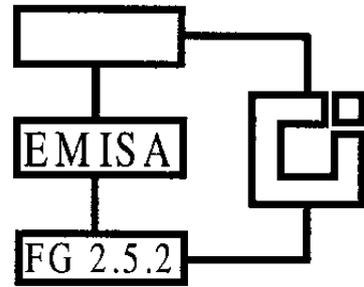


# EMISA FORUM



Mitteilungen der GI-Fachgruppe "Entwicklungsmethoden  
für Informationssysteme und deren Anwendung"

Heft 2

1997

## Inhalt:

|   |    |
|---|----|
| <b>Editorial</b> .....  | 1  |
| <b>Aus der Fachgruppe</b> .....   | 3  |
| <b>Fachbeiträge</b>   |    |
| J. Angele, R. Perkuhn, R. Studer, A. Oberweis, G. Zimmermann, F. Maurer,<br>B. Dellen, G. Pews, W. Stein:<br>Abschlußbericht der GI-Arbeitsgruppe "Vergleichende Analyse von<br>Problemstellungen und Lösungsansätzen in den Fachgebieten Information Systems<br>Engineering, Software Engineering und Knowledge Engineering" ..... | 11 |
| <b>Dissertationen</b> .....   | 59 |
| <b>Tagungsberichte</b> .....  | 64 |
| <b>Buchbesprechungen</b> .....  | 66 |
| <b>Ankündigungen</b> .....  | 68 |

# Impressum

Das Mitteilungsblatt der Fachgruppe 2.5.2 "Entwicklungsmethoden für Informationssysteme und deren Anwendung (EMISA)" im Fachausschuß 2.5 der Gesellschaft für Informatik (GI) erscheint zweimal jährlich, und zwar im Januar und im August. Es wird den Mitgliedern der Fachgruppe zugesandt. Mitglied kann jeder werden, der sich für die Zielsetzungen der Fachgruppe interessiert und sich mit einer stichwortartigen Angabe seiner Arbeitsgebiete anmeldet. Es wird ein jährlicher Mitgliedsbeitrag von 15 DM (Nicht-GI-Mitglieder 30 DM) erhoben.

Durch das Mitteilungsblatt sollen aktuelle Informationen unter den Mitgliedern der Fachgruppe ausgetauscht werden. Die Herausgeber bitten daher alle Leser, sich möglichst rege an der Gestaltung des Blattes zu beteiligen.

Es können für die Fachgruppe relevante Beiträge unter anderem zu folgenden Rubriken eingereicht werden:

- Fachbeiträge
- Berichte aus der Praxis
- Projektberichte (auch über laufende Projekte)
- Vorstellung von Arbeitsgruppen
- Buchbesprechungen
- Aktuelle Publikationen, Dissertationen, Habilitationen
- Tagungsberichte
- Call for Papers, Einladungen, Programme
- Leserbriefe, E-Mail-Zuschriften

Mit der Zusendung eines Beitrags an einen der Herausgeber (die Adressen werden auf der dritten Umschlagseite angegeben) ist das Einverständnis zur Veröffentlichung im Mitteilungsblatt verbunden. Jeder Beitrag wird ohne Begutachtung veröffentlicht (wenn er dem Themenspektrum der EMISA entspricht), er gibt nur die individuelle Meinung des Autors wieder. Die Beiträge können in deutscher oder englischer Sprache abgefaßt sein. Sie sollten keine Seitennumerierung enthalten. Dringende Mitteilungen können mittels E-Mail an die Redaktion ([oberweis@wiwi.uni-frankfurt.de](mailto:oberweis@wiwi.uni-frankfurt.de)) geschickt werden.

Redaktionsschluß für die Januar-Ausgabe ist der 5. Dezember, für die August-Ausgabe der 7. Juli (wenn nicht anders angekündigt).

**Herausgeber:** Leitungsgremium der Fachgruppe EMISA

**Auflage:** 2000

Ergänzend zum EMISA FORUM gibt es die Möglichkeit, unter World Wide Web Informationen mit Relevanz für die Fachgruppe bereitzustellen (<http://sunsite.informatik.rwth-aachen.de/Societies/GI-EMISA/>; Ansprechpartner: Manfred Jeusfeld, [jeusfeld@Informatik.RWTH-Aachen.de](mailto:jeusfeld@Informatik.RWTH-Aachen.de)).

Es gibt außerdem eine E-Mail-Liste EMISA MAIL, über die Mitteilungen von allgemeinem Interesse für EMISA-Mitglieder an die EMISA-Mitglieder, die sich in die Liste eingetragen haben, verteilt werden können. Insbesondere ist an Ankündigungen und Programme von Tagungen, Workshops und Arbeitsgruppen gedacht, aber auch an (Verweise auf) andere Beiträge, die in das EMISA FORUM passen würden.

E-Mail-Verteiler: [emisa@informatik.uni-hannover.de](mailto:emisa@informatik.uni-hannover.de)  
An/Abmeldung bitte an: [majordomo@informatik.uni-hannover.de](mailto:majordomo@informatik.uni-hannover.de)  
mit der Rumpfzeile: `[un]subscribe emisa [<E-Mail-Adresse>]`  
weitere Anfragen an: [emisa-owner@informatik.uni-hannover.de](mailto:emisa-owner@informatik.uni-hannover.de)

Ein Archiv mit den letzten Mails findet sich im WWW unter der URL: <http://www.informatik.uni-hannover.de/emisa>.

# EDITORIAL

Liebe EMISA-Mitglieder,

ich möchte mich in meinem vorläufig letzten Editorial als Fachgruppensprecher heute lediglich auf ein paar technische Hinweise beschränken.

Im kommenden Oktober wird unsere Fachgruppenleitung neu zu wählen sein, so daß die nächste Mitgliederversammlung von besonderer Bedeutung ist. Wir haben ganz bewußt das derzeit weithin Beachtung findende Thema „Workflows“ für unser diesjähriges Fachgruppentreffen gewählt, um Ihnen die Entscheidung, nach Darmstadt zu kommen und mit zu wählen, zu erleichtern. Also an dieser Stelle meine eindringliche Bitte: **Kommen Sie und geben Sie Ihre Stimme ab!** Wir haben auch diesmal aus Kostengründen auf die Organisation einer Briefwahl verzichtet, wie dies z.B. unsere Schwester, die Fachgruppe Datenbanksysteme, dieses Jahr tut. Ich habe nachfolgend die *vorläufige* Kandidatenliste zusammengestellt, die bis zur Durchführung der Wahl ergänzt oder abgeändert werden kann. Sollten Sie also Vorschläge hierzu oder auch selbst Interesse an einer Mitarbeit in der Fachgruppenleitung haben, melden Sie sich, spätestens in Darmstadt.

## **Vorläufige Kandidatenliste für die Wahl des EMISA-Leitungsgremiums:**

- Stefan Jablonski, Universität Erlangen-Nürnberg
- Heinrich Jasper, Hochschule Vechta
- Manfred Jeusfeld, Technische Hochschule Aachen
- Roland Kaschek, Universität Klagenfurt
- Josef Küng, Universität Linz
- Kurt Lautenbach, Universität Koblenz-Landau
- Andreas Oberweis, Universität Frankfurt
- Erich Ortner, Technische Hochschule Darmstadt
- Hansjürgen Paul, Institut Arbeit und Technik, Gelsenkirchen
- Helmut Thoma, Novartis, Basel
- Gottfried Vossen, Universität Münster

Ich werde alle Kandidaten bitten, in Darmstadt anwesend zu sein, um sich den Fachgruppenmitgliedern, jedenfalls den anwesenden, persönlich vorstellen zu können, denn ich erwarte nicht von Ihnen, daß Sie sich auf eine „Katze im Sack“ einlassen.

Im letzten Editorial hatte ich von den Plänen der Fachgruppenleitung hinsichtlich eines Brainstorming über unser Fachgebiet berichtet, das tatsächlich Ende Februar in Wettringen bei Münster stattgefunden hat und — jedenfalls aus meiner Sicht — sehr fruchtbar war. Allerdings ist es uns bisher nicht gelungen, das angekündigte Papier fertigzustellen; die kollaborative Arbeit einer zweistelligen Anzahl von Autoren ist doch schwieriger als angenommen. Wir werden das aber noch liefern.

Unangenehmer als das fehlende Papier ist die Mitteilung, daß ich von der im Herbst 1996 erfolgten Autorisierung durch die Mitgliederversammlung zur Erhöhung des Mitgliedsbeitrags Gebrauch machen muß: Der **Beitrag** der EMISA wird **ab 1998** von DM 15,- **auf DM 18,- erhöht**. Mir ist klar, daß dies wieder eine Reihe von Austritten zur Folge haben wird, aber die finanzielle Situation der GI läßt nach wie vor noch immer keinen uneingeschränkten Zugriff auf unsere Rücklagen aus Tagungseinnahmen zu. Schlimmer wirkt sich die Tatsache aus, daß die Druck-, vor allem aber die Portokosten für das vor Ihnen liegende Forum in letzter Zeit erheblich gestiegen sind und wir bereits in diesem Jahr draufzahlen. Am Ende einer Amtszeit gehört so etwas natürlich zu den unangenehmsten Entscheidungen, die man fällen kann.

Ansonsten danke ich Ihnen für das mir in den letzten drei Jahren entgegengebrachte Vertrauen, hoffe, viele von Ihnen in Darmstadt begrüßen zu können, und wünsche Ihnen einen erholsamen Sommer.

Münster, im Juli 1997

Gottfried Vossen

# **Einladung zur Mitgliederversammlung der GI-Fachgruppe EMISA**

Im Rahmen des Fachgruppentreffens der GI-Fachgruppe 2.5.2 EMISA an der TH Darmstadt (8. bis 10. Oktober 1997) findet die diesjährige Mitgliederversammlung 1997 der Fachgruppe statt. Zur Teilnahme sind alle Mitglieder der Fachgruppe eingeladen und aufgefordert.

**Ort:** TH Darmstadt, Zentrum für graphische Datenverarbeitung,  
Rundeturmstr. 6, 64283 Darmstadt

**Zeitpunkt:** 9. Oktober 1997, ca. 17.30 - 18.30 Uhr

## **Tagesordnung:**

1. Genehmigung der Tagesordnung und des Protokolls der Mitgliederversammlung 1996
2. Aktivitäten der EMISA
3. Mitgliederbestand und Finanzen
4. EMISA FORUM
5. Neuwahl des Leitungsgremiums
6. Verschiedenes

Das Leitungsgremium  
der GI-FG 2.5.2 EMISA

## **EMISA-Veranstaltungen und Veranstaltungen mit EMISA-Beteiligung:**

- **EMISA-Fachgruppentreffen 1997**  
**"Workflow-Management-Systeme im Spannungsfeld einer Organisation"**  
8. - 10. Oktober 1997, TH Darmstadt
- Treffen der GI-Arbeitsgruppe  
**"Petrinetze und Informationssysteme in der Praxis"**  
20. Oktober 1997, GMD Darmstadt
- GI-Workshop  
**"Modellierung '98"**  
12./13. März 1998, Universität Münster
- **EMISA-Workshop**  
**"Lebenszyklus großer Informationssysteme"**  
2. - 4. Juni 1998, Schloß Dagstuhl,  
Call for Papers: siehe <http://www.sunsite.informatik.rwth-aachen.de/Societies/GI-EMISA/>

# INHALTSÜBERSICHT

## EMISA-Fachgruppentreffen - Tutorien

*Andreas Oberweis, (J.W. Goethe-Universität Frankfurt)*  
Geschäftsprozeßmodellierung

*Stefan Jablonski (Friedrich-Alexander-Universität Erlangen-Nürnberg)*  
Workflow-Management-Systeme - Modellierung und Architektur

## EMISA-Fachgruppentreffen - Fachbeiträge

*M. Böhm (Technische Universität Dresden)*  
Einführung von Workflow-Management-Anwendungen durch schrittweise Entwicklung von Realisierungsalternativen für Geschäftsprozesse

*T. Goesmann, K. Just-Hahn, T. Löffeler, R. Rolles (FhG ISST Dortmund, Universität Dortmund, Universität des Saarlandes)*

Flexibilität als Ziel beim Einsatz von Workflow-Management-Systemen - Methoden zur Anpassung, Aushandlung und kontinuierlichen Verbesserung

*P. Heintl (Universität Erlangen-Nürnberg)*  
Einsetzbarkeit eines Workflow-Management-Systems in der Universitätsverwaltung

*M. Hoffmann, K.-U. Loser (Universität Dortmund)*  
Mitarbeiter-orientierte Modellierung und Planung von Geschäftsprozessen bei der Einführung von Workflow-Management

*S. Kirn, U. Kümmerling (Technische Universität Ilmenau)*  
Organisatorische Perspektiven beim Einsatz von Workflow-Management-Systemen

*R. Klamma, P. Peters, M. Jarke (Technische Hochschule Aachen)*  
Eine Untersuchung der DV-Unterstützung von Informations- und Arbeitsflüssen im Qualitätsmanagement bei kleinen und mittelständischen Unternehmen der Fertigungsindustrie

*E. Ortner (Technische Hochschule Darmstadt)*  
Brauchen wir für den Einsatz flexibler Workflow-Management-Systeme eine neue Gestaltungslehre der Arbeit?

*H. Paul (Institut Arbeit und Technik Gelsenkirchen)*  
Workflowmanagementsysteme und Organisationsgestaltung - Werkzeuge des Wandels?

*M. Rosemann, M. zur Mühlen (Universität Münster)*  
Modellierung der Aufbauorganisation in Workflow-Management-Systemen: Kritische Bestandsaufnahme und Gestaltungsvorschläge

*R. Siebert (Universität Stuttgart)*  
Anpassungsfähige Workflows zur Unterstützung unstrukturierter Vorgänge

*M. Weske (Universität Münster)*  
Überlegungen zur Flexibilisierung von Workflow-Management-Systemen

## Arbeitstreffen des Arbeitskreises „Modellierung und Ausführung von Workflows“

Das aktuelle Programm des Arbeitstreffens finden Sie unter der www-Adresse  
<http://wwbdb.inf.tu-dresden.de/~www/AK-Workflow/>

# INFORMATIONEN

## Veranstaltungsort

Das Fachgruppentreffen findet im Kernbereich der Technischen Hochschule Darmstadt am Zentrum für Graphische Datenverarbeitung (ZGDV), Rundeturmstr. 6, statt. Falls Sie mit dem PKW anreisen, folgen sie der Ausschilderung "Stadtmitte" bzw. "THD Stadtmitte", keinesfalls "THD Lichtwiese". Vom Hauptbahnhof können Sie die Buslinien D (Richtung Ostbahnhof) und F (Richtung Oberwaldhaus) bis zur Haltestelle Schloß (Ernst-Ludwig-Platz) benutzen. Weitere Informationen finden Sie unter <http://stadt.darmstadt.gmd.de/touris/index.html>.

Wegen einer Zimmerbestellung wenden Sie sich bitte an:

Verkehrsverein Darmstadt e.V.  
Luisenplatz 5  
64283 Darmstadt  
Tel.: 06151/133160 Fax: 06151/132075

## Organisation

Für Fragen hinsichtlich des EMISA Fachgruppentreffens bzw. des Arbeitstreffens des Arbeitskreises „Modellierung und Ausführung von Workflows“ wenden Sie sich bitte an die Organisatoren:

Prof. Dr. Stefan Jablonski  
Universität Erlangen-Nürnberg  
Lehrstuhl für Datenbanksysteme  
Martensstraße 3  
D-91058 Erlangen

Tel. [49] (9131) 32090  
Fax. [49] (9131) 85-7885  
e-mail: [jablonski@informatik.uni-erlangen.de](mailto:jablonski@informatik.uni-erlangen.de)  
<http://www6.informatik.uni-erlangen.de/sj.html>

Prof. Dr. Erich Ortner  
Technische Hochschule Darmstadt  
Fachgebiet Wirtschaftsinformatik 1  
Hochschulstraße 1  
D-64289 Darmstadt

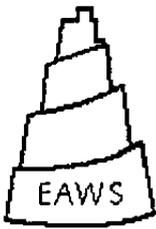
Tel. [49] (6151) 164309  
Fax. [49] (6151) 164301  
e-mail: [ortner@hrzpub.th-darmstadt.de](mailto:ortner@hrzpub.th-darmstadt.de)

## WWW-Informationen

Allgemeine Informationen zur GI-Fachgruppe EMISA  
<http://sunsite.informatik.rwth-aachen.de/Societies/GI-EMISA/>

Zusätzliche und aktuelle Informationen zum EMISA Fachgruppentreffen  
<http://www.bwl.th-darmstadt.de/bwl8/emisa97>

www-Seiten des Arbeitskreises „Modellierung und Ausführung von Workflows“  
<http://wwwdb.inf.tu-dresden.de/~www/AK-Workflow/>



# Anmeldung



Bitte senden Sie zur Anmeldung folgendes Formular bis spätestens 30.09.1997 an die angegebene Adresse. Die Teilnahme ist kostenlos!

Anmeldung per Email ist willkommen! Es werden nur E-Mail-Anmeldungen bestätigt.

---

An  
Prof. Dr. E. Ortner  
Technische Hochschule Darmstadt  
FG Wirtschaftsinformatik I  
64289 Darmstadt  
Fax: 06151-164301  
emisa97@bwl.th-darmstadt.de

## Anmeldung zum EMISA-Fachgruppen-Treffen 1997

Ihr Name: .....

Ihre Organisation: .....  
.....  
.....

Email oder FAX: .....

Teilnahme am gemeinsamen Abendessen am 09.10.\*): [ ]

GI-Mitglied : [ ]

EMISA-Mitglied : [ ]

Student/in: [ ]

Der Tagungsband wird auf dem Fachgruppentreffen verkauft.  
Interesse am Tagungsband (ca. 20 DM) [ ]

\*) Teilnahme auf private Kosten

Datum: .....

Unterschrift: .....

# **Petrinetze und Vorgehensmodelle zur Gestaltung von Informationssystemen**

- Petrinetze und Informationssysteme in der Praxis -

Peter Langner, INNOBIS GmbH

Am 03. März 1997 fand die siebente Tagung der Arbeitsgruppe "Petrinetze und Informationssysteme in der Praxis" zu dem Thema „Petrinetze und Vorgehensmodelle zur Gestaltung von Informationssystemen“ in Berlin statt. Es wurde von der Firma INNOBIS in Zusammenarbeit mit dem Fraunhofer-Institut für Software und Systemtechnik (FhG ISST) durchgeführt. Die interdisziplinäre Arbeitsgruppe trifft sich zweimal im Jahr zum Erfahrungsaustausch. Eingeladen hatten die Fachgruppen "Petrinetze und verwandte Systemmodelle" (FG 0.0.1) und "Entwicklungsmethoden für Informationssysteme und deren Anwendungen" (EMISA, FG 2.5.2) der Gesellschaft für Informatik (GI).

Im ersten Vortrag berichtete Herr Dr. Deiters vom FhG ISST Darmstadt von den Vorgehensweisen und Erfahrungen bei der petrinetz-basierten Modellierung im Rahmen seiner Projekte. In den Projekten werden zur Modellierung von Geschäftsprozessen die sogenannten FUNSOFT-Netze verwendet. Sie stellen eine Erweiterung der gefärbten Netze dar. Es wurde die „Sprache“ der Netze erweitert, ihre graphische Darstellung sowie ihre Verwendungsweise.

In einem zweiten Vortrag stellte Frau U. Petersen der UBIS GmbH das Werkzeug Bonapart vor. Bonapart dient zur Modellierung und Analyse von Geschäftsprozessen und basiert auf einem Prozeßmodell, das in ein Petrinetz übersetzt werden kann.

Im Anschluß an die Vorträge diskutierten die 19 Teilnehmer aus Universität und Wirtschaft in drei Kleingruppen u.a. die Themen Objektorientierung und Petrinetze sowie Simulation und Analyse auf der Basis von Petrinetzen.

Das nächste Treffen der Arbeitsgruppe findet am 20. Oktober 1997 von 9.30h bis 17.00h in Darmstadt statt. Das Treffen wird von der Gesellschaft für Mathematik und Datenverarbeitung (GMD) durchgeführt. Thema des Treffens ist

## **Modelle - Bindeglied zwischen Praxis und formalen Spezifikationen mit Petrinetzen**

Vorgesehen sind am Vormittag als Inputreferate zwei Vorträge mit den Themen

*Formale Spezifikation von Zeitaspekten*

H.J. Burkardt (GMD)

*Formale Spezifikation räumlich verteilter Systeme und ihre Simulation*

R. Prinoth (GMD)

Am Nachmittag findet ein Erfahrungsaustausch in Kleingruppen statt.

Zur Mitarbeit in der Arbeitsgruppe eingeladen sind alle, die Petrinetze im Kontext von Informationssystemen einsetzen oder einsetzen möchten, um z.B. Geschäftsprozesse zu analysieren, interaktive Systeme benutzerorientiert zu beschreiben, Informationssysteme zu modellieren und zu dokumentieren, organisatorische Abläufe zu gestalten, etc.

Zur Anmeldung für das nächste Treffen, für Rückfragen und für weitere Informationen wenden Sie sich bitte bis zum 26. September 1997 an:

Herrn Peter Langner  
INNOBIS Unternehmensberatung und Software GmbH  
Willhoop 7  
D-22453 Hamburg  
Tel. +49 (0) 40 55 487-0  
Fax. +49 (0) 40 55 487-499  
E-Mail: info@innobis.de  
Internet: <http://www.innobis.de>

## MODELLIERUNG '98

12./13. März 1998      Universität Münster

### Gemeinsamer Workshop der GI-Fachgruppen

Petri-Netze (0.0.1), Knowledge Engineering (KE, 1.5.1), Requirements Engineering (RE, 2.1.6)  
OO-Software-Entwicklung (OOSE, 2.1.9), Entwicklungsmethoden für Informationssysteme und deren  
Anwendungen (EMISA, 2.5.2), Vorgehensmodelle für die betriebliche Anwendungsentwicklung (5.1.1),  
Modellierung betrieblicher Informationssysteme (MOBIS, 5.2.1)

### Zielsetzung

Verschiedene Fachgruppen der Gesellschaft für Informatik befassen sich mit der Modellierung unterschiedlicher Aspekte von Software- bzw. Informationssystemen. Die dafür entwickelten Konzepte und Vorgehensweisen vorzustellen, miteinander zu vergleichen und ihre jeweiligen Stärken und Schwächen zu diskutieren ist der Sinn dieses Workshops. Dies wollen wir durch

- eingeladene Übersichtsvorträge aus den einzelnen Fachgruppen
- themenzentrierte Sitzungen mit konträren bzw. sich ergänzenden Kurzvorträgen
- viel Zeit für Diskussionen

erreichen.

### Eingeladene Übersichtsvorträge

Jörg Desel, Uni Karlsruhe:

FG Petrinetze

Gregor Engels, Uni Paderborn:

FG Objekt-Orientierte Software-Entwicklung

Matthias Jarke, RWTH Aachen:

FG Requirements Engineering

Heinrich C. Mayr, Uni Klagenfurt:

FG Entwicklungsmethoden für Informationssysteme und deren Anwendungen

Günther Müller-Luschnat, FAST e.V., München:

FG Vorgehensmodelle für die betriebliche Anwendungsentwicklung

Ulrich Reimer, Rentenanstalt/Swiss Life, Zürich:

FG Knowledge Engineering

Elmar J. Sinz, Uni Bamberg:

FG Modellierung betr. Informationssysteme

### Positionspapiere

Für die Teilnahme am Workshop ist die Einreichung eines Positionspapieres Voraussetzung. Beiträge sind gleichermaßen aus Industrie und Forschung erwünscht. Gemäß der Zielsetzung des Workshop sollten Positionspapier

papiere nicht rein fachgruppenspezifische Problemstellungen behandeln, sondern als Diskussionsgrundlage zu Querschnittsthemen dienen, wie etwa:

- Customizing
- Domänen- und Referenzmodelle
- Integration informeller/formaler Darstellungen
- Modellanimation und -simulation
- Modellierungsmetaphern und -paradigmen
- Modellierungsprozesse
- Nachvollziehbarkeit (Traceability)
- Strukturierungskonzepte (Abstraktion, Modularisierung, Sichten)
- Validierung und Verifikation von Modellen
- Verwendung von Beispielen und Szenarios
- Wiederverwendung

Positionspapiere von bis zu 2000 Wörtern sind in ASCII oder Postscript zu senden an

`modellierung98@informatik.rwth-aachen.de`

Die eingereichten Papiere werden vom Programmkomitee begutachtet. Basierend auf den Einreichungen werden Sitzungen zu verschiedenen Querschnittsthemen gebildet. Nach Möglichkeit werden die Beiträge so ausgewählt, daß konträre Standpunkte zu Wort kommen. Die akzeptierten Positionspapiere werden in Form eines technischen Berichtes vorab veröffentlicht.

### Termine

|                                  |               |
|----------------------------------|---------------|
| Deadline für Positionspapier:    | 1. Nov. 1997  |
| Benachrichtungen für Vorträge:   | 20. Dez. 1997 |
| Endfassung der Positionspapiere: | 24. Jan. 1998 |
| Verteilung der Positionspapiere: | 15. Feb. 1998 |

### Tagungsleitung

Klaus Pohl, RWTH Aachen  
Andy Schürr, RWTH Aachen  
Gottfried Vossen, Uni Münster

### Lokale Organisation

Gottfried Vossen, Uni Münster

### Programmkomitee

Gerhard Chroust, Uni Linz

Ulrich Frank, Uni Koblenz

Martin Glinz, Uni Zürich

Gerti Kappel, Uni Linz

Roland Kaschek, Uni Klagenfurt

Horst Lichter, ABB Heidelberg

Kurt Lautenbach, Uni Koblenz

Frank Maurer, Uni Kaiserslautern

Andreas Oberweis, Uni Frankfurt

Erich Ortner, Uni Darmstadt

Barbara Paech, TU München

Klaus Pohl, RWTH Aachen (Leitung)

Andy Schürr, RWTH Aachen (Leitung)

Rudi Studer, Uni Karlsruhe

Rüdiger Valk, Uni Hamburg

Heinz Züllighoven, Uni Hamburg

# **Abschlußbericht der GI-Arbeitsgruppe „Vergleichende Analyse von Problemstellungen und Lösungsansätzen in den Fachgebieten Information Systems Engineering, Software Engineering und Knowledge Engineering“**

Jürgen Angele, Rainer Perkuhn, Rudi Studer  
Universität Karlsruhe, Institut AIFB, 76128 Karlsruhe  
e-mail: {angele, perkuhn, studer}@aifb.uni-karlsruhe.de

Andreas Oberweis, Gabriele Zimmermann  
J.W. Goethe-Universität Frankfurt, Lehrstuhl für Wirtschaftsinformatik II,  
Postfach 111932, 60054 Frankfurt/M.  
e-mail: {oberweis, gzimmerman}@wiwi.uni-frankfurt.de

Frank Maurer, Barbara Dellen, Gerd Pews  
Universität Kaiserslautern, Fachbereich Informatik, AG Expertensysteme  
Postfach 3049, 67653 Kaiserslautern  
e-mail: {dellen, maurer, pews}@informatik.uni-kl.de

Wolfgang Stein  
DeTeMobil GmbH, Abt. K324 IS-Datenbanken, Postfach 300463, 53184 Bonn  
e-mail: wolfgang.stein@t-mobil.de

## **1. Einleitung**

Mit Fragen der methodischen Unterstützung des Entwicklungsprozesses von Softwaresystemen beschäftigen sich verschiedene Teildisziplinen der Informatik. Speziell sind hier das Software Engineering, das Information Systems Engineering und das Knowledge Engineering zu nennen. Während das Software Engineering insbesondere Beiträge zur Beschreibung des Entwicklungsprozesses durch Vorgehensmodelle und zur Beschreibung (nicht-)funktionaler Aspekte von Softwaresystemen geliefert hat, beschäftigte man sich im Information Systems Engineering zunächst primär mit der Modellierung statischer Aspekte von Informationssystemen durch semantische Datenmodelle. In den zurückliegenden Jahren gewannen dynamische Aspekte jedoch immer mehr an Bedeutung. Im Knowledge Engineering wurden ursprünglich hauptsächlich Fragen der methodischen Unterstützung der Wissenserhebung untersucht, in jüngster Zeit bekamen jedoch Methoden zur Wissensmodellierung und -wiederverwendung ein immer stärkeres Gewicht.

Bei einer näheren Betrachtung dieser Fachgebiete zeigt es sich, daß einerseits eine Vielzahl gemeinsamer Fragestellungen und Lösungsansätze existiert, andererseits aber auch sehr unterschiedliche Problemstellungen untersucht werden.

Um das gegenseitige Verständnis zwischen diesen verschiedenen Fachgebieten zu verbessern, wurden von mehreren GI-Fachgruppen gemeinsam in den Jahren 1990, 1992 und 1994 drei

Workshops „Informationssysteme und Künstliche Intelligenz“ veranstaltet (siehe [Kar90], [Stu92], [LuM94]). Auf diesen Workshops zeigte es sich jedoch, daß ein eingehendes Verstehen der jeweils anderen Disziplinen und ein fundiertes Herausarbeiten von Gemeinsamkeiten und Unterschieden im Rahmen von einzelnen, in zweijährigem Abstand stattfindenden Workshops mit jeweils wechselnden Teilnehmern nur schwer zu erreichen ist. Deshalb beschlossen die GI-Fachgruppen Knowledge Engineering (FG 1.5.1), Software Engineering (FG 2.1.1) und Entwicklungsmethoden für Informationssysteme und deren Anwendung (EMISA) (FG 2.5.2) 1992 die Einrichtung einer gemeinsamen Arbeitsgruppe. Ziel der Arbeitsgruppe sollte es sein, „Problemstellungen und Lösungsansätze in den Fachgebieten Information Systems Engineering, Software Engineering und Knowledge Engineering vergleichend zu analysieren, um so Gemeinsamkeiten und Unterschiede besser charakterisieren zu können.“

Die Arbeitsgruppe konstituierte sich im Rahmen des Workshops „Querbezüge des Knowledge Engineering zu Methoden des Software Engineering und der Entwicklung von Informationssystemen“ auf der 2. Deutschen Tagung Expertensysteme [AnS93]. Anfangs beteiligten sich zehn verschiedene Gruppen bzw. Einzelpersonen an der Arbeitsgruppe (ein Überblick ist in [ALO+93] zu finden. Zur Fokussierung der Arbeiten beschloß die Arbeitsgruppe, sich primär mit den Themen *Vorgehensmodelle* und *Methoden* zu beschäftigen. Unter einem *Vorgehensmodell* wurde dabei die „Festlegung der bei der Entwicklung eines Systems durchzuführenden Arbeitsschritte verstanden, ... Beziehungen zwischen den Arbeitsschritten sind ebenso festzulegen wie Anforderungen an die zu erzeugenden Ergebnisse.“ [ALO+93]. Als eine *Methode* wurde eine „systematische Handlungsvorschrift zur Lösung von Aufgaben einer bestimmten Art verstanden.“ [ALO+93]. Dementsprechend wurde in der Arbeitsgruppe der Begriff *Methodik* im Sinne von Methodensammlung verwendet.

