Erschienen in: Sinz, E. (Hg.): Modellierung betrieblicher Informationssysteme. Proceedings der MobIS-Fachtagung 1999. Bamberg 1999, S. 154-169

Eine Architektur zur Spezifikation von Sprachen und Werkzeugen für die Unternehmensmodellierung

Prof. Dr. Ulrich Frank
Institut für Wirtschaftsinformatik, Universität Koblenz-Landau
Rheinau 1, 56075 Koblenz

Zusammenfassung

Bei der Planung, Entwicklung und Einführung betrieblicher Informationssysteme werden vielfältige Modelle verwendet. I.d.R. sind die für unterschiedliche Zwecke (strategische Planung, Organisationsgestaltung, Systementwurf ...) eingesetzten Modelle aber nur für einen kleinen Kreis von Betrachtern verständlich. Das führt zu Kommunikationsproblemen, Inkonsistenzen zwischen unterschiedlichen Modellen und vermeidbarer Doppelarbeit. Im vorliegenden Beitrag wird eine Architektur zur Spezifikation von Sprachen für die Unternehmensmodellierung vorgestellt, die diesem Problem entgegenwirkt. Sie beinhaltet eine erweiterbare Menge dedizierter Modellierungssprachen. Jede dieser Sprachen kann als eine Fachsprache angesehen werden, die für eine bestimmte Aufgabenstellung bewährte Begriffe und eine eingängige graphische Notation bietet. Die Definition der Sprachen erfolgt durch Metamodelle und zusätzliche formale Integritätsbedingungen. Um eine konsistente Bearbeitung verschiedener Teilmodelle zu unterstützen und zudem Code-Generierung und Simulationen zu ermöglichen, werden die Metamodelle mit Hilfe von Objektmodellen für den Entwurf einer integrierten Modellierungsumgebung rekonstruiert. Neben der Spracharchitektur werden in dem Beitrag Teile der Metamodelle sowie Beispiele für die Anwendung der Modellierungssprachen dargestellt.

1. Einleitung

Die Wirtschaftlichkeit der Entwicklung und Nutzung betrieblicher Informationssystemen ist nach wie vor i.d.R. unbefriedigend. Wesentliche Gründe dafür können in einem Mangel an Integration und Wiederverwendung vermutet werden. Dies gilt jeweils für unterschiedliche Aspekte. Wenn wir Integration zunächst als "Überwindung von Friktionen" bzw. "Eingliederung in ein größeres Ganzes" verstehen, dann ist neben der systemtechnischen Integration von Anwendungen und Softwarekomponenten (im Unternehmen selbst wie auch mit den Systemen von Kunden und Lieferanten) die Integration der Phasen des Software-Lebenszyklus wie auch die Integration von Informationssystemen in die jeweilige Organisation und Unternehmensstrategie zu berücksichtigen. Im Hinblick auf die wissenschaftliche Betrachtung betrieblicher Informationssysteme macht es zudem Sinn, eine engere Integration der betroffenen Disziplinen (wie Betriebswirtschaftslehre, Informatik, Wirtschaftsinformatik) zu fordern. In ähnlicher Weise bezieht sich das Bemühen um Wiederverwendung nicht nur auf Software-Artefakte, sondern auch auf Entwurfs- und Gestaltungswissen - sowohl im Hinblick auf Informationssysteme wie auch im Hinblick auf Unternehmensorganisation und -strategie. Dazu kommt die Wiederverwendung von Terminologien, Modellen und Theorien aus verschiedenen wissenschaftlichen Disziplinen.

Die Ursachen für den skizzierten Integrationsbedarf sind vor allem darin zu sehen, daß unterschiedliche Sprachen verwendet werden. Dies gilt sowohl für die Definition von Systembestandteilen als auch für

die verschiedenen Fachsprachen, die für die Beschreibung und die Analyse der verschiedenen Aufgabenbereiche in Unternehmen verwendet werden. Die Integration verschiedener sprachlich dargestellter Systeme erfolgt über gemeinsame Konzepte bzw. Begriffe. Wenn man einen an den Informationsgehalt angelehnten Semantikbegriff verwendet, kann man sagen, daß das Integrationsniveau mit der Semantik gemeinsamer Begriffe zunimmt. Dadurch wird eine selektive, verläßliche und damit effiziente Kommunikation möglich. Im Kontext betrieblicher Informationssysteme vollzieht sich auch Wiederverwendung wesentlich über sprachliche Artefakte. Die Wiederverwendbarkeit eines Artefakts hängt einerseits von seinem Einsatzbereich ab, andererseits von der Sprache, in der es beschrieben ist: Je verständlicher einem potentiellen Wiederverwender diese Sprache erscheint, desto besser die Chancen der Wiederverwendung. Die Semantik der Artefakte weist gleichsam ein dialektisches Verhältnis zur Wiederverwendbarkeit auf: Mit zunehmender Semantik sinkt tendenziell die Häufigkeit der Wiederverwendung, gleichzeitig steigt aber der Wiederverwendungsnutzen in den Fällen, denen die spezielle Semantik gerecht wird.

Im engeren Bereich der Software-Entwicklung zielen konzeptionelle Modelle (Datenmodelle, Objektmodelle, Datenflußdiagramme ...) darauf, Integration und Wiederverwendung zu fördern. Der Einsatz betrieblicher Informationssysteme empfiehlt allerdings häufig auch die Berücksichtigung anderer Abstraktionen (z.B. Unternehmensstrategie, Geschäftsprozesse ...). Um die Integration der damit verbundenen Sichten wie auch die Wiederverwendung von Artefakten über verschiedene Sichten hinweg zu unterstützen, wird seit geraumer Zeit der Begriff "Unternehmensmodell" diskutiert. Zur Erstellung von Unternehmensmodellen benötigen wir verschiedene Modellierungssprachen, die in geeigneter Weise integriert sind. Bevor wir darauf eingehen, soll im folgenden zunächst die Forschungsstrategie, auf die der im weiteren dargestellte Ansatz beruht, skizziert werden.

2. MEMO: Grundannahmen und Forschungsstrategie

MEMO (Multi Perspective Enterprise Modelling) ist eine Methode zur Unterstützung der Unternehmensmodellierung. Sie besteht aus graphischen Modellierungssprachen sowie einem Vorgehensmodell. Im folgenden konzentrieren wir uns auf die Modellierungssprachen. Der Entwurf dieser Sprachen basiert auf einer Reihe von Annahmen:

- Der Entwurf, die organisatorische Integration und Nutzung betrieblicher Informationssysteme erfordert die Berücksichtigung unterschiedlicher *Perspektiven* und mit ihnen einhergehender Fachsprachen.
- Angesichts der Komplexität des Gegenstands empfehlen sich Modelle auf verschiedenen Abstraktionsebenen. Um die Integration der mit verschiedenen Perspektiven korrespondierenden Modelle zu fördern, müssen die jeweils verwendeten Modellierungssprachen gemeinsame Konzepte aufweisen.

- Modelle dienen nicht zuletzt als Medium der zwischenmenschlichen Kommunikation. Es wird davon ausgegangen, daß graphische Darstellungen dazu häufig besonders gut geeignet sind.
- Unternehmensmodelle sind i.d.R. ausgesprochen umfangreich und weisen vielfältige Abhängigkeiten auf. Je präziser Modelle beschrieben sind, desto eher lassen sich gehaltvolle Aussagen über ihre Integrität machen. Gleichzeitig bieten sich Teile von Unternehmensmodellen für die Generierung von Code bzw. die Durchführung von Simulationen an, was den Einsatz von Werkzeugen voraussetzt. Beide Aspekte empfehlen eine formalsprachliche Beschreibung der Modelle.

