Quality Assurance Methods and the Open Source Model

Knöll, Heinz-Dieter; Otte, Tobias ; Moreton, Robert

Published in:
Wirtschaftsinformatik - Ein weites Feld

Publication date:
2008

Document Version
Publisher's PDF, also known as Version of record

18. Jahrgang, Heft 1, Juni 2008, ISSN 0939-8821

LEUPHANA
UNIVERSITÄT LÜNEBURG

FinAL
Forum Informatics At Leuphana

Wirtschaftsinformatik

„Ein weites Feld"

Festschrift für
Prof. Dr. Horst Meyer-Wachsmuth

Information Science: Technical Reports and Working Papers
Hrsg: Hinrich E. G. Bonin
Volgershall 1, D-21339 Lüneburg
Phone: xx49.4131.6775175 Fax: xx49.4131.6775140

Inhaltsverzeichnis

Prof. Dr. H. Meyer-Wachsmuth
mw@uni.leuphana.de

Jahrgang: 1942

Ausbildung:

∗ Abitur	1962
∗ Diplom	1969 (Festkörperphysik, Universität Hamburg)
∗ Promotion	1973 (Hochenergiephysik, Universität Hamburg)

Berufliche Entwicklung:

∗ 1969 - 1973	wissenschaftlicher Mitarbeiter beim Deutschen Elektronen-Synchrotron (DESY) in Hamburg
∗ 1973 - 1975	wissenschaftlicher Assistent an der RWTH Aachen
∗ 1975 - 1982	Hauptabteilungsleiter "Systemplanung" bei der Bertelsmann AG
∗ seit 1982	Professor für "Systemtechnik" im FB W der Fachhochschule Nordostniedersachsen

während dieser Zeit unter anderem tätig als

- Leiter des Rechenzentrums der Fachhochschule und Universität in Lüneburg (bis 1988)
- Dekan (1991 - 1993)
- Vorsitzender EDV-Kommission von Fachhochschule und Universität (mehrere Wahlperioden)
- Senator (mehrere Wahlperioden)

∗ 1998 - 2005	Vizepräsident der FH Nordostniedersachsen bzw. Universität Lüneburg

Sonstiges:

∗ 1988 - 1992	Mitglied des Vorstands von SAVE
∗ 1987	Praxissemester bei Siemens AG (DI) mit Forschungsschwerpunkt ”RZ-Automatisierung”
∗ 1993	Praxissemester bei Siemens Nixdorf AG mit Forschungsschwerpunkt ”Softwareentwicklungsmethodik”
∗ seit 1982	vielfältige Managementberatung

Hobbies:

Reiten, Tennis, Windsurfen

Wirtschaftsinformatik *. . . ein weites Feld . . .*

— Festschrift für Prof. Dr. Horst Meyer-Wachsmuth —

Vorwort des Herausgebers

Wirtschaftsinformatiker beschäftigen sich mit Gestaltung und Betrieb von Systemen der computergestützten Informationsverarbeitung für betriebswirtschaftliche Aufgaben.
↪[Me+04]

Maschinisierung von Kopfarbeit

Unter dem wörtlich genommenen Motto „*Ein weites Feld*“[1] enthält dieser Band der FINAL-Reihe[2] ein weit gefächertes Angebot über Aspekte und Fragen zum Verstehen der Disziplin *Wirtschaftsinformatik* (WI). Was ist WI? Was vermittelt der Begriff WI? Wie kann WI definiert werden — in der Vergangenheit, in der Gegenwart und in der Zukunft?

Diese Art von „Was-ist-Fragen“ ist eigentlich nur noch für Dispute im Kreis von Philosophen (in ihrem Elfenbeinturm?) interessant. Schnell geht es dabei hauptsächlich um feinste Feinheiten der Begriffsdefinitionen. Nun hat gerade *Horst Meyer-Wachsmuth* stets betont, dass man sich nicht auf unfruchtbare Dispute einlassen soll, sondern dass es gilt, die konkreten Probleme der Praxis in den Fokus zu nehmen. Seinem Rat wird hier gefolgt. Statt einer formalen Definition der Disziplin WI wird nur angenommen: WI sei eine Wissenschaft, die sich primär mit Informationen im Kontext von Maschinen befasst, oder anders formuliert geht es um die „*Maschinisierung von Kopfarbeit*“ (↪ [Na92]). Damit stellt sich die Frage nach der Begriffsklärung von „Information“.[3] Im Allgemeinen[4] werden dabei zwei Aspekte genannt (↪ z. B. [Rech04] S. 92):

1. Information ist gedeutete (interpretierte) Nachricht oder Mitteilung. Sie entsteht erst beim Empfänger der Nachricht.
2. Information an sich, als reale Erscheinung, die für sich selbst existiert, die transportiert und womöglich gemessen, in Teile zerlegt oder aus Teilen zusammengesetzt werden kann, gibt es nicht. Der Begriff „Information“ ist vielmehr eine Verdinglichung des Informierens (*Hypostasierung*, wie die Philosophen sagen), so wie viele abstrakte Substantive Hypostasierung sind, zum Beispiel „Sein“, „Nichts“, „Identität“, „Denken“.

Praxisrelevant ist eine Unterscheidung in syntaktische und semantische Information (↪ z. B. [Rech03] S. 321):

- Syntaktische Information
  - ist ein Maß für die kürzeste Codierung der Nachricht
  - ist quantifizierbar
  - braucht keinen Empfänger, steckt objektiv in der Nachricht
  - ist bestimmt durch Alphabet und Auftrittswahrscheinlichkeit seiner Zeichen

[1] Dieses Zitat entstammt dem berühmten Roman „Effi Briest“ (1895) von Theodor Fontane. Im Jahre 1995 hat es der Nobelpreisträger Günter Grass zum Titel seines Werkes über die Wiedervereinigung von BRD und DDR gewählt (↪ [Grass95]).

[2] ↪ `http://final.uni-lueneburg.de/` (online 13-May-2008)

[3] Das Diskussionsfeld ist beim Begriff „Information“ sehr weit und reicht tief in die Vergangenheit zurück. So wird beispielsweise Moses von lateinischen Autoren *informator populi* genannt und bei Thomas von Aquin heißt Bildung „informatio“. (↪[Kl03] S. 268)

[4] Allerdings gilt noch immer die These: *Das Informationszeitalter kann sich nicht einigen über den Begriff „Information“.* (↪[Kl03] S. 267)

- Semantische Information
  - bezeichnet die Bedeutung einer Nachricht für den Empfänger
  - ist nicht quantifizierbar
  - braucht einen Empfänger und entsteht erst bei ihm
  - ist bestimmt durch die Bedeutung für den Empfänger

Eine Antwort auf die Frage nach dem notwendigen Selbstverständnis der Disziplin WI zeichnet sich ab:

- durch eine Beschreibung des Verhältnisses zur Informatik einerseits und zur Betriebswirtschaft andererseits und
- anhand der Erläuterung von charakteristischen Elementen.

WI ≡ Informatik + Betriebswirtschaft

WI ist diejenige Informatik,
die für Unternehmen und Verwaltungen nützlich ist.
WI-Motto des Umfeldes von *Horst Meyer-Wachsmuth*

Ist die Disziplin WI eine selbständige *Angewandte Informatik* oder eine besondere *Anwendung der Informatik*? Die erste Auffassung entspricht eher einer Aggregationsbeziehung zwischen Informatik und WI. Die WI ist dann gleichzeitig (Bestand-)Teil der Disziplin der Informatik und der Betriebswirtschaft (↪Abbildung 1). Die zweite Auffassung betont eher eine Vererbungsbeziehung. Die WI ist dann eine spezielle Ausrichtung der Disziplinen Informatik und Betriebswirtschaft. Beide sind in der Elternrolle und übertragen auf die WI ihr gesamtes Wissen und Instrumentarium (↪Abbildung 2). Diese unterschiedlichen Zuordnungen der WI zu den Disziplinen Informatik und Betriebswirtschaft lassen sich durch eine formale Notation betonen. In beiden Abbildungen ist die WI als Klasse notiert und zwar gemäß der *Unified Modeling Language* (UML).[5]

Elemente der WI

Unstrittig kann davon ausgegangen werden, dass der Gegenstand der Disziplin WI mindestens die fünf wesentlichen Elemente umfaßt: Hardware, Software, Organisation, Nutzer und Betroffene. Im konkreten Arbeitsbereich der WI haben diese eng miteinander verzahnten Elemente ganz unterschiedliche Ausprägungen. Solche unterschiedlichen Ausprägungen beeinflussen zwangsläufig das individuelle Bild von der WI-Disziplin.

WI-(Denk)Welt

Zum Beispiel kann die WI-(Denk)Welt primär geprägt sein von persönlichen Erfahrungen durch den Umgang mit einem Notebook oder mit einem voll computerisierten Unternehmen. Ein verantwortlicher Konzernmanager wird beispielsweise seine Erfahrungen mit großen Rechenzentren und mit vielen hunderten Computern im Netz in den Mittelpunkt seiner WI-Betrachtung stellen. Beide Sichten, Erklärungsmuster, Erwartungen, Einschätzungen, kurz WI-(Denk)Welten, unterscheiden sich total.

Horst Meyer-Wachsmuth war und ist ein engagierter Befürworter der Position, dass WI-Lehre, WI-Forschung und WI-Technologietransfer sich auf konkrete Erfahrungen mit leistungsfähigen Rechenzentren abstützen muss. Die heute oft, weil leicht praktizierbare WI-Lehre mit Notebooks der

[5] UML ↪ `http://www.uml.org/` (online 13-May-2008)

Studierenden ist nicht hinreichend, um sich eine praxisrelevante WI-(Denk)Welt zu erschließen. *Horst Meyer-Wachsmuth* war daher stets ein wesentlicher Promotor für den Auf- und Ausbau eines eigenen Rechenzentrums der Leuphana Universität Lüneburg (zunächst für die beiden organisatorisch getrennten Institutionen *Fachhochschule Nordostniedersachsen* und *Universität Lüneburg*, vormals *Pädagogische Hochschule Lüneburg*). Frühzeitig hat *Horst Meyer-Wachsmuth* dieses Rechenzentrum als Knoten in das *Deutsche Forschungsnetz* (DFN[6]) integriert (↪Abbildung 3). Dieses von der Wissenschaft selbst organisierte Kommunikationsnetz verbindet Hochschulen und Forschungseinrichtungen miteinander und ist nahtlos in den europäischen und weltweiten Verbund der Forschungs- und Wissenschaftsnetze eingepasst. Die DFN-Mitgliedschaft ist damit eine hervorragende „Quelle“ für die frühzeitige Umsetzung von Innovationen und Trends in die WI-Lehre, WI-Forschung und den WI-Technologietransfer.

Klar ist, dass eine aktive Mitwirkung in relevanten Gremien der WI zwingend geboten ist, wenn in Lehre, Forschung und Technologietransfer die aktuelle WI-(Denk)Welt zum Tragen kommen soll. Konsequenterweise hat *Horst Meyer-Wachsmuth* sich daher für eine Mitwirkung in einer Anwendervereinigung von Großrechnern entschieden. Während in anderen Institutionen das *Programming-in-the-Small* die Hauptsäule ihres WI-Engagements bildete, konnte *Horst Meyer-Wachsmuth* seine SAVE[7]-Vorstandserfahrungen einbringen und vermehrt das *Programming-in-the-Large*[8] in den WI-Fokus rücken. Aus dieser WI-Perspektive war es logisch, dass *Horst Meyer-Wachsmuth* sich beispielsweise mit *Transaktionsmonitoren*[9] und *Online Transaction Processing* befasste und sein WI-Kollegium für diese schwierigen, aber praxisrelevanten Themen motivierte. Schon im Rahmen seines Studiums (1962 – 1969) hatte er das „*Programmieren elektronischer Datenverarbeitungssysteme*“[10] intensiv praktiziert und später als verantwortlicher IT-Manager[11] und als IT-Unternehmensberater[12] mit der strategischen IT-Planung kombiniert.