Außerdem einigte man sich in der Arbeitsgruppe darauf, die Arbeiten anhand einer vergleichenden Fallstudie durchzuführen. In Abwandlung des oft verwendeten IFIP Beispiels [OSV82] wurde als Aufgabenstellung für die Fallstudie die Entwicklung eines (wissensbasierten) Systems zur Tagungsverwaltung ausgewählt.

Im Rahmen ihrer Arbeit organisierte die Arbeitsgruppe noch einen weiteren Workshop „Vorgehensmodelle und Methoden zur Entwicklung komplexer Softwaresysteme“, der auf der 18. Deutschen Jahrestagung für Künstliche Intelligenz durchgeführt wurde [KuS94].

Leider zeigte es sich in der laufenden Arbeit der Arbeitsgruppe, daß es insbesondere für Mitglieder aus der Wirtschaft sehr schwierig ist, sich über einen längeren Zeitraum aktiv an einer derartigen Arbeitsgruppe zu beteiligen. So blieben für die letzte Phase der Arbeitsgruppe nur noch vier Gruppen übrig, die auch in diesem Abschlußbericht vertreten sind. Von daher sollte klar sein, daß dieser Abschlußbericht keine alle Aspekte umfassende Analyse sein kann, sondern sich vielmehr auf Schlußfolgerungen beschränken muß, die auf Grund der analysierten Methodiken möglich sind. Gleichwohl beinhalten diese Methodiken aus Sicht der Autoren typische methodische Vorgehensweisen in den beteiligten Fachgebieten.

Um einen systematischen Vergleich der Methodiken zu ermöglichen, erarbeitete die Arbeitsgruppe einen Kriterienkatalog, mit dem charakteristische Eigenschaften einer Methodik erfaßt werden können [Kri97]. Dieser Kriterienkatalog wird nachfolgend verwendet, um jede der vier Methodiken detailliert zu charakterisieren.

## Literatur

- [ALO+93] J. Angele, D. Landes, A. Oberweis und R. Studer: Vorgehensmodelle und Methoden zur Systementwicklung. In: H. Reichel (ed.): *Informatik - Wirtschaft - Gesellschaft, Proc. 23. GI-Jahrestagung*, Dresden, September 1993, Informatik aktuell, Springer, Berlin, 1993.
- [AnS93] J. Angele und R. Studer (eds.): Arbeitsunterlagen zum Workshop *Querbezüge des Knowledge*

*Engineering zu Methoden des Software Engineering und der Entwicklung von Informationssystemen, 2. Deutsche Tagung Expertensysteme, Hamburg, Februar 1993.*

- [Kar90] D. Karagiannis (ed.): *Information Systems and Artificial Intelligence: Integration Aspects*. Lecture Notes in Computer Science 474, Springer, Berlin, 1990.
- [Kri97] *Kriterienkatalog für den Vergleich von Softwareentwicklungsmethodiken*.  
„ftp://aifbmozart.aifb.uni-karlsruhe.de/pub/mike/Fragebogen.ps.Z“
- [KuS94] J. Kunze und H. Stoyan (eds.): *KI-94 Workshops, 18. Deutsche Jahrestagung für Künstliche Intelligenz*. Saarbrücken, September 1994, Gesellschaft für Informatik e.V.
- [LuM94] K. von Luck und H. Marburger (eds.): *Management and Processing of Complex Data Structures - Proceedings of the 3rd Workshop Information Systems and Artificial Intelligence*. Lecture Notes in Computer Science 777, Springer, Berlin, 1994.
- [OSV82] T.W. Olle, H.G. Sol und A.A. Verrijn-Stuart (eds.): *Information Systems Design Methodologies: A Comparative Review*. North-Holland, 1982.
- [Stu92] R. Studer (ed.): *Informationssysteme und Künstliche Intelligenz - Modellierung*. Informatik Fachberichte 303, Springer, Berlin, 1992.

## **2. Modellbasiertes und Inkrementelles Knowledge Engineering: der MIKE-Ansatz**

(Jürgen Angele, Rainer Perkuhn und Rudi Studer)

Im folgenden werden die wesentlichen Prinzipien und Methoden des MIKE-Ansatzes (Modellbasiertes Inkrementelles Knowledge Engineering) beschrieben. MIKE bietet eine Sammlung von Methoden zur Erhebung, Interpretation, Formalisierung und Implementierung von Wissen zur Erstellung eines wissensbasierten Systems. Ziel von MIKE ist es, die Vorteile von Lebenszyklusmodellen, Prototyping und formalen Spezifikationsprachen in einem übergreifenden Ansatz für das Knowledge Engineering zu integrieren.

### **Einleitung**

In der „Software-Krise“ der sechziger Jahre zeigte sich, daß zur Erstellung großer und komplexer Systeme anstelle der bis dahin praktizierten Methoden bessere, „ingenieurmäßige“ Methoden erforderlich waren. Eine ähnliche Entwicklung zeichnete sich in den letzten Jahren auch bei wissensbasierten Systemen ab: die Vorgehensweise zur Entwicklung kleiner prototypischer Expertensysteme läßt sich nicht auf große Systeme übertragen, die längere Zeit benutzt und gewartet werden müssen. Für solche wissensbasierten Systeme sind wiederum methodisch fundierte Entwicklungsansätze notwendig.

Mit MIKE [AFS96] beschreiben wir in diesem Artikel einen solchen Ansatz. MIKE stellt eine integrierte Sammlung von Methoden zur Erhebung, Interpretation, Formalisierung und Implementierung von Wissen zur Entwicklung wissensbasierter Systeme dar. Für die Beschreibung der Resultate der Entwicklungsschritte bietet MIKE jeweils dafür geeignete Modelle und zugehörige Beschreibungssprachen an. Die einzelnen Entwicklungsschritte sind in ein Vorgehensmodell eingebettet, das besonderes Augenmerk auf die Integration von Prototyping richtet.

Im folgenden werden zunächst grundlegende Annahmen und Prinzipien von MIKE erläutert, bevor im zweiten Abschnitt die Methoden und Beschreibungssprachen vorgestellt werden. Die Anwendung der Methoden wird im dritten Kapitel anhand des Vorgehensmodells von MIKE aufgezeigt.

## 2.1 Prinzipien des Ansatzes

### 2.1.1 Modellbasiertes Knowledge Engineering

#### **Knowledge Engineering als Modellierungsprozeß**

Bei der Entwicklung wissensbasierter Systeme soll ein Modell entstehen, das in Bezug auf die gestellte Aufgabe ähnliche Lösungen wie ein Experte produziert, diesen aber nicht notwendigerweise simuliert. Ein großer Teil des erforderlichen Wissens ist dem Experten nicht bewußt, sondern in seinen Erfahrungen verborgen und deshalb nicht direkt durch Befragung zugänglich. Somit kann die Entwicklung eines solchen Systems nicht als bloßer Transfer von Wissen vom Experten in den Computer, sondern muß vielmehr als die Entwicklung eines neuen Modells durch den Knowledge Engineer angesehen werden (vgl. z.B. [JIS93]).

#### **Der Knowledge Engineer als Moderator**

Wegen des teilweise unbewußten Wissens wird in MIKE ein Knowledge Engineer als Moderator für den Modellierungsprozeß als notwendig erachtet. Ein Erstellen des Modells durch Eingabe von Fakten durch den Experten selbst ist nur für sehr einfache Anwendungsbereiche oder bei starker Unterstützung durch vordefinierte Shells möglich. Problematisch an diesen Shells ist bislang ihre Beschränkung auf einen Anwendungsbereich oder eine spezielle Problemlösungsmethode [Pup93].

### 2.1.2 Unterschiedliche Ebenen der Wissensrepräsentation

Zur Reduktion der Komplexität des Entwicklungsprozesses wird die Lücke zwischen dem Wissen des menschlichen Experten und dem endgültigen System in mehreren Schritten überbrückt (vgl. Abb. 2-1). Dadurch können in jedem Entwicklungsschritt bestimmte Aspekte unabhängig von anderen Aspekten betrachtet werden.

Eine abstrakte Beschreibung des Systems auf konzeptueller Ebene unabhängig von Realisierungsdetails erfolgt im *Modell der Expertise*. Eine solche Beschreibung entspricht der Systemspezifikation im Software Engineering. Sie kann zudem für Erklärungskomponenten und zur Wartung des endgültigen Systems herangezogen werden. Dadurch wird der Entwicklungsprozeß in zwei Teile geteilt: der erste Teil betrachtet nur konzeptuelle Aspekte, im zweiten Teil werden Entwurfs- und Implementierungsaspekte betrachtet.

#### **Beschreibung auf konzeptueller Ebene**

Das Modell der Expertise muß trotz seiner Unabhängigkeit von Realisierungsaspekten eindeutig und präzise beschrieben sein. Die Lücke zwischen der informalen Beschreibung des Wissens und dem Modell der Expertise muß also weiter unterteilt werden.

Interviewtranskripte, Auszüge aus Fachbüchern, Fallbeschreibungen usw. als Ergebnisse der Wissenserhebung werden im sog. *Erhebungsmodell* [Neu94] durch natürlichsprachliche Protokolle beschrieben, die die Grundlage für die Interpretation und Strukturierung des Wissens sind. Ziel der Interpretation ist es, relevante Konzepte sowie grundlegende Schritte des Experten bei der Problemlösung und ihre Abfolge zu identifizieren und miteinander in Bezug zu setzen. Das Resultat dieses Schrittes wird in MIKE im sog. *Strukturmodell* dargestellt [Neu94]. Konzepte und Aktivitäten werden dort mit Hilfe vordefinierter Knotentypen beschrieben und durch Kanten verbunden, die bestimmte Beziehungen zwischen Knoten ausdrücken. Während die Kanten eine formale Bedeutung besitzen, sind die Inhalte der Knoten nach wie vor informal beschrieben. Somit ist das Strukturmodell eine semiformale Repräsentation, die mit dem auf Hyperme-

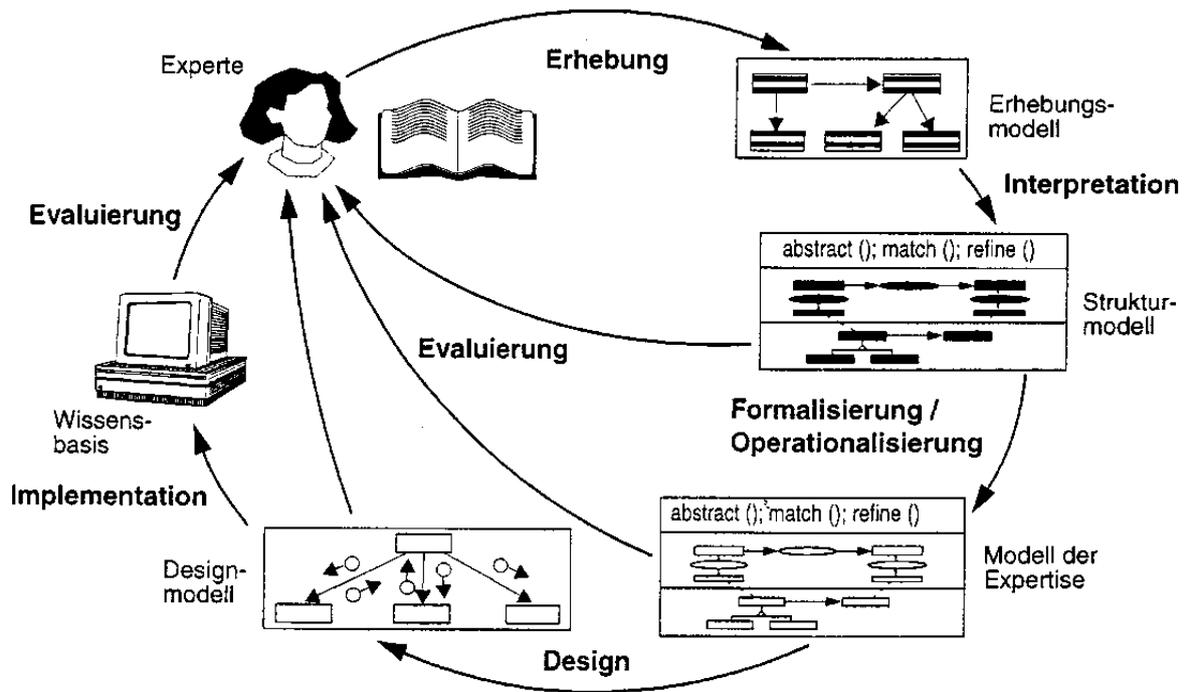


Abb. 2-1 Repräsentationsebenen in MIKE

diatechniken basierenden MIKE-Tool erstellt und modifiziert wird. Der Strukturierungsprozeß selbst gibt dem Knowledge Engineer und dem Experten eine frühe Rückkopplung. Die semiformale Repräsentation des Wissens stellt eine gute Kommunikationsbasis mit dem Experten dar. Der Inhalt des Strukturmodells kann zur Entwicklung der Erklärungskomponente des Expertensystems benutzt werden. Im Strukturmodell werden zusätzlich Modellierungsentscheidungen dokumentiert, die die spätere Wartung des Systems unterstützen.

Ausgehend von der semiformalen Repräsentation im Strukturmodell wird das Wissen dann formalisiert. Ergebnis des Formalisierungsschrittes ist das Modell der Expertise, zu dessen Beschreibung in MIKE die Sprache KARL ([FAS97], [Fen95]) entwickelt wurde. KARL basiert auf Hornlogik mit Negation, reichert diese aber um epistemologische Modellierungsprimitive an, die denen des Strukturmodells entsprechen. Somit ergibt sich ein bruchloser Übergang zwischen der semiformalen und der formalen Darstellung.

Die Eindeutigkeit und Präzision einer formalen Beschreibung kann dazu beitragen, einzelne Problemlösungsschritte besser zu verstehen. Zudem kann die formalisierte Problemlösungsmethode den weiteren Akquisitionsprozeß steuern und Grundlage für eine formale Verifikation des Modells sein. Eine formale Beschreibung läßt sich auf eine ausführbare Beschreibung abbilden, so daß die Erstellung und Evaluierung des Modells der Expertise durch Prototyping unterstützt werden kann. In MIKE ist kein expliziter Abbildungsschritt erforderlich, weil für KARL neben einer modelltheoretischen eine operationale Semantik definiert wurde und KARL damit direkt ausführbar ist [Ang93].

### Beschreibung auf realisierungsnaher Ebene

Im Modell der Expertise werden die funktionalen Anforderungen an das zu entwickelnde System beschrieben. Nichtfunktionale Anforderungen, wie z.B. Effizienz der Ausführung, Wartbarkeit etc., die die Realisierung des wissensbasierten Systems betreffen, werden erst in der Entwurfsphase im sog. *Designmodell* beschrieben und umgesetzt. Im Designmodell werden Designentscheidungen zu nichtfunktionalen Anforderungen in Bezug gesetzt und die Auswir-

kung von Entscheidungen zur Umsetzung dieser Anforderungen beschreiben. Das Designmodell schließt somit die Lücke zwischen dem Modell der Expertise und der endgültigen Implementierung des Systems [Lan95].

### **Explizite Querbezüge**

Die unterschiedlichen Modelle, in denen das Wissen repräsentiert wird, beschreiben entweder das gleiche Wissen auf eine andere Art oder enthalten zusätzliches Wissen, das aber in enger Verbindung zu Wissen in anderen Modellen steht. Um diese Bezüge für Dokumentation, Wartung und Erklärungsfähigkeit nutzen zu können, werden diese Querbezüge explizit repräsentiert.

#### **2.1.3 Wiederverwendung**

Um den Entwicklungsaufwand zu reduzieren und die Qualität des wissensbasierten Systems sicherzustellen, sollten Teile von Modellen wiederverwendbar sein. Im Modell der Expertise wird Problemlösungswissen unabhängig von konkretem Anwendungswissen beschrieben und kann deshalb für eine ähnliche Aufgabe in einem anderen Anwendungsbereich genutzt werden. Umgekehrt können Teile des anwendungsspezifischen Wissens für andere Aufgaben wiederverwendet werden. MIKE bietet hierzu eine Bibliothek sowohl semiformal als auch formal beschriebener Problemlösungsmethoden an.

## **2.2 Vorgehensmodell**

Für MIKE ist in [FAS97] und [Neu93] ein Prozeßmodell beschrieben, das den Knowledge Engineer im gesamten Entwicklungsprozeß leitet (vgl. Abb. 2-2). Das Prozeßmodell gibt bestimmte Aktivitäten und die Art der Abschlußdokumente vor; es beschreibt die Reihenfolge der Aktivitäten und die Kriterien zum Übergang in die nächste Aktivität. Der gesamte Knowledge Engineering Prozeß unterteilt sich in mehrere Phasen: die Wissensakquisitionsphase, die Entwurfsphase, die Implementierungsphase und die Evaluierungsphase. Diese Aktivitäten werden in dieser Reihenfolge zyklisch durchlaufen (vgl. [Boe88]). Jeder Durchlauf erzeugt einen Prototyp, der sich sukzessive an die gestellten Anforderungen annähert. Das endgültige System wird also durch evolutionäres Prototyping [Flo84] erstellt.

Die Wissensakquisition wird in die Aufgabenanalyse, die Modellbildung und die Modellevaluierung unterteilt. In der Aufgabenanalyse erfolgt die hierarchische Dekomposition der Aufgabe und die Beschreibung der Abhängigkeiten zwischen den Teilaufgaben. Zudem erfolgt die Auswahl der Teilaufgabe, die als nächstes angegangen werden soll. Die Modellbildung besteht aus Wissenserhebung, Interpretation, Formalisierung und Operationalisierung. In der Wissenserhebung erfolgt Erhebung von Wissen mittels Literatur und die Befragung des Experten. Das Ergebnis ist eine Menge von Protokollen und wird in MIKE im Erhebungsmodell repräsentiert. Im Interpretationsschritt erfolgt die Konzeptualisierung und Strukturierung des Wissens und dessen semiformale Beschreibung im Strukturmodell. In einem Formalisierungsschritt wird das Strukturmodell in das Modell der Expertise in KARL überführt. Das ausführbare Modell kann jetzt im Evaluierungsschritt gegenüber den Anforderungen des Experten und des Anwenders durch Testen anhand konkreter Fallbeispiele validiert werden. Die Schritte innerhalb der Wissensakquisition werden zyklisch durchlaufen, bis ein adäquates Modell entstanden ist. Die Entwicklung dieses Modells erfolgt also durch exploratives Prototyping [Flo84]. In der Entwurfsphase erfolgt zusätzlich die Erfassung, Beschreibung und Umsetzung nichtfunktionaler Anforderungen. Das Designmodell als Ergebnis dieses Schrittes wird mit DesignKARL beschrieben. Die Struktur des Modells der Expertise sollte im Designmodell weitgehend erhalten

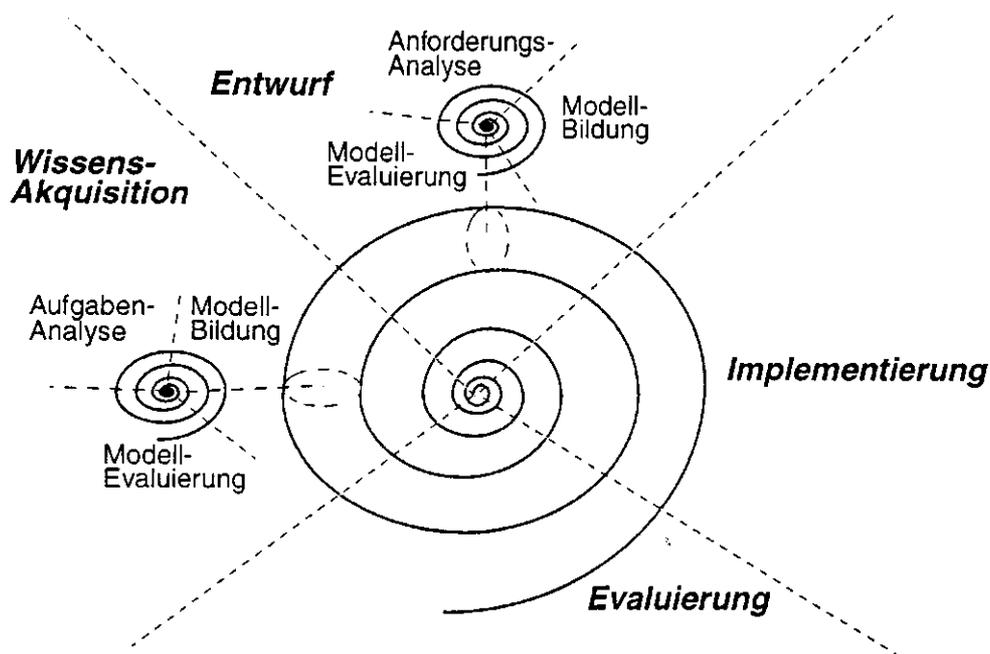


Abb. 2-2 Das Prozeßmodell von MIKE

werden. Auch die Entwurfsphase wird in eine Anforderungsanalyse, eine Modellbildungsphase und eine Evaluierungsphase unterteilt, die zyklisch durchlaufen werden, und beinhaltet die Möglichkeit der Evaluierung von Designentscheidungen durch experimentelles Prototyping.

Nach der Entwurfsphase erfolgt die Implementierung des Systems in einer geeigneten Zielumgebung.

## 2.3 Methoden, Beschreibungssprachen

MIKE bietet ein Spektrum von Modellen und Beschreibungssprachen mit unterschiedlichen Schwerpunkten. Das Erhebungs- und Strukturmodell repräsentiert Wissen in der frühen Akquisitionsphase informal oder semiformal. Die Sprache KARL wurde zur formalen und ausführbaren Beschreibung des Modells der Expertise entwickelt. Das Designmodell und DesignKARL integrieren nichtfunktionale Anforderungen und erleichtern den Übergang vom Modell der Expertise zum endgültigen System.

### 2.3.1 Das Erhebungs- und Strukturmodell

Das Strukturmodell [Neu93] beschreibt Wissen semiformal in Form von Konzepten und Aktivitäten und stellt den Datenfluß und den Kontrollfluß zwischen Aktivitäten dar. Eine solche semiformale Repräsentation erlaubt, den Experten in den Strukturierungsprozeß komplexen Wissens zu integrieren. Somit wird die Kooperation zwischen Experten und Knowledge Engineer verbessert und die Formalisierung vereinfacht.

Das Erhebungs- und Strukturmodell besteht aus speziellen Knoten- und Kantentypen. Ein Knoten stellt ein Hypermediadokument dar, das einen Zustand, einen Prozeß oder ein Konzept ggf. multimedial beschreibt. Eine Kante stellt eine bestimmte Beziehung zwischen zwei Knoten dar.

Das Erhebungsmodell repräsentiert das Ergebnis der Wissenserhebung und enthält informale Wissensprotokolle, die in sog. Protokollknoten gespeichert werden.

Das Strukturmodell wird aufbauend auf den Protokollen erstellt. Um über die unterschiedlichen Wissensrepräsentationen hinweg weitgehend die gleiche Strukturierung zu erhalten, bietet das

Strukturmodell Beschreibungselemente, die leicht auf entsprechende Modellierungsprimitive des Modells der Expertise in der formalen Sprache KARL abgebildet werden können:

- Die Knoten des Aktivitätenkontexts beschreiben jeweils einen Schritt des Problemlösungsprozesses. Aktivitäten können mit Verfeinerungskanten hierarchisch verfeinert werden. Der oberste Aktivitätenknoten stellt die gesamte Problemlösungsmethode dar.
- Reihenfolgekanten und Entscheidungsknoten beschreiben die zeitliche Abfolge der Aktivitäten im Reihenfolgekontext.
- Der Konzeptkontext beschreibt alle Konzepte. Konzeptknoten können durch is-a Kanten, part-of Kanten oder Kanten, die beliebige Beziehungen beschreiben, verbunden sein.
- Der Strukturkontext beschreibt den Datenfluß zwischen Aktivitäten einer Hierarchiestufe.

Das MIKE-Tool unterstützt die Entwicklung des Erhebungs- und Strukturmodells durch graphische Editoren.

### 2.3.2 Das Modell der Expertise in KARL

Das konzeptuelle Modell, auf dem KARL ([FAS97], [Fen95]) basiert, wurde aus dem KADS Modell der Expertise [SWB93] abgeleitet und unterscheidet mehrere Arten von Wissen (siehe Abb. 2-3). *Domänenwissen* besteht aus Wissen über den Anwendungsbereich des Systems. Zu dessen Beschreibung integriert KARL Frames und Logik in der Teilsprache L-KARL. Terminologisches Wissen wird durch eine Konzepttaxonomie beschrieben. Fakten und regelhaftes Wissen werden in logischen Formeln ausgedrückt, in denen auf Instanzen der Taxonomie, auf deren Attributwerte und auf die Taxonomie selbst Bezug genommen werden kann. In Abb. 2-3 werden z.B. die Klassen Sitzung und Paper aufgezeigt. Eine Sitzung beinhaltet dabei mehrere Papiere (Teil-Ganzes-Beziehung).

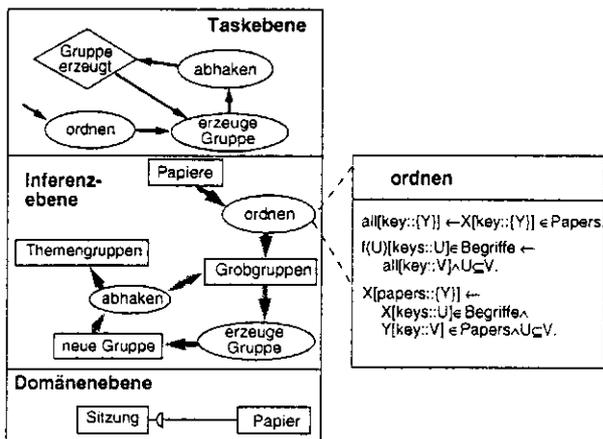


Abb. 2-3 Beispiel eines Teils des Modells der Expertise zur Tagungsorganisation

*Inferenzwissen* definiert sowohl die einzelnen Inferenzen (Inferenzaktionen), die während des Problemlösungsprozesses ausgeführt werden, als auch sog. Rollen, die Ein- und Ausgaben der Inferenzen enthalten. Es werden drei Arten von Rollen unterschieden: Views liefern Domänenwissen für eine Inferenzaktion, Stores modellieren den Datenfluß zwischen verschiedenen Inferenzaktionen, Terminatoren erlauben, Ergebnisse des Problemlösungsprozesses zurück auf die Domänenebene zu schreiben. In Abb. 2-3 erhält z.B. die Inferenzaktion *ordnen* ihre Eingabe aus dem Store *Papiere* und schreibt ihre Ausgaben in den Store *Grobgruppen*. L-KARL wird zur Spezifikation

der Inferenzaktionen, zur Definition einer generischen, problemlösungsmethodenspezifischen Terminologie innerhalb der Rollen und Inferenzaktionen und zur Abbildung zwischen dieser Terminologie und der anwendungsspezifischen Terminologie der Domänenebene benutzt.

Auf der Taskebene wird mit der Teilsprache P-KARL die Reihenfolge beschrieben, in der Inferenzaktionen auf der Inferenzebene aktiviert werden. P-KARL beinhaltet Schleife, Alternative und Sequenz als Kontrollkonstrukte. Wissen auf der Inferenz- und Taskebene ist

anwendungsunabhängig und beschreibt den Problemlösungsprozeß somit generisch. Abb. 2-3 zeigt eine Schleife, in der die Inferenzaktionen *erzeuge Gruppe* und *abhaken* ausgeführt werden.

### 2.3.3 Die Sprache KARL

KARL besteht aus den beiden Teilsprachen L-KARL und P-KARL. L-KARL dient zur Darstellung (statischen) strukturellen Wissens und wurde auf Basis von F-Logik [KLW95] entwickelt. P-KARL dient zur Darstellung dynamischen (prozeduralen) Wissens und wurde auf der Basis von dynamischer Logik entworfen.

F-Logik und L-KARL bieten im Vergleich zu Hornlogik mit Negation zusätzliche Modellierungsprimitive, die zur Darstellung vieler Sachverhalte geeigneter sind, ohne jedoch die modelltheoretische Semantik von Prädikatenlogik zu verlassen. L-KARL integriert Ideen aus semantischen und objektorientierten Datenmodellen in einen logischen Ansatz. Die Beschreibung der Inferenzaktion *ordnen* in Abb. 2-3 erfolgt durch mehrere L-KARL Regeln:

$$\text{all [key::\{Y\}]} \leftarrow X [\text{key::\{Y\}}] \in \text{Papers.}$$
$$f(U)[\text{keys::U}] \in \text{Begriffe} \leftarrow \text{all [key::V]} \wedge U \subseteq V.$$
$$X [\text{papers::\{Y\}}] \leftarrow X [\text{keys::U}] \in \text{Begriffe} \wedge Y [\text{key::V}] \in \text{Papers} \wedge U \subseteq V.$$

Die erste Regel sammelt alle Schlüsselwörter  $Y$  auf. Die zweite Regel bildet alle möglichen Teilmengen dieser Schlüsselwörter. Jede dieser Teilmengen bildet einen Begriff. Die letzte Regel ordnet jedem dieser Begriffe eine Menge von Papieren zu. Ein Papier wird einem Begriff zugeordnet, wenn die Menge der Schlüsselwörter des Papiers eine Obermenge der Menge der Wörter darstellt, die den Begriff charakterisieren.

Eine einfache Anweisung in P-KARL besteht aus dem Aufruf einer Inferenzaktion. An komplexen Anweisungen existieren zwei Schleifenvarianten, Verzweigung und Sequenz. Anweisungen können zu Subtasks zusammengefaßt werden.

