ansätze eingeflossen.

3.2 Arbeitsschwerpunkt B: Generische Kommunikationsprotokolle

Arbeitspaket B1: Modellierung und Erfassung der Kompositionsintelligenz

Für die Klasse der Interaktionsmuster wurden ein Kompositionsmodell sowie Sprach- und Werkzeugunterstützung entwickelt.

Kompositionsmodell. Der Protokollentwurf mit SDL-Patterns unterscheidet sich von traditionellen SDL-Entwurfsmethoden durch eine kollaborationsorientierte Systemsicht [37]. Kollaborationen sind im Sinne von Systemfunktionalitäten (z.B. Verbindungsaufbau oder Datentransfer) zu verstehen und stellen damit nur einen Teil des Gesamtsystems dar. Sie sind ein wesentliches Strukturierungsmittel zur Beschreibung verteilter Systeme und müssen vom Protokollingenieur entsprechend der funktionalen und nicht-funktionalen Systemanforderungen entworfen werden.

Sprachunterstützung. Zur Verbesserung des Entwurfsprozesses wurde die Notation CoSDL entwickelt, die es erlaubt, initiale Entwurfsideen in Form von Kollaborationsdiagrammen zu beschreiben [37]. CoSDL ist eine graphische Notation, die sich stark an SDL anlehnt und (ähnlich zu MSC) Interaktionsverhalten zwischen SDL-Prozessen beschreibt. Im Gegensatz zu MSC basiert CoSDL aber nicht auf der Spezifikation von Beispielabläufen, sondern beschreibt vollständige Kollaborationen zwischen SDL-Prozessen. Im Prozessmodell des SDL-Pattern-Ansatzes werden Kollaborationsdiagramme während der Systemanalyse eingesetzt und unterstützen die Selektion von SDL-Patterns. CoSDL wurde beim Re-Engineering eines ATM-Signalisierungsprotokolls erprobt [4].

Werkzeugunterstützung. Das prototypische Werkzeug SPEEDI (s. Arbeitspaket B2) unterstützt die kollaborationsorientierte Systemsicht bereits teilweise.

Arbeitspaket B2: Verbreiterung und Bewertung des Protokollbaukastensystems

Das Protokollbaukastensystem besteht aus einem inkrementellen, musterbasierten Entwurfsprozess sowie einer SDL-Pattern-Sammlung. Im Berichtszeitraum sollte das initiale Protokollbaukastensystem [35] [36] unter anderem durch eine geeignete Werkzeugunterstützung ausgebaut und darüber hinaus bewertet werden.

Musterbasierter Entwurfsprozess. Ausgangspunkt für den Protokollentwurf sind Kommunikationsanforderungen, die in einem ersten Schritt mit dem Ziel der iterativen Behandlung in Teilanforderungen dekomponiert werden. Mit jedem nun folgenden Durchlauf werden weitere Teilanforderungen analysiert und in einen SDL-Entwurf überführt. Aufgabe der Kollaborationsanalyse ist dabei die Erstellung einer konzeptionellen Systemarchitektur (bzw. die Ergänzung der Architektur des letzten Entwicklungsschrittes) sowie die Identifikation und Modellierung von Kollaborationen zwischen den vorhandenen Architekturelementen. Der eigentliche Entwurfsschritt wird ebenfalls inkrementell durchgeführt, wobei die zuvor identifizierten Kollaborationen schrittweise in den vorhandenen SDL-Entwurf integriert werden. Dies kann durch Anwendung von SDL-Patterns oder konventionell, ohne Musterunterstützung, geschehen. Die SDL-Pattern-Anwendung besteht ihrerseits aus den Teilschritten Selektion, Adaption und Komposition. Das Resultat eines Durchlaufs ist eine ausführbare SDL-Entwurfsspezifikation, die validiert und in nachfolgenden Durchläufen ergänzt wird. Sind alle Teilanforderungen abgearbeitet, ist der inkrementelle Entwurfsprozess abgeschlossen, und die resultierende validierte SDL-Entwurfsspezifikation dient als Eingabe für die nachfolgende Implementierungsphase.

SDL-Pattern-Sammlung und Beschreibungsrahmen. Der in der ersten Förderperiode definierte Beschreibungsrahmen für SDL-Patterns [33] wurde durch die Einführung von Metasymbolen zur Beschreibung der SDL-Fragmente weiterentwickelt. Die bisherige Notation unterstützte lediglich die Darstellung der invarianten Teile einer Entwurfslösung adäquat, nicht jedoch die Darstellung der zulässigen Adaptionen an einen konkreten Kontext. Die eingeführten Metasymbole erfassen die erlaubten Variationen und werden bei der Adaption eines Musters aufgelöst, so dass ein komponierbares SDL-Fragment entsteht. Die SDL-Pattern-Sammlung selbst wurde aufgrund der in mehreren Fallstudien gesammelten Erfahrungen weiterentwickelt und ausgebaut (s. Arbeitspaket B3). Ferner wurde der Zusammenhang zwischen der in der objektorientierten Modellierung verwendeten Spezialisierung und der Anwendung von SDL-Patterns untersucht [40].

Evaluierung und Quality Improvement Paradigm. Zur Evaluierung und systematischen Verbesserung wurde der SDL-Pattern-Ansatz gemäß des Quality Improvement Paradigms (QIP) erweitert [31] [32]. In Kooperation mit Teilprojekt A1 wurde ein Messprogramm zur Erfassung des Verbesserungs- und Wiederverwendungspotenzials konzipiert [10] [11] [12]. Die während der Durchführung von Fallstudien mit Hilfe von maßgeschneiderten Fragebögen erfassten Messdaten wurden analysiert, die Ergebnisse dienten als Ausgangspunkt für die Verbesserung des Ansatzes. Unter anderem wurden verschieden Maturity-Levels für SDL-Patterns definiert [38]. Die gesammelten Erfahrungselemente wurden in der SFB-EDB verwaltet. Weiterhin wurden die durchgeführten Fallstudien mit ihren Messdaten in der SFB-EDB dokumentiert. Der SDL-Pattern-Ansatz wird seit 1998 im Praktikum “Entwicklung von Kommunikationssystemen” der AG Rechnernetze eingesetzt und dort in diesem Jahr in Kooperation mit Teilprojekt A1 evaluiert.

Werkzeugunterstützung. Zur Unterstützung der musterbasierten Protokollentwicklung wurde das rechnergestützte Werkzeug *SPEEDI* (SDL-PattErn EDItor) realisiert [29] [56]. SPEEDI unterstützt die *Anwendung* (Selektion, Adaption und Einbettung) von SDL-Patterns in SDL-Spezifikationen auf der Basis von SDL-PR [8]. Ferner *dokumentiert* SPEEDI die Reihenfolge und Platzierung von Musteranwendungen [3], wodurch die orthogonale, musterbasierte Struktur der SDL-Spezifikation festgehalten wird. Das *Browsen* durch den integrierten SDL-Pattern-Pool wird durch SPEEDI bereits rudimentär unterstützt [8]. Weitere Unterstützung wird durch die Einlagerung der Mustersammlung in die SFB-EDB geboten [11].

Arbeitspaket B3: Engineering von maßgeschneiderten Protokollen im SFB-Anwendungsfeld

Gegenstand dieses Arbeitspaketes ist die Durchführung von Fallstudien zur Evaluierung und Verbesserung des SDL-Pattern-Ansatzes. Zur Bewertung wurde zunächst eine *informelle* Evaluierung anhand folgender Fallstudien durchgeführt:

Neuentwicklung von Protokollen. Entwickelt wurden Kommunikationssubsysteme auf der Basis von Token Ring und CAN [7]. Diese Fallstudien dienten als Machbarkeitsstudien für den Einsatz des SDL-Pattern-Ansatzes.

Reengineering von Protokollen. Betrachtet wurden Protokolle aus der Internet Protocol Suite: IPv6 [1], RTP [2] und ST2+ [6] [13]. Diese Fallstudien dienten der Untersuchung, ob bereits bestehende Protokolle mit ihren integrierten Entwurfslösungen durch das Protokollbaukastensystem erfasst sind bzw. ob wesentliche Lücken in der SDL-Pattern-Sammlung bestehen.

Reengineering von SDL-Spezifikationen. Betrachtet wurde das ATM-Signalisierungsprotokoll (UNI 3.1) [4]. In dieser Fallstudie konnte gezeigt werden, dass eine musterbasierte Spezifikation wesentlich zur Systematisierung und Verständlichkeit beiträgt. Es stellte sich außerdem heraus, dass die in der Mustersammlung angebotenen SDL-Entwurfslösungen im wesentlichen denen entsprechen, die von externen SDL-Designern verwendet worden sind.

Die Reihenfolge der durchgeführten Fallstudien ist durch einen zunehmenden Komplexitätsgrad sowie zunehmende Durchgängigkeit des Ansatzes gekennzeichnet. Die gewonnen Erkenntnisse wurden informell festgehalten und flossen jeweils in eine Weiterentwicklung des Ansatzes ein. Beispiele hierfür sind Verbesserungen in der Beschreibung bestehender Muster, Verfeinerung von Prozessschritten oder auch die Entwicklung und Einführung einer Notation zur Beschreibung von Kollaborationen.

Arbeitspaket B4: Automatische Kodegenerierung für maßgeschneiderte Protokolle

Zur prototypischen Implementierung von Kommunikationssubsystemen wurde der Kodegenerator *EnvGen (Environment Generator)* [34] [5] entwickelt. EnvGen ermöglicht in Kombination mit dem kommerziellen Kodegenerator SDT-Cadvanced eine vollautomatische Implementierung und Einbettung von SDL-Spezifikationen. Dazu werden direkt aus der SDL-Spezifikation Umgebungsfunktionen für die Laufzeitbibliothek von SDT-Cadvanced generiert, mit deren Hilfe offene SDL-Systeme über existierende Basistechnologien kommunizieren. Derzeit wird die Anbindung an ATM, CAN, QNX-IPC und BSD-Sockets unterstützt, die Anbindung an UART ist in Vorbereitung. Weitere Basistechnologien können bei Bedarf integriert werden. Ein Teilergebnis dieser Arbeit zeigt, dass sich das im Teilprojekt etablierte Pattern-Konzept auch für den Bereich der Umgebungsschnittstelle einsetzen läßt [56] [52]. In Ergänzung zu den in Arbeitspaket B3 beschriebenen Fallstudien wurden prototypische Implementierungen des objektbasierten Kommunikationssubsystems auf der Basis von CAN [34] und eines ATM-Signalisierungsprotokolls (UNI 3.1) [4] automatisch generiert und getestet.

3.3 Arbeitsschwerpunkt C: Echtzeitfähige generische Kommunikationsprotokolle

Arbeitspaket C1: Integration von QoS-Architekturkomponenten in das Protokollbaukastensystem

Das Ergebnis der Analyse von Gemeinsamkeiten zwischen Echtzeitkommunikationssystemen wurde in der ersten Förderperiode in einem allgemeinen QoS-Architekturmodell für Kommunikationssysteme festgehalten. Danach umfassen QoS-Architekturen im wesentlichen Komponenten für die Benutzerinteraktion, die Reservierung der räumlich verteilten Ressourcen, die Zugangskontrolle, das QoS-Monitoring, die Verkehrskontrolle und das Scheduling. Für jede dieser Architekturkomponenten lässt sich in der Literatur eine Fülle von Mechanismen zur Realisierung nachweisen. Der Versuch, eine Auswahl dieser Mechanismen in das Protokollbaukastensystem zu integrieren, hat unterschiedliche Erkenntnisse erbracht:

- Durch mehrere Fallstudien zur Konfigurierung von Ressourcenreservierungsprotokollen [34] [41] [50] konnte der Protokollbaukasten soweit ausgebaut werden, dass in der aktuellen Ausbaustufe eine ausreichende Überdeckung der SDL-Entwurfsspezifikationen erzielt wird. Eine weitere Spezialisierung der Protokollbausteine für Ressourcenreservierungsprotokolle hätte ein zu geringes Wiederverwendungspotenzial zur Folge.
- Das Protokollbaukastensystem unterstützt die Erstellung von SDL-Entwurfsspezifikationen und ist insofern auf die Entwicklung verteilter, nebenläufiger Systeme ausgerichtet. Einige Komponenten der QoS-Architektur wie die Zugangskontrolle, die Verkehrskontrolle und das Scheduling haben jedoch einen vorwiegend algorithmischen, sequentiellen Charakter und werden daher nicht unter Verwendung von SDL konzipiert. Aus diesem Grund können sie

nicht durch den Protokollbaukasten in seiner jetzigen Konzeption abgedeckt werden.