Der Entwurf entsprechender Modellierungssprachen stellt insofern eine große Herausforderung dar, als ihre Bewertung vom Verwendungszweck und den Präferenzen der Sprachanwender abhängt. Beides ist ex ante kaum vollständig abzuschätzen. Gleichzeitig ist die Spezifikation einer Modellierungssprache mit einem großen Aufwand verbunden. Vor diesem Hintergrund wurde folgende Vorgehensweise gewählt: Zunächst wird eine Sprache semi-formal mittels Metamodellierung (siehe 3) beschrieben. Anschließend wird diese Sprache in der Modellierung eingesetzt, was i.d.R. zu einer Reihe von Anpassungen sowohl der abstrakten Syntax und Semantik wie auch der konkreten Syntax (Notation) führt. Erst wenn die Sprache auf diese Weise eine befriedigende Reife erlangt hat, wird sie formalisiert.

Die mit MEMO verbundene Forschungsstrategie ist allerdings nicht allein auf den Sprachentwurf gerichtet. Vielmehr soll die Anwendung der Sprachen durch Modellierungsprinzipien bzw. Entwurfsmuster unterstützt werden. Langfristig soll zudem eine Reihe von Referenzmodellen für ausgewählte Einsatzgebiete entstehen (vgl. 5).

3. MEMO: Spracharchitektur und Werkzeugumgebung

Unternehmensmodelle dienen der Abbildung verschiedener Perspektiven auf ein Unternehmen. Damit stellt sich die Frage, welche Perspektiven berücksichtigt werden sollten. In der Literatur finden sich dazu unterschiedliche Vorschläge - sowohl im Hinblick auf die Differenzierung der Perspektiven als auch hinsichtlich ihre Konzeptualisierung ([FeSi97], [Sch97], [ESP93], [SoZa92]). Eine ideale Differenzierung läßt sich objektiv kaum ermitteln. Tendenziell kann man fordern, daß die gewählten Perspektiven gängigen Sichtweisen und mit ihnen korrespondierenden Aufgabenstellungen gerecht werden sollten, um auf diese Weise den Wahrnehmungs- und Konzeptualisierungsmustern der verschiedenen Anwendergruppen entgegenzukommen. Vor diesem Hintergrund wird in MEMO eine Unterteilung in drei Perspektiven (Strategie, Organisation, Informationssystem) vorgenommen, die jeweils in verschiedene Aspekte differenziert werden (Ressourcen, Struktur, Leistungserstellung, Ziele, relevante Umwelt). Auf diese Weise entstehen 15 Foki, die sich auf bestimmte Themen konzentrieren (eine ausführliche Darstellung findet sich in [Fra97]). Die von MEMO bereitgestellten Sprachen sind darauf gerichtet, Modelle für verschiedene Foki (jeweils einen oder mehrere) dieses Bezugsrahmens zu beschreiben. So korrespondiert ein

Organisationsmodell mit allen Foki der Perspektive Organisation. Der Fokus Informationssystem/Struktur wird durch ein Objektmodell dargestellt. Foki der strategischen Perspektive lassen sich mit Hilfe von Modellen strategischer Geschäftseinheiten und von Wertketten [Por85] abbilden. Abb. 1 zeigt einen Überblick über verschiedene Teilmodelle und den zwischen ihnen bestehenden Beziehungen.

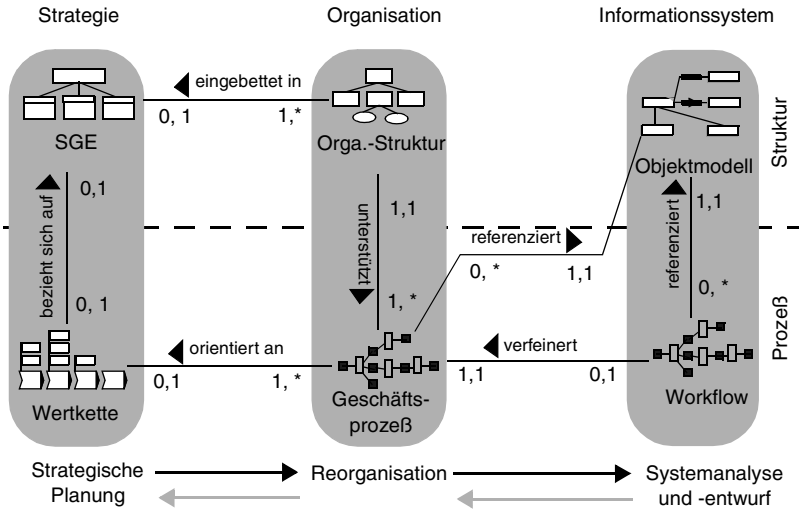
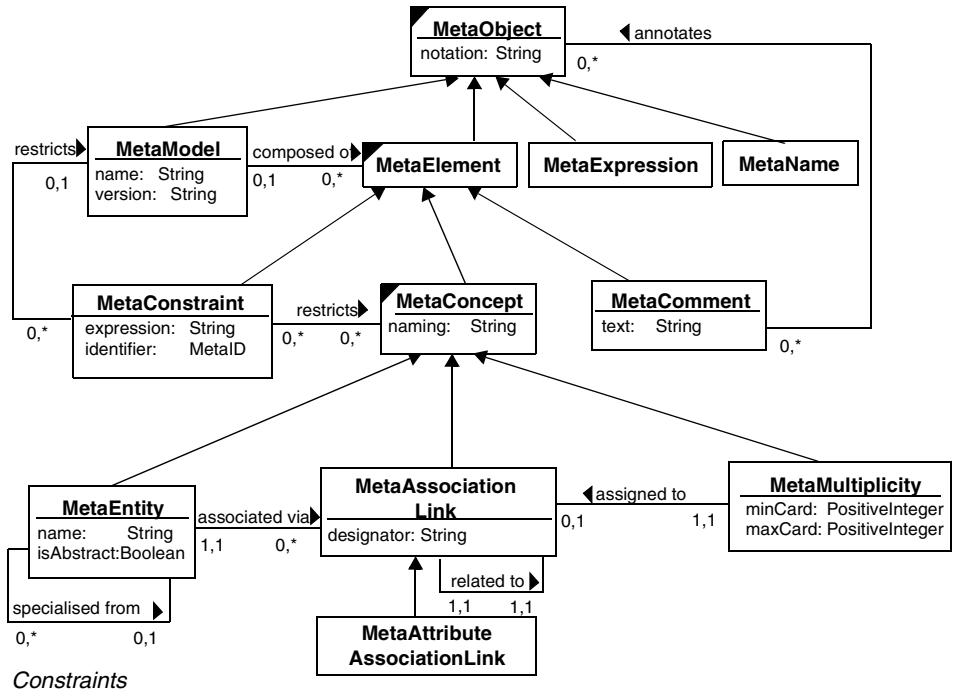