Zur Erhöhung der Verständlichkeit und somit zur Unterstützung der Kommunikation mit dem Experten existieren für die meisten Modellierungsprimitive graphische Repräsentationen. Zur Darstellung der Domänenebene werden OMT-Diagramme [RBP+91] verwendet. Zur Beschreibung der Input-/Outputbeziehungen der Inferenzaktionen werden Datenflußdiagramme [You89] eingesetzt. Die Abfolge der Aktivierung von Inferenzaktionen wird durch Programmflußdiagramme beschrieben. Alle graphischen Repräsentationen sind hierarchisch verfeinerbar.

In [Fen95] ist die Semantik von KARL modelltheoretisch und in [Ang93] operational definiert. Basierend auf dieser Semantik und optimierten Auswertungsalgorithmen wurde für KARL ein Interpreter mit einem integrierten Debugger entwickelt, so daß eine KARL-Spezifikation durch Testen validiert werden kann.

### 2.3.4 Das Designmodell

Das Modell der Expertise konzentriert sich auf die konzeptuelle Beschreibung des zur Problemlösung erforderlichen Wissens sowie der verwendeten Problemlösungsmethode. Nichtfunktionale Aspekte wie Effizienz der Realisierung, Wartungsfreundlichkeit usw., aber auch Einschränkungen durch die intendierte Zielumgebung werden erst in der Entwurfsphase genauer betrachtet, da Designentscheidungen vorrangig auf solche Anforderungen zurückzuführen sind.

Um u.a. die spätere Wartung des wissensbasierten Systems zu erleichtern, ist es erforderlich, im

Designmodell nicht nur das Designprodukt als das Resultat des Designprozesses zu beschreiben, sondern auch die nichtfunktionalen Anforderungen und ihre Bezüge zu Designentscheidungen zu dokumentieren.

Zur Beschreibung von Datenstrukturen und Algorithmen wurde die Sprache KARL zu DesignKARL [Lan94] erweitert. DesignKARL bietet darüber hinaus zusätzliche Strukturierungspolymere wie Cluster und Module.

Zusätzlich ermöglicht DesignKARL den Designprozeß selbst zu beschreiben, indem Entwurfsentscheidungen dokumentiert und in Bezug zu den betroffenen Teilen des Designproduktes einerseits und den sie motivierenden nichtfunktionalen Anforderungen andererseits gesetzt werden können. Es lassen sich Interaktionen zwischen Designentscheidungen (z.B. gegenseitiger Ausschluß oder Implikation) sowie Anforderungen (z.B. positive oder negative Korrelation) beschreiben [LaS95].

Die Umsetzung in die Zielumgebung kann für ausgewählte Zielumgebungen durch ein Werkzeug unterstützt werden, das Teile des Designmodells automatisch auf Codestücke für die entsprechende Zielumgebung abbildet.

## 2.4 Schluß

In MIKE erfolgt der Übergang von den informalen Wissensprotokollen zum endgültigen System sanft in mehreren aufeinanderfolgenden Schritten, was die Komplexität jedes einzelnen Schrittes entscheidend verringert. Die informaleren Repräsentationen eignen sich gut für die Kommunikation mit dem Experten und zur Dokumentation. Die formalen Repräsentationen beschreiben Wissen präzise unter Abstraktion von Implementierungsdetails. Die Verwendung generischer Teile in Form von Problemlösungsmethoden kann den Entwicklungsprozeß verkürzen und den gesamten Akquisitionsprozeß steuern. Die einzelnen Phasen des Entwicklungsprozesses werden durch eine Reihe von Werkzeugen unterstützt.

MIKE wurde inzwischen in verschiedenen Fallstudien sehr erfolgreich eingesetzt. In [PFL+96] wird die Entwicklung eines Aufzugkonfigurationssystems beschrieben. Die Entwicklung eines Systems zur Unterstützung des Recyclings von Bauschutt wird in [FLS+95] beschrieben. Für die Sprache KARL wurde inzwischen eine Nachfolgerversion vorgeschlagen [ADP+96]. Eine vollständige graphische Repräsentation wird in Zukunft in das MIKE-Tool durch die Sprache G-KARL integriert [AnS97].

### Danksagung

Wir danken Dieter Fensel, Susanne Neubert und Dieter Landes, die viel zu den Ideen, die hier dargestellt wurden, beigetragen haben.

### Literatur

- [ABG+93] N. Aussenac, G. Boy, B. Gaines, M. Linster, J.-G. Ganascia, and Y. Kodratoff (eds.): *Knowledge Acquisition for Knowledge-Based Systems*. Lecture Notes in Artificial Intelligence 723, Springer, Berlin, 1993.
- [ADP+96] J. Angele, S. Decker, R. Perkuhn, and R. Studer: Modeling Problem-Solving Methods in New KARL. In: *Proceedings of the 10th Knowledge Acquisition for Knowledge-Based Systems Workshop (KAW'96)*, Banff, Kanada, November 1996.
- [AFS96] J. Angele, D. Fensel, and R. Studer: Domain and Task Modeling in MIKE. In: A. Sutcliffe, D. Benyon, F. van Assche (Eds.): *Domain Knowledge for Interactive System Design, Proceedings of IFIP WG 8.1/13.2 Joint Working Conference*, Geneva, May 1996, Chapman & Hall, 1996.
- [Ang93] J. Angele: *Operationalisierung des Modells der Expertise mit KARL*. Dissertationen zur Künstlichen Intelligenz, Band 53, infix, St. Augustin, 1993.

- [AnS97] J. Angele and R. Studer: G-KARL, A Graphical, Formal, and Executable Specification Language. In: P. Mertens, H.Voss (eds.): *Expertensysteme '97*, Infix - Verlag, St. Augustin, 1997.
- [Boe88] B.W. Boehm: A Spiral Model of Software Development and Enhancement. *ACM SIGSOFT 11(4)*, 1988, 21-42.
- [FAS97] D. Fensel, J. Angele, and R. Studer: The Knowledge Acquisition and Representation Language KARL. To appear in: *IEEE Transactions on Knowledge and Data Engineering*, 1997.
- [Fen95] D. Fensel: *The Knowledge Acquisition and Representation Language KARL*. Kluwer Academic Publisher, Boston, 1995.
- [Flo84] C. Floyd: A Systematic Look at Prototyping. In R. Budde et.al. (eds.): *Approaches to Prototyping*, Springer, Berlin, 1984, 1-18
- [FLS+95] W. Fichtner, D. Landes, Th. Spengler, M. Ruch, O. Rentz und R. Studer: Der MIKE Ansatz zur Modellierung von Expertenwissen im Umweltbereich - dargestellt am Beispiel des Bauschuttrecyclings. In: H. Kremers et al. (eds.): *Space and Time in Environmental Information Systems, Proc. 9th Int. Symposium on Computer Science for Environmental Protection*, Metropolis Verlag, Berlin, September 1995.
- [JIS93] Sonderheft „Knowledge Acquisition as Modelling“, Teil I und II, *International Journal of Intelligent Systems 8(1) / 8(2)*, 1993.
- [KLW95] M. Kifer, G. Lausen, and J. Wu: Logical Foundations of Object-Oriented and Frame-Based Languages. *Journal of the ACM*, vol. 42, 1995, 741-843.
- [Lan94] D. Landes: DesignKARL - A Language for the Design of Knowledge-based Systems. In: *Proc. 6th International Conference on Software Engineering and Knowledge Engineering (SEKE'94)*, Jurmala, Lettland, 1994, 78-85.
- [Lan95] D. Landes: *Die Entwurfsphase in MIKE - Methode und Beschreibungssprache*. Dissertationen zur Künstlichen Intelligenz, Band 84, infix, St. Augustin, 1995.
- [LaS95] D. Landes and R. Studer: The Treatment of Non-Functional Requirements in MIKE. In: *Proceedings of the 5th European Software Engineering Conference (ESEC '95)*, Sitges, 1995, Springer LNCS, Vol. 989, 1995.
- [Neu93] S. Neubert: Model Construction in MIKE (Model Based and Incremental Knowledge Engineering). In: [ABG+93], 200-219.
- [Neu94] S. Neubert: *Modellkonstruktion in MIKE - Methoden und Werkzeuge*. Dissertationen zur Künstlichen Intelligenz, Band 60, infix, St. Augustin, 1994.
- [PFL+96] K. Poeck, D. Fensel, D. Landes, and J. Angele: Combining KARL and CRLM for Designing Vertical Transportation Systems. *International Journal of Human-Computer Studies (IJHCS)*, 44, 1996, 435-467.
- [Pup93] F. Puppe: *Systematic Introduction to Expert Systems: Knowledge Representation and Problem-Solving Methods*. Springer Verlag, Berlin, 1993.
- [RBP+91] J. Rumbaugh, M. Blaha, W. Premerlani, F. Eddy, and W. Lorensen: *Object-Oriented Modeling and Design*. Prentice Hall, 1991.
- [SWB93] G. Schreiber, B. Wielinga, and J. Breuker (eds.): *KADS. A Principled Approach to Knowledge-Based System Development*. Academic Press, London, 1993.
- [You89] E. Yourdon: *Modern Structured Analysis*. Prentice-Hall. Englewood Cliffs, 1989.

### 3. CoMo-Kit - Modellierung und Operationalisierung von wissensintensiven Arbeitsabläufen

(Frank Maurer, Barbara Dellen, Gerhard Pews)

#### Einleitung

Knowledge Management, die Verwaltung des Wissens eines Unternehmens und die Entwicklung von effektiven Zugriffsmechanismen darauf, ist ein Schlüsselfaktor zum Erfolg im sich verschärfenden globalen Wettbewerb. Im CoMo-Kit Projekt der Universität Kaiserslautern werden Methoden, Techniken und Werkzeuge zur Modellierung von flexiblen und wissensintensiven Arbeitsabläufen und zu deren Abwicklungsunterstützung entwickelt. Das zur Ausführung einer Aufgabe benötigte Wissen kann, z.B. in Form von Hypertextdokumenten, in CoMo-Kit an diese gekoppelt werden und steht dann während der Aufgabenbearbeitung zur Verfü-

gung.

Im Rahmen dieses Beitrags konzentrieren wir uns auf die Beschreibung der Modellierungssprache und der -werkzeuge, den CoMo-Kit Modeler. Die Abwicklungskomponente, der CoMo-Kit Scheduler, ist eine hochgradig flexible Workflow-Engine. Die ihr zugrunde liegenden Techniken bilden einen aktuellen Schwerpunkt unserer Arbeiten und sind in [DMP97] und [MaP94] näher erläutert.

Eine Anwendung von CoMo-Kit, die im Rahmen des Sonderforschungsbereichs 501 „Entwicklung großer Systeme mit generischen Methoden“ verfolgt wird, ist die Modellierung und Abwicklung von Softwareentwicklungsprozessen. In [DMM+97] bzw. [VDM+96] wird dieser Ansatz und eine Integration von CoMo-Kit mit der Softwareprozeßmodellierungssprache MVP-L näher diskutiert. CoMo-Kit erlaubt die Beschreibung von SE-Vorgehensmodellen und der im Verlauf eines SE-Prozesses zu erzeugenden Produkte: Software Engineering ist in diesem Sinne ein spezieller wissensintensiver Arbeitsablauf und der CoMo-Kit Scheduler unterstützt dann sowohl die vertikale als auch die horizontale Verfolgbarkeit von Entscheidungen. Auf diese (Meta-)Ebene gehen wir im folgenden nicht ein.

### 3.1 Prinzipien des Ansatzes

Die im Rahmen des CoMo-Kit Projektes entwickelten Techniken lassen sich in Domänen anwenden, in denen flexible und wissensintensive Arbeitsprozesse unterstützt werden müssen. Im folgenden werden die Grundprinzipien des Ansatzes vorgestellt. In den nächsten Abschnitten gehen wir dann auf das Vorgehensmodell und die Modellierungssprache näher ein.

*Explizite Modellierung von Prozeß- & Produktmodellen:* Ausgehend von Knowledge- und Software Engineering-Ansätzen haben wir eine Sprache zur Beschreibung von wissensintensiven Arbeitsabläufen entwickelt. Diese erlaubt die Beschreibung von Prozeßmodellen (bestehend aus Prozessen, Methoden, Parametern), von Produktmodellen (zur Beschreibung von Datenstrukturen), konkreten Wissensseinheiten (in Form von Instanzen der Produktmodelle) und von ausführenden Agenten.

*Bereitstellung einer Workflowmaschine:* In wissenbasierten Systemen wird zwischen dem Domänenwissen (hier: Prozeßmodell) und dem Interpreter, der dieses Wissen nutzt, unterschieden. Wir haben uns diesen Ansatz zu Nutze gemacht und eine Workflowmaschine entwickelt, die Prozeßmodelle als Eingabe erhält und mit diesem Wissen die Abwicklung eines Projektes unterstützt und koordiniert. Dazu gehört die Verwaltung des aktuellen Projektzustands und der im Projektverlauf entstehenden Produkte, die Bereitstellung von Arbeitslisten für die Benutzer und die Weiterleitung wichtiger Information an die Benutzer. Der Ansatz und die entwickelten Werkzeuge können zum Prototyping und zur Simulation eingesetzt werden.

*Aufgabenorientierte Wissensorganisation:* An jede in dem Prozeßmodell enthaltene Aufgabe kann das zu ihrer Bearbeitung benötigte Wissen angekoppelt werden. In der Regel geschieht dies in CoMo-Kit durch die Zuordnung einer oder mehrerer WWW-Seiten, die im Verlauf der Systementwicklung in Zusammenarbeit mit Anwendungsexperten erstellt werden. Durch die Zuordnung der relevanten WWW-Seiten zu einer Aufgabe wird das Lost-in-Hyperspace-Problem reduziert: Der Aufgabenbearbeiter muß nicht erst nach relevanten Einsprungseiten ins World Wide Web suchen, sondern kann direkt auf die benötigten Informationen zugreifen.

### 3.2 Vorgehensmodell

Ausgangspunkt der Systementwicklung mit CoMo-Kit ist eine Menge von (natürlichsprachli-

chen) Protokollen, die die Ziele und Anforderung möglichst umfassend und genau beschreiben. Sie werden mit dem Protokolleditor erfaßt und im Repository des CoMo-Kit Modelers gespeichert. Der Entwickler kann nun in diesen Protokollen relevante Textteile selektieren und dann Prozeß- oder Produktmodelle erzeugen. Dabei wird automatisch ein Hyperlink aufgebaut und so Verfolgbarkeit zwischen den Texten und den Modellen hergestellt.

Die Reihenfolge der Erzeugung der einzelnen Teilmodelle ist nicht festgelegt, sondern kann in Abhängigkeit von den Gegebenheiten des aktuellen Projektes frei gewählt werden. Als Ergebnis werden Prozeß-, Produkt- und Ressourcenmodelle erzeugt, die in ihrer Gesamtheit einen Arbeitsablauf und das zu seiner Abwicklung benötigte Wissen beschreiben. Die Modellierungssprache ist im folgenden Abschnitt näher erläutert.

Nach Abschluß der Modellierung werden für alle im Produktmodell spezifizierten Klassen entsprechende Smalltalk-Klassen generiert, die dann während der Abwicklung von der Workflow-Engine instanziiert werden. Zusätzlich erzeugt das System noch Masken zum Editieren von Instanzen. Diese können mit Hilfe eines graphischen Interface-Builders überarbeitet werden.

Falls die Bearbeitung von einzelnen Teilaufgaben des Arbeitsprozesses automatisiert werden soll, sind entsprechende Programme zu entwickeln, die mit der Workflow-Engine über eine definierte Schnittstelle kommunizieren können und sich so in den Arbeitsablauf integrieren lassen. Für diesen Schritt enthält CoMo-Kit zur Zeit keine Werkzeugunterstützung.

CoMo-Kit Modelle werden durch die Workflow-Engine operationalisiert. Daher kann das CoMo-Kit System zum Prototyping von Geschäftsprozessen eingesetzt werden.

### 3.3 Sprache und Methoden des CoMo-Kit Modelers

Zur Beschreibung von wissensintensiven Arbeitsabläufen verwendet unser Ansatz vier verschiedene Grundkonzepte: Prozesse, Methoden, Produkte und Ressourcen. Diese werden wir im folgenden näher erläutern.

#### 3.3.1 Prozesse (oder auch: Aufgaben)

Ein Prozeß beschreibt eine Aktivität, die während der Durchführung eines Arbeitsablaufs bearbeitet werden muß. Die Beschreibung eines Prozesses besteht in CoMo-Kit aus mehreren Teilen:

*Ziel:* Das Ziel des Prozesses beschreibt explizit, was bei der Bearbeitung der Aufgabe erreicht werden soll. Es wird in textueller, informaler Form definiert und kann vom Aufgabebearbeiter während der Abwicklung gelesen werden.

*Parameter:* Ein Parameter dient als Platzhalter für eine Eingabe oder eine Ausgabe eines Prozesses. Eingaben werden zur Bearbeitung einer Aufgabe benutzt, können aber nicht verändert werden. Als Ergebnis der Aufgabebearbeitung werden die Ausgabeparameter mit Werten belegt. Für jeden Parameter eines Prozesses ist sein Wertebereich zu definieren. Ausgaben können als optional gekennzeichnet werden. Ausgabeparameter einer Aufgabe können die Eingaben anderer Aufgaben sein (→ Methoden).

*Hintergrundwissen:* Das Hintergrundwissen zu einem Prozeß besteht aus einer Liste von Referenzen zu Informationen, die zur Bearbeitung der Aufgabe nützlich sind (z.B. firmeninterne Standards, Gesetzestexte & -kommentare o.ä.). Das Hintergrundwissen wird als Konstante angesehen und nicht als Ausgabe eines anderen Prozesses erst erzeugt.

*Vorbedingung:* Die Vorbedingung eines Prozesses definiert, wann er gestartet werden kann. Sie wird in Form eines booleschen Ausdrucks angegeben und bezieht sich auf das Vorhan-

densein und/oder die Belegung der Eingabeparameter.

*Nachbedingung:* Die Nachbedingung wird in Form eines booleschen Ausdrucks definiert. Dieser wird *nach* der Bearbeitung der Aufgabe evaluiert und prüft, ob die Bearbeitung der Aufgabe erfolgreich war (d.h. ob die Ausgaben der Nachbedingung genügen).

*Anforderungen an Aufgabenbearbeiter:* Für jeden Prozeß wird eine Liste von Kriterien definiert, die ein Bearbeiter erfüllen muß, um den Prozeß durchführen zu können. Beispielsweise kann eine Aufgabe erfordern, daß der Bearbeiter Mitglied des Programmkomitees ist (→ Agenten).

*Liste von Methoden:* Jedem Prozeß ist eine Liste von → Methoden zugeordnet.

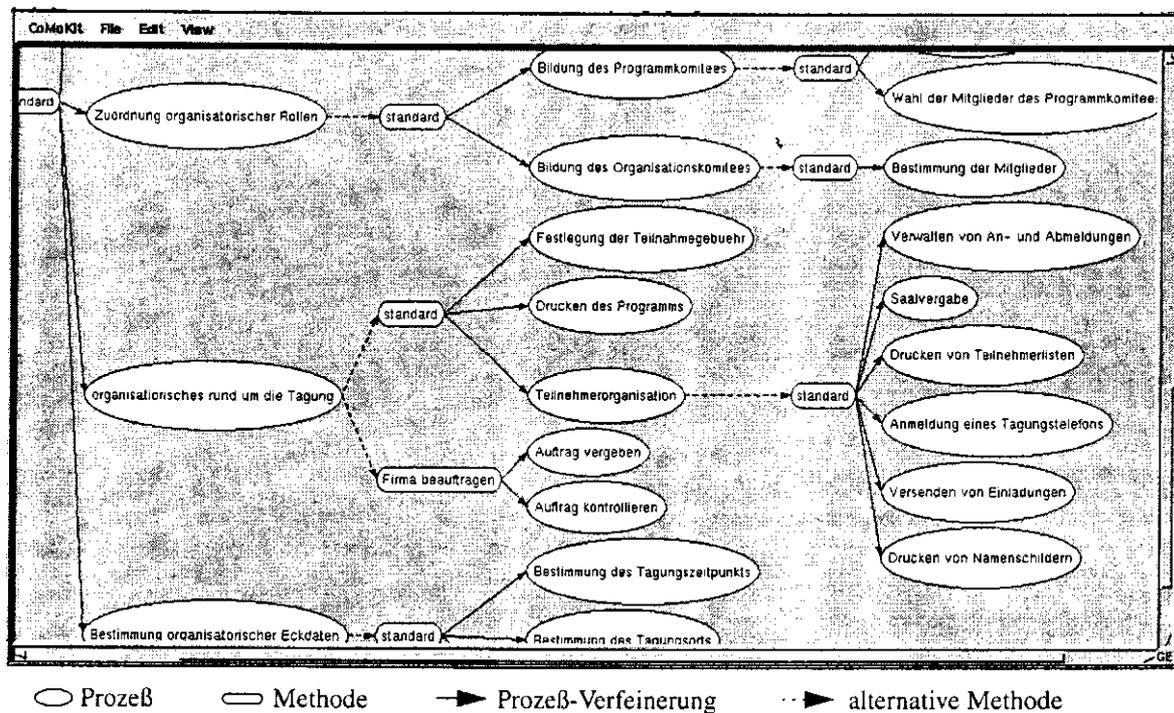


Abb. 3-1 Prozeßverfeinerung

### 3.3.2 Methoden

Eine Methode beschreibt, *wie* das Ziel eines Prozesses erreicht werden kann, ein Prozeß definiert ausschließlich, *was* das Ziel ist. CoMo-Kit erlaubt die Definition von mehreren alternativen Methoden für eine Aufgabe, d.h. es ist möglich, verschiedene Wege zur Erreichung eines Ziels zu beschreiben und dann bei der Abwicklung einen geeigneten auszuwählen.

Methoden werden durch → Agenten ausgeführt.

Wir unterscheiden zwischen atomaren und komplexen Methoden:

*Atomare Methoden* belegen die Ausgabeparameter des Prozesses direkt. Wird eine atomare Methode textuell (Process script) beschrieben, dann kann sie nur von einem menschlichen Bearbeiter ausgeführt werden. Eine zukünftige Erweiterung von CoMo-Kit wird process programs umfassen, die in einer formalen Sprache spezifiziert werden und dann direkt von einem Computer ausgeführt werden können.

Eine *komplexe Methode* zerlegt einen Prozeß in mehrere, einfacher zu lösende Teilprozesse. Desweiteren spezifiziert sie die Abbildung der Ausgabeparameter von Teilprozessen auf die Eingabeparameter anderer Teilprozesse und beschreibt so den (horizontalen) Informationsfluß zwischen Aufgaben. Zusätzlich bildet sie die Ein- und Ausgabeparameter des Oberprozesses

auf Ein- und Ausgabeparameter der Teilprozesse ab und definiert so einen vertikalen Informationsfluß.

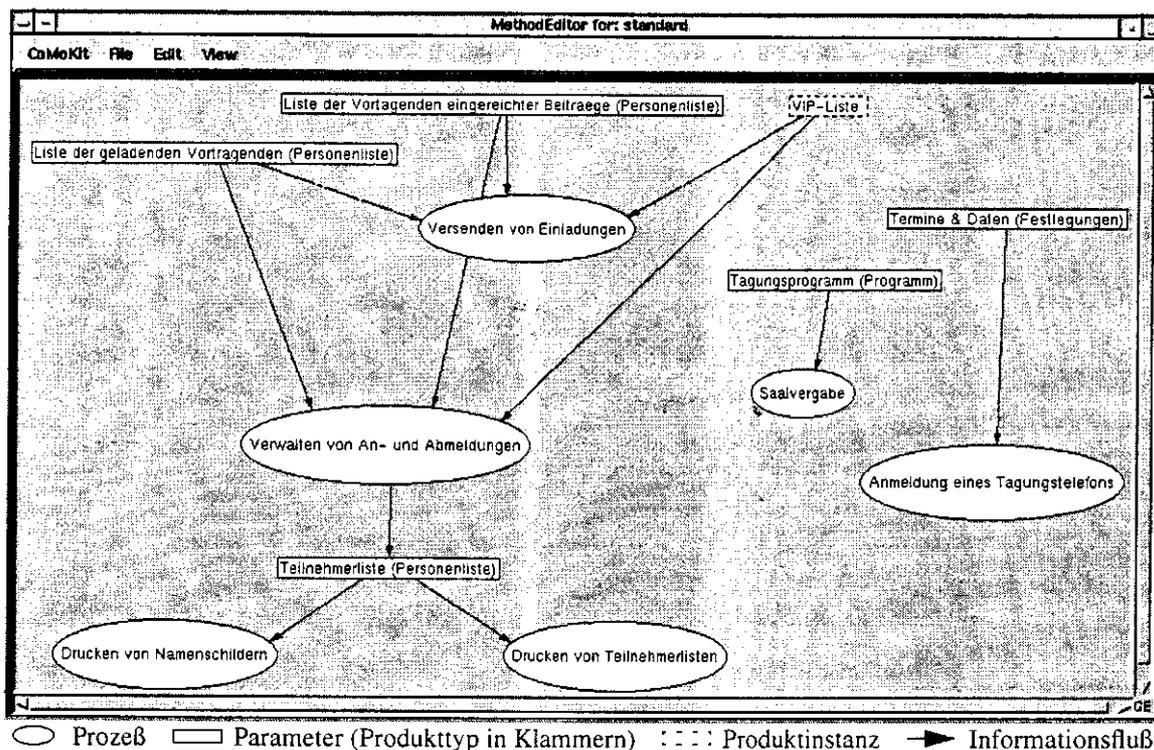


Abb. 3-2 horizontaler Informationsfluß-Editor

### 3.3.3 Produkte

Zur Modellierung der Produktdaten folgt CoMo-Kit einem objektzentrierten Ansatz. Wie üblich unterscheiden wir zwischen Produkttypen und Produktinstanzen (kurz: Produkten). Jeder Produkttyp definiert eine Menge von Slots, denen jeweils ein Typ zugeordnet ist. Als Typ ist ein Basistyp (z.B. STRING, REAL, SET, ...) oder ein bereits definierter Produkttyp zugelassen. Letzteres erlaubt die Spezifikation beliebig komplexer Objektstrukturen.

### 3.3.4 Ressourcen: Agenten & Werkzeuge

Ressourcen werden zur Abwicklung von Prozessen benötigt. *Agenten* sind aktive Einheiten, die (passive) *Werkzeuge* zur Durchführung ihrer Aufgaben benutzen. Wir unterscheiden zwei Arten von Agenten: *Aktoren* (auch: *Bearbeiter*) sind menschliche Agenten, die Aufgaben erledigen. *Process programs* werden von *Computeragenten* abgewickelt.

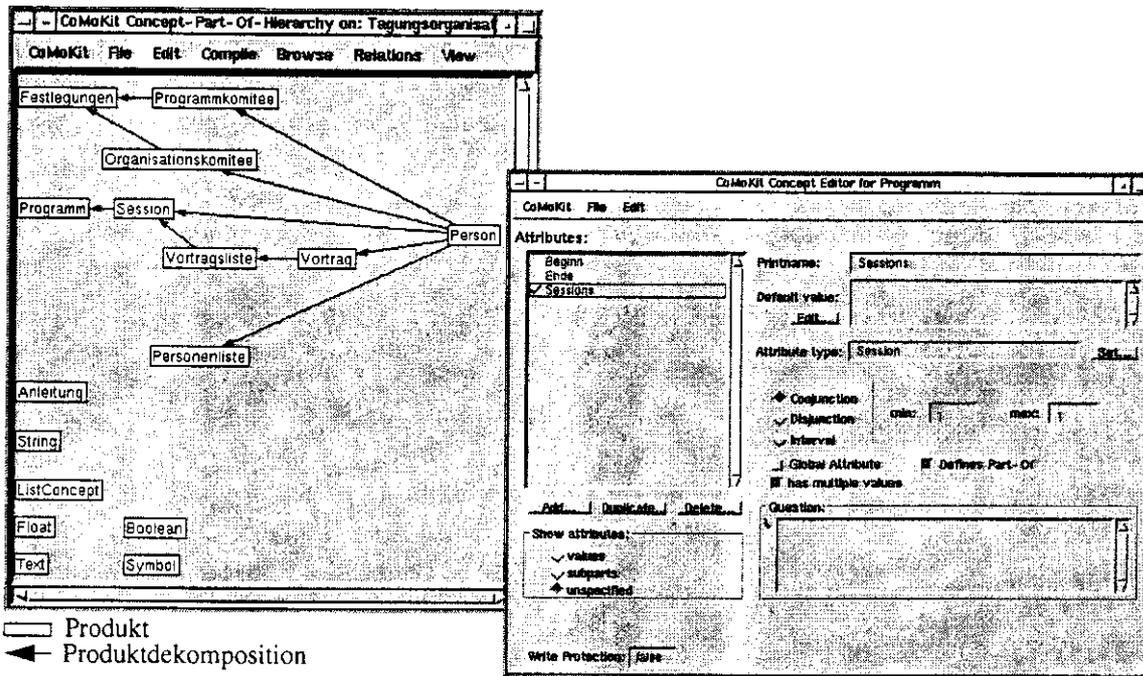


Abb. 3-3 Produkttyp-Editoren

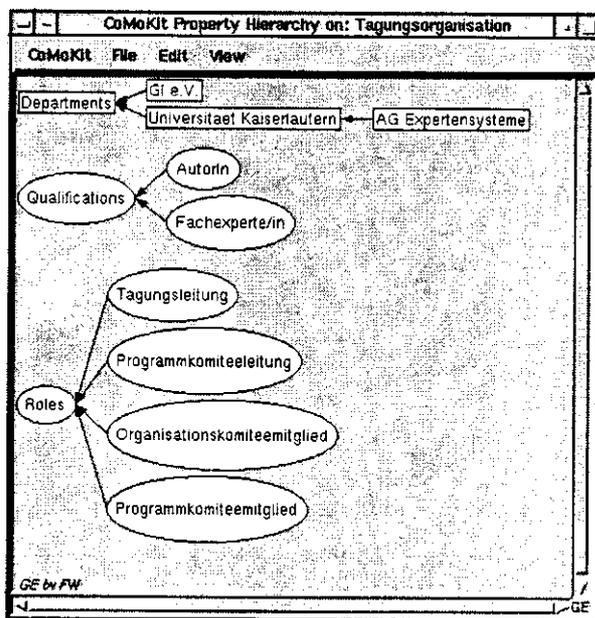


Abb. 3-4 Editor zum Definieren von Agenteneigenschaften

CoMo-Kit verwaltet für jeden Agenten die Menge seiner Eigenschaften (Properties). Dies kann als Personalinformationssystem angesehen werden. Wir unterscheiden zwischen drei verschiedenen Arten von Eigenschaften: Qualifikationen (q), Rollen (r) und der organisatorischen Einordnung (o). Beispielsweise kann der Agent „Mathias Mustermann“ Mitglied des Programmkomitees sein (r), an der Universität Kaiserslautern angestellt sein (o) und organisatorische Erfahrung besitzen (q).

