T U M

I N S T I T U T   F Ü R   I N F O R M A T I K

SysLab – Abschlußbericht

Manfred Broy, Ruth Breu, Franz Huber, Ingolf Krüger,
Bernhard Rumpe, Wolfgang Schwerin

TUM-I0008
Februar 00

TECHNISCHE UNIVERSITÄT MÜNCHEN

TUM-INFO-02-I0008-100/1.-FI

Druck: Institut für Informatik der
Technischen Universität München

SysLab
1.10.1994 – 30.9.1999
Abschlußbericht

Manfred Broy, Ruth Breu, Franz Huber,
Ingolf Krüger, Bernhard Rumpe, Wolfgang Schwerin

Forschungslabor SysLab
Fakultät für Informatik
Technische Universität München
http://www4.in.tum.de
Finanziert durch Mittel der DFG aus dem Leibniz-Programm

Inhaltsverzeichnis

1 Ausgangspunkt und Fokus für die Forschungsarbeiten

Der Leibniz-Preis der Deutschen Forschungsgemeinschaft ermöglicht es, im Gegensatz zu vielen anderen Forschungsförderungswegen, wo grundsätzlich auf Antrag durchgeplante Forschungsprojekte genehmigt werden, Forschung sehr flexibel und aktuell zu organisieren. Insbesondere können schnell neue Fragestellungen aufgegriffen und soweit aufbereitet werden, daß sie die Grundlage für weiterführende Forschungsarbeiten ergeben, die dann über Anträge in den üblichen Forschungsförderungswegen weitergeführt werden können.

1.1 Ausgangspunkt

Das Gebiet Softwaretechnik ist geprägt von einem schnellen Wandel und seiner umfassenden technischen und betriebswirtschaftlichen Bedeutung. Dies führt dazu, daß in der Wirtschaft große Anstrengungen in Forschung und Entwicklung unternommen werden. Dabei stehen neben technischen Neuerungen immer auch marktpolitische Ziele stark im Vordergrund. Dementsprechend ist die Praxis der Softwaretechnik nicht immer nur von technischen und wissenschaftlichen Qualität geprägt sondern stärker noch von Markstrategie und Wettbewerb. Hinzu kommt eine schnelle und starke Verschmelzung von Aspekten der Softwaretechnik mit den unterschiedlichsten Anwendungsgebieten.

Die akademische Forschung im Bereich der Softwaretechnik hat entsprechend dieser praktischen Situation einen schwierigen Stand. Einerseits ist sie bemüht, die in der Praxis in großer Fülle hervorgebrachten und eingesetzten ad hoc Methoden zu konsolidieren und dafür eine wissenschaftliche Fundierung zu entwicklen, gleichzeitig muß sie versuchen, in der technischen Entwicklung in der Praxis mitzuwirken und diese zu beeinflussen und nicht zuletzt immer wieder auch alternative Ansätze hervorzubringen, um fragwürdige technische Lösungen der Praxis zu hinterfragen und Gegenvorschläge zu erarbeiten. Verglichen mit dem ungeheueren Aufwand, mit dem die Industrie die Entwicklungen vorantreibt, gerät die akademische Forschung zwangsläufig immer wieder ins Hintertreffen. Vor diesem Hintergrund läuft sie in Gefahr sich eigene Forschungsthemen zu suchen, die, wenn auch anspruchsvoll und im Sinne der Grundlagenforschung wissenschaftlich qualitativ hochstehend, doch ohne nachhaltigen Einfluß auf die Praxis bleiben. Dadurch entsteht eine oftmals beklagte Kluft zwischen Wissenschaft und Praxis.

Vor diesem Hintergrund hat sich die Forschungsgruppe von Prof. Dr. Manfred Broy zum Ziel gesetzt, eine starke Integration zwischen akademischer Grundlagenforschung und praktischen Fragestellungen zu erreichen. Der Leibniz-Preis mit seinen weitgehend flexiblen Möglichkeiten Forschungsfragen anzugehen war dafür ein idealer Ausgangspunkt. Ziel der angestrebten Arbeiten war neben einer engen Zusammenführung von erarbeiteten Ergebnissen der Grundlagenforschung mit praktischen Fragestellungen auch die Konzeption und Durchführung weiterer Forschungsprojekte auf Basis zusätzlich eingeworbener Drittmittel. Dadurch konnte insbesondere sichergestellt werden, daß die Mittel der Deutschen Forschungsgemeinschaft aus dem Leibniz-Programm ausschließlich Fragen der Grundlagenforschung zugute kamen, während die dabei tangierten praktischen Fragestellungen durch Mittel aus angelagerten Industrieprojekten bestritten werden konnten. Die beschriebene Strategie konnte in der Laufzeit von fünf Jahren erfolgreich umgesetzt werden, weil im Umfeld der Grundlagenforschungsarbeiten des Forschungslabors SysLab, die durch die Mittel der Deutschen Forschungsgemeinschaft gefördert wurden, eine Vielzahl von kleineren Industrieprojekten, aber auch mehrere größere Forschungsvorhaben gestartet werden konnten, die mit den Grundlagenarbeiten in einem engen inhaltlichen Austausch waren. Grundsätzlich kann festgestellt werden, daß mit den Leibniz-Mitteln die Forschergruppe um Prof. Broy erfolgreich eine deutliche Ausweitung ihres wissenschaftlichen Spektrums vorgenommen und eine Synergie zwischen praktischen und theoretischen Fragestellungen im Bereich der Softwaretechnik erreicht hat.

1.2 Die Bedeutung der Softwareentwicklung

In Produkten, Dienstleistungen und firmeninternen Arbeitsgängen, aber auch in der Wissenschaft, erlangen Softwaresysteme zunehmend strategische Bedeutung im Hinblick auf Effizienz, Qualität, Produktivität und Flexibilität. Dadurch wird Software definitiv zu einem kritischen Erfolgsfaktor jedes Unternehmens und jeder Forschungseinrichtung[1].

Der rasante Fortschritt im Bereich der Computertechnik eröffnet immer weitere Möglichkeiten des Softwareeinsatzes in wissenschaftlichen, technischen und kommerziellen Anwendungsgebieten. Heute ist Software ein integraler Bestandteil fast aller hochwertigen technischen Produkte und Dienstleistungen. Ganze Branchen, wie Banken und Versicherungen, aber auch der gesamte administrative Bereich und nicht zuletzt das Gesundheitswesen sind von modernen Softwaresystemen zur Unterstützung von Geschäftsabläufen vital abhängig. Die Effektivität moderner Unternehmen und Organisationen wird maßgeblich durch den Einsatz effizienter betrieblicher Informationssysteme und Workflow Management Systeme bestimmt. Moderne Informations- und Kommunikationstechnik eröffnet neue Vertriebswege, die zu einer zunehmenden Internationalisierung der Märkte führen. Auch im Fertigungsbereich ist der Fortschritt der Automatisierung nicht mehr aufzuhalten. Produktivität und Durchlaufzeiten werden hier maßgeblich von der Leistungsfähigkeit und Zuverlässigkeit der zur Produktionssteuerung eingesetzten Software beeinflußt. Selbst der Leistungsumfang der Erzeugnisse wird nicht zuletzt von der darin integrierten Software bestimmt. Die enormen Innovationen bei fast allen hochwertigen technischen Produkten, wie beispielsweise bei Haushaltsgeräten, im Bereich Unterhaltungselektronik, aber auch im Automobilbereich, wären ohne den Einsatz eingebetteter Softwaresysteme nicht denkbar. Die Entwicklung der Software für diese Produkte stellt einen stetig wachsenden Anteil der Gesamtkosten dar.

Das breite Spektrum für den Einsatz von Computertechnik und dazugehörenden Softwaresystemen hat dazu geführt, daß Software in immer größerem Maß zu einem eigenständigen Wirtschaftsgut geworden ist. Bei den Gesamtkosten von Computersystemen bzw. Anwendungssystemen ist der relative Wertanteil der Software in den letzten Jahrzehnten ständig gestiegen. So ist die Informationstechnik- und Telekommunikationsbranche ein aufstrebender Wirtschaftszweig und stetig wachsender Dienstlei-

[1] Dieser Abschnitt ist teilweise dem Leitfaden zum Software Engineering [BEP+99] entnommen

stungssektor geworden.

Die zunehmende Integration von Softwareprodukten in Form eingebetteter Systeme stellt in vielen Unternehmen die Software und deren Entwicklung eine strategische Aufgabe dar, die durch eigene oder externe Entwicklungsabteilungen bewältigt werden muß. Während früher größere Unternehmen ihre Software im wesentlichen selbst entwickelten, ist dies heute eher die Ausnahme. Zunehmend wird Software nicht mehr im eigenen Haus entwickelt, sondern bei externen Softwarehäusern in Auftrag gegeben. Dies gilt immer häufiger auch für die Bereitstellung und den Betrieb der gesamten DV-Infrastruktur. Die Vielfalt der in modernen Unternehmen eingesetzten Software und der Aufwand für ihre Entwicklung zwingt immer mehr Unternehmen, sich bei den Eigenentwicklungen auf Kernkompetenzen zu beschränken. Durch diesen Outsourcing-Trend hängt die Existenz einer wachsenden Zahl von Unternehmen von der erfolgreichen Durchführung extern oder intern vergebener Sofwareentwicklungsprojekte ab.

2 Überblick über das SysLab-Forschungslabor

Am Institut für Informatik der Technischen Universität München wurde unter der Leitung von Prof. Dr. Manfred Broy ein Forschungslabor zur System- und Softwaretechnik, kurz SysLab, betrieben. Das von der Deutschen Forschungsgemeinschaft mit den Mitteln in Höhe von 1.5 Millionen DM eines Leibnizpreises geförderte Forschungslabor hat am 1.10.1994 mit seinen Arbeiten begonnen. Die Förderung durch DFG-Mittel aus dem Leibniz-Programm endete nach einer Laufzeit von fünf Jahren am 30.9.1999.

Das Forschungslabor SysLab hat in den ersten 5 Jahren seiner Laufzeit die grundlagenorientierten Forschungsaktivitäten am Lehrstuhl von Prof. Broy im Bereich der Softwaretechnik gebündelt und die Basis für eine stärker an praktischen Fragestellungen der Softwaretechnik orientierte Ausrichtung der gesamten Forschergruppe gelegt. Das Forschungslabor war der Grundstein und Wegbereiter einer Reihe weiterer Projekte des Lehrstuhls, von denen hier stellvertretend der Forschungsverbund Softwaretechnik (FORSOFT) erwähnt sei. In den nachfolgenden Abschnitten sind die Aktivitäten und die daraus resultierenden Ergebnisse des Forschungslabors detaillierter erläutert.

Die Flexibilität der durch den Leibnizpreis zur Verfügung gestellten Mittel erlaubte der dadurch geförderten Gruppe von wissenschaftlichen Mitarbeitern gleichzeitig ein hohes Maß an finanzieller Planungssicherheit und forscherischer Gestaltungsfreiheit. So trat kein langer Leerlauf zwischen Planung (normalerweise festgelegt durch einen Antrag) und dem Projektbeginn (festgelegt durch die Mittelbewilligung) auf, sondern es konnte der neueste Stand der Softwaretechnik direkt in aktuelle Planungen einbezogen werden. Auch im weiteren Verlauf des Projekts konnten neue internationale Strömungen und Weiterentwicklungen der Softwaretechnik flexibel integriert werden und damit die Zielsetzung des Forschungslabors SysLab angepaßt und insgesamt sehr effektiv umgesetzt werden.

Dadurch wurde auch eine hohe internationale Sichtbarkeit des Forschungslabors möglich, die sich inhaltlich bei der Mitwirkung und Einflußnahme auf die Weiterentwicklung der Softwaretechnik in Wissenschaft und Praxis widerspiegelt. Beispiele dafür finden sich bei der Definition der Unified Modeling Language (im Bereich objektorientierter Modellierung durch die OMG zum Standard erhoben), den Message Sequence Charts (im Bereich Telekommunikation durch die ITU standardisiert), den Statecharts (hierarchische Zustandsautomaten mit weiter Verbreitung) und dem Workflow/Geschäftsprozeß-Ansatz. Durch die nachfolgend beschriebenen wissenschaftlichen Veröffentlichungen, die aktive Teilnahme an themenbezogenen Konferenzen und Workshops sowie die Organisation einiger im wissenschaftlichen Bereich stark beachteter Workshops und der internationalen Konferenz ≪UML≫'99 wurde insbesondere in den genannten Bereichen aktiv Einfluß genommen.

