Konzeptuelle Modellierung: Obsoleter Kostentreiber oder zentraler Erfolgsfaktor der digitalen Transformation?

4

Ulrich Frank

4.1 Einleitung

Dieser Beitrag ist einem Kollegen gewidmet, der sich wie kaum ein anderer in Forschung und Lehre um die Förderung der konzeptuellen Modellierung betrieblicher Informationssysteme verdient gemacht hat. Da die konzeptuelle Modellierung zu den tragenden Säulen unserer Disziplin gehört und damit zur identitätsstiftenden Profilbildung der Wirtschaftsinformatik beiträgt, hat er so auch einen wesentlichen Beitrag zur Konsolidierung des Fachs geleistet. In den letzten Jahren sind nun Entwicklungen zu verzeichnen, die den Eindruck nahelegen mögen, dass die konzeptuelle Modellierung in Wissenschaft und Praxis an Bedeutung verliert. So ist der Anteil der Forscher in der Wirtschaftsinformatik, die im Bereich der konzeptuellen Modellierung arbeiten, seit einigen Jahren rückläufig. Noch deutlicher wird dieser Schwund, wenn man sich in den Reihen junger Wissenschaftler umschaut. Hier findet man nur noch ganz wenige, zu deren zentralem Forschungsprofil Themen der konzeptuellen Modellierung gehören. Auch wenn wohl eine Reihe unterschiedlicher Aspekte zu dieser Entwicklung beigetragen hat, scheint sie im Wesentlichen auf die Neuausrichtung der Wirtschaftsinformatik zurückzuführen sein, die gegen Ende des letzten Jahrhunderts begann. Dieser Prozess einer Neuorientierung der Disziplin, der noch nicht abgeschlossen ist, ist vor allem durch zwei miteinander verwobene Entwicklungen gekennzeichnet. So haben sich einerseits in der deutschen Universitätslandschaft traditionelle Legitimationsmuster anwendungsorientierter Disziplinen geändert. Während früher das Einwerben beachtlicher Drittmittelvolumina aus der Industrie Ansehen und Unabhängigkeit zur Folge hatten, werden nun mehr und mehr andere Indikatoren wissenschaftlicher

U. Frank (✉)
Universität Duisburg-Essen, Essen, Deutschland
E-Mail: ulrich.frank@uni-due.de

T. Benker et al. (Hrsg.), *Geschäftsprozessorientierte Systementwicklung*,
DOI 10.1007/978-3-658-14826-3_4

Leistungsfähigkeit in den Mittelpunkt gerückt (Schauer 2010). Dabei kommt Publikationen in angesehenen Zeitschriften eine wichtige Rolle zu. Andererseits sah sich die Wirtschaftsinformatik der Forderung gegenüber, ihre Forschung stärker international auszurichten, was nicht zuletzt durch Publikationen in international angesehenen Zeitschriften nachzuweisen sei. Während beide Aspekte für sich genommen überaus sinnvoll erscheinen, sind sie allerdings mit einer bedenklichen Konsequenz verbunden. So hatte das nordamerikanische „Information Systems" eine andere Entwicklung genommen als die Wirtschaftsinformatik. Anstatt die Forschung auf die Gestaltung betrieblicher Informationssysteme auszurichten, wurde die Disziplin eher als eine spezielle Sozialwissenschaft konzipiert, die Voraussetzungen und Folgen des Einsatzes von Informationstechnologie in Organisationen vor allem empirisch nach Maßgabe eines behavioristischen Forschungsmodells untersucht. Gleichzeitig hatten die Kollegen in USA früher damit begonnen, die Reputation ihrer Zeitschriften und Konferenzen gezielt zu fördern (Frank et al. 2008). In der Folge erschien es einer zunehmenden Zahl von jungen Wissenschaftlern sinnvoll, die Wissenschaftskonzeption von „Information Systems" zu adaptieren, um so einen besseren Zugang zu international hoch angesehenen Zeitschriften und Konferenzen zu finden.

Es gibt wohl zwei weitere Gründe dafür, dass das Interesse an der Forschung zur konzeptuellen Modellierung in der Wirtschaftsinformatik nachgelassen hat. So stößt man mitunter auf die Ansicht, dass es doch schon genug, wenn nicht gar zu viele Modellierungsansätze gäbe. Weitere Forschung sei also gleichsam nicht geeignet, neue Erkenntnisse oder einen zusätzlichen Nutzen für die Praxis zu schaffen. Daneben hat vor allem ein ambitioniertes Forschungsprogramm der konzeptuellen Modellierung nicht die erhofften Ergebnisse erbracht: Der Entwurf von Referenzmodellen galt lange Zeit als ein idealtypisches Forschungsziel, da es einen hohen wissenschaftlichen Anspruch mit einer überaus attraktiven Perspektive für die praktische Nutzung verband. Es ist allerdings bis auf wenige Ausnahmen nicht gelungen, große Referenzmodelle in der wissenschaftlichen Forschung zu entwickeln oder sie gar in der Praxis als solche zu etablieren. Dies führte zu einer gewissen Ernüchterung und mitunter anzutreffenden grundsätzlichen Zweifeln an der Machbarkeit von Referenzmodellen. Davon abgesehen gibt es aber durchaus eine Reihe eindrucksvoller Beispiele für einen gelungenen Transfer wissenschaftlich entwickelter Modellierungsansätze in die Praxis. Neben dem ARIS-Ansatz von Scheer (Scheer 1991), dem darauf aufbauenden „Handels-H" von Becker und Schütte (Becker und Schütte 2004) ist hier vor allem an die Arbeiten von Sinz zu denken. Seine Erweiterung des ER-Modells, das „SERM" (Sinz 1990) fand Eingang in die Systementwicklung des weltweit größten Anbieters von Unternehmenssoftware. Das später zusammen mit Ferstl entwickelte SOM (Ferstl und Sinz 2008) bewährte sich in einer beachtlichen Zahl von Praxisprojekten.

Die Rolle der konzeptuellen Modellierung in der Praxis ist allerdings ambivalent. In vielen Software-Entwicklungsprojekten stellen der Entwurf und die Pflege von Modellen nicht nur ein zentrales Instrument zur Förderung der Kommunikation zwischen Anwendern und Entwicklern dar. Sie werden vielmehr darüber hinaus als notwendige Voraussetzung für die Realisierung anforderungsgerechter Software angesehen. Zudem haben Ansätze zur modellgetriebenen Software-Entwicklung in den letzten Jahren eine beachtliche Resonanz

gefunden (France und Rumpe 2007). Sinz hat auf diesem Feld wichtige Beiträge zur Transformation von Geschäftsprozessmodellen in ausführbare Workflow-Schemata geleistet (Krücke und Sinz 2011; Pütz und Sinz 2010).

Darüber hinaus hat die zunehmende Komplexität von IT-Infrastrukturen den Bedarf an Modellierungsansätzen zur Unterstützung des IT-Managements gefördert. Dedizierte Ansätze der Unternehmensmodellierung wie auch des verwandten *Enterprise Engineering* haben auf unterschiedlichen Abstraktions- und Detaillierungsniveaus Eingang in die Praxis gefunden. Neben solchen Anzeichen für die Wertschätzung der Modellierung in der Praxis finden sich aber auch einige gegenläufige Indizien. So wird in manchen Unternehmen die konzeptuelle Modellierung als eher obsoletes Beiwerk angesehen, das erhebliche Kosten verursacht, aber vom Kunden, der ja nur an der Software interessiert sei, nicht geschätzt wird. Daneben wird ein grundsätzliches Argument bemüht, das Machbarkeit und Sinnhaftigkeit der Modellierung in Frage stellt: Die umfassende Modellierung einer Domäne erfordert einen erheblichen Kosten- und Zeitaufwand. Die irgendwann zu erwartenden Ergebnisse seien durch die bis dahin veränderte organisatorische Realität längst überholt (Ciborra 1987) – mit entsprechend unerfreulichen Folgen für den Investitionsschutz.