Abb 1: Beziehungen zwischen ausgewählten Teilmodellen in MEMO

Die Erstellung eines Modells erfordert eine Modellierungssprache. Gemäß dem Anspruch der Unternehmensmodellierung sollten die verwendeten Modellierungssprachen möglichst eng integriert sein. Dieser Anspruch ließe sich besonders gut umsetzen, wenn man für alle Teilmodelle eine generelle Modellierungssprache verwenden würde. Das Entity Relationship Modell ist wegen der unzureichenden Semantik seiner Konzepte dazu kaum hinreichend. Diese Einschränkung gilt für neuere objektorientierte Modellierungssprachen wie UML [Rat97] oder OML [FiHe96] nicht. Dennoch sind solche Sprachen nicht für alle Foki geeignet. Es handelt sich gleichsam um Fachsprachen des Software-Engineering. Begriffe anderer Fachsprachen, die für unseren Anwendungszweck wichtig sind (Organisation, Strategie, BWL), werden nicht berücksichtigt. Entsprechende Fachbegriffe unterstützen die Strukturierung eines Problems und damit die Erstellung einschlägiger Modelle. Eine dedizierte Modellierungssprache stellt sorgfältig spezifizierte Fachbegriffe bereit und befreit den Anwender damit von der Last, diese Begriffe selbst rekonstruieren zu müssen. Dabei ist auch daran zu denken, daß Syntax und Semantik genereller Modellierungssprachen nicht dem unzulässigen Gebrauch von Fachkonzepten entgegenwirken. Wenn beispielsweise das Konzept "Organisationseinheit" in UML als Klasse abgebildet wird, wäre es syntaktisch und semantisch korrekt, eine zyklische Beziehung "ist verantwortlich für" einzuführen. Der Sprachanwender könnte einem solchen Sprachgebrauch nur durch die explizite Einführung einer Integritätsbedingung entgegenwirken. Anders bei einer dedizierten Modellierungssprache. Hier könnte der Fachbegriff "Organisationseinheit" Bestandteil der Sprache sein, wodurch eine unzulässige Verwendung bereits durch die Sprachbeschreibung ausgeschlossen wäre. Ein weiteres Argument gegen gene-

relle Modellierungssprachen ergibt sich aus den verfügbaren graphischen Notationen. So bietet etwa UML keine leistungsfähigen Notationen für die Visualisierung von Geschäftsprozessen oder Organisationsstrukturen.

Vor dem Hintergrund dieser Überlegungen ist eine Architektur für die Spezifikation von Sprachen zur Unternehmensmodellierung entstanden. Sie sieht eine, ggfs. im Zeitverlauf wachsende Zahl von Modellierungssprachen vor. Die Integration der Sprachen erfolgt durch die Verwendung eines einheitlichen Meta-Metamodells (Abb. 2) und gemeinsamer Konzepte. Eine ausführliche Beschreibung des Meta-Metamodells findet sich zusammen mit einem Vergleich mit anderen Meta-Metamodellen in [Fra98a].



Constraints

Spezialisierungen dürfen nicht zyklisch erfolgen.

$\forall me{:}V_{MetaEntity} \bullet \neg(me(\rightarrow_{specialisedfrom})^{+} cpt)$

Eine minimale Kardinalität muß kleiner oder gleich der korrespondierenden maximalen Kardinalität sein.

$\forall mmp{:}V_{MetaMultiplicity} \bullet (mmp.\min Card <= mmp.\max Card)$

Eine Instanz von MetaAttributeAssociationLink darf keiner Instanz von MetaAttributeAssociationLink zugeordnet sein.

$\forall mal{:}V_{MetaAttributeAssociationLink} \bullet (mal \rightarrow_{relatesto} \rightarrow mal) = \varnothing$

Ein Attribut (eine Instanz von MetaEntity) ist mit genau einer Instanz von MetaEntity assoziiert (durch eine Assoziation zwischen MetaAssociationLink und MetaAttributeLink).

$\forall mmp{:}V_{MetaMultiplicity} \bullet (mmp \rightarrow_{assignedto \bullet MetaAssociationLink} \leftarrow_{relatedto \bullet MetaAttributeAssocLink}) \bullet mmp.\min Card = 1 \bullet mmp.\max Card = 1$

Assoziation

Generalisierung

Abb. 2: MEMO Meta-Metamodell. Die Integritätsbedingungen sind in GRAL spezifiziert.

Die Definition einer Sprache mittels einer Grammatik bietet zwar günstigere Voraussetzungen für die Überprüfung der Syntax eines Modells. Dennoch haben wir uns für die Verwendung graphischer Meta-

modelle entschieden, weil auf diese Weise ein Paradigmenwechsel von der Objekt- zur Metaebene vermieden wird - was (hoffentlich) die Verständlichkeit einer Sprachbeschreibung bei den Sprachanwendern fördert. Ein weiterer Grund für die Bevorzugung von Metamodellen ergibt sich durch den Einsatz einschlägiger Modellierungswerkzeuge. Sie werden typischerweise objektorientiert entworfen. Metamodelle lassen sich durch Objektmodelle unmittelbarer abbilden als Grammatiken. Dabei kann aber nicht übersehen werden, daß graphische Metamodelle für die Sprachbeschreibung allein nicht hinreichend sind. Deshalb werden sie zunächst durch natürlichsprachliche Integritätsbedingungen ergänzt. Die Formalisierung einer Sprache geschieht auf der Grundlage des formalisierten Meta-Metamodells und der formalen Beschreibung der Integritätsbedingungen. Für beide Zwecke wird die Sprache GRAL (Graph Specification Language) verwendet. GRAL erlaubt die Spezifikation von Integritätsbedingungen auf sog. TGraphen. Ein TGraph ist ein gerichteter Graph, der aus Knoten und Kanten besteht. Sowohl Knoten als auch Kanten sind typisiert und können durch Attribute beschrieben werden. GRAL beinhaltet eine Menge vordefinierter Prädikate, die zur Beschreibung wichtiger Eigenschaften von Graphen verwendet werden können (z.B. "isAcyclic (G)", "isNeighbourOf (G,v,W)"). Weitere Prädikate können in der Spezifikationssprache Z definiert werden. Um Prädikate auf TGraphen zu spezifizieren, müssen alle Pfade es Graphen, die für diese Prädikat möglicherweise von Bedeutung sind, traversiert werden. Dazu existieren effiziente Algorithmen. Die in den Abbildungen 2 und 5 verwendeten GRAL-Ausdrücke dienen lediglich der Veranschaulichung der Syntax von GRAL. Die vollständige Spezifikation von GRAL findet sich in [Frz97].

Bei den Metamodellen handelt es sich um Sprachbeschreibungen. Um eine Verwendung der Sprachen in einschlägigen Modellierungswerkzeugen zu unterstützen, sind die Metamodelle mit Hilfe von Objektmodellen zu rekonstruieren. Dazu wird die MEMO-OML (Object Modelling Language [Fra98b]) verwendet. Ein Objektmodell ist dem korrespondierenden Metamodell zwar grundsätzlich ähnlich, dennoch ist eine Rekonstruktion nicht trivial. So bietet es sich mitunter an, die reichhaltigeren Konzepte der MEMO-OML (Mehrfachvererbung, Aggregation, Delegation) zu nutzen, um Teile des Metamodells zweckmäßiger zu beschreiben. Daneben ist daran zu denken, daß ein entsprechendes Objektmodell zusätzliche Angaben enthält, die für ein Werkzeug benötigt werden - etwa im Hinblick auf die Versionsverwaltung oder die Codegenerierung. Die Objektmodelle der dedizierten Spracheditoren werden in einem gemeinsamen Modell zusammengeführt, das die Grundlage für die integrierte Modellierungsumgebung MEMOCenter bietet. Die Spracharchitektur von MEMO sowie ihr Verhältnis zum Werkzeugentwurf sind in Abb. 3 dargestellt. Die *semantische* Integration der verschiedenen in MEMOCenter verwalteten Modelle wird durch gemeinsame Konzepte der jeweils verwendeten Sprachen erreicht. Beispielsweise kommt das Konzept "BasicService" sowohl in der MEMO-OML als auch in der MEMO-OrgML (Organisation Modelling Language) vor (Abb. 5). MEMOCenter unterstützt zudem *annotationale* Inte-

gration. Dazu kann der Benutzer zwischen zwei Modellelementen eine Beziehung etablieren, deren Semantik allein durch die Interpretation des zugehörigen Kommentars erfolgt. Beispielsweise kann im Strategiemodell das Konzept "Kundenorientierung" eingeführt werden. Es korrespondiert mit einer Geschäftsregel "Ein Geschäftsprozeß sollte von genau einem Mitarbeiter verwaltet werden.". Diese Regel würde im Organisationsmodell mit Hilfe einer geeigneten Kardinalität ausgedrückt werden, die wiederum mit "Kundenorientierung" assoziiert werden könnte.