Neben den wissenschaftlichen Aktivitäten wurden unter Einstz eingeworbener Drittmittel mehrere Kooperationen mit Industriepartnern, wie Siemens und BMW, durchgeführt, insbesondere aber indirekt durch eine Reihe an das Forschungslabor angegliederter Projekte Wissenstransfer in die Industrie aktiv vorangetrieben.

3 Ausgangspunkt und Durchführung des Forschungslabors

Die Erstellung komplexer Softwaresysteme ist von großer technischer, wirtschaftlicher, aber auch wissenschaftlicher Bedeutung. Der hohe Anwendungsdruck, bedingt durch die schnelle Entwicklung der Rechnersysteme und ihre inzwischen große Verbreitung hat dazu geführt, daß die heute in der Praxis gebräuchlichen Methoden der Softwareentwicklung primär heuristisch geprägt sind. Grundsätzlichere wissenschaftliche Fragestellungen sind in diesem Kontext kaum oder nur sehr isoliert verfolgt worden. Auffällig ist, daß die durchaus umfangreichen Arbeiten im akademischen Bereich, unter dem Stichwort *formale Methoden*, die praktischen Aspekte der Softwareentwicklung bisher nur ungenügend berücksichtigen und umgekehrt in die Praxis nur zögerlich Eingang gefunden haben.

3.1 Fundierung von Softwareentwicklungsmethoden

Softwareentwicklungsmethoden bestehen aus einer Sammlung von Beschreibungstechniken und methodischen Empfehlungen zu Vorgehen und Werkzeugunterstützung. Sie sind heuristisch begründet und meist auf spezielle Anwendungsgebiete ausgerichtet. Dabei treten grundsätzliche Fragen in den Hintergrund. So fehlt den angebotenen Beschreibungstechniken in aller Regel eine ausreichende semantische und methodische Fundierung. Vernachlässigt wird insbesondere die Integration der innerhalb einer Methode bzw. in verschiedenen Methoden propagierten Beschreibungstechniken und den damit verbundenen Sichten auf das zu entwickelnde System.

Dieser Mangel wurde in den letzten Jahren auch bei der fortschreitenden Entwicklung der Unified Modeling Language (UML) sichtbar. Jedoch hat sich hier gezeigt, daß eine semantische Fundierung nicht nur ein technisch wissenschaftlicher, sondern vor allem auch ein Prozeß der Meinungsbildung und damit ein soziologischer Prozeß ist. Dem wurde durch verstärkte internationale Präsenz und durch aktive Teilnahme an Diskussionen um die Standardisierung der UML Rechnung getragen.

Eine wohldurchdachte semantische Fundierung und methodische Integration der benutzten Beschreibungstechniken ist nach wie vor wesentliche Voraussetzung für eine größere Systematisierung und für eine mächtige Werkzeugunterstützung im Entwicklungsprozeß.

3.2 Grundsätzliche Bedeutung

Über die direkte Anwendung im Rahmen einer Softwareentwicklung hinaus sind Beschreibungstechniken der Informatik auch von weitreichender wissenschaftlicher Bedeutung. Ein standardisierter, gut aufeinander abgestimmter Baukasten von Beschreibungstechniken wird sich in der Zukunft als ebenso wichtig und einflußreich erweisen, wie die Modellierungstechniken der Mathematik in ihrer Wirkung auf die Physik in der 2. Hälfte des vorigen Jahrhunderts. Die in der Mathematik erarbeiteten Techniken der Modellierung von physikalischen Vorgängen waren die entscheidende Basis für eine Vielzahl wissenschaftlicher Erkenntnisse, insbesondere in den Naturwissenschaften. Einen vergleichbaren Beitrag dürften die Beschreibungstechniken der Informatik für die Gestaltung komplexer Softwaresysteme, aber auch für die Konzeption und Analyse technischer und betriebwirtschaftlicher Systeme erbringen. Vor dem Hintergrund dieser grundsätzlichen Bedeutung ist eine wissenschaftliche Fundierung eine Aufgabe mit hoher Priorität.

3.3 Auswahl des Anwendungsbereichs

Es ist wohlbekannt und akzeptiert, daß es keine Methode geben kann, die für alle Einsatzgebiete der Software- und Systementwicklung gleichermaßen anwendbar ist. Betriebswirtschaftliche Informationssysteme, E-Commerce und Internet-basierte Systeme, Softwareprodukte wie Betriebssysteme, Übersetzer, Datenbanken, Textverarbeitungssysteme oder eingebettete Steuer- und Überwachungssysteme in Produktionsanlagen oder mobilen Systemen, sowie Telekommunikationssysteme erfordern unterschiedliche Techniken und Methoden.

Eine fundierte Methodik muß daher spezifisch auf die Schwerpunkte des jeweiligen Einsatzgebiets des Systems eingehen. Da gerade die methodische Entwicklung betrieblicher Informationssysteme starke Defizite bei der formalen Fundierung aufweist, wurde in SysLab speziell dieser Anwendungsbereich intensiver behandelt. Betriebliche Informationssysteme sind im allgemeinen geprägt durch einen großen, stark strukturierten Datenanteil und komplexe interne Prozesse. Eine Anbindung von Schnittstellen an technische Systeme, bis hin zur Verarbeitung analoger Signale, wurde durch die Nutzung von Synergieeffekten in Kooperation mit anderen Projekten am Lehrstuhl erreicht.

3.4 Zielrichtung

Das primäre Ziel des SysLab-Forschungslabors war es, einen deutlichen Beitrag zur Entwicklung einer auf mathematischen Prinzipien basierenden, in der Praxis anwendbaren Software-Entwicklungsmethodik für verteilte, interaktive objektorientierte Systeme zu leisten. Dabei steht der Zielbereich betrieblicher Informationssysteme mit großem, stark strukturiertem Datenraum, typischerweise unter Einbeziehung graphischer Benutzeroberflächen, einer zentralen oder dezentralen Datenbank in einem verteilten Netz von Anwendungen im Vordergrund.

Der in SysLab geleistete Beitrag zu dieser Methodik

- unterstützt die Entwicklung durch eine Reihe sorgfältig gewählter Sichten,
- integriert die verschiedenen Sichten durch Abstützung auf ein mathematisches Systemmodell und ermöglicht so rigorose Konsistenzprüfungen,
- bietet eine Auswahl einzelner Arbeitsschritte an, die die Weiterentwicklung, Verfeinerung, Komposition oder Transformation von Dokumenten der einzelnen Sichten erlaubt, und
- unterstützt damit ein inkrementelles, auf die Verfolgung von Anforderungen und Entwurfsentscheidungen ausgerichtetes Vorgehen.

3.5 Organisationsstruktur des Forschungslabors

Das Forschungslabor war in seiner Laufzeit von insgesamt fünf Jahren in zwei organisatorische Abschnitte geteilt.

Die erste Phase erstreckte sich über den Zeitraum vom 1.10.1994 bis zum 31.1.1997 und diente der Erarbeitung grundsätzlicher Lösungsansätze im Software Engineering. Die breit gefächerte Aufgabenstellung verteilt sich auf die drei Teilprojekte - Beschreibungstechniken, Methodik und Werkzeuge und war geprägt von starker und intensiver Einbindung bisheriger Ergebnisse anderer Projekte der Forschergruppe.

Mit Beginn des Forschungsverbunds FORSOFT, dessen Wegbereiter SysLab war, hat sich auch SysLab neu ausgerichtet, und verstärkt grundsätzliche Fragestellungen im Software Engineering behandelt. Wie auch nachfolgend beschrieben haben sich SysLab und FORSOFT dabei gegenseitig ergänzt und vervollständigt.

Die Organisationsstrukltur und die jeweilis beteiligten Mitglieder des Forschungslabors können der Tabelle 1 entnommen werden.

3.6 Einbettung des Forschungslabors

In der Forschungsgruppe von Prof. Manfred Broy wird seit längerem, insbesondere im Teilprojekt A6 des Sonderforschungsbereiches (SFB 342), die mathematische Modellierung verteilter Systeme erarbeitet. In diesem Kontext wurden auch Arbeiten zur axiomatischen Spezifikation durchgeführt. Die Ergebnisse, und hier speziell die Techniken zur Verfeinerung, Abstraktion und Komposition, bildeten die Grundlage für die wissenschaftliche Fundierung der Beschreibungstechniken des Forschungslabors. Sie wurde im Wesentlichen auf graphische Beschreibungstechniken adaptiert und so einer Einbettung in eine pragmatische Methodik zugänglich gemacht.

In der Forschungsgruppe werden ständig Projekte in unmittelbarer Kooperation mit der Industrie durchgeführt. Der enge Kontakt des Forschungslabors zu diesen Projekten

Name	von	bis	Funktion/Arbeitsgebiet
Ruth Breu	01.10.95	31.07.99	Stipendium/Methodik
Christian Facchi	01.10.94	31.12.95	Interaktion
Radu Grosu	01.10.94	01.10.98	Beschreibungstechniken/OO
Rudolf Hettler	01.10.94	31.12.95	Datenmodellierung
Franz Huber	01.01.95	31.09.99	Werkzeuge
Cornel Klein	01.10.94	31.01.97	Echtzeit, Automaten
Ingolf Krüger	13.03.98	30.09.99	Interaktion
Barbara Paech	01.10.94	31.01.97	Leitung/Methodik
Bernhard Rumpe	01.10.94	31.01.97	Automaten, Verfeinerung
	01.02.97	30.09.99	Leitung/UML
Wolfgang Schwerin	01.02.97	30.09.99	Produktmodell
Veronika Thurner	15.11.94	31.01.97	Methodik

Tabelle 1: Mitglieder des SysLab Forschungslabors

hat sichergestellt, daß sich die Arbeiten nicht von der industriellen Praxis abkoppelten und damit eine praktische Umsetzbarkeit gewährleistet ist.

Das Forschungslabor war auch der Grundstein für eine Reihe weiterer Projekte, die Fragestellungen mit engem Zusammenhang zu SysLab bearbeiten. Das Forschungslabor ist damit Wegbereiter dieser Nachfolgeprojekte und stellt insbesondere sicher, daß das in SysLab erarbeitete Wissen über die schriftliche Dokumentation hinaus weiterentwickelt und angewendet wird und die Arbeiten fortgeführt werden.

Unter den mit SysLab kooperierenden Projekten seien insbesondere die folgenden erwähnt:

Focus (als Teilprojekt A6 des SFB 342) Die mathematische Fundierung der in SysLab eingeführten Beschreibungstechniken wurde unter Benutzung der Theorie stromverarbeitender Funktionen entwickelt. Dies führte auch zu Synergieeffekten im Werkzeugbereich, wo mit AutoFocus ein von SysLab und Focus gleichermaßen unterstützter Werkzeugprototyp entstanden ist.

Arcus und Bellevue (gefördert durch das BMBF bzw. durch die DFG) Beide Projekte haben sich mit Fragestellungen beschäftigt, wie die Architektur von Softwaresystemen beschrieben bzw. entwickelt werden kann. Beide Projekte haben in hohem Maß auf den von SysLab festgelegten Beschreibungstechniken aufgesetzt.

Quest (gefördert durch das BSI) Zielsetzung des Projekts Quest ist die Umsetzung graphischer Beschreibungstechniken, wie sie Autofocus bietet, in die formale textuelle Logik VSE-2, um damit spezifizierte Eigenschaften formal zu beweisen. Damit bildet Quest eine Anbindung von Verifikationsmethoden an die von SysLab bereitgestellten graphischen Beschreibungstechniken.

FORSOFT Forschungsverbund (gefördert von der Bayerischen Forschungsstiftung und einer Reihe von Firmen) Die Zielsetzungen des Forschungslabors SysLab und die erhaltenen grundlegenden Ergebnisse bilden die formale und methodische Grundlage für große Teile des industrienahen Forschungsverbunds Softwaretechnik (FORSOFT). Insbesondere die Teilprojekte A1 (Software-Architekturen), A3 (Reengineering), C2 (Modellierung eingebetteter Software) und Z (Zentralprojekt) haben intensiv mit SysLab zusammengearbeitet bzw. Ergebnisse von SysLab in ihren Arbeiten weiterentwickelt. Dazu zählen die entwickelten Beschreibungstechniken, deren mathematische Fundierung und prototypische Umsetzung in einem Werkzeug.

