Informatik Forschung und Entwicklung manuscript No.
(will be inserted by the editor)

Methodik, Sprachen und Grundlagen des Software Engineering

Abschlußbericht des Forschungslabors SysLab (1.10.1994 – 30.9.1999)

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

Forschungslabor SysLab, Fakultät für Informatik, Technische Universität München, D-80290 München
(e-mail: {broy,breur,huberf,kruegeri,rumpe,schwerin}@in.tum.de)

(2001) 16: pg. 53-59

Finanziert durch Mittel der DFG aus dem Leibniz-Programm

1 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 eines Leibnizpreises in Höhe von 1.5 Millionen DM 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 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 Einsatz 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.

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, ty-

pischerweise 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.

Die breit gefächerte Aufgabenstellung verteilte sich auf drei Teilprojekte: *Methodik*, *Beschreibungstechniken* und *Werkzeuge* und war geprägt von starker und intensiver Einbindung vorhandener Ergebnisse anderer Projekte der Forschergruppe.

2 Ergebnisse

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

- Vorgehensmodell und Methodik,
- Grundlagen: Syntax und Semantik,
- Software-Architektur,
- Beschreibungstechniken, aufgegliedert in
  - Message Sequence Charts (MSC),
  - Statecharts und verwandte Automatenmodelle,
  - Prozeßdiagramme zur Geschäftsprozeß- und Workflow-Modellierung und
  - UML,
- Werkzeugunterstützung,
- Organisation wissenschaftlicher Treffen

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

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 zeitaufwendige 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 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. 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 darüber hinaus 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.

Grundlagen: Syntax und Semantik

Wesentliche Erkenntnisse zur Auswahl geeigneter Beschreibungstechniken, bzw. deren syntaktischen und semantischen Varianten, wurden durch Arbeiten zur Fundierung objektorientierter Konzepte gewonnen.

Zentrales Ergebnis hierbei ist die Definition eines mathematischen Systemmodells als kompakte Beschreibung aller wesentlichen Eigenschaften von Softwaresystemen. 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*. Des weiteren werden Eigenschaften der Objekthierarchie, Kommmunikationswege und -arten, sowie die Handhabung von Objektkapselung und Identität festgelegt. Insbesondere werden Aspekte der Nebenläufigkeit und Synchronisation definiert.

Die Definition des Systemmodells basiert auf der These strombasierter Komponenten, insbesondere auf der im Teilprojekt A6 des Sonderforschungsbereichs 342 entwickelten formalen Methode Focus.

Basierend auf dem Systemmodell wurde eine Reihe spezifischer Aspekte untersucht. So wurden eine Erweiterung für analoge Echtzeitsysteme definiert, Mobilitäts- und Modularitätsaspekte untersucht und eine Integration von Ereignis-, Nachrichten- und Methodenkonzept vorgenommen.

Die in SysLab entwickelte Technik der strikten Trennung von Syntax und Semantik sowie der expliziten Formulierung einer semantischen Domäne (d.h. des Systemmodells) bietet wesentliche Vorteile.

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.

Software-Architektur

Unter dem Begriff *Software-Architektur* verstehen wir 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 interaktions- und 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 Software-Architektur-Begriff präziser zu fassen und gezielt Eigenschaften guter Software-Architekturen herauszuarbeiten, wurden 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 anderen Projekten wurden die Beschreibungstechniken aus ROOM und der UML auf ihre Einsatzmöglichkeit als Standardbeschreibungssprache für Software-Architekturen untersucht. 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. Die in SysLab entwickelten Techniken zur schrittweise Verfeinerung von statischen Datenfluß-Architekturen 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, sowie Beschreibungstechniken und Vorgehensweisen zur Modellierung von Komponentenarchitekturen.

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 mehreren Arbeiten wurden Aspekte, wie die Definition geeigneter Verfeinerungs- und Kompositionsbegriffe für MSCs und die Verwendung von MSCs sowohl für kleine Interaktionsszenarien als auch für vollständige Interaktionsarchitekturen untersucht. Ein vollautomatisiertes Verfahren zur Konstruktion ablauffähiger Progamme aus MSC-Spezifikationen wurde zum Patent gemeldet. Das Deutsche Patentamt hat die Erteilung in Aussicht gestellt. Das Verfahren schließt 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 eine auf hybride Systeme zugeschnittene MSC-Variante entwickelt und ihre Integration in eine Entwurfsmethodik für diese Systemklasse vorbereitet.