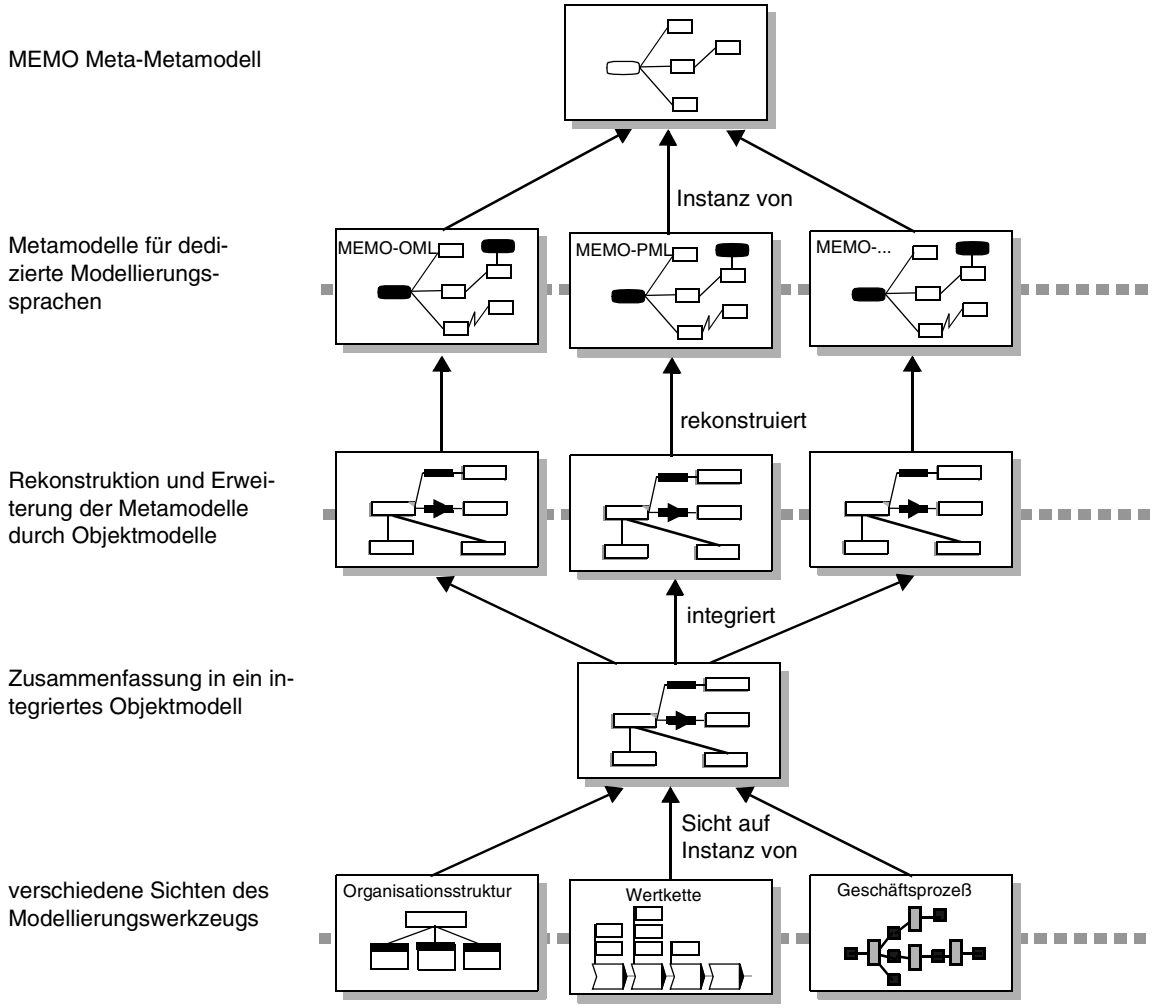


Abb. 3: Spracharchitektur von MEMO und ihre Abbildung auf Werkzeuge

Die betriebliche Realität ist durch eine Reihe von Begriffen gekennzeichnet, die sich gegen eine gehaltvolle Formalisierung sperren (wie etwa "Kundenorientierung", "Wettbewerbsfähigkeit" ...). Dennoch mag es wichtig sein, auch solche Konzepte in einem Unternehmensmodell abzubilden. Die in MEMO angestrebte Formalisierung der Modellierungssprachen bedeutet nicht, daß alle Aspekte des erstellten Modells vollständig formal zu definieren sind. So kann man einerseits Konzepte verwenden, die nur teil-

weise formal beschrieben sind. Andererseits ist daran zu denken, daß auch Konzepte mit spärlicher formaler Semantik durch eine geeignete Instanzierung für den Betrachter reichhaltige Informationen in strukturierter Weise bereitstellen können. So kann beispielsweise das Attribut "Erläuterung" des Konzepts "Unternehmensziel" eine gehaltvolle natürlichsprachliche Beschreibung enthalten. Demgegenüber bietet die Verwendung formaler Sprachen grundsätzlich die Möglichkeit, innerhalb von Modellen Berechnungen oder Beweise durchzuführen sowie Modelle für Simulationen zu nutzen.

Der Entwurf einer graphischen Notation ist eine delikate Aufgabe. Es gibt zwar eine Reihe abstrakter Anforderungen an die Gestaltung einer Notation (z.B. [Rum96]). Im konkreten Fall ist die Evaluation von Notationselementen allerdings schwierig, da die Wahrnehmung und Interpretation einzelner Symbole im Kreise der potentiellen Benutzer erheblich variieren dürfte. Gegenwärtig werden in MEMO solche Notationen verwendet, die in den jeweiligen Einsatzbereichen eine nennenswerte Verbreitung gefunden haben. So wird eine Organisationsstruktur mit Hilfe von Organigrammen dargestellt, gewisse Aspekte der Unternehmensstrategie werden als Wertketten [Por85] visualisiert. Um eine flexible Anpassung an individuelle Präferenzen zu unterstützen, erlaubt MEMOCenter die Modifikation graphischer Symbole.

4. Beispiele ausgewählter Modellierungssprachen

Zur Zeit bietet MEMO drei Modellierungssprachen. Davon ist lediglich die MEMO-OML vollständig in GRAL formalisiert [Zic99], was ebenfalls für große Teile der MEMO-OrgML gilt [Wen97]. Demgegenüber werden die Konzepte wie auch die Notation der MEMO-SML (Strategy Modelling Language, [Rie96]) gegenwärtig einer umfangreichen Revision unterzogen. Erst danach wird eine Formalisierung vorgenommen. Um einen Eindruck von der Unternehmensmodellierung mit MEMO zu vermitteln, werden im folgenden die MEMO-OrgML und die MEMO-OML in Teilen näher betrachtet.