4 Ergebnisse

Dieser Abschnitt beschreibt die in SysLab entwickelten Ergebnisse gegliedert nach folgenden Themenfeldern:

- Das SysLab-Buch,
- Vorgehensmodell und Methodik,
- Übersichtsartikel und -vorträge,
- Grundlagen: Syntax und Semantik,
- Software-Architektur,
- Beschreibungstechniken, aufgegliedert in
  - Message Sequence Charts (MSC),
  - Statecharts und verwandte Automatenmodelle,
  - Prozeßdiagramme zur Geschäftsprozeß- und Workflowmodellierung und
  - UML,
- Werkzeugunterstützung,
- Organisation wissenschaftlicher Treffen und
- Lehre

Diese Themenfelder sind eng vernetzt und weisen starke Bezüge zu weiteren Fragen der Softwareentwicklung auf.

Als herausragendes Ergebnis des SysLab-Forschungslabors ist derzeit ein Buch mit dem Titel „Applied Software Engineering Principles for UML" in Arbeit.

4.1 Das SysLab-Buch

Das SysLab-Buch [BBH+00] mit dem Titel „Applied Software Engineering Principles for UML", das sich derzeit in Entstehung befindet, stellt eine kondensierte Fassung wesentlicher Ergebnisse von SysLab dar, die insbesondere im Hinblick auf Praxistauglichkeit und Anwendbarkeit in der Softwareentwicklung aufbereitet sind.

Zur Motivation des Buchs

Gerade durch die in SysLab intensiv praktizierte Zusammenarbeit mit der Industrie stellte sich heraus, daß grundlegende und weithin bekannte Techniken der Softwareentwicklung, wie Abstraktion, Kapselung oder Modularisierung bei der Verwendung der UML kaum genutzt werden. Systematische, gezielte Anwendung von Beschreibungstechniken oder Transformationen zwischen diesen werden im allgemeinen nicht betrieben. Es besteht eine deutlich erkennbare Lücke zwischen Techniken des Projektmanagements „im Großen" am einen Ende, wo Standardvorgehensmodelle wie das V-Modell oder der Rational Unified Process zum Einsatz kommen, und den durch verschiedene Beschreibungstechniken repräsentierten Konzepten einer Entwurfsmethode wie der UML am anderen Ende (dazu siehe auch Abbildung 1). Diese Lücke zu verkleinern ist die Zielset-

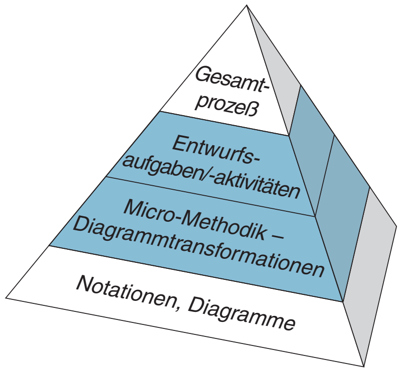

Abbildung 1: Schichten einer Entwurfsmethode

zung des Buchs, das im wesentlichen auf der Ebene der beiden mittleren in Abbildung 1 dargestellten Schichten angesiedelt ist. Diese Schichten stellen die in SysLab erarbeiteten Techniken zum methodischen Umgang mit Beschreibungstechniken der UML dar. Das Buch wendet sich also in erster Linie an Softwareentwickler, die die UML in der Praxis methodisch einsetzen wollen und ist daher auch für die Lehre prädestiniert.

Somit besteht der Hauptbeitrag des Buchs darin, einen Werkzeugkasten wohlfundierter und gleichzeitig eingängiger Techniken zur methodischen Manipulation der UML-Notationen zu liefern, wozu eine umfassende Menge von Verfeinerungsregeln, Kompositions- und Restrukturierungstechniken und ähnlicher Verfahren gehören.

Notation und Konzepte

Die notationelle Basis der im Buch behandelten Techniken sind die Kern-Notationen der UML, also Klassendiagramme, Objektdiagramme, Sequenzdiagramme, Zustandsdiagramme, sowie die Object Constraint Language (OCL) als textuelle Ergänzung zur Definition von Invarianten und Constraints im allgemeinen.

Da die Notationen der UML einen äußerst umfangreichen Baukasten mit teilweise auch redundanten Modellierungselementen bieten, wählen wir für das Buch eine in methodischer Hinsicht gerechtfertigte Untermenge aller notationellen Elemente der UML, nehmen aber auch, wo aus methodischer Sicht sinnvoll, Erweiterungen der Notationen vor.

Transformationen und methodische Richtlinien

Der Hauptbeitrag des Buchs besteht in der Darstellung konstruktiver Regeln zur Manipulation der genannten UML-Notationen. Diese Regeln verkörpern anerkannte Softwareentwurfsprinzipien. Im Einzelnen werden Tranformationsregeln angegeben, die unter anderem folgende Bereiche abdecken:

- Verfeinerungstechniken für verschiedene Notationen,
- Ableitung abstrakten Verhaltens von konkretem,
- Ableitung von Zustandsdiagrammen aus Sequenzdiagrammen,
- Refactoring von Klassendiagrammen, sowie
- Codeerzeugung aus Zustandsdiagrammen.

Methode

Da die im Buch dargestellten Techniken in erster Linie auf Methodik im Kleinen abzielen, spielt das verwendete Prozeßmodell im Großen im Buch lediglich eine untergeordnete Rolle. Jedoch werden auch wichtige Bestandteile typischer solcher Prozeßmodelle beschrieben und dabei insbesondere das Zusammenspiel der präsentierten „Micro-Methodik" mit der Softwareentwicklung im Großen erläutert.

Kerninhalte des Buchs wurden bereits auf den Konferenzen „TOOLS Europe '99" und „TOOLS Pacific '99" im Rahmen eines Tutorials einem Fachpublikum vorgestellt [BKR99] und in einer Fallstudie [BBH+99] erprobt.

4.2 Vorgehensmodell und Methodik

Bei der Entwicklung umfangreicher, komplexer Anwendungssysteme ist eine genau auf die Ziele des Entwicklungsprojekts abgestimmte Vorgehensweise unverzichtbar. Dabei ist es wichtig, die Informationen in kleinen, strukturierten Portionen zu erfassen, und den Fluß der in vorangegangenen Schritten erhobenen Informationen so zu gestalten, daß kein Verlust und keine zeitaufendige Mehrfacherfassung von Informationen auftritt. Ziel ist ein systematisches, auf Qualität und Wirtschaftlichkeit ausgerichtetes Vorgehen mit hoher Anwenderakzeptanz.

Eine zentrale Rolle bei der Softwareentwicklung spielen die in späteren Abschnitten diskutierten Beschreibungstechniken, die dazu genutzt werden, unterschiedliche Sichten auf das Softwareprodukt zu definieren. Sie werden in unterschiedlichen Entwicklungsphasen benutzt und weiterverarbeitet, bis schließlich das gesamte Softwaresystem in ausführbarer Form beschrieben ist.

Die verwendeten Beschreibungstechniken erhalten eine präzise Semantik anhand eines mathematischen Systemmodells [RKB95, KRB96] und lassen sich so semantisch und methodisch integrieren. Dokumente der benutzen Beschreibungstechniken werden dabei

- neu erstellt,
- erweitert (Verfeinerung),
- zusammengesetzt (Komposition),
- geteilt („Separation of Concerns") oder
- von einer Beschreibungstechnik in eine andere umgesetzt (Transformation und Generierung).

Wir fassen all diese Schritte unter dem Oberbegriff *Transformation* zusammen [Rum96]. Ein Transformationsschritt kann in seiner allgemeinsten Form als Abbildung einer Menge von Dokumenten auf eine neue Menge von Dokumenten verstanden werden.

Neben der sorgfältig zu wählenden Semantik für die einzelnen Beschreibungstechniken ist eine stimmige semantische, methodische und werkzeugtechnische Integration zwischen diesen Beschreibungstechniken der wesentlichste Bestandteil von SysLab.

Eine formale Semantik für Dokumente einer Beschreibungstechnik kann erst dann wirklich vorteilhaft eingesetzt werden, wenn Verfahren existieren, die eine systematische und korrekte Manipulation von Dokumenten zulassen. So kann z. B. eine zustandsbasierte Verhaltensbeschreibung für eine Klasse unter Einhaltung der Regeln eines adäquaten Verfeinerungskalküls auf Unterklassen vererbt werden.

Transformationsschritte auf Dokumenten bilden die Grundlage des Vorgehensmodells und stellen damit die Verbindung zwischen den formal fundierten Beschreibungstechniken und der Methodik dar.

Eine *präzise Methodik* besteht aus

- einem mathematischen Systemmodell,
- einer Anzahl von Beschreibungstechniken mit formaler Semantik durch das Systemmodell,
- einer Verfeinerungsrelationen abgestützt auf die Semantik,
- einer Menge von Entwicklungs- und Transformationsschritten, die im Sinne der Verfeinerungsrelation korrekt sind und
- pragmatischen Richtlinien, die den Softwareentwickler bei der Anwendung der Transformationen unterstützen.

Eine *formale Methode* bietet darüberhinaus globale Richtlinien, die charakterisieren, welche Transformationsschritte in welchen Phasen der Entwicklung und in welcher Reihenfolge angewendet werden können. Sie bietet den Anschluß zu Fragen des Projektmanagements, bestehend aus Kosten-, Qualitäts-, Zeit-, Ressourcen- und Personalplanung.

Durch die methodische Gruppierung von Transformationsschritten entsteht so ein *formal fundiertes Vorgehensmodell*. Die Formalität des Vorgehensmodells bedeutet jedoch keineswegs eine starke Einschränkung der möglichen Entwicklungsschritte. Vielmehr kann aufgabenabhängig aus einer geeigneten Menge von Transformationen ausgewählt werden.

Vorstudien und Beschreibungen zur von SysLab vorgeschlagenen Methode sind in [Pae95, Pae96b, FNDR98, PR97d, Bre99] zu finden.

4.3 Übersichten

Der von der Gruppe verfolgte Ansatz fundierter Software-Entwicklung wurde in zahlreichen eingeladenen Vorträgen und Publikationen nicht nur dem Fachpublikum, sondern auch der breiteren Öffentlichkeit vermittelt. Dies ist um so wichtiger, als der Beherrschbarkeit großer Software-Systeme in den nächsten Jahren eine immer größer werdende Bedeutung zukommen wird.

Die Vorträge und Arbeiten beschäftigen sich vor allem mit der Rolle der Mathematik bei Software-Entwicklung und Fundierung von Methoden [BHP+98, Bro96c, Bro96d, Bro97c, Bro97d, Bro96f].

Allgemeinere Arbeiten und Vorträge befassen sich mit der Rolle von Software in der Informationsgesellschaft [Bro94, Bro96e, Bro98b] und fortgeschrittener Softwaretechnik als Wettbewerbspotential [Bro96g].

4.4 Grundlagen: Syntax und Semantik

Wesentliche Erkenntnisse zur Auswahl geeigneter Beschreibungstechniken, bzw. deren syntaktischen und semantischen Varianten, wurden durch Arbeiten zur Fundierung objektorientierter Konzepte gewonnen. So wurden in [BG99] das Zusammenspiel von Ereignissen, Nachrichten und Methoden, sowie in [GS95, GS96a, GS96b, BGR+99] Mobilitätsaspekte untersucht, und in [BGH+97b, BGH+97c] Überlegungen zur prinzipiellen Integration von Beschreibungstechniken vorgenommen.

Die Frage der mathematischen Spezifikation von Systemen und Komponenten wurde auch in [Bro95a, Bro95b, Bro95c, Bro97b] untersucht; in [Bro96b, Bro96e] wurde diskutiert, wie informelle Beschreibungstechniken mit mathematischen Spezifikationen zusammenspielen. In [Bro98a] wurden dabei speziell Aspekte der Modularität untersucht.

Weitere Untersuchungen algebraischer Spezifikationen wurden hinsichtlich der von ihnen verwendeten Konzepte zur Spezifikation verteilter Softwaresysteme [Bro96a], bzw. des Datenanteils von Softwaresystemen [Het95, Het96] und zur Definition von Softwaresichten [Pae96a] vorgenommen.