Für jede Eigenschaft kann definiert werden, wie stark sie ausgeprägt ist, z.B. kann „Mathias Mustermann“ sehr viel organisatorische Erfahrung haben.

Eigenschaften können hierarchisch angeordnet werden. So ist z.B. die Eigenschaft „Erfahrung bei der Organisation von Tagungen“ eine Verfeinerung der Eigenschaft „organisatorische Erfahrung“.

## Literatur

- [DMM+97] B. Dellen, F. Maurer, J. Münch, M. Verlage: Enriching Software Process Support by Knowledge-based Techniques. To appear in: *International Journal on Software Engineering and Knowledge Engineering*, 1997.
- [DMP97] B. Dellen, F. Maurer, G. Pews: Knowledge Based Techniques to Increase the Flexibility of Workflow Management. To appear in: *Data & Knowledge Engineering Journal*, 1997.
- [MaP94] F. Maurer, J. Paulokat: Operationalizing Conceptual Models Based on a Model of Dependencies. In:

A. Cohn, ed., *Proceedings of the 11th European Conference on Artificial Intelligence (ECAI '94)*, John Wiley & Sons, Ltd., 1994.

[VDM+96] M. Verlage, B. Dellen, F. Maurer, J. Münch, A Synthesis of two Process Support Approaches. In: *Proceedings of the 8th Software Engineering and Knowledge Engineering Conference (SEKE'96)*, Knowledge Engineering Institute, June 1996.

## 4. Entwicklung von Informationssystemen mit INCOME/STAR - Prinzipien, Vorgehensmodell und Methoden

(Andreas Oberweis, Wolffried Stucky\*, Gabriele Zimmermann)

### Einleitung

Im Rahmen des Arbeitskreises „Vergleichende Analyse von Problemstellungen und Lösungsansätzen in den Fachgebieten Information Systems Engineering, Software Engineering und Knowledge Engineering“ sollen einige konkrete Ansätze aus den genannten Bereichen hinsichtlich der verwendeten Prinzipien, Methoden und Vorgehensmodelle verglichen werden. Gegenstand der vorliegenden Betrachtung ist der INCOME/STAR-Ansatz, der dem Fachgebiet Information Systems Engineering zuzuordnen ist.

Ziel des INCOME/STAR-Projekts [OSS94], das zwischen 1991 und 1995 am Institut für Angewandte Informatik und Formale Beschreibungsverfahren (AIFB) der Universität Karlsruhe durchgeführt wurde, war die Konzeption und prototypmäßige Implementierung einer kooperativen Entwicklungs- und Wartungsumgebung - INCOME/STAR - für verteilte Informationssysteme. Ausgangsbasis für die Projektplanung bildete INCOME [NOS+92], ein ebenfalls am Institut AIFB entwickeltes Werkzeug für die Modellierung und Simulation betrieblicher Abläufe. Zentrale Idee von INCOME ist die methodische Integration von semantischer Datenmodellierung und höheren Petri-Netzen beim Informationssystem-Entwurf.

Im INCOME/STAR-Projekt wurden Methoden der konzeptuellen Daten- und Verhaltens-Modellierung weiterentwickelt, und zwar insbesondere im Hinblick auf eine adäquate Berücksichtigung komplex strukturierter Objekte. Darüber hinaus wurde ein evolutionäres Vorgehensmodell für die Systementwicklung erarbeitet. Für dieses Vorgehensmodell wurde ein Aktivitäten-Managementsystem konzipiert; zur Unterstützung kooperativer Systementwicklungen wurde außerdem Groupware-Funktionalität in INCOME/STAR integriert.

### 4.1 Prinzipien des Ansatzes

Um eine gemeinsame Begriffsbasis für den Arbeitskreis zu schaffen, wurden in [LÖS94] einige zentrale Begriffe aus der Welt der Methoden und Vorgehensmodelle definiert. Danach ist unter einem (softwaretechnischen) *Prinzip* ein „allgemeingültiger, abstrakter Grundsatz“ bzw. ein „Leitfaden“ zu verstehen, der „wünschenswerte Eigenschaften des Softwareprozesses und seiner Produkte/Dokumente“ beschreibt.

Bei der Systementwicklung mit INCOME/STAR finden folgende Prinzipien Anwendung:

- evolutionäre/inkrementelle Entwicklungsstrategie,
- Integration von statischen und dynamischen Systemaspekten,

---

\* Anschrift: Universität Karlsruhe, Institut AIFB, 76128 Karlsruhe; email-Adresse: stucky@aifb.uni-karlsruhe.de

- frühzeitige (simulationsgestützte) Validierung von Anforderungen unter Einbeziehung von Endbenutzern,
- rechnergestützter Vorgehensmodelleinsatz,
- Unterstützung von teamorientierter, kooperativer Systementwicklung.

## 4.2 Vorgehensmodell

Der Entwicklungsprozeß in INCOME/STAR wird durch das evolutionäre Vorgehensmodell ProMISE (**Process Model for Information Systems Evolution**) [SOS94] gesteuert. Es ist in die Phasen „Strategische IS-Planung“, „Projekt-Planung“, „Informationsbedarfsermittlung und -analyse“, „Konzeptuelle Modellierung“, „Datenbankentwurf & -Definition / Programmwurf & Implementation“ sowie „Systemtest und Einführung“ gegliedert (Abb. 4-1).

Charakteristisch ist, daß die Daten- und Verhaltensmodellierung in den frühen Phasen integriert erfolgt. Erst beim Entwurf bzw. bei der Implementation wird zwischen „Datenbank-Definition“ und „Anwendungsprogramm-Implementation“ unterschieden. Die enge Verquickung von Datenbank-Entwurf und -Definition bzw. von Programmwurf und Implementation ergibt sich aus dem Einsatz von Datenbank- bzw. Programmgeneratoren.

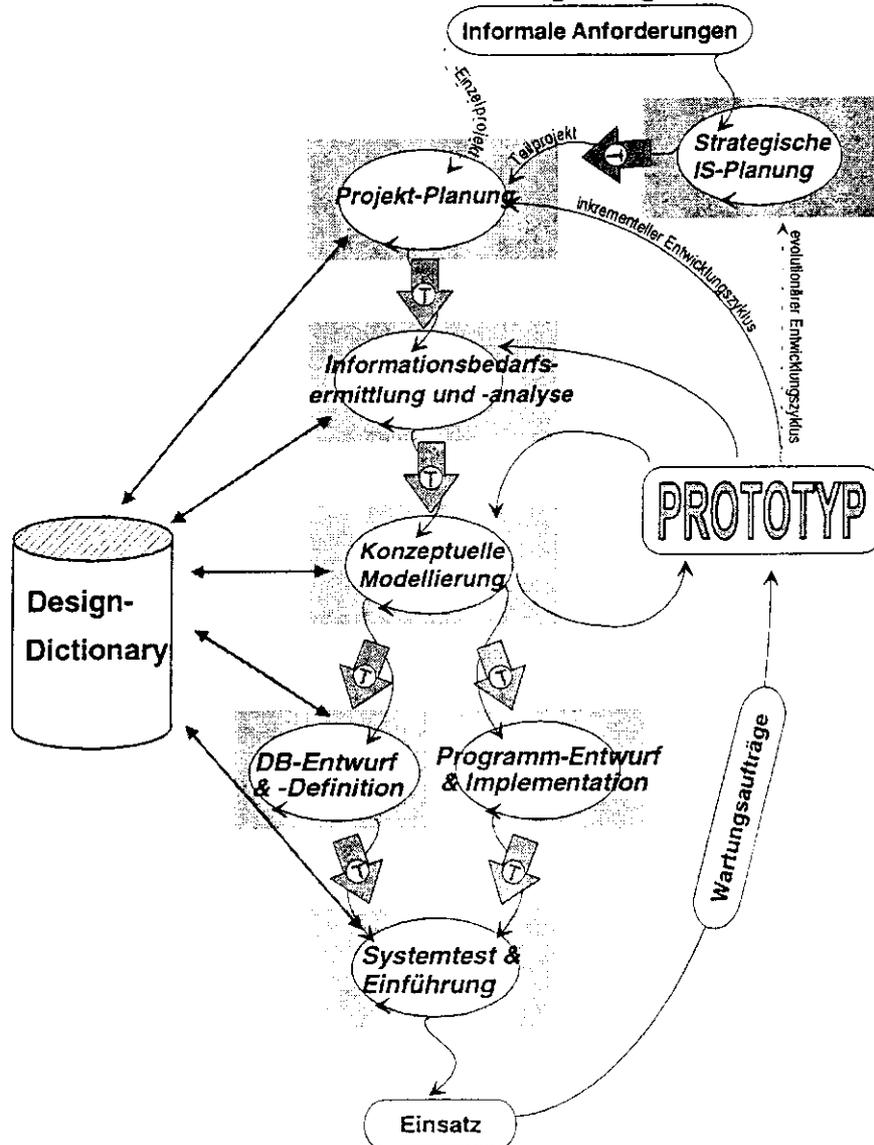


Abb. 4-1 Vorgehensmodell ProMISE

Das Konzeptuelle Schema ist direkt ausführbar und wird damit zum Prototypen für das System. So können eventuelle Defizite aus vorangegangenen Phasen offengelegt und notwendige Rücksprünge ausgelöst werden.

Eine explizite Wartungsphase fehlt: ProMISE begreift Wartung als Prozeß der evolutionären Weiterentwicklung. Durch Wartungsaufträge und Erweiterungsvorschläge, die sich beim Einsatz des Systems ergeben, wird ein neuer Evolutionszyklus in Gang gesetzt. Dabei kann jede Veränderung zunächst am konzeptuellen Schema prototypisch erprobt werden.

Alle im Verlauf des Entwicklungsprozesses entstehenden Dokumente werden in einem zentralen Design-Dictionary verwaltet.

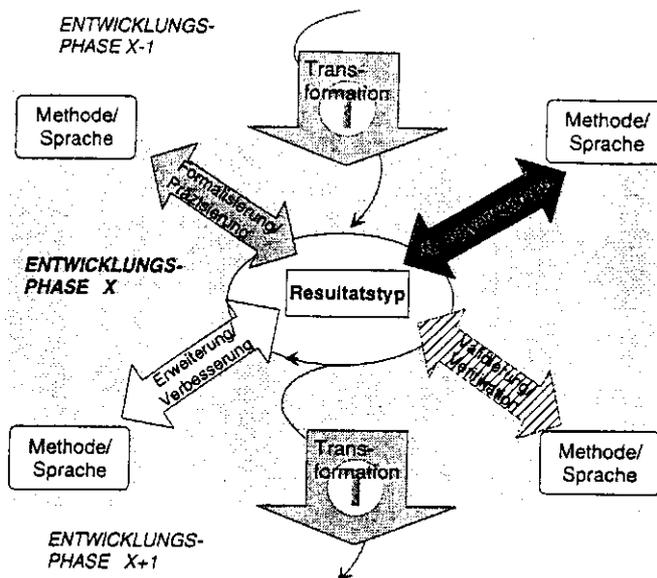


Abb. 4-2 Einzelne Phasen in ProMISE

Die einzelnen Phasen beginnen mit einem - so weit wie möglich formalen - Transformationsschritt, bei dem die Dokumente (Resultate) der vorhergehenden Phase in (initiale) Dokumente der aktuellen Phase überführt werden (symbolisiert durch die grauen Pfeile mit dem „T“-Symbol in Abb. 4-1 bzw. Abb. 4-2). Die resultierenden Dokumente werden in einer zyklischen Folge von mehr oder weniger formalen Aktivitäten so lange bearbeitet, bis ein zufriedenstellendes Ergebnis erreicht ist (vgl. Abb. 4-2). Aktivitäten können formalisieren/ präzisieren, vorhandene Dokumente aus dem Design-Dictionary wiederverwenden, zusätzliche Informationen beschaffen (Erweiterung/ Verbesserung) oder Qualitätskontrollen durchführen (Validierung/Verifizierung).

### 4.3 Sprachen und Methoden

Eine vollständige Darstellung aller in INCOME/STAR eingesetzten Sprachkonzepte ist im Rahmen dieser Kurzdarstellung nicht möglich. Bei den verwendeten Sprachen und Methoden handelt es sich durchweg um Erweiterungen bekannter Ansätze (z.B. höhere Petri-Netze, Semantische Datenmodelle), auf die an der entsprechenden Stelle verwiesen wird. Die nachfolgenden Ausführungen beschränken sich daher im wesentlichen darauf, die für INCOME/STAR vorgenommenen Neuerungen und Erweiterungen zu erläutern. Allerdings sind auch diese teilweise zu umfangreich, um sie an dieser Stelle erschöpfend zu behandeln. Es sollte jedoch möglich sein, die grundlegenden Neuerungen der jeweiligen Methode zu erfassen.

#### 4.3.1 Informationsbedarfsermittlung und -analyse: Strukturierte Glossare

In der Phase der Informationsbedarfsermittlung und -analyse wird eine semiformale Darstellung der statischen Strukturen und dynamischen Abläufe der Diskurswelt (*Anforderungsschema*) erstellt. Dazu werden - ausgehend von den aus der vorangegangenen Planungsphase bekannten Hauptanforderungen und Hauptfunktionen des Systems - mögliche Informationsquellen identifiziert und die erhaltenen Informationen in einem systematischen Prozeß nach und nach vervollständigt und in eine strukturierte Form überführt.

Die Methode basiert auf Verfahren von [DeD83], das an einigen Stellen leicht modifiziert wur-

de: Zunächst werden die zu unterstützenden Aufgaben und die damit befaßten Organisationseinheiten identifiziert, und es wird ein *Anforderungssammelplan* erstellt. Die Anforderungssammlung erfolgt durch Befragung der relevanten Personen und Analyse ggf. vorhandener Benutzersichten bzw. Datenbestände. Die Informationen werden nach Daten-, Operations- und Ereignisanforderungen klassifiziert und in strukturierter Form in sogenannten *Glossaren* (Abb. 4-3) abgelegt. Zu den verschiedenen ermittelten Datentypen werden die Mengen möglicher Datenbestände geschätzt, und die Häufigkeit der Durchführung bestimmter Operationen wird ermittelt. Zur Vermeidung von Redundanz und Mehrdeutigkeit werden Homonyme und Synonyme in den Beschreibungen ermittelt und entfernt.

| Daten-Glossar                       |                        |   |                  |                          |           |                  |                    |
|-------------------------------------|------------------------|---|------------------|--------------------------|-----------|------------------|--------------------|
| Organisationseinheit:<br>Ersteller: |                        | Programm<br>Scherrer                      |                  | Version: 1<br>10.03.1995 |           |                  |                    |
| Ident.                              | Bezeichnung            | Beschreibung                              | Wertebereich     | Häufigkeit               | Ref.-Satz | Synonym          | Unter-<br>menge v. |
| D001                                | Programmkomitee        | Menge aller Programmkomiteemitglieder     | Menge v. Strings | 1                        | D3        | Programmausschuß |                    |
| D002                                | Leiter Programmkomitee |   | String           | 1                        | D2        |                  |                    |
| D003                                | Call-for-papers        | Aufforderung zum Einreichen von Beiträgen | Text             | 1                        | D5        |                  |                    |
| ...                                 | ...                    | ...                                       | ...              |                          |           |                  |                    |

Abb. 4-3 Beispiel für ein Daten-Glossar

#### 4.3.1.1 Konzeptuelle Daten- und Verhaltensmodellierung: Semantische Datenmodelle und höhere Petri-Netze

In INCOME/STAR werden statische und dynamische Systemaspekte in ein konzeptuelles Schema integriert. Zur Beschreibung statischer, objektbezogener Aspekte stehen zwei unterschiedliche semantische Datenmodelle zur Verfügung. Für die Modellierung dynamischer Systemaspekte (Abläufe) werden höhere Petri-Netze [BRR87] eingesetzt. Die Integration von objektbezogenen Aspekten und Abläufen erfolgt, indem den statischen Elementen des Petri-Netz-Modells (den sogenannten „Stellen“) Datenschemata zugeordnet werden.

#### 4.3.1.2 Semantische Datenmodelle

Bei der Datenmodellierung in INCOME/STAR kann - je nach Entwicklungsphase, Zielgruppe oder persönlichen Präferenzen - mit zwei unterschiedlichen Datenmodellen gearbeitet werden: zum einen mit dem semantisch-hierarchischen Objektmodell SHM nach [BrR84], zum anderen mit einer Variante des Entity/Relationship-Modells (E/R-Modell). Dabei wird auch ein kontrollierter Übergang von einem Modell zum anderen methodisch unterstützt.

Im Zusammenhang mit dem E/R-Modell hat es sich gezeigt, daß es insbesondere der Entwurf großer ER-Diagramme erforderlich macht, zur Unterstützung einer schrittweisen Vorgehensweise Verfeinerungs- und Vergrößerungstechniken bereitzustellen, wie sie auch im Zusammenhang mit dem SHM-Modell eingesetzt werden. Verschiedene ER-Clustering-Techniken

wurden vorgeschlagen ([FeM86], [RaS92], [TWB+89]). Dabei werden größere Ausschnitte aus einem detaillierten Schema auf sogenannte *Entity-Cluster* abgebildet, die (komplexe) Entity-Typen eines weniger detaillierten „höheren“ ER Schemas darstellen. INCOME/STAR erweitert diese Techniken, wobei zwischen drei Arten von Clustering unterschieden wird [Jae96]:

1. Beim *Entity Clustering* wird aus einem detaillierten ER-Schema ein „höheres“ Übersichtsdiagramm abgeleitet, das von einigen Details abstrahiert. Bestimmte Ausschnitte des detaillierten Ausgangsschemas werden zu sogenannten Entity-Clustern zusammengefaßt, diese werden im höheren ER-Schema zu (komplexen) Entity-Typen.
2. Mittels *einfachem Relationship-Clustering* können Beziehungstypen verfeinert und Integritätsbedingungen präzisiert werden. Ein Beispiel gibt Abb. 4-4: der Beziehungstyp *leitet* wird durch einfaches Relationship-Clustering (im Zusammenhang mit der Verfeinerung des Entity-Typs *PK-Mitglied*) präzisiert. Die Verfeinerung drückt aus, daß nur diejenigen Programmkomiteemitglieder, die Chairperson sind, eine Sitzung leiten.

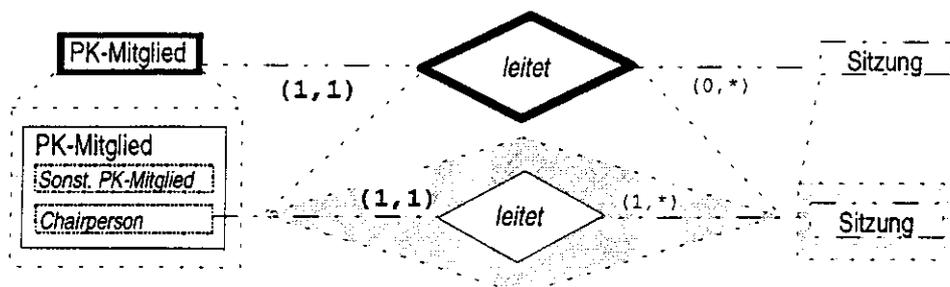


Abb. 4-4 Beispiel für Entity-Clustering und einfaches Relationship-Clustering

3. Beim *komplexen Relationship-Clustering* werden einzelne Beziehungstypen durch ganze ER-Schemata ersetzt. Der Unterschied zum einfachen Relationship-Clustering besteht darin, daß nicht nur Beziehungstypen zu mehreren ähnlichen Beziehungstypen verfeinert werden, sondern daß darüber hinaus neue Entity- und Beziehungstypen eingeführt werden.

#### 4.3.1.3 Höhere Petri-Netze

Zur Beschreibung des Systemverhaltens in INCOME/STAR wurde ein neuartiger Petri-Netz-Typ, die sogenannten NR/T-Netze ( $NF^2$ -Relationen/Transitionen-Netze), entwickelt. Dabei handelt es sich um eine Erweiterung von Prädikate/Transitionen-Netzen (Pr/T-Netzen) [Gen87], die es ermöglicht, verteilte Abläufe und komplexe Objektstrukturen in integrierter Form zu modellieren. Die Erweiterungen sind „aufwärtskompatibel“, d.h. jedes Pr/T-Netz kann auch als spezielles NR/T-Netz interpretiert werden. Bei der Verwendung herkömmlicher Pr/T-Netze kommt es immer dann zu Problemen, wenn diese in Kombination mit Datenmodellen eingesetzt wurden, die auch komplexe Objektstrukturen zulassen. Es ist beispielsweise nicht möglich, Nebenläufigkeit auf komplex strukturierten Objekten zu beschreiben. Falls Dokumente in digitaler Form vorliegen (wie im folgenden Beispiel das Programm einer Konferenz), kann es aber durchaus vorkommen, daß unterschiedliche Sachbearbeiter gleichzeitig auf unterschiedliche Komponenten desselben *Dokuments* zugreifen.

Das NR/T-Netz in Abb. 4-5 beschreibt eine solche Situation: Der Stelle Konferenzprogramm ist ein komplex strukturierter Objekttyp zugeordnet. Die Markierung der Stelle Konferenzprogramm (in Abb. 4-5 als grau unterlegte Tabelle dargestellt) entspricht dann einer genesteten Relation vom Typ Konferenzprogramm, deren Attribute teilweise wiederum genestete Relationen sind (vgl. Attribut Tagesprogramm).

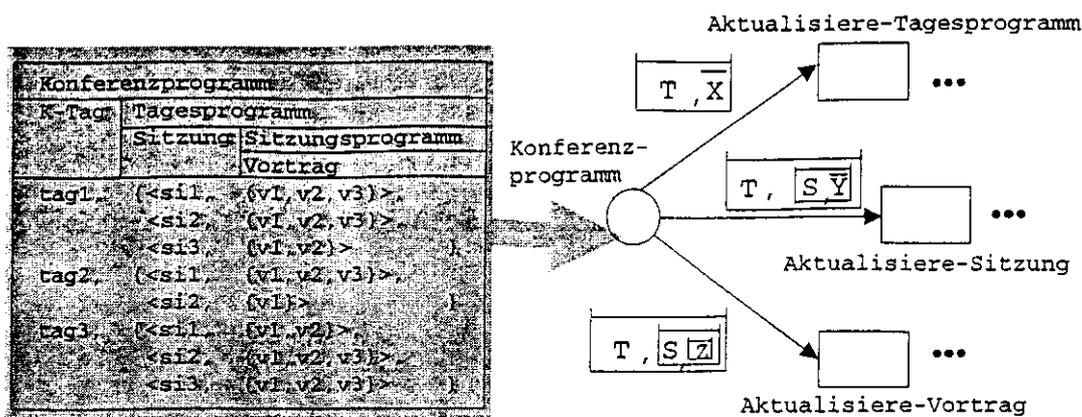


Abb. 4-5 NR/T-Netz

Eine Transition in einem NR/T-Netz stellt eine Klasse von Operationen auf den in den Eingangs- und Ausgangsstellen der Transition enthaltenen Relationen dar. Das Stattfinden einer einzelnen dieser Operationen entspricht dem „Schalten“ der jeweiligen Transition. Operationen betreffen nicht zwingend ganze Tupel einer gegebenen Relation, sondern können auch auf 'Subtupeln' einer Relation stattfinden. Die drei Transitionen in Abb. 4-5 modellieren jeweils eine andere Art von Zugriff auf das Konferenzprogramm, nämlich auf das Programm eines gesamten Tages (Transition Aktualisiere-Tagesprogramm), auf einzelne Sitzungen (Aktualisiere-Sitzung) oder aber auf einzelne Vorträge einer bestimmten Sitzung (Aktualisiere-Vortrag). Dabei kann der Zugriff auf verschiedene Sitzungen desselben Tages bzw. auf unterschiedliche Vorträge derselben Sitzung nebenläufig erfolgen. Über sogenannte „Filtertabellen“ an den Netzkanten wird festgelegt, welche Daten aus den Eingangsstellen zu entnehmen sind, d.h. auf welche Tage, Sitzungen oder Vorträge zugegriffen wird.

Eine ausführliche, formale Einführung in das NR/T-Netz Konzept und weitere Beispiele finden sich in [Obs96].

#### 4.4 Ausblick

In „konventionellen“ Informationssystemen, wie sie mit INCOME/STAR entwickelt werden, wird die Datenverwaltungsfunktionalität von Datenbank-Managementsystemen bereitgestellt, während die Ablaufausführung durch individuell zu erstellende (und daher nur mit verhältnismäßig großem Aufwand später noch änderbare) Anwendungs-Software realisiert wird.

In jüngster Zeit werden Workflow-Managementsysteme eingesetzt, um auch auf der Ablaufseite entsprechende flexible Unterstützung bereitzustellen: betriebliche Abläufe werden in einem Ablaufschema modelliert, welches dann vom Workflow-Managementsystem ausgeführt wird.

Seit 1995 werden im Anschlußprojekt INCOME/WF Methoden und Werkzeuge für die Entwicklung workflow-basierter Informationssysteme entwickelt, eine Übersicht gibt [OSS+97].

#### Literatur

- [BrR84] M.L. Brodie, D. Ridjanovic: On the design and specification of database transactions. In: M.L. Brodie, J. Mylopoulos, J.W. Schmidt (Hrsg.): *On Conceptual Modelling*, Springer 1984.
- [BRR87] W. Brauer, W. Reisig, G. Rozenberg (Eds.): *Petri nets: central models and their properties*. Advances in Petri Nets '86. Springer 1987.
- [DeD83] V. De Antonellis, B. Demo: Requirements collection and analysis. In: S. Ceri (Hrsg.): *Methodology and Tools of Database Design*, North-Holland 1983, S. 9-24.
- [FeM86] P. Feldman, D. Miller: Entity model clustering: structuring a data model by abstraction. *The*

- Computer Journal*, Vol. 29(4) 1986, S. 348-360.
- [Gen87] H.J. Genrich: Predicate/transition nets. In [BRR87].
- [Jae96] P. Jaeschke: *Integrierte Unternehmensmodellierung - Techniken zur Informations- und Geschäftsprozeßmodellierung*. Deutscher Universitäts-Verlag 1996.
- [LöS94] P. Löhr, G. Scherrer: Software Engineering - Methoden und Vorgehensmodelle. *EMISA FORUM*, Heft 2 1994, S. 80-82.
- [NOS+92] T. Németh, A. Oberweis, F. Schönthaler, W. Stucky: *INCOME: Arbeitsplatz für den Programmwurf interaktiver betrieblicher Informationssysteme*. Universität Karlsruhe, Institut für Angewandte Informatik und Formale Beschreibungsverfahren, Forschungsbericht 251, 1992.
- [ObS96] A. Oberweis, P. Sander: Information system behavior specification by high-level Petri nets. *ACM Transactions on Information Systems*, Vol. 14(4) 1996, S. 380-420.
- [OSS94] A. Oberweis, G. Scherrer, W. Stucky: INCOME/STAR: Methodology and tools for the development of distributed information systems. *Information Systems*, Vol. 19(8) 1994, S. 641-658.
- [OSS+97] A. Oberweis, R. Schätzle, W. Stucky, W. Weitz, G. Zimmermann: INCOME/WF - A Petri net based approach to Workflow Management. In: H. Krallmann (Hrsg.): *Wirtschaftsinformatik '97*, Springer 1997, S. 557-580.
- [RaS92] O. Rauh, E. Stickel: Entity tree clustering - a method for simplifying ER design. *Proceedings of the 11th International Conference on the Entity-Relationship Approach*, Karlsruhe 1992, S. 62-78.
- [SOS94] G. Scherrer, A. Oberweis, W. Stucky: ProMISE - a process model for information system evolution. *Proceedings of the 3rd Maghrebian Conference on Software Engineering and Artificial Intelligence*, Rabat 1994, S. 27-36.
- [TWB+89] T.J. Teorey, G. Wei, D.L. Bolton et al.: ER model clustering as an aid for user communication and documentation in database design. *Communications of the ACM*, Vol. 32(8) 1989, S. 975-987.

## 5. Die Methode MORE/RT

(Wolfgang Stein)

### Einleitung

Im folgenden wird die objektorientierte Analysemethode MORE/RT (Method for object-oriented Requirements Engineering of Real-Time Systems) [Ste95] vorgestellt, die dem Fachgebiet Software Engineering zuzuordnen ist. Die Methode integriert bekannte Ansätze wie z.B. [Boo94] und [RBP+91] und bietet darüberhinaus spezielle Konzepte für die Entwicklung von verteilten, nebenläufigen Echtzeitsystemen. Sie wurde bei Prof. Dr.-Ing. habil. Helmut Balzert am Lehrstuhl für Software-Technik der Ruhr-Universität Bochum entwickelt.

### 5.1 Prinzipien des Ansatzes

Im folgenden werden exemplarisch einige charakteristische Konzepte der Methode MORE/RT skizziert:

- die Unterbrechbarkeit der Bearbeitung und die Fehlerbehandlung,
- die implementierungsunabhängige Spezifikation der Kommunikation und Synchronisation auf hohem Abstraktionsniveau,
- die zeitbehaftete Simulation des Systemmodells,
- das Abstraktionskonzept.

#### 5.1.1 Die Unterbrechbarkeit der Bearbeitung und die Fehlerbehandlung

Die Services eines Modellobjektes können i.a. nicht in beliebiger Reihenfolge ausgeführt werden. Meistens muß eine bestimmte Reihenfolge eingehalten werden, damit die Attribute sinnvolle Werte annehmen.

Die Analysemethoden verwenden Objektlebenszyklen, um die Reihenfolge der Services zu spezifizieren. Viele Methoden beschreiben aber den Zusammenhang zwischen den Objekten bzw. Klassen einerseits und den Objektlebenszyklen andererseits nicht hinreichend. Es entstehen zwei weitgehend unabhängige Dokumente. Die Integration der Objektlebenszyklen in das restliche Systemmodell fehlt.