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). 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 sowie eine Erweiterung von Automaten für diskrete und analoge Echtzeitsysteme definiert.

STDs besitzen unterschiedliche Einsatzgebiete:

- Ein STD kann verwendet werden, um den kompletten *Lifecycle* eines Objekts zu beschreiben. Dabei spiegeln Zustände i.w. Äquivalenzklassen des Datenzustandsraums (Attribute) des Objekts wider.
- 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*.

Für die verschiedenen Varianten von STDs wurden Regelsätze angegeben, die eine Verfeinerung sowie eine Verschmelzung (Verhaltens-Komposition) von STDs erlauben.

Workflow- und Geschäftsprozeß-Modellierung

Während die oben besprochenen Beschreibungstechniken konzeptionell vor allem für die Beschreibung von Software-Systemen entwickelt wurden, stellt die Workflow- und Geschäftsprozeß-Modellierung das System "Unternehmen" in den Vordergrund. Modelliert werden 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.

In einzelnen Arbeiten wird zudem auf den Aspekt schrittweisen Entwurfs eingegangen und ein semantisch fundierter Verfeinerungsansatz für Geschäftsprozeßbeschreibungen vorgestellt. Zudem wird ein Ansatz zur Modellierung von Vorgängen vorgeschlagen und die Technik der Metamodellierung für die Konzeption von Arbeitsabläufen verwendet.

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. Zudem wurde ein integrierter Ansatz entwickelt, der die UML Beschreibungstechniken auf eine einheitliche Basis stellt und ihre Rolle innerhalb der Methodik des anwendungsfallgesteuerten Entwurfs präzisiert.

Durch die mit unseren 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.

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.

In SysLab wurden 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 Werkzeuge werden im folgenden kurz beschrieben.

AutoFocus AutoFocus 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 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 1 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 auszuführen. Aus konsistenten Systemspezifikationen können Prototypen generiert werden und in einer Simulations- und Visualisierungsumgebung zur Ausführung gebracht werden.