Die daraus resultierenden Überlegungen und die mit Focus [BFG+93a, BFG+93b] zur Verfügung stehende Theorie strombasierter Komponenten führte zur Definition des mathematischen Systemmodells [RKB95, GKR96, KRB96, GKR96] als kompakte Beschreibung aller wesentlichen Eigenschaften von Softwaresystemen. Das Systemmodell ist keine syntaktische Beschreibung, sondern die mathematische Formulierung von Softwaresystemen als semantische Domäne. Es dient als Ausgangspunkt, um den Beschreibungstechniken durch Abbildung ins Systemmodell eine Bedeutung zu geben. Für das Verständnis der SysLab Methodik ist es für den Anwender nicht notwendig, diese präzise Beschreibung zu kennen. Stattdessen wird das Systemmodell als Grundlage für die semantische und methodische Integration der von Methodenentwicklern definierten Notationen verwendet.

Das Systemmodell beschreibt ein System als eine sich dynamisch ändernde Menge von *Objekten*, die in endlich viele *Klassen* gruppiert werden. Objekte haben einen *Zustand*, der durch ihre *Attribute* und genau genommen auch durch die Menge der aktiven Operationen und deren Zustände bestimmt wird. Eine *Signatur* beschreibt die Menge der eingehenden und ausgehenden *Nachrichten*, die sich in *Methodenaufrufe* und *Return-Nachrichten* einteilen läßt. Des weiteren werden Eigenschaften der Objekthierarchie, Kommmunikationswege und -arten, sowie das Handling von Objektkapselung und Identität festgelegt. Insbesondere werden Aspekte der Nebenläufigkeit und Synchronisation definiert. Darüberhinaus wurde eine Erweiterung für analoge Echtzeitsysteme definiert [SRS99b, SRS99a].

Die grundlegende Problematik der adäquaten Definition einer Semantik für Beschreibungstechniken wurde in [Rum98a] beschrieben und ein Vergleich der in SysLab verwendeten Technik mit einer Reihe anderer Ansätze vorgenommen. Dabei konnte festgestellt werden, daß die SysLab Technik der strikten Trennung von Syntax und Semantik, sowie die explizite Formulierung der semantischen Domäne (d.h. des Systemmodells) wesentliche Vorteile bietet. Insbesondere kann damit die Korrektheit von Transformationsschritten auf graphischen Beschreibungstechniken a priori gezeigt werden und der Anwender muß später nicht explizit mit der semantischen Domäne arbeiten. So entstehen graphische Formalismen [Rum98c].

4.5 Software-Architektur

Der beträchtliche Umfang betrieblicher Informationssysteme, die zunehmende Vernetzung ihrer Teilsysteme untereinander, die oftmals lange Einsatzdauer, sowie die mannigfaltigen Aufgaben, die während des Entwicklungsprozesses anfallen, erfordern ein tiefes Verständnis der zugrundeliegenden Software-Architektur, um ein Produkt mit der geforderten Funktions-, Termin-, Kosten-, und Qualitätstreue erzielen zu können.

Unter dem Begriff *Software-Architektur* verstehen wir dabei die Struktur eines Software-Systems, bestehend aus einer Menge von (Teil-)Komponenten und ihren Beziehungen untereinander. Diese Beziehungen können sowohl statischer als auch dynamischer Natur sein; die Festlegung eines Datenbankschemas etwa ist – in der Regel – ein statischer Aspekt, während die Aufrufbeziehungen zwischen verteilten Objekten während der Ausführung des Systems wechseln können, und somit einen dynamischen Aspekt der Software-Architektur darstellen.

Zur Beschreibung von Software-Architekturen hat sich in den letzten Jahren eine sichtenorientierte Vorgehensweise durchgesetzt. Hierbei werden die unterschiedlichen Architekturaspekte mittels geeigneter grafischer Beschreibungstechniken, wie etwa interaktionsund zustandsbasierten Ablaufbeschreibungen, repräsentiert. Dies erleichtert, in Analogie zur Vorgehensweise bei der Gebäudearchitektur, eine Fokussierung der Beschreibung auf den jeweils im Zentrum der Betrachtung stehenden Aspekt.

Um den oben genannten Software-Architektur-Begriff präziser zu fassen und gezielt Eigenschaften guter Software-Architekturen herauszuarbeiten, wurden im Rahmen einer engen Zusammenarbeit mit den bereits erwähnten Projekten Arcus, Bellevue und FORSOFT folgende Fragestellungen untersucht:

- Welche statischen und dynamischen Systemsichten eignen sich besonders zur Beschreibung von Software-Architekturen?
- Welche (grafischen) Beschreibungstechniken eignen sich besonders zur Darstellung dieser Systemsichten?
- Wie ausdrucksstark müssen die Beschreibungstechniken jeweils sein?
- In welcher Beziehung stehen die einzelnen Systemsichten und Beschreibungstechniken untereinander?
- Wie sehen geeignete Konsistenzbegriffe für unterschiedliche, überlappende Systemsichten aus?
- Welche Verfeinerungsbegriffe lassen sich für die jeweiligen Systemsichten definieren und effektiv einsetzen?
- Wie lassen sich unterschiedliche Systemsichten methodisch ineinander überführen?

Aufbauend auf Vorarbeiten aus Arcus, die sich stärker mit der Definition des Begriffs Software-Architektur [HHKR96, HHKR97], mit Entwurfsmustern [CK97], sowie einem Vergleich vorhandener Architekturbeschreibungssprachen [HR97] befaßten, wurden die Beschreibungstechniken aus ROOM und der UML auf ihre Einsatzmöglichkeit als Standardbeschreibungssprache für Software-Architekturen untersucht [RSRS99]. Message Sequence Charts (MSCs) und verwandte Notationen, wie etwa Extended Event Traces (EETs) und die Sequenzdiagramme der UML kristallisierten sich dabei als eine wesentliche Technik zur Beschreibung von Komponenteninteraktionen in Software-Architekturen heraus [BHKS97a, BHKS97b, Pae97]. Die in SysLab entwickelten Techniken zur schrittweise Verfeinerung von statischen Datenfluß-Architekturen [PR97c, PR97d, PR99] erweitern die für verhaltensorientierte Spezifikationen bereits existierenden Ansätze, so daß damit die Grundlagen für eine systematische Entwicklung sowohl der statischen, wie auch der dynamischen Architekturaspekte gelegt sind.

Vor dem Hintergrund der zunehmenden Bedeutung von Komponentenarchitekturen entstanden präzise Beschreibungen des Komponentenbegriffs an sich [Bro96f, BDH+98], sowie Beschreibungstechniken und Vorgehensweisen zur Modellierung von Komponentenarchitekturen [HRR98a, HRR98b, BHRS97]

4.6 Message Sequence Charts (MSC)

Wie im vorangegangenen Abschnitt bereits erwähnt, haben sich Message Sequence Charts (MSCs) als standardisierte Notation zur Beschreibung von Komponenteninteraktionen in verteilten Software-Architekturen in verschiedenen Varianten etabliert. In der UML beispielsweise stehen die von den MSCs abgeleiteten Sequenzdiagramme zur Verfügung; ihr Haupteinsatzgebiet ist dabei die Darstellung kurzer Interaktionsszenarien.

Der über alle Phasen des Entwicklungsprozesses hinweg durchgängige Einsatz von MSCs als Spezifikationstechnik erfordert ein tiefes Verständnis der semantischen Grundlagen, der methodisch sinnvollen Einsatzmöglichkeiten, sowie der Beziehungen zu anderen Beschreibungstechniken.

Im Rahmen von SysLab wurde ein besonderer Schwerpunkt auf die Grundlagen des methodischen Einsatzes von MSCs gelegt. In [BGH+97a] wurde, aufbauend auf [BHKS97a, BHKS97b], die Anwendung von MSCs sowohl für kurze Interaktionsszenarien, als auch für vollständige, umfassende Interaktionsarchitekturen untersucht.

Die semantischen Grundlagen, sowie die Definition geeigneter Verfeinerungs- und Kompositionsbegriffe für MSCs wurden und werden in [BK98, Bro98c, Krü99a, Krü99b] untersucht. Auch dabei steht der zielgerichtete methodische Einsatz von MSCs im Mittelpunkt der Betrachtung. So liefert beispielsweise die semantische Fundierung in [BK98] die Grundlage des Übergangs von Interaktionsspezifikationen für Komponentennetzwerke hin zu Spezifikationen für einzelne Komponenten. Diese Arbeit wurde in [BGK98, KGSB99] um ein vollautomatisches Verfahren zur Konstruktion ablauffähiger Programme aus MSC-Spezifikationen ergänzt. Dieses Verfahren wurde zum Patent angemeldet; das Deutsche Patentamt hat die Erteilung inzwischen in Aussicht gestellt. [BK98] und [BGK98, KGSB99] schliessen damit eine wesentliche Lücke beim methodischen Übergang von den frühen Phasen der Anforderungsanalyse zu Spezifikation, Design und Implementierung.

MSCs werden – über den Telekommunikationsbereich, dem ursprünglichen Haupteinsatzgebiet für MSCs hinaus, – zunehmend im Umfeld des Entwurfs technischer, und insbesondere eingebetteter Systeme, genutzt. Um auch in diesem Anwendungsgebiet ihren effektiven und durchgängigen Einsatz zu ermöglichen, wurde in [GKS99] eine auf hybride Systeme zugeschnittene MSC-Variante entwickelt und ihre Integration in eine Entwurfsmethodik für diese Systemklasse vorbereitet.

Diese Beiträge finden ihre Fortsetzung in [Krü99b], wo im Rahmen einer Promotionsarbeit sowohl die semantische Fundierung, als auch der methodische Einsatz dieser Beschreibungstechnik untersucht wird, und im SysLab-Buch [BBH+00].

4.7 Statecharts und verwandte Automatenmodelle

Automaten oder Zustands-Übergangs-Systeme bilden das Bindeglied zwischen den an Zustand bzw. Struktur orientierten Beschreibungstechniken und der an Interaktionen orientierten Sicht der MSCs.

Wir unterscheiden zwischen der konkreten graphischen Darstellung, genannt *Zustands-Übergangs-Diagramm* (STD), und der damit beschriebenen *Zustands-Übergangs-Maschine*

(STM) [GKRB96, GR95, Bro97a, Bro97e]. Die konkrete Darstellung mittels STD hat notwendigerweise nur endlich viele Zustände und Transitionen, während bereits der Zustandsraum einer STM im allgemeinen unendlich ist, wenn ein STD zur Beschreibung von Objektverhalten eingesetzt wird.

Für eine strukturierte Darstellung können STD-Zustände hierarchisch angeordnet werden. Eine den StateCharts ähnliche Zerlegung des Objektzustandsraums in gekapselte, interagierende Teilobjekte ist damit (zunächst) nicht intendiert. Gemäß dem objektorientierten Systemmodell beschränken sich STDs auf das Einlesen einzelner Zeichen.

Neben einer graphischen wurde auch eine tabellarische Darstellung von Automaten untersucht [Bro98d, FR98] sowie eine Erweiterung von Automaten für diskrete und analoge Echtzeitsysteme definiert [GS98a, GS98b, GSB98a, GSB98b].

STDs besitzen unterschiedliche Einsatzgebiete:

- Ein STD kann verwendet werden, um den kompletten *Lifecycle* eines Objekts zu beschreiben [PR97b]. Dabei spiegeln Zustände i.w. Äquivalenzklassen des Datenzustandsraums (Attribute) des Objekts wieder.
- Ein STD kann eine einzelne *Operation* beschreiben, die aufgrund anderer Objektaufrufe „Wartezustände" hat. Die STD-Zustände sind hier Kontrollzustände der Operation.
- Weiterhin eignen sich STDs auch zur Beschreibung von *Rollen* und *Features* [PR97a, KPR97].

Für die verschiedenen Varianten von STDs wurden Regelsätze angegeben, die eine Verfeinerung, sowie eine Verschmelzung (Verhaltens-Komposition) von STDs erlauben [Rum96, PR94, RK96, Rum98b].

4.8 Workflow und Geschäftsprozeßmodellierung

Während die oben besprochenen Beschreibungstechniken konzeptionell vor allem für die Beschreibung von Software-Systemen entwickelt wurden, stellt die Workflow- und Geschäftsprozeßmodellierung das System “Unternehmen” in den Vordergrund. Modelliert werden dabei Arbeitsabläufe und Vorgänge im Unternehmen, wobei kausale und zeitliche Abhängigkeiten betrieblicher Abläufe, Auslöser, Bearbeiter und benötigte Ressourcen erfaßt werden.