Das „*weite Feld*“ wächst permanent

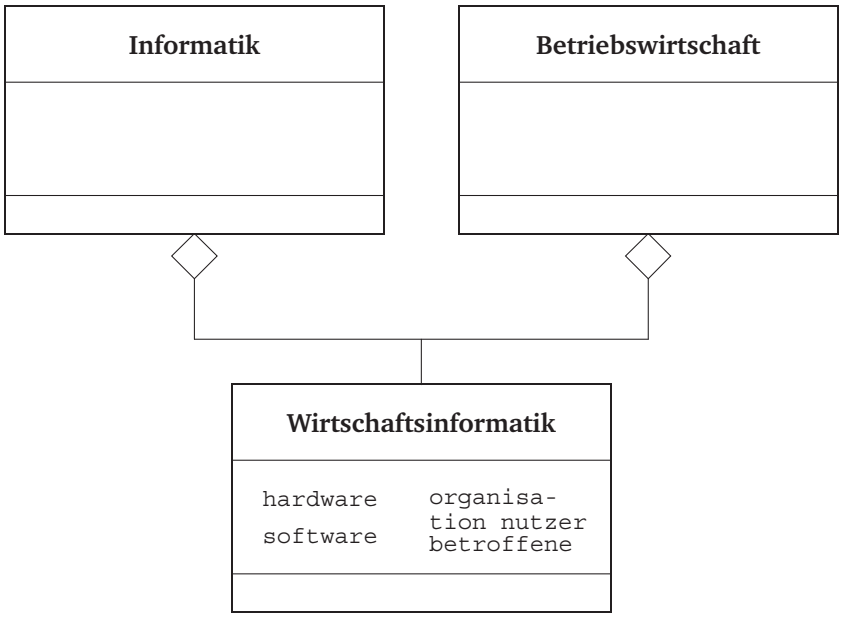
Klar ist auch, dass sich das ≫*weite Feld*≪ der WI laufend vergrößert. In welche Richtungen? Wie zeigt sich die WI zukünftig? Was sind Hoffnungen, Visionen und Pläne der WI-Promotoren? Was kennzeichnet die nächste Technikgeneration, die der WI prägende Impulse gibt? Wenn die bisher genutzten Generationen schlagwortartig bezeichnet werden können als:

1. *Generation:* COBOL-IBM-Mainframe,
2. *Generation:* UNIX-DEC-Minicomputer,
3. *Generation:* Windows-Intel-PC und
4. *Generation:* HTTP-Client/Server-Generation.

Wie lautet dann das Schlagwort der nächsten Generation? Vielleicht …? Halt, *Horst Meyer-Wachsmuth* würde jetzt davor warnen, dieses Vorwort zu einem Science-Fiction-Beitrag mutieren zu lassen. Deshalb hier zurück zu den unstrittigen WI-Elementen. Für diese lassen sich die folgenden plakativen Thesen hinreichend nachvollziehbar prognostizieren:

Hardware: **Alle Technik am Netz**
Der Visionär sieht jede Waschmaschine, jede Drehbank, jede Produktionsstation am Netz. Der Produktmanager telefoniert mit seiner „Waschmaschine“ um den aktuellen Zustand von ihr zu erfahren.

[6] ↪ `http://www.dfn.de/` (online 14-May-2008)
[7] SAVE ≡ Siemens-Informationstechnik Anwender Vereinigung e. V. (vormals SCOUT)
[8] Dieser Begriff wurde 1976 von *Frank DeRemer* und *Hans Kron* geprägt (↪ [DeKr76]).
[9] Z. B. UTM ≡ *Universal Transaction Monitor* (↪ [MW95])
[10] Komplexe Programme, geschrieben in FORTRAN (*Formula Translation*), prägten seine Dissertation.
[11] In der Funktion eines Hauptabteilungsleiters bei der Bertelsmann AG.
[12] In seiner gutachterlichen Tätigkeit ging es häufig um Entscheidungsfragen über IT-Systemarchitekuren.



Legende:
WI als Bestandteil der Informatik; notiert in UML

X	≡	Klasse X
X <>— Y	≡	Aggregation: Y ist Bestandteil von X
name	≡	Attribut, auch Slot oder Variable genannt

Abbildung 1: WI als Bestandteil der Informatik

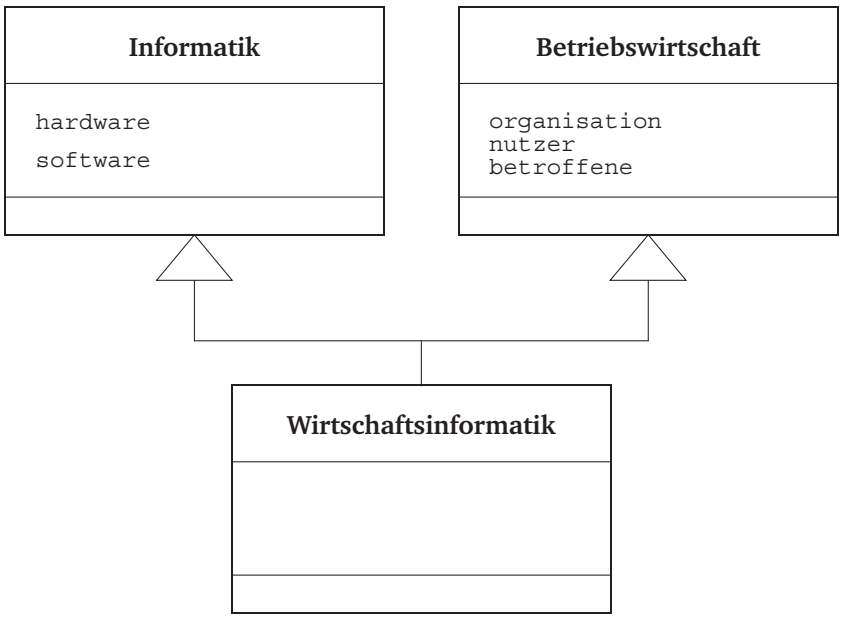
Software: **Selbstreplizierende Programme**
Der Visionär sieht autonome Programme im Netz, die sich durch „Kopulation“ mit anderen Programmen weiterentwickeln. Der Virus von heute wird das Arbeitspferd von morgen.

Organisation: **Virtuelle juristische Personen**
Der Visionär sieht virtuelle Organisationseinheiten, die eine juristische Person sind, beispielsweise ähnlich haften wie eine Aktiengesellschaft und eine steuerliche Ansässigkeit haben.

Nutzer & Betroffene: **Mehr Lebensqualität durch „Maschinisierung von Kopfarbeit“**
Der Visionär sieht, wie sich die „Maschinisierung von Kopfarbeit“ zur bedeutsamen Quelle für mehr Lebensqualität entwickelt. Der Verstärker des Wissens mutiert zum Verstärker der Lebensqualität für Viele.

Die Beiträge in diesem Band beleuchten Kern- und Expansionsflächen dieses „*weiten Feldes*“. Wir veröffentlichen sie hier, um das Wirken von *Horst Meyer-Wachsmuth* zu würdigen. Ohne ihn, unseren „MW“ — wie er stets von den WI'lern, seien es Kollegen oder Studierende, genannt wird — wäre das Department Informatik & Wirtschaftsinformatik (IWI) der Fakultät III der Leuphana Universität Lüneburg, mit seinen Instituten, nicht in dieser Form und mit diesem WI-Fundus entstanden. Im MW'schen Sinne mögen die Beiträge zu neuen Einsichten und Aktivitäten für Lehre, Forschung und Technologietransfer anregen.

Lüneburg im Mai 2008
Hinrich E. G. Bonin



Legende:
WI als spezielle Informatik; notiert in UML

X	≡	Klasse X
X ◁— Y	≡	Vererbungsbeziehung: Y hat die Eigenschaften von X
name	≡	Attribut, auch Slot oder Variable genannt

Abbildung 2: WI als spezielle Informatik

```
...
inetnum:        193.174.32.0 - 193.174.47.255
netname:        GRZLG-NET
descr:          Gemeinsames Rechenzentrum
descr:          der Fachhochschule Nordostniedersachsen
descr:          und der Universitaet Lueneburg
descr:          Studienort Lueneburg
country:        DE
...
irt:            IRT-DFN-CERT
address:        DFN-CERT Services GmbH
address:        Heidenkampsweg 41
address:        D-20097 Hamburg
address:        Germany
...
```

Legende:
Anfrage bei *RIPE NCC*, der unabhängigen, nicht gewinnorientierten Organisation zur Internet-Unterstützung
↪ `http://www.db.ripe.net/whois` (online 14-May-2008)

Abbildung 3: Eigenes Rechenzentrum — eine nachhaltige Quelle für die WI

Literaturverzeichnis

[DeKr76] Frank DeRemer / Hans Kron; PROGRAMMING-IN-THE LARGE VERSUS PROGRAMMING-IN-THE-SMALL, University of California, Santa Cruz
↪ `http://www.cs.umd.edu/class/spring2003/cmsc838p/General/pitl.pdf` (online 15-May-2008)

[Coy+92] Wolfgang Coy / Frieder Nake / Jörg-Martin Pflüger / Arno Rolf / Jürgen Seetzen / Dirk Siefkes / Reinhard Stransfeld (Hrsg.); Sichtweisen der Informatik, Braunschweig, 1992. [Eine gesellschaftspolitisch geprägte Analyse und Perspektive der Informatik.]

[Grass95] Grass, Günter: Ein weites Feld, Göttingen (Steidl Verlag), 1. Auflage 1995, ISBN 3-88243-366-3. [Eine Auseinandersetzung mit der deutschen Wiedervereinigung, verbunden mit einem Disput über das Werk von Theodor Fontane. Unterstellt wird eine Analogie der Konstellationen bei der Reichsgründung 1870/71, in der Art und Weise wie Theodor Fontane (1819-1898) sie erlebte, mit der Wiedervereinigung von BRD und DDR im Jahre 1989.]

[Kl03] Helmut Klemm; Ein großes Elend — Das Informationszeitalter kann sich nicht einigen über den Begriff „Information“, in: Informatik Spektrum, Band 26, Heft 4, August 2003, S. 267–273. [Primär eine Fortführung der Begriffsdiskussion, die von Günter Ropohl (Technikwissenschaftler, Universität Frankfurt) in der Zeitschrift „Ethik und Sozialwissenschaften“ angestoßen wurde.]

[Me+04] Peter Mertens / Reimut Bodendorf / Wolfgang König / Arnold Picot / Matthias Schumann /Thomas Hess; Grundzüge der Wirtschaftsinformatik, Berlin Heidelber u. a. (Springer) 1991, fünfte, neubearbeitete Auflage 1998, ISBN 3-540-63752-4; achte Auflage 2004, ISBN 3-540-40687-5. [Ein Lehrbuch, das integrierte Anwendungssysteme in den Mittelpunkt der Betrachtungen stellt.]

[MW95] Meyer-Wachsmuth, Horst; UTM in einer offenen Welt — TP-Monitore in offenen Client/Server-Architekturen,in: Proceedings der 10. SAVE-Tagung, Stuttgart 1995 (Hrsg. W. Zorn) [Hinweis:SAVE ≡ Siemens-Informationstechnik Anwender Vereinigung e. V.)]

[Na92] Frieder Nake; Informatik und Maschinisierung von Kopfarbeit, in: [Coy+92], S. 181-201. [Zum Schaffen von Frieder Nake ↪ [Rö03].]

[Rech03] Peter Rechenberg; Zum Informationsbegriff der Informationstheorie, in Informatik-Spektrum, Band 26, Heft 5, 2003, S. 317–326. [Eine fundierte Erläuterung des Begriffs „Information“.]

[Rech04] Peter Rechenberg; Stellungnahme des Verfassers zur Diskussion des Begriffs Information, in Informatik-Spektrum, Band 27, Heft 1, 2004, S. 92–93. [Eine Zusammenfassung von Diskussionsbeiträgen über den Begriff „Information“.]

[Rö03] Karl-Heinz Rödiger (Hrsg.); Algorithmik — Kunst — Semiotik, Hommage für Frieder Nake, Heidelberg (Synchron Wissenschaftsverlag) 2003, ISBN 3-935025-60. [Festschrift für Frieder Nake, einer der großen Pioniere der Computergraphik.]

Lieber Herr Prof. Meyer-Wachsmuth,

es ist mir eine besondere Freude, mich nicht nur als Vorsitzender des Fördervereins sondern auch als ehemaliger Student an Sie zu wenden. Sie zählen zu denjenigen meiner akademischen Lehrer, von denen ich wertvolle Impulse für meinen persönlichen Werdegang empfangen habe – nicht nur auf der Ebene der Informatik.