- Der Protokollbaukasten dient zur generischen Auslegung des Entwicklungsprozesses von (Echtzeit-)Kommunikationssubsystemen, indem Architekturstile und Entwurfslösungen mit hohem Wiederverwendungspotenzial in einem Pool gesammelt und verbessert werden. Das konfigurierte Produkt selbst hat keinen besonders ausgeprägten generischen Charakter. Der Versuch, das QoS-Achitekturmodell (bzw. QoS-Komponenten) in den Protokollbaukasten zu integrieren, hat zu einem verbesserten Ansatz geführt, der auch eine generische Auslegung der konfigurierten Produkte vorsieht. Ziel ist hier die Entwicklung einer generischen Architektur, die es erlaubt, eine Familie vorbestimmter (Echtzeit-)Kommunikationssubsysteme zu instanziieren.

Arbeitspaket C2: Analyse des deterministischen Zeitverhaltens und Effizienzsteigerungsmaßnahmen für die Konfigurierung von zeitkritischen Kommunikationsprotokollen

In Arbeitspaket C2 sollten Verfahren zur Analyse der Vorhersagbarkeit des Zeitverhaltens und Maßnahmen zur Effizienzsteigerung konfigurierter Protokolle entwickelt werden. Aufgrund des Weggangs zweier Mitarbeiter des Teilprojekts B4 im Mai 1999 konnte dieses Arbeitspaket nicht wie vorgesehen bearbeitet werden.

Arbeitspaket C3: Fallstudien zum Entwurf von Echtzeitprotokollen mit generischen Methoden

Um die Konfigurierung von Protokollspezifikationen durch die Implementierung von Prototypen absichern zu können, wurde die Auswahl der aus der ersten Förderperiode zur Verfügung stehenden Kommunikationsbasistechnologien [51] ergänzt. Die neu eingesetzten Basistechnologien (ATM, CAN) wurden wo nötig zu im Sinne des QoS-Architekturmodells vollständigen Echtzeitkommunikationssystemen für die im SFB eingesetzte Zielplattform ausgebaut [34] [49].

Für den CAN-Feldbus wurde ein echtzeitfähiges Kommunikationssubsystem entwickelt, das im Gegensatz zu herkömmlichen Installationen zur Laufzeit ein dynamisches Ressourcenmanagement erlaubt [7]. Als wesentliche Architekturkomponenten wurden ein objektbasiertes Kommunikationsprotokoll, Netzwerkmanagementprotokolle zur Knoteninitialisierung und Ressourcenreservierung, Quality-of-Service Mechanismen zur Verkehrskontrolle und zum Scheduling sowie ein Gerätetreiber zur Kommunikation mit der CAN-Netzwerkkarte entwickelt. Die Ressourcenmanagementanteile wurden mit Hilfe des Protokollbaukastensystems entwickelt.

4 Zusammenarbeit mit anderen Teilprojekten

In Kooperation mit **Teilprojekt A1** wurde der SDL-Pattern-Ansatz in die SFB-EDB integriert. Auf methodischer Seite ist die Konzipierung und Erstellung eines konkreten Messprogramms auf Basis von GQM zur Evaluierung und Verbesserung des SDL-

Pattern-Ansatzes hervorzuheben [10] [11] [12], das in einer Fallstudie [2] eingesetzt wurde. Die Integration des Messprogramms in den SDL-Pattern-Ansatz resultiert zunächst in einer Erweiterung des SDL-Pattern-basierten Entwurfsprozesses um einen Verbesserungszyklus [31] [32] [38]. Weiterhin wurde die SFB-EDB um die Definition SDL-Pattern-spezifischer Ablagestrukturen angereichert und mit Inhalten (Prozessmodell, SDL-Pattern-Sammlung etc.) gefüllt. Dies wurde zunächst für den Web-basierten Prototypen der SFB-EDB durchgeführt [39]; eine Umstellung der SFB-EDB auf objektrelationale Datenbanktechnologie wird in **Teilprojekt A3** vorgenommen, wobei der SDL-Pattern-Ansatz als Anwendungsbeispiel dient [30] [9]. Der aus der A1/B4-Kooperation resultierende Teil der SFB-EDB wurde in der SDL-Community im Rahmen einer Tooldemonstration [61] sowie eines Tutorials [37] vorgestellt.

Die im SFB verwendeten Entwicklungsmethodiken werden zur Einlagerung in die SFB-EDB von **Teilprojekt B1** in Form formaler Prozessmodelle erfasst. Für SDL-spezifische Entwicklungsprozesse - u.a. der SDL-Pattern-basierte Entwurfsprozess oder der von Telelogic unterstützte SOMT-Prozess - geschah dies unter Nutzung des Expertenwissens von B4 [17].

Mit **Teilprojekt B5** wurde ein SFB-Praktikum konzipiert und durchgeführt, das die Ansätze beider Teilprojekte beim Entwurf und der Implementierung verteilter Systeme integriert. Zum Einsatz kamen dort der SDL-Pattern-Ansatz, kombiniert mit einer anschließenden Kodegenerierung und Anbindungsmechanismen an betriebssystemtechnische Aspekte. Ferner fand während der gesamten Förderperiode eine enge Abstimmung der Teilprojekte mit dem Ziel, die Entwicklung von Kommunikations- und Betriebssystemkomponenten zu verzahnen und zu synchronisieren, statt.

In Kooperation mit **Teilprojekt C1** wurde die Spezifikationstechnik FoReST entwikkelt und in mehreren Fallstudien eingesetzt. Diese Aktivitäten sind in Arbeitsschwerpunkt A sowie in Querschnittprojekt Q4 näher beschrieben.

In Abstimmung mit **Teilprojekt D1** wurde im Rahmen einer gemeinsam betreuten Diplomarbeit [7] ein Echtzeitkommunikationssubsystems auf Basis von CAN entwikkelt. Die Aufgabe von B4 bestand in der musterbasierten Entwicklung des Subsystems [41], hierbei insbesondere der objektbasierten Kommunikation auf Anwendungsebene, desweiteren aus der Anbindung an das QoS-Architekturmodell [49], sowie der Kodegenerierung [34]. Das Subsystem wurde zur verteilten Steuerung eines simulierten Inversen Pendels eingesetzt und auf der CeBIT präsentiert [59].

Ein Schwerpunkt der Kooperation bestand durch die Mitarbeit in den **Querschnittprojekten**, in denen alle Teilprojekte Mitarbeiterzeit zur Verfügung gestellt haben. Die Ergebnisse dieser Teams sind in eigenen Berichten dargestellt.

5 Eigene Veröffentlichungen im Berichtszeitraum

Projektarbeiten

[1] Schwaiger, M.: *Musterbasierte Konfigurierung des Internet-Protokolls IPv6 mit SDL*, Dezember 1997

[2] van Lier, S.: *SDL-Pattern basiertes Reengineering von RTP (Realtime Transport Protocol)*, August 1998

Diplomarbeiten

[3] Cisowski, D.: *Ein Editor zur Dokumentation von SDL Patterns*, März 1999

[4] Groß, T.: *Re-Engineering und Implementierung eines ATM-Signalisierungsprotokolls mit SDL-Patterns*, September 1999

[5] Kühlmeyer, A.: *SDT-basierte Anbindung von SDL-Entwurfsspezifikationen an verschiedene Basistechnologien*, März 1999

[6] Schaible, P.: *Pattern-basierte Konfigurierung des Reservierungsprotokolls ST2+ und Erstellung eines SDL-Frameworks für Ressourcenreservierungsprotokolle*, September 1997

[7] Schneider, M.: *Entwurf und Implementierung eines echtzeitfähigen Kommunikationssubsystems für CAN*, Februar 1998

[8] Schwaiger, M.: *Werkzeugunterstützte Anwendung von SDL Patterns*, März 1999

SFB-Berichte

[9] Feldmann, R., Geppert, B., Mahnke, W., Ritter, N., Rößler, F.: *The ORDBMS-Based SFB 501 Experience Base - Exemplified by the SDL-Pattern Approach*, SFB 501 Report 08/99, Fachbereich Informatik, Universität Kaiserslautern, 1999

[10] Feldmann, R., Geppert, B., Rößler, F.: *Towards an Experimental Evaluation of SDL-Pattern based Protocol Design*, SFB 501 Report 04/98, Fachbereich Informatik, Universität Kaiserslautern, 1998

[11] Feldmann, R., Geppert, B., Rößler, F.: *A Repository-Based Improvement Cycle for Reuse-Driven SDL System Development*, SFB 501 Report 11/98, Fachbereich Informatik, Universität Kaiserslautern, 1998

[12] Feldmann, R., Geppert, B., Rößler, F.: *First Results from an Experimental Evaluation of SDL-Pattern Based Protocol Design*, SFB 501 Report 03/99, Fachbereich Informatik, Universität Kaiserslautern, 1999

[13] Geppert, B., Rößler, F., Schaible, P.: *Pattern-based Configuring of the Internet Stream Protocol ST2+ with SDL*, SFB 501 Report 12/97, Fachbereich Informatik, Universität Kaiserslautern, 1997

[14] Glässer, U., Gotzhein, R., Prinz, A.: *SDL Formal Semantics Definition*, Technischer Bericht tr-rsfb-99-065, Fachbereich Mathematik und Informatik, Universität Paderborn, 1999

[15] Gotzhein, R., Kronenburg, M., Peper, C.: *Reuse in Requirements Engineering: Discovery and Application of a Real-Time Requirement Pattern*, SFB 501 Report 8/98, Fachbereich Informatik, Universität Kaiserslautern, 1998

[16] Kronenburg, M., Peper, C.: *An Example of a FoReST Problem Specification*, SFB 501 Report 01/1999, Fachbereich Informatik, Universität Kaiserslautern, 1999

[17] Petersen, E., Münch, J., Geppert, B.: *Ein formales Prozeßmodell für die Software-Entwicklungsmethode SOMT*, SFB 501 Report 03/98, Fachbereich Informatik, Universität Kaiserslautern, 1998

Online-Dokumente

[18] Kronenburg, M., Peper, C.: *Specification of the Case Study "Light Control System"*, Dagstuhl-Seminar Requirements Capture, Documentation, and Validation, 13.-18. Juni 99, http://rn.informatik.uni-kl.de/~forest/examples

Vorträge

[19] Geppert, B.: *SDL Patterns - Ein Ansatz zur generischen Protokollentwicklung,* Forschungsseminar, Humboldt Universität Berlin, 17.2.1998

[20] Geppert, B.: *Der SDL-Pattern-Ansatz*, Workshop Software-Reuse des SFB 501, Kaiserslautern, 22.-23. Februar 1999

[21] Gotzhein, R.: *Formale Spezifikation von Echtzeitanforderungen*, GI-Fachgespräch über Quality of Service, Mannheim, 12.-13.3.1998

[22] Gotzhein, R., Geppert, B., Rößler, F.: *Musterbasierte Entwicklung von Kommunikationsprotokollen mit SDL*, RIO6-Kolloquium, Daimler-Benz Aerospace AG, Bremen, 16.6.1998

[23] Gotzhein, R.: *Engineering-like Development of Customized Communication Software*, Kolloquium, Universität Zürich, 25.6.1998

[24] Gotzhein, R.: *Musterbasierte Systementwicklung*, Daimler-Benz Research, Berlin, 2.7.98