Arbeitsabläufe können manuell durchgeführt werden oder auch (semi-)automatisch durch Integration von Software-Systemen. So steht eine Geschäftsprozeßmodellierung oft am Anfang eines Software-Entwurfs, insbesondere wenn die Aufgaben des zu erstellenden Informationssystems a priori noch unklar sind.

Geschäftsprozesse werden meist graphisch oder textuell repräsentiert. Die Beschreibungstechniken sind informell und haben sich aus der Praxis heraus entwickelt. Durch die in SysLab vorgenommene semantische Fundierung erhalten die Beschreibungstechniken nicht nur eine formale Semantik mit Vorteilen wie Eindeutigkeit und Konsistenz, sondern es wird auch die Grundlage für eine Integration mit anderen Phasen und Techniken des Systementwurfs geschaffen. Hierbei ist vor allem der Übergang von der Modellierung von betrieblichen Abläufen zur Modellierung von Software-Systemen zu nennen.

Arbeiten, die sich mit den Beschreibungstechniken der Geschäftsprozeßmodellierung und ihrer semantischen Fundierung beschäftigen, sind [Thu97a, Thu97b, Thu98]. In [RT98] wird zudem auf den Aspekt schrittweisen Entwurfs eingegangen und ein semantisch fundierter Verfeinerungsansatz für Geschäftsprozeßbeschreibungen vorgestellt. Ein grundlegender Ansatz zur Modellierung von Vorgängen wird in [KP98a] vorgeschlagen, [KP98b] verwendet die Technik der Metamodellierung für die Konzeption von Arbeitsabläufen.

4.9 UML

Mit der immer deutlicher werdenden Dominanz der Unified Modeling Language (UML) hat sich das Forschungslabor SysLab auch auf diesen Ansatz ausgerichtet, und den entstehenden Standard zum einen auf syntaktische, semantische und methodische Defizite untersucht, zum anderen durch Vorträge und Seminare Projektpartnern und weiteren Firmen einen Überblick über die Leistungen der UML und ihren aktuellen Stand vermittelt.

Ein wesentlicher Beitrag war die Beschreibung einer Vorgehensweise zur Formalisierung der UML Beschreibungstechniken auf Basis der in SysLab aus anderen Tätigkeiten gewonnenen Erkenntnisse [BHH+97a, BHH+97b], die in Einzelaspekten weiter beleuchtet wurde [BGH+98]. In [Bre99] wird ein integrierter Ansatz vorgestellt, der die UML-Beschreibungstechniken auf eine einheitliche Basis stellt und ihre Rolle innerhalb der Methodik des nutzungsfallgesteuerten Entwurfs präzisiert.

Durch die mit diesen Veröffentlichungen erreichte internationale Aufmerksamkeit konnten wir an der Gründung der mittlerweile international bekannten Gruppe *pUML* (precise UML) partizipieren und die in SysLab gewonnenen Erkenntnisse einbringen [EFLR98, EKR99, FELR98, EBF+98].

Die so gewonnenen Erkenntnisse gipfeln letztendlich im SysLab-Buch, dessen wesentliche Inhalte auch in einem gleichnamigen Tutorial [BKR99] vermittelt werden.

4.10 Werkzeugunterstützung

Einen deutlichen Anteil an den Aktivitäten des Forschungslabors in SysLab nahm der Bereich der Werkzeugentwicklung ein. Diese wurde grundsätzlich eng verzahnt mit der Lehre durchgeführt: Wesentliche Teile der Werkzeugentwicklung wurden in Projektpraktika mit größeren Entwicklerteams von Studierenden realisiert, weitere Teile im Rahmen von Fortgeschrittenenpraktika, Systementwicklungsprojekten und Diplomarbeiten.

Grundsätzlich wurden in SysLab zwei Werkzeugentwicklungsprojekte durchgeführt, bzw. in enger Zusammenarbeit mit anderen Projekten am Lehrstuhl realisiert. So wurde zum einen zusammen mit dem Teilprojekt A6 des Sonderforschungsbereichs 342 und später auch mit dem Forschungsverbund Softwaretechnik das Werkzeug AutoFocus entwickelt. Zum anderen wurde innerhalb von SysLab mit Frisco der Prototyp einer

Dokument-basierten CASE-Plattform als Fallstudie realisiert. Beide Projekte werden im folgenden kurz beschrieben.

AutoFocus

AutoFocus [HSS96, HSSS96] ist ein Werkzeug für den komponentenbasierten Entwurf verteilter und eingebetteter Systeme, wie beispielsweise Steuerungssysteme für Fertigungsanlagen, Avioniksysteme, aber auch Buchungssysteme und Multimediadienste in intelligenten Netzwerken.

AutoFocus hat zum Ziel, praxisorientierte Systementwicklungsansätze mit formalen Techniken zu integrieren und stellt daher für Entwickler industriell gebräuchliche, graphische Beschreibungsmittel zur Verfügung, mit denen verteilte Systeme in verschiedenen, hierarchischen Sichten (u.a. Struktur-, Verhaltens- und Datensicht) entworfen werden können. Um mathematisch exakte Verifikation zu ermöglichen, sind diese Beschreibungsmittel in die formale Methode Focus, entwickelt im Teilprojekt A6 des Sonderforschungsbereichs 342, eingebettet.

Die verwendeten Beschreibungstechniken für die genannten Sichten, Systemstrukturdiagramme (SSD) für die statische Sicht eines Systems, Zustandsübergangsdiagramme (STD) zur Verhaltensbeschreibung von Komponenten, Datentypdefinitionen zur Festlegung der in einem System verarbeiteten Datentypen, sowie EETs, einer Variante von MSCs, zur Spezifikation von Beispielabläufen eines Systems oder seiner Komponenten, weisen konzeptuell viele Gemeinsamkeiten zu den in der UML verwendeten auf. Abbildung 2 zeigt eine Bildschirm-Hardcopy des AutoFocus-Werkzeugs mit einem Projekt-Browser (oben links), in dem der Inhalt des verwendeten Repositories hierarchisch dargestellt wird, einem SSD-Editor (oben mitte), einem EET-Editor (oben rechts), sowie einem STD-Editor (unten links) mit einem geöffneten Transitionseditor (unten rechts).

AutoFocus bietet die Möglichkeit, umfangreiche, auch benutzerdefinierte Konsistenzprüfungen auf Systemspezifikationen [HSE97] auszuführen. Aus konsistenten Systemspezifikationen können Prototypen generiert werden und in einer Simulations- und Visualisierungsumgebung zur Ausführung gebracht werden [HS97, HMR+98].

Das unterliegende formale Modell Focus kann in AutoFocus über eine Anbindung an verschiedene Model Checker genutzt werden, um formale Verifikation von Systemeigenschaften durchzuführen.

Im Zuge der Integration formaler und praxisorientierter Konzepte bietet AutoFocus insbesondere die typische Funktionalität von CASE -Werkzeugen: AutoFocus besitzt eine verteilte Client/Server-Architektur (siehe Abbildung 3), ausgerichtet auf den Einsatz in lokalen Netzen bzw. in Weitverkehrsnetzen. Im zentralen Repository des Servers werden alle Entwicklungsprojekte gespeichert. Hier werden auch Mehrfachzugriffe auf Entwicklungsdokumente im Mehrbenutzerbetrieb koordiniert und die Versionsverwaltung dieser Dokumente durchgeführt. Auf der Client-Seite stehen die teilweise bereits in Abbildung 2 gezeigten Werkzeuge zur Projektverwaltung und die graphischen Editoren für die einzelnen Beschreibungstechniken zur Verfügung. AutoFocus wurde plattformunabhängig in der Programmiersprache *Java* realisiert, kann also auf den meisten gebräuchlichen Systemplattformen verwendet werden.

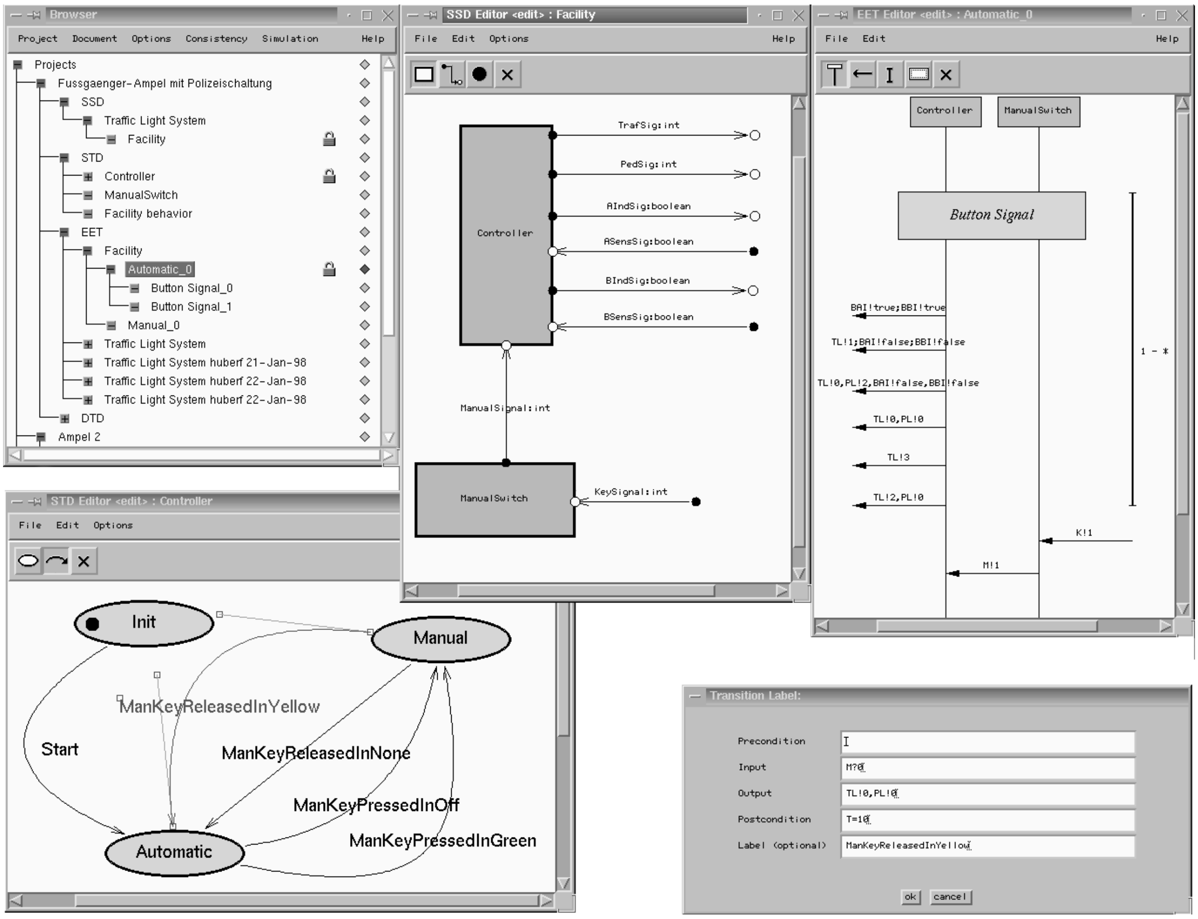

Abbildung 2: AutoFocus Projekt-Browser und Editoren für SSDs, STDs und EETs

Gegenstand aktueller Arbeiten an AutoFocus ist der Ausbau der methodischen Unterstützung von Entwicklern im Hinblick auf modularen Systementwurf [HS98], sowie ein Re-Design der Architektur des Werkzeugs [BHRS97].

AutoFocus wurde zwischenzeitlich in mehreren Fallstudien eingesetzt (beispielsweise zur Spezifikation und Verifikation einer Ampelsteuerung, siehe [HMS$^+$98]) und an externe Partner und Unternehmen im Rahmen von Evaluierungslizenzen[2] herausgegeben.