Ich werde im Folgenden zu zeigen versuchen, dass die konzeptuelle Modellierung in Wissenschaft und Praxis trotz eines mitunter schwierigen Umfelds nach wie vor zum Fundament der Wirtschaftsinformatik gehört und dass es gute Gründe dafür gibt, dass sie auch in Zukunft ein zentrales Forschungsthema darstellen sollte.

4.2 Konzeptuelle Modellierung im Rahmen der Systementwicklung

Traditionell wird die konzeptuelle Modellierung vor allem als ein Instrument der Analyse und des Entwurfs von Software-Systemen angesehen. Dabei kann zwischen generischen („general purpose“) Ansätzen und solchen, die auf die Besonderheiten bestimmter Domänen („domain-specific“) gerichtet sind, unterschieden werden.

4.2.1 Generische Modellierungsansätze

Die Entwicklung von Anwendungssystemen ist darauf gerichtet, Programme zu implementieren, die die Anforderungen prospektiver Anwender erfüllen. Die Ausführung eines Programms wird letztlich durch die Ausführung von Maschineninstruktionen realisiert, die für einen Prozessor spezifiziert sind. Um die Lesbarkeit von Programmen zu verbessern und eine höhere Produktivität zu erreichen, wird von maschinennahen Sprachen abstrahiert. Stattdessen werden zumeist höhere Programmiersprachen eingesetzt. Auch höhere Programmiersprachen sind allerdings nicht geeignet, ein Programm in einer Weise zu beschreiben, die für Anwender verständlich ist. Die Systementwicklung muss deshalb die Barriere zwischen der Sprache der Anwender und den jeweils verwendeten Programmiersprachen

überwinden. Darüber hinaus empfiehlt die Komplexität großer Software-Systeme Abstraktionen, die die Gesamtkomplexität wirksam reduzieren. Die konzeptuelle Modellierung trägt beiden Aspekten Rechnung. Zum einen ermöglicht sie, anders als Programmcode, die Betrachtung ausgewählter Gestaltungsdimensionen eines Software-Systems. Hier ist vor allem an statische, funktionale und dynamische Abstraktionen zu denken. Zum anderen unterstützt sie eine systematische Abbildung von Begriffen, die in der jeweils relevanten Diskurswelt verwendet werden, auf Konzepte von Implementierungssprachen. Dies wird durch eine auffällige Parallele zwischen den Grundlagen natürlichsprachlicher Begriffsbildung und den elementaren Konzepten des Entwurfs von Software-Systemen ermöglicht. Die philosophische Ontologie zielt seit jeher auf die Erfassung sprachlicher Basiskonzepte, auf die alle Ausdrücke der menschlichen Sprache(n) – und damit gleichsam die Verfasstheit der uns zugänglichen Welt – reduziert werden können. Unter den vielen einschlägigen Vorschlägen in der Philosophie finden sich einige (z. B. Grossmann 1983; Bunge 1977), die weitgehend mit den Basiskonzepten der Systementwicklung übereinstimmen. So spricht Bunge u. a. von „things“, „properties of things“, „classes of things“ und „regelkonformen Zuständen“. Grossmann zählt u. a. „classes“, „properties“ und „relations“ zu den ontologischen Basisbegriffen. Die Parallelen zu Konzepten der Systementwicklung wie „Entität“, „Klasse“, „Attribut“, „Zustand“ und „Integritätsbedingung“ sind offensichtlich. Dies ist insofern erstaunlich als Bunge und Grossmann ihre Vorschläge ohne Kenntnis der Systementwicklung entworfen haben und umgekehrt die Entwickler generischer Modellierungssprachen wie des ERM, der UML, von DFDs oder von Zustandsdiagrammen vermutlich dazu nicht auf die Philosophie rekurrierten. Die Verwendung generischer Basiskonzepte ermöglicht es nun mit Hilfe generischer Modellierungssprachen („general purpose modelling languages“, „GPML“) natürlichsprachlich beschriebene Sachverhalte in einer Struktur abzubilden, die mit der Struktur von Implementierungssprachen direkt korrespondiert. Diese intermediäre Funktion von Modellierungssprachen ist in Abb. 4.1 illustriert.

Konzeptuelle Modelle bieten eine Reihe offenkundiger Vorteile. Sie bieten eine gemeinsame Referenz zur Förderung der Kommunikation zwischen Software-Entwicklern und Anwendern. Darüber hinaus unterstützen sie den Entwurf leistungsfähiger Informationssysteme, indem sie die Abstraktion auf Gemeinsamkeiten verschiedener Anwendungssysteme fördern und damit einen wichtigen Beitrag zur Wiederverwendung und zur Systemintegration leisten. Die Bereitstellung von Modellierungssprachen allein ist allerdings kaum eine hinreichende Unterstützung der erfolgreichen Durchführung von Modellierungsprojekten. Vielmehr ist dazu ein ergänzendes Vorgehensmodell angezeigt, das zusammen mit den jeweils verwendeten Modellierungssprachen eine Modellierungsmethode bildet. Generische Modellierungsmethoden sind dabei auf Vorgehensmodelle beschränkt, die sich auf eine Vielzahl von Projekten anwenden lassen.

Generische Modellierungsansätze haben einen offensichtlichen Vorteil: Sie erlauben eine weite Einsatzbandbreite. Der Einsatz korrespondierender Werkzeuge kann damit von erheblichen Skaleneffekten profitieren. Zudem ist zu erwarten, dass am Arbeitsmarkt qualifizierte Fachkräfte verfügbar sind, die mit entsprechenden Ansätzen vertraut sind. Dies gilt besonders für Ansätze, die auf verbreiteten generischen Modellierungssprachen wie der UML oder dem ERM beruhen.

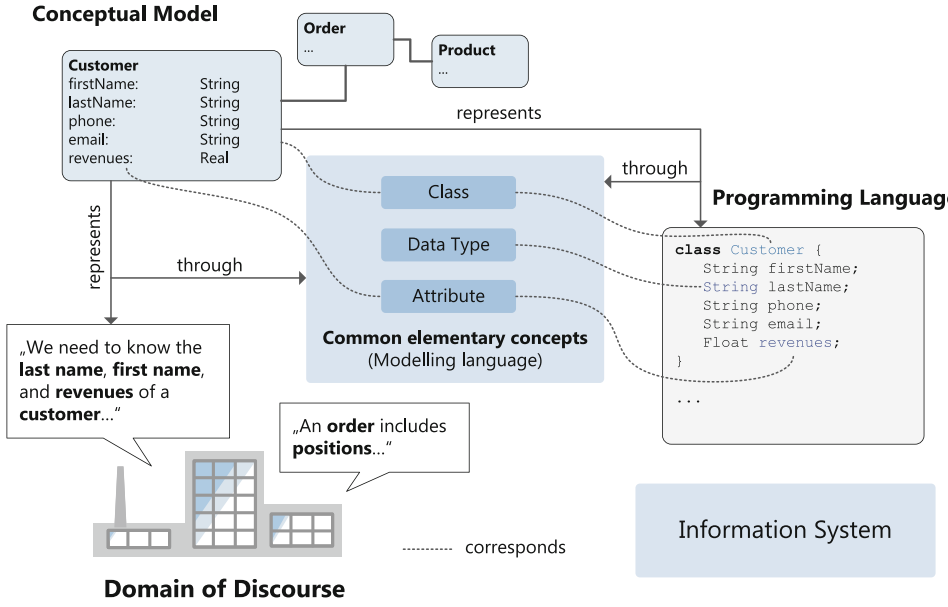

Abb. 4.1 Illustration der Transformationsfunktion generischer Modellierungssprachen