Die objektorientierte Modellierung hat bei der Entwicklung von MEMO eine zentrale Rolle gespielt. Das liegt zum einen daran, daß in einer frühen Phase sämtliche Perspektiven objektorientiert beschrieben wurden. Erst später wurden dedizierte Modellierungssprachen eingeführt - aus Gründen, die in 3 dargestellt sind. Andererseits wird für die Entwicklung von Modellierungswerkzeugen eine Sprache für den Software-Entwurf benötigt. Dabei sprechen wichtige Gründe für einen objektorientierten Ansatz. Mittlerweile zeichnet sich mit der UML eine Standard für objektorientierte Modellierungssprachen ab. Auch wenn die Bedeutung gerade von Sprachstandards unbestritten ist, haben wir uns gegen eine Verwendung der UML entschieden - vor allem wegen mancher Unzulänglichkeiten der UML (eine Evaluation findet sich in [FrPr97]). Damit ist eine Verwendung der UML im Kontext von MEMO allerdings nicht grundsätzlich ausgeschlossen, da die Konzepte der MEMO-OML auf die UML abgebildet werden können - allerdings nicht ohne einen gewissen Verlust an Semantik. Andererseits bietet die MEMO-OML auch

den Anwendern von UML eine hilfreiche Unterstützung, da sie Konzepte spezifiziert, die in der UML vage bleiben (wie etwa die Semantik von Vererbung).

4.1 MEMO-OML

Die MEMO-OML ist eine konsequent objektorientierte Modellierungssprache, deren Konzepte durch Smalltalk inspiriert sind. So werden z.B. Attribute von Klassen mit Hilfe von Klassen (und nicht mit Datentypen) spezifiziert. Jeder Klasse kann zudem eine Metaklasse zugeordnet werden. Im Unterschied zu UML ist in MEMO-OML die Semantik von Vererbung genau spezifiziert. Das drückt sich u.a. darin aus, daß Regeln für die kovariante Redefinition ([Mey97], S. 621 ff.) geerbter Operationen festgelegt sind. Eine solche Definition hat zwar den Nachteil, daß sie von der konkreten Vererbungssemantik der verwendeten Implementierungssprache abweichen kann. Dem steht allerdings der Vorteil gegenüber, daß Modelle, in denen Vererbungsbeziehungen enthalten sind, eine eindeutige Semantik haben. Die Spezifikation von MEMO-OML ist nicht zuletzt wegen der differenzierten Behandlung von Vererbungsregeln sehr umfangreich [Fra98b].

Ein weiterer Unterschied zu UML ist in der Einführung des Konzepts "Delegation" zu sehen. Dabei handelt es sich um eine Beziehung auf Instanzenebene. Ein Objekt ("Rolle"), das eine Nachricht empfängt, für die es selbst über keine entsprechende Operation verfügt, leitet diese Nachricht transparent an ein anderes Objekt ("Rolleninhaber") weiter. Vordergründig weist Delegation Gemeinsamkeiten mit Vererbung auf, es gibt aber auch deutliche Unterschiede. So repräsentieren sowohl ein Rollen- wie auch das assoziierte Rolleninhaber-Objekt jeweils ein Objekt in der abgebildeten Domäne. Im Fall einer Unterklasse wird lediglich ein Objekt - durch die Instanz der Unterklasse - repräsentiert. Eine ausführliche Darstellung von Delegation findet sich in [Fra99].

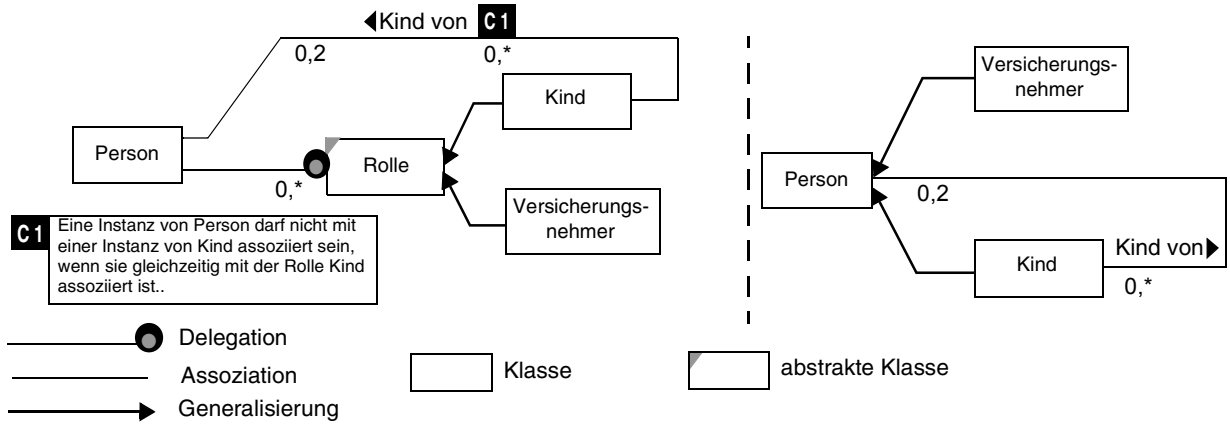


Abb. 4: Delegation (links) vs. Vererbung (rechts)

In Abb. 4 ist ein Beispiel für die Verwendung von Delegation einer entsprechenden Verwendung von Vererbung gegenübergestellt. Es sollte verdeutlichen, daß Delegation hier gegenüber gängigen Formen der Vererbung die bessere Wahl ist: Sobald ein Kind selbst zum Versicherungsnehmer wird, würde Vererbung die Instanzierung eines neuen Objekts der Klasse Versicherungsnehmer und die Übertragung des

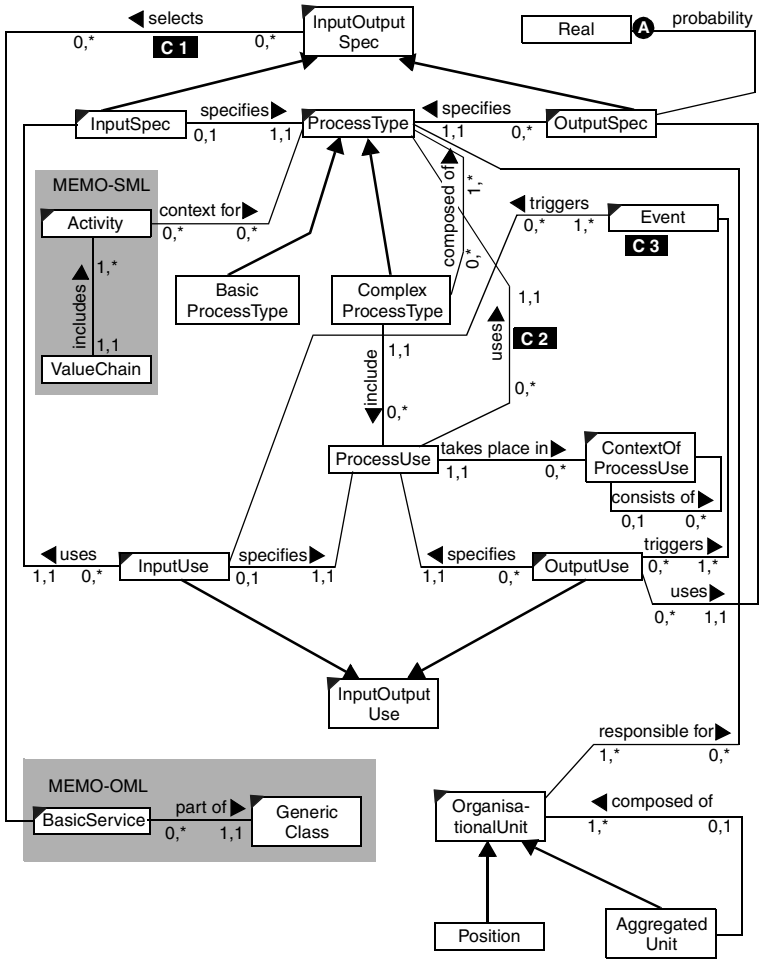
Zustands (einschließlich der Referenzen) der zu löschenden Instanz der Klasse Kind nötig machen. Delegation erfordert lediglich das Anlegen einer neuen Rolle sowie ggfs. das Löschen einer bestehenden Rolle. Der Zustand des zugehörigen Objekts der Klasse Person bleibt unberührt.