[25] Gotzhein, R.: *Pattern-based System Development*, Kolloquium, Ecole Polytechnique Federale, Lausanne, 9.7.98

[26] Gotzhein, R.: *Wiederverwendung von Lösungskomponenten und Entwicklungs-Know-how*, Deutsche Telekom, Saarbrücken, 16.12.1998

[27] Gotzhein, R.: *Musterbasierte Entwicklung von Kommunikationsmiddleware*, Kolloquium, Fachbereich Informatik, Universität Braunschweig, 12. Juli 1999

[28] Peper, C.: *Wiederverwendung bei der Anforderungsspezifikation: Der FoReST-Ansatz*, Workshop "Software-Reuse" des SFB 501, Kaiserslautern, 22.-23. Februar 1999

Tagungsbeiträge

[29] Cisowski, D. , Geppert, B., Rößler, F., Schwaiger, M.: *Tool Support for SDL Patterns*, Proceedings of the 1st Workshop of the SDL Forum Society on SDL and MSC (SAM98), Berlin, Juni 1998

[30] Feldmann, R., Geppert, B., Mahnke, W., Ritter, N., Rößler, F.: *An ORDBMS-Based Reuse Repository Supporting the Quality Improvement Paradigm - Exemplified by the SDL-Pattern Approach*, in: Proceedings of the TOOLS USA 2000, 34th International Conference & Exhibition, Santa Barbara, USA, July/August 2000

[31] Feldmann, R.L, Geppert, B., Rößler, F.: *Continuous Improvement of Reuse-Driven SDL System Development*, Proceedings of the 11th International Conference on Software Engineering and Knowledge Engineering (SEKE'99), Kaiserslautern, Juni 1999

[32] Feldmann, R.L, Geppert, B., Rößler, F.: *An Integrating Approach for Developing Distributed Software Systems - Combining Formal Methods, Software Reuse, and the Experience Base*, Proceedings of the 5th International Conference on Engineering of Complex Computer Systems (ICECCS'99), Las Vegas, USA, Oktober 1999

[33] Geppert, B., Gotzhein, R., Rößler, F.: *Configuring Communication Protocols Using SDL Patterns*, in: A. Cavalli, A. Sarma (Eds.), SDL'97 - Time for Testing, Elsevier Science Publishers, Proceedings of the 8th SDL-Forum, Evry, France, September 1997

[34] Geppert, B., Kühlmeyer, A., Rößler, F., Schneider, M.: *SDL-Pattern based Development of a Communication Subsystem for CAN*, Proceedings of the IFIP Joint International Conference on Formal Description Techniques & Protocol Specification, Testing, and Verification (FORTE/PSTV'98), Paris, Frankreich, Kluwer Academic Publishers, November 1998

[35] Geppert, B., Rößler, F.: *Combining SDL and Pattern-Based Design for the Customization of Communication Subsystems*, in: A. Wolisz, I. Schieferdekker, A. Rennoch (Hrsg.), Formale Beschreibungstechniken für verteilte

Systeme, GMD-Studien Nr. 315, GI/ITG-Fachgespräch, Berlin, 19. -20. Juni 1997

[36] Geppert, B., Rößler, F.: *Generic Engineering of Communication Protocols - Current Experience and Future Issues*. Proceedings of the 1st IEEE International Conference on Formal Engineering Methods (ICFEM'97), Hiroshima, Japan, November 1997

[37] Geppert, B., Rößler: *Pattern-Based Reuse with SDL*, Tutorial Notes, 9th SDL Forum (SDL'99), Montreal, Kanada, 21.-25. Juni 1999

[38] Geppert, B., Rößler, F., Feldmann, R.L.: *Systematic Evolution of Reusable SDL Patterns*, Proceedings of the 2nd European Reuse Workshop (ERW'98), Madrid, Spanien, 1998

[39] Geppert, B., Rößler, F., Feldmann, R.L., Vorwieger, S.: *Combining SDL Patterns with Continuous Quality Improvement: An Experience Factory Tailored to SDL Patterns*, Proceedings of the 1st Workshop of the SDL Forum Society on SDL and MSC (SAM98), Berlin, Juni 1998

[40] Geppert, B., Rößler, F., Gotzhein, R.: *Pattern Application vs. Inheritance in SDL*, 3rd IFIP International Conference on Formal Methods for Open Object-Based Distributed Systems (FMOODS'99), Florenz, Italien, Kluwer Academic Publishers, Februar 1999

[41] Geppert, B., Rößler, F., Schneider, M.: *Using SDL Patterns for the Design of a CAN-based Communication Subsystem*, Proceedings of the 8th GI/ITG-Fachgespräch Formale Beschreibungstechniken für verteilte Systeme (FBT'98), Cottbus, 1998

[42] Glässer, U., Gotzhein, R., Prinz, A.: *Towards a New Formal SDL Semantics Based on Abstract State Machines*, in: R. Dssouli, G.v. Bochmann, Y. Lahav (eds.), SDL'99 - The Next Millenium, Proc. of the 9th SDL FORUM, Elsevier Science B.V., Juli 1999

[43] Gotzhein, R., Geppert, B., Rößler, F., Schaible, P.: *Towards a New Formal SDL Semantics*, Workshop of the SDL Forum Society on SDL and MSC (SAM98), Berlin, 29.6.-1.7.98

[44] Gotzhein, R., Kronenburg, M., Peper, C.: *Reuse in Requirements Engineering: Discovery and Application of a Real-Time Requirement Pattern*, 5th International Symposium on Formal Techniques in Real-Time and Fault-Tolerant Systems (FTRTFT'98), Lecture Notes in Computer Science 1486, Springer, Lyngby, Denmark, September 14-15, 1998. pp. 65-74

[45] Gotzhein, R., Kronenburg, M., Peper, C.: *Pattern-Based Requirements Capture Applied: The SFB 501 Case Study*, in: E. Börger, B. Hörger, D. Parnas, D.

Rombach (Hrsg.), Requirements Capture, Documentation, and Validation, Dagstuhl-Seminar-Report 242, Juni 1999

[46] Kronenburg, M., Peper, C.: *Definition and Instantiation of a Reference Model for Problem Specifications*, 11th International Conference on Software Engineering and Knowledge Engineering (SEKE'99), Kaiserslautern, 1999

[47] Peper, C.: *Transformations in Pattern-Based System Specifications*, 9. GI/ITG-Fachgespräch Formale Beschreibungstechniken für verteilte Systems (FBT'99), München, Herbert-Utz-Verlag, 1999

[48] Peper, C., Gotzhein, R., Kronenburg, M.: *A Generic Approach to the Formal Specification of Requirements*. Proceedings of the 1st IEEE International Conference on Formal Engineering Methods (ICFEM'97), Hiroshima, Japan, November 1997, pp. 252-261

[49] Rößler, F., Geppert, B.: *Applying Quality of Service Architectures to the Field-Bus Domain*, Proceedings of the 2nd IEEE International Workshop on Factory Communication Systems (WFCS'97), Barcelona, Spain, 1997

[50] Rößler, F., Geppert, B., P. Schaible, P.: *Re-Engineering of the Internet Stream Protocol ST2+ with Formalized Design Patterns*, Proceedings of the 5th International Conference on Software Reuse (ICSR5), Victoria, British Columbia, Canada, 1998

[51] Rößler, F., Kühlmeyer, A.: *Implementing Real-Time Communication on a Token-Ring Network*. Proceedings of the 6th Open Workshop on High Speed Networks (HSN'97), Stuttgart, October, 1997

[52] Schaible, P., Gotzhein, R.: *Einige Aspekte der Entwicklung maßgeschneiderter Kommunikationsmiddleware*, GI/ITG-Fachgespräch "Formale Beschreibungstechniken für verteilte Systeme (FBT'2000)", 22.-23.6.2000, Lübeck

Zeitschriften / Monografien

[53] Avenhaus, J., Gotzhein, R., Härder, T., Litz, L., Madlener, K., Nehmer, J., Richter, M., Ritter, N., Rombach, D., Schürmann, B., Zimmermann, G.: *Entwicklung großer Systeme mit generischen Methoden - Eine Übersicht über den Sonderforchungsbereich 501*, Informatik, Forschung und Entwicklung, 13(4):227–234, Dezember 1998

[54] Börger, E., Gotzhein, R. (Guest Editors): *Requirements Engineering: The Light Control Case Study*, Special Issue of the Journal of Universal Computer Science, Springer, to appear 2000

[55] Gotzhein, R., Bredereke, J. (Guest Editors): *Trends in Formal Description Techniques*, Theme Issue of Computer Networks and ISDN Systems, Vol. 30 (1998), Numbers 9-10, May 1998

[56] Gotzhein, R., Schaible, P.: *Pattern-Based Development of Communication Subsystems*, in: Annals of Telecommunications, Special Issue on Protocol Engineering, Vol. 54, No. 11-12, 1999, pp. 508-525

[57] Kronenburg, M., Peper, C.: *Application of the FoReST Approach to the Light Control Case Study*, Journal of Universal Computer Science (J.UCS), Special Issue on “Requirements Engineering: The Light Control Case Study”, Springer, 2000 (accepted for publication)

Industriemessen und Demonstrationen

[58] Gotzhein, R., Geppert, B., Peper, C., Rößler, F., Schaible, P.: *Ingenieurmäßige Entwicklung maßgeschneiderter Kommunikationssoftware*, CeBIT 98, Hannover, 19.-25.3.1998

[59] Gotzhein, R., Geppert, B., Peper, C., Rößler, F., Schaible, P.: *Maßschneiderung echtzeitfähiger Kommunikationssoftware*, CeBIT 98, Hannover, 19-25.3.1998

[60] Gotzhein, R., Geppert, B., Rößler, F.: *Advanced Software Reuse through Formal Methods*, Technologietransferkongress “IKT-Anwendungen auf Netzen” im Rahmen des III. Mannheimer Unternehmerforums, 6./7. Oktober 1997

[61] Feldmann, R. L., Vorwieger, S., Geppert, B., Rößler, F.: *An Experience Base Tailored to SDL Patterns* (Tool Demonstration), 1st Workshop of the SDL Forum Society on SDL and MSC (SAM98), Berlin, 29.6.- 1.7.1998

Programmkomitees mit besonderem SFB-Bezug

[62] Gotzhein, R.: Steering Committee Member des Internationalen Workshops “*SDL and MSC (SAM98)*” in Berlin, 29.6.-1.7.98; Organisation und Leitung der Sitzungen “Patterns and SDL”

6 Fremdveröffentlichungen

[63] V. R. Basili, G. Caldiera, D. Rombach: *The Experience Factory*, in: J. J. Marciniak (ed.), Encyclopedia of Software Engineering, Volume I, John Wiley & Sons, 1994, pp. 469-476

[64] T. Schmidt-Samoa: *FoReST - Entwurf und Implementierung einer Umgebung zur Erstellung formaler Systemanforderungsbeschreibungen*, Diplomarbeit, Fachbereich Informatik, Universität Kaiserslautern, 1999

[65] P. Zave, M. Jackson: *Four Dark Corner of Requirements Engineering*, ACM Transactions on Software Engineering and Methodology 6(1), 1997, pp. 1-30

Teilprojekt B5

GeneSys - Generische Systemsoftware

Leiter

Prof. Dr. Jürgen Nehmer
geb. 13.2.1942
Universität Kaiserslautern
Postfach 3049
67653 Kaiserslautern
Tel.: 0631/205-4020
Fax: 0631/205-3558
nehmer@informatik.uni-kl.de

Mitarbeiter

Ergänzungsausstattung:

Dipl. Inform. L. Baum
Dipl. Inform. M. Becker

Grundausstattung:

M. Sc. V. Tamara

1 Kurzfassung der Aufgabenstellung

Betriebssysteme stellen eine unverzichtbare Plattform für die Laufzeitunterstützung großer Anwendungssysteme dar. Gerade im Bereich der eingebetteten Systeme mit ihren stark divergierenden Anforderungen an Umfang und Qualität der benötigten Betriebssystemdienste besteht naturgemäß ein großer Bedarf an angepassten und hoch optimierten Lösungen. Heute verfügbare Ansätze zur Betriebssystemkonfiguration unterstützen eine derartige Maßschneiderung nur unzureichend.