In der Methode MORE/RT werden die Lebenszyklen eines Objektes als Objektzustandsgraph modelliert. Objektzustandsgraphen sind ein integrierter Bestandteil des Systemmodells. Konsistenzprüfungen zwischen dem Objektzustandsgraphen und den Servicedefinitionen eines Objektes werden unterstützt. Die Objektzustandsgraphen sind bei der Interpretation des Systemmodells wichtig. Im einzelnen wird hier entschieden:

- welche Botschaft aus der Botschaftwarteschlange eines Objektes abgearbeitet wird,
- ob ein Service unterbrochen oder abgebrochen wird,
- ob eine Fehlerbehandlung durchgeführt wird.

Die speziellen Transitionstypen der Objektzustandsgraphen sind geeignet, um beispielsweise schnelle Reaktionen auszuführen oder Zustandsübergänge mit Prioritäten zu versehen.

### 5.1.2 Die implementierungsunabhängige Spezifikation der Kommunikation und Synchronisation

Natürliche Objekte arbeiten i.a. parallel. Beispielsweise kann die LeiterIn des Programmkomitees für die begutachteten Beiträge bereits Zusagen und Absagen versenden, während andere Beiträge zeitgleich noch begutachtet werden. Dazu sind geeignete Kommunikations- und Synchronisationskonzepte erforderlich, die auf dem Botschaftskonzept aufbauen.

Zur Inter-Objekt-Kommunikation bietet die Methode MORE/RT den Austausch von Botschaften zwischen einzelnen Objekten an. Für die einzelnen Kommunikationsverbindungen kann mittels der Botschaftskennung z. B. spezifiziert werden, ob der Sender nach dem Absenden der Botschaft weiterarbeiten kann oder ob er zunächst auf die Antwortbotschaft warten muß. Die Botschaftskennung enthält Angaben über das Antwortzeitverhalten, die Laufzeit und die Relevanz der Botschaft. Mittels des Antwortzeitverhaltens der einzelnen Botschaften [IR], [LR], [NR] und der Antwortkennung der Services [R+], [R-] kann die synchrone und asynchrone Kommunikation zwischen einzelnen Objekten implementierungsunabhängig und auf einem hohem Abstraktionsniveau sehr elegant modelliert werden.

Systeme, die zur Überwachung, zur Steuerung oder zur Regelung eines Prozesses eingesetzt werden, benötigen bestimmte Prozeßdaten. Die Prozeßdaten müssen aus dem Prozeß gewonnen werden und in den Speicher des Systems übertragen werden. Die Aktualisierung der systeminternen Datenspeicher muß innerhalb definierter Zeitabstände erfolgen, damit das System stets mit den aktuellen Werten arbeitet.

Die Methode MORE/RT besitzt das Konzept der Monitorattribute zur Verarbeitung der Prozeßdaten. Monitorattribute sind Attribute, deren Wert von einem anderen Objekt implizit zyklisch aktualisiert wird. Häufig werden Monitorattribute direkt von Schnittstellenobjekten aktualisiert. Monitorattribute ermöglichen die schnelle Aktualisierung der Attributwerte ohne die Datenkapselung grob zu verletzen. Die implizite zyklische Aktualisierung der Attributwerte reduziert den Aufwand zur Erstellung des Systemmodells erheblich. Innerhalb eines Objektes können die Services stets auf die aktualisierten Werte der Monitorattribute zugreifen, ohne daß die Synchronisation explizit modelliert werden muß.

### 5.1.3 Die zeitbehaftete Simulation des Systemmodells

Angesichts der Komplexität moderner Systeme ist die Überprüfung eines semiformalen Systemmodells auf Korrektheit, Widersprüchlichkeit und Vollständigkeit ohne Simulation des Systemverhaltens praktisch unmöglich. Darüberhinaus ist die Simulation die Grundlage für die Animation des Systemverhaltens.

Zwei Kernprobleme der Simulation eines zu entwickelnden Systems sind die Modellbildung einerseits und die spätere, detailgetreue Realisierung des optimierten Modells andererseits. Diese Probleme können durch die Integration der Simulation in eine objektorientierte Analysemethode gelöst werden. In diesem Fall erfolgt die Modellbildung mittels der objektorientierten Analysemethode. Die Übereinstimmung des simulierten Systemmodells mit der späteren Implementierung wird von der durchgängigen, objektorientierten Software-Entwicklung unterstützt, die eine detailgetreue Implementierung des Systemmodells erleichtert. Die Implementierung wird sich naturgemäß so verhalten wie das Analysemodell. Es ist daher sinnvoll, ein Simulationsverfahren in eine objektorientierte Analysemethode zu integrieren.

Eine bekanntes Konzept zur Simulation bietet die Methode OOSE [JCJ+92] mit dem use case model. Ein use case ist die verbale Beschreibung eines typischen Anwendungsfalls für das System. Ein Nachteil des use case-Konzepts ist die verbale Beschreibung, die keine Möglichkeiten zur rechnerunterstützten Animation bietet. Damit fehlt der Bezug zum objektorientierten Systemmodell.

In der Methode MORE/RT wurde der Grundgedanke des use case-Konzepts zum Szenariokonzept weiterentwickelt. Das Szenariokonzept integriert eine funktionelle, deterministische Simulation in die Methode MORE/RT. Ein Szenario beschreibt einen typischen Anwendungsfall des Systems. Es wird durch vier Komponenten definiert: das MORE/RT-Systemmodell, die Anfangszustandsbelegung der einzelnen Objekte, die Liste der externen Botschaften und optional kann eine Terminationsbedingung angegeben werden.

Ein Szenario in der Methode MORE/RT ist semantisch z. T. mit einem use case in der Methode OOSE vergleichbar. Im Gegensatz zum use case kann mittels eines Szenarios das Verhalten des Systemmodells auf dem Bildschirm eines Rechners animiert werden. Während ein use case rein textuell beschrieben wird, besteht ein Szenario aus graphischen und textuellen Komponenten. Darüberhinaus können im Szenario Zeitanforderungen definiert werden, die zur zeitbehafteten Simulation einzelner Komponenten und des Gesamtsystems benötigt werden.

Die Methode MORE/RT ermöglicht für jeden Service die Angabe der Verarbeitungszeit und für jede Botschaft die Angabe der Laufzeit. Die Zeitabschätzungen sind die Grundlage für worst-case-Kalkulationen der Antwortzeiten des Gesamtsystems. Sie können durch die Simulation des Systemmodells überprüft werden. Darüberhinaus kann das Zeitverhalten des Systemmodells mit Zeitdiagrammen dargestellt werden.

### 5.1.4 Das Abstraktionskonzept

Ein geeignetes Hierarchiekonzept ist notwendig, um die Architektur dieser Systeme auf unterschiedlichen Abstraktionsebenen darzustellen.

Die Methode RDD [WWW90] verwendet ein brauchbares Abstraktionskonzept mit explizit definierten Schnittstellen der Subsysteme. Die Methode MORE/RT erweitert das Abstraktionskonzept so, daß die Hierarchien sowohl anhand der Aggregatstrukturen als auch anhand der Vererbungsstrukturen gebildet werden können. Dieses Abstraktionskonzept ermöglicht die hierarchische Dekomposition des Systemmodells in Teilmodelle mit definierten Schnittstellen auf mehreren Diagrammen.

## 5.2 Vorgehensmodell

Das globale Vorgehensmodell zur Systementwicklung mit der Methode MORE/RT ist in Abb. 5-1 dargestellt.

| Phase                     | Methode | Sprachen, Modelle  |
|---------------------------|---------|--|
| Analyse des Ist-Zustandes | MORE/RT | Szenarien, Objektdiagramme, Zeitdiagramme  |
| Anforderungsdefinition    | MORE/RT | Klassendiagramme, Objektzustandsgraphen, Pseudocode, Szenarien, Objektdiagramme, Zeitdiagramme |
| Entwurf                   | COOD    | Klassendiagramme, Objektzustandsbeschreibung, Pseudocode                                       |
| Implementierung           | COOD    | C++, Ada   |

Abb. 5-1 Globales Vorgehensmodell

In den Phasen Entwurf und Implementierung wird das mit MORE/RT entwickelte Systemmodell mit der Methode COOD [Hüs95] weiterentwickelt. Die Methoden MORE/RT und COOD ergänzen sich. Zwischen den einzelnen Phasen sind Iterationen vorgesehen, die vom CASE-Werkzeug unterstützt werden.

## 5.3 Sprachen und Methoden

### 5.3.1 Klassendiagramme

Ein Klassendiagramm beschreibt die statischen Beziehungen zwischen Klassen und Objekten. Jede Klasse des MORE/RT-Modells wird durch ihr Klassensymbol repräsentiert. Außerdem werden Vererbungsstrukturen, Aggregationsstrukturen, Assoziationsstrukturen, Kommunikationsstrukturen, Inkarnationsstrukturen und Abstraktionsstrukturen dargestellt. Die einzelnen Strukturen repräsentieren unterschiedliche Konzepte. Sie können in unterschiedlichen Diagrammen dargestellt werden. Bei komplexen Systemmodellen hat sich die folgende Trennung bewährt:

- Ein Diagramm beschreibt Vererbungsstrukturen, Aggregationsstrukturen und Assoziationsstrukturen.
- Ein zweites Diagramm beschreibt die Kommunikationsstrukturen.
- Ein drittes Diagramm beschreibt die Inkarnationsstrukturen.

Jedes Klassendiagramm wird mit einem Diagrammtypenschild versehen. Es enthält den Namen des MORE/RT-Modells, die Versionsnummer, das Erstellungsdatum, das letzte Änderungsdatum, den Namen des Systemanalytikers und den Bearbeitungsstatus.

### 5.3.2 Objektzustandsgraphen

Objektzustandsgraphen basieren auf Statecharts [Har88] und deren Weiterentwicklung zu Objectcharts [CHB92]. Sie verwenden ein hybrides Moore-/Mealy-Automatenmodell zur Beschreibung des dynamischen Verhaltens.

### 5.3.3 Pseudocode

Der Pseudocode wird zur Definition der Klassen mit ihren Attributen und Services eingesetzt. Die Services können mit beliebigen Texten beschrieben werden. Falls allerdings eine Simulation des MORE/RT-Modells geplant ist, müssen die Services im Pseudocode (MORE/RT-PC) angegeben werden, damit der Modellinterpret MINT die Servicerümpfe interpretieren kann. Anweisungen, die nicht interpretiert werden sollen, müssen als Kommentar gekennzeichnet werden.

### 5.3.4 Szenarien

MORE/RT unterstützt die Simulation eines Systemmodells. Zur Simulation werden mehrere Szenarien (use cases) [JCJ+92] definiert. Jedes Szenario beinhaltet eine externe Botschaftssequenz. Sie enthält Botschaften, die aus der Systemumgebung an das System gesendet werden und entsprechende szenariotypische Reaktionen im Systemmodell hervorrufen.

### 5.3.5 Objektdiagramme

Ein Objektdiagramm beschreibt die statischen Beziehungen zwischen typischen Instanzen, wie sie während des Systembetriebs auftreten können. Das Ziel ist es, exemplarisch und anschaulich sowohl den Ist-Zustand als auch das zu entwickelnde System zu demonstrieren. Jede Instanz des MORE/RT-Modells wird durch ihr Objektsymbol repräsentiert. Außerdem können Aggregationsstrukturen, Assoziationsstrukturen, Kommunikationsstrukturen, Inkarnationsstrukturen und Abstraktionsstrukturen dargestellt werden.

Jedes Objektdiagramm wird mit einem Diagrammtypenschild versehen, das den Namen des MORE/RT-Modells, die Versionsnummer, das Erstellungsdatum, das letzte Änderungsdatum, den Namen des Systemanalytikers und den Bearbeitungsstatus enthält.

### 5.3.6 MORE/RT-Zeitdiagramme

MORE/RT verwendet spezielle Zeitdiagramme, die das dynamische Verhalten des Systems beim Ablauf eines Szenarios beschreiben. Zeitdiagramme werden beim Entwurf digitaler Schaltungen verwendet, um das Zeitverhalten der Schaltungen zu beschreiben.

Das MORE/RT-Zeitdiagramm ist ein zweidimensionales Diagramm mit einer Zeitachse und einer Objektachse. Auf der Zeitachse wird die Simulationszeit aufgetragen. Auf der Objektachse werden diejenigen Objekte aufgetragen, die im Rahmen eines Szenarios betrachtet werden sollen. Für jedes Objekt der Objektachse wird zu jedem Simulationszeitpunkt der Objektzustand bestimmt und in das Diagramm eingetragen.

Für jedes Szenario können die betrachteten Objekte bestimmt werden. Darüberhinaus kann ein Objekt besonders hervorgehoben werden.

## Literatur

- [Boo94] G. Booch: *Object Oriented Analysis and Design with Applications*. 2nd ed., Benjamin/Cummings Publishing Company Inc. Redwood City, CA 1994.
- [CHB92] D. Coleman, F. Hayes, and S. Bear: Introducing Objectcharts or How to Use Statecharts in Object-Oriented Design. In: *IEEE Transactions on Software Engineering*, Vol. 18, No. 1, Jan. 1992.
- [EKW92] D. E. Embley, B. D. Kurtz, and S. N. Woodfield: *Object-Oriented Systems Analysis – a Model-Driven Approach*. Prentice Hall Englewood Cliffs, New Jersey 1992.
- [Har88] D. Harel: On Visual Formalisms. *Communications of the ACM*, Vol. 31, No. 5, May 1988, pp. 514-531.
- [Hüs95] T. Hüsener: *Objektorientierter Entwurf von nebenläufigen, verteilten und echtzeitfähigen Software-Systemen*, Dissertation, Spektrum Akademischer Verlag Heidelberg, 1995.

- [JCJ+92] I. Jacobson, M. Christerson, P. Jonsson, and G. Övergaard: *Object-Oriented Software Engineering*. Addison-Wesley Wokingham, England 1992.
- [RBP+91] J. Rumbaugh, M. Blaha, W. Premerlani, F. Eddy, and W. Lorenson: *Object-Oriented Modelling And Design*. Prentice Hall Englewood Cliffs, New Jersey 1991.
- [Ste95] W. Stein: *Objektorientierte Analyse für nebenläufige Systeme*. Dissertation, BI-Verlag Mannheim 1995.
- [WWW90] R. Wirfs-Brock, B. Wilkerson, and L. Wiener: *Designing Object-Oriented Software*. Prentice Hall Englewood Cliffs, New Jersey 1990.

## 6. Ergebnis der Umfrage im Überblick

### 6.1 Allgemeines, Übergeordnetes

#### 6.1.1 Übergeordnetes Vorgehensmodell

|  | MIKE   | CoMo-Kit   | INCOME / STAR  | MORE / RT  |
|--|--|--|--|--|
| <b>Art des Vorgehensmodells</b>                  | zyklisch   | zyklisch   | linear, mit Rücksprünge  | zyklisch   |
| <b>Prototyping integriert in Vorgehensmodell</b> | exploratives, experimentelles und evolutionäres Prototyping  | exploratives Prototyping   | exploratives und evolutionäres Prototyping   | wird nicht explizit unterstützt  |
| <b>Phasen / erzeugte Modelle</b>                 | Analyse (Erhebung, Interpretation, Formalisierung) / Erhebungsmodell, Strukturmodell, Modell der Expertise, Design / Designmodell Implementierung / Wissensbasis | objektorientierte Datenmodelle mittels automatisch generierter Smalltalk-Klassen, hierarchische Aufgabenmodelle, Datenflußgraphen, Protokolle mit Hyperlinks | Planung, Informationsbedarfsermittlung und -analyse, konzeptuelle Modellierung, DB-Entwurf & -Definition, Programm-Entwurf & Implementation, Integration & Test, Einsatz (keine explizite Wartungsphase, da evolutionärer Ansatz), Anforderungsschema (Glossardarstellung), integriertes konzeptuelles Daten- und Verhaltensschema, Datenbank-Schema (logisch, extern, intern) | Analyse / Systemmodell mit Klassendiagrammen, Zustandsdiagrammen, textuelle Spezifikation, Objekt-Zustands-Wechseldiagrammen, Szenarieneschreibungen, Entwurf / Systementwurf mit Klassendiagrammen, Verteilung der Objekte in der Systemlandschaft, textuelle Beschreibung in Pseudocode, Simulationsergebnissen, Implementierung / ausführbares Programm |

|                              | MIKE  | CoMo-Kit   | INCOME / STAR   | MORE / RT   |
|------------------------------|---|--|---|---|
| <b>Projektmanagement</b>     | Vorgehensmodell in das Projektmanagement integriert | ---  | teilweise in das Vorgehensmodell integriert   | extern definiert  |
| <b>Werkzeugunterstützung</b> | Guideline-Komponente                                | Modellierungskomponente für Daten-, und Aufgabenmodelle, Komponente zur Interpretation der erstellten Modelle, Informationskomponente zum Abrufen von Projektinformation | Process engine, Dictionary, Process editor, Hypertext-Strukturierte Vorgehensmodell-Dokumentation | die CASE-Plattformen TORE/RT und COOD unterstützen von der Analyse über den Entwurf bis zur Implementierung |

6.1.2 Methodenintegration

|  | MIKE   | CoMo-Kit  | INCOME / STAR  | MORE / RT  |
|--|--|---|--|--|
| <b>Integration von Modellen</b>                | unterschiedliche Dokumente enthalten direkte Bezüge zueinander   | Integration von Datenmodell, Aufgabenmodell und Datenflußgraphen, Hyperlinks zu Protokollen       | insbesondere Integration von Ablauf-, Daten- und Organisationsmodellierung   | Integration von Klassendiagrammen, Objektzustandsgraphen, Objektzustandssequenzen und textueller Spezifikation                 |
| <b>Übergang zwischen den Methoden / Phasen</b> | durch Prozeßmodell unterstützt; fließender, weitestgehend strukturhaltender Übergang zwischen informaler Beschreibung und System | fließend  | Phasen sind durch (semi-)formale Transformations-schritte miteinander verbunden (in späteren Phasen: Einsatz v. Generatoren)   | Verfahren zur Cross-Referenzprüfung zwischen den Klassendiagrammen, den Objektzustandsgraphen und der textuellen Spezifikation |
| <b>Werkzeugunterstützung</b>                   | MIKE-Tool: enthält integriert alle Modelle und unterstützt integrierend die methodische Erstellung der Modelle                   | graphische Editoren zum Erstellen von Modellen, Scheduler (Interpreter) zum Abwickeln der Modelle | graphische Editoren für Semantische Datenmodelle und Petri-Netze, Simulatoren für Petri-Netze, Editor zur Formulierung graphischer Anfragen an Simulationsdatenbanken (die genannten Werkzeuge sind integriert, d.h. es kann z.B. vom Editor aus ein Simulationslauf gestartet werden) | das Werkzeug unterstützt die Cross-Referenzprüfung auf der syntaktischen Ebene   |

6.1.3 Traceability

|   | MIKE  | CoMo-Kit  | INCOME / STAR   | MORE / RT   |
|---|---|---|---|---|
| <b>Verwendete Methode zur Unterstützung von Traceability:</b> | Hypertext- / -media-Strukturierung und strukturierhaltender Entwurf                 | Hypertext-Links zwischen Aufgaben und Protokollen                                 | zentrales Design-Dictionary zur strukturierten Verwaltung aller Entwicklungsdokumente | verschiedene Versionen eines Modells, Bezug zwischen Analyse, Entwurf und Implementierung in zentraler DB |
| <b>einbezogene Modelle</b>                                    | alle  | alle  | alle  | alle  |
| <b>Werkzeugunterstützung</b>                                  | MIKE-Tool: automatische Verzeigerung zwischen den Wissens-elementen in den Modellen | Modellierungskomponente: Hypertext-Verbindungen zwischen Protokollen und Modellen | Dictionary, graphische Anfragesprache für Versionsdaten                               | Versionsverwaltung innerhalb der einzelnen Phasen, Navigation zwischen den einzelnen Phasen               |

## 6.2 Sprachen / Methoden / Modelle

|   | MIKE / Strukturmodell   | MIKE / KARL   | MIKE / DesignKARL   | CoMo-Kit  |
|---|---|---|---|---|
| 6.2.1 Name der Sprache / Methode / des Modells:                   | <b>Strukturmodell</b>   | <b>KARL</b>   | <b>DesignKARL</b>   | <b>CoMo-Kit-Ontologie</b>   |
| 6.2.2 kurze Charakterisierung der Sprache / Methode / des Modells | semiformale Beschreibung von Konzepten und Relationen, Aktivitäten und deren Dekomposition, Datenfluß und Kontrollfluß; Konzepte, Aktivitäten und deren Beziehungen untereinander werden über vorgegebene Knoten- und Kantentypen beschrieben, dabei ist ein elementarer Baustein ein Hypermedia-Dokument | formale und ausführbare Sprache zur Beschreibung von statischem Wissen (Ontologie und Domänenmodell) und dynamischem Wissen (Kontroll- / Datenfluß) | Erweiterung des Modells der Expertise zur Spezifikation von Datenstrukturen und Algorithmen über die Strukturierungsprimitive Cluster und Modul, Unterstützung des Entwurfsprozesses durch Verwaltung der Abhängigkeiten zwischen den Anforderungen | natürliche Hypertextprotokolle, Aufgaben, Aufgabenhierarchisierung, Methoden, Agenten, Datenflußbeziehungen, Konzeptklassen- und Instanzen, Attribute und Agenten |
| Möglichkeit der Verfeinerung / Hierarchisierung:                  | Konzepte / Aktivitäten  | Ontologie, Kontroll- / Datenfluß  | DModule, PModule  | Spezialisierung und part-of-Beziehungen in Datenmodell, Aufgabendeckungsposition im Aufgabenmodell  |
| Einsatz in welcher Phase  | Interpretation  | Formalisierung  | Design  | beim Erstellen der Protokolle, bei der Spezifikation der Modelle  |

## 6.2 Sprachen / Methoden / Modelle

|   | INCOME / IBEA  | INCOME / NR/T-Netze  | INCOME / SDM  | MORE / RT   |
|---|--|--|---|---|
| 6.2.1 Name der Sprache / Methode / des Modells:                   | <b>strukturierte Glossare</b>  | <b>NR/T-Netze</b>  | <b>Semantische Datenmodellierung</b>  | <b>MORE/RT-PC</b>   |
| 6.2.2 kurze Charakterisierung der Sprache / Methode / des Modells | vom System zu unterstützende Aufgaben und damit befaßte Organisationseinheiten werden identifiziert, und ein Anforderungssammelplan erstellt, Anforderungssammlung erfolgt durch Befragung der relevanten Personen und Analyse ggf. vorhandener Datenbestände. Informationen werden nach Daten-, Operations- und Ereignisanforderungen klassifiziert und in strukturierter Form in sog. Glossaren abgelegt | neuer Typ höherer Petri-Netze: eine Erweiterung von Prädikate/Transitionen-Netzen, die es ermöglicht, verteilte Abläufe und komplexe Objektstrukturen in integrierter Form zu modellieren. | Semantisches Hierarchisches Objektmodell nach Brodie / Ridjanovic, Entity-Relationship Modell mit Erweiterungen zur Vergrößerung / Verfeinerung | Pseudocode zur Beschreibung einer Systemstruktur und des Systemverhaltens, die Spezifikation des Zeitverhaltens ist möglich |
| Möglichkeit der Verfeinerung / Hierarchisierung:                  | ---  | Ja   | Clustering-Techniken  | Aggregatstrukturen, Schachtelung von Funktionen, Prozeduraufrufe  |
| Einsatz in welcher Phase  | Informationsbedarfsermittlung und -analyse   | konzeptuelle Modellierung  | Konzeptuelle Modellierung   | Analyse, Grobentwurf  |

6.2.3 Zweck und Zielgruppe der Sprache / Methode / des Modells

| MIKE / Strukturmodell   | MIKE / KARL   | MIKE / DesignKARL   | CoMo-Kit  |
|---|---|---|---|
| frühe Strukturierung<br>Dokumentation (Verweise in Protokolle),<br>Kommunikationsbasis (graphisch / textuell)<br>für Experte und Knowledge Engineer | eindeutige, formale Spezifikation der funktionalen Aspekte des zu entwickelnden Systems, für Knowledge Engineer | Unterstützung des Übergangs vom Modell der Expertise zum System, Dokumentation von Entwurfsentscheidungen, für Knowledge Engineer, speziell: Systemdesigner | Anwendungsexperte zusammen mit Systemanalytiker, Softwareplaner- und entwickler |

6.2.4 Repräsentierbarkeit

| MIKE / Strukturmodell            | MIKE / KARL                      | MIKE / DesignKARL                | CoMo-Kit                           |
|----------------------------------|----------------------------------|----------------------------------|------------------------------------|
| textuell strukturiert, graphisch | textuell strukturiert, graphisch | textuell strukturiert, graphisch | textuell unstrukturiert, graphisch |

6.2.5 Semantik der Sprache / Methode / des Modells

| MIKE / Strukturmodell           | MIKE / KARL  | MIKE / DesignKARL                         | CoMo-Kit   |
|---------------------------------|--|---|--|
| informal dokumentierte Semantik | dokumentiert durch Compiler / Interpreter (operationale Semantik), formal dokumentierte deklarative Semantik | formal dokumentierte deklarative Semantik | dokumentiert durch Compiler / Interpreter (operationale Semantik), informal dokumentierte Semantik |

6.2.3 Zweck und Zielgruppe der Sprache / Methode / des Modells

|  | INCOME / IBEA   | INCOME / NR/T-Netze  | INCOME / SDM  | MORE / RT  |
|--|---|--|---|--|
|  | Sammlung, schrittweise Präzisierung u. Strukturierung von Systemanforderungen, „Schnittstelle“ zwischen Anwender / Do-main-Experte und Ent-wickler, | Zweck ist die formale, im-plementationsunabhängi-ge Beschreibung betrieblicher Abläufe. Zielgruppe sind System-Analytiker und -Entwick-ler, in Verbindung mit Pro-totyping auch Anwender | Zweck des Modells ist die präzise, implementations-unabhängige Beschrei-bung von relevanten Objektstrukturen. Ziel-gruppe sind Systemanaly-tiker, in Verbindung mit Prototyping auch Anwen-der. | MORE/RT-PC ist maschi-nell interpretierbare Spra-che. Verhalten mit MORE/RT-PC dokumentierter Modelle kann am Compu-ter animiert werden. |

6.2.4 Repräsentierbarkeit

|  | INCOME / IBEA                        | INCOME / NR/T-Netze | INCOME / SDM | MORE / RT                        |
|--|--------------------------------------|---------------------|--------------|----------------------------------|
|  | textuell strukturiert, tabel-larisch | grafisch            | grafisch     | textuell strukturiert, gra-fisch |

6.2.5 Semantik der Sprache / Methode / des Modells

|  | INCOME / IBEA                   | INCOME / NR/T-Netze                        | INCOME / SDM                    | MORE / RT  |
|--|---------------------------------|--|---------------------------------|--|
|  | informal dokumentierte Semantik | formal dokumentierte de-klarative Semantik | informal dokumentierte Semantik | dokumentiert durch Com-piler / Interpreter (opera-tionale Semantik), informal dokumentierte Semantik |

6.2.6 Beschreibbarkeit von Aspekten (Sprachprimitive angeben)

|                                    | MIKE / Strukturmodell  | MIKE / KARL                  | MIKE / DesignKARL  | CoMo-Kit  |
|------------------------------------|--|------------------------------|--|---|
| Zeitaspekte:                       | ---  | ---                          | ---  | ---   |
| Parallelität:                      | ---  | ---                          | ---  | Ja  |
| deklarative Aspekte (z.B. Regeln): | ---  | L-KARL (Domäne / Inferenzen) | L-KARL, Datenstrukturen  | in Vorbereitung   |
| prozedurale Aspekte:               | Sequenz, Iteration, Alternative  | P-KARL (Kontrollwissen)      | P-KARL, Algorithmen  | ---   |
| Interaktion mit Benutzer:          | ---  | ---                          | ---  | GUI-Builder   |
| Interaktion mit Systemumgebung:    | ---  | ---                          | ---  | Anbindung von Tools   |
| vages / unsicheres Wissen:         | ---  | ---                          | ---  | ---   |
| Ausnahmebehandlung:                | ---  | ---                          | ---  | integraler Bestandteil der Methodik   |
| Agenten:                           | ---  | ---                          | ---  | Rollen, Qualifikationen, Aufbauorganisation   |
| nicht-funktionale Aspekte:         | NFR-Kontext: Effizienz, Wartbarkeit / Erweiterbarkeit/Flexibilität, Verständlichkeit, Robustheit, Portabilität, Wiederverwendbarkeit, Anforderungen von Systemarchitektur und Umgebung | ---                          | Effizienz, Wartbarkeit/ Erweiterbarkeit / Flexibilität, Verständlichkeit, Robustheit, Portabilität, Wiederverwendbarkeit, Anforderungen von Systemarchitektur und Umgebung | Abhängigkeitsverwaltung im Scheduler (Interpreter) ermöglicht die Behandlung von Ausnahmesituationen und Modellierungsfehlern |

6.2.6 Beschreibbarkeit von Aspekten (Sprachprimitive angeben)

|                                    | INCOME / IBEA   | INCOME / NR/T-Netze  | INCOME / SDM   | MORE / RT  |
|------------------------------------|---|--|--|--|
| Zeitaspekte:                       | Glossareintrag  | Uhren  | ---  | Zeitdiagramme, Timerobjekte  |
| Parallelität:                      | ---   | Netzstruktur   | ---  | zwischen Objekten  |
| deklarative Aspekte (z.B. Regeln): | ---   | Checkpoint-Transitionen, Checkpoint-Transitionfolgen                 | Aggregation, Spezialisierung / Generalisierung, Gruppierung, Beziehungen, Kardinalitäten | Aggregation, Vererbungsstruktur, Assoziation   |
| prozedurale Aspekte:               | Glossareintrag  | Netzstruktur   | ---  | ---  |
| Interaktion mit Benutzer:          |   | indirekt   | ---  | über Systemanalytiker  |
| Interaktion mit Systemumgebung:    | Glossareintrag  | indirekt   | ---  | über Schnittstellenobjekte   |
| vages / unsicheres Wissen:         | Glossareintrag  | ---  | ---  | ---  |
| Ausnahmebehandlung:                | Glossareintrag  | Ausnahmebedingung und Ausnahmebehandlungsnetz                        | ---  | Ausnahmebedingung und -behandlung in Objektzustandsgraphen   |
| Agenten:                           | ---   | ---  | ---  | ---  |
| nicht-funktionale Aspekte:         | Glossareinträge für geschätzte Mengen möglicher Datenbestände, Häufigkeit der Durchführung bestimmter Operationen | Sicherheit, Zuverlässigkeit, Effizienz, Kosten indirekt beschreibbar | ---  | Zeitaspekte können formal spezifiziert werden, andere nicht-funktionale Aspekte können in unstrukturierter Form beschrieben werden |

|                           | MIKE / Strukturmodell                | MIKE / KARL | MIKE / DesignKARL  | CoMo-Kit |
|---------------------------|--------------------------------------|-------------|--|----------|
| wann / wie berücksichtigt | formalisiert zu DesignKARL im Design | ---         | dokumentieren im DesignKARL-Prozeßmodell Entscheidungen für das DesignKARL-Produktmodell während des Designs | ---      |