Im Rahmen des Projektes Quest wurde AutoFocus, wie schon in Abschnitt 3.6 erwähnt, durch die Anbindung verschiedener Verifikations- und Testwerkzeuge erweitert. Mit diesen Erweiterungen beteiligten sich AutoFocus und Quest an der *FM'99 Tool Competition*, einem Modellierungswettbewerb für formale Softwareentwurfswerkzeuge auf dem *Formal Methods '99 World Congress* in Toulouse. Dabei erzielte das AutoFocus/Quest-Werkzeug in einem Teilnehmerfeld von insgesamt 12 Entwicklungs-

[2] Eine Evaluierungsversion von AutoFocus kann auf der AutoFocus Homepage im World Wide Web (http://autofocus.informatik.tu-muenchen.de) per Download bezogen werden.

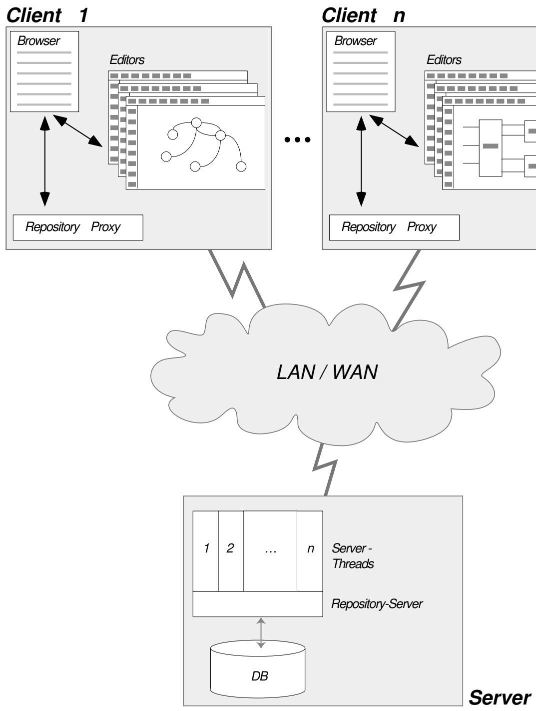

Abbildung 3: Systemarchitektur von AutoFocus

werkzeugen aus Industrie und Forschung die beste Wertung und belegte den ersten Platz des Wettbewerbs.

Frisco

Frisco ist eine Fallstudie für die Werkzeugentwicklung, die zum Ziel hat, eine CASE-Tool Plattform für Dokument-basiertes Software-Engineering zu realisieren. Im weiteren Umfeld gehören zu den Zielen von Frisco, das vollständig in Java implementiert ist, der Aufbau von weitergehendem Know-how im Bereich Java sowie im Bereich objektorientierter Modellierung, speziell natürlich der UML. Bei den Implementierungsarbeiten konnte insbesondere auf den in AutoFocus mit Java gesammelten Erfahrungsschatz [BH97] zurückgegriffen werden.

Den Kernanteil von Frisco stellt das sogenannte Frisco Open Editor Framework (Frisco OEF) dar, das auf Java-Basis einen komponentenbasierten Ansatz zur flexiblen Komposition zusammengesetzter Dokumente aus Einzeldokumenten realisiert. Im Hinblick auf die für Anwender sichtbare Funktionalität steht dabei die Möglichkeit im Vordergrund, verschiedene textuelle und graphische Beschreibungstechniken möglichst einfach in einem Dokument kombinieren zu können und werkzeugseitig Unterstützung bei der Sicherung der Konsistenz der Einzeldokumente zu erhalten. OEF stellt hierfür die Infrastruktur sowie eine gewisse Basisfunktionalität wie auch eine einheitliche Oberfläche und Präsentationsschicht zur Verfügung, die von den einzelnen, in OEF integrierten

Werkzeugkomponenten weiter spezialisiert wird.

Werkzeugkomponenten, die in OEF integriert und damit in Frisco verfügbar sind, sind eine Reihe graphischer Editoren für UML-Notationen und ein Parser für die funktionale und algebraische Spezifikationssprache Frisco F [LHR99], die ebenfalls im Rahmen der Entwicklung der Frisco-Plattform realisiert wurde. Dabei wurde in einer speziellen Komponente [FR98] ein Verfeinerungskalkül für Automaten realisiert, und so die prinzipielle Machbarkeit demonstriert.

Ein erstes Resümee von Erfahrungen in der Entwicklung von Frisco, und speziell des Frisco OEF, wurde in [HRR99] veröffentlicht.

Wie in den vorausgehenden Abschnitten beschrieben, verwendet SysLab eine Menge ausgewählter Beschreibungstechniken aus der UML. Frisco unterstützt diesen Satz von Beschreibungtechniken weitestgehend. Darüber hinaus sind sowohl die SysLab-Beschreibungstechniken als auch die SysLab-Methodik in ein Dokument-basiertes Konzept eingebettet, das darauf abzielt, durch systematische Kombination und Verfeinerung von Dokumenten im Entwurfsprozeß eine zunehmend genauere Systembeschreibung zu entwickeln. Dieses Konzept wird, wie oben beschrieben, durch den Dokument-basierten Ansatz von Frisco explizit unterstützt. Damit konnten wertvolle Erkenntnisse für die Werkzeug-basierte Softwareentwicklung aus dem Frisco-Prototyp gewonnen und als Basis für weitere Werkzeugplanungen genutzt werden.

4.11 Organisation wissenschaftlicher Treffen

Teilweise aufgrund der fachlichen Anerkennung der geleisteten Arbeit, teilweise aber auch durch die bei Fachgesprächen und durch die precise-UML-Gruppe geschlossenen Kontakte wurde es möglich, mehrere wissenschaftliche Workshops sowie eine international anerkannte Konferenz zu diesem Themengebiet zu organisieren. Dadurch gelang es, die Präsenz der erreichten Leistungen noch wesentlich zu verstärken. Im einzelnen waren dies folgende Treffen:

PSMT'98 Workshop on Precise Semantics for Software Modeling Techniques [BCMR98b, BCMR98a].

RTSE'97 Workshop on Requirements Targeting Software and Systems Engineering [BR98b, BR98a].

EACBS'98 Workshop on Engineering Automation for Computer Based Systems, Monterey, 1998.

Informatik und Kommunikationstechnik Zeitschriftenausgabe, Springer Verlag [BS98].

ECOOP'97 Workshop on Precise Semantics for Object-Oriented Modeling Techniques [KR97a, KR97b].

OOPSLA'97 Workshop Object-oriented Behavioral Semantics [KRS97b, KRS97a].

ECOOP'98 Workshop on Precise Behavioral Semantics (with an Emphasis on OO Business Specifications) [KR98a, KR98b].

OOPSLA'98 Workshop on Behavioral Semantics of OO Business and System Specifications [KRS98]. Die Beiträge wurden als Buch veröffentlicht [KRS99].

≪UML'99≫ 2nd International Conference on the UML [FR99].

GROOM GI Arbeitstreffen: Grundlagen Objektorientierter Modellierung [vdBBR98].

4.12 Lehre

Wie bereits oben erwähnt, wurde ein Großteil der in SysLab entwickelten Werkzeuge im Rahmen von Praktika und Diplomarbeiten realisiert. Dabei stand stets nicht nur das zu entwickelnde Produkt im Mittelpunkt, sondern vor allem auch das Einüben von Techniken der Spezifikation, Teamarbeit und Projektplanung. In [BBHP95, BH97] werden die Erfahrungen aus zwei Projektpraktika geschildert.

Zudem floß das in SysLab gewonnene Wissen über Software-Entwicklung und Systemmodellierung in eine Empfehlung für den Informatik-Unterricht an Gymnasien mit ein [HB96].

5 Abschließende Bewertung

Abschließend ist zu sagen, daß die Einrichtung des Leibnizpreises der Deutschen Forschungsgemeinschaft eine außerordentlich effektive und flexible Maßnahme zur Förderung der Forschung darstellt, die es den geförderten Forschern ermöglicht weit über das aktuelle Tagesgeschehen hinaus eine Vision zu entwerfen, in die Tat umzusetzen und ihr im nationalen und internationalen Wissenschaftsbetrieb Aufmerksamkeit zu verschaffen. Damit kann letztlich die Spitzenposition der deutschen Forschung erhalten und ausgebaut werden.

Das SysLab Team dank den Deutschen Forschungsgemeinschaft für die großzügige und flexible Förderung.

Literatur

[BBH+99] R. Breu, M. Broy, F. Huber, I. Krüger, G. Popp, B. Rumpe, and W. Schwerin. Anwendung von UML Softwareentwicklungstechniken: Eine Fallstudie. Technical Report TUM-I9729, Technische Universität München, 1999.

[BBH+00] R. Breu, M. Broy, F. Huber, I. Krüger, B. Rumpe, and W. Schwerin. *Applied Software Engineering Principles for UML*. to appear, 2000.

[BBHP95] W. Bartsch, K. Bergner, R. Hettler, and B. Paech. Studenten entwickeln Universelles Hochschulinformationssystem: Erfahrungen aus einem Softwaretechnik-Praktikum. *Software Engineering im Unterricht der Hochschulen '95, German Chapter of the ACM*, 1995.

[BCMR98a] M. Broy, D. Coleman, T. Maibaum, and B. Rumpe. PSMT – ICSE'98 Workshop on Precise Semantics for Software Modeling Techniques. In *Proceedings of International Conference on Software Engineerig (ICSE'98) Addendum*. IEEE Computer Society, 1998.

[BCMR98b] M. Broy, D. Coleman, T. Maibaum, and B. Rumpe. PSMT - Workshop on Precise Semantics for Software Modeling Techniques. Technical Report TUM-I9803, Technische Universität München, 1998.

[BDH+98] M. Broy, A. Deimel, J. Henn, K. Koskimies, F. Plasil, G. Pomberger, W. Pree, M. Stal, and C. Szyperski. What Characterizes a Software Component. *Software Concepts & Tools*, 19:49–56, 1998.

[BEP+99] M. Broy, H. Ehler, B. Paech, B. Rumpe, and V. Thurner. *Software Engineering: Schlüssel für Prozeßbeherrschung und Informationsmanagement*. TCW Report. Leitfaden für die Wirtschaft, 1999.

[BFG+93a] M. Broy, C. Facchi, R. Grosu, R. Hettler, H. Hußmann, D. Nazareth, F. Regensburger, O. Slotosch, and K. Stølen. The Requirement and Design Specification Language SPECTRUM, An Informal Introduction, Version 1.0, Part 1. Technical Report TUM-I9312, Technische Universität München, 1993.

[BFG+93b] M. Broy, C. Facchi, R. Grosu, R. Hettler, H. Hußmann, D. Nazareth, F. Regensburger, O. Slotosch, and K. Stølen. The Requirement and Design Specification Language SPECTRUM, An Informal Introduction, Version 1.0, Part 2. Technical Report TUM-I9312, Technische Universität München, 1993.

[BG99] R. Breu and R. Grosu. Relating Events, Messages and Methods of Multiple-Threaded Objects. *Journal of Object-Oriented Programming (to appear)*, 1999.

[BGH+97a] R. Breu, R. Grosu, Ch. Hofmann, F. Huber, I. Krüger, B. Rumpe, M. Schmidt, and W. Schwerin. Exemplary and Complete Object Interaction Descriptions. In H. Kilov, B. Rumpe, and I. Simmonds, editors, *Proceedings OOPSLA'97 Workshop on Object-oriented Behavioral Semantics.* TUM-I9737, 1997.

[BGH+97b] R. Breu, R. Grosu, F. Huber, B. Rumpe, and W. Schwerin. Towards a Precise Semantics for Object-Oriented Modeling Techniques. In H. Kilov and B. Rumpe, editors, *Proceedings ECOOP'97 Workshop on Precise Semantics for Object-Oriented Modeling Techniques.* TUM-I9725, 1997.

[BGH+97c] R. Breu, R. Grosu, F. Huber, B. Rumpe, and W. Schwerin. Towards a Precise Semantics for Object-Oriented Modeling Techniques. In J. Bosch and S. Mitchell, editors, *Object-Oriented Technology, ECOOP'97 Workshop Reader.* Springer Verlag, LNCS 1357, 1997.

[BGH+98] R. Breu, R. Grosu, F. Huber, B. Rumpe, and W. Schwerin. Systems, Views and Models of UML. In M. Schader and Axel Korthaus, editors, *The Unified Modeling Language, Technical Aspects and Applications.* Physica Verlag, Heidelberg, 1998.

[BGK98] M. Broy, R. Grosu, and I. Krüger. Deutsche Patentanmeldung, Aktenzeichen 19837871.8, 1998.