Das Teilprojekt hat sich die Aufgabe gestellt, Methoden, Techniken und Werkzeuge zu entwickeln, um die Herstellung eines für eine bestimmte Anwendung maßgeschneiderten Betriebssystems in bisher nicht gekanntem Ausmaß zu unterstützen. Im Zentrum der Untersuchungen stehen hierbei nicht die Entwicklung grundlegend neuer Betriebssystemstrukturen, sondern die generische Auslegung komponentenbasierter Betriebssysteme und die werkzeuggestützte Anpassung dieser Systeme nach Anwendungsanforderungen, wobei neben funktionalen Eigenschaften insbesondere auch nichtfunktionale Anforderungen an das Betriebssystem zu berücksichtigen sind. Der wissenschaftliche Ansatz basiert auf einer Rahmenarchitektur für Betriebssysteme und einer darauf abgestimmten Komponententechnologie, die eine Wiederverwendung von Komponenten auf Kodeebene unterstützt und die flexible Maßschneiderung der Komponenten über generische Parameter unter weitgehender Einbeziehung von Generatortechniken ermöglicht. Darauf aufsetzend erlauben Verfahren zur Kodierung und Wiederverwendung von Expertenwissen die Steuerung des Entwicklungsvorgangs durch Anwendungsvorgaben.

2 Entwicklung des Teilprojekts in der Förderperiode

Aufgrund der Annahme eines Rufes an die Universität Trier kurz vor Beginn der Förderungsperiode schied Herr Dr. Sturm aus der Leitung des Teilprojekts aus.

3 Durchgeführte Arbeiten und Ergebnisse

3.1 Arbeitsschwerpunkt A: Basistechnologien für generische Systemsoftware

Im Rahmen des Arbeitsschwerpunktes A wurden in der vergangenen Förderungsperiode die zur Komposition maßschneiderbarer Laufzeitplattformen erforderlichen Basistechnologien untersucht. Gerade im Bereich der Gebäudeautomation besteht ein großer Maßschneiderungsbedarf aufgrund der unterschiedlichen Anforderungen an die Teilsysteme und des hohen Grades an Ressourcenausnutzung, der auch feingranulare Änderungen auf der Kodeebene erforderlich macht. Zur Steigerung des Wiederverwendungsgrades wurde das Konzept der generischen Komponenten angewandt. Hierbei handelt es sich um Softwarebausteine, deren Eigenschaften in begrenztem Maße über sogenannte generische Parameter in Kombination mit entsprechenden

Generatortechniken angepasst werden können, ohne notwendigerweise manuelle Änderungen am Komponentenkode vornehmen zu müssen. Die im Rahmen dieses Arbeitsschwerpunktes durchgeführten Arbeiten lassen sich grob zwei Aufgabenbereichen zuordnen: der Entwicklung der erforderlichen Basistechnologien zur Realisierung generischer Komponenten und dem Einsatz dieser Technologien bei der Fertigung von maßschneiderbaren Laufzeitplattform-Komponenten.

Basistechnologien generischer Komponenten

Die in der vergangenen Förderungsperiode geleisteten Entwicklungsarbeiten basieren auf dem Konzept der generischen Komponenten [22]. Herkömmliche, komponentenbasierte Softwareentwicklung ist im Umfeld stringenter Ressourcenanforderungen meistens nicht anwendbar, da die Komponenten entweder zu speziell sind, um sie sinnvoll wiederzuverwenden, oder aber wegen ihrer Allgemeinheit zu viele Ressourcen verbrauchen. Mit generischen Komponeten wird diesem Mißstand begegnet, indem über generische Parameter eine Maßschneiderung zentraler Komponenteneigenschaften ermöglicht wird. Hierbei können drei verschiedene Parametertypen identifiziert werden: Selektions-, Generierungs- und Kodeparameter. Selektionsparameter wählen aus einer Menge von vorgefertigten Varianten aus. Über Generierungsparameter werden entsprechende Generatoren gesteuert, welche abhängig von den aktuellen Parameterbelegungen Varianten erzeugen und diese in vorgesehene Lücken einfügen. Schließlich geben Kodeparameter überall dort, wo hochgradig individuelle Anpassungen erforderlich sind, Hilfestellungen beim manuellen Ausfüllen von belassenen Lücken.

Bei der internen Umsetzung von generischen Parametern und der Bereitstellung von zugehörigen Generatoren können verschiedene Techniken eingesetzt werden. In einem ersten Ansatz wurden hierfür der C-Präprozessor und geeignete programmiersprachliche Konzepte, wie z.B. Objekt-Orientierung oder Templates, verwendet [6]. Hiermit lassen sich Selektions- und teilweise Generierungsparameter recht effektiv realisieren. Zur weitergehenden Unterstützung von Generierungsparametern wurde anhand des sed-Editors [31] untersucht, wie Standardwerkzeuge zur Erzeugung von speziellen Generatoren genutzt werden können [22]. Über entsprechende Markierungen im Kode war es möglich, die notwendigen Transformationen durch sed-Skripte zu steuern und somit relativ einfach mächtige Generatoren zu erstellen. Während sich die Unterstützung für Kodeparameter bislang nur auf entsprechend kommentierte Lücken beschränkte, gewinnt man auch hier mit dem im Rahmen des Teilprojektes entwickelten Ansatz der Java-Metaprogrammierung einen hohen Grad an Flexibilität. Komponentenbeschreibungen können dadurch zu aktiven Dokumenten gemacht werden, die selbständig die erforderlichen Generatoren aufrufen und vielfältige Unterstützung beim Ausfüllen von belassenen Lücken leisten. Ein ähnlich flexibler, jedoch mit geringerem Aufwand verbundener Ansatz beruht auf dem Einsatz der weitverbreiteten XML-Beschreibungssprache [26] und verwandter Transformationstechniken [1]. Wie beim sed-Ansatz werden hierbei Markierungen in den Komponentenbeschreibungen eingefügt, die externe, frei gestaltbare Generatoren steuern. Die ersten Erfahrungen mit diesem vielversprechenden Ansatz legen weiterführende systematische Untersuchungen im Rahmen der kommenden Förderungsperiode nahe.

Fertigung maßschneiderbarer Laufzeitplattform-Komponenten

Die Fertigung eigener maßschneiderbarer Laufzeitplattform-Komponenten erfolgte im Hinblick auf die Evaluierung der entwickelten Basistechnologien zur Unterstützung feingranularer Maßschneiderung. Um den Realisierungsaufwand für Laufzeitplattform-Komponenten gering zu halten, wurde in einem ersten Ansatz versucht, frei verfügbare Mikrokerne als Grundlage für höhere Systemdienste einzusetzen. In [5] wurde eine Vielzahl verschiedener, frei verfügbarer Mikrokerne wie L4 [28], RTEMS [30] und RTX [29] hinsichtlich ihrer Architektur, Funktionalität und Tauglichkeit für die Arbeiten im Kontext des B5-Teilprojektes untersucht. Aufgrund fehlender Unterstützung von Echtzeitfähigkeit, übergroßen Funktionsumfangs und beschränkter Hardwareunterstützung erwiesen sich diese Kerne allerdings als ungeeignet für die angestrebte Bereitstellung maßschneiderbarer Laufzeitplattformen. Erschwerend kommt hinzu, dass ein Betriebssystemkern wesentlich die Architektur der gesamten Laufzeitplattform mitbestimmt und insbesondere auch im Kern erhebliches Maßschneiderungspotential vorhanden ist, das bei diesem Ansatz nur schwer zu nutzen wäre.

Der daraufhin durchgeführten Entwicklung eigener generischen Komponenten ging eine umfangreiche Untersuchung von Varianten elementarer Betriebssystemkonzepte und deren Abhängigkeiten untereinander voran. Im Rahmen einer simulativen Laufzeitplattform SimKernel [6] wurden hierfür 15 variable Aspekte betrachtet, wie z.B. verschiedene Varianten von Schedulingverfahren, Synchronisations- und Kommunikationsmechanismen, Speicher- und Threadmanagementstrategien. Die dabei gesammelten Erfahrungen über die Elementarvarianten und deren Abhängigkeiten wurden mit Hilfe der Design-Space-Technik [27, 24] modelliert, um später den Einsatz und die Weiterentwicklung der Varianten unterstützen zu können. Aufbauend auf den so gewonnenen Erkenntnissen wurden in [8] Elementarvarianten für die auf der Intel x86-Architektur ausführbare RTK-Laufzeitplattform erstellt, die im Wesentlichen das Scheduling und die Synchronisation von Threads übernehmen. RTK wurde dabei speziell für den Einsatz in tief eingebetteten Systemen konzipiert und bietet eine für solche Systeme hinreichende Funktionalität. Zur Quantifizierung der Laufzeit- und Speichereffizienz der einzelnen Varianten wurden verschiedene Ausprägungen der RTK-Plattform erstellt und analysiert. Die hieraus gewonnenen Resultate lieferten eine wichtige Entscheidungsgrundlage für die Maßschneiderung der RTK-Plattform.

Basierend auf den gesammelten Erfahrungen und einer gezielten Analyse der gemeinsamen und unterschiedlichen Bestandteile der untersuchten Elementarvarianten wurden nachfolgend generische Komponenten gefertigt. Hierbei kamen die entwickelten Basistechnologien bei der Realisierung der erforderlichen Generizität zum Einsatz und konnten auf ihre Tauglichkeit hin untersucht und weiterentwickelt werden. Als Resultat dieser Integrationsarbeiten entstand unter anderem die gc-RTK-Plattform, dem vollständig auf generischen Komponenten basierenden Pendant zu RTK, welche dem SFB für die Entwicklung tief eingebetteter Systeme zur Verfügung gestellt wird. Diese Plattform nimmt aber auch eine wichtige Stellung innerhalb des Teilprojekt selbst ein, da sie die Grundlage für Analysen von Basistechnologien bildet und künftig zum Gegenstand der anwendungsgestützten Konfiguration werden soll.

3.2 Arbeitsschwerpunkt B: Anwendungsgesteuerte Komposition maßgeschneiderter Systemsoftware

Ziel des Arbeitsschwerpunktes B war es, aufbauend auf dem Konzept generischer Softwarekomponenten Verfahren zur anwendungsgesteuerten Komposition maßgeschneiderter Systemsoftware zu entwickeln. Mit Hilfe geeigneter Werkzeuge sollte es Anwendungsentwicklern mit grundlegenden Betriebssystemkenntnissen ermöglicht werden, eine für ihre Anwendung maßgeschneiderte Laufzeitplattform herzustellen. Die Arbeiten in diesem Schwerpunkt konzentrierten sich daher hauptsächlich auf die Systematisierung sowie die methodische und werkzeugtechnische Unterstützung von drei Aufgabenbereichen: der Erfassung von Anwendungsanforderungen an Laufzeitplattformen, der Abbildung von Anforderungen auf geeignete generische Komponenten und der werkzeugunterstützten Konfiguration und Komposition dieser Komponenten zu einer maßgeschneiderten Laufzeitplattform.

Anforderungserfassung

Die Entwicklung einer maßgeschneiderten Laufzeitplattform bedingt die Abstimmung der funktionalen und nichtfunktionalen Eigenschaften der Plattform auf die individuellen Bedürfnisse einer konkreten Anwendung. Da jedoch nicht nur die Anwendung die erforderlichen Eigenschaften der Laufzeitplattform bestimmt, sondern umgekehrt die Eigenschaften der Laufzeitplattform auch zwangsläufig den Entwurf und die Implementierung der Anwendung beeinflussen, ist es erforderlich, Anwendungsentwicklung und Plattformentwicklung zu verzahnen und bereits sehr früh mit der Erfassung von Anwendungsanforderungen zu beginnen. Unter dieser Prämisse wurden zwei sich ergänzende Ansätze entwickelt, die es erlauben, Anforderungen an die Laufzeitplattform bereits während des Anwendungsentwurfs zu berücksichtigen und bis zu einem gewissen Grad automatisch zu erfassen.