4.2.2 Domänenspezifische Modellierungsansätze

Den Vorteilen generischer Modellierungsansätze stehen gravierende Nachteile gegenüber. Dieser Umstand wird besonders deutlich, wenn man sich vorstellt mit einer generischen Sprache wie der UML kommunizieren zu müssen. Eine solche Sprache würde lediglich primitive Konzepte wie „Klasse", „Attribut", „Assoziation" etc. bereitstellen. Jeder Kommunikationsgegenstand müsste dann mit diesen Konzepten beschrieben werden. Die Konsequenz ist offensichtlich: Die Effizienz von Kommunikation wäre drastisch reduziert; zudem wäre die Gefahr von Missverständnissen erheblich. Die Erstellung von Modellen mit generischen Modellierungssprachen wie dem ERM oder der UML erfolgt aber genau nach diesem Muster! Domänenspezifische Modellierungssprachen (DSML) sind darauf gerichtet, Modellierer mit Konzepten zu unterstützen, die aus der Fachsprache einer Domäne rekonstruiert wurden. Auf diese Weise wird einerseits die Produktivität der Modellierung deutlich erhöht, da der Modellierer domänenspezifische Begriffe nicht mehr selbst unter Rückgriff auf primitive Basiskonzepte spezifizieren muss. Andererseits fördern DSML die Integrität und damit die Qualität von Modellen. Die abstrakte Syntax und Semantik dieser Sprachen trägt dazu bei, unsinnige Modelle in einem deutlich größeren Umfang zu verhindern als dies für GPML der Fall ist. Darüber verfügen DSML zumeist über eine dedizierte konkrete Syntax, wodurch die Verständlichkeit von Diagrammen gefördert wird. Abb. 4.2 veranschaulicht diese Unterschiede zwischen DSML und GPML mit Hilfe eines Beispiels zur Modellierung von IT-Infrastrukturen. Während die GPML es ermöglicht, eine unsinnige Assoziation wie „Server läuft auf ERP-System" zu definieren, würden die einer korrespondierenden DSML inhärenten Integritätsbedingungen dies verhindern.

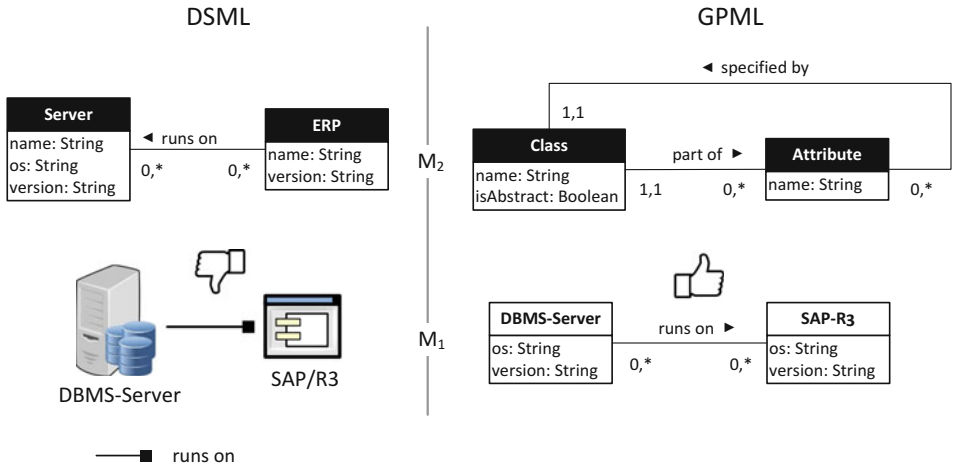

Abb. 4.2 Gegenüberstellung von DSML und GPML

Die Wirkungsweise von DSML zur Unterstützung der Abbildung von Fachsprachen auf Implementierungssprachen wird in Abb. 4.3 illustriert. Um Friktionen zu vermeiden, wird dazu wie im Beispiel dargestellt idealtypisch auch eine korrespondierende domänenspezifische Programmiersprache verwendet. Das Beispiel macht zudem deutlich, dass der Entwurf von DSML mit einer erheblichen Herausforderung verbunden ist: Es muss entschieden werden wie speziell die Sprachkonzepte sein sollen – wodurch letztlich ein Zielkonflikt adressiert wird: Je spezieller die Konzepte, desto höher ihr potenzieller Beitrag zur Förderung von Modellierungsproduktivität und Modellintegrität, desto geringer aber die Wiederverwendungsreichweite und damit die ökonomischen Skaleneffekte.

Ergänzend zu DSML unterstützen domänenspezifische Bezugsrahmen die Erstellung und Analyse von Modellen für bestimmte Domänen. In der Wirtschaftsinformatik kommt dabei Bezugsrahmen zur Strukturierung von Unternehmen, wie sie in der Unternehmensmodellierung eingesetzt werden (z. B. Zachman 1987; Frank 2013), eine große Bedeutung zu. Während entsprechende DSML eine Strukturierung von Unternehmen im Detail unterstützen, kommt den domänenspezifischen Bezugsrahmen eher die Funktion zu, ein abstraktes Modell, man könnte auch sagen: einen grundlegenden Begriff, von Unternehmen zu entwerfen. Ein solches Modell dient dazu, unter den Beteiligten ein gemeinsames Verständnis des Unternehmens zu fördern. Gleichzeitig bietet es einen Interpretationsrahmen für die Erfassung möglicher Probleme sowie Ansatzpunkte zur schrittweisen Verfeinerung einer daran anknüpfenden Untersuchung. Der von Ferstl und Sinz eingeführte systemtheoretisch/kybernetische Bezugsrahmen (Ferstl und Sinz 2008) erfüllt diesen Anspruch in mehrfacher Hinsicht sehr überzeugend. So stellt die Konzeptualisierung des Unternehmens als soziotechnisches System, dessen Untersuchung die Identifikation von Regelkreisen empfiehlt, nicht nur ein in der Betriebswirtschaftslehre (Ulrich 2001; Kosiol et al. 1965) und – in verschiedenen Ausprägungen – in der Soziologie (Parsons 1951; Luhmann 1984; Giddens 1984) anerkanntes Untersuchungsparadigma dar. Darüber hinaus ist die systemtheoretische Perspektive in den

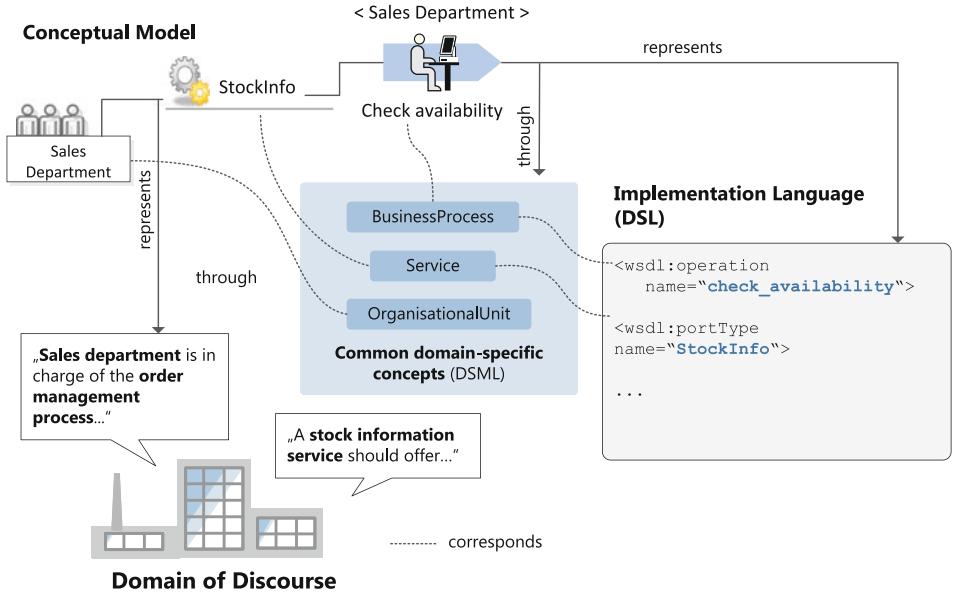

Abb. 4.3 Illustration der Transformationsfunktion domänenspezifischer Modellierungssprachen