Ich spreche aus eigener Erfahrung, wenn ich sage, dass Sie immer ein offenes Ohr für die Belange der Studierenden hatten. Mit Ihrem herzlichen Auftreten und Ihrer charmanten Korrektheit waren Sie Ihren Studenten und Studentinnen nicht nur Respektsperson sondern auch Vorbild.

Als Verfechter einer praxisbezogenen Lehre haben Sie sich stets für einen gedeihlichen Austausch zwischen Wirtschaft und Wissenschaft eingesetzt, ohne dabei Ihre Ansprüche an eine gründliche, wissenschaftliche Vorgehensweise aufzugeben.

Durch Ihre Methodik haben Sie beispielhaft gezeigt, wie wichtig es ist, vielfältige Sichtweisen zuzulassen, ein Thema unter verschiedenen Aspekten zu beleuchten und dabei selbstkritisch auch die Rolle des Betrachters zu reflektieren.

Ich habe auch Ihren Pragmatismus sehr zu schätzen gelernt. Als ich Ihnen seinerzeit das Thema meiner Diplomarbeit in groben Zügen skizziert habe, erwiderten Sie schmunzelnd: „Ich weiß ja, was Sie meinen. Aber Sie sagen es nicht. Sagen Sie doch einfach, was Sie meinen.“ Daraus spricht eine Verbindlichkeit, die für mein weiteres Leben zu einer wichtigen Leitlinie geworden ist: In einer Gesellschaft, die zunehmend in Beliebigkeit zu versinken droht, ist es entscheidend, einen Standpunkt präzise und unmissverständlich darzustellen.

Lieber Herr Prof. Meyer-Wachsmuth, ich danke Ihnen und wünsche Ihnen von Herzen alles Gute für den nun vor Ihnen liegenden neuen Lebensabschnitt.

Ihr

Stefan Manzke
Dipl. Wirtschaftsinformatiker (FH)
1. Vorsitzender
des Fördervereins Netzwerk Wirtschaft
der Universität Lüneburg e.V.

Virtuelle Akten: Balance zwischen Ordnung und Unordnung

von
Hinrich E. G. Bonin[1]

Zusammenfassung

Im öffentlichen Sektor war, ist und bleibt das (geschäfts-)ordnungsgemäße Erstellen, Ablegen, Wiederauffinden, Bearbeiten und Zustellen von Schriftstücken, die zusammen eine Akte bilden, die „Königsaufgabe" für die Informations- und Kommunikationstechnik. Die „totale" Vernetzung ermöglicht beim Schriftgut der öffentlichen Verwaltung durchgreifende Innovationen. Die bewährte Automationsformel: *Akte(nberge) & Archiv & Registratoren ≡ Datenbank & Recherchesoftware & Systemverwalter* ist ergänzbar durch *virtuelle Akten* kombiniert mit *elektronischen Poststellen*.

Beteiligte übernehmen mit dem persönlichen Computer im Netzwerk Verantwortung für ihre Beiträge und ihre Post. Die Verknüpfung der Schriftstücke zu Akten und Archiven gewährleisten „*links*". Diese „Virtualisierung"[2] ist heute eine konkrete Option für die zweckmäßige Balance zwischen Ordnung und Unordnung beim komplexen System „elektronische Akten(berg)bearbeitung". Sie ermöglicht einen hohen Grad an Selbstorganisation am einzelnen Arbeitsplatz und sichert damit große Elastizität des Gesamtsystems.

Computing Reviews Classification: H.0; H.4.1; J.1

Schlagwörter: virtuelle Akte, Schriftgutverwaltung, Verwaltungsinformatik, Granulat, *Links*, *E-Government*

1 Virtuelle Akte: Bearbeitung ist und bleibt schwierig!

Land auf, Land ab wird unter dem Motto *E-Government*[3] ein quasi unermessliches Innovationspotential für den öffentlichen Sektor proklamiert. Eine durchgreifende Änderung der klassischen Vorgangsbearbeitung steht an. Die Basis dazu bildet die Ablösung der papierbezogenen Akte durch die *virtuelle Akte*. Mit einprägsamen Sprüchen wie: „*Die Daten sollen laufen, nicht die Bürger!*" oder gar „*Die Zukunft gehört dem papierlosen Büro!*" ist es für den Sachbearbeiter vor Ort nicht getan. Er lässt sich nur von einem einleuchtenden Beispiel — oder präziser formuliert — von einem überzeugenden Leitbild, das aufzeigt, wie das Bessere im Verwaltungsalltag konkret aussieht oder zumindest in naher Zukunft aussehen könnte, in seiner Arbeitsweise beeinflussen. Erst die Überzeugungskraft eines solchen Leitbildes bewirkt eine Umsetzung im konkreten Verwaltungsalltag.

[1] Prof. Dr. rer. publ. Dipl.-Ing. Dipl.-Wirtsch.-Ing. Hinrich E. G. Bonin lehrt Informatik an der Leuphana Universität Lüneburg, Institute of Computer Sciences (ICS), Email: bonin@uni.leuphana.de, Adresse: Campus Volgershall, Volgershall 1, D-21339 Lüneburg, Germany.

[2] In der Informatik wird der Begriff *Virtualisierung* für unterschiedliche Konzepte und Technologien verwendet. Oft steht dieser Begriff für Methoden, die es ermöglichen, Ressourcen von Computer(netzwerke)n aufzuteilen, indem eine zusätzliche Abstraktionsschicht zwischen Nutzer und Ressource eingeführt wird.

[3] Üblicherweise versteht man unter *E-Government* i. w. S. computerbasierte Geschäftsprozesse innerhalb und zwischen staatlichen Institutionen sowie zwischen diesen Institutionen und Bürgern (incl. Unternehmen).

„*Mit E-Government ist im Allgemeinen die Vorstellung verbunden, öffentliche Leistungen über das Internet bzw. auf der Basis des Internets bereitzustellen.*" (↪ [Schu08] S. 66)

„*Zur Zeit sind E-Government und E-Governance die Buzzwords in der Verwaltungsinformatik.*" (↪ [KlBo05], Vorwort)

Lfd	Leitbild	Hauptrolle der Bearbeiter
1	Fabrik	**Bearbeiter** = **„Teilhaber“** an allumfassender Anwendungssoftware (Geschäftsprozesse ⇔ „Behörden“ -Datenmodell)
2	Dienst-leister	**Bearbeiter** = **„Teilnehmer“** im Rahmen allgemeiner Dienste (Grundfunktionen bereitstellen im *Client/Server*-Modell)
3	Problem-löser	**Bearbeiter** = **„Täter“** und damit verantwortlich für seinen Anteil an der *virtuellen Akte* („*Links*“ bilden die Akte)

Legende:
Pointiert skizzierte Leitbilder, ohne die vielfältigen Mischformen zu nennen.

Tabelle 1: Leitbilderskizzen für die Bearbeitung virtueller Akten

1.1 Leitbilder für die Bearbeitung virtueller Akten

Wird stets hinreichende Informations- und Kommunikationstechnik als verfügbar unterstellt, dann zeichnen sich drei Leitbilder für die Bearbeitung von virtuellen Akten ab (↪ Tabelle 1):

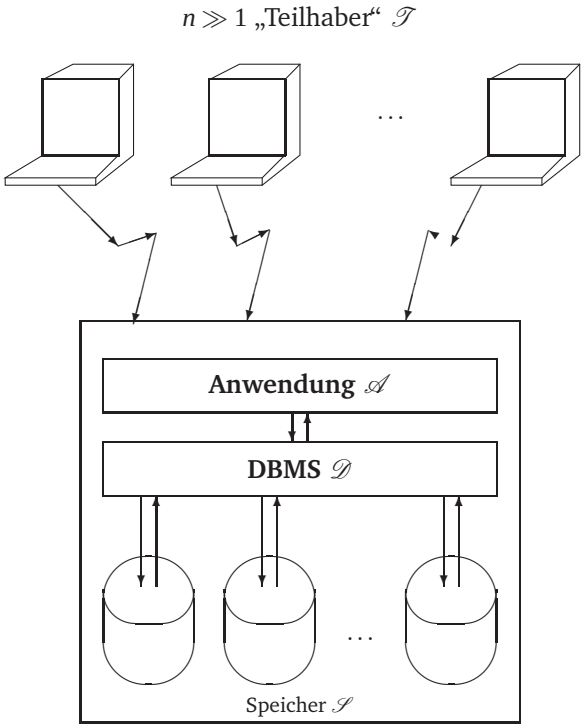
1. Leitbild **Fabrik**
   Die „Geschäftsprozesse“ prägen die elektronische Aktenbearbeitung. Abstrakt betrachtet entsprechen sie Produktionsprozessen in einer Fabrik. Die Akte ist daher primär das Resultat einer (weitgehend) **planbaren Bearbeitungsschrittfolge**. Die einzelnen Bearbeitungsschritte können — analog zur erfolgreichen Fabrikautomation — mit einer Standardsoftware im Verbund mit einem leistungsfähigen Datenbankmanagementsystem wirksam unterstützt werden (↪ Abbildung 1).

2. Leitbild **Dienstleistung**
   Die Akte entsteht fallweise und nur „wenige“ Schritte liegen vorab fest. Daher kann sich die standardmäßige Unterstützung für alle Bearbeiter nur auf einen allgemeingültigen Grundservice (z. B.: Aktenverfolgung & Zugriff) beziehen. Jeder einzelne Bearbeiter benötigt mehr oder weniger individuelle fallbezogene Anwendungssoftware. (↪ Abbildung 2).

3. Leitbild **Problemlösen**
   Aktenbearbeitung gleicht dem Lösen von Problemen. Die Computerunterstützung muss daher primär auf den Inhalt eines Schriftstückes zielen. Dazu benötigt der zuständige Schriftstückbearbeiter jedoch die Entscheidungsfreiheit über die anzuwendende Software, einschließlich der Datenhaltungsform.

   Die Akte ist nicht „Primärzweck“. Sie muss nicht permanent als bestimmbarer realer Speicherbereich (≈ Aktenordner) existieren, sondern sie braucht nur auf Anforderung zusammenstellbar zu sein. Die Bearbeiter haben nur den Zugriff auf die aktenrelevanten Schriftstücke auf ihrem Arbeitsplatzrechner zu gewährleisten (↪ Abbildung 3).

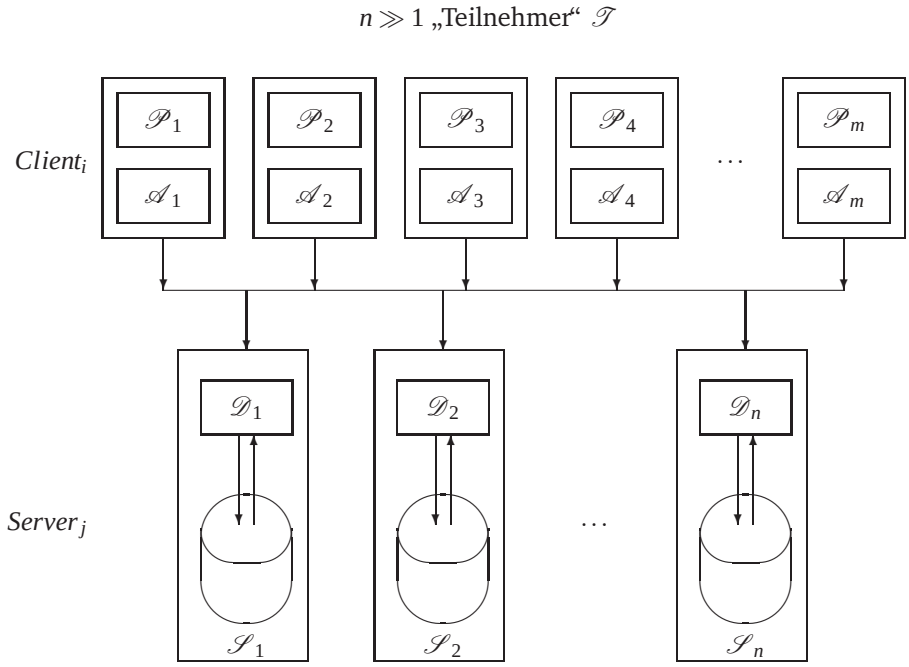
In jedem Fall ist erfreulicherweise zumindest der Hauspostdienst weitgehend entlastet. Jedoch entscheidend für eine nützliche elektronische Aktenbearbeitung ist das Einbeziehen der Eingangspost. Jedes eingehende Schriftstück ist dem zuständigen Bearbeiter elektronisch zur Verfügung zu stellen. Darüber hinaus ist eine Schnittstelle zu den papierbezogenen alten Akten zu schaffen. In einer