Die in Anwendungsentwürfen enthaltenen Informationen implizieren in aller Regel bereits eine Reihe von Anforderungen an die jeweilige Laufzeitunterstützung. Die Anzahl modellierter aktiver Komponenten sowie deren Interaktionsbeziehungen geben beispielsweise erste Hinweise auf eine erforderliche Thread-Unterstützung und die benötigten Kommunikationsmechanismen; Eine Abschätzung des Modellierungsumfangs für Teilsysteme und Komponenten läßt darüber hinaus unter Umständen erste Rückschlüsse auf Rechenzeit- und Speicherplatzbedarf zu. Ausgehend von der für verteilte Kontrollsysteme naheliegenden Spezifikationssprache SDL [25] wurde daher in [4] ein Analysesystem entwickelt, das eine Reihe relevanter Anwendungseigenschaften automatisch aus konventionellen SDL-Modellen extrahiert. Durch Parsen der textuellen SDL-PR-Repräsentation eines Anwendungsentwurfs können somit automatisch Informationen beispielsweise über die Anzahl statisch und dynamisch anzulegender Threads, die Anzahl von Prozeduraufrufen, die Verwendung von Shared

Memory und die statische und dynamische Speicherkomplexität von Prozessen abgeleitet werden.

In [3] wurde darüber hinaus untersucht, inwieweit mit Hilfe gezielter Überspezifikation weitergehende Anforderungen an die Laufzeitplattform explizit in SDL-Anwendungsmodellen berücksichtigt werden können. Ziel war hierbei eine einheitliche Beschreibung von Entwurfsentscheidungen, beispielsweise bezüglich der Nutzung bestimmter Plattformdienste oder der Wahl von Interprozess-Kommunikationsmechanismen und Schedulingstrategien, mit existierenden SDL-Sprachmitteln. Das zu diesem Zweck entwickelte Konzept basiert auf einer Kombination von SDL-Mustern und vordefinierten Block- und Signal-Bezeichnern. Unterschiedliche Interaktionsmechanismen wie Pipes, FIFOs, Message Queues oder auch Shared Memory können so durch Präfixe vor Signalnamen spezifiziert werden, während die Nutzung von Systemsoftwarediensten auf wohldefinierte Signale an die Laufzeitplattform repräsentierende Proxy-Blöcke abgebildet wird. Um die auf diese Weise explizit modellierten Anforderungen bereits für die Prototypgenerierung nutzen zu können, wurde für die SDL-Entwicklungsumgebung SDT unter Solaris eine Laufzeitbibliothek entwickelt, welche die genannten Spezifikationselemente im generierten Kode erkennt und automatisch auf die gewünschten Laufzeitplattformmechanismen umsetzt.

Abbildung von Anforderungen auf Lösungen

Um eine Menge von Anforderungen möglichst einfach auf existierende Lösungsstrukturen und wiederverwendbare generische Komponenten abbilden zu können, ist es sinnvoll, bereits die Erfassung der Anforderungen auf die verfügbaren Lösungen auszurichten [18]. Mit dem Konzept der Erweiterten Design Spaces wurde hierzu eine Technik weiterentwickelt [27, 24] und eingesetzt [7], die eine semiformale Beschreibung sowohl von Anforderungen als auch von Eigenschaften beliebiger Softwareartefakte erlaubt. Design Spaces werden von einer Menge von Dimensionen aufgespannt, welche die relevanten Charakteristika des jeweils betrachteten Artefakts repräsentieren. Die möglichen Ausprägungen bezüglich dieser Charakteristika machen die sogenannten Kategorien der jeweiligen Dimensionen aus. Während ein Design Space somit einer Art Klassifikationsschema entspricht, beschreibt eine Auswahl einzelner Kategorien in einem Design Space entweder erwünschte Eigenschaften eines gesuchten Artefakts oder tatsächlich erbrachte Leistungen eines konkreten Produkts. Zur Erfassung von Anforderungen an die in Arbeitsschwerpunkt A entwickelten bzw. eingesetzten Laufzeitplattformen wurden geeignete Anforderungs-Design-Spaces aufgestellt, welche in Form von Fragebögen die Mengen möglicher Anforderungen abdecken, die sinnvoll mit der jeweiligen Plattform erfüllt werden können. Um den Umgang mit diesen Anforderungs-Design-Spaces zu erleichtern, wurden zwei Werkzeuge entwickelt, die die Erstellung und Manipulation eines Design Spaces bzw. das Erstellen von Design-Space-Profilen [7] unterstützen.

Das Design-Space-Konzept sieht über die reinen Klassifikationsmechanismen hinaus auch die Möglichkeit vor, Abhängigkeiten zwischen Kategorien und damit zwischen einzelnen Eigenschaften betrachteter Artefakte durch Angabe gewichteter Korrelationen zu dokumentieren. So lassen sich beispielsweise automatisch überprüfbare Konsistenzbedingungen für Anforderungen oder Abhängigkeiten zwischen Konfigurationsparametern definieren. Insbesondere können auch Korrelationen zwischen Anforderungs-Design-Spaces und solchen Design-Spaces spezifiziert werden, die wiederverwendbare generische Komponenten beschreiben. Auf diese Weise kann Expertenwissen über sinnvolle Zusammenhänge zwischen bestimmten Anforderungen und passenden Lösungen in Form geeignet ausgewählter und konfigurierter Komponenten explizit festgehalten und einer werkzeuggestützten Verarbeitung zugänglich gemacht werden.

Konfigurationswerkzeuge

Zur systematischen Umsetzung dieser Möglichkeiten in einen teilautomatischen Auswahl- und Konfigurationsvorgang wurde zunächst ein umfassendes Konzept [19] basierend auf einer Menge über Korrelationen verknüpfter Design Spaces erarbeitet, welches unter anderem auch ein entsprechendes Vorgehensmodell umfaßt. Demgemäß nimmt der Anwendungsentwickler zunächst eine Charakterisierung seiner Anwendung anhand eines Anwendungs-Design-Spaces vor. Da dieser Design-Space im Vokabular der Anwendungsentwicklung gehalten ist, sind hierzu keine speziellen Betriebssystemkenntnisse erforderlich. Auf Erfahrungswerten beruhende Korrelationen bilden diese Charakterisierung der Anwendung dann auf ein Anforderungsprofil in einem Anforderungs-Design-Space ab, welches ggf. an den Stellen, an denen noch keine Erfahrungswerte bezüglich geeigneter Abbildungen vorliegen, vom Anwendungsentwickler zu ergänzen ist. Ausgehend von diesem Anforderungprofil bilden Korrelationen die Anforderungen weiter auf eine geeignete Plattformarchitektur, und anschließend auf entsprechende Anforderungen an einzelne Komponenten der gewählten Plattformarchitektur ab. Diese Komponentenanforderungen ermöglichen schließlich eine teilautomatische Auswahl konkreter Komponentenimplementierungen und die Belegung der generischen Komponentenparameter. Eine erste Umsetzung dieses Konzepts in ein Unterstützungwerkzeug [7] wurde auf Basis des Betriebssystems QNX [32] vorgenommen. Im Zuge einer iterativen und interaktiven Eingabe von Anforderungen an die Laufzeitplattform QNX übernimmt besagtes Werkzeug automatisch die Auswertung aller Design-Space-Korrelationen und wählt — soweit per geeigneter Korrelationen abgedeckt — selbständig die erforderlichen QNX-Komponenten und deren Konfigurationsparameter.

3.3 Arbeitsschwerpunkt C: Experimente im Anwendungsfeld Gebäudeautomation

Ziel des Arbeitsschwerpunktes C war die kontrollierte Sammlung von Daten bezüglich der Vor- und Nachteile der in den Schwerpunkten A und B entwickelten Methoden und Werkzeuge. Dabei galt es, sowohl die Verwendbarkeit generischer Betriebssystemkomponenten für die effiziente Entwicklung maßgeschneiderter Laufzeitplattformen zu überprüfen, wie auch die Verfahren und Werkzeuge zur Unterstützung des Anwendungsentwicklers während der Maßschneiderung einer Laufzeitplattform zu validieren. Die Durchführung kontrollierter Experimente mit aussagekräftiger Vergleichsgrundlage erwies sich hierbei aufgrund einer unzureichenden Menge an Testentwicklern und mangelnder Vergleichbarkeit der Ausgangsbedingungen als unmöglich. Nichtsdestotrotz konnten im Rahmen von Fallstudien die Annahmen bezüglich einer Reduktion von Entwicklungsaufwand und Fehlerhäufigkeit sowie der Erfüllbarkeit nichtfunktionaler Anforderungen bestätigt werden.

Eine derartige Fallstudie [10] wurde in enger Zusammenarbeit mit dem Teilprojekt B4 durchgeführt, um die Auswirkungen des Einsatzes generischer Betriebssystemkomponenten wie auch generischer Kommunikationssysteme im Umfeld von Gebäudeautomationssystemen zu untersuchen. Auf Basis der generischen Komponenten der gc-RTK-Plattform wurde hierzu eine Laufzeitunterstützung für ein verteiltes, funktionell jedoch stark vereinfachtes Gebäudekontrollsystem entwickelt, wobei die erforderlichen Kommunikationsmechanismen aus dem Teilprojekt B4 integriert wurden. Anschließend wurde eine weitgehend äquivalente Systemunterstützung mit dem kommerziellen Betriebssystem QNX erstellt. Ein Vergleich der beiden Ansätze ergab einen ähnlichen Zeitaufwand für die Bereitstellung der Plattformen (zu erklären durch die manuelle Zusammenstellung der RTK-Komponenten ohne jegliche Werkzeugunterstützung) und deutliche Vorteile der RTK-Plattform gegenüber QNX bezüglich der erzielbaren Laufzeit- und insbesondere Speichereffizienz. Abgesehen davon zeigte diese Fallstudie auch erfolgreich die nahtlose Integration von Produkten der Teilprojekte B4 und B5.

Eine zweite Fragestellung betraf die mit Hilfe von Unterstützungswerkzeugen erzielbaren Effekte in Bezug auf Entwicklungsaufwand und Fehleranfälligkeit bei der Maßschneiderung von Laufzeitplattformen. Hierzu wurden mehrere QNX-Laufzeitplattformen für ausgewählte Anwendungsszenarien aus dem Bereich der Gebäudeautomation maßgeschneidert. Im Rahmen der Arbeiten zu dem Design-Space-basierten Konfigurationswerkzeug für QNX [7] wurden die Entwicklungen zum einen mit Hilfe des Konfigurationswerkzeuges, zum anderen manuell vorgenommen. Insbesondere die automatische Konsistenzprüfung, die Leitung des Anwenders durch die Konfiguration in einer bestimmten Reihenfolge und die automatische Vervollständigung der Konfiguration auf Basis von Heuristiken trugen letztlich zu einer deutlich verkürzten Entwicklungszeit und einer leicht verringerten Fehleranzahl im Falle der werkzeuggestützten Maßschneiderung bei.

4 Zusammenarbeit mit anderen Teilprojekten

Mit dem Teilprojekt B4 bestand ein enger Erfahrungsaustausch betreffend die Parametrisierung von Kommunikations- und Betriebssystemdiensten sowie die Gestaltung geeigneter Schnittstellen, der Einfluß auf die Erarbeitung der entsprechenden Techniken in den Arbeitsschwerpunkten A und B hatte. Der bereits in der ersten Förderungsperiode begonnene Abgleich der eingesetzten Techniken und Schnittstellen erwies sich hierbei als erfolgreich: so konnte im Rahmen der im Arbeitsschwerpunkt C durchgeführten Fallstudie die einfache Integrierbarkeit der Produkte aus den Teilprojekten B4 und B5 gezeigt werden.