Ingenieurwissenschaften weit verbreitet. Damit ist der SOM-Ansatz gut geeignet, betriebswirtschaftliche und technische Perspektiven zu unterstützen und zu integrieren. Gleichzeitig reflektiert die besondere Betonung eines Transaktionskonzepts eine Handlungstheorie, die es nicht nur erlaubt, die Erstellung von Unternehmensmodellen systematisch anzuleiten, sondern auch die Überprüfung von Modellen auf Vollständigkeit unterstützt. Dadurch bereichert SOM die Methoden der Unternehmensmodellierung um eine besondere Perspektive. Nur der Ansatz von Dietz (Dietz 2006) beinhaltet ebenfalls ein elaboriertes Transaktionskonzept zur Strukturierung wirtschaftlichen Handelns.

4.2.3 „Agile" Methoden

Der prospektive Nutzen konzeptueller Modellierung ist offensichtlich: Sie fördert die strukturierte Betrachtung eines komplexen Gegenstands zusammen mit verschiedenen Gruppen. Domänenspezifische Ansätze stellen dem Modellierer darüber hinaus noch wiederverwendbares Wissen in Form durchdachter Konzeptualisierungen des jeweiligen Gegenstands und korrespondierender Vorgehensweisen bereit. Die modellbasierte Entwicklung von Software-Systemen ist dennoch in die Kritik geraten. Insbesondere sogenannte agile Ansätze (z. B. Beck und Andres 2004; Cockburn 2002) legen den Eindruck nahe, dass die Erstellung von Modellen für die erfolgreiche Bewältigung von Software-Entwicklungsprojekten eher hinderlich als förderlich ist. Die agile Bewegung wurde von häufig akademisch qualifizierten Software-Entwicklern auch als Kritik an den Methoden der Informatik, die an den

Universitäten gelehrt werden, initiiert. Diese Kritik ist in Teilen berechtigt. So wird in traditionellen Methoden der Informatik vor allem auf die Feinheiten einer formalen Spezifikation fokussiert und dabei außer Acht gelassen, dass Software-Entwicklung zumeist einen intensiven Austausch nicht nur mit den Anwendern, sondern auch zwischen den Entwicklern erfordert. Es gehört deshalb zu den zentralen Prinzipien der agilen Ansätze, die Schaffung produktiver Kommunikationsbedingungen zu betonen. Zudem, auch dies eine Kritik an traditionellen Ansätzen, die mitunter als Kritik an der Modellierung verstanden wird, sollen Auswüchse einer Projektbürokratie, die sich etwa in einem ausufernden Dokumentationswesen äußern, vermieden werden. Darüber hinaus basieren agile Ansätze auf der gleichsam ontologischen Annahme, dass Anforderungen einem fortlaufenden Wandel unterworfen sind, weshalb eine umfassende Modellierung gar nicht lohnend ist. Vielmehr solle man der Komplexität des Gegenstands dadurch begegnen, dass man das Gesamtsystem in überschaubare Komponenten herunterbricht, die dann möglichst rasch zu implementieren sind. Dem unvermeidbaren Wandel wird dann durch entsprechende Anpassungen begegnet, wozu auf diverse „Refactoring"-Muster zurückgegriffen werden kann.

Auch wenn agile Ansätze wichtige Aspekte betonen, die in traditionellen Software-Entwicklungsmethoden fälschlicherweise vernachlässigt wurden, ginge es zu weit, darin eine überzeugende Kritik an der konzeptuellen Modellierung zu sehen. Das hat mehrere Gründe. So ist die konzeptuelle Modellierung ja nicht zuletzt darauf gerichtet, die Kommunikation zwischen Entwicklern und Anwendern zu fördern, was ganz im Sinn der Proponenten agiler Verfahren sein sollte. Insbesondere die Unternehmensmodellierung ist darauf gerichtet, die prospektiven Anwender einzubinden und den Entwurf von Systemen zu fördern, die an den Zielen von Unternehmen ausgerichtet sind. Daneben ist es unstrittig, dass Modellierung, also Abstraktion, das einzige Mittel darstellt, um ein System so zu konstruieren, dass es komfortabel und sicher an mögliche Anforderungsänderungen angepasst werden kann. Der populäre Hinweis, man möge doch erst mal implementieren, bevor man allzu viel Zeit auf die Modellierung verschwendet, kann von wenig erfahrenen Entwicklern leicht missverstanden werden. Entsprechende Software ist dann auch nicht mehr mit ausgefeilten Refactoring-Verfahren zu retten. Tatsächlich schließen agile Methoden die Modellierung per se keinesfalls aus. Vielmehr weiß jeder erfahrene Verfechter agiler Verfahren, dass es kontraproduktiv wäre, auf konzeptuelle Modelle zu verzichten. Insofern wird die Bedeutung der konzeptuellen Modellierung durch agile Verfahren kaum in Frage gestellt. Vielmehr – und das ist durchaus positiv – betonen sie die Notwendigkeit, neben der Modellierung auch den sozialen und wirtschaftlichen Kontext der Systementwicklung angemessen zu berücksichtigen.

4.3 Die Nutzung von Modellen zur Laufzeit von Anwendungssystemen

Traditionell wurden konzeptuelle Modelle vor allem als Analyse- und Entwurfsinstrumente angesehen, waren also weitgehend auf die Systementwicklung beschränkt. Es war allerdings schon immer offensichtlich, dass Modelle auch eine deutlich bessere Grundlage für

die Wartung von Software-Systemen darstellen als Code. In der Praxis hat sich jedoch gezeigt, dass die Software-Wartung zumeist unter großem Zeitdruck erfolgt und deshalb oft direkt am Code durchgeführt wird. Eine spätere Synchronisierung von Code und korrespondierenden Modellen findet dann zumeist wegen des damit verbundenen Aufwands nicht statt. Bei näherer Betrachtung wird deutlich, dass Modelle nicht nur die Wartung von Software-Systemen verbessern könnten, sondern auch einen wichtigen Beitrag zu einer vielseitigeren und anspruchsvolleren Nutzung von Anwendungssystemen leisten können. Um zu verstehen, wie ein solcher Beitrag aussehen könnte, ist es sinnvoll sich vor Augen zu führen, dass eine Nutzung von Software ohne Modelle nicht möglich ist. Software-Systeme sind linguistische Artefakte und damit immateriell. Sie sind in einer für die meisten Anwender unverständlichen (Maschinen-) Sprache realisiert. Software kann nur sinnvoll genutzt werden, wenn sie durch eine an die Sprache der Anwender angelehnte Repräsentation ergänzt wird. Mit anderen Worten: Wir können Software nur nutzen, wenn wir ein Modell dieser Software haben, das uns eine sinnvolle Interpretation der von der Software verwalteten Daten und der darauf verfügbaren Operationen erlaubt. Entsprechende Modelle werden allerdings zumeist nicht mit der Software bereitgestellt. Vielmehr wird das Modell, das sich ein Anwender von der jeweils genutzten Software bildet durch die Benutzungsschnittstelle vermittelt. Einen weiteren Beitrag leisten Handbücher, Schulungsmaßnahmen und die Lernprozesse, die mit der Nutzung der Software verbunden sind. Die so entstehenden Modelle sind aber häufig unvollständig und irreführend – mit der Folge, dass Anwender die Software nur unzureichend verstehen. Das führt nicht nur dazu, dass Software nicht effizient genutzt wird, sondern auch dazu, dass Anwendungssysteme eine Entfremdung von der mit ihnen verrichteten Arbeit fördern: Man folgt den (vermeintlichen) Vorgaben einer kaum verstandenen Maschine und verliert den Blick dafür, in welchem Zusammenhang der eigene Beitrag mit den Zielen des Unternehmens steht.