Legende:

$\text{Akte}_{elektronische}$	$\equiv$	Resultat einer geplanten Bearbeitungsschrittfolge
$n \gg 1$	$\equiv$	Die Beschäftigten einer Behörde oder einer großen Abteilung nutzen gemeinsam eine Datenbank.
$\longrightarrow\nearrow\searrow$	$\equiv$	Datenfernübertragung $\mathscr{T}$ = Terminal $\approx$ PC, PDA, Handy
Anwendung $\mathscr{A}$	$\equiv$	Anwendungssoftware mit Funktionen: Aktenzeichenvergabe, Ablage, Wiedervorlage, Zustandsangabe, Weiterleiten, Zustellen etc. Akte = $\sum$Dokumente $\approx$ $\sum$logische Dateien
DBMS $\mathscr{D}$	$\equiv$	Datenbankmanagementsoftware (z. B.: $\mathscr{D}$ = ADABAS, DB2 oder ORACEL)
$\mathscr{S}$	$\equiv$	Großer Speicher $\approx$ gesamtes Archiv / Registratur einer großen Organisationseinheit (Behörde)

Abbildung 1: Leitbild **Fabrik**: DBMS & Transaktionsunterstützung

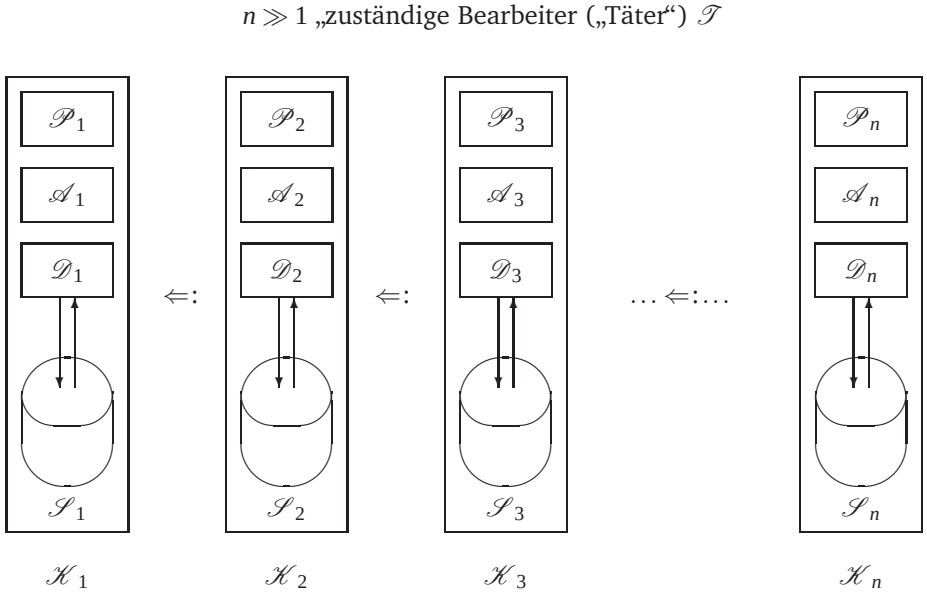


Legende:

↪ Abbildung 1

$\text{Akte}_{elektronische}$	≡	Service der computergestützten Schriftgutverwaltung
$\mathscr{P}_k$	≡	Individuell angepasste Präsentationssoftware („Desktop“)
$Client_i$	≡	Leistungsfähiger Arbeitsplatzrechner pro $\mathscr{T}$
$Server_j$	≡	Datenbankrechner in „Rechenzentrums“-Umgebung
$\mathscr{S}_j$	≡	Speicher ≈ (Teil-)Archiv
$\longrightarrow$	≡	Netzwerk verknüpft jeden $Client_i$ mit jeden $Server_j$

Abbildung 2: Leitbild **Dienstleistung**: Client/Server-Modell



Legende:

$\hookrightarrow$ Abbildung 1 und Abbildung 2

$\text{Akte}_{elektronische}$	$\equiv$	virtuelles Konstrukt zusammengesetzt aus Schriftstücken „autonomer“ Bearbeiter
$\mathscr{K}_i$	$\equiv$	Arbeitsplatzrechner als „autonomer“ Knoten im Netzwerk
$\mathscr{S}_j$	$\equiv$	Speicher $\approx \sum$Dokumente eines Bearbeiters
$\Leftarrow:$	$\equiv$	Allgemeines Hochgeschwindigkeitsnetz verknüpft jeden Knoten $\mathscr{K}_i$ mit jedem anderen Knoten

Abbildung 3: Leitbild **Problemlösen**: Arbeitsplatzcomputer als Knoten in einem allgemeinen Netzwerk

Gleichung pointiert zusammengefasst:

$$\begin{array}{|ll|}\hline \text{Akte}_{elektronisch} \equiv & \sum \text{Schriftstückbearbeitung}_{elektronisch} \\ & + \text{Scannen}(\text{Eingangspost}) \\ & + \text{Schnittstelle}(\text{Altaktenbestand}) \\ \hline \end{array}$$

Wenn die Computerunterstützung sich nicht auf eine simple Aktenverfolgung reduziert, muss sie die heutige Poststelle & Registratur mit umfassen. Die Schnittstelle (*interface*) zur Außenwelt einer Behörde ist daher wesentlicher Bestandteil einer elektronischen Aktenbearbeitung.

1.2 Gestaltungsrelevantes Granulat

Moderne Lösungen bilden auf dem Bildschirm möglichst realitätsnahe Symbole für Archiv, Aktenschrank, Aktenordner und Schriftstück (Dokument) ab. Jedes Schriftstück ist **eindeutig** einer Akte und jede Akte eindeutig **einem** Archiv zugeordnet. Die Bearbeitung einer Akte ist verbunden mit einem Navigieren in dieser Hierarchie (Baumstruktur) (↪ Abbildung 4). Betrifft ein Schriftstück mehr als eine Akte, dann sind entweder Kopien des Schriftstückes entsprechend abzulegen, oder es sind Verknüpfungshinweise (≡ *links*) zu den betroffenen Akten zu setzen.

Seit dem Welterfolg des Fotokopierers ist allgemein bekannt, dass eine Flut von Kopien besonders im Fall der Fortschreibung des Originals mannigfaltige Probleme aufwirft. „*Links*“ ermöglichen das Zusammenfassen von Schriftstücken zu Akten, nicht nur entlang einer Baumstruktur. Eine Akte läßt sich fallweise — zum Beispiel zweck- oder adressatenbezogen — zusammenstellen. Dabei können die „*Links*“ über die Grenzen eines Computers hinaus weisen, so dass mit Hilfe von schnellen Netzen auch örtlich verteilte Schriftstücke eine Akte abbilden können.

Wird diese Option aus einer allgemeineren Perspektive betrachtet, dann verdeutlicht sie, dass die Akte nicht mehr die gestaltungsrelevante Einheit (≡ Granulat) der Softwarearchitektur ist. Das Schriftstück übernimmt die Rolle dieses Granulats, da es selbst die „*Links*“ für seine Verknüpfung zu einer virtuellen Akte enthält.

1.3 Navigation im heterogenen Computernetz

Wenn die „*Links*“ wesentlicher Bestandteil des Granulats (Schriftstücks) sind und die höheren Aggregationsebenen (Akte, Archiv, Behörde) nur als Ergebnis eines Abarbeitungsprozesses (≡ virtuell) existieren, dann verschärft sich im Vergleich zum klassischen DBMS-Konzept das Gewährleistungsproblem. Wie wird sichergestellt, dass die virtuelle Akte bei Bedarf zu einer realen Akte ausgedruckt werden kann? Dazu sind die „*Links*“[4] in einer geeigneten Reihenfolge zu durchlaufen, um eine Sequenz der betroffenen Schriftstücke zu erhalten. Diese Linearisierung, die alle betroffenen Knoten eines Netzwerkes abzuarbeiten hat, setzt einen definierten Einstieg (Startknoten) voraus. Dieses Startschriftstück muss allen potentiellen Zugreifern bekannt sein. Es stellt eine Art „*homepage*“ für die virtuelle Akte dar.[5]

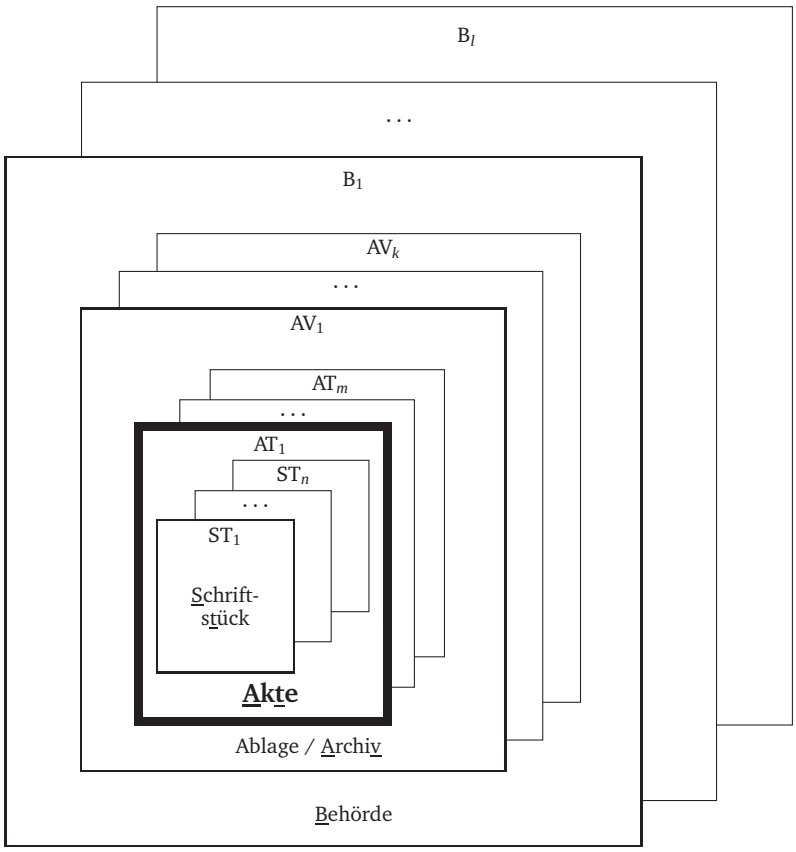
Der klassische Zugriffspfad[6] von der Behörde (*root*-Adresse) zum Schriftstück (Baumblatt) wird ergänzt durch den direkten Zugriff auf die „*Homepage* der Akte“. Diese enthält die „*Links*“ zu den Beschreibungen der virtuellen Hierarchieebenen.

Auf der Benutzungsoberfläche (*desktop*) ist dieses Navigieren kinderleicht. Ein Mausklick auf eine hervorgehobene Markierung im Schriftstück und schon wird das Folgedokument vom Computer

[4] Sie können durchaus einen komplexen nicht zyklusfreien Graphen bilden.

[5] Beginnend im Wintersemester 1994/95 wurden im Rahmen der Veranstaltung „Anwendungen in der öffentlichen Verwaltung“ (zunächst im Fachbereich Wirtschaft der Fachhochschule Nordostniedersachsen, dann im Institute of Computer Sciences der Leuphana Universität Lüneburg) in Form von vielen Projekten Mischformen der Leitbilder „Dienstleistung“ und „Problemlösen“ implementiert. Dabei wurde die Frage des Startschriftstückes eingehend untersucht.

[6] ↪ Abbildung 4



Legende:

Schriftstück	≡	„Zugriffsgranulat“ der Akte ≈ Dokument – häufig 1 bis n DIN A4 Seiten
Schriftgut(struktur)	≡	Mehrere Schriftstücke bilden eine Akte (Hauptakte mit gegebenenfalls mehreren Nebenakten). Mehrere Akten werden in einem Archiv (oder einer Ablage) gemeinsam verwaltet. Ein Archiv oder mehrere Archive bilden den „Informationsfundus“ (≈ Arbeitsnachweis & offene Aufgaben) einer Behörde.

Abbildung 4: Zugriff: Behörde → Archiv → Akte → Schriftstück

einer anderen Behörde gelesen. Die Oberfläche versteckt die zwingend erforderlichen Adressierungskonventionen. Ohne Standards und Netze mit problemadäquaten Zugriffszeiten, das heißt in der Regel mit großen Bandbreiten, ist die virtuelle Akte nicht realisierbar.