Die Formalisierung von Komponentenabhängigkeiten und Heuristiken zur Abbildung von Anforderungen auf Lösungen wurde in Zusammenarbeit mit dem Teilprojekt C1 vorgenommen. Ferner wurden mit dem Teilprojekt C1 erste vielversprechende Ansätze für eine Formalisierung der Design-Space-Technik diskutiert, die eine weitergehende Unterstützung bei der automatischen Erkennung von Inkonsistenzen in Anforderungen und der Auswertung von Entwurfsregeln ermöglichen würde.

Bei dem Entwurf eines komplexen Gebäudeautomationssystems im Rahmen des Querschnittsprojekts Q2 übernahm das Teilprojekt beratende Funktionen in Hinblick auf die Wahl einer geeigneten Laufzeitplattform und die Nutzung angebotener Laufzeitplattformdienste.

5 Eigene Veröffentlichungen im Berichtszeitraum

Projektarbeiten

[1] M. Lösch: *Beschreibung generischer Komponenten mit XML*, AG Systemsoftware, Universität Kaiserslautern, 2000

[2] M. Näher: *Das generische Potenzial von LINUX*, AG Systemsoftware, Universität Kaiserslautern, 2000

[3] E. Becher: *Modellierung von Betriebssystemanforderungen in SDL mit SDT*, AG Systemsoftware, Universität Kaiserslautern, 1998

[4] T. See: *Erfassung von BS-Anforderungen aus SDL-Entwürfen*, AG Systemsoftware, Universität Kaiserslautern, 1998

[5] A. Schank: *Evaluierung von Mikrokernen als Basis der Entwicklung generischer Systemsoftware*, AG Systemsoftware, Universität Kaiserslautern, 1998

Diplomarbeiten

[6] A. Eckert: *Modellierung und werkzeuggestützte Verarbeitung von Parameterabhängigkeiten bei generischen Komponenten*; AG Systemsoftware, Universität Kaiserslautern, 1999

[7] A. Gilbert: *Design-Space-basierte Konfiguration des Betriebssystems QNX*; AG Systemsoftware, Universität Kaiserslautern, 1999

[8] M. Iannizzi: *Exemplarische Realisierung eines Scheduler-Subsystems für einen Realzeit-Betriebssystemkern als generische Komponente*, AG Systemsoftware, Universität Kaiserslautern, 1998

[9] T. Leyer: *Ein Konfigurationswerkzeug für generische Komponenten am Beispiel von QNX-Modulen*, AG Systemsoftware, Universität Kaiserslautern, 1998

SFB-Berichte

[10] L. Baum, M. Becker, L. Geyer, G. Molter, P. Schaible: *Integration generischer Ansätze der Kommunikationssystem- und Laufzeitplattformentwicklung*, SFB 501 Report 09/00, 2000

[11] T. Kramp, M. Adrian, and R. Koster: *An Open Framework for Real-time Scheduling Simulation*, SFB 501 Report 01/00, 2000

[12] T. Kramp, G. Coulson: *The Design of a Flexible Communications Framework for Next-Generation Middleware*, SFB 501 Report 12/99, 1999

[13] T. Kramp: *Free Jazz: An User-Level Real-Time Threads Package Designed for Flexibility*, SFB 501 Report 9/98, 1998

Vorträge

[14] M. Becker: *Generative Maßschneiderung von Betriebssystemen*, GI-Fachtagung, Fachgruppe Betriebssysteme, Erlangen, März 2000

[15] J. Nehmer: *Trends in der Betriebssystemforschung*, Technische Universität München, Januar 2000

[16] L. Baum: *GeneSys – Ein Überblick,* Heinz Nixdorf Institut, Universität Paderborn, Mai 1999

[17] L. Baum: *Generierung maßgeschneiderter Varianten aus vorgefertigten Betriebssystemkomponenten,* Reuse-Workshop des SFB 501, Februar 1999

Tagungsbeiträge

[18] L. Baum, M. Becker, L. Geyer, A. Gilbert, G. Molter, V. Tamara: *Supporting Component-Based Software Development Using Domain Knowledge*, Int'l Conference on Systemics, Cybernetics and Informatics (SCI), special event on Generative and Component-Based Software Engineering, Orlando, USA, Juli 2000

[19] L. Baum, M. Becker, L. Geyer, G. Molter: *Mapping Requirements to Reusable Components using Design Spaces*, Proc. of the IEEE Int'l Conference on Requirements Engineering (ICRE-2000), Schaumburg/Chicago, USA, Juni 2000

[20] J. Nehmer: *Software Reuse: Potential, Concepts, Risks*, Proc. of the IEEE Workshop on System Design Automation (SDA'2000), Rathen, März 2000

[21] L. Baum, T. Kramp: *Towards a Uniform Modeling Technique for Resource-Usage Scenarios*, Proc. of the 1999 Int'l Conf. on Parallel and Distributed Processing Techniques and Applications (PDPTA'99), Las Vegas, USA, Juni 1999

[22] L. Baum: *Towards Generating Customized Run-time Platforms from Generic Components*, Proc. of the 11th Conference on Advanced Information Systems Engineering (CAISE'99), 6th DC, Heidelberg, Juni 1999

[23] L. Baum, M. Becker, L. Geyer, G. Molter, P. Sturm: *Driving the Composition of Run-time Platforms by Architectural Knowledge*, Proc. of the 8th ACM SIGOPS European Workshop, Portugal, September 1998

[24] L. Baum, L. Geyer, G. Molter, S. Rothkugel, P. Sturm: *Architecture-centric Software Development Based on Extended Design Spaces*, Proc. of the 2nd ARES Workshop (Esprit 20477), Las Palmas de Gran Canaria, February 1998

6 Fremdveröffentlichungen

[25] R. Braek; O. Haugen: *Engineering Real Time Systems — An Object Oriented Methodology using SDL*, Prentice-Hall, 1993

[26] S. Holzner: *XML Complete*, McGraw-Hill, 1998

[27] T. G. Lane: *Studying Software Architecture Through Design Spaces and Rules*, Technical Report CMU/SEI-90-TR-18, Carnegie Mellon Univ., 1990

[28] J. Liedke: *On microkernel construction*, Proc. of the 15th ACM Symposium on Operating System Principles (SOSP), ACM Press, 1995

[29] Real Time Executives, Inc.: *The RTX Real-Time Kernel,* Real Time Executives Inc., http://www.rtx.com

[30] *Real Time Executive for Multiprocessor Systems* — Release 3.5.1 for Intel i386 C, ftp://lancelot.gcs.redstone/pub/rtems

[31] W. R. Stevens: *Advanced Programming in the UNIX Environment*, Addison-Wesley, 1992

[32] QNX Software Systems Ltd.: *QNX System Architecture*, System Documentation, QNX Software Systems Ltd., Canada, 1993

Teilprojekt B10

Anwendungsentwicklung mit vorkonfektionierten Systemen

Leiter

Prof. Dr. Jürgen Nehmer
geb. 13.2.1942
Universität Kaiserslautern
Postfach 3049
67653 Kaiserslautern
Tel.: 0631/205-4020
Fax: 0631/205-3558
nehmer@informatik.uni-kl.de

Mitarbeiter

Ergänzungsausstattung:

Dipl. Inform. L. Geyer

Grundausstattung:

Dipl. Inform. G. Molter

1 Kurzfassung der Aufgabenstellung

Das Teilprojekt B10 hat sich die Aufgabe gestellt, die Entwicklung von Anwendungssoftware durch eine weitreichende Wiederverwendung domänenspezifischer Softwareartefakte zu unterstützen. Die in der ersten Förderungsperiode (1998-2000) gesammelten Erfahrungen zeigen, dass in Anwendungsfeldern mit potenziell großem Lösungsraum entsprechende Konzepte schon in frühe Phasen der Softwareentwicklung integriert werden müssen, um systematische Wiederverwendung zu unterstützen. Im Teilprojekt B10 wird dazu der Ansatz verfolgt, geeignete Softwarearchitekturen als Träger für wiederverwendbares Domänenwissen sowie als Rahmen für alle Wiederverwendungsaktivitäten einzusetzen. Die Kombination einer solchen Architektur mit dazu kompatiblen wiederverwendbaren Produkten und einem speziell angepassten Entwicklungsprozess stellt ein vorkonfektioniertes System dar, das im Laufe der Anwendungsentwicklung ergänzt und auf die spezifische Aufgabenstellung zugeschnitten wird.

Im Gegensatz zu anderen produktorientierten Teilprojekten im SFB 501, die sich auf die Entwicklung von Laufzeitplattformen sowie auf die dabei auftretenden Interdependenzen mit der Anwendungsentwicklung konzentrieren (B4, B5), zielen die in B10 erarbeiteten Ansätze auf die Unterstützung der gesamten Anwendungsentwicklung ab. Dazu wird ein generischer, domänenunabhängiger Technologierahmen entwickelt, der die grundlegende Vorgehensweise sowie Techniken zur Anwendungsentwicklung mit vorkonfektionierten Systemen beschreibt.

Ein einzelnes vorkonfektioniertes System stellt eine Instanz dieses Rahmens für ein spezifisches Anwendungsfeld dar. Als exemplarisches Anwendungsfeld wird hierfür in B10 die Domäne Gebäudeautomation betrachtet.

Im Rahmen des Teilprojektes werden folgende wissenschaftliche Aufgaben bearbeitet:

a) Bereitstellung eines Technologierahmens für wiederverwendungsorientierte, architekturbasierte Softwareentwicklung

b) Erarbeitung eines Prozessmodells für architekturzentrierte Softwareentwicklung, das auf systematische Wiederverwendung unterschiedlicher Arten von Produkten ausgelegt ist (z.B. Modellfragmente und Softwarekomponenten)

c) Entwicklung von Mess- und Analysemethoden zur Überwachung der Anwendungsentwicklung unter Ausnutzung wiederverwendbaren Wissens

d) Unterstützung der Entwicklung vorkonfektionierter Systeme für noch nicht erschlossene Domänen, sowie der Wartung und Weiterentwicklung existierender vorkonfektionierter Systeme

In der ersten Förderungsperiode wurden insbesondere die Grundlagen zu a) und b) gelegt, sowie Untersuchungen bezüglich der mit vorkonfektionierten Systemen erreichbaren Wiederverwendbarkeit angestellt. Unter Anwendung und Erweiterung der erarbeiteten Techniken und Resultate soll in der zweiten Förderungsperiode die Softwareentwicklung mit vorkonfektionierten Systemen methodisch stärker unterstützt werden; weitere Schwerpunkte werden die Neuentwicklung vorkonfektionierter Systeme sowie die Rückkopplung von Verbesserungen und Erweiterungen sein.

2 Entwicklung des Teilprojekts in der Förderperiode

Vor Anlaufen der Förderung dieses Teilprojekts hat Dr. Sturm einen Ruf an die Universität Trier angenommen und ist von diesem Zeitpunkt an aus der Leitung des Teilprojekts ausgeschieden.

3 Durchgeführte Arbeiten und Ergebnisse

3.1 Arbeitsschwerpunkt A: Konzeption und Realisierung vorkonfektionierter Softwaresysteme

Im ersten Arbeitsschwerpunkt sind in der Berichtsperiode die Grundkonzepte für die Softwareentwicklung mit vorkonfektionierten Systemen untersucht worden. Dazu wurde zunächst ein Strukturierungskonzept entwickelt, das die offene Rahmenstruktur solcher Systeme umsetzt, die zum einen geeignete wiederverwendbare Artefakte enthalten, aber auch wohldefinierte Lücken für anwendungsspezifische Funktionalität. Das einzelne vorkonfektionierte System als Instanz dieses Rahmens unterstützt dabei die gezielte Vervollständigung der definierten Lücken, sowie einen nahtlosen Übergang von den Abstraktionen des jeweiligen Anwendungsfeldes zu den Elementen des vorkonfektionierten Systems.