4.2 MEMO-OrgML

Üblicherweise wird die Erstellung eines Unternehmensmodells nicht mit dem Entwurf eines Objektmodells beginnen. Statt dessen wird zunächst die zukünftige Unternehmensstrategie zu beschreiben sein. Vor diesem Hintergrund sind wesentliche Geschäftsprozesse und die darauf abgestimmte Organisationsstruktur zu entwerfen. Die organisatorische Gestaltung wird durch die MEMO-OrgML unterstützt. Sie bietet Konzepte zur Beschreibung von Organisationsstrukturen ("OrganisationalUnit", "Position", "Project", ...), operativen Ressourcen ("Faxgerät", "Telefon", "Formular" ...) und von Geschäftsprozessen. Der Beschreibung von Geschäftsprozessen kommt insofern eine besondere Bedeutung zu als sie zusätzliche Sprachkonzepte zur Festlegung temporaler Aspekte benötigt. Im Unterschied zu use cases [Jac+94], die wesentlich auf einer natürlichsprachlichen Beschreibung beruhen, erlaubt MEMO-OrgML eine detailliertere und präzisere Spezifikation von Geschäftsprozessen. Gängige formale Sprachen zur Prozeßmodellierung wie Petri-Netze oder Prozeßalgebren bieten zwar eine präzise Semantik, es fehlen ihnen allerdings i.d.R. anwendungsnahe Sprachkonzepte, die eine übersichtliche Visualisierung auch komplexer Geschäftsprozesse unterstützen (ein Überblick findet sich in [Mar98]). Zudem sind einschlägige Ansätze zur Integration von Petrinetzen mit Objektmodellen (in [Phi99] findet sich ein ausführlicher Vergleich) zumeist durch die Verwendung stark vereinfachter Sprachen für die objektorientierte Modellierung gekennzeichnet. Ereignisgesteuerte Prozeßketten (EPK) stellen einen Kompromiß dar. Sie kommen allerdings für MEMO unmittelbar nicht in Frage, da es ihnen an einer engen Integration mit anderen Sprachen der Unternehmensmodellierung fehlt.

Die zentralen Konzepte zur Prozeßmodellierung in MEMO-OrgML sind ProcessType, ProcessUse, ContextOfProcessUse, InputSpec, OutputSpec und Event. ProcessType ist ein abstraktes Konzept, das in zwei konkrete Konzepte spezialisiert wird: ComplexProcessType und BasicProcessType. Eine Instanz von ComplexProcessType besteht aus n Instanzen von ProcessType. Um zwischen verschiedenen Erscheinungsformen einer Instanz von ProcessType zu unterscheiden, wurde das Konzept ProcessUse eingeführt. Jede Instanz von ComplexProcessType kann aus mehreren Instanzen von ProcessType bestehen, die jeweils genau einer Instanz von ProcessUse zugeordnet werden können. Um ein Beispiel zu geben: In einem Geschäftsprozeß sei eine Instanz des ProcessType "Erstellung des Anwenderhandbuchs" enthalten, die u.a. aus Instanzen des Prozeßtyps "Abbildung erstellen" besteht. Die verschiedenen Erscheinungsformen von "Abbildung erstellen" können durch die Zuordnung jeweils einer Instanz ProcessUse differenziert werden. Die Ausführung eines ProcessType kann eine InputSpec erfordern und

eine oder mehrere OutputSpec erzeugen. Dabei handelt es sich jeweils um Container für Informationen - Formulare, Akten oder Objekte bzw. Objektdienste, die in einem assoziierten Objektmodell spezifiziert sind. Um ggfs. eine Simulation zu ermöglichen, kann jedem OutputSpec eine Wahrscheinlichkeit zugeordnet werden. Abb. 5 zeigt einen Ausschnitt aus dem Metamodell der MEMO-OrgML und illustriert dessen Integration mit anderen Metamodellen mittels gemeinsamer Konzepte.



C 1 Nur solche Instanzen von InputSpec, die einer Instanz von BasicProcessType zugeordnet sind, können eine Instanz von BasicService auswählen.

$$\forall cpt{:}V_{ComplexProcessType} \bullet (cpt \rightarrow_{specifiedby \bullet InputSpec} \rightarrow_{selects}) = \varnothing$$

C 2 Instanzen von ProcessType dürfen nicht zyklisch dekomponiert werden.

$$\forall cpt{:}V_{ComplexProcessType} \bullet \neg(cpt(\rightarrow_{consistsof} \rightarrow_{uses})^{+} cpt)$$

C 3 Es darf keine isolierten Ereignisse geben.

$$\forall ev{:}V_{Event} \bullet \left(degree(et, \rightarrow_{triggers}) > 0 \wedge degree(et, \leftarrow_{triggers}) > 0\right)$$

spezialisiert von
benutzt
Attribut
Instanz von MetaEntity
abstrakte Instanz von MetaEntity

Abb. 5: Ausschnitt aus dem MEMO-OrgML Metamodell

Ereignisse dienen der Spezifikation des Kontrollflusses. Sie entstehen durch das Eintreten bestimmter Prozeßzustände. Es gibt drei Basiskonstrukte für die Spezifikation des Kontrollflusses: Teilprozesse kön-

nen *sequentiell*, *alternativ* oder *parallel* ausgeführt werden. Als abgeleitetes Konstrukt wird zudem eine Schleife angeboten. Mitunter kann ein Geschäftsprozeßtyp als Spezialisierung eines anderen Typs angesehen werden. So mag beispielsweise die Bearbeitung eines Schadens im Bereich der Haftpflichtversicherung der Schadensbearbeitung im Falle einer Feuerversicherung ähnlich sein. Ein gemeinsames, ggfs. abstraktes Oberkonzept könnte also eingeführt werden, um die Vorteile einer entsprechenden Generalisierung zu nutzen. Auch wenn es sich bei der Spezialisierung von Geschäftsprozessen um ein scheinbar intuitives Konzept handelt, ist es uns nicht gelungen, eine befriedigende formale Spezifikation zu erstellen. Deshalb gibt es lediglich die Möglichkeit, Spezialisierungsbeziehungen als ergänzende Information für den Betrachter einzuführen. Bis auf die Einschränkung, azyklisch sein zu müssen, beinhalten sie allerdings keine formale Semantik und damit auch keine Restriktion hinsichtlich der Redefinition "geerbter" Eigenschaften.