[BGR+99] K. Bergner, R. Grosu, A. Rausch, A. Schmidt, P. Scholz, and M. Broy. Focusing on Mobility. In Ralph H. Sprague and Jr., editors, *Proceedings of the Thirty-Second Annual Hawaii International Conference on System Sciences.* IEEE Computer Society, 1999.

[BH97] K. Bergner and F. Huber. Systems Development with Java: Experiences from a Practical Project Course in Software Engineering. In D. Budgen, G. Hoffnagle, and J. Trienekens, editors, *Proceedings Eighth International Workshop on Software Technology and Engineering Practice (STEP'97), pp. 382-389.* IEEE Computer Society, 1997.

[BHH+97a] R. Breu, Ursula Hinkel, Ch. Hofmann, C. Klein, B. Paech, B. Rumpe, and V. Thurner. Towards a Formalization of the Unified Modeling Language. In *Proceedings of ECOOP'97.* Springer Verlag, LNCS, 1997.

[BHH+97b] R. Breu, Ursula Hinkel, Ch. Hofmann, C. Klein, B. Paech, B. Rumpe, and V. Thurner. Towards a Formalization of the Unified Modeling Language. Technical Report TUM-I9726, Technische Universität München, 1997.

[BHKS97a] M. Broy, C. Hofmann, I. Krüger, and M. Schmidt. Using Extended Event Traces to Describe Communication in Software Architectures. In *Asia-Pacific Software Engineering Conference and International Computer Science Conference, Hong Kong.* IEEE Computer Society, 1997.

[BHKS97b] M. Broy, Ch. Hofmann, I. Krüger, and M. Schmidt. A Graphical Description Technique for Communication in Software Architectures. Technical Report TUM-I9705, Technische Universität München, 1997.

[BHP+98] M. Broy, F. Huber, B. Paech, B. Rumpe, and K. Spies. Software and System Modeling Based on a Unified Formal Semantics. In M. Broy and B. Rumpe, editors, *Requirements Targeting Software and Systems Engineering, International Workshop RTSE'97, LNCS 1526*. Springer, 1998.

[BHRS97] K. Bergner, F. Huber, A. Rausch, and M. Sihling. Component-Oriented Redesign of the CASE-Tool Auto-Focus. Technical Report TUM-I9752, Technische Universität München, 1997.

[BK98] M. Broy and I. Krüger. Interaction Interfaces – Towards a Scientific Foundation of a Methodological Usage of Message Sequence Charts. In *ICFEM'98*, 1998.

[BKR99] R. Breu, I. Krüger, and B. Rumpe. Applied Software Engineering Principles for UML, 1999.

[BR98a] M. Broy and B. Rumpe, editors. *Requirements Targeting Software and Systems Engineering*, volume 1526. International Workshop RTSE '97, Lecture Notes in Computer Science, 1998. pages 43-68.

[BR98b] M. Broy and B. Rumpe. RTSE'97 - Workshop on Requirements Targeting Software Engineering. Technical Report TUM-I9807, Technische Universität München, 1998.

[Bre99] R. Breu. Konzepte, Techniken und Methodik der objektorientierten Entwurfs - Ein integrierter Ansatz. In *Habilitationsschrift, Technische Universität München*, 1999.

[Bro94] M. Broy. Software - eine andauernde Herausforderung, 1994. Siemens Vortrag.

[Bro95a] M. Broy. Advanced Component Interface Specification. In T. Ito and A. Yonezawa, editors, *Theory and Practice of Parallel Programming - International Workshop TPPP'94*. Springer, 1995.

[Bro95b] M. Broy. Mathematical System Models as a Basis of Software Engineering. *Computer Science Today*, 1995.

[Bro95c] M. Broy. Mathematics of Software Engineering. In *Mathematics of Program Construction*. Springer Verlag, 1995.

[Bro96a] M. Broy. Algebraic Specification of Reactive Systems. In M. Nivat and M. Wirsing, editors, *Algebraic Methodology and Software Technology*. Springer, 1996.

[Bro96b] M. Broy. Formal Description Techniques - How Formal and Descriptive are they. In *FORTE IX*. Chapman & Hall, 1996.

[Bro96c] M. Broy. Mathematik des Software-Engineering. In I. Wegener, editor, *Highlights aus der Informatik*, pages 229–252. Springer Verlag, 1996.

[Bro96d] M. Broy. Perspectives of System Informatics. In B. Björner, M. Broy, and I. V. Potosin, editors, *Second International Andrei Ershov Memorial Conference*, Lecture Notes in Computer Science 1181, 1996.

[Bro96e] M. Broy. *Software in Technik und Automation*, itg-fachbericht Software Engineering - von der Wissenschaft zur Anwendung in der Technik, pages 19–30. VDE Verlag, Berlin-Offenbach, 1996.

[Bro96f] M. Broy. Towards a Mathematical Model of a Component and Its Use. In *Componentware Users Conference 1996, Munich, Proceedings.* SIGS Publications, 1996.

[Bro96g] M. Broy. Wettbewerbsvorteile durch fortgeschrittenen Softwaretechnik, 1996. Münchner Management Tage Vortrag.

[Bro97a] M. Broy. Interactive and Reactive Systems: States, Observations, Experiments and all that... In *Foundations of Computer Science*, volume 1337 of *Lecture Notes in Computer Science*, pages 279–286. Springer Verlag, 1997.

[Bro97b] M. Broy. Mathematical Methods in System and Software Engineering. In *Mathematical Methods in Program Development*, 1997.

[Bro97c] M. Broy. Schwerpunktthema: Formale Methoden in der Praxis. *it+ti-Informationstechnik und Technische Informatik*, 3, 1997.

[Bro97d] M. Broy. Semantic Concepts for Software Architectures. Technical Report TUM-19746, Technische Universität München, 1997.

[Bro97e] M. Broy. The Specification of System Components by State Transition Diagrams. Technical Report TUM-I9729, Technische Universität München, 1997.

[Bro98a] M. Broy. A Logical Basis for Modular Software and Systems Engineering. In B. Rovan, editor, *SOSFAM '98: Theory and Practice in Informatic*, volume 1521 of *Lecture Notes in Computer Science*, pages 19–35, 1998.

[Bro98b] M. Broy. Kultur in der Informationsgesellschaft - die digitale Revolution frißt ihre Eltern - Einschätzungen aus der Sicht eines Informatikers. In *Kultur in der Informationsgesellschaft*, pages 65–70, 1998.

[Bro98c] M. Broy. On the Meaning of Message Sequence Charts (Key Note). In Y. Lahav et al., editor, *Proceedings of the 1st Workshop of the SDL Forum Society Workshop on SDL & MSC*, volume I, pages 13–34, 1998.

[Bro98d] M. Broy. Pragmatic and Formal Specification of System Properties by Tables. Technical Report TUM-19802, Technische Universität München, 1998.

[BS98] M. Broy and O. Spanol, editors. *Informatik und Kommunikationstechnik.* Springer Verlag, 2. edition, 1998.

[CK97] J. Coldewey and I. Krüger. Form-Based User Interface - The Architectural Patterns. In F. Buschmann and D. Riehle, editors, *Proceedings of the 1997 European Pattern Languages of Programming Conference, Irsee, Germany.* Siemens Technical Report 120/SW1/FB, 1997.

[EBF+98] A. Evans, J-M. Bruel, R. France, K. Lano, and B. Rumpe. Making UML Precise. In *OOPSLA'98 Workshop on "Formalizing UML. Why and How?", Vancouver Canada*, 1998.

[EFLR98] A. Evans, R. France, K. Lano, and B. Rumpe. Developing the UML as a Formal Modelling Notation. In P.-A. Muller and J. Bézivin, editors, *≪UML'98≫ Beyond the notation. International Workshop Mulhouse France.* Ecole Superieure Mulhouse, Universite de Haute-Alsace, 1998.

[EKR99] A. Evans, S. Kent, and B. Rumpe. UML Semantics FAQ. In *Workshop Reader of ECOOP'99.* LNCS, Springer Verlang, to appear 1999.

[FELR98] R. France, A. Evans, K. Lano, and B. Rumpe. The UML as a formal modeling notation. *Computer Standards & Interfaces*, 19:325–334, 1998.

[FNDR98] M. Fuchs, D. Nazareth, D. Daniel, and B. Rumpe. BMW-ROOM An Object-Oriented Method for ASCET. In *SAE'98, Cobo Center (Detroit, Michigan, USA), February 23 - 26, 1998.* Society of Automotive Engineers, 1998.

[FR98] M. Fahrmair and B. Rumpe. Frisco STDA - Werkzeug zur methodischen Bearbeitung von Automaten. Technical Report TUM-I9815, Technische Universität München, 1998.

[FR99] R. France and B. Rumpe, editors. *≪UML'99≫ The Unified Modeling Language. Beyond the Standard. Proceedings of the 2nd International Conference on the UML.* Springer Verlag, LNCS 1723, 1999.

[GKR96] R. Grosu, C. Klein, and B. Rumpe. Enhancing the SysLab System Model with State. Technical Report TUM-I9631, Technische Universität München, 1996.

[GKRB96] R. Grosu, C. Klein, B. Rumpe, and M. Broy. State Transition Diagrams. Technical Report TUM-I9630, Technische Universität München, 1996.

[GKS99] R. Grosu, I. Krüger, and T. Stauner. Hybrid Sequence Charts. Technical Report TUM-I9914, Technische Universität München, 1999.

[GR95] R. Grosu and B. Rumpe. Concurrent Timed Port Automata. Technical Report TUM-I9533, Technische Universität München, 1995.

[GS95] R. Grosu and K. Stølen. A Denotational Model for Mobile Point-to-Point Dataflow Networks. Technical Report TUM-I9527, Technische Universität München, 1995.

[GS96a] R. Grosu and K. Stølen. A Model for Mobile Point-to-Point Data-flow Networks without Channel Sharing. In M. Wirsing, editor, *AMAST'96*. LNCS, 1996.

[GS96b] R. Grosu and K. Stølen. Specification of Dynamic Networks. In M. Haveraaen and O. Owe, editors, *Proceedings of the 8th Nordic Workshop on Programming Theory, Oslo, Norway*. University of Oslo, 1996.

[GS98a] R. Grosu and T. Stauner. Modular and Visual Specification of Hybrid Systems – An Introduction to HyCharts. Technical Report TUM-I9801, Technische Universität München, 1998.

[GS98b] R. Grosu and T. Stauner. Visual Description of Hybrid Systems. In *Workshop On Real Time Programming (WRTP'98)*. Elsevier Science Ltd., 1998.

[GSB98a] R. Grosu, T. Stauner, and M. Broy. A Modular Visual Model for Hybrid Systems. In *Formal Techniques in Real Time and Fault Tolerant Systems (FTRTFT'98)*. Springer-Verlag, 1998.

[GSB98b] R. Grosu, G. Stefanescu, and M. Broy. Visual Formalisms Revisited. In *Proceedings of the CSD'98*, Aizu-Wakamatsu City, Fukushima, 1998.

[HB96] P. Hubwieser and M. Broy. Der informationszentrierte Ansatz: Ein Vorschlag für eine zeitgemäße Form des Informatikunterrichts am Gymnasium. Technical Report TUM-19624, Technische Universität München, 1996.

[Het95] R. Hettler. *Entity/Relationship-Datenmodellierung in axiomatischen Spezifikationssprachen*. Dissertation, Technische Universität München, 1995.

[Het96] R. Hettler. Description Techniques for Data in the SysLab Method. Technical Report TUM-I9632, Technische Universität München, 1996.

[HHKR96] C. Hofmann, E. Horn, W. Keller, and K. Renzel. The Field of Software Architecture. Technical report, Technische Universität München, 1996.

[HHKR97] C. Hofmann, E. Horn, W. Keller, and K. Renzel. Approaches to Software Architecture. In M. Broy, E. Denert, K. Renzel, and M. Schmidt, editors, *Software Architectures and Design Patterns in Business Applications*. Technische Universität München, TUM-I9746, 1997.

[HMR+98] F. Huber, S. Molterer, A. Rausch, B. Schätz, M. Sihling, and O. Slotosch. Tool-supported Specification and Simulation of Distributed Systems. In B. Krämer, N. Uchihira, P. Croll, and S. Russo, editors, *Proceedings International Symposium on Software Engineering for Parallel and Distributed Systems, pp. 155-164, ISBN 0-8186-8467-4*. IEEE Computer Society, Los Alamitos, California, 1998.