Architektur

Als zentrales Konzept zum Erreichen dieser Ziele hat sich in der vergangenen Förderungsperiode der Begriff der Software-Architektur herausgestellt. Im Teilprojekt B10 wird dabei ein sichtenbasierter Ansatz zur Architekturbeschreibung zurundegelegt. Die Architektur eines Systems wird durch eine Reihe spezifischer Sichten und Strukturen repräsentiert, die jeweils bestimmte Aspekte des Systems herausprojizieren und so deren gezielte Untersuchung ermöglichen.

Wie in [13] beschrieben, ist Architektur gerade im Umfeld der Entwicklung komplexer Systeme eine hohe Bedeutung beizumessen. Zum einen beeinflusst die Struktur eines Systems in hohem Maße seine Eigenschaften, insbesondere nichtfunktionale Eigenschaften wie Skalierbarkeit und Fehlertoleranz; zum anderen stellt die Architektur als Abstraktion des entwickelten Systems ein geeignetes Mittel zur Berherrschung seiner Komplexität dar.

Desweiteren kommt Architektur eine Schlüsselrolle für jegliche Produkt-Wiederverwendung zu [8][13][14]. Architekturelle Kompatibilität der wiederverwendeten Artefakte mit den bereits erstellten Strukturen und Systembestandteilen ist eine notwendige Voraussetzung für ihre erfolgreiche Integration. Diese Kompatibilität zielt beispielsweise auf Aspekte wie die zugrundeliegenden Konzepte und Interaktionsmechanismen ab, aber auch auf die Auslegung des Artefakts im Hinblick auf bestimmte nichtfunktionale Eigenschaften. Architekturelles Wissen beschreibt damit einerseits Anforderungen an die zu verwendenden wiederverwendbaren Artefakte, die bei der Auswahl eines Produkts berücksichtigt werden müssen, andererseits wird diese Selektionsaufgabe dadurch aber vereinfacht, da die Menge potentiell passender Kandidaten für die Wiederverwendung verkleinert wird.

Vorkonfektionierte Systeme

Wegen ihrer hohen Bedeutung für die erfolgreiche Realisierung von Softwareprojekten stellen Architekturen ihrerseits wertvolles wiederverwendbares Wissen dar. Im Konzept vorkonfektionierter Systeme werden daher Architektur-Wiederverwendung, architekturbasierte Softwareentwicklung und Produkt-Wiederverwendung kombiniert.

Wie in [13] und [14] beschrieben, kann ein vorkonfektioniertes System als noch unvollständige Anwendung angesehen werden. Es beinhaltet eine explizite Beschreibung der zugrundeliegenden Architektur sowie eine Auswahl dazu kompatibler wiederverwendbarer Produkte, von den grundlegenden Abstraktionen und Strukturen des Anwendungsfeldes bis hin zu Softwarekomponenten. Die Architektur stellt dabei die Grundlage für die Realisierung einer spezifischen Anwendung dar, ein Skelett, in das sowohl die im System enthaltenen wiederverwendbaren Produkte als auch neu zu realisierende, anwendungsspezifische Funktionalität eingefügt werden können.

In der vergangenen Förderungsperiode wurden im Teilprojekt B10 vorkonfektionierte Systeme für die Anwendungsgebiete Gebäudeautomation [1][2] und Multimedia-Server [4] realisiert. Anhand dieser Systeme wurden verschiedene für das Konzept vorkonfektionierter Systeme grundlegende Aspekte betrachtet. Bei der Entwicklung der beiden Systeme für das Anwendungsgebiet Gebäudeautomation wurde besonderes Augenmerk auf die hier möglichen architekturellen Alternativen sowie auf eine weitgehende Unterstützung von Wiederverwendung bei der Anwendungsentwicklung gelegt. In beiden Systemen wurden indirekte Kommunikationsmechanismen für die interne Ablaufsteuerung verwendet, die die Integration von wiederverwendbaren und anwendungsspezifischen Anwendungsbestandteilen sehr vereinfachen, sich aber bezüglich der Einbettung anwendungsspezifischer Funktionalität unterscheiden.

Anhand des vorkonfektionierten Systems für Multimedia-Server wurde exemplarisch untersucht, wie spezielle nichtfunktionale Anforderungen (hier die Auslieferung von Multimedia-Strömen mit garantierten Qualitätsmerkmalen) bei der initialen Auslegung des Systems berücksichtigt und anschließend bei der Entwicklung einzelner Anwendungen gezielt umgesetzt werden können. Zusammen mit der auch in diesem Fall geforderten Flexibilität beeinflussten diese Anforderungen wesentlich das architekturelle Design des vorkonfektionierten Systems.

Die Wiederverwendungseigenschaften dieses Systems wurden anschließend im Rahmen einer Fallstudie untersucht. Dabei wurde auf Basis des erstellten vorkonfektionierten Systems ein Server für ein komplexes, im ursprünglichen Design nicht berücksichtigtes Multimedia-Datentransfer- und Kontrollprotokoll realisiert. Die im vorkonfektionierten System vorhandenen Strukturen, Mechanismen und Produkte erwiesen sich hierbei als tragfähige Basis; beispielsweise konnte der neue Server innerhalb der dem vorkonfektionierten System zugrundeliegenden Architektur und ohne strukturelle Änderungen realisiert werden.

3.2 Arbeitsschwerpunkt B: Werkzeugunterstützung für vorkonfektionierte Softwaresysteme

Das Ziel dieses Arbeitsschwerpunktes bestand darin, geeignete Techniken, Methoden und Werkzeuge für die Anwendungsentwicklung mit vorkonfektionierten Systemen bereitzustellen. Dazu wurden zunächst bei der Entwicklung der oben erwähnten vorkonfektionierten Systeme verschiedene Beschreibungstechniken eingesetzt und evaluiert, sowie einige Techniken speziell für die Belange vorkonfektionierter Systeme weiterentwickelt. Diese Techniken wurden anschließend in ein initiales Vorgehensmodell für die Anwendungsentwicklung mit vorkonfektionierten Systemen integriert; des Weiteren wurden entsprechende Werkzeuge bereitgestellt, um ihre Anwendung zu unterstützen.

Beschreibung vorkonfektionierter Systeme

Das zentrale Strukturierungsmerkmal eines vorkonfektionierten Systems ist die ihm zugrundeliegende Architektur. Wie oben erwähnt, basiert der im Teilprojekt B10 gewählte Architektur-Beschreibungsansatz auf einem Sichtenmodell; einzelne Aspekte der Architektur werden in verschiedenen Sichten jeweils von unterschiedlichen Standpunkten aus dargestellt. Dieser Ansatz gewährleistet eine große Flexibilität für eine dem jeweiligen vorkonfektionierten System angepasste Architekturbeschreibung. Als Folge gibt es aber auch keine allgemeingültige Menge von Beschreibungstechniken, die für alle vorkonfektionierten Systeme geeignet ist. Statt dessen ist für jedes System ein eigener Satz an Beschreibungsmethoden anzugeben.

Im Folgenden sind beispielhaft einzelne Sichten und Beschreibungstechniken aufgezählt, die im Rahmen der entwickelten vorkonfektionierten Systeme Verwendung fanden.

- Architekturbeschreibungen mit der ADL Darwin [19] bieten die Möglichkeit, die Aufteilung des Gesamtsystems in Komponenten und die sie verbindenden Konnektoren zu beschreiben. Die formale Semantik der ADL lässt bereits auf dieser Abstraktionsebene gezielte Analysen zu.
- Auf niedrigeren Abstraktionsebenen wurden objektorientierte Modellierungstechniken verwendet, um das Design einzelner Komponenten zu beschreiben, beispielsweise die OMT-Methode [20] und die UML [21]. Die Vielzahl der mit UML spezifizierbaren Modelle erlaubt eine umfassende Beschreibung verschiedener Aspekte des Komponentendesigns.
- Zur Formulierung von Anforderungen bei der Anwendungsentwicklung mit vorkonfektionierten System wurden Szenarien oder Use-Cases [21] benutzt,

die eine sehr natürliche Darstellung der Anforderungen ermöglichen. Solche Szenarien können bereits während der Entwicklung des vorkonfektionierten Systems auf bestimmte Systemstrukturen abgebildet werden; dieses Wissen kann bei der Anwendungsentwicklung zur Unterstützung des architekturellen Designs genutzt werden. Eine weitere Anwendungsmöglichkeit von Szenarien ist Architekturanalyse beispielsweise in Form der erweiterten SAAM-Methode [11].

- Als weitere Technik zur Beschreibung von Anforderungen kam die Software Cost Reduction-Methode (SCR) [18] zum Einsatz. Die Anforderungsspezifikation anhand dieser Methode kann dabei durch domänenspezifische Anforderungsrahmen und -templates [12] unterstützt werden, die auf generische Weise für das jeweilige Anwendungsgebiet typische Problemstellungen beschreiben.
- Das einem vorkonfektionierten System zugrundeliegende Vokabular wurde jeweils in einem Dictionary zusammengefasst und definiert.
- Bei der Modellierung der Datenstrukturen eines Systems wurde auf Entity-Relationship-Diagramme [17] zurückgegriffen.
- Design Space-Profile [15] stellen einen weiteren Bestandteil der Beschreibung vorkonfektionierter Systeme dar. Zum einen kann auf diese Weise das generelle Anwendungsgebiet eines vorkonfektionierten Systems umrissen werden. Zum anderen können in entsprechenden Sub-Design Spaces die relevanten Charakteristika von im vorkonfektionierten System enthaltenen wiederverwendbaren Produkten festgehalten bzw. Anforderungen an noch zu realisierende anwendungsspezifische Systembestandteile ausgedrückt werden.
- Zur Beschreibung nichtfunktionaler Eigenschaften – beispielsweise der von einem vorkonfektionierten System benötigten CPU-Zeit und Kommunikationsbandbreite – wurde eine spezielle, in der AG Systemsoftware entwickelte Technik für die Modellierung von Ressourcenverfügbarkeit und -Nutzung [16] eingesetzt.

Softwareentwicklung mit vorkonfektionierten Systemen

Die Anwendungsentwicklung mit vorkonfektionierten Systemen basiert auf der Kombination der Wiederverwendung einer Architektur als Strukturierungskonzept mit dem Konzept der architekturbasierten Wiederverwendung, d.h. der Nutzung architekturellen Wissens zur Unterstützung von Produkt-Wiederverwendung. Dadurch sowie durch den im Vorfeld geplanten Einsatz wiederverwendbarer Produkte können Effektivität und Wiederverwendungsgrad bei der Durchführung von Anwendungsprojekten erhöht werden. Wie in [13] und [14] dargestellt, müssen diese Aspekte bei der Durchführung aller Entwicklungsaktivitäten explizite Berücksichtigung finden. Anders als im Falle von ad-hoc-Wiederverwendung sind Wiederverwendungsaspekte hier fester Bestandteil des Entwicklungsprozesses; sie können geplant werden und bei Kosten- und Aufwandsabschätzungen Berücksichtigung finden.

Im Teilprojekt B10 wurde in der vergangenen Förderungsperiode eine solche Vorgehensweise für die Anwendungsentwicklung mit vorkonfektionierten Systemen erarbeitet, die Wiederverwendungsaspekte und die Nutzung architekturellen Wissens explizit aufnimmt; sie wird in [12] im Überblick beschrieben. In einer internen Fallstudie wurde diese Vorgehensweise anschließend erfolgreich anhand eines einfachen vorkonfektionierten Systems für das Anwendungsgebiet Gebäudeautomation erprobt. Insbesondere durch die Bereitstellung erprobter Strukturen, die bei der Anwendungsentwicklung mit geringfügigen Änderungen übernommen werden können, wurde in dieser Fallstudie der Entwicklungsaufwand beträchtlich verringert.