Mag sein, dass die virtuelle Akte aufgrund dieser Essentials in manchen ländlichen Regionen derzeit noch nicht realisierbar erscheint.[7] Die prinzipielle Machbarkeit kann jedoch nicht mehr bezweifelt werden, zumal ein gesicherter Zugriff im laufenden Betrieb permanent gewährleistet werden kann. Einerseits können berechtigte Zugreifer das Schriftstück stets erreichen und andererseits werden unberechtigte Zugreifer hinreichend zuverlässig abgewiesen.[8]

2 Balance zwischen Ordnung und Unordnung

Die virtuelle Akte im Leitbild „Problemlösen“ zielt auf eine begrenzte Gruppe von Beteiligten. Der Fokus liegt auf einer zweckmäßigen Selbstorganisation ihrer Bearbeitung. Aus der heutigen Behördenperspektive stehen jedoch Tonnen von Aktenbergen und Kubikmeter von belegtem Archivstauraum im Mittelpunkt. Solche riesigen Aktenmengen mit täglich hohem Zuwachs bedürfen der strikten Einhaltung einer vorgegebenen Ordnung. Ohne einen gültigen Aktenplan, nach dem sich jedermann richtet, sind solche „Aktenberge“ nicht zu managen.

Die moderne Physik lehrt, dass Ordnung, Chaos und Zufall eng miteinander verknüpft sind[9]. Vorgänge, die auf der Ebene von kleinsten Elementarteilchen („Quarks“) als Zufall anzusehen sind, stellen sich auf einer höheren Aggregationsstufe als Chaos dar. Dieses erweist sich auf einer noch höheren Stufe als wohl geordnet. Die virtuelle Akte, konstruiert aus individuell mehr oder weniger chaotisch markierten „*Links*“, befördert sicherlich die Tendenz zur Unordnung. Wenn das komplexe System „Aktenberg“ sich an Veränderungen der (Um)Welt anpassen soll, dann gilt auch hier die naturwissenschaftlich begründete Hypothese: *Komplexe adaptive Systeme funktionieren am besten in einem Zwischenbereich von Ordnung und Unordnung*.

Das Innovationspotential der elektronischen Aktenbearbeitung liegt damit in einer neuen Balance zwischen Ordnung und Unordnung. Die starren Baumstrukturen eines vorgegebenen Aktenplans können ergänzt werden durch ein eigenverantwortliches Setzen von (zusätzlichen) „*Links*“.

3 Fazit & Ausblick

Virtuelle Akten sind eine Option für eine bessere Balance zwischen Ordnung und Unordnung für das komplexe System „elektronische Akten(berg)bearbeitung“ . Sie ermöglichen einen höheren Grad an Selbstorganisation am Arbeitsplatz und sichern damit eine größere Elastizität des Gesamtsystems.

Für die Verwaltungspraxis ergeben sich daraus vielfältige Handlungsempfehlungen, zum Beispiel:

1. Schaffung eines behördenübergreifenden Breitbandnetzes („sichere Datenautobahnen“ für ALLE)
2. Betrieb von leistungsfähigen Netzknoten (insbesondere Qualifizierung der Beteiligten)
3. Elektronisches Verfügbarmachen der gesamten papierbezogenen Eingangspost (Scannen)
4. Regelung der elektronischen Eingangs- und Ausgangspost (Organisations- & Personenadressierung; Signaturen & Verschlüsselung)

[7] Zum Beispiel erfragt der Landkreis Lüneburg im Mai 2008 die tatsächliche Situation bezüglich der Breitband-Internetanschlüsse in allen Haushalten und Unternehmungen in seinem Gebiet. Ein Anlass dafür sind Klagen über eine Unterversorgung (↪ [LG08]).

[8] Basis dafür ist die aktuelle Web-Technologie, deren Ausgangspunkt auf der Standardisierung der Adressierung (*Universal Resource Identifiers* (URI)), des Netzwerkprotokolls (*Hypertext Transfer Protocol* (HTTP)) und der Markierungsnotation (*Hypertext Markup Language* (HTML)) fußt (↪ z. B. [Ber94])

[9] ↪z. B. [Gell94].

5. Rückgliederung von zentralen Registratur- und Archivfunktionen auf Knoten im Netz
6. Schaffung von Schnittstellen zum Altaktenbestand

Offensichtlich bewirken solche Maßnahmen die „Ubiquität[10] von Daten“[11] indem sie die Zugriffszeit beinahe auf Null reduzieren. Unstrittig können dann Kapazitäten bei Zentralstellen (Hausboten Poststelle, Registratur und Archiv) reduziert werden oder ganz entfallen.

Die eigentliche Innovation liegt jedoch in der verbesserten Adaptionsfähigkeit an dynamische Veränderungen der Umwelt. Über die quantitative Bedeutung und Wirkung dieses Pluspunktes kann heute nur spekuliert werden. Berechtigt vermutbar ist allerdings, dass mit dem Leitbild „Problemlösen“ die Computerunterstützung nicht nur bei der Akten**verwaltung** im Sinne des (geschäfts-)ordnungsgemäßen Erstellens, Ablegens, Wiederauffindens, Bearbeitens und Zustellens stehen bleibt, sondern auch die Qualität des Akteninhalts erreicht.

Literaturverzeichnis

[Ber94] Tim Berners-Lee / Robert Cailliau / Ari Luotonen / Henrik Frystyk Nielsen / Arthur Secret; The World-Wide Web, in: Communications of the ACM, Vol. 37, No. 8, August 1994, pp. 76–82.

[Gell94] Murray Gell-Mann; Das Quark und der Jaguar — Vom Einfachen zum Komplexen – die Suche nach einer neuen Erklärung der Welt, übersetzt von I. Leipold / Th. Schmidt, München Zürich (Piper) 1994.

[KlBo05] Sayeed Klewitz-Hommelsen / Hinrich Bonin (Hrsg.); Die Zeit nach dem E-Government, Münster (LIT Verlag) 2005, ISBN 3-8258-8188-1. [Der Band skizziert Trends in der Verwaltungsinformatik.]

[LG08] Landkreis Lüneburg; Umfrage zu Internetanschlüssen in Haushalten und Unternehmen im Landkreis Lüneburg, Mai 2008, ↪ `http://www.lueneburg.de/breitband` (online 12-May-2008)

[Rei94] Heinrich Reinermann; Informationstechnik für die Verwaltung der Zukunft — Innovative Verwaltungsentwicklung auf informationstechnischen Infrastrukturen — in: Carl Böhret / Hermann Hill / Helmut Klages (Hrsg.); Staat und Verwaltung im Dialog mit der Zukunft, Baden-Baden (Nomos Verlagsgesellschaft), 1994, S. 214–227.

[Schu08] Tino Schuppan; Gebietsreform im E-Government-Zeitalter — Potenziale und Erfahrung auf kommunaler Ebene, in: Verwaltung & Management — Zeitschrift für moderne Verwaltung, 14. Jahrgang, Heft 2, März/April 2008 (Nomos Verlagsgesellschaft), ISSN 0947-9856, S. 66–78.

[10] Ubiquität ≡ Allgegenwart (Gottes) — hier im Sinne überall verfügbar
[11] ↪ dazu [Rei94] S. 217.

Einsatz von Workflow-Managementsystemen in der Hochschulverwaltung[1]

Burkhardt Funk, Peter Niemeyer
Institut für elektronische Geschäftsprozesse (IEG)
Volgershall 1, 21339 Lüneburg
{funk|niemeyer}@uni.leuphana.de

1 Einleitung

Geschäftsprozessmanagement ist eines der zentralen Themen der Wirtschaftsinformatik. Unter dem Begriff der CSCW-Systeme (Computer Supported Cooperative Work) wurden Anwendungen entwickelt, die geeignet sind, Geschäftsprozesse vollständig oder teilweise zu automatisieren. Hierbei wird unterschieden zwischen Groupware- und Workflowlösungen. Groupware-Lösungen[2] unterstützen den Kommunikationsprozess innerhalb von Teams z.B. mit Hilfe von E-Mail-Anwendungen oder virtuellen Teamräumen. Im Vordergrund steht dabei der unstrukturierte Informationsaustausch, d.h., der genaue Ablauf der Kommunikation kann aufgrund der Aufgabenstellung vorab nicht im Detail festgelegt werden. Workflow-Lösungen[3] werden eingesetzt, wenn wiederkehrende Geschäftsprozesse – auch solche, die über Standortgrenzen hinaus verteilt sind – automatisiert werden sollen. Berücksichtigt man die Möglichkeiten der Interaktion von Nutzern mit Workflow Management Systemen (WfMS) im Rahmen von Geschäftsprozessen, bieten diese Systeme Unterstützung bei der Führung auf Distanz.

Während in Unternehmen das Prozessmanagement heute zu den wichtigsten Aufgaben der für Organisation und IT verantwortlichen Bereiche[4] zählt, werden wiederkehrende Verwaltungsprozesse in Hochschulen häufig nicht gezielt entwickelt, kontrolliert, optimiert und systemisch unterstützt[5] - und dies, obwohl sich die Wirtschafts- und Verwaltungsinformatik dieser Herausforderung bewusst ist und mittlerweile seit etwa 20 Jahren damit auseinandersetzt[6]. Hochschulen haben in der Vergangenheit auf den Hochschulbereich und die spezifischen Anforderungen zugeschnittene Anwendungssysteme (z. B. HIS) aber auch Standardsoftwaresysteme (z.B. SAP) eingeführt. Eine konsequente Ausrichtung an Prozessen ist bisher jedoch nicht gelungen.

Die vorliegende Arbeit vermittelt zunächst einen Überblick über CSCW-Lösungen (Kapitel 2) und skizziert anschließend den Nutzen und Funktionsumfang von WfMS. Anhand einer der heute zentralen Standards im Workflow Management (BPEL[7]) werden die Möglichkeiten der

[1] Einzelne Teile dieses Artikels beruhen auf einem Beitrag der Autoren in dem Buch „Leadership in Distributed Organisations“ (2007)
[2] Vergleiche dazu J.H.E. Andriessen (2002)
[3] Vergleiche dazu W. van der Aalst und K.M. van Hee (2002)
[4] Vergleiche dazu J. Becker et al. (2005)
[5] Vergleiche dazu S. Müller (2008)
[6] Vergleiche dazu T. Sommerlatte und E. Wedekind (1990)
[7] Business Process Execution Language, wird teilweise auch als BPEL4WS (BPEL for Web Services) bezeichnet. Ein weiterer wichtiger Standard, der sich in den letzten Jahren herausgebildet hat, ist die Business Process Modeling Notation, kurz BPMN (vergleiche S.A. White, 2004), die im wesentlichen eine einheitliche graphische Modellierung unterstützt

plattformunabhängigen Geschäftsprozessautomation erläutert. Kapitel 4 erörtert die verschiedenen Interaktionstypen zwischen Benutzern und Workflows und leitet daraus die entsprechenden Anforderungen an WfMS ab. Das letzte Kapitel widmet sich der Anwendung der vorgestellten Technologien und Methoden vor dem Hintergrund des Prozessmanagements in Hochschulen.

2 Computer Supported Cooperative Work

In dem Forschungsfeld Computer Supported Cooperative Work (CSCW) erforschen Wissenschaftler aus verschiedenen Disziplinen[8] seit den späten achtziger Jahren die Rolle der Informations- und Kommunikationstechnologie im Rahmen kooperativer Arbeit. Zu den zentralen Themen gehören

- das Erfassen und Verstehen von Zusammenarbeit, Kommunikation und Koordination in Organisationen
- die Entwicklung von Methoden und Werkzeugen für die Unterstützung arbeitsteiliger Prozesse
- die Bewertung und Evaluation kooperations- und kommunikationsunterstützender Werkzeuge

Auf den folgenden Seiten untersuchen wir zunächst die funktionalen Ziele, zu deren Erreichung CSCW-Systeme beitragen können. Aufbauend auf einer Klassifikation der Arbeitsabläufe in Unternehmen geben wir dann einen Überblick über die heute erhältlichen Softwarelösungen im CSCW-Umfeld.