Es deutet vieles darauf hin, dass die Repräsentation von Software durch geeignete konzeptuelle Modelle in Zukunft immer wichtiger wird. Organisationen werden immer weiter von Software durchdrungen. Mitarbeiter nehmen ihr Tätigkeitsumfeld und mehr und mehr das ganze Unternehmen durch die Software-Systeme wahr, mit denen sie arbeiten. Kunden bilden sich ihr Bild von einem Unternehmen immer weniger durch physisch vermittelte Eindrücke wie sie etwa durch den Besuch eines Kaufhauses entstehen oder durch die Interaktion mit Vertretern eines Unternehmens. Stattdessen wird das Unternehmen in der Wahrnehmung der Kunden mehr und mehr durch die Software repräsentiert, über die Transaktionen angebahnt und durchgeführt werden – überspitzt formuliert: Die Software *ist* das Unternehmen. Daraus folgt einerseits, dass Mitarbeiter und andere Akteure das Unternehmen nur dann verstehen, wenn sie die Software verstehen. Dabei geht es nicht allein um die vordergründige Benutzung von Software. Wann immer ein Mitarbeiter den Bereich des Unternehmens, für den er verantwortlich ist, neuen Gegebenheiten anpassen möchte, sollte er dazu beurteilen können, mit welchem Aufwand die u.U. erforderlichen Änderungen an der jeweils genutzten Software verbunden sind – und bestenfalls diese Änderungen selbst durchführen können. Es ist offensichtlich, dass die Repräsentation von Software durch Code den meisten Anwendern nicht zugänglich wäre. Konzeptuelle Modelle, die Begriffe

widerspiegeln, die dem Anwender vertraut sind, wären dazu deutlich besser geeignet. Modelle, die mit ihren Metamodellen integriert sind, in diesem Sinn also reflexiv sind, würden die Anwender zudem darin unterstützen, sich die konzeptuelle Grundlage einer Software, also etwa die Definition wichtiger Sprachkonzepte, selbstständig zu erschließen und ggfs. anzupassen. Wenn darüber hinaus die Änderung von Modellen eine Änderung des korrespondierenden Codes nach sich ziehen würde, könnte ein dermaßen unterstützter Anwender die Software, die er benötigt, selbst ändern. Eine solche Form des „user empowerment" durch Modelle wird seit einiger Zeit diskutiert (z. B. Krogstie 2007), eine Umsetzung sieht sich aber grundlegenden Beschränkungen gängiger Programmiersprachen gegenüber: Da sie nur über eine Klassifikationsebene verfügen, müssen Modelle in Modelleditoren durch Objekte auf M_0 dargestellt werden – auch wenn sie tatsächlich auf M_1 oder höher angesiedelt sind. Daraus folgt zunächst, dass die entsprechenden Objekte nicht weiter instanziiert werden können. Das wiederum impliziert, dass Code aus Modellen nur generiert werden kann und dass es zwei getrennte Repräsentationen von Modellen und Code geben muss – was zu den bekannten Synchronisationsproblemen führt.

Sprachen, die mehrere Klassifikationsebenen ermöglichen (Clark et al. 2008; Frank 2014), erlauben dagegen die Abbildung von Modellen auf dem jeweils intendierten Klassifikationsniveau und damit eine gemeinsame Repräsentation von Modellen und Code. Damit werden Systeme möglich, die zur Laufzeit über ein Modell ihrer selbst verfügen. Darüber hinaus können sie zudem mit Modellen der Umgebung, in der sie eingesetzt werden, angereichert werden. Ein Beispiel dafür stellt selbstreferenzielle Unternehmenssoftware dar (Frank und Strecker 2009). Entsprechende Systeme integrieren Unternehmenssoftware mit einem korrespondierenden Unternehmensmodell. Dadurch erhalten Nutzer einerseits die Möglichkeit durch ein Modell der Software zu navigieren. Andererseits haben sie Zugang zu einem damit integrierten Modell des Handlungssystems. Auf diese Weise können sie nachvollziehen, wie Software und der jeweils unterstützte Handlungskontext (z. B. ein Geschäftsprozess) zusammenhängen und ggfs. gemeinsam geändert werden können. Abb. 4.4 illustriert die verschiedenen Abstraktionsebenen eines solchen Systems und die sich daraus ergebenden Navigationsmöglichkeiten. Selbstreferentielle Unternehmenssoftware ist also geeignet die Mündigkeit von Anwendern zu fördern („user empowerment").

4.4 Zur Wirtschaftlichkeit der Modellierung

Konzeptuelle Modellierung verursacht beachtliche Kosten. Sie erfordert den Einsatz besonders qualifizierter Mitarbeiter und ist häufig mit einem erheblichen, vorab schwer einschätzbaren Zeitaufwand verbunden. Zudem sind größere Modellierungsprojekte mit dem Risiko verbunden zu scheitern. Gleichzeitig ist der Nutzen der Modellierung schwer erfassbar und in seiner Gesamtheit kaum quantifizierbar. Es ist deshalb nachvollziehbar, wenn Entscheidungsträger, die sich einer auf quantifizierten Kosten/Nutzen-Relationen basierenden Rationalität verpflichtet fühlen, Modellierungsprojekten skeptisch gegenüberstehen. Berichte aus der Praxis über Modellierungsprojekte, die trotz eines immensen