### 6.2.7 Werkzeugunterstützung zur Modellierung

|  | MIKE / Strukturmodell                              | MIKE / KARL              | MIKE / DesignKARL | CoMo-Kit   |
|--|--|--------------------------|-------------------|--|
|  | grafische Editoren zur Erstellung und Modifikation | KARL-Editor im MIKE-Tool | ---               | Modellierungskomponente erlaubt die Spezifikation von Daten, Aufgaben, Relationen, GUI-Builder zum Editieren der Modelle |

### 6.2.8 Validierbarkeit des Ergebnismodells

|                             | MIKE / Strukturmodell                | MIKE / KARL                                | MIKE / DesignKARL  | CoMo-Kit                             |
|-----------------------------|--------------------------------------|--|--|--------------------------------------|
| Bezugspunkt der Validierung | Problemlösungsfähigkeit des Experten | Experte                                    | Experte, Anwender  | Protokolle bzw. Experte              |
| Methode zur Validierung     | exploratives Prototyping (Animation) | Testen, Kommunikation (Zwischenergebnisse) | experimentelles Prototyping durch hybriden Prototyp (vorgesehen) | Plausibilitätsprüfung durch Anwender |
| Werkzeugunterstützung       | MIKE-Tool                            | KARL-Interpreter / Debugger                | ---  | Scheduler (Interpreter)              |

|                         | INCOME / IBEA | INCOME / NR/T-Netze | INCOME / SDM | MORE / RT |
|-------------------------|---------------|---------------------|--------------|-----------|
| wann/wie berücksichtigt | ---           | ---                 | ---          | ---       |

### 6.2.7 Werkzeugunterstützung zur Modellierung

|  | INCOME / IBEA      | INCOME / NR/T-Netze  | INCOME / SDM  | MORE / RT   |
|--|--------------------|--|---|---|
|  | Editor, Dictionary | Grafische Editoren, Simulator, Dictionary, Tool für grafische Anfragen | Grafische Editoren, Dictionary, Generatoren, Analysewerkzeuge | grafische Editoren, zentrale DB, Konsistenzprüfung, Animation |

### 6.2.8 Validierbarkeit des Ergebnismodells

|                             | INCOME / IBEA   | INCOME / NR/T-Netze   | INCOME / SDM   | MORE / RT                                     |
|-----------------------------|---|---|--|---|
| Bezugspunkt der Validierung | natürlichsprachliche Beschreibungen durch befragte Personen               | informale Anforderungen der Auftraggeberseite               | informale Anforderungen der Auftraggeberseite                      | dokumentierte Anforderungen des Auftraggebers |
| Methode zur Validierung     | Plausibilitäts-/Vollständigkeitsprüfungen, Vorgehensmodell, Dokumentation | Simulation, Prototyping                                     | Prototyping, Simulation / Animation in Verbindung mit Petri-Netzen | Animation des Systemmodells                   |
| Werkzeugunterstützung       | Beziehungsmatrizen  | Simulator, Tool für grafische Anfragen an Simulationenläufe | Simulator, Formulargenerator, Cross-Referenz-Matrizen              | Animation                                     |

### 6.2.9 Verifizierbarkeit des Ergebnismodells

|  | MIKE / Strukturmodell | MIKE / KARL | MIKE / DesignKARL | CoMo-Kit |
|--|-----------------------|-------------|-------------------|----------|
|  | ---                   | ---         | ---               | ---      |

### 6.2.10 Wartbarkeit des Ergebnismodells

|  | MIKE / Strukturmodell   | MIKE / KARL  | MIKE / DesignKARL | CoMo-Kit   |
|--|---|--|-------------------|--|
| Methode zur Fehlererkennung / -beseitigung     | Kommunikation, Evaluation, exploratives Prototyping (Animation) | Ausführung, exploratives Prototyping, Testen (Animation) | ---               | Abwickeln der Modelle                            |
| Methode zur Adaption (an andere Anforderungen) | Kommunikation, Evaluation, exploratives Prototyping (Animation) | Ausführung, exploratives Prototyping, Testen (Animation) | ---               | Umplanung, Modelländerung                        |
| Methode zur Verbesserung (z.B. Effizienz)      | Kommunikation, Evaluation, exploratives Prototyping (Animation) | Ausführung, exploratives Prototyping, Testen (Animation) | ---               | Umplanung, Modelländerung                        |
| Werkzeugunterstützung                          | MIKE-Tool (Animation)   | KARL-Interpreter / Debugger                              | ---               | Modellierungskomponente, Scheduler (Interpreter) |

### 6.2.9 Verifizierbarkeit des Ergebnismodells

|                              | INCOME / IBEA                   | INCOME / NR/T-Netze                   | INCOME / SDM | MORE / RT |
|------------------------------|---------------------------------|---------------------------------------|--------------|-----------|
| Bezugspunkt der Verifikation | Glossare                        | Spezifikation von Systemeigenschaften | ---          | ---       |
| Methode zur Verifikation     | Glossaranalyse (lokal / global) | Erreichbarkeitsanalysen               | ---          | ---       |
| Werkzeugunterstützung        | Beziehungsmatrizen              | Analyse-Tools                         | ---          | ---       |

### 6.2.10 Wartbarkeit des Ergebnismodells

|  | INCOME / IBEA  | INCOME / NR/T-Netze  | INCOME / SDM  | MORE / RT                               |
|--|--|--|---|---|
| Methode zur Fehlererkennung/-beseitigung       | Plausibilitäts- / Vollständigkeitsprüfungen                    | Simulation, Prototyping  | Prototyping, Simulation / Animation, Analysetechniken | Animation des Systemverhaltens          |
| Methode zur Adaption (an andere Anforderungen) | ---  | Simulation   | ---   | ---                                     |
| Methode zur Verbesserung (z.B. Effizienz)      | ---  | Simulation   | Analysetechniken                                      | ---                                     |
| Werkzeugunterstützung                          | Dictionary, Formulareditor, basierend auf SQL-Forms (Prototyp) | Simulator, Tool für grafische Anfragen an Simulationsläufe, Dictionary | Dictionary  | Berechnung von Antwortzeiten, Animation |

6.2.11 Qualitätsmerkmale (Metriken) für das Ergebnismodell

|  | MIKE / Strukturmodell | MIKE / KARL | MIKE / DesignKARL | CoMo-Kit |
|--|-----------------------|-------------|-------------------|----------|
|  | ---                   | ---         | ---               | ---      |

6.2.12 Wiederverwendung

|  | MIKE / Strukturmodell   | MIKE / KARL   | MIKE / DesignKARL | CoMo-Kit                       |
|--|---|---|-------------------|--------------------------------|
| Welche Modellteile können wiederverwendet werden:                | als Template: komplette Strukturierung einer generischen Beschreibung einer Problemlösungsmethode | Bibliothek von formalen Modellen (Ontologien und generischen Problemlösungsmethoden) und Teilen davon | ---               | Alle, außer den Protokollen    |
| Methode zur Auswahl/Retrieval/Anpassung wiederverwendbarer Teile | Beschreibung über Features, Matching der Beschreibung gegen Bibliothek von Templates              | interaktiv  | ---               | Case Based Reasoning (geplant) |
| Werkzeugunterstützung  | MIKE-Tool (grafische Feature-Tabelle und Matching)  | ---   | ---               | ---                            |

6.2.11 Qualitätsmerkmale (Metriken) für das Ergebnismodell

|  | INCOME / IBEA  | INCOME / NR/T-Netze                              | INCOME / SDM | MORE / RT                                    |
|--|--|--|--------------|--|
|  | Vollständigkeit, Widerspruchsfreiheit, Redundanzfreiheit | Lebendigkeit, Deadlocks, Durchlaufzeiten, Kosten | ---          | Struktur und Zeitverhalten des Systemmodells |

6.2.12 Wiederverwendung

|  | INCOME / IBEA  | INCOME / NR/T-Netze  | INCOME / SDM                               | MORE / RT  |
|--|--|--|--|--|
| Welche Modelteile können wiederverwendet werden:                 | einzelne Anforderungssammelblätter / Glossare, wenn Systeme mit verwandter Funktionalität realisiert werden sollen | Referenznetze für bestimmte Anwendungen, Kleine „Standard“-Netzbausteine | Referenzschemata für bestimmte Anwendungen | Klassen, Aggregatstrukturen und Teilmodelle                              |
| Methode zur Auswahl/Retrieval/Anpassung wiederverwendbarer Teile | ---  | Delta-Analyse  | Delta-Analyse                              | werkzeugunterstütztes Kopieren von Elementen verschiedener Systemmodelle |
| Werkzeugunterstützung  | ---  | ---  | ---  | grafische Editoren   |

## 7. Resümee

Die Lehrbücher [Som92], [SoK93], [DKS93] bzw. die einschlägige Literatur der drei Gebiete Software Engineering, Information System Engineering und Knowledge Engineering drücken im allgemeinen eine strikte Trennung der drei Gebiete aus. So findet man darin nur sehr selten Querverweise zu den jeweils benachbarten Gebieten. Dies hat zum einen historische Ursachen und zum anderen liegt dies sicher in der oft sehr strikten Trennung der Gruppen, die jeweils in den unterschiedlichen Gebieten Methoden entwickeln und einsetzen, begründet. So existieren weder auf der Forschungsseite, noch auf der Entwicklungsseite, noch auf der Anwenderseite etablierte gemeinsame Foren, in denen Erkenntnisse der unterschiedlichen Gruppen ausgetauscht werden könnten.

In die betriebliche Praxis finden überwiegend Methoden des Software Engineering Eingang. In speziellen Fällen werden auch Methoden des Information System Engineering, z.B. zur Beschreibung betrieblicher Abläufe oder zum Entwurf von betrieblichen Informationssystemen, angewandt. Methoden des Knowledge Engineering haben, auch zum Bau wissensbasierter Systeme, bisher praktisch keinen Eingang in die betriebliche Praxis gefunden. Hier werden Systeme also noch am häufigsten weitgehend auf der Codeebene, also ohne Einsatz einer Methodik für die Analyse und den Entwurf, entwickelt. Allerdings ist in der Praxis auch noch keine vollständige Durchdringung des Einsatzes von entsprechenden Methoden in den anderen Gebieten zu verzeichnen.

Natürlich weisen die einzelnen Methodiken, und das zeigte sich auch im Verlaufe der Arbeit dieser Arbeitsgruppe, Charakteristiken auf, die speziell auf die jeweiligen Anwendungsgebiete zugeschnitten sind.

So erlauben Methoden des Software Engineering im Gegensatz zu den Methoden der anderen Gebiete die Entwicklung und Beschreibung komplexer Algorithmen, wie z.B. Fourieranalyse o.ä. Oft lassen sich Realzeitaspekte, wie sie z.B. für Systeme zur Prozeßkontrolle benötigt werden, beschreiben. Bei Methodiken, die eng mit der Strukturierten Analyse verwandt sind, wird ein System ausgehend von den Prozessen und Algorithmen, also der Dynamik des Systems, entwickelt. Bei objektorientierten Methodiken stehen dagegen die Daten und deren Struktur im Vordergrund. Letztere bieten deshalb auch umfangreichere Methoden zur Beschreibung der statischen Strukturen. Diese sind deshalb näher verwandt mit Methoden des Information System Engineering und gingen sogar zum Teil aus deren Methoden hervor. Im Bereich der statischen Modellierung ist deshalb eine Trennung zwischen den Methoden des Information System Engineering und objektorientierten Methoden weitgehend aufgehoben. Objektorientierte Methoden eignen sich sehr gut zur Entwicklung diskreter Systeme, bei denen in jedem Schritt die Kenntnis der Zustände nur sehr weniger Objekte notwendig ist und bei denen die Ergebnisse nur auf einfache Art mit den auftretenden Ereignissen verknüpft sind. Strukturierte Ansätze scheinen dagegen besser für komplexe Algorithmen, für Probleme, bei denen das Resultat vom Zusammenspiel vieler Objekte (z.B. globale Optimierungsprobleme), abhängt und für Probleme hoher kombinatorischer Komplexität (z.B. Zuordnungsprobleme) geeignet zu sein. In den gängigen Software Engineering Methoden werden bis auf das realisierte System nur semiformale Sprachen zur Beschreibung von Modellen eingesetzt. Dies hat zur Konsequenz, daß solche Modelle weder formalen Methoden, wie z.B. Verifikationsmethoden, zugänglich sind, noch daß sie zu ihrer Validierung ausgeführt werden können.

Methoden des Information System Engineering bieten insbesondere ausgefeilte Methoden zur Beschreibung der statischen Struktur (Schemabeschreibungen) der Daten an. Modellierungsprimitive aus den semantischen Datenmodellen ermöglichen die Beschreibung vielfältiger Bezie-

hungstypen zwischen einzelnen Klassen (Entitytypen) von Objekten. Bei der Beschreibung der Dynamik von Systemen stehen hier oft Verteilungsaspekte im Vordergrund, wie sie z.B. zur Modellierung betrieblicher Abläufe benötigt werden. Damit einher gehen die Beschreibbarkeit paralleler Abläufe sowie temporaler Aspekte. Während Nebenläufigkeit und Verteilung im Information System Engineering meistens konzeptuelle Aspekte darstellen, werden diese Aspekte im Software Engineering oft erst während des Systementwurfs näher betrachtet. Im Gegensatz zum Software Engineering eignen sich die Beschreibungsformalismen nicht zur Darstellung komplexer, z.B. arithmetischer Algorithmen. Bei der Entwicklung werden innerhalb der Analyse und des Entwurfs oft früh formale Sprachen eingesetzt, die es erlauben, Modelle durch Simulation validieren zu können.

Im Knowledge Engineering werden im Gegensatz zu den anderen Gebieten das Wissen über den konkreten Anwendungsbereich und das Wissen über den Ablauf der Problemlösung, die sog. Problemlösungsmethode, klar getrennt. Die Problemlösungsmethode wird generisch, d.h. unabhängig vom konkreten Anwendungsbereich formuliert. Dies soll zur Wiederverwendung sowohl von Problemlösungsmethoden als auch von Teilen des anwendungsbereichspezifischen Wissens beitragen. Zur Modellierung der Struktur der Daten werden die Möglichkeiten des Information System Engineering noch um geeignete Primitive zur Beschreibung komplexer axiomatischer Zusammenhänge (Regeln und Konsistenzbedingungen) ergänzt. Im Gegensatz zu objektorientierten Methoden des Software Engineering sind Problemlösungsmethoden insbesondere für Probleme, bei denen das Resultat vom Zusammenspiel vieler Objekte abhängt, und für Probleme hoher kombinatorischer Komplexität geeignet. Sie eignen sich deshalb insbesondere zur Modellierung entscheidungsunterstützender Systeme. Allerdings eignen sich die Beschreibungsmittel, genauso wie beim Information System Engineering, nicht zur Darstellung komplexer Algorithmen, wie z.B. arithmetischer Algorithmen. Verteilungs- und Zeitaspekte spielen hier praktisch keine Rolle. Ebenso wie im Information System Engineering werden bei der Entwicklung frühzeitig formale Beschreibungsmittel eingesetzt.

Bei näherer Betrachtung der in den unterschiedlichen Gebieten entwickelten Methoden stellt sich allerdings auch heraus, daß die o.g. strikte Trennung der Gebiete nicht gerechtfertigt ist. Es existieren sehr viele Methoden, die jeweils mit leichten Variationen in den unterschiedlichen Methodiken verwendet werden. Allen Entwicklungsmethodiken gemeinsam ist die sukzessive Entwicklung unterschiedlicher Abstraktions- und Formalisierungsebenen, die den Übergang von der rein informalen Beschreibung der Funktion und der Anforderungen an das System bis hin zum realisierten System in mehrere Schritte unterteilen. Beim Information Systems Engineering und beim Knowledge Engineering werden darin auch frühzeitig formale Beschreibungsformalismen angewandt, die die Validierung durch Ausführung bzw. den Zugang für formale Verfahren ermöglichen. Gegenüber rein linearen Vorgehensweisen bei der Entwicklung, wie z.B. durch das klassische Wasserfallmodell beschrieben, scheint es allgemeiner Konsens zu sein, durch Methoden des Prototyping frühzeitig Rückkopplung der erzeugten Modelle gegenüber den tatsächlichen Anforderungen zu gewinnen. Allerdings herrschen in der betrieblichen Praxis meist lineare Vorgehensweisen vor, da z.B. die Überprüfbarkeit von Meilensteinen vertraglichen Vereinbarungen entgegen kommt. Bei den einzelnen Beschreibungsformalismen nähern sich insbesondere die Sprachen zur Beschreibung statischer Strukturen weitgehend aneinander an. Diese Formalismen bieten meist die Möglichkeiten, wie sie für semantische Datenmodelle entwickelt wurden. Vererbungsstrukturen und unterschiedliche Beziehungstypen zwischen Klassen (Entitytypen) sind hier die Regel. Bei der Beschreibung der Dynamik eines Systems werden in allen Gebieten hierarchische Beschreibungsformalismen verwendet, um die Gesamtkomplexität eines Systems besser beherrschbar zu machen. Zur Beschreibung der Dynamik werden im Knowledge Engineering oft Methoden aus dem Software Engineering eingesetzt, die an die Notwendigkeiten des Knowledge Engineering adaptiert wur-

den.

Es ist klar, daß innerhalb der Arbeitsgruppe nur bestimmte Aspekte der unterschiedlichen Methodiken verglichen werden konnten. So blieben z.B. die Entwicklung von Benutzungsoberflächen, das gesamte Thema Projektmanagement, weitergehende formale Ansätze, wie z.B. Verifikationsmethoden, gänzlich unbetrachtet. Die Einbeziehung aller möglichen Aspekte war aus Zeit- und Aufwandsgründen nicht möglich. So wurden die Betrachtungen auf den jeweiligen Kern einer Entwicklungsmethodik eingeschränkt.

Jedes der drei Gebiete füllt seine speziellen Anforderungen mit entsprechenden Entwicklungsmethoden und Beschreibungsformalisten aus. Es gibt jedoch sehr viele überlappende Bereiche und Methoden. Somit können alle drei Gebiete von einem Erfahrungsaustausch untereinander profitieren. Die Aufgaben, die Softwaresysteme in Zukunft lösen werden, werden, auch gefördert durch Vernetzung und Verteilung, immer komplexer. Es ist deshalb zu erwarten, daß solche komplexen Gesamtsysteme Anteile enthalten werden, die am besten mit Methoden des Software Engineering entwickelt werden. Andere Anteile werden besser den Methoden des Information Engineering zugänglich sein und wiederum andere Anteile werden wissensbasierte Charakterzüge aufweisen. Es sollten deshalb in Zukunft diese einzelnen Bereiche nicht mehr nur isoliert betrachtet werden. Statt dessen sollten die Anforderungen von solchen komplexen Systemen als Zusammenspiel dieser unterschiedlicher Anteile auch bei der weiteren Entwicklung der Methodiken berücksichtigt werden. Dies bezieht sich allerdings nicht nur auf die inhaltlichen Aspekte, sondern es müssen auch geeignete Organisationsformen geschaffen werden, um den Erfahrungsaustausch der unterschiedlichen Gruppen aus den unterschiedlichen Gebieten zu fördern. Mit dieser Arbeitsgruppe wurde dazu ein erster Schritt getan. Darüber hinaus sollten in Organisationen, wie z.B. der GI, die strikte Abgrenzung und hierarchische Strukturierung der Fachgebiete durch eine flexiblere Struktur überwunden werden, um Kommunikation und Kooperation zwischen den unterschiedlichen Gruppierungen zu fördern. Darüber hinaus sollten auch innerhalb der Lehre in den einzelnen Gebieten Aspekte aus den anderen Gebieten sehr viel stärker einfließen, um frühzeitig ein Verständnis für die jeweils andere Fachrichtung und für die Integration der Ansätze zu etablieren.

### **Danksagung**

Die Autoren danken allen weiteren (zeitweiligen) Mitgliedern der Arbeitsgruppe für ihre konstruktive Mitarbeit: H.J. Cleef, D. Landes, M. Leppert, P. Löhr-Richter, S. Neubert, H.J. Ott, Th. Pirlein, M. Riebisch, G. Schöpke, A. Schwanke, B. Thelen und H. Voß.

### **Literatur**

- [DKS93] J.-M. David, J.-P. Krivine, R. Simmons (eds.): *Second Generation Expert Systems*. Springer, 1993.
- [SoK93] A. Solvberg, D.C. Kung: *Information Systems Engineering*. Springer, 1993.
- [Som92] I. Sommerville: *Software Engineering*. Addison-Wesley, 1992.

## Anhang

### Fallstudie: Wissensbasiertes Informationssystem zur Tagungsorganisation

Es ist ein wissensbasiertes Informationssystem zur Unterstützung bei der Organisation einer Tagung zu realisieren. Der grobe Ablauf einer Tagung ist wie folgt vorgegeben:

Eine Tagung wird von einer oder mehreren Organisationen, z.B. der Gesellschaft für Informatik (GI) oder der Association for Computing Machinery (ACM), veranstaltet. Zu einer Tagung wird zunächst die TagungsleiterIn, der Tagungsort und der Tagungstitel festgelegt sowie das Organisationskomitee gebildet. Es wird eine LeiterIn für das Programmkomitee bestimmt, die dann - in Absprache mit der TagungsleiterIn - die Mitglieder des Programmkomitees auswählt.

Das Programmkomitee kümmert sich um die Zusammenstellung des Programms und organisiert dazu zunächst einen „Call for papers“, der an mögliche InteressentInnen verschickt wird. Personen, die einen Vortrag halten möchten, reichen bis zu einem vorgegebenen Termin drei Kopien eines Manuskripts bei der LeiterIn des Programmkomitees ein. Diese ordnet die Manuskripte einem Fachgebiet zu und verteilt sie möglichst gleichmäßig auf die Mitglieder des Programmkomitees, die für die Begutachtung zuständig sind. Dabei sind verschiedene Anforderungen zu beachten. Ein Manuskript, dessen AutorIn aus dem Umfeld eines Programmkomiteemitglieds stammt, darf diesem nicht zur Begutachtung gegeben werden. Das Programmkomiteemitglied muß aus demselben oder einem angrenzenden Fachgebiet stammen, das dem Manuskript zugeordnet wurde. An einem bestimmten Termin findet eine Programmkomiteesitzung statt, bei der die besten Vorträge ausgewählt werden und zu einem Programm, bestehend aus einer Menge von Sitzungen mit einer festgelegten Anzahl von eingereichten und jeweils einem eingeladenen Vortrag, zusammengestellt werden. Eingeladene Vorträge werden von renommierten WissenschaftlerInnen gehalten, die von dem Programmkomitee vorgeschlagen werden. Zur zeitlichen Gliederung des Programms gibt es folgende Einschränkungen: Die Sitzungen sollen gleichmäßig auf die Konferenztage verteilt werden, ohne daß Lücken - mit Ausnahme von Kaffee- und Mittagspausen - entstehen. Es sollen so wenig parallele Sitzungen wie möglich stattfinden, und es sollen möglichst keine verwandten Sitzungen parallel stattfinden.

Die LeiterIn des Programmkomitees verschickt Zusagen und Absagen an die Personen, die Manuskripte zur Tagung eingereicht haben, und ernennt für die einzelnen Sitzungen der Tagung SitzungsleiterInnen aus dem Kreis des Programmkomitees. Dabei dürfen keine parallelen Sitzungen mit derselben SitzungsleiterIn belegt werden, und eventuell gegebene zeitliche Restriktionen der SitzungsleiterInnen müssen berücksichtigt werden. Den AutorInnen der akzeptierten Manuskripte wird ein Termin mitgeteilt, an dem die endgültige Fassung für den Tagungsband vorliegen muß. Bei Bedarf muß die ProgrammkomiteeleiterIn die Endfassung anmahnen. Die ProgrammkomiteeleiterIn muß sich um einen Verlag kümmern, der den Druck der Tagungsbände übernimmt. HerausgeberIn des Tagungsbandes ist die LeiterIn des Programmkomitees.

Das Organisationskomitee legt eine Teilnahmegebühr für die Tagung fest. Es ist verantwortlich für das Drucken von Programmen, die Versendung von Einladungen für die Tagung, die Verwaltung der An- und Abmeldungen zur Tagung, die Verbuchung der eingehenden Tagungsgebühren (evtl. mit Mahnung), die Hotelreservierungen, die Saalvergabe für die einzelnen Sitzungen, das Drucken von Teilnehmerlisten und Namensschildern, die Anmeldung eines Tagungstelephons sowie die Bildung eines Tagungsbüros. An einer Konferenz nehmen in- und ausländische Personen teil. Es sollen unterschiedliche Tagungen verwaltet werden, entsprechend stellen die Anzahl der Programmkomiteemitglieder, die Anzahl der Vorträge pro Sitzung, die Anzahl der eingeladenen Vorträge, die Vortragsdauern usw. veränderliche Größen dar.

# Dissertationen

## HSM - Methode und Werkzeug zur modellbasierten Hypertext-Entwicklung

Sigrun Goll, Marburg

### **Einleitung**

An der Philipps-Universität Marburg wurde im Rahmen einer Dissertation eine universelle Modellierungsmethode für Hypertext erstellt, die basierend auf den semantischen Datenmodellen die Kombination hierarchischer und Netzstrukturen unterstützt und darüber hinaus die Definition rekursiver Strukturen ermöglicht.

Die Ziele bei der Entwicklung der HSM-Methode (Hypertext mit semantischen Datenmodellen) lagen in der Konzeption einer einfachen, intuitiv verständlichen und zugleich mächtigen Strukturierungsmethode, um dem vielzitierten „Löst in Hypertext“-Syndrom entgegenzuwirken. Hierzu wurden Strukturierungsmechanismen konzipiert, mit deren Hilfe ein Autor ein Modell seiner Hypertext-Anwendung erstellen kann. Für die graphische Spezifikation von HSM-Modellen steht dem Autor ein Modellierungswerkzeug zur Verfügung. Ein solches Modell soll einem künftigen Leser einen Gesamtüberblick über die Anwendung geben.

### **Die Konzepte der HSM-Methode**

Die wesentlichen Konzepte von HSM bilden die drei Abstraktionsmechanismen *Aggregation*, *Gruppierung* und *Netzwerk-Assoziation*.<sup>1</sup> Die Aggregation ist vergleichbar mit der herkömmlichen hierarchischen Strukturierungsform, wie sie sich in jedem Inhaltsverzeichnis eines Buchs präsentiert: Die Untergliederung in Kapitel und Unterabschnitte. Ein Aggregat setzt sich zusammen aus einem Aggregationsknoten, der über hierarchische Links mit einer Menge von Nachfolgern verbunden ist.

Die Gruppierung unterstützt die Übersichtlichkeit eines Modells, da sich mit ihr repetitive Strukturen angemessen darstellen lassen. Ein Autor kann sie verwenden, wenn er gleichartige „Dinge“ durch analoge Strukturen beschreiben möchte. Ein Gruppierungsknoten *gruppiert* entweder einfache Knoten oder komplexe Strukturen.

Die Netzwerk-Assoziation stellt insofern den Kernpunkt der Methode dar, da sie der Repräsentation von semantischen Beziehungen zwischen Informationen dient und gleichzeitig die Bildung rekursiver Strukturen unterstützt. Ein *Graphknoten* innerhalb eines HSM-Modells repräsentiert eine unterliegende Netzstruktur, die aus Knoten und (bezeichneten) Links besteht. Innerhalb der Netzstruktur können wiederum Graphknoten auftreten, so daß eine beliebige Anzahl von geschachtelten Strukturen erstellt werden kann.

### **Orthogonale Anwendbarkeit**

Ein besonderer Vorteil der HSM-Methode liegt darin, daß die zugehörigen Operationen (Aggregieren, Gruppieren und Assoziieren) orthogonal anwendbar sind. Dies bedeutet, daß alle drei Operationen auf beliebigen Knotentypen und Strukturen ausgeführt werden können. So können beispielsweise Netzstrukturen und/oder gruppierte Strukturen aggregiert werden oder aggregierte Strukturen Bestandteil einer Netzstruktur sein.