2.1 Einsatzbereiche von CSCW-Systemen

Nach Teufel et.al.[9] klassifiziert man die Prozesse der Gruppenarbeit nach der Intensität der Zusammenarbeit in der Gruppe, die über ihren Beitrag zu Kommunikation, Kooperation und Koordination gemessen wird. Das daraus resultierende 3K-Modell führt zu einer ersten groben Klassifizierung von Gruppenarbeit unterstützenden Anwendungen:

- *Kommunikationsunterstützung*: Hier geht es primär um die Überwindung zeitlicher und räumlicher Distanzen. Zur Überwindung zeitlicher Distanz dienen dabei etwa asynchrone Kommunikationsdienste wie elektronische Post, während die Überwindung räumlicher Distanz vornehmlich von synchronen Kommunikationsdiensten wie Konferenzsystemen unterstützt wird.
- *Kooperationsunterstützung*: Hierunter fallen z.B. Dienste zum gemeinsamen Zugriff auf Dokumente und zur Terminüberwachung.
- *Koordinationsunterstützung*: Solche Systeme unterstützen die Koordination von Tätigkeiten und Ressourcen. Sie sind in der Lage, Arbeitsabläufe abzubilden und die ein-

[8] Neben der Wirtschaftsinformatik sind hier die Disziplinen Soziologie, Psychologie, Arbeits- und Organisationswissenschaften, Wirtschaftswissenschaften, Anthropologie und Ethnologie zu nennen. Zur Begriffsklärung vergleiche (Greif 1988) und (Hasenkamp 1994)
[9] Vergleiche dazu TEUFEL (1995)

zelnen Aufgaben in der richtigen Reihenfolge an die zuständigen Bearbeiter zu delegieren.

Nachdem wir einen ersten Blick auf die relevanten Prozesse der Gruppenarbeit geworfen haben, wollen wir uns nun den Arbeitsabläufen zuwenden, die das eigentliche Ziel der Gruppenarbeit sind, den Geschäftsprozessen. Nach Gadatsch[10] lassen sich die Arbeitsabläufe in einer Organisation nach folgenden Kriterien klassifizieren (vgl. Tabelle 1)

- Strukturierungsgrad
- Repetitive Elemente
- Freiheitsgrade in der Ablaufsteuerung
- Automatisierungsgrad

Tabelle 1: Klassifizierung von Arbeitsabläufen nach Gadatsch

	ad hoc	fallbezogen	allgemein
Strukturierungsgrad	nicht strukturierbar	nicht vollständig strukturierbar	vollständig strukturierbar
Ausführungshäufigkeit	einmalig	selten	häufig
Freiheitsgrade	sehr hohe Freiheitsgrade für den Bearbeiter	es gibt Freiheitsgrade für den Bearbeiter	keine Freiheitsgrade für den Bearbeiter
Automatisierungsgrad	Arbeitsschritte nicht im Voraus definierbar	Arbeitsschritte teilweise im Voraus definierbar	Arbeitsschritte im Voraus definierbar

Unter *Ad Hoc Abläufen* werden individuelle Vorgänge verstanden, die nicht automatisiert werden können. Typische Beispiele sind kreative Tätigkeiten, wie z.B. das Erarbeiten einer Werbekampagne. Während die Umsetzung einer Werbekampagne häufig aus ähnlichen Aktivitäten besteht, gilt dies für das Erarbeiten der Kampagne vermutlich nicht.

Dem gegenüber stehen so genannte *allgemeine Arbeitsabläufe*, welche sich durch vollständige Strukturierbarkeit und hohe Automatisierbarkeit auszeichnen. Als Beispiel kann der Urlaubsantrag in einem Unternehmen dienen: der Antragsteller gibt die gewünschten Urlaubstage an. Nach einer Plausibilitätsprüfung (sind noch genug Urlaubstage verfügbar? Sind mögliche Vertreter zur selben Zeit im Urlaub?) wird der Antrag entweder abgelehnt, oder zur Freigabe an den Personalverantwortlichen weitergereicht, der dann zustimmt oder die Ablehnung begründet. Im Falle der Zustimmung werden die Daten noch im Personalverwaltungssystem verbucht, schließlich wird der Antragsteller über das Ergebnis seines Antrages informiert.

Zwischen diesen beiden Gruppen liegen die *fallbezogenen Arbeitsabläufe*, die sich vor allem durch die hohen Freiheitsgrade auszeichnen, die einzelnen Bearbeitern einzuräumen sind. Als Beispiel kann die Einstellung von Mitarbeitern dienen: nach Eingang der Bewerbung erfolgt eine erste formale Prüfung, die zur Absage oder einem ersten Telefon-Interview führen kann. An dieser Stelle wäre auch zu prüfen, ob es eine passende unbesetzte Stelle gibt. Bei erfolgreichem Verlauf des Telefoninterviews könnte dann ein persönliches Gespräch unter Beteili-

[10] Vergleiche dazu GADATSCH (2005)

gung der betroffenen Abteilung folgen. Während sich Teile dieses Ablaufes gut automatisieren lassen (Korrespondenzen, Terminplanung etc.), können andere Arbeitsschritte aufgrund großer Entscheidungsspielräume nur schwer maschinell unterstützt werden (z.B. die Prüfung, ob eine Bewerbung zu einer offenen Stelle passt.)

2.2 IT-Lösungen

Abgestimmt auf die beschriebenen Klassen von Arbeitsabläufen werden heute die folgenden Softwarelösungen angeboten, die funktional teilweise starke Überlappungen aufweisen (vgl. Abbildung 1):

- *Kommunikationssysteme* dienen primär zur Unterstützung unstrukturierter Arbeitsabläufe. So unterstützen E-Mail-Systeme den asynchronen Austausch, Chat-Lösungen den synchronen Austausch von Nachrichten.
- *Groupware Systeme* dienen in erster Linie zur Unterstützung von wenig strukturierten Arbeitsabläufen mit geringer Wiederholungsrate, also insbesondere individuellen Abläufen wie kreativen Prozessen. Sie sind in der Lage den beteiligten Personen große Freiheitsgrade einzuräumen und enthalten in der Regel die Funktionalität eines Kommunikationssystems. Darüber hinaus unterstützen sie z.B. die Abstimmungen von Gruppenterminkalender, Zugriff auf gemeinsame Dokumente und sonstige Information.
- *Workflow Management Systeme* unterstützen vor allem stark strukturierbare Prozesse, die häufig auszuführen sind und nur fest definierte Interaktionen mit Anwendern enthalten.
- *Sonstige Lösungen*: Neben den beschriebenen Lösungen gibt es noch eine Reihe weiterer Systeme, wie etwa Projektmanagement Lösungen (für individuelle Abläufe) oder Speziallösungen für konkrete, teilweise branchenspezifische Prozesse, die in der Regel einen hohen Strukturierungsgrad aufweisen (z.B. ERP-Systeme).

Wie in Abbildung 1 dargestellt kommen bei der Unterstützung von semistrukturierten und azyklischen Arbeitsabläufen sowohl traditionelle Groupware-Funktionalitäten, als auch Workflow Management Systeme zum Einsatz.[11]

[11] Vor diesem Hintergrund ist zu beobachten, dass immer mehr Groupware Systeme um traditionelle WfMS-Funktionalität angereichert werden, und umgekehrt.

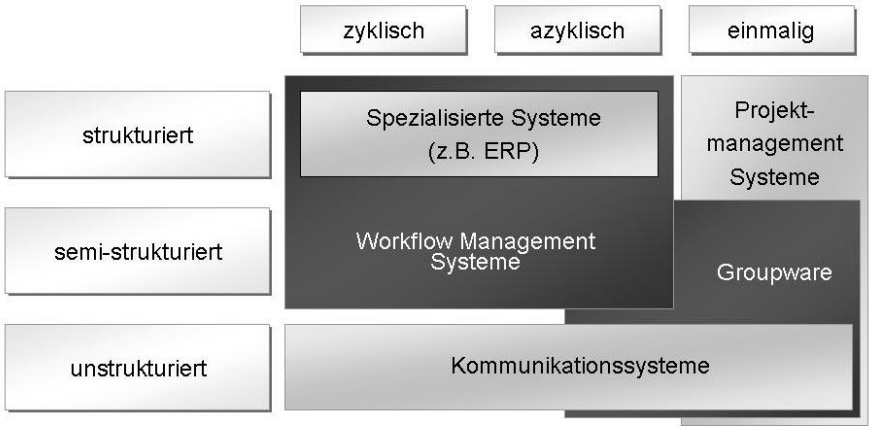


Abbildung 1: Überblick CSCW-Lösungen

3 Workflow Management Systeme

Der WfMC[12] folgend differenzieren wir zwischen der betriebswirtschaftlichen Beschreibung eines Arbeitsablaufes, dem *Geschäftsprozess*, und der Automatisierung des Arbeitsablaufes, dem *Workflow*. [13]

Genauer definiert die WfMC einen *Geschäftsprozess* als Menge von abhängigen Aktivitäten, die durch ein betriebswirtschaftliches Ziel sowie eine Organisationsstruktur bestimmt werden. Ein Geschäftsprozess kann sowohl automatisierbare als auch manuelle Arbeitsschritte enthalten. Als *Workflow* bezeichnen wir hingegen einen teilweise oder vollständig automatisierten Geschäftsprozess, in dem Dokumente, Informationen oder Aufgaben zwischen Teilnehmern entsprechend einer Menge von Ausführungsregeln übertragen werden. Unter einem Workflow Management System (WfMS) verstehen wir, der WfMC folgend, ein Anwendungssystem, das Definition, Ausführung und Verwaltung von Workflows unterstützt.

Im Folgenden gehen wir zunächst auf die Funktionen und den Nutzen von Workflowmanagement Systemen ein und geben dann einen kurzen Einblick in die Business Process Execution Language BPEL, die heute zu den wesentlichen Standards im Workflow Management zählt.

3.1 Funktionen von Workflow Management Systemen

Ein Workflow Management System unterstützt die Modellierung, die Ausführung und die Verwaltung von Workflows. Im Rahmen der Modellierung müssen die Arbeitsabläufe, die

[12] Die Workflow Management Coalition (WfMC) ist ein 1993 gegründeter Verbund von Herstellern, Anwendern und Wissenschaftlern, die sich das Ziel gesteckt haben, ein einheitliches Referenzmodell für Workflow Management Systeme zu etablieren.
[13] Vgl. WFMC (1999)

Organisationsstruktur und die Systemlandschaft der angebundenen Applikationen definiert werden. Hierzu stehen meist grafische Editoren zur Verfügung. Bei der Definition eines Arbeitsablaufes wird der Gesamtablauf als Abfolge einzelner Aktivitäten abgebildet, die konkreten Benutzern, Rollen oder Anwendungen zugeordnet werden können. Die einzelnen Aktivitäten werden dabei in Abhängigkeit von Ereignissen aufeinanderfolgend oder parallel angeordnet, beliebige Teilsequenzen können iteriert werden. Die Modellierung der Organisationsstruktur ist erforderlich, da die Zuordnung einzelner Aktivitäten zu Bearbeitern häufig dynamisch erfolgt. In einem Workflow Urlaubsantrag könnte beispielsweise die Freigabe durch den direkten Vorgesetzten erforderlich sein. In einer konkreten Ausführung des Workflows kann der Vorgesetzte über die Organisationsstruktur aus dem Antragsteller abgeleitet werden. Neben den Workflow Aktivitäten, die einem menschlichen Bearbeiter zugeordnet sind, gibt es auch solche, die von einem Anwendungssystem abgearbeitet werden. Die inhaltliche Prüfung eines Urlaubsantrages könnte von einer Anwendung durchgeführt werden, die den zentralen Firmenkalender verwaltet. Damit das WfMS solche Anwendungen einbinden kann, muss die entsprechende Systemlandschaft modelliert werden. Zu allen einbindbaren Anwendungen müssen die Schnittstellen, eine logische Adresse und auch die physische Adresse hinterlegt sein.

In der Praxis gibt es eine Reihe von Abhängigkeiten zwischen dem Ablaufmodell auf der einen und Organisationsmodell und Systemlandschaft auf der anderen Seite. Daher stellt WfMS eine Simulationsfunktion zur Verfügung, mit der geprüft werden kann, ob das Zusammenspiel der genannten Modelle konsistent ist.