Es gibt drei Diagrammarten zur Darstellung von Prozessen. Ein *Dekompositionsdiagramm* dient der Darstellung der Kompositionshierarchie von Prozessen. Ein *Generalisierungsdiagramm* zeigt die erwähnten Generalisierungsbeziehungen zwischen Prozeßtypen. Das *Prozeßdiagramm* erlaubt die detaillierte Beschreibung eines Prozesses. Um unterschiedlichen Ansprüchen an die Detaillierung eines Prozeßmodells gerecht zu werden, bietet MEMO eine vereinfachte ("light") Version der graphischen Notation. Abb. 6 gibt ein Beispiel für die Beschreibung eines Prozeßmodells in MEMOCenter.

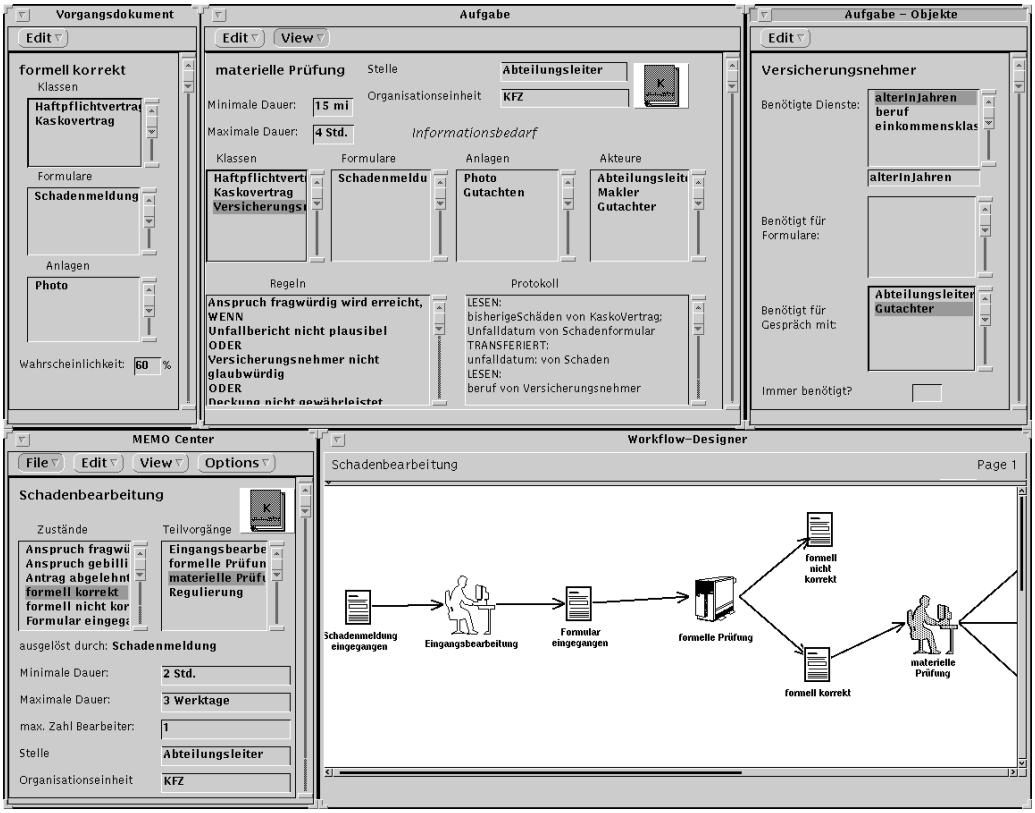


Abb. 6: Editor für die Modellierung von Geschäftsprozessen in MEMOCenter

Ein mit der MEMO-OrgML erstelltes Organisationsmodell unterstützt eine Reihe von Analyseverfahren. So können durch die Zuordnung von Informationstypen zu InputSpec und OutputSpec Medienbrüche ermittelt werden. Die Ermittlung von Flaschenhälsen wird durch die Zuordnung von minimalen und maximalen Ausführungszeiten ermöglicht. Weitere Konzepte dienen der Unterstützung von Simulationen. Dazu werden Instanzen benötigt, die für einen bestimmten Simulationskontext repräsentativ sind. Gleichzeitig ist zu berücksichtigen, daß Simulationen besondere Angaben erfordern mögen. Wenn beispielsweise für eine Simulation ein repräsentativer Angestellter benötigt wird, kann die Anzahl der durch Krankheit pro Jahr ausgefallenen Arbeitstage wichtig sein - eine Eigenschaft, die für die Verwaltung konkreter Instanzen i.d.R. nicht in Frage kommt. Aus diesen Gründen stellt MEMO-OrgML das Konzept einer *prototypischen Instanz* bereit, das mit einer Klasse im Objektmodell assoziiert sein kann, aber letztlich nur der Verwendung im Rahmen von Simulationen dient. Ein Workflow-Modell kann durch geeignete Abstraktionen aus einem Geschäftsprozeßmodell abgeleitet werden. Dazu werden nur solche Teile eines Geschäftsprozeßmodells übernommen, die für die rechnergestützte Verwaltung eines Prozesses relevant sind - also z.B. Ereignisse, die mit Zustandsänderungen von Objekten korrespondieren oder Dienste von Objekten, die in einzelnen Teilprozessen in Anspruch genommen werden. Eine für die Ausführung von Workflow-Spezifikationen geeignete Architektur liegt vor [Nat96], ist aber noch nicht implementiert.

5. Fazit und Ausblick

Die Verwendung graphischer Modelle im Umfeld der Planung und Einführung von Informationssystemen ist gängige Praxis. So werden üblicherweise von Strategieberatern, Organisationsexperten und Systemanalytikern graphische Darstellungen zur Dokumentation der jeweiligen Tätigkeit verwendet. Der perspektivenübergreifende Austausch bzw. die Wiederverwendung solcher Dokumente wird allerdings dadurch erschwert, daß es unterschiedliche Fachsprachen gibt. Das Verständnis der verwendeten graphischen Darstellung ist zudem häufig nur möglich, wenn implizite Konventionen bekannt sind. Die Folge sind neben vordergründigen Kommunikationsproblemen vielfältige Friktionen und vermeidbare Doppelarbeit. MEMO erlaubt es, diesen Schwierigkeiten entgegenzuwirken: Die angebotenen Sprachen reflektieren einerseits die Bedeutung aufgabenspezifischer Konzepte und deren Visualisierung. Gleichzeitig aber wird durch die Integration der Sprachen die Chance verbessert, verschiedene Modelle in einer Werkzeugumgebung gemeinsam zu verwalten. Damit können nicht nur modellübergreifende Integritätsbedingungen überprüft werden. Es entsteht gleichzeitig ein Medium für die perspektivenübergreifende Kommunikation, da eine Navigation zwischen den Teilmodellen möglich wird.