---

<sup>1</sup> Eine ausführlichere intuitive Beschreibung und eine formale Definition der Methode findet sich in S. Goll „Methode und Werkzeug zur modellbasierten Hypertext-Entwicklung“, Görlich & Weiershäuser GmbH, Marburg 1996

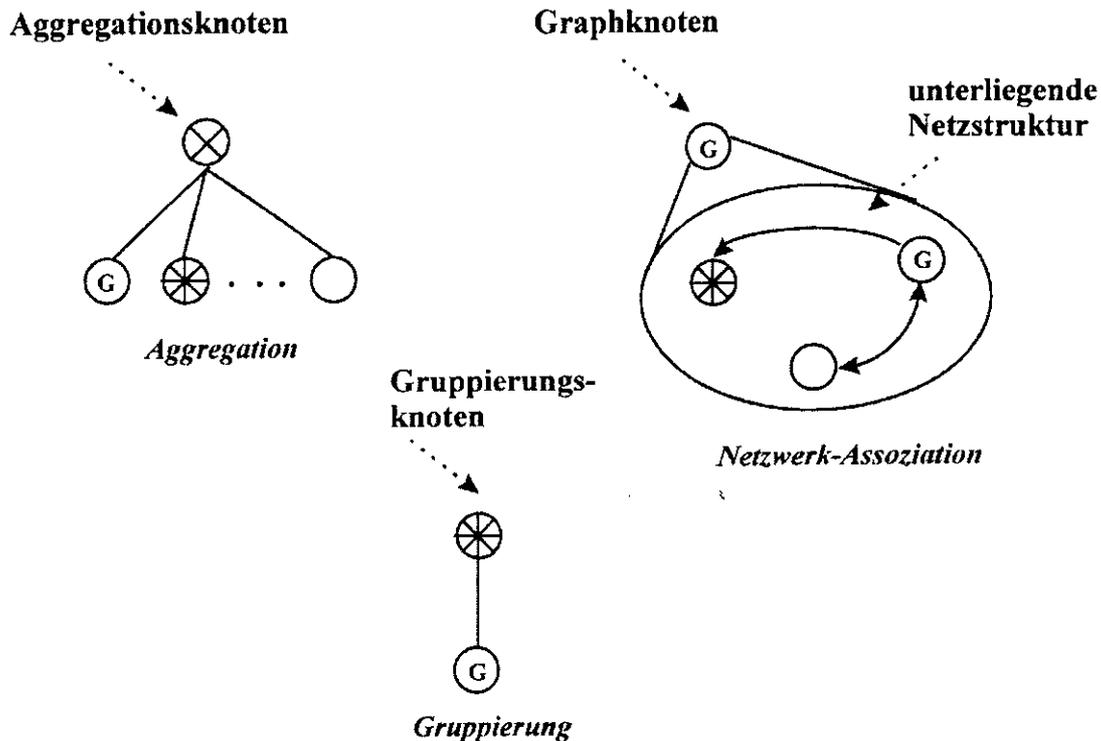


Abbildung 1: Graphische Repräsentation der Aggregation, Gruppierung und Netzwerk-Assoziation

### Verfeinerung und Abstraktion

Der Hypertext-Autor kann bei dem Entwurf eines Modells die Top down- oder die Bottom up-Vorgehensweise wählen. Typischerweise wird der Autor zunächst die grundlegenden Bestandteile seines Modells - in einer Beispielanwendung „Projekt“ könnten dies *Aufgabenstellung*, *Projektplanung*, *Projektleiter* und *Mitarbeiter* sein - und deren Beziehungen festlegen und anschließend detailliertere Strukturen spezifizieren (Top down).

Die HSM-Methode sieht - wie bereits erwähnt - auch vor, daß spezifizierte Strukturen abstrahiert werden können. So könnte eine Anwendung *Projekt* beispielsweise durch die Operation *Assoziieren* in ein übergeordnetes Modell *Unternehmensbereiche* eingebunden werden (s. Abbildung 2).

### Werkzeugunterstützung

Für die Erstellung von HSM-Modellen wurde ein HSM-Editor (Modellierungswerkzeug) entwickelt, der die Modellkomponenten (Knoten und Links) mit ihren semantischen und graphischen Eigenschaften verwaltet. Der Autor wird durch eine geeignete Benutzerführung bei der Top down- und Bottom up-Vorgehensweise unterstützt. Die Verfeinerung eines Graphknotens erfolgt in einem separaten Fenster, in dem die gesamte Funktionalität des HSM-Editors zur Verfügung steht. Die spezifizierte Netzstruktur kann über eine *include* Operation in das übergeordnete Modell eingefügt werden. *Shrink* und *expand* Operationen dienen dazu, die Übersichtlichkeit des Modells zu verbessern.

Ein HSM-Modell kann automatisch in eine Hypertext-Struktur transformiert werden, die in ein Hypertext-System (Präsentationswerkzeug) eingelesen werden kann. Hierzu wurden Transformationsvorschriften spezifiziert, die für jede Modellkomponente angeben, in welche (Hypertext-) Datenstruktur sie übersetzt werden soll. Für gruppierte Strukturen kann der Autor bei der Modellierung angeben, wie häufig sie reproduziert werden sollen. Das Ergebnis der Transformation ist eine Hypertext-Struktur, die aus Knoten und (hierarchischen und bezeich-

neten) Links besteht. Mit Hilfe des Hypertext-Systems kann ein Autor diese Struktur mit den konkreten Informationen seines Anwendungsgebietes füllen. Das Präsentationswerkzeug verfügt über zwei Modi, einen Baum- und einen Netzmodus. Steht der aktuelle Knoten in einer hierarchischen Beziehung, d. h. er besitzt einen Vorgänger und/oder Nachfolger, so wird diese (partielle) Hierarchie in separaten Listenfenstern angezeigt. Andernfalls ist lediglich der Inhalt des aktuellen Knotens sichtbar.

Beide Werkzeuge wurden in VisualWorks implementiert<sup>2</sup>. Um die HSM-Modellierungsmethode für World Wide Web - Anwendungen nutzen zu können, ist in einem nächsten Schritt die Realisierung einer HTML-Schnittstelle geplant.

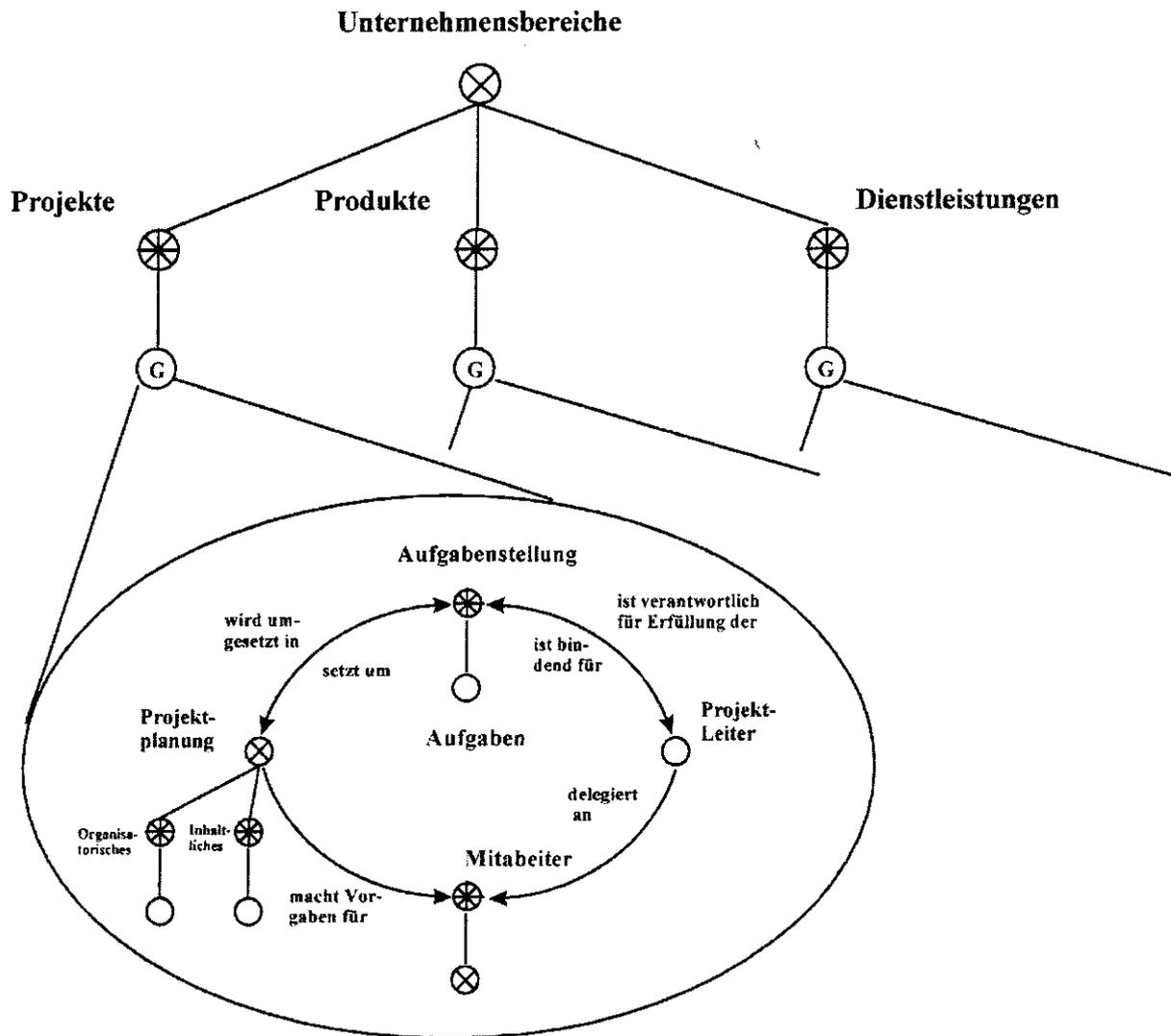


Abbildung 2: Integration eines Modells „Projekt“ in ein übergeordnetes Modell „Unternehmensbereiche“

**Adresse:**

Dr. Sigrun Goll  
 Savignystr. 9  
 35037 Marburg  
 Email: sg@informatik.uni-marburg.de

<sup>2</sup> Die Realisierung des HSM-Modellierungswerkzeuges wurde von Jürgen Lorenz im Rahmen der Diplomarbeit „Einsatz von objektorientierten Techniken zu Analyse, Entwurf und Implementierung eines Modellierungswerkzeuges für Hypertext-Anwendungen“, Marburg 1995, durchgeführt.

Fakultät für Informatik  
der Technischen Universität München

## Formale Methodik des Entwurfs verteilter objektorientierter Systeme<sup>1</sup>

*Bernhard Rumpe*

Von der Fakultät für Informatik der Technischen Universität München zur  
Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigte Dissertation.

Vorsitzender: Univ.-Prof. Dr. J. Eickel

Prüfer der Dissertation:

1. Univ.-Prof. Dr. M. Broy
2. Univ.-Prof. Dr. M. Paul

Die Dissertation wurde am 26.6.1996 bei der Technischen Universität  
München eingereicht und durch die Fakultät für Informatik am 12.12.1996  
angenommen.

---

<sup>1</sup> Die Dissertation ist als Buch beim Herbert Utz Verlag Wissenschaft München  
erschienen. Sie ist unter der ISBN 3-89675-149-2 im Buchhandel erhältlich. Sie kann auch  
über den Autor (Email:rumpe@informatik.tu-muenchen.de, Tel. 089-289-28129) bezogen  
werden.

# Kurzfassung

In dieser Arbeit wird eine formale Grundlage für eine objektorientierte Methodik entwickelt. Zur Spezifikation von Struktur und Verhalten verteilter objektorientierter Systeme werden Beschreibungstechniken für Objektmodelle, Klassen und Transitionssysteme definiert.

Für die zustandsbasierte Beschreibung nichtdeterministischen Komponentenverhaltens wird die Theorie buchstabierender Automaten entwickelt. Es werden eine konkrete Darstellungsform, eine abstrakte Syntax, eine denotationelle und eine operationelle Semantik angegeben und gezeigt, daß beide Semantiken übereinstimmen.

Für buchstabierende Automaten wird ein Verfeinerungskalkül definiert, der zur Transformation von abstrakten in detaillierte Verhaltensbeschreibungen verwendet werden kann. Es wird gezeigt, daß dieser Kalkül bezüglich der Semantikdefinition korrekt ist. Der Kalkül wird für die Spezialisierung und die Vererbung von Verhaltensbeschreibungen in verteilten objektorientierten Systemen eingesetzt.

Ein Systemmodell charakterisiert eine Menge von verteilten objektorientierten Systemen, die aus asynchron kommunizierenden Agenten aufgebaut sind. Das Systemmodell dient als Basis für die Definition einer integrierten, formalen Semantik für die oben genannten Beschreibungstechniken.

Methodische Entwicklungsschritte erlauben die Verfeinerung von Dokumenten dieser Beschreibungstechniken. Basierend auf der formalen Semantik der Verfeinerungsschritte werden präzise Aussagen über das Zusammenspiel der verwendeten Beschreibungstechniken definiert.

Die Verbindung graphischer Beschreibungstechniken mit einer integrierten, formalen Semantik nutzt Synergieeffekte formaler und praxisorientierter Ansätze der Softwaretechnik.

# Tagungsberichte

## Software-Vorgehensmodelle im Zentrum des Interesses

### **Bericht vom Workshop der Fachgruppe 5.1.1**

Eine Möglichkeit, die Qualität von Software Produkten zu verbessern, ist der Einsatz eines Vorgehensmodells, in dem die einzelnen Schritte der Software-Entwicklung und die dabei entstehenden Zwischen- und Endprodukte genau festgelegt werden. Dieser Ansatz wird auch von internationalen Standards wie ISO 900x und ISO 12207 unterstützt.

Das wachsende Interesse an diesem Thema bewies der 4. Workshop 'Vorgehensmodelle - Einführung, betrieblicher Einsatz, Werkzeug-Unterstützung und Migration' in Berlin-Adlershof, 17. und 18.3. 1997. Etwa 60 Teilnehmer und 6 ausstellende Firmen hörten 2 Tage lang Vorträge und angeregte Diskussionen zu dem Thema. Besondere Aufmerksamkeit erregte der Vortrag 'Vom Frust zur Lust bei der Einführung eines Vorgehensmodells (Frau Wiemers), die lebhaft und launig die Probleme der Einführung derartiger neuer Paradigmen in der Praxis beleuchtete. Gerhard Chroust beschrieb die Konfliktsituationen bei dem Zusammenspiel von CASE-Werkzeugen und Vorgehensmodellen.

Ein Schwerpunktthema des Workshops war die Werkzeugunterstützung für Vorgehensmodelle, die in mehreren Vorträgen behandelt und durch eine Werkzeug-Ausstellung ergänzt wurde.

Ein Tagungsband ist bei der GMD als GMD-Studie 311 erschienen und ist dort erhältlich.

Im Rahmen des Workshops wurde auch das Leitungsgremium der GI-Fachgruppe 5.1.1 ('Vorgehensmodelle für die betriebliche Anwendungsentwicklung') neu gewählt:

Neuer und alter Sprecher der Fachgruppe ist Günther Müller-Luschnat (FAST, München, [gml@fast.de](mailto:gml@fast.de)). Zum stellvertretenden Sprecher wurde Dr. Ralf Kneuper (Darmstadt, [ralf\\_kneuper@compuserve.com](mailto:ralf_kneuper@compuserve.com)) gewählt. Weitere Mitglieder des Leitungsgremiums sind Prof. Gerhard Chroust (Uni Linz), Thomas Fischer (Isar-Amper-Werke, München) und Rainer Midderhoff (IABG, München).

Der Erfolg der Tagung löste den Wunsch nach einer Fortsetzung aus: Im März 1998 findet der nächste Workshop statt, der sich besonders der Frage von Vorgehensmodellen im Kontext des objekt-orientierten Paradigmas widmen wird.

Dr. Ralf Kneuper, Darmstadt

Hansjürgen Paul (Hg.)

Projektberichte des Instituts Arbeit und Technik, Gelsenkirchen, 1997  
ISSN 1430-8738

## Modellierung von Aufbau- und Ablauforganisation: von der Technozentrik zur Anthropozentrik

Am 3. und 4. April 1996 war die Abteilung Produktionssysteme des Instituts Arbeit und Technik in Gelsenkirchen Gastgeber des Workshops *Modellierung von Aufbau- und Ablauforganisation: von der Technozentrik zur Anthropozentrik*. Der Workshop ist Bestandteil einer Veranstaltungsreihe, die von der Arbeitsgruppe »Modellierung in soziotechnischen Systemen« der Fachgruppe »Entwicklungsmethoden für Informationssysteme und deren Anwendung« der Gesellschaft für Informatik e.V. (GI) durchgeführt und von Prof. Dr. Erich Ortner (TH Darmstadt) und Dr.-Ing. Hansjürgen Paul (Institut Arbeit und Technik) moderiert wird.

Der Reader stellt eine Sammlung der Vorträge dar, die während des Workshops gehalten wurden. Dabei sind die Diskussionsergebnisse des Workshops aufgearbeitet und von den Autoren in die vorliegenden Texte eingearbeitet worden.

*Aus dem Inhalt:* Werkzeuge organisationalen Handelns? • Modellierung und Konzeption organisations- und aufgabenangemessener betrieblicher Anwendungssysteme • Ablaufmodellierung als Chance für partizipative Softwareentwicklung – Ein Erfahrungsbericht • Modellierungsnotationen für prospektive, gestaltungsorientierte Technikfolgenforschung • Fuzzy-basierte Modellierung unscharfer Informationen in der Entwicklungs- und Konstruktionsphase • Steuerungssituationen in Workflow-Management-Systemen • Der Ordnungsrahmen der Grundsätze ordnungsmäßiger Modellierung aus Sicht der Anthropozentrik • Modellieren in anthropozentrischer Sicht – die Perspektive betrieblicher Interessenvertretungen

H. Paul, 1997



Bestellcoupon für Direktbestellung. Bitte unterschrieben an das IAT senden oder Fax +49.(0)209.1707-110

Bitte senden Sie kostenlos

... Ex. **Modellierung von Aufbau- und Ablauforganisation... (IAT-PB 97-02)**

an:

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_ (Firmenstempel oder Anschrift in Druckbuchstaben)

Institut Arbeit und Technik  
im Wissenschaftszentrum NRW  
Z. Hd. Frau Doris Steinmann  
Munscheidstraße 14  
D-45886 Gelsenkirchen

# Buchbesprechungen

## **ORACLE7 Datenbanken erfolgreich realisieren Entwurf, Entwicklung, Tuning**

von Frank Roeing, Vieweg Verlag, 1996,  
266 Seiten, 98,- DM, ISBN 3-528-05521-9

Bei Datenbankprojekten kommen heute meistens relationale Datenbanksysteme zum Einsatz. Diese relationalen Systeme sind in den letzten Jahren wesentlich komplexer geworden. Das gilt besonders für das Produkt der Firma Oracle: Der Schritt von Version 6 nach Version 7 brachte beispielsweise Trigger, Datenbankprozeduren und die Unterstützung der referentiellen Integrität und von Check-Bedingungen.

Diese Komplexität wird heute vielleicht zu sehr durch Schlagworte wie Data Warehouse, WWW, OLAP oder Objektorientierung verdeckt. Im Zusammenspiel mit Datenbanksystemen entsteht die Gefahr verschiedener Welten, wobei es besonders mit der Objektorientierung schon irritierende Erfahrungsberichte gibt: So machte z.B. der Leiter eines objektorientierten Projekts die klassisch relational entworfene Datenbank für die schlechte Performanz verantwortlich, ein anderer glaubte sie dagegen nur dadurch gerettet zu haben, daß er Konzessionen an den objektorientierten Entwurf weitgehend wieder aus dem Datenmodell hinauswarf.

Etwas Klärung könnte man von Frank Roeings Buch erwarten, da der Autor in seinem Werk den „Bogen von der Konzeption bis zum Einsatz spannen“ wollte und der Kaiserslauterner Professor Theo Härder in seinem Geleitwort diese Verbindung von Modellierung und technischer Umsetzung als wesentlichen Vorzug des Buches darstellt. Zudem gibt der Autor EDV-Fortgeschrittene und Profis als Zielgruppen an.

Entäuschend dann das, was man nach der Einführung im Kapitel „Datenmodelle“ vorfindet: Das Übliche der zehn und mehr Jahre alten Datenbanklehrbücher wird hier eingeführt und erläutert, also das Entity-Relationship-Modell, wie man seine Entity-Typen findet, den Übergang zu Relationen und die Normalformenlehre. Stichworte wie OLAP und Data Warehouse werden nicht erwähnt, vielleicht sind die Ausführungen zur Modellierung von Zeit und Versionen eine Reaktion auf diese aktuellen Themen. Der Begriff Objektorientierung fällt häufiger, allerdings nur in einer mehr allgemeinen Form, etwa wenn in der Einführung eine „Philosophie der Objektorientierung“ ab 1990 diagnostiziert und damit eine größere Datenorientierung belegt wird.

Nach dem Kapitel „Datenmodelle“ behandelt das Kapitel „Entwurf“ die technische Umsetzung mit Oracle 7. Die Abbildung der Wertebereichtstypen wird unter intensivem Einsatz von Check-Bedingungen beschrieben, die grundlegenden Elemente wie Tabelle, View, Index und Tablespace mit den möglichen Parametrisierungen vorgestellt, Integritätsregeln von der Spalten- und Zeilenintegrität bis zur referentiellen Integrität klassifiziert und Trigger als weiteres Mittel der Integritätssicherung erläutert. Zwei Abschnitte widmen sich dem Modelltuning und dem technischen Tuning, dann wird noch auf Datenschutz, Data-Dictionary und verschiedene Stolpersteine eingegangen.

Die „Datenmodelle“ mit 85 Seiten und der „Entwurf“ mit 115 Seiten nehmen fast den gesamten Raum ein. Die folgenden drei Kapitel umfassen nur zwischen zehn und zwanzig Seiten: Die „Anwendungsentwicklung“ geht auf die Anwendungsverteilung, Transaktionen und wieder auf das Tuning ein, das „Reverse Engineering“ zeigt Schritte, mit denen man aus einem bestehenden System seine Oracle 7-Datenbank gewinnen kann, und im Kapitel „Oracle-Systeme“ finden sich abschließende Worte zu Wartbarkeit, Integrität, Performanz und unterstützenden Werkzeugen.

Das breite Programm setzt der Autor auf dem engen Raum didaktisch hervorragend um. Er behandelt den Stoff auf seinem roten Faden genau, während Abschweifungen entweder nicht vorkommen oder nur sehr lax ausgeführt werden. Schreiben kann der

Autor sehr gut, der Drucksatz ist vorbildlich und das gebundene Buch liegt angenehm in der Hand. An den zahlreichen Tips zur technischen Umsetzung und den Hinweisen zur Verbindung mit dem konzeptionellen Modell ist eigentlich nicht viel auszusetzen, eigentlich nur, daß die Oracle 7-Vorstellung vielleicht doch noch zu sehr bei Adam und Eva anfängt und so für den Oracle 7-erfahreneren Leser die interessanten Passagen dann doch immer nur zu kurz aufflackern und dann schnell wieder dem nächsten Thema Platz machen.

Dem Buch ist durchaus ein breites Publikum zu wünschen — dafür gibt zu viele Oracle 7-Datenbanken, für die auch die Mittel von Oracle 6 gereicht hätten. Den größten Nutzen könnten demnach Umsteiger von anderen Systemen oder von Oracle 6 erlangen, weiter diejenigen, die zwar schon länger mit Oracle 7 zu tun haben, aber wegen anderen Schwerpunkten nicht aus den Anfangsgründen herausgekommen sind. Empfehlenswert auch für Manager, die Tools aus anderen Welten mit dem Knöpfchen zur automatischen Datenbankgenerierung verkauft bekommen haben und dann ihren Datenbankleuten wegen den trotzdem verbratenen Stunden böse sind.

Solch ein Leser wird vermutlich das Buch als Bereicherung, die wesentlichen Punkte von Oracle 7 abgedeckt und die Mäkelei am „Datenmodelle“-Kapitel als ungerecht empfinden. Mit der Objektorientierung läßt sich dessen Antiquiertheit jedoch durchaus belegen: Erstens gibt es seit Jahren OO-Projekte, die Oracle als Datenbanksystem verwenden und wo regelmäßig Fragen der anfangs beschriebenen Art auftauchen. Zweitens gehen aktuelle Datenbankehrbücher auf den objektorientierten Entwurf und die Abbildung auf das relationale Modell ausführlich ein (Beispiel: „Datenbankeinsatz“ von Stefan Lang und Peter Lockemann, 1995). Drittens steht zwar Oracle 7 auf dem Buchtitel, und erst das gerade erscheinende Oracle 8 soll die Möglichkeit enthalten, eigene Objekttypen zu realisieren, und die Pressemeldungen von einer Java-Einbindung und einer Oracle-Unterstützung der Rational-UML entstammen der jüngsten Vergangenheit, aber es gab schon zu Zeiten der Version 6 Oracle-Papiere zu einer „objektorientierten Perspektive“ und erste Schritte auf dem dort vorgezeichneten Weg waren in Oracle 7 zu finden, beispielsweise die Möglichkeit, in PL/SQL geschriebene Funktionen in die Select-Klausel aufzunehmen. Dieser Mangel hinsichtlich der Objektorientierung ist im Buch substantiell, denn der Autor setzt zu oft Entities mit Objekten gleich, ohne auf den wesentlichen Unterschied zwischen Entity-Relationship-Modellen und objektorientiertem Entwurf einzugehen. Da wird die Welt zu sehr mit der Brille von Oracle\*CASE gesehen.

Grundsätzlich also sowohl viel Licht als auch ein großer Schatten. Wünschenswert wären mehr Bücher, die sich dem gesamten Weg von der Datenbankkonzeption bis zu deren technischer Umsetzung mit einem konkreten Datenbanksystem stellen. Und die sollten nicht mehr Seiten benötigen als Frank Roeing und könnten auch mal auf die neue Einführung von E-R-Modell, Normalformen und Triggern verzichten und stattdessen ausführlicher praktisches Erfahrungswissen vermitteln.

Jürgen Kunz, Schöllbronn

# Ankündigungen

(Für Hinweise auf EMISA-Veranstaltungen und Veranstaltungen mit EMISA-Beteiligung siehe auch Seite 3 - 10!)

- **EMISA-Fachgruppentreffen 1997**  
**"Workflow-Management-Systeme im Spannungsfeld einer Organisation",**  
**8. - 10. Oktober 1997, TH Darmstadt**
- Seventh International Workshop on "Foundations of Models and Languages for Data and Objects",  
15. - 19. September 1997 in Minsin
- "Informatik '97", 27. Jahrestagung der Gesellschaft für Informatik  
24. - 26. September 1997 in Aachen
- 4. Workshop "Algorithmen und Werkzeuge für Petrinetze"  
2. - 3. Oktober 1997 in Berlin
- Treffen der GI-Arbeitsgruppe  
"Petrinetze und Informationssysteme in der Praxis"  
10. Oktober 1997 in Darmstadt
- GI-FA5.1 Fachtagung "Software-Management '97"  
29. - 31. Oktober 1997 in München
- 3. Workshop des GI-AK 5.2.1.2 "Zeitorientierte betriebliche Informationssysteme"  
(ZobIS3)  
14. November 1997 in Zürich
- GI-Workshop  
"Modellierung '98"  
12./13. März 1998 in Münster
- 22. Jahrestagung der Gesellschaft für Klassifikation  
4. - 6. März 1998 in Dresden
- **EMISA-Workshop "Lebenszyklus großer Informationssysteme"**  
2. - 4. Juni 1998 in Schloß Dagstuhl,  
Organisation: H. Jasper, S. Jablonski, Call for Papers: siehe  
<http://www.sunsite.informatik.rwth-aachen.de/Societies/GI-EMISA/>
- Euromicro Workshop on "Software Process and Product Improvement"  
25. - 27. August 1998 in Västerås/Schweden

# Informatik '97



27. Jahrestagung der Gesellschaft für Informatik  
Aachen, 24.-26. September 1997

## Informatik als Innovationsmotor

### Hauptvorträge

*Prof. Dr.-Ing. Dr. h.c. mult. H.-J. Warnecke, Präsident der Fraunhofer-Gesellschaft:*  
Informatik -- Motor im Wandel der Strukturen

*Dr. R. Hüber, Direktor "Kommunikationstechnologien" der EU-Kommission i.R.:*  
Elektronischer Handel -- Neuer Fokus der Informatik

*Dr. G. Papadopoulos, Chief Technology Officer, Sun Microsystems:*  
Future Directions in Network Computing

*Prof. Dr. Th. Reps, University of Wisconsin / DARPA:*  
The Use of Program Profiling for Software Maintenance,  
with Applications to the Year 2000 Problem

- **Erstmals : Spitzenforschung auf einen Blick**  
18 Sonderforschungsbereiche und Schwerpunktprogramme der DFG
- **Bildung – Forschung – Innovation**  
40 Fachvorträge und hochrangig besetzte Podiumsdiskussionen
- **Die Euregio Aachen**  
Vorträge, Firmen- und Institutsbesichtigungen -- und kulturelle Highlights!
- **3 Tutorials und 14 Workshops**  
Hintergrundinformation und neueste Ergebnisse zu aktuellen Themen
- **Doktorandentreffen und Studierendenprogramm**  
Kontakte untereinander und mit zukünftigen Arbeitgebern

### Anmeldung

Informatik '97  
Informatik V, RWTH Aachen  
Ahornstraße 55  
52056 Aachen

Tel.: +49 +241 80 21 501  
Fax.: +49 +241 88 88 148  
gi-97@informatik.rwth-aachen.de

### Tagungsleitung

Matthias Jarke  
RWTH Aachen  
  
Klaus Pasedach  
Philips GmbH

Klaus Pohl  
RWTH Aachen

### Unterstützung

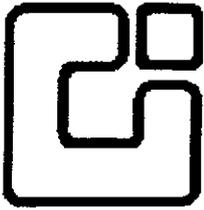
Aachener-Münchener Informatik,  
Computerzeitung, Compuware GmbH,  
Deutsche Telekom AG, Ericsson Eurolab,  
Freund und Dirks, IHK Aachen,  
Mitsubishi GmbH, ORACLE Deutschland GmbH,  
Parsytec Computer GmbH, Philips GmbH,  
REGINA e.V., Sparkasse Aachen,  
Stadt Aachen, Sun Microsystems GmbH

Jetzt Frühzahlerrabatt nutzen !

<http://www-i5.informatik.rwth-aachen.de/GI97/>

**RWTH** RHEINISCH  
WESTFÄLISCHE  
TECHNISCHE  
HOCHSCHULE  
AACHEN  
Fachgruppe Informatik  
- 25 Jahre -





## Einladung zum 4. Workshop

# Algorithmen und Werkzeuge für Petrinetze

2. - 3. Oktober 1997

## Humboldt-Universität zu Berlin

veranstaltet von der Fachgruppe 0.0.1 "Petrinetze und verwandte System- modelle" der Gesellschaft für Informatik.

Der Workshop konzentriert sich auf Fragestellungen zur Analyse und Simulation von Petrinetz-Modellen und auf Erfahrungen mit der Implementierung von Analyse-, Simulations- und Visualisierungswerkzeugen. Aktuelle Ergebnisse, Werkzeuge und auch Überblicke können dargestellt werden. Einen Schwerpunkt des Treffens sind der Erfahrungsaustausch und die Diskussion.