[HMS+98] F. Huber, S. Molterer, B. Schätz, O. Slotosch, and A. Vilbig. Traffic Lights - An AutoFocus Case Study. In *1998 International Conference on Application of Concurrency to System Design, pp. 282-294*. IEEE Computer Society, 1998.

[HR97] C. Hofmann and K. Renzel. Beschreibungssprachen für Software-Architekturen. In M. Broy, E. Denert, K. Renzel, and M. Schmidt, editors, *Software Architectures and Design Patterns in Business Applications*. Technische Universität München, TUM-I9746, 1997.

[HRR98a] F. Huber, A. Rausch, and B. Rumpe. Component Interface Diagrams: Putting Components to Work. Technical Report TUM-I9831, Technische Universität München, 1998.

[HRR98b] F. Huber, A. Rausch, and B. Rumpe. Modeling Dynamic Component Interfaces. In M. Singh, B. Meyer, J. Gil, and R. Mitchell, editors, *TOOLS 26, Technology of Object-Oriented Languages and Systems, pages 58-70*. IEEE Computer Society, 1998.

[HRR99] F. Huber, O. Rabe, and B. Rumpe. Frisco Open Editor Framework (OEF). In *Erfahrungen mit Java – Projekte aus Industrie und Hochschule*, Heidelberg, 1999. dpunkt-Verlag.

[HS97] F. Huber and B. Schätz. Rapid Prototyping with AutoFocus. In A. Wolisz, I. Schieferdecker, and A. Rennoch, editors, *Formale Beschreibungstechniken für verteilte Systeme, GI/ITG Fachgespräch 1997, pp. 343-352*. GMD Verlag (St. Augustin), 1997.

[HS98] F. Huber and B. Schätz. Specification Modules for Methodical System Development. In H. König and P. Langendörfer, editors, *8. GI/ITG Fachgespräch Formale Beschreibungstechniken für verteilte Systeme*. Shaker Verlag, Aachen, 1998.

[HSE97] F. Huber, B. Schätz, and G. Einert. Consistent Graphical Specification of Distributed Systems. In J. Fitzgerald, C. B. Jones, and P. Lucas, editors, *FME '97: 4th International Symposium of Formal Methods Europe, Lecture Notes in Computer Science 1313, pp. 122 - 141*. Springer, 1997.

[HSS96] F. Huber, B. Schätz, and K. Spies. AutoFocus - Ein Werkzeugkonzept zur Beschreibung verteilter Systeme. In Ulrich Herzog, editor, *Formale Beschreibungstechniken für verteilte Systeme*. Universität Erlangen-Nürnberg, 1996.

[HSSS96] F. Huber, B. Schätz, A. Schmidt, and K. Spies. AutoFocus - A Tool for Distributed Systems Specification. In *Proceedings FTRTFT'96 - Formal Techniques in Real-Time and Fault-Tolerant Systems, P. 467-470*. Springer Verlag, LNCS 1135, 1996.

[KGSB99] I. Krüger, R. Grosu, P. Scholz, and M. Broy. From MSCs to Statecharts. In *DIPES'98*. Kluwer, 1999.

[KP98a] R. Kaschek and B. Paech. Grundlagen der Modellierung von Vorgängen. In S. Jablonski, M. Böhm, and W. Schulze, editors, *Workflow-Management: Entwicklung von Anwendungen und Systemen - Facetten einer neuen Technologie*, pages 33–46. dpunkt-Verlag, 1998.

[KP98b] R. Kaschek and B. Paech. Metamodellierung von Arbeitsabläufen. In S. Jablonski and M. Böhm und W. Schulze, editors, *Workflow-Management: Entwicklung von Anwendungen und Systemen - Facetten einer neuen Technologie*, pages 47–64. dpunkt-Verlag, 1998.

[KPR97] C. Klein, Ch. Prehofer, and B. Rumpe. Feature Specification and Refinement with State Transition Diagrams. In P. Dini, editor, *Fourth IEEE Workshop on Feature Interactions in Telecommunications Networks and Distributed Systems*. IOS-Press, 1997.

[KR97a] H. Kilov and B. Rumpe. ECOOP'97 Workshop on Precise Semantics for Object-Oriented Modeling Techniques. Technical Report TUM-I9725, Technische Universität München, 1997.

[KR97b] H. Kilov and B. Rumpe. Summary of ECOOP'97 Workshop on Precise Semantics of Object-Oriented Modeling Techniques. In J. Bosch and S. Mitchell, editors, *Object-Oriented Technology – ECOOP'97 Workshop Reader*. Springer Verlag Berlin, LNCS 1357, 1997.

[KR98a] H. Kilov and B. Rumpe. Second ECOOP Workshop on Precise Behavioral Semantics (with an Emphasis on OO Business Spcifications). Technical Report TUM-I9813, Technische Universität München, 1998.

[KR98b] H. Kilov and B. Rumpe. Second ECOOP Workshop on Precise Behavioral Semantics (with an Emphasis on OO Business Specifications). In J. Bosch and S. Demeyer, editors, *Object-Oriented Technology – ECOOP'98 Workshop Reader*. Springer Verlag Berlin, LNCS 1543, 1998.

[KRB96] C. Klein, B. Rumpe, and M. Broy. A stream-based mathematical model for distributed information processing systems - SysLab system model -. In E. Naijm and J.-B. Stefani, editors, *FMOODS'96, Formal Methods for Open Object-based Distributed Systems*. ENST France Telecom, 1996.

[KRS97a] H. Kilov, B. Rumpe, and I. Simmonds. Object-Oriented Behavioral Semantics - With an Emphasis on Semantics of Large OO Business Specifications. In *OOPSLA'97 Conference Addendum to the Proceedings*. ACM press, 1997.

[KRS97b] H. Kilov, B. Rumpe, and I. Simmonds. OOPSLA'97 Workshop on OO Behavioral Semantics. Technical Report TUM-I9737, Technische Universität München, 1997.

[KRS98] H. Kilov, B. Rumpe, and I. Simmonds. Seventh OOPSLA Workshop on Behavioral Semantics of OO Business and System Specifications. Technical Report TUM-I9820, Technische Universität München, 1998.

[KRS99] H. Kilov, B. Rumpe, and I. Simmonds, editors. *Behavioral Specificatons for Business Systems*. Kluwer Academic Publisher, 1999.

[Krü99a] I. Krüger. Towards the Methodical Usage of Message Sequence Charts. In K. Spies and B.Schätz, editors, *Formale Beschreibungstechniken für verteilte Systeme. FBT'99*, 9. GI/ITG Fachgespräch, pages 123–134. Herbert Utz Verlag, June 1999.

[Krü99b] I. Krüger. *Using MSCs for Design and Validation of Distributed Software Components*. PhD thesis, Technische Universität München, 1999. (in preparation).

[LHR99] Ch. Lesny, F. Huber, and B. Rumpe. Frisco F – Eine funktionale, logische und algebraische Spezifikationssprache. Technical Report TUM-I9901, Technische Universität München, 1999.

[Pae95] B. Paech. A Methodology Integrating Formal and Informal Software Development. In *ICSE'95, Workshop on Formal Methods Application in Software Engineering Practice*, 1995.

[Pae96a] B. Paech. Algebraic View Specification. In *AMAST96: Algebraic Methodology and Software Technology*, 1996.

[Pae96b] B. Paech. Formal User-Centered Requirements Engineering. In Chris Jason and Sara Jones, editors, *User-Centered Requirements Engineering Workshop: Integrating Methods from Software Engineering and Human-Computer Interaction*, 1996.

[Pae97] B. Paech. A Framework for Interaction Description with Roles. Technical Report TUM-I9731, Technische Universität München, 1997.

[PR94] B. Paech and B. Rumpe. A new Concept of Refinement used for Behaviour Modeling with Automata. In *FME'94, Formal Methods Europe, Symposium '94*. Springer, 1994.

[PR97a] B. Paech and B. Rumpe. State Based Service Description. In J. Derrick, editor, *Formal Methods for Open Object-based Distributed Systems*. Chapman-Hall, 1997.

[PR97b] B. Paech and B. Rumpe. Towards Formal System Development with Views. In *Proceedings of BCS FACS/EROS One Day Workshop: Making Objects-oriented Methods More Rigorous*. Dept. of Computing, Imperial College, 1997.

[PR97c] J. Philipps and B. Rumpe. Refinement of Information Flow Architectures. In M. Hinchey, editor, *ICFEM'97 Proceedings, Hiroshima, Japan*. IEEE CS Press, 1997.

[PR97d] J. Philipps and B. Rumpe. Stepwise Refinement of Data Flow Architectures. In M. Broy, E. Denert, K. Renzel, and M. Schmidt, editors, *Software Architectures and Design Patterns in Business Applications*. Technische Universität München, TUM-I9746, 1997.

[PR99] J. Philipps and B. Rumpe. Refinement of Pipe And Filter Architectures. In *Proceedings of the Formal Methods'99 Conference*. Springer Verlag, LNCS, to appear 1999.

[RK96] B. Rumpe and C. Klein. Automata Describing Object Behavior. In H. Kilov and W. Harvey, editors, *Specification of Behavioral Semantics in Object-Oriented Information Modeling, P. 265-286*. Kluwer Academic Publishers, 1996.

[RKB95] B. Rumpe, C. Klein, and M. Broy. Ein strombasiertes mathematisches Modell verteilter informationsverarbeitender Systeme -Syslab Systemmodell-. Technical Report TUM-I9510, Technische Universität München, 1995.

[RSRS99] B. Rumpe, M. Schoenmakers, A. Radermacher, and A. Schürr. UML + ROOM as a Standard ADL? In *ECOOP'99 Workshop Proceedings*, to appear 1999.

[RT98] B. Rumpe and V. Thurner. Refining Business Processes. In H. Kilov, B. Rumpe, and I. Simmonds, editors, *Second ECOOP Workshop on Precise Behavioral Semantics (with an Emphasis on OO Business Specifications)*. Techniche Universität München, TUM-I9820, 1998.

[Rum96] B. Rumpe. *Formale Methodik des Entwurfs verteilter objektorientierter Systeme*. Dissertation, Technische Universität München, 1996.

[Rum98a] B. Rumpe. A Note on Semantics (with an Emphasis on UML). In H. Kilov and B. Rumpe, editors, *Second ECOOP Workshop on Precise Behavioral Semantics*. Technische Universität München, TUM-I9813, 1998.

[Rum98b] B. Rumpe. Formale Methodik des Entwurfs verteilter objektorientierter Systeme. In *Ausgezeichnete Informatikdissertationen 1997*. B. G. Teubner Stuttgart, 1998.

[Rum98c] B. Rumpe. Graphic Formalisms. In H. Ehrig, G. Engels, F. Orejas, and M. Wirsing, editors, *Semi-Formal and Formal Specification Techniques for Software Systems*. Dagstuhl-Seminar-Report 218, 1998.

[SRS99a] T. Stauner, B. Rumpe, and P. Scholz. Hybrid System Model. Technical Report TUM-I9903, Technische Universität München, 1999.

[SRS99b] T. Stauner, B. Rumpe, and P. Scholz. Integrating Continuous and Discrete Information Flows. In *to appear*, 1999.

[Thu97a] V. Thurner. Business Process Modeling in Software Development. In J.-P. Tolvanen and A. Winter, editors, *CAiSE'97 Doctoral Consortium, Fachbericht Informatik 14/97*. Universität Koblenz-Landau, Institut für Informatik, 1997.

[Thu97b] V. Thurner. Making it Their Idea – A Case Study: Customer Participation and Communication in BPR. In N. Callaos, C. M. Khoong, and E. Cohen, editors, *SCI'97, Volume 2, pp. 112-119*. International Institute of Informatics and Systemics IIIS, 1997.

[Thu98] V. Thurner. A Formally Founded Description Technique for Business Processes. In B. Krämer, N. Uchihira, P. Croll, and S. Russo, editors, *Software Engineering for Parallel and Distributed Systems, PDSE98*, pages 254–261. IEEE Computer Society, 1998.

[vdBBR98] M. von der Beeck, R. Breu, and B. Rumpe. 4tes GROOM-Arbeitstreffen: Pragmatische und formale Techniken in der OO Modellierung. In *Softwaretechnik–Trends*. GI, 1998.