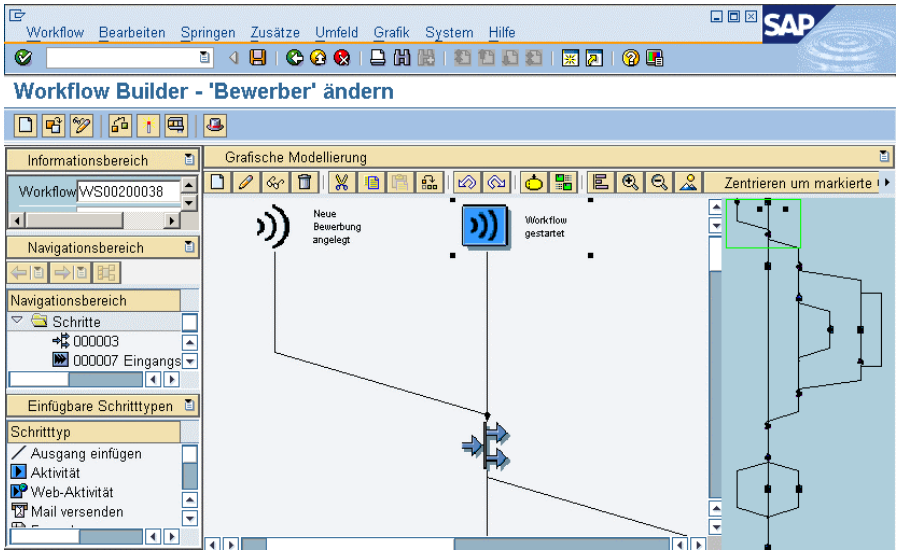


Abbildung 2: Definition eines Workflows im System SAP Netweaver (© SAP AG)

Bei der Ausführung eines Workflows sind die oben beschriebenen Modelle (Ablaufmodell, Organisationsmodell und Systemlandschaft) auszuwerten:

- Im Rahmen einer Rollen-Auflösung müssen die geeigneten Mitarbeiter aus dem Organisationsmodell abgeleitet werden
- Bei human tasks sind die Bearbeiter zu informieren (z. B. über E-Mail)
- Bei maschinellen Aktivitäten sind entsprechende Datenbestände an die verantwortliche Applikation zu senden; das Ergebnis der Bearbeitung ist für etwaige Folgeaktivitäten vorzuhalten

Um die Nachvollziehbarkeit der Ausführungen zu gewährleisten, wird jede einzelne Aktivität protokolliert: welche Applikation ist wann mit welchen Daten aufgerufen worden? Welches Ergebnis hat die Applikation an das WfMS zurückgemeldet? Welche Rolle wurde wann wie aufgelöst? Welche Bearbeitungsfehler sind aufgetreten?

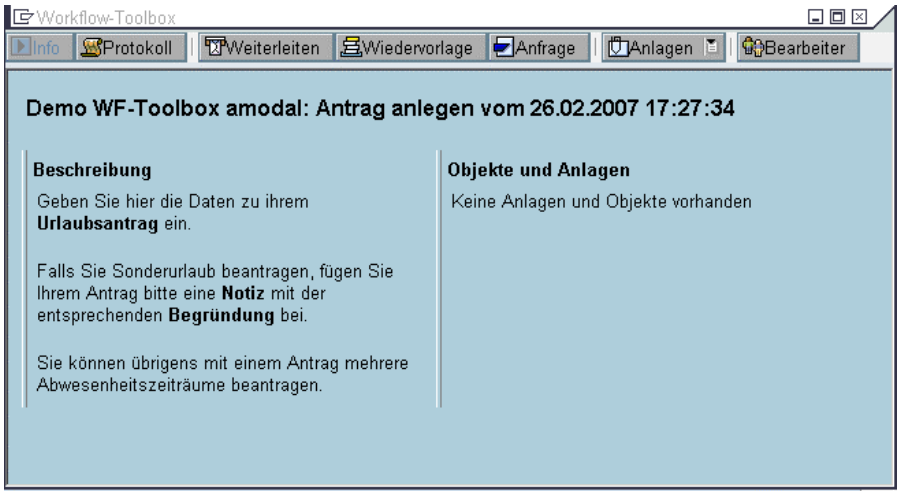


Abbildung 3: Beispiel einer Worklflow-Aktivität im System SAP Netweaver (© SAP AG)

Im Rahmen der Administration von Workflows auf dem WfMS sind unterschiedliche Auswertungen für das Management, die beteiligten Bearbeiter und die WfMS Administration erforderlich. Das WfMS stellt hierzu entsprechende Reportingfunktionalität bereit (vgl. Tabelle 2).

Tabelle 2: Auswertungsanforderungen an ein WfMS

Adressat	Fragestellung
Management	• Welche offenen Aktivitäten gibt es? • Wie hoch sind die durchschnittlichen Bearbeitungszeiten? • Wie häufig wird ein Workflow verwendet? • Wie hoch sind die Fehlerraten?
Bearbeiter	• Welche offenen Aktivitäten habe ich? • Welche Aktivitäten habe ich gestern bearbeitet? • Welche Fehler sind in der letzten Woche aufgetreten? • Welche Urlaubsanträge habe ich im letzten Monat abgelehnt?
WfMS Administration	• Wie sind die Workflow Aktivitäten auf die Bearbeiter bzw. Anwendungssyteme verteilt? • Welche Verarbeitungsfehler sind aufgetreten? • Welche Nachrichten wurden wann an welche Fremdsysteme geschickt?

3.2 Nutzen des Workflowmanagements

Welchen Nutzen hat nun der Einsatz eines WfMS für ein Unternehmen? Aufgrund der funktionsübergreifenden Automatisierung des Aufgaben- und Informationsflusses ist mit einer Stei-

gerung von Effizienz und Produktivität der unterstützten Geschäftsprozesse zu rechnen. Die lückenlose Dokumentation der einzelnen Arbeitsschritte und Entscheidungen kann zur Qualitätssicherung beitragen und erhöht die Auskunftsfähigkeit gegenüber Kunden. Da die Automatisierung eines Geschäftsprozesses durch ein WfMS grundsätzlich die Modellierung des Prozesses voraussetzt, verfügt ein Unternehmen beim Einsatz von WfMS über ein explizites Verständnis der betroffenen Prozesse. Schließlich kann der richtige Einsatz von Workflows dazu beitragen, die informationelle Absicherung der Führungskräfte zu erhöhen, in dem z. B. bei Freigabeschritten jeweils alle entscheidungsrelevanten Kontextinformationen zur Verfügung gestellt werden.

Mitarbeiter, die in der Rolle von Bearbeitern an der Durchführung von Workflows mitwirken, profitieren ebenfalls von dem Einsatz des WfMS. So vereinfachen sich einzelne Arbeitsabläufe durch automatische Anbindung von Anwendungssystemen oder die Möglichkeit, einzelne Aktivitäten „per Knopfdruck“ zu delegieren. Auch die Möglichkeit, aktuelle ToDo-Listen aus dem WfMS generieren zu lassen, wird in vielen Fällen als Vereinfachung wahrgenommen.

3.3 Die Business Process Execution Language (BPEL)

Da heute in vielen Unternehmen eine ganze Reihe unterschiedlicher WfMS[14] im Einsatz sind, wird es immer wichtiger, dass WfMS auch Arbeitsabläufe verarbeiten können, die auf einem anderen WfMS modelliert worden sind. Darüber hinaus kann es erforderlich sein, dass unterschiedliche WfMS gemeinsam den Ablauf eines Workflows abbilden.

Der Schlüssel zur Interoperabilität liegt in einer standardisierten Modellierungssprache, die auf beliebigen Systemen erfasst und ausgeführt werden kann. Einen solchen Ansatz verfolgt die Business Process Execution Language (BPEL), die im Rahmen der OASIS[15] von Firmen wie IBM, SAP, Microsoft, Oracle und zahlreichen anderen Softwareherstellern entwickelt wird. Entstanden im Jahr 2002 aus einer Reihe von Vorgängersprachen wie XLANG oder WSFL ist BPEL heute in der Version 2.0 einer der wichtigsten Standards für die ausführungsnahe Modellierung von Geschäftsprozessen und damit Grundlage der meisten WfMS.

Ein BPEL-Dokument[16] legt fest, welche Web Services in einem Geschäftsprozess in welcher Reihenfolge aufgerufen werden sollen, ob diese Aufrufe synchron oder asynchron erfolgen und welche Vorbedingungen für ihren Aufruf erfüllt sein müssen. BPEL ermöglicht somit die Orchestrierung von Web Services[17] im Rahmen von Geschäftsprozessen. Die sog. Executable Business Processes[18] beschreiben den Ablauf und die involvierten Partner (und ihre Schnittstellen) ausreichend detailliert, um mit Hilfe einer Workflow Engine[19] einen Prozess auf dieser Basis ausführen zu lassen.

[14] Dies können sowohl Stand-Alone-Systeme als auch beispielsweise in ERP-Systeme integrierte Workflow-Umgebungen sein
[15] http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsbpel
[16] Vergleiche Nicolescu et al. (2007)
[17] Mit Hilfe sog. Abstract Business Processes können Workflows in einer abstrahierten Form beschrieben werden. Hierbei wird insbes. der Kommunikationsablauf zwischen Partnern beschrieben (dies kann auch im Sinne einer Choreographie im Gegensatz zur Orchestrierung erfolgen)
[18] http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html
[19] Diesen Begriff wird synonym zum WfMS verwendet

Um den Status von Geschäftsprozessen zu speichern, bietet BPEL die Deklaration von Variablen an. Für die Spezifikation der Ablauflogik stehen mit Schleifen, Bedingungen, Sequenzen und Ausnahmen (Exceptions) die grundlegenden „strukturierten Aktivitäten“ zur Verfügung, die aus Programmiersprachen bekannt sind. BPEL unterstützt damit die „Programmierung im Großen“.

4 Human Interaction in Workflows

Geschäftsprozesse können meist nicht vollständig ohne menschliche Interaktion automatisiert werden. Stattdessen erfordern einzelne Aktivitäten innerhalb von Geschäftsprozessen die Interaktion mit Personen oder Gruppen von Personen. Beispiele hierfür sind einfache Freigaben (z.B. eines Artikels in einem Redaktionsprozess) oder umfangreiche Aufgaben wie die Vorbereitung und Durchführung eines Personalinterviews, die im Wesentlichen außerhalb eines WfMS erledigt werden müssen. Die Möglichkeit zur Interaktion von Personen mit Geschäftsprozessen setzt entsprechende Komponenten innerhalb und Schnittstellen zu WfMS voraus, die als Human Workflow Services bezeichnet werden. Im Folgenden werden zunächst die verschiedenen Interaktionstypen zwischen Personen und Workflows beschrieben und anschließend die Anforderungen an Human Workflow Services abgeleitet. Darauf aufbauend werden die funktionalen Komponenten eines Human Workflow Services beschrieben.

4.1 Interaktionstypen

Aufgaben, sog. Tasks[20], sind Grundlage der Kommunikation zwischen Personen und Workflows. Die Bearbeitung von Aufgaben in einer Workflowinstanz kann dabei zeitlich parallel oder sequentiell erfolgen. Aufgaben verfügen in WfMS über einen Lebenszyklus im Sinne eines Aufgabenstatus. Aufgaben erhalten zum Zeitpunkt ihrer Generierung durch das WfMS den Status „offen“ und können durch weitere Eigenschaften spezifiziert werden. Die wichtigsten Eigenschaften davon sind:

- Verantwortlichkeit: welche Person, Rolle oder Gruppe von Personen ist für die Bearbeitung der Aufgabe verantwortlich? Meist werden Aufgaben in Workflows Gruppen zugewiesen, die eine bestimmte Rolle im System aufweisen. Eine einzelne Person dieser Gruppe kann dann die Aufgabe für sich beanspruchen (Aufgabenstatus: beansprucht) und bearbeiten.
- Beschreibung: die Aufgabe wird im Klartext oder durch Referenzen auf Aufgabenbeschreibungen beispielsweise in unternehmensinternen Handbüchern beschrieben, häufig ist bereits der Aufgabentitel (z.B. „Einkaufsfreigabe durch Teamleiter“) ausreichend aussagekräftig.
- Priorität und Deadline: Welche Bedeutung (Priorität) hat die Aufgabe im Vergleich zu anderen Aufgaben, die an die Gruppe vergeben wurden, und bis wann muss die Aufgabe spätestens erledigt werden? Bei Überschreiten der Deadline erhält die Aufgabe den Status „fehlgeschlagen“.
- Kontext (Output aus Sicht des WfMS): Voraussetzung für die Erledigung vieler Aufgaben ist die Kenntnis bestimmter Informationen, so erfordert die Einkaufsfreigabe durch den Teamleiter beispielsweise die Kenntnis, welche Materialien oder Produkte,

[20] Diese werden von einigen Autoren und Softwareherstellern auch als work items bezeichnet.

zu welchem Preis, in welcher Menge und von welchem Lieferanten gekauft werden sollen. Diese Informationen müssen entweder direkt mit der Aufgabenübernahme durch die Person oder auf Nachfrage z.B. über ein entsprechendes Portal bereitgestellt werden.