### Veranstaltungsort:

Humboldt-Universität zu Berlin  
Institut für Informatik  
Axel-Springer-Straße 54a  
10117 Berlin

und

Fraunhofer ISST  
Kurstraße 33  
10117 Berlin

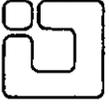
### Leitung:

Jörg Desel, Universität Karlsruhe  
Ekkart Kindler, Humboldt-Universität zu Berlin  
Andreas Oberweis, J.W. Goethe-Universität Frankfurt

### Themen (mit Bezug zu Petrinetzen):

- Entscheidungsalgorithmen für dynamische Eigenschaften
- Zustandsraumreduktion
- Temporale Logik und Model-Checking
- Analyse mit algebraischen und linear algebraischen Methoden
- Simulationsverfahren





Gesellschaft für Informatik e.V.  
Fachausschuß 5.1: Management der Anwendungsentwicklung  
und -wartung

## Fachtagung

# Software- Management '97

## München

29. - 31. Oktober 1997

# Programm und Einladung

## TUTORIEN

Mittwoch, 29. Oktober 1997 von 9.00 Uhr bis 12.30 Uhr

### TUTORIAL I: Das V-Modell

Helmut Hummel und Rainer Middendorf, IABG Ottobrunn

Das V-Modell, seit 1992 im Bereich des Verteidigungsministeriums vorgeschrieben und nach einer Erprobungsphase auch für den Einsatz im zivilen Verwaltungsbereich der Bundesbehörden empfohlen, wird derzeit von öffentlichen Auftraggebern und privatwirtschaftlichen Auftragnehmern gleichermaßen genutzt. Auf der Basis von Einsatzverfahren und technischer Evolution wurde das V-Modell überarbeitet und liegt nun in einer fortgeschrittenen Version vor.

Das Tutorial hat zum Ziel, die fortgeschriebene Version des Vorgehensmodells des Bundes, das V-Modell '97, vorzustellen. Dazu wird das V-Modell '97 in seiner neuen Struktur und in seinen Inhalten diskutiert. Speziell wird dabei auf die Änderungen in den Submodellen "Systemerstellung", "Projektmanagement", "Qualitätssicherung" und "Konfigurationsmanagement" eingegangen. Weiter wird auch eine Neuerung, die sogenannte Handbuchsammlung, vorgestellt.

Das Tutorial wendet sich an Projektmanager, die eigenverantwortlich IT-Vorhaben planen und durchführen müssen, und an alle, die sich über die Änderungen am V-Modell informieren möchten.

### TUTORIAL II: How to manage a web farm?

Andreas Schoberth, FAST e.V. München

Dienstleister im Bereich Web-Hosting betreuen typischerweise Größenordnungen von einigen Dutzend Kunden aus Mittelstand und Großunternehmen. Die Einrichtung und der Betrieb einer entsprechenden Zahl von WWW-Servern (oft in Verbindung mit Mail, News, FTP, etc.) stellt spezielle Anforderungen an die technische und organisatorische Projektentwicklung und den Serverbetrieb. Einerseits ergeben sich Synergien aufgrund der Ähnlichkeit der einzelnen Installationen, andererseits muß individuellen Kundenwünschen Rechnung getragen werden.

Der Vortragende berichtet aus den Erfahrungen mit dem Betrieb einer solchen "web farm" und stellt im Einzelfall auch die Detaillösungen vor.

Wichtige Themen:

- Multithreading-Server
- Entwicklungsumgebung und -prozeduren
- Sicherheitsaspekte
- automatisierte Erstellung von Zugriffsstatistiken
- Kalkulationsmodelle, bandbreitenbasierter/volumenbasierter
- Pflege und Dauerbetrieb ("Web Operation Center")
- Abrechnungsorganisation

### TUTORIAL III: Objektorientiertes Systems Reengineering

Harry Sneed, SES Ottobrunn

Dieses Tutorial setzt sich mit der Integration zwischen den alten monolithischen hostbasierten prozeß-orientierten Anwendungssystemen und den neuen verteilten Client/Server objektorientierten Systemen auseinander. Im Mittelpunkt steht die Strategie der objektorientierten Unternehmensmodellierung verbunden mit Reengineering und Wrapping Techniken. Es werden die drei Migrationsalternativen

- Ablösung
  - Konversion und
  - Kapselung
- vorgestellt und ihre Kosten/Nutzen aufgezeigt. Außerdem berichtet der Referent über seine Projekt-Erfahrungen in den Bereichen
- Software-Sanierung,
  - Datenkonversion,
  - Software-Downsizing und
  - Software-Wrapping.

Besonders betont wird die Rolle der CORBA-Norm und der IDL-Schnittstellensprache bei der Verbindung alter und neuer Systeme.

**Mittwoch, den 29. Oktober 1997**

10.30 Uhr

- Sitzung 2: Experience Factory - Strategie und Bewertung
- Dieter Landes und Kurt Schneider, Daimler Benz Urm
  - Systematische Aufbereitung und Nutzung von Erfahrungen aus Softwareprojekten bei Daimler Benz
  - Alex Lukassen, COLT Telecom Frankfurt/M. Planung und Realisierung einer DV-Strategie Eberhard Stickel, Europa-Universität Viadrina Frankfurt/O.
  - Der Einsatz der Optionspreistheorie zur Bewertung von Software-Entwicklungsprojekten

19.00 Uhr Abendveranstaltung

**Freitag, den 31. Oktober 1997**

9.00 Uhr Tutorien

10.30 Uhr Pause

11.00 Uhr Fortsetzung der Tutorien bis

12.30 Uhr

**Tagungsbeginn**

Begrüßung

Gastvortrag I

Gerhard Knolmayer, Universität Bern  
Determining Work Units in Year 2000  
Maintenance Projects

15.30 Uhr Kaffeepause

16.00 Uhr Sitzung 1: Projektmanagement

- Robert Hürten, Hürten & Partner Unternehmensberatung Steuerung der Produktivität in der Software-Entwicklung auf der Basis eines erweiterten Berichtswesens
- Rainer Stahl und Detlev Frick, Deutsche Telekom, Essen Projektorganisation in komplexen IV-Entwicklungsprojekten
- Axel Fell, GEZ Köln  
Prioritätensteuerung von Projekten bei der GEZ

17.30 Uhr

Gastvortrag II

Ernst Denerf, sd&amp;M München

Design-in-the-large und in-the-small -  
Software-Architektur und Objektorientierung

18.30 Uhr

Bayerischer Abend

Dinner Speech

**Donnerstag, den 30. Oktober 1997**

9.00 Uhr

Gastvortrag III  
Peter Elzer, Technische Universität Clausthal  
Software-Management im Wandel

10.00 Uhr

Kaffeepause

17.00 Uhr

- Workshop  
W. D. Nagl, Oracle München  
Internet-Ausstrahlungen
- 17.00 Uhr:  
Mitgliedervers.  
GI-FG 5.1.2
- 17.30 Uhr:  
Mitgliedervers.  
GI-FG 5.1.3
- 18.00 Uhr:  
Mitgliedervers.  
GI-FG 5.1.1

8.30 Uhr

Gastvortrag IV  
R. Studer, Universität Karlsruhe  
Vorgehensmodelle und Methoden des  
Information Systems Engineering, Software  
Engineering und Knowledge Engineering -  
Abschlussbericht aus einer GI-Arbeitsgruppe

9.30 Uhr

- Sitzung 5: Erfahrungsberichte
- Peter Buxmann, Frank Rose und Wolfgang König, J.W. Goethe-Universität Frankfurt/M.  
Wiederverwendung von Software-Elementen: Das Java-Repository
  - Helmut Häck, Bergisch Gladbach  
Ein Erfahrungsbericht aus der Praxis des betrieblichen Software-Managements
  - Norbert Schulte, Organisations- und IV-Beratung Tönisvorst  
Einführung vom Projektmanagement oder Die Operation am offenen Herzen - Ein Erfahrungsbericht oder Eine Krankengeschichte

11.00 Uhr

Kaffeepause

11.30 Uhr

Sitzung 6: Workflow-Management und  
Prozeßautomation

- Barbara Dellen, Frank Maurer und Gerhard Pews, Universität Kaiserslautern  
Wissensbasierte Techniken zur Flexibilisierung von Workflowsystemen
- Paul Grünbacher und Susanne Hofer, Johannes Kepler Universität Linz  
Prozeßautomation in kleinen Software-Unternehmen
- Roland Holten, Rüdiger Striemer und Mathias Weske, Westf. Wilhelms-Universität  
Münster, Fraunhofer Institut für Software-  
und Systemtechnik Dortmund  
Vorgehensmodelle zur Entwicklung von  
Workflow-Anwendungen - Eine  
vergleichende Darstellung

13.00 Uhr

Verabschiedung

**Anmeldung zur GI-Fachtagung  
SOFTWARE-MANAGEMENT '97**

29.-31. Oktober

**Programmkomitee:**

Prof. Dr. G. Chroust (Uni Linz)  
K.B. Elbrechter (Commerzbank Frankfurt)  
H.-J. Etzel (Eschborn)  
Prof. Dr. H. Heilmann (Uni Stuttgart)  
Prof. Dr. W. Hesse (Uni Marburg)  
Prof. Dr. K. Hildebrand (FH Ludwigshafen)  
Dr. H. Hummel (IABG Ottobrunn)  
Dr. R. Kneuper (Darmstadt)  
Prof. Dr. G. Knolmayer (Uni Bern)  
Prof. Dr. W. König (Uni Frankfurt/M.)  
A. Lukassen (DVAG Frankfurt/M.)  
K. Manny (CDA Datentechnik Backnang)  
PD Dr. A. Meier (CSS Versicherung Luzern)  
Prof. Dr. H.C. Mayr (Uni Klagenfurt)  
Prof. Dr. W. Mellis (Uni Köln)  
G. Müller-Luschnat (FAST e.V. München)  
Dr. W.D. Nagl (Oracle München)  
Prof. Dr. A. Oberweis (Leitung, Uni Frankfurt/M.)  
Prof. Dr. E. Ortner (TH Darmstadt)  
Dr. K. Pohl (RWTH Aachen)  
Dr. R. Richter (Uni Karlsruhe)  
Prof. Dr. G. Saake (Uni Magdeburg)  
H. Sneed (SAS GmbH Ottobrunn)  
Prof. Dr. T. Spitta (Uni Bielefeld)  
Prof. Dr. W. Stucky (Uni Karlsruhe)  
Prof. Dr. R. Studer (Uni Karlsruhe)  
H. Thoma (Ciba, Basel)

**Organisationskomitee:**

H. Sneed (SES GmbH Ottobrunn)  
B. Wix (Oracle Institut München)

An

**SES Logo**  
Software-Engineering Service GmbH  
Rosenheimer Landstraße 37

D-85521 Ottobrunn

oder per Fax: ++49 (0)89 608 53 671

**Hiermit melde ich mich zur GI-Fachtagung Software-Management '97 an.**

Vor-, Name: \_\_\_\_\_ Abt.: \_\_\_\_\_  
Firma: \_\_\_\_\_

Straße: \_\_\_\_\_  
PLZ: \_\_\_\_\_ Ort: \_\_\_\_\_

Tel.: \_\_\_\_\_ Fax: \_\_\_\_\_

**Tagungskosten**

- Konferenzteilnahme
- Normalgebühr\* DM 960,-
- GI-Mitglieder\* (GI-Nr. angeben) DM 720,-
- Hochschulangehörige\* DM 440,-
- Studenten\* (Studentenausweis) DM 240,-

\* Diese Preise sind nach §4 Nr.22a UStG von der MwSt befreit und beinhalten den Tagungsband und Kaffeepausen.

Mittagessen und Übernachtungen sind nicht im Preis enthalten.

- Zusätzliche Teilnahme an:
- Tutorien # DM 400,-
  - Abendveranstaltung # DM 60,-  
Bayerischer Abend am 29.10.1997

**Hotelreservierung**

Hotelreservierungen an:  
**Holiday Inn Crowne Plaza**  
Leopoldstr. 194  
80804 München  
Telefon: 089-38179-0  
Fax: 089-38179-888

Die Tagung findet in diesem Hotel statt. Begrenzte Zimmerkontingente stehen mit Hinweis auf die Firma **ORACLE** zum Sonderpreis zur Verfügung. Reservieren Sie bitte frühzeitig. Sollte das von uns reservierte Kontingent erschöpft sein, wenden Sie sich bitte an uns, wir helfen Ihnen weiter.

# Diese Preise enthalten gesetzl. MwSt.

**Zahlungsweise**

Nach Erhalt der Rechnung ist die Tagungsgebühr durch Überweisung auf das **Konto 4442547-01** der Deutsche Bank München (**BLZ: 700 700 10**) zu begleichen. Als Verwendungszweck muß unbedingt Ihr Vor-/Zuname und GI97 angegeben werden.  
Nach Erhalt der Rechnung sind die Tutoriengebühr und die Kosten für die Abendveranstaltung auf das **Konto 4442547** der Deutsche Bank München (**BLZ 700 700 10**) zu überweisen.  
Andere Zahlungsmöglichkeiten stehen nicht zur Verfügung.

Datum: \_\_\_\_\_ Unterschrift: \_\_\_\_\_

### **3. Workshop Zeitorientierte betriebliche Informationssysteme (ZobIS3) des Arbeitskreises 5.2.1.2 der Gesellschaft für Informatik**

**Zeit: 1997-11-14**

**Ort: Schweizerische Bankgesellschaft, Konferenzgebäude Grüenhof, Zürich**

Der 1995 gegründete Arbeitskreis 5.2.1.2 der Gesellschaft für Informatik hat 1995 bei der Westdeutschen Landesbank und 1996 an der Universität Cottbus erfolgreiche Workshops abgehalten. Aus dem letztgenannten Workshop entstand das Anfang 1997 erschienene Schwerpunktheft der Zeitschrift Wirtschaftsinformatik mit dem Titel "Das Jahr 2000-Problem und andere Aspekte zeitorientierter Informationssysteme".

Der nächste Workshop des Arbeitskreises wird am 14. November 1997 in Zürich von der Schweizerischen Bankgesellschaft (UBS) ausgerichtet werden. Zu diesem Workshop sind Beiträge insbesondere zu folgenden Themenkreisen erwünscht:

- Jahr 2000-Projekte
- Time-Handler als Kern einer Informationsarchitektur
- Modellierung temporaler Datenbanken
- Zeitbezogene Informationen in Repositories
- Zeitbezogene Daten am World-Wide-Web
- Management von Zeitreihen-Daten
- Spatio-temporale Informationssysteme
- Berücksichtigung zeitlicher Aspekte in Data-Warehouse-Systemen

Angenommene Beiträge werden Anfang 1998 im Rundbrief der Fachgruppe 5.2 der Gesellschaft für Informatik "Informationssystem-Architekturen" veröffentlicht. Der Text dieser Veröffentlichungen darf maximal 8 Seiten umfassen. Die Texte der Beiträge müssen bis spätestens 1997-08-20 vorliegen.

Die Teilnahme am Workshop ist kostenlos.

Die Teilnehmerzahl an dem Workshop ist auf rund 40 Personen beschränkt.

Sofern Sie an dem Workshop teilnehmen wollen, teilen Sie dies bitte bis zum 1997-06-30 mit nachstehendem Anmeldeformular mit.

---

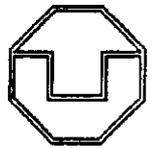
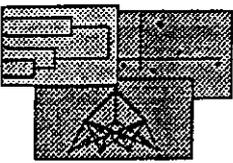
An das Institut für Wirtschaftsinformatik der Universität Bern  
Abt. Information Engineering  
Frau Elisabeth Berger  
Engehaldenstrasse 8  
CH 3012 Bern  
Schweiz  
Fax: 0041.31.631.4682  
E-Mail: berger@ie.iwi.unibe.ch

### 3. ZobIS-Workshop in Zürich

Sehr geehrte Frau Berger,

- Ich habe Interesse an einer Teilnahme am obengenannten Workshop.  
Bitte senden Sie mir eine Einladung zur Teilnahme.
  
- Ich würde am 3. ZobIS-Workshop in Zürich gerne einen Vortrag mit dem Arbeitstitel: ..... halten. Den endgültigen Beitrag werde ich bis 1997-08-20 übersenden.

Name: ..... Vorname: .....  
Firma/Uni: ..... Funktion: .....  
Adresse: ..... PLZ/Ort: .....  
Tel.: ..... Fax: .....  
Email: .....



The Gesellschaft für Klassifikation e.V. (GfKI), a member of the International Federation of Classification Societies (IFCS), will hold its 22nd Annual Conference by invitation of the University of Dresden under the title

## Classification in the Information Age

The scientific program will include plenary talks, survey lectures, and contributed papers in special sections. Emphasis will be placed on the topic

Challenges for the Information Society Concerning Data Analysis and Knowledge Processing.

The following sections are planned (organizers in brackets):

- Mathematical and statistical methods for data analysis and classification (Bock)
- Representation of data and visualization (Mathar)
- Classification for speech and pattern recognition (Hoffmann, Ney)
- Multimedia, WWW and Internet (Schader)
- Navigation and classification on the Internet (Geyer-Schulz, Neumann)
- Data handling in virtual work places and business worlds (Bodendorf)
- Machine learning, reinforcement learning, PAC-learning (Ritter)
- Information systems and data bases (Esswein, Oberweis)
- Subject indexing and knowledge organization in data networks (Hermes)
- Conceptual analysis of data and knowledge (Ganter)
- Finance, capital markets, and risk management (Huschens, Locarek-Junge)
- Quality management, buying behaviour and marketing (Baier, Balderjahn)
- Data analysis in retailing (Decker, Hruschka)
- Psychology and social sciences (Feger, Rost)
- Data analysis in health economics (Opitz, Schipp)
- Case classifications in medicine (Klar, Lochmann)
- Genome analysis (Ludwig)
- Bioinformatics, epidemiology, and public health (Gefeller, Lausen)
- Archaeology (Vach, Weber, Zimmermann)
- Environment informations (Doberkat, Kuhlen)

Within the framework of the conference workshops, tutorials, and software demonstrations are scheduled, e.g.:

- a vocational course for librarians (Lohmann)
- a tutorial on "*Stammbaumrekonstruktion mit ARB*" (Ludwig)

Conference languages are German and English. A Proceedings volume will be published in English by Springer-Verlag. Interested persons will obtain further information if they send a preliminary registration. Authors who intend to present a talk are requested to submit an abstract by

**November 15, 1997**

to the local organizer Prof. Dr. Locarek-Junge. After acceptance of the talk (information is provided in December 1997) an English manuscript can be submitted for the refereeing process of the conference Proceedings. Formatting instructions for abstracts and manuscripts for the Proceedings volume can be obtained via WWW-browsers from <http://www.gfki.de> or by sending an e-mail to [gfki98@tu-dresden.de](mailto:gfki98@tu-dresden.de) with 'subject' *format*.

---

Program committee: W. Gauß; I. Balderjahn, H.H. Bock, F. Bodendorf, R. Decker, H.J. Hermes, B. Lausen, H. Locarek-Junge, R. Mathar, O. Opitz, G. Ritter

Local organizer: H. Locarek-Junge

Correspondence to: Prof. Dr. H. Locarek-Junge, Technische Universität Dresden, Fakultät Wirtschaftswissenschaften, Lehrstuhl für Betriebswirtschaftslehre, insbes. Finanzwirtschaft, Mommsenstr. 13, D-01062 Dresden, Phone: +49-351/463-5993, Fax: +49-351/463-5404, e-mail: [gfki98@tu-dresden.de](mailto:gfki98@tu-dresden.de), Bank account: gfki 98 TU Dresden, no. 5360230858 at Hypo-Bank Dresden, routing code (BLZ) 85020890

---

Please copy and return this page by November 15, 1997 to:

Prof. Dr. H. Locarek-Junge  
Technische Universität Dresden  
Fakultät Wirtschaftswissenschaften  
Lehrstuhl für Betriebswirtschaftslehre, insbes. Finanzwirtschaft  
MommSENstr. 13  
D-01062 Dresden

- I intend to participate in the 1998 annual conference of the GfKI. Please send me the registration form.
- Please, send further information concerning the 1998 annual conference of the GfKI.
- by mail  by e-mail

- I intend to present a talk, entitled:
- \_\_\_\_\_
- \_\_\_\_\_

Abstract<sup>\*)</sup> is enclosed.

Abstract<sup>\*)</sup> will follow by November 15, 1997.

\*) Formatting instructions for abstracts and manuscripts for the Proceedings volume can be obtained via WWW-browsers from <http://www.gfki.de> or by sending an e-mail to [gfki98@tu-dresden.de](mailto:gfki98@tu-dresden.de) with 'subject' *format*.

- I am interested in attending the following tutorials:
- \_\_\_\_\_
- \_\_\_\_\_

- I would like to suggest a tutorial on
- \_\_\_\_\_

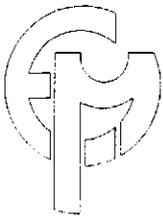
organized by (name, address):

\_\_\_\_\_

- Further remarks:
- \_\_\_\_\_

Sender:

|                          |  |
|--------------------------|--|
| Name, first name, title: |  |
| Address:                 |  |
| Phone/Fax:               |  |
| e-mail:                  |  |



# EUROMICRO 98

24th EUROMICRO Conference  
Engineering Systems and Software for the next decade

## EUROMICRO WORKSHOP ON SOFTWARE PROCESS and PRODUCT IMPROVEMENT

Västerås, Sweden, August, 25th-27th, 1998

### REFERENCE ADDRESSES

#### Programme Chairman

Gerhard Chroust  
System Engineering and Automation  
Kepler University Linz  
4040 Linz, Austria  
Tel: +43-732-2468-865  
Fax: +43-732-2468-878  
E-mail: GC@SEA.UNI-LINZ.AC.AT

#### Programme Committee

Veikko Seppanen (Vice-Chairman), SF  
Matthew Cooke, E  
Mark Dowson, USA  
Gregor Engels, D  
Alfonso Fuggetta, I  
Volker Gruhn, D  
Paul Grünbacher, A  
Volkmar H. Haase, A  
Gerold Johnson, USA  
Konrad Klöckner, D  
Günther Koch, D  
Nazim Madhavji, CDN  
Rainer Miederhoff, D  
Marco Maiocchi, I  
Risto Nevalainen, SF  
Andreas Oberweis, D  
Markku Oivo, SF  
John M. Roche, UK  
Joc Sanders, IRL  
Samuli Saukkonen, SF  
Erwin Schoitsch, A  
Jouni Simila, SF  
Jean-Martin Simon, F  
Larry Strader, USA  
Wolfgang B. Strigel, CDN  
J.M. Troya, E  
Hakan Wickberg, S

#### Organizing Chairman

Lennart Lindh  
IDT/CUS  
Dept. of Real-Time Computer Systems  
Mälardalen University,  
P.O.Box 883, S-721 23  
Västerås, Sweden  
Tel: +46 (0)21-101545  
Fax: +46 (0)21-103110  
E-mail: lennart.lindh@mdh.se

#### Euromicro Manager

Chiquita Snippe-Marlisa  
P.O.Box 2346  
7301 EA Apeldoorn  
The Netherlands  
Tel: +31 - 55 - 5795 503  
Fax: +31 - 55 - 5795 509  
E-mail: euromicro@standby.nl

### CALL FOR PAPERS

The success of software companies hinges on the quality of their software products, which in turn is a function of the quality of their software development process. Several approaches exist:

- improving, measuring and certifying the quality of the software development process (e.g. via ISO 9000, CMM, BOOTSTRAP or ISO 15504)
- improving the software design methodologies themselves by including new paradigms like object oriented programming, component-based software development
- emphasizing re-use via object-orientation, reengineering and software-factory concepts
- modelling and enacting the software development process by process models and their interpreters, including work flow systems
- improving methods for verification and validation via testing, reviews and formal proofs
- providing adequate development environments, repositories and tool integration
- decreasing the impacts of faults by fool-proving and fail-soft systems.

Submission to above topics are especially welcome which demonstrate the potential of new approaches in actual applications, describe experiments, experience and case studies, of process improvement experiments ( e.g. ESSI- PIEs, ESPITI- and SPIRE- projects) and take the view-point of small and very small software development organizations.

### SUBMISSION OF PAPERS

Both regular papers and short contributions are invited. Please refer to the General Call for Papers for EUROMICRO 98. Prospective authors are encouraged to e-mail PostScript version of their full paper to the programme chairman. If electronic submission is not possible, four copies of the full paper should be sent by post.

The regular papers should not exceed 4000 words and include an abstract of up to 150 words. The title page should show clearly the name, mailing address, the e-mail address and fax number of the author to contact as well as the topic areas for the submitted paper.

The same applies to short contributions, but here the number of words are limited to 1200 with an abstract of up to 60 words.

The Workshop Proceedings will be published by IEEE Computer Society and will be available for the Conference. Papers exceeding 8 pages (3 pages for short contributions) will be charged NLG 100,- per page in excess.

The following signed statement should be included on the title page: *"Neither this paper nor any version close to it has been or is being offered elsewhere for publication. All necessary clearances have been obtained for the publication of this paper. If accepted, the paper will be made available in Camera-ready forms by June 15th, 1998, and it will be personally presented at the EUROMICRO 98 Conference by the author or one of the co-authors. The presenting author(s) will pre-register for EUROMICRO 98 before the due date of the Camera-ready paper."*

### IMPORTANT DATES

|                             |                     |
|-----------------------------|---------------------|
| Deadline for submission:    | February 28th, 1998 |
| Notification of Acceptance: | April 30th, 1998    |
| Deadline for final version: | June 15th, 1998     |

### SPECIAL SESSIONS

Proposals on special sessions or panels are welcome. Please send suggestions to the program chairman before October 15th, 1997. Submission and reviewing of special session papers will follow the same procedures of the regular papers.

### GENERAL INFORMATION

The workshop is an integral part of the EUROMICRO 98 conference. You will find additional information on EUROMICRO 98 in: **General Call for papers, EUROMICRO 98.**

The Call for papers available for workshops under the umbrella of EUROMICRO 98 are:

- **Digital Systems Design: Architectures, Methods and Tools**
- **Dependable Computing Systems**
- **Multimedia and Telecommunications**
- **Computational Intelligence**
- **Software Process and Product Improvement**
- **Network Computing**

### UPDATED INFORMATION

The WWW page for this workshop is: <http://www.sea.uni-linz.ac.at/EU98SP1/index.htm>  
The EUROMICRO WWW site in Vienna: <http://europub.ict.tuwien.ac.at/index.htm>

## Fachexperten im EMISA Leitungsgremium:

|   |   |
|---|---|
| <p>Wirtschaftsinformatik/FG 5.2.1:</p> <p>Dr. Heiko Raue<br/>RWE-Telliance<br/>Abteilung NOS<br/>Hollestr. 7b<br/>D 45127 Essen<br/>Tel. 0201-12-20649<br/>Fax 0201-12-20385<br/>E-Mail Heiko.Raue@RWE-Telliance.De</p>   | <p>Petri-Netze:</p> <p>Prof. Dr. Gert Scheschonk<br/>C.I.T. Communication and Information<br/>Technology GmbH<br/>Ackerstraße 71-76<br/>D 13355 Berlin<br/>Tel. 030-4636077<br/>Fax 030-4641610<br/>E-Mail sheshonk@tfh-berlin.de</p> |
| <p>Knowledge Engineering:</p> <p>Prof. Dr. Rudi Studer<br/>Universität Karlsruhe<br/>Institut für Angewandte Informatik und<br/>Formale Beschreibungsverfahren<br/>Kollegiengebäude am Ehrenhof<br/>D 76128 Karlsruhe<br/>Tel. 0721-608-3923<br/>Fax 0721-693717<br/>Email studer@aifb.uni-karlsruhe.de</p> |   |

## EMISA Leitungsgremium

|   |   |
|---|---|
| <p>Prof. Dr. Gottfried Vossen<br/>(Sprecher des Leitungsgremiums)<br/>Universität Münster<br/>Institut für Wirtschaftsinformatik<br/>Greverenerstr. 91<br/>D 48159 Münster<br/>Tel. 0251-9275-103<br/>Fax 0251-83-9754<br/>Email vossen@uni-muenster.de</p> | <p>Prof. Dr. Andreas Oberweis<br/>(Redaktion EMISA FORUM)<br/>J.W. Goethe-Universität Frankfurt<br/>Lehrstuhl Wirtschaftsinformatik II<br/>Mertonstr. 17<br/>D 60054 Frankfurt/Main<br/>Tel. 069-798-28722/28998<br/>Fax 069-798-25073<br/>Email oberweis@wiwi.uni-frankfurt.de</p> |
| <p>Prof. Dr. Stefan Jablonski<br/><br/>Universität Erlangen-Nürnberg<br/>Informatik VI<br/>Martensstr. 3<br/>D 91058 Erlangen<br/>Tel. 09131-857885<br/>Fax 09131-32090<br/>Email jablonski@informatik.uni-erlangen.de</p>                                  | <p>PD Dr. Heinrich Jasper<br/><br/>Hochschule Vechta<br/>Umweltwissenschaften<br/>Postfach 1553<br/>D 49364 Vechta<br/>Tel. 04441-15-286/344<br/>Fax 04441-15-445<br/>Email heinrich.jasper@uni-vechta.de</p>   |
| <p>Dr. Manfred Jeusfeld<br/><br/>RWTH Aachen,<br/>Informatik 5<br/>Ahornstr. 55<br/>D 52056 Aachen<br/>Tel. 0241-80-21510<br/>Fax 0241-8888-321<br/>Email jeusfeld@informatik.rwth-aachen.de</p>  | <p>Dr. Roland Kaschek<br/><br/>Universität Klagenfurt<br/>Institut für Informatik<br/>Universitätsstr. 67<br/>A 9022 Klagenfurt<br/>Tel. +43-463-2700-527<br/>Fax +43-463-2700-505<br/>Email kaschek@ifi.uni-klu.ac.at</p>  |
| <p>Prof. Dr. Udo Lipeck<br/><br/>Universität Hannover<br/>Institut für Informatik<br/>Lange Laube 22<br/>D 30159 Hannover<br/>Tel. 0511-762-4951<br/>Fax 0511-762-4961<br/>Email ul@informatik.uni-hannover.de</p>  | <p>Prof. Dr. Erich Ortner<br/><br/>Technische Hochschule Darmstadt<br/>Fachgebiet Wirtschaftsinformatik I<br/>Hochschulstr. 1<br/>D 64289 Darmstadt<br/>Tel. 06151-164309<br/>Fax 06151-164301<br/>Email ortner@bwl.th-darmstadt.de</p>   |
| <p>Dipl.-Inform. Helmut Thoma<br/><br/>Novartis Services AG<br/>Information Services<br/>Postfach<br/>CH-4002 Basel<br/>Tel. 0041-61-697-3419<br/>Fax 0041-61-697-2469<br/>E-Mail: helmut.sl-thoma@chbs.MHS.CIBA.COM</p>                                    |   |

**REDAKTIONSSCHLUSS FÜR  
DIE NÄCHSTE AUSGABE  
EMISA FORUM HEFT 1/1998:**

**5. DEZEMBER 1997**