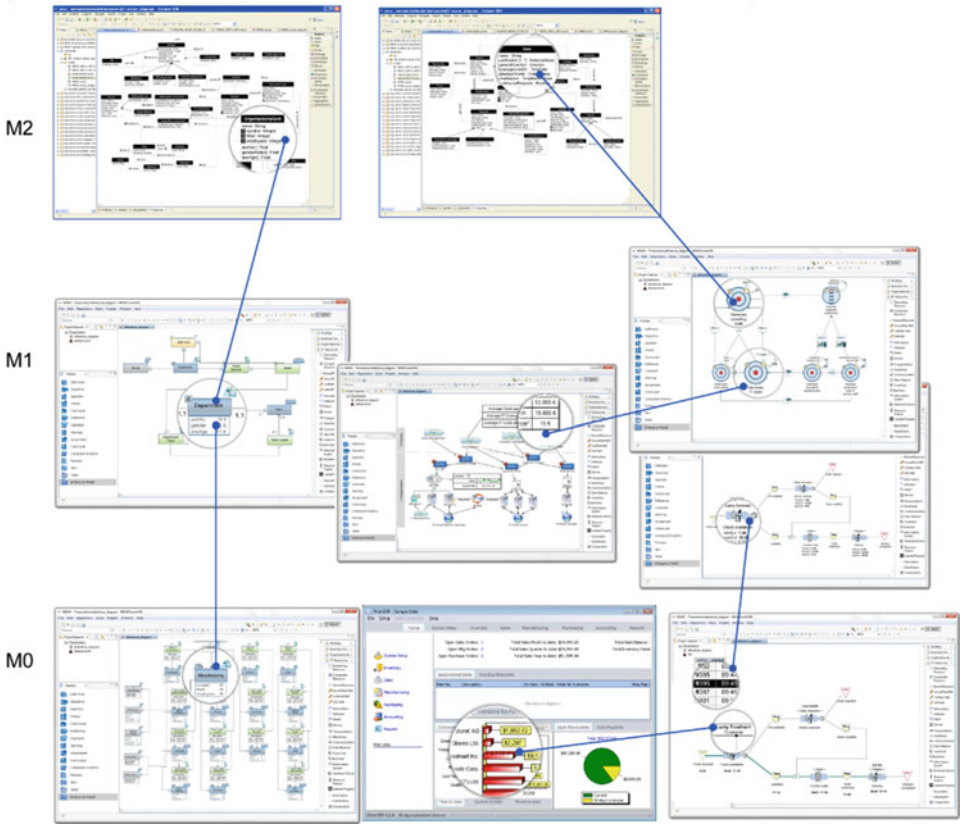

Abb. 4.4 Illustration der Klassifikationsebenen eines selbstreferenziellen Unternehmenssoftware-Systems

Aufwands gescheitert sind, tun ein weiteres, um diese Skepsis zu fördern. Demgegenüber sind Berichte über einen hohen finanziellen Nutzen, der durch Modellierung ermöglicht wurde, eher selten zu finden (z. B. Pick und Klein 2002). Die Forschung hat bisher wenig getan, um Zweifeln an der Wirtschaftlichkeit der konzeptuellen Modellierung dezidiert entgegenzuwirken. Dieser Umstand ist wohl vor allem auf zwei Gründe zurückzuführen. So geht die Modellierungsforschung von der optimistischen – aber wie wir noch sehen werden: durchaus gut begründbaren – Präsupposition aus, dass Modellierung geeignet ist, einen substanziellen Wertbeitrag zu schaffen. Zum anderen sind die Ermittlung und vor allem die Quantifizierung eines Modellierungsnutzens mit schwerwiegenden epistemologischen Problemen verbunden (Frank 2000). Sinnvoller erscheinen deshalb Ansätze, die darauf gerichtet sind, die möglichen Wirkungen der Modellierung transparent zu machen. Wolff führt dazu eine Modellierungssprache ein, die es ermöglicht vor der Durchführung von Modellierungsprojekten Nutzen und Risiken der Modellierung zu erfassen (Wolff 2008). Daneben sind Studien hilfreich, die darauf gerichtet sind, Voraussetzungen erfolgreicher

Modellierung zu identifizieren. Hier ist neben Sprachen und Werkzeugen nicht zuletzt an den relevanten organisatorischen Kontext zu denken (Maier 1996).

Die Schwierigkeiten, die mit einer Quantifizierung des Nutzens der Modellierung verbunden sind, sind allerdings kaum geeignet, grundlegende Zweifel am Nutzen der Modellierung zu hegen. Ein solches Muster würde formal dem Argument entsprechen, dass Führungskräfte nicht nützlich seien, weil sich ihr Nutzen nicht exakt erfassen lässt („competence is hard to judge …", Perrow 1986, S. 11) – oder, näher an unserem Thema: dass man den Nutzen von IT bestreitet, weil er nicht überzeugend gemessen werden kann. Auch der – berechtigte – Hinweis auf Risiken, die mit Modellierungsprojekten verbunden sind, ist kaum geeignet, Modellierung per se in Frage zu stellen. Modellierungsprojekte mögen scheitern; doch das gilt sicher auch für Software-Entwicklungsprojekte im Allgemeinen. Niemand käme deshalb auf die Idee, den Nutzen von Software anzuzweifeln.

Während die Argumente gegen die Nützlichkeit der Modellierung bei näherer Betrachtung schnell verblassen, gibt es eine Reihe schwerwiegender Argumente dafür, dass die konzeptuelle Modellierung von zentraler Bedeutung für den Entwurf und die Nutzung betrieblicher Informationssysteme ist. Sie betreffen zunächst die Leistungsfähigkeit und die Wirtschaftlichkeit von Informationssystemen. Es ist unstrittig, dass die Integration von Anwendungssystemen eine wichtige Voraussetzung für die effiziente Nutzung von Informationssystemen darstellt. Integration wird dadurch erreicht, dass die zu integrierenden Anwendungssysteme zum Zweck der Kommunikation auf gemeinsame Konzepte verweisen können. Statische Integration erfordert dazu gemeinsame statische Konzepte zur Beschreibung von Daten (Datentypen, Klassen etc.). Funktionale Integration wird ermöglicht durch die Erfassung von Funktionen, die anwendungsübergreifend bekannt sind. Dynamische Integration schließlich erfordert neben einem Prozessschema gemeinsame Ereignistypen. Diese gemeinsamen Konzepte werden idealtypisch unabhängig von den Programmiersprachen der zu integrierenden Anwendungen beschrieben, um die Integration weiterer Anwendungssysteme, die in anderen Sprachen implementiert wurden, zu vereinfachen. Dazu ist von den Besonderheiten der Konzepte einzelner Anwendungssysteme auf gemeinsame Konzepte zu abstrahieren. Konzeptuelle Modelle sind deshalb in besonderer Weise geeignet, die für Integrationsziele bedeutsamen Gemeinsamkeiten einer Menge von Anwendungen darzustellen.

Wiederverwendung ist wohl der wichtigste Hebel zur Förderung der Wirtschaftlichkeit der Software-Entwicklung. Es sind also Artefakte zu entwerfen, die den Anforderungen einer größeren Zahl von Anwendungssystemen entsprechen. Auch hier gilt es, durch Abstraktion auf gemeinsame Konzepte die Grundlage für die Entwicklung wiederverwendbarer Artefakte zu schaffen. Die Anpassbarkeit von Software ist ein weiterer kritischer Erfolgsfaktor. Dies gilt umso mehr in Zeiten des raschen digitalen Wandels. Um Software auf sich ändernde Anforderungen vorzubereiten, sind Modelle erforderlich, die den als invariant angesehenen Kern der Software von den im Zeitverlauf möglicherweise variablen Teilen separieren. Dazu werden leistungsfähige Abstraktionen benötigt, die Gemeinsamkeiten heutiger und zukünftiger Anforderungen erfassen und eine möglichst komfortable und sichere (also möglichst monotone) Verfeinerung zulassen. Dazu ist auch ein intensiver

Austausch mit Domänenexperten erforderlich. Allein dadurch ist eine unmittelbare Darstellung in Code weitgehend ausgeschlossen. Auch der oben skizzierte Einsatz von Modellen zur Laufzeit verspricht eine Anreicherung von Software, die einen deutlichen Zusatznutzen in Aussicht stellt.