Die bisherige Arbeit an MEMO hat sich vor allem auf die Spezifikation von Modellierungssprachen und die Entwicklung korrespondierender Modellierungswerkzeuge gerichtet. Die Sprachkonzepte wie auch

die graphische Notation wurden durch die Erstellung zahlreicher Modelle evaluiert und mehrfach revidiert. Dabei ist ein größeres Modell mit Referenzcharakter entstanden. Es beschreibt Geschäftsprozesse und Informationsobjekte eines Universitätsinstituts. Auf seiner Basis wurde in Smalltalk ein Framework implementiert, das für den Anwendungsbereich der Literatur- und Medienverwaltung konkretisiert wurde. Ergänzend zu den Sprachen bietet MEMO ein Vorgehensmodell, das mittels der MEMO-Modellierungssprachen beschrieben ist. Es besteht aus zeitlich geordneten Aufgabenblöcken, denen u.a. Ziele, Erfolgskriterien, Rollen, Modelle bzw. Dokumente und sonstige Ressourcen zugeordnet sind. Zudem wurde mit den sog. Ereignisgesteuerten Methodenketten (EMK) eine weitere Prozeßbeschreibungssprache in die Spracharchitektur von MEMO aufgenommen, die eine Schnittstelle zu den in der Praxis weit verbreiteten EPK ermöglicht ([Rit99]).

Der Gegenstand der Unternehmensmodellierung empfiehlt eine interdisziplinäre Kooperation zwischen Wirtschaftsinformatik, Betriebswirtschaftslehre und Informatik. Dazu sind gehaltvolle Schnittstellen im Sinne gemeinsamer Konzepte bzw. Sichtweisen erforderlich. Bisher hat sich vor allem eine ausgesprochen konstruktive Zusammenarbeit mit einer Arbeitsgruppe der Informatik ergeben. Die Schnittstelle dazu waren die zunächst semi-formalen Sprachbeschreibungen, deren Formalisierung ein klassisches Arbeitsgebiet der Informatik darstellt. Die mit dem Entwurf von Referenzmodellen verbundene stärkere Betonung betriebswirtschaftlicher Inhalte bietet Schnittstellen zur Betriebswirtschaftslehre - auch wenn die weitgehend formalen MEMO-Modellierungssprachen dazu sicher kein ideales Medium darstellen. Aber einerseits ist für ein hinreichendes Verständnis der Modelle eine genaue Kenntnis der Sprachspezifikation nicht immer erforderlich, andererseits sind die Modelle ohnehin mit natürlichsprachlichen Annotationen anzureichern. Eine entsprechende Zusammenarbeit stellt in Aussicht, existierende Gestaltungsrichtlinien aus dem Bereich der strategischen Planung oder der Organisationslehre in Form von Entwurfsmustern für die Unternehmensmodellierung nutzbar zu machen. Das wäre ein Beitrag dazu, verschiedene Wissensgebiete der Unternehmensmodellierung in gleicher Weise zu strukturieren, was auf eine Verbesserung der perspektiven- bzw. disziplinübergreifenden Kommunikation hoffen läßt. Dabei entsteht auch ein Berührungspunkt zu einem über verschiedene Disziplinen (allerdings nicht interdisziplinär) angelegten Forschungsgebiet, das die Erstellung, Repräsentation, Pflege und Vermittlung von Wissen zum Gegenstand hat: das sog. Wissensmanagement. Unternehmensmodelle bieten eine konzeptionelle Grundlage für die Verwaltung relevanten Wissens in Unternehmen. Instanzierte Unternehmensmodelle eröffnen durch graphische Visualisierungen und zahlreiche Recherche- und Navigationsmöglichkeiten eine reizvolle Perspektive auf die Repräsentation und Vermittlung organisationaler Wissensbestände.

Literatur

[ESP93] ESPRIT Consortium AMICE (Eds.): CIMOSA: Open System Architecture for CIM. Berlin et al.: Springer 1993

[FeSi97] Ferstl, O.K.; Sinz, E.J.: Grundlagen der Wirtschaftsinformatik. Bd. 1; 3. Aufl., München, Wien: Oldenbourg 1997

[FiHe96] Firesmith, D.; Henderson-Sellers, B.; Graham, I.; Page-Jones, M.: OPEN Modeling Language (OML). Reference Manual. Version 1.0. 8 December 1996 http://www.csse.swin.edu.au/OPEN/comn.html

[Fra97] Frank, U.: Enriching Object-Oriented Methods with Domain Specific Knowledge: Outline of a Method for Enterprise Modelling. Arbeitsberichte des Instituts für Wirtschaftsinformatik, Nr. 4, Universität Koblenz-Landau, 1997

[FrPr97] Frank, U.; Prasse, M.: Ein Bezugsrahmen zur Beurteilung objektorientierter Modellierungssprachen - veranschaulicht am Beispiel vom OML und UML. Arbeitsberichte des Instituts für Wirtschaftsinformatik, Nr. 6, September 1997

[Fra98a] Frank, U.: The MEMO Meta-Metamodel. Arbeitsberichte des Instituts für Wirtschaftsinformatik, Nr. 9, Koblenz 1998

[Fra98b] Frank, U.: The MEMO Object Modelling Language (MEMO-OML). Arbeitsberichte des Instituts für Wirtschaftsinformatk, Nr. 10, Koblenz 1998

[Fra99] Frank, U.: Delegation: An Important Concept for the Appropriate Design of Object Models. Angenommen für: Journal of Object-Oriented Programming

[Frz97] Franzke, A.: GRAL 2.0: A Reference Manual. Fachberichte Informatik, Universität Koblenz-Landau, 1997

[Jac+94] Jacobson, I. et al.: The Object Advantage. Business Process Reengineering with Object Technology. Wokingham 1994

[Mar98] Marx, T.: Software-Entwurf und Workflow-Modellierung mit Petri-Netzen. Dissertation, Universität Koblenz-Landau 1998

[Mey97] Meyer, B.: Object-Oriented Software Construction. 2nd. ed., Upper Saddle River, NJ: Prentice Hall 1997

[Nat96] Nathe, E.: Konzeption und Entwurf eines Workflow-Management-Systems unter Berücksichtigung des prozeßorientierten Managements. Diplomarbeit, Universität Koblenz-Landau 1996

[Phi99] Philippi, S.: Synthese von Petri-Netzen und objektorientierten Konzepten. Dissertation, Universität Koblenz-Landau 1999

[Por85] Porter, M.E.: Competitive Advantage. New York 1985

[Rat97] Rational: UML-Semantics. Version 1.1. 09/01/1997 (http://www.rational.com)

[Rie96] Riemenschnitter, R.: Rechnerunterstützte strategische Planung: Rekonstruktion und objektorientierte Modellierung strategischer Konzepte ausgehend vom Ansatz Porters. Diplomarbeit, Universität Koblenz-Landau 1996

[Rit99] Rittgen, P.: Objektorientierte Analyse mit EMK. In: Rundbrief des GI-Fachausschusses 5.2, 6. Jahrgang, Heft xx, Bamberg, September 1999, S. xx-xx

[Rum96] Rumbaugh, J.: Notation notes: Principles for choosing notation In: Journal of Object-Oriented Programming, Vol. 8, No. 10, May 1996, pp. 11-14

[Sch97] Scheer, A.-W.: Wirtschaftsinformatik. Referenzmodelle für industrielle Geschäftsprozesse. 7. Aufl., Berlin, Heidelberg et al. 1997

[SoZa92] Sowa, J. F.; Zachman, J. A.: Extending and formalizing the framework for information systems architecture. In: IBM Systems Journal, Vol. 31, No. 3, pp. 590-616, 1992

[Wen97] Wenzel, J.: Entwurf einer Modellierungssprache zur Beschreibung von Geschäftsprozessen im Rahmen der Unternehmensmodellierung. Diplomarbeit, Universität Koblenz-Landau 1997

[Zic99] Zickhardt, J.: Formalisierung von MEMO-OML. Diplomarbeit, Universität Koblenz-Landau, Koblenz 1999