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

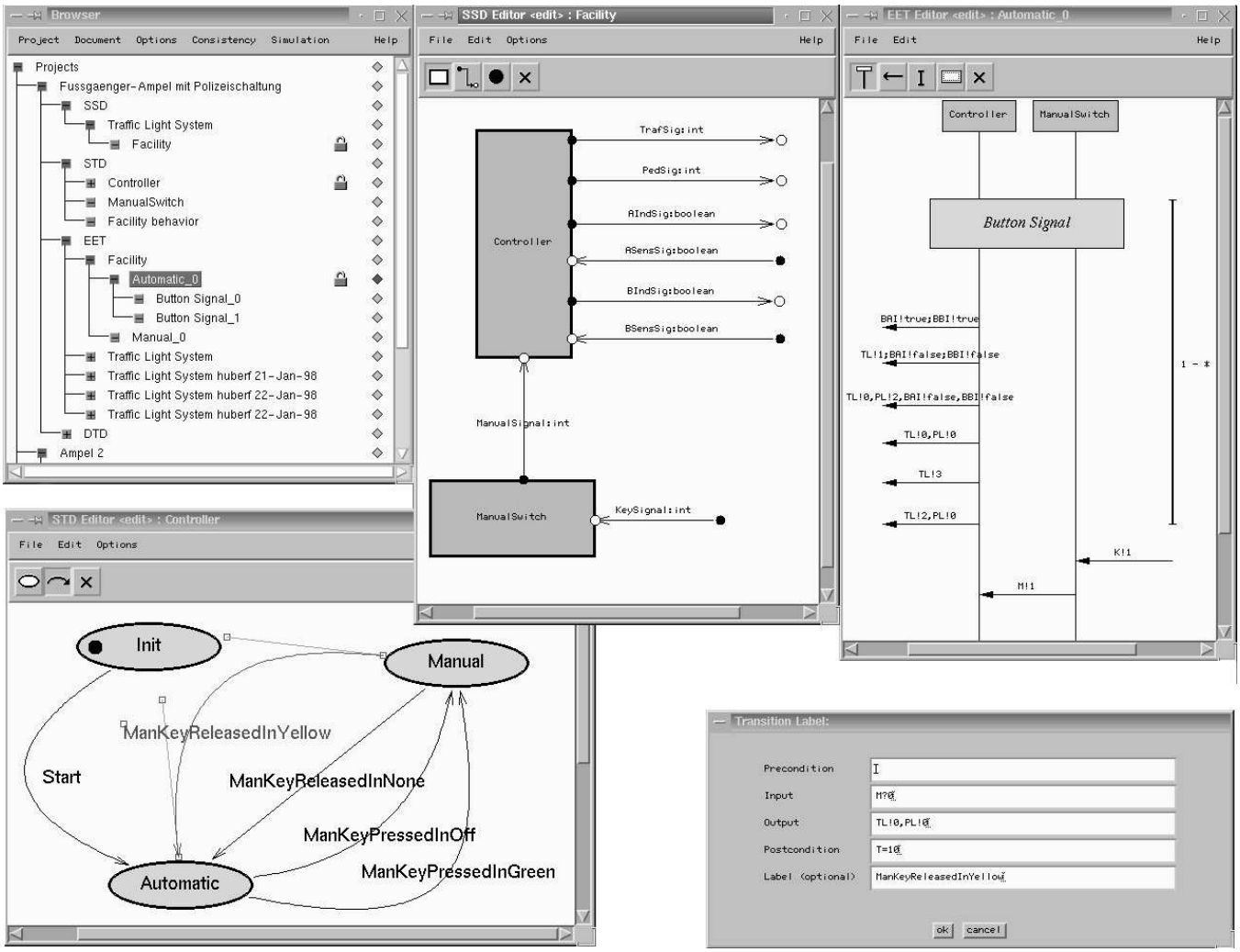

Abb. 1 AutoFocus Projekt-Browser und Editoren für SSDs, STDs und EETs

Im Zuge der Integration formaler und praxisorientierter Konzepte bietet AutoFocus insbesondere die typische Funktionalität von CASE -Werkzeugen: AutoFocus besitzt eine verteilte Client/Server-Architektur, 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 in Abbildung 1 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.

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

AutoFocus wurde zwischenzeitlich in mehreren Fallstudien eingesetzt (beispielsweise zur Spezifikation und Verifikation einer Ampelsteuerung) und an externe Partner und Unternehmen im Rahmen von Evaluierungslizenzen[1] herausgegeben.

Im Rahmen des Projektes Quest wurde AutoFocus 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 Entwicklungswerkzeugen aus Industrie und Forschung die beste Wertung und belegte den ersten Platz des Wettbewerbs.

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

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 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, die ebenfalls im Rahmen der Entwicklung der Frisco-Plattform realisiert wurde. Dabei wurde in einer speziellen Komponente ein Verfeinerungskalkül für Automaten realisiert und so die prinzipielle Machbarkeit demonstriert.

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.

Organisation wissenschaftlicher Treffen

Im Rahmen des SysLab-Projekts sowie durch die in der pUML-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.

RTSE'97 Workshop on Requirements Targeting Software and Systems Engineering.

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

Informatik und Kommunikationstechnik Zeitschriftenausgabe, Springer Verlag.

ECOOP'97 Workshop on Precise Semantics for Object-Oriented Modeling Techniques.

OOPSLA'97 Workshop Object-oriented Behavioral Semantics.

ECOOP'98 Workshop on Precise Behavioral Semantics (with an Emphasis on OO Business Specifications).

OOPSLA'98 Workshop on Behavioral Semantics of OO Business and System Specifications. Die Beiträge der genannten vier Workshops wurden in einem Buch aufgearbeitet.

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

GROOM GI Arbeitstreffen: Grundlagen Objektorientierter Modellierung.

3 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.

Danksagung Das SysLab Team dankt der Deutschen Forschungsgemeinschaft für die großzügige und flexible Förderung.

Literatur

Eine ausführliche Version des Abschlußberichtes mit den Referenzen auf die 124 im Rahmen vom SysLab entstandenen Veröffentlichungen ist verfügbar unter http://www4.informatik.tu-muenchen.de/reports/BBKRSOO.html.