Neben diesen positiven Wirkungen der Modellierung auf die Wirtschaftlichkeit von Informationssystemen gibt es ein grundsätzliches Argument für die Sinnhaftigkeit der konzeptuellen Modellierung. Die Entwicklung, der Einsatz und die Pflege von Informationssystemen erfordern die Analyse des jeweiligen Einsatzkontextes, die Betrachtung möglicher Architekturalternativen, die Erfassung möglicher Problemkonstellationen u.v.m. All dies ist uns nur möglich, indem wir uns den Gegenstand der Betrachtung durch Begriffe erschließen: „Es gibt aber, außer der Anschauung, keine andere Art, zu erkennen, als durch Begriffe. Also ist die Erkenntnis eines jeden, wenigstens des menschlichen, Verstandes eine Erkenntnis durch Begriffe, nicht intuitiv, sondern diskursiv." (Kant 1974, B 93, A 68). Wer also meint, konzeptuelle Modellierung sei entbehrlich, sagt damit nichts Anderes als dass Denken entbehrlich sei. Das heißt im Übrigen natürlich nicht, dass Modellierung *immer* zu nützlichen Ergebnissen führt. Denken kann ja bekanntlich auch in die Irre führen. Im Unterschied zu Kant, aber mit ähnlicher Intention, spricht Wartofsky explizit von Modellen, die er als intentionale Repräsentationen auffasst, mit denen Wissen angeboten wird: „Models are proffered truths. To proffer truth is the human means of acquiring knowledge. In this sense, cognitive acquisition, human learning is essentially mediated by representation. … there is no knowledge without representation" (Wartofsky 1979, S. xviii).

4.5 Abschließende Anmerkungen

Die umfassende Transformation von Wirtschaft und Gesellschaft ist noch in ihrem Anfangsstadium. Wir wissen nicht, wie sie sich im Einzelnen vollziehen wird. Wir wissen allerdings, dass Arbeitswelt und Alltag mehr und mehr von Informationstechnologie durchdrungen werden. Die Wirtschaftsinformatik gehört zu den Disziplinen, die im Zentrum dieses Wandels stehen. Es gehört deshalb zu unseren vornehmsten Pflichten, eine Unterstützung dafür zu bieten, wie die digitale Transformation in systematischer und reflektierter Weise zu gestalten ist anstatt sie lediglich mit Interesse zu beobachten und zu kommentieren. Dazu muss es gelingen, mit einer Komplexität umzugehen, die deutlich über die heutiger Informationssysteme und ihrer Einsatzkontexte hinausgeht: Es geht nicht mehr allein darum, Systeme für relativ stabile Anforderungen zu gestalten, sondern sie so zu entwerfen, dass sie an verschiedene zukünftige Entwicklungspfade anpassbar sind. Wir benötigen also Modelle, die noch deutlich mehr als dies bisher der Fall war, mögliche zukünftige Welten antizipieren – nicht nur, um Systeme dadurch flexibler zu machen, sondern auch, um eine gehaltvolle Grundlage für eine differenzierte Analyse von Handlungsoptionen, die durch den digitalen Wandel eröffnet werden, zu schaffen. Bernd Mahr drückt diese zentrale Rolle von Modellen sehr treffend aus: „Mit Modellen machen wir uns die Wirklichkeit des Vergangenen und die Möglichkeiten des Zukünftigen zur Gegenwart" (Mahr 2015, S. 329)

Die Modellierungsforschung ist dabei keineswegs auf ihren traditionellen Fokus, also die Analyse und den Entwurf von Informationssystemen beschränkt. Vielmehr können die von ihr entwickelten Sprachen und Modelle auch für andere Themenfelder, die bisher der Betriebswirtschaftslehre vorbehalten waren, gewinnbringend genutzt werden. So ist etwa das Rechnungswesen das klassische Informationssystem der Unternehmen. Die traditionellen Modelle der Kosten- und Leistungsrechnung werden aber nach wie vor weitgehend unabhängig von einer Abbildung auf Software-Systeme betrachtet. Die Nutzung geeigneter DSML oder konzeptioneller Bezugsrahmen verspricht eine bessere Integration betriebswirtschaftlicher Terminologien in betriebliche Informationssysteme und könnte zudem einen Beitrag zur dringend gebotenen engen Zusammenarbeit zwischen Wirtschaftsinformatikern und Betriebswirten leisten.

Modellierungsforschung ist damit geeignet den identitätsstiftenden Kern der Wirtschaftsinformatik zu bilden, da sie sich durch die Fokussierung auf Sprachen, Methoden und Werkzeuge zur gemeinsamen Modellierung von Handlungssystemen und den sie unterstützenden Informationssystemen deutlich von ihren Nachbardisziplinen differenzieren kann. Gleichzeitig kann sie eine gerade für die Bewältigung des digitalen Wandels bedeutsame Integration von Erkenntnisangeboten aus verschiedenen Disziplinen unterstützen, um so eine differenzierte, multiperspektivische Sicht auf dieses komplexe Phänomen zu fördern und die notwendige interdisziplinäre Kooperation anzuregen (vgl. Frank et al. 2014). Der von Sinz und Ferstl entwickelte systemtheoretisch/kybernetische Bezugsrahmen zur Modellierung von Unternehmen und den in diesen eingesetzten Informationssystemen ist dazu in besonderer Weise geeignet, weil er auf ein generelles erkenntnistheoretisches Paradigma (Bertalanffy 1968) zurückgreift, das nicht nur auf den Untersuchungsgegenstand vieler Disziplinen anwendbar ist, sondern auch geeignet ist, die Änderung von Systemen zu untersuchen.

Die Entwicklung und Erprobung von Abstraktionen, die es uns ermöglichen eine zunehmend durch digitale Artefakte geprägte Zukunft differenziert zu denken und zu gestalten, ist eine überaus anspruchsvolle wissenschaftliche Aufgabe. Sie empfiehlt den Entwurf neuer Sprachen, die benutzerorientierte Aufbereitung von Modellen, die Integration von Modellierungswerkzeugen und Anwendungssystemen. Die Forschung sieht sich dabei einer Reihe erheblicher Herausforderungen gegenüber. So ist der Aufwand, der mit dem Entwurf von Modellen, Modellierungssprachen und -werkzeugen verbunden ist, so groß, dass die Kapazität einzelner Lehrstühle kaum ausreicht, um umfassende Lösungen, die auch für den Transfer in die Praxis geeignet sind, zu realisieren. Das empfiehlt die Bündelung von Ressourcen und die Fokussierung auf gemeinsam zu entwickelnde Artefakte. Ein solcher Ansatz steht aber in deutlichem Kontrast zu einer wichtigen Säule der Organisation von Wissenschaft: dem Wettbewerb der Erkenntnisangebote. Es sind hier also Formen der wissenschaftlichen Zusammenarbeit zu entwickeln, die Synergien freisetzen aber gleichzeitig die Differenzierung individueller Beiträge nicht ausschließen.

Darüber hinaus sieht sich die Modellierungsforschung vor dem Hintergrund des tief greifenden digitalen Wandels einem subtilen, aber essenziellen Problem gegenüber: Wandel vollzieht sich nicht zuletzt sprachlich. Vordergründig bedeutet dies, dass wir die