Unterstützungstechniken und -werkzeuge

Die oben bereits erwähnte Design Space-Technik [15] kann im Kontext vorkonfektionierter Systeme für verschiedene Zwecke eingesetzt werden. Zum einen erlaubt sie die Überprüfung von Konsistenzkriterien für die Erfassung von Anforderungen oder für bestimmte Designentscheidungen. Zum anderen können in Form von Korrelationen im Design Space Heuristiken zur Abbildung beispielsweise von Anforderungen auf geeignete Lösungsstrukturen ausgedrückt werden.

In der vergangenen Förderungsperiode wurde im Teilprojekt B10 schwerpunktmäßig untersucht, wie die Auswahl eines geeigneten vorkonfektionierten Systems für eine bestimmte Anwendung mit Hilfe der Design Space-Technik unterstützt werden kann. Eine weitere Anwendung finden Design Spaces in späteren Entwicklungsphasen bei der Auswahl wiederverwendbarer Produkte, die vom eingesetzten vorkonfektionierten System zur Verfügung gestellt werden. Dazu wurden Design Spaces für Gebäudeautomationssysteme [1][2] sowie für Multimedia-Server [4] aufgestellt. In [10] wurde gezeigt, wie Design Spaces als Basistechnologie in einem Produktlinien-Ansatz verwendet werden können, der es bei geeigneter Werkzeugunterstützung erlauben soll, Mitglieder einer Produktfamilie automatisch zu generieren.

Bei der Verwendung von Design Spaces stellt deren inhärente Komplexität ein prinzipielles Problem dar. Zur Beherrschung dieser Komplexität wurden in der vergangenen Förderungsperiode in Zusammenarbeit mit dem Teilprojekt B5 zwei Werkzeuge entwickelt. Das erste Werkzeug, *Reboost*, erlaubt die Erstellung und Wartung von Design Spaces; es bietet die Möglichkeit, durch einen Design Space zu navigieren und entsprechende Beschreibungselemente einzufügen, bzw. den Design Space an eine neue Situation anzupassen. Das zweite Werkzeug, *D-Space-1*, wurde im Rahmen einer Diplomarbeit [3] entwickelt; es dient dazu, Belegungen in einem Design Space vorzunehmen, d.h., Anforderungsprofile sowie Profile wiederverwendbarer Produkte zu erstellen. Das Tool unterstützt den Entwickler dabei durch die automatische Überprüfung der im Design Space kodierten Abhängigkeiten.

Eine weitere im Zentrum der Arbeiten im Teilprojekt B10 stehende Technik ist die erweiterte SAAM-Analyse [9][11]. SAAM ist ein szenariobasiertes Analyseverfahren, bei dem typische Interaktionen mit einem System mit Hilfe von Szenarien beschrieben werden; bei ihrer Auswertung wird untersucht, ob und mit welchem Aufwand sie mit der vorliegenden Architektur umgesetzt werden können. Im Umfeld vorkonfektionierter Systeme wird die erweiterte SAAM-Analyse beispielsweise dafür eingesetzt, die Wiederverwendungseigenschaften sowohl eines ganzen Systems als auch einzelner wiederverwendbarer Produkte zu evaluieren sowie den Rahmen für deren Einsetzbarkeit zu dokumentieren. Desweiteren sind SAAM-Analysen fester Bestandteil von auf vorkonfektionierten Systemen basierenden Anwendungsprojekten, um die Integrität und Qualität der entwickelten Anwendung zu überwachen, sowie zur Unterstützung von Auswahl und Parametrisierung wiederverwendbarer Artefakte.

Neben den zuvor erwähnten Design Space-Werkzeugen wurden in der vergangenen Förderungsperiode in B10 zwei weitere Werkzeuge entwickelt [6]: ein datenbankgestütztes Protokollierungswerkzeug, das während der Durchführung von Fallstudien die einfache, aber dennoch exakte Protokollierung von Aufwand und entdeckten Fehlern erlaubt, sowie ein ebenfalls datenbankgestütztes Werkzeug für Erstellung und Pflege domänen- und anwendungsspezifischer Dictionaries. Ein solches Dictionary ist Bestandteil jedes vorkonfektionierten Systems; es definiert die darin vorkommende Begriffswelt sowie die Zusammenhänge zwischen diesen Begriffen. Während der Realisierung einer spezifischen Anwendung wird dieses Dictionary um projektspezifisches Wissen erweitert bzw. verfeinert; neben komfortablem Suchen und Browsing des Dictionaries werden auch diese Änderungsaktivitäten von dem realisierten Werkzeug unterstützt.

Eine Evaluierung der zuvor erstellten Werkzeuge erfolgte an Hand kleinerer Fallstudien, mit denen der Nutzen beispielsweise der Design Space-Werkzeuge bestätigt werden konnte. Durch die beiden Werkzeuge wurden Erstellung und Handhabung von Design Spaces sowie die Überprüfung der Einhaltung der Konsistenzbedingungen im Vergleich zu manueller Erstellung und Konsistenzprüfung signifikant erleichtert. Das Dictionary-Werkzeug wurde darüberhinaus bereits im Rahmen des Querschnittsprojekts Q1 [7] im SFB erfolgreich eingesetzt; dadurch konnten den beteiligten Entwicklern Definitionen aller im Entwicklungsprojekt auftretenden Begriffe und Konzepte zeitnah und konsistent zur Verfügung gestellt werden.

4 Zusammenarbeit mit anderen Teilprojekten

Bei den Arbeiten im Querschnittsprojekt Q1 wurden von den beteiligten B10-Mitarbeitern Aspekte der Wiederverwendung von Domänenwissen sowie des Konzepts der geplanten Wiederverwendung in das im Team erstellte Referenzprozessmodell eingebracht. Die Berücksichtigung dieser Gegebenheiten stellte eine Voraussetzung für die anschließende Einordnung der Arbeiten des Teilprojekts in das Referenzprozessmodell dar.

Im Querschnittsprojekt Q2 wurde in einer Fallstudie im Rahmen eines Praktikums ein komplexes Gebäudeautomationssystem entworfen und auszugsweise implementiert. Dabei waren Mitarbeiter aus dem Teilprojekt B10 an den Reviews des Systemdesigns beteiligt, um zum einen die entwickelten Architekturen zu bewerten, und zum anderen Design-Empfehlungen im Hinblick auf die Realisierung spezifischer nichtfunktionaler Eigenschaften zu geben.

Das Querschnittsprojekt Q3 wurde durchgeführt, um die in den vergangenen Förderungsperioden gesammelten Erfahrungen bezüglich der Domäne Gebäudeautomation in einem Domänenmodell zusammenzufassen. Der Beitrag des Projekts B10 zu diesem Querschnittsprojekt bestand unter anderem in der Bereitstellung von Verfahren zur Domänenanalyse; als initiales Verfahren wurde hier die Commonality Analysis des FAST-Ansatzes [22] ausgewählt. Im Laufe der Domänenanalyse wurde dieses Verfahren an die Bedürfnisse der Domäne, sowie des personellen Umfeldes angepasst. Die Ergebnisse des Projekts in Form des angepassten Domänenanalyseverfahrens und eines Modells der Domäne Gebäudeautomation sind in einem SFB-Bericht [5] zusammengefasst.

5 Eigene Veröffentlichungen im Berichtszeitraum

Diplomarbeiten

[1] S. Albus: *Eine Design Space-Klassifikation und Realisierung eines eventgesteuerten vorkonfektionierten Systems zur Gebäudeautomation*, AG Systemsoftware, Universität Kaiserslautern, 1998

[2] S. Bötzel: *Eine Design Space-Klassifikation und Realisierung eines marktbasierten vorkonfektionierten Systems zur Gebäudeautomation*, AG Systemsoftware, Universität Kaiserslautern, 1998

[3] A. Gilbert: *Design-Space-basierte Konfiguration des Betriebssystems QNX*, AG Systemsoftware, Universität Kaiserslautern, 1999

[4] F. Hüther: *Ein vorkonfektioniertes System für echtzeitfähige Multimedia-Server, 1998*

SFB-Berichte

[5] L. Geyer, M. Kronenburg, R. Merz, B. Schürmann: *Domain Modeling in the SFB 501*, SFB 501 Report 07/00, Dept. of Computer Science, University of Kaiserslautern, 2000

[6] L. Baum, M. Becker, L. Geyer, G. Molter: *Zwei Unterstützungswerkzeuge für Experimente auf dem Gebiet domänenspezifischer Softwareentwicklung*, SFB 501 Report 14/99, Dept. of Computer Science, University of Kaiserslautern, 1999

[7] R. Feldmann, J. Münch, S. Queins, S. Vorwieger, G. Zimmermann: *Baselining a Domain-Specific Software Development Process*, SFB 501 Report 02/99, Dept. of Computer Science, University of Kaiserslautern, 1999

Vorträge

[8] J. Nehmer: *Softwarearchitektur – Ein Schlüsselkonzept zur Bewältigung der Softwarekrise*, Universität Würzburg, Juli 2000

[9] L. Geyer: *Architekturanalyse in wiederverwendungsorientierten Entwicklungsprozessen,* Reuse-Workshop des SFB 501, Februar 1999

Tagungsbeiträge

[10] L. Baum, M. Becker, L. Geyer, A. Gilbert, G. Molter, V. Tamara: *Supporting Component-Based Software Development Using Domain Knowledge*, Proc. of the 4th World Multiconference on Systemics, Cybernetics and Informatics SCI 2000, Orlando, USA, 23-26 Juli 2000

[11] G. Molter: *Integrating SAAM in Domain-Centric and Reuse-based Development Processes*, Proc. of the 2nd Nordic Workshop on Software Architecture (NOSA), Ronneby, Sweden, 12-13 August 1999

[12] L. Baum, M. Becker, L. Geyer, G. Molter: *A Process View on Architecture-Based Software Development*, Proc. of the 1st Working IFIP Conference on Software Architecture (WICSA-1), San Antonio, Texas, USA, 22-24 Feb. 1999

[13] L. Baum, M. Becker, L. Geyer, G. Molter: *The Role of Architecture for Complex Systems Development*, Proc. of the 11th. Int'l Conf. on Software Engineering and its Applications 1998, Paris, France, 8-10 Dec. 1998

[14] L. Baum, M. Becker, L. Geyer, G. Molter: *Using Software Architecture as a Catalyst for Reuse*, Proc. of European Reuse Workshop 1998, Madrid, Spain, 4-6 Nov. 1998

[15] L. Baum, L. Geyer, G. Molter, S. Rothkugel, P. Sturm: *Architecture-centric Software Development Based on Extended Design Spaces*, Proc. of the 2nd ARES Workshop (Esprit 20477), Las Palmas de Gran Canaria, February 1998

[16] L. Baum, T. Kramp: *Towards a Uniform Modeling Technique for Resource-Usage Scenarios*, Proc. of the 1999 Int'l Conf. on Parallel and Distributed Processing Techniques and Applications (PDPTA'99), Las Vegas, Nevada, USA, 28. June - 2. July 1999

6 Fremdveröffentlichungen

[17] P. P. Chen: *The Entity-Relationship Model – Towards a Unified View of Data*, ACM Transactions on Database Systems, Vol. 1, No. 1, S. 9-36, März 1976

[18] Heninger, K. L.: *Specifying software requirements for complex systems: New techniques and their application*, IEEE Transactions on Software-Engineering, SE-6(1), pp. 2-13, 1980

[19] Jeff Magee, Naranker Dulay, Susan Eisenbach, Jeff Kramer: *Specifying Distributed Software Architectures*; Proc. of the 5th European Software Engineering Conference (ESEC '95), Sitges, September 1995, Springer LNCS 989, 1995, pp. 137-153

[20] J. Rumbaugh, M. Blaha, W. Premerlani, F. Eddy, W. Lorensen: *Object-Oriented Modeling and Design*, Prentice Hall, 1991

[21] J. Rumbaugh, I. Jacobsen, G. Booch: *The Unified Modeling Language – Reference Manual*, Addison-Wesley, Object Technology Series, 1999

[22] D. Weiss: *Software Product-Line Engineering: A Family-Based Software Development Process*, Addison Wesley, 1999