Einschränkungen gegenwärtiger Modellierungs- und Programmierparadigmen untersuchen sollten, um sie ggfs. durch neue Paradigmen zu ersetzen. Im Hinblick auf die konzeptuelle Modellierung bedeutet dies aber vor allem, dass wir die Begriffe, die uns unsere Sprache gibt, die unser wichtigstes Werkzeug sind, hinterfragen müssen. Nur so können wir die „Grenzen" unserer Welt (Wittgenstein 1961) überwinden, um neue mögliche Welten denken zu können. Ein solches Unterfangen ist mit großer Unsicherheit verbunden, da wir gleichsam den Boden unter unseren Füßen demontieren während wir gehen. Derrida spricht deshalb in diesem Zusammenhang von einer „absoluten Gefahr", da die „künftige Welt … die Werte von Zeichen, gesprochenem Wort und Schrift in ihr erschüttert haben wird" und wir heute noch keine Sprache dafür haben, diese Zukunft zu beschreiben (Derrida 1984, S. 15). Man mag dies als eine philosophisch überzeichnete Sicht auf die Herausforderungen der Modellierung in Zeiten des Wandels ansehen. Man kann darin aber auch den besonderen intellektuellen Reiz wie auch die spezifische gesellschaftliche Verantwortung der Modellierungsforschung sehen. Unabhängig davon ist es unstrittig, dass Modellierungssprachen und -methoden weiterentwickelt werden müssen, um den sich im Wandel befindlichen Konturen unserer Welt weiter gerecht zu werden, sie also erkennbar zu machen. Um die dazu nötige Kompetenz zu entwickeln, kommt der Lehre eine zentrale Bedeutung zu (vgl. dazu auch Sinz 2008). Dazu reicht es allerdings nicht aus, sich auf traditionelle Einführungen in die Datenmodellierung oder die UML zu beschränken und durch einen Überblick über die Geschäftsprozessmodellierung zu ergänzen. Vielmehr geht es darum, Studierenden zu vermitteln, dass Modellierungssprachen und -methoden zu den wichtigsten Werkzeugen der Wirtschaftsinformatik gehören, die man nur dann erfolgreich einsetzen kann, wenn man sie sich erschlossen hat und ihre spezifischen Einsatzvoraussetzungen und -grenzen differenziert beurteilen kann.

Das Feld, das Elmar Sinz ganz wesentlich mit bestellt hat, ist nach wie vor überaus fruchtbar. Es wird auch in Zukunft Erkenntnisse zeitigen, die unser Verständnis davon, wie wir Informationssysteme entwerfen, nutzen und pflegen, weiter vertiefen, und die darüber hinaus eine zentrale Grundlage dafür darstellen, die Herausforderungen, die mit der digitalen Transformation verbunden sind, zu meistern.

Literatur

Beck K, Andres C (2004) Extreme programming explained: embrace change. Addison-Wesley, Boston

Becker J, Schütte R (2004) Handelsinformationssysteme: Domänenorientierte Einführung in die Wirtschaftsinformatik. Redline Wirtschaft, Frankfurt am Main

Bertalanffy L (1968) General system theory: foundations, development, applications. Braziller, New York

Bunge M (1977) Treatise on basic philosophy: volume 3: ontology I: the furniture of the world. Reidel, Dordrecht

Ciborra C (1987) Reframing the role of computers in organizations: the transaction costs approach. Off Technol People 3(1):17–38

Clark T, Sammut P, Willans J (2008) Superlanguages: developing languages and applications with XMF. Ceteva, Sheffield

Cockburn A (2002) Agile software development. Addison-Wesley, Boston
Derrida J (1984) Grammatologie. Suhrkamp, Frankfurt am Main
Dietz JLG (2006) Enterprise ontology: theory and methodology. Springer, Berlin/New York
Ferstl OK, Sinz EJ (2008) Grundlagen der Wirtschaftsinformatik. Oldenbourg, München
France RB, Rumpe B (2007) Model-driven development of complex software: a research roadmap. In: Briand LC, Wolf AL (Hrsg) Workshop on the Future of Software Engineering (FOSE '07). IEEE CS Press, Minneapolis, S 37–54
Frank U (2000) Evaluation von Artefakten in der Wirtschaftsinformatik. In: Heinrich LJ (Hrsg) Evaluation und Evaluationsforschung in der Wirtschaftsinformatik. Handbuch für Praxis, Lehre und Forschung. Oldenbourg, München, S 35–48
Frank U (2013) Multi-perspective enterprise modeling: foundational concepts, prospects and future research challenges. Software and systems modeling. http://www.springerlink.com/openurl.asp?genre=article&id=doi:10.1007/s10270-012-0273-9
Frank U (2014) Multilevel modeling: toward a new paradigm of conceptual modeling and information systems design. Bus Inform Sys Eng 6(6):319–337
Frank U, Strecker S (2009) Beyond ERP systems: an outline of self-referential enterprise systems: requirements, Conceptual foundation and design options. Essen
Frank U, Schauer C, Wigand R (2008) Different paths of development of two information systems communities: a comparative study based on peer interviews. Commun AIS 22:391–412
Frank U, Strecker S, Fettke P, vom Brocke J, Becker J, Sinz EJ (2014) The research field „modeling business information systems". Bus Inf Sys Eng 6(1):39–43
Giddens A (1984) The constitution of society: outline of the theory of structuration. Polity Press, Cambridge
Grossmann R (1983) The categorical structure of the world. Indiana University Press, Bloomington
Kant I (1974) Kritik der reinen Vernunft, Bd 1. Suhrkamp, Frankfurt am Main
Kosiol E, Szyperski N, Chmielewicz K (1965) Zum Standort der Systemforschung im Rahmen der Wissenschaften. ZfbF 17:337–378
Krogstie J (2007) Modelling of the people, by the people, for the people. In: Krogstie J, Opdahl A, Brinkkemper S (Hrsg) Conceptual modelling in information systems engineering. Springer, Berlin/Heidelberg, S 305–318
Krücke A, Sinz EJ (2011) Entwurf partieller SOA auf der Grundlage von Geschäftsprozessmodellen. In: Sinz EJ, Bartmann D, Bodendorf F, Ferstl OK (Hrsg) Dienstorientierte IT-Systeme für hochflexible Geschäftsprozesse. University of Bamberg Press, Bamberg, S 287–312
Luhmann N (1984) Soziale Systeme: Grundriß einer allgemeinen Theorie. Suhrkamp, Frankfurt am Main
Mahr B (2015) Modelle und ihre Befragbarkeit Grundlagen einer allgemeinen Modelltheorie. Erwägen Wissen Ethik 26(3):329–342
Maier R (1996) Benefits and quality of data modelling – results of an empirical analysis and definition of a framework for quality management. In: Thalheim B (Hrsg) Proceedings of the 15th international conference on conceptual modelling ER '96. Springer, Berlin, S 245–260
Parsons T (1951) The social system. Free Press u. a, Glencoe
Perrow C (1986) Complex organizations: a critical essay. McGraw-Hill, New York
Pick RA, Klein G (2002) Model management as a component of a knowledge management system: capturing modeling knowledge in the enterprise. In: 8th Americas conference on information systems. S 226–232
Pütz C, Sinz EJ (2010) Model-driven derivation of BPMN workflow schemata from SOM business process models. Enterp Model Inf Sys Archit 5(2):57–72
Schauer C (2010) Die Wirtschaftsinformatik im internationalen Wettbewerb: Vergleich der Forschung im deutschsprachigen und nordamerikanischen Raum. Gabler, Wiesbaden

Scheer A-W (1991) Architektur integrierter Informationssysteme: Grundlagen der Unternehmensmodellierung. Springer, Berlin/New York

Sinz EJ (1990) Das Entity-Relationship-Modell (ERM) und seine Erweiterungen. Handbuch der modernen Datenverarbeitung 27(152):17–29

Sinz EJ (2008) Modellierung in der Wirtschaftsinformatik-Weiterbildung. In: Desel J, Glinz M (Hrsg) Modellierung in Lehre und Weiterbildung. Insititut für Informatik, Zürich, S 7–16

Ulrich H (2001) Die Unternehmung als produktives soziales System: Grundlagen der allgemeinen Unternehmungslehre. Haupt, Bern

Wartofsky MW (1979) Models: representation and the scientific understanding. Reidel, Dordrecht

Wittgenstein L (1961) Tractatus logico-philosophicus. Routledge & Paul/Humanities Press, London/New York

Wolff F (2008) Ökonomie multiperspektivischer Unternehmensmodellierung: IT-Controlling für modell-basiertes Wissensmanagement. Gabler, Wiesbaden

Zachman JA (1987) A framework for information systems architecture. IBM Sys J 26(3):276–292