Den Status „abgeschlossen“ erhalten Aufgaben, nachdem sie erfolgreich bearbeitet wurden. Das damit verbundene Ergebnis kann verschiedene Ausprägungen haben: im einfachsten Fall wird die Aufgabe (z.B. „Kunde über Lieferverzögerung informieren“) ausschließlich als erledigt gekennzeichnet. Andere Aufgaben erfordern Freigaben (Ablehnungen) oder die Eingabe umfangreicher Daten in Form von automatisch erzeugten Formularen.

Welche über die zuvor genannten hinausgehenden Interaktionstypen (Freigabe, Aufgabenübernahme und -bearbeitung) gibt es? Nicht immer ist es dem WfMS möglich zu entscheiden, welcher Person oder Gruppe eine Aufgabe zugeordnet werden soll. In diesen Fällen wird die Aufgabe beispielsweise einem Teamleiter zugewiesen, der dann auf Grundlage außerhalb des WfMS liegender Fakten (z.B. Auslastung oder Kompetenz einzelner Mitarbeiter) die Aufgabe an eine Person oder Gruppe delegiert. D.h. die Aufgabe erhält einen neuen Verantwortlichen, und das WfMS sorgt automatisch dafür, dass die Aufgabe in der Aufgabenliste (Worklist) der angesprochenen Gruppe erscheint. Wird eine Aufgabe nicht innerhalb eines vorgegebenen Zeitraums bearbeitet, wird diese als „fehlgeschlagen“ gekennzeichnet, und es kommt zur Information der Leitungsebene (Eskalation). Welche Personen im Falle der Eskalation informiert werden, kann bei der Aufgabenvergabe fallspezifisch festgelegt oder aber anhand eines in einem Verzeichnisdienst hinterlegten Organigramms entschieden werden. Die Eskalation kann, falls sie durch die nächste Leitungsebene unberücksichtigt bleibt, über mehrere Stufen (bis zur Unternehmensführung) propagiert werden.

Eine weitere Interaktionsmöglichkeit mit WfMS besteht in der Initiierung von Workflows und Sub-Workflows. Workflows können automatisch (z.B. Beauftragung eines Mitarbeiters im Debitorenmanagement nach einer Rücklastschrift) durch das WfMS oder manuell (z.B. Antrag eines Kunden auf Kontoeröffnung am Bankschalter) durch einen Mitarbeiter initiiert werden. Dabei kann es sinnvoll sein, eine umfangreiche Aufgabe, die im Rahmen eines Workflows bearbeitet werden soll, als Sub-Workflow zu initiieren und den erfolgreichen Abschluss als Kriterium für den weiteren Ablauf des Haupt-Workflows zu definieren. Die vorzeitige Terminierung von Workflows kann analog der Initiierung manuell erfolgen. So könnte beispielsweise der Mitarbeiter einer Personalabteilung die Instanz eines Bewerbungsworkflows nach einem negativen Interview beenden und weitere bereits angesetzte Interviewtermine mit dem Bewerber absagen. Abschließend soll an dieser Stelle die Fehlerbehandlung[21] als Interaktionstyp erwähnt werden.

Zusammenfassend können folgende Interaktionstypen[22] benannt werden: Aufgabenübernahme (und -bearbeitung), Delegation, Eskalation, Initiierung und Terminierung von Sub-Workflows sowie Fehlerbehandlung.

[21] Fehlerbehandlung tritt in Workflows in zwei Ausprägungen auf. Die erste wird als sog. Fault Handling bezeichnet, hierbei wird definiert, wie ein Workflow im Falle von Ausnahmen (z.B. die Reservierung eines Fluges ist nicht möglich) reagieren soll. Die zweite Ausprägung wird als Compensation Handling bezeichnet und wird in Situationen angewendet, in denen eine Teilaufgabe bereits vollständig bearbeitet und abgeschlossen wurde und nun aufgrund einer Ausnahme rückabgewickelt werden muss (z.B. nachdem eine Reisebuchung erfolgt ist, nimmt der Kunde die Reiserücktrittsversicherung in Anspruch).

[22] Es gibt weitere Interaktionstypen, bei denen die Administration (z.B. Vergabe von Rechten) und Konfiguration (z.B. Integration mit anderen WfMS) des WfMS im Vordergrund stehen. Darüber hinaus besteht die Möglichkeit,

4.2 Komponenten eines Human Workflow Service

BPEL-basierte und -konforme WfMS bieten wie in Kapitel 3 beschrieben keine expliziten Konzepte zur Einbindung von Personen in Workflows an. Stattdessen müssen zusätzliche Dienste (Human Workflow Services) angebunden werden. Im Folgenden werden die wesentlichen Anforderungen und Komponenten eines solchen Human Workflow Services anhand der Abbildung 4 erläutert. Diese Darstellung orientiert sich dabei am Oracle BPEL Process Manager, weist jedoch die allgemein erforderlichen Komponenten auf.

Wie in Abbildung 4 dargestellt ist ein Human Workflow Service das Bindeglied zwischen einer Process-Engine, Verzeichnisdiensten (z.B. LDAP), Benachrichtigungsdiensten (z.B. SMS, Pager) und einer Vielzahl möglicher Client-Anwendungen, die wiederum als Bindeglied zu den Benutzern fungieren. Als Client-Anwendung kann beispielsweise ein Unternehmensportal dienen (dieses wird üblicherweise noch andere Funktionen, z.B. als Kommunikationsplattform oder Dokumentenmanagement, übernehmen). Mit Hilfe des Portals werden Personen über neue für sie anstehende Aufgaben informiert. Darüber hinaus können Inputdaten für die Bearbeitung einzelner Aufgaben bereitgestellt und Ergebnisdaten in Form von aufgabenspezifischen Formularen erhoben werden. Als Client-Anwendung können auch andere Anwendungen genutzt werden, die regelmäßig von Personen eingesetzt werden und über eine geeignete, anpassbare Benutzerschnittstelle verfügen. Hierzu zählen beispielsweise Front-Ends von ERP-Systemen oder auch erweiterte E-Mail-Clients, die über Worklist-Funktionalitäten (z.B. Outlook) verfügen.

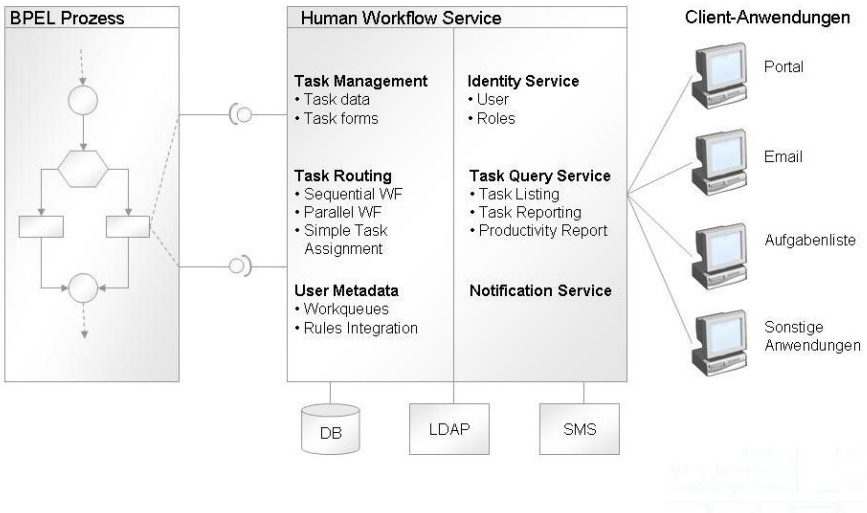


Abbildung 4: Komponenten eines Human Workflow Services und Anbindung an ein BPEL-basiertes WfMS [23]

einzelne Workflow-Instanzen und übergreifende Kennzahlen (z.B. Durchlaufzeiten, Fehlerquoten) zu analysieren und zu überwachen.

[23] In Anlehnung an Oracle http://www.oracle.com/technology/bpel/index.html. Eine lesenswerte Diskussion der wesentlichen Komponenten von Human Workflow Services findet sich auch in http://msdn2.microsoft.com/en-us/arcjournal/bb267382.aspx

Für die Generierung von Aufgaben auf der Basis einer Anforderung aus einem Workflow ist das Aufgabenmanagement (Task Management) verantwortlich. In Zusammenarbeit mit dem Identitätsdienst (Identity Service) wird zunächst der berechtigte Personenkreis festgelegt. Anschließend werden die für die Bearbeitung erforderlichen Daten zur Präsentation vorbereitet. Sofern die Erledigung der Aufgabe die Eingabe von Daten erfordert, übernimmt das Aufgabenmanagement darüber hinaus die Erstellung entsprechender Formulare (das sog. Rendering), die dann beispielsweise in einem Portal angezeigt werden können. Der Identitätsdienst stellt die Schnittstelle zu Verzeichnisdiensten wie LDAP dar und hat neben der Zuordnung von Personen zu Rollen die Aufgabe, Personen zu authentifizieren und zu autorisieren sowie in Verzeichnisdiensten abgebildete Organisationsstrukturen für Workflows beispielsweise im Rahmen von Delegation und Eskalation nutzbar zu machen. Eigenschaften und Regeln für einzelne Benutzer, die nicht in externen Verzeichnisdiensten abgebildet werden, können in der Komponente User Metadata gepflegt werden. Hier können beispielsweise unterschiedliche Aufgabenlisten pro Benutzer (z.B. eine Aufgabenliste mit niedrig und eine mit hoch priorisierten Aufgaben) definiert und Vertretungs- und Urlaubsregelungen festgelegt werden.

Das Task Routing ist für Reihenfolge, Parallelität und Synchronisation von Aufgaben verantwortlich. Über die Nutzung sog. Workflowmuster kann festgelegt werden, welcher Benutzergruppe wann eine bestimmte Aufgabe vorgelegt wird und ob diese beispielsweise im Rahmen einer Abstimmung parallel und verdeckt ausgeführt werden soll. Der Task Query Service erfüllt zwei Aufgaben: zum einen wird der Dienst verwendet, wenn Client-Anwendungen benutzerspezifische Anfragen bzgl. eigener Aufgaben haben oder den Status einer bestimmten Workflowinstanz abfragen wollen. Zum anderen kann der Dienst genutzt werden, um allgemeine Reports beispielsweise zur Entwicklung der Produktivität zu generieren. Der Notification Service in Abbildung 4 schließlich stellt die Schnittstelle zu technischen Kommunikationsdiensten wie beispielsweise einem SMS Gateway zur Verfügung.

Bisher hat sich noch kein Standard für die Interaktion zwischen Personen und Workflows herausgebildet. Allerdings gibt es seit Mitte 2005 Bestrebungen von Firmen wie IBM und SAP, BPEL um Konzepte zur Unterstützung von Interaktionen zwischen Personen und Workflows zu ergänzen (BPEL4People). Im Vordergrund steht dabei die Einführung von sog. People Activities – den Aufgaben mit ihren zuvor genannten Eigenschaften – und People Links, die Prozessrollen abbilden sollen. Ob und wann diese Bestrebungen in einem akzeptierten Standard münden werden, ist noch nicht absehbar. Festzuhalten bleibt jedoch, dass bereits heute mit der Einführung von Human Workflow Services komplexe Workflows unter Einbindung von Personen modelliert, umgesetzt und ausgeführt werden können.

5 Anwendung im Hochschulumfeld

Die in den vorangegangenen Kapiteln vorgestellten Konzepte und Technologien haben einen hohen Reifegrad erreicht und ermöglichen in der Praxis die Realisierung entsprechender Effizienzpotentiale in der Ablauforganisation.

Wie eingangs erwähnt gibt es nach unserer Erkenntnis an vielen deutschen Hochschulen kein konsequentes Prozessmanagement, geschweige denn eine technische Plattform zur Unterstüt-

zung der Prozessautomatisierung. Selbst ein explizites (und dokumentiertes) Prozessverständnis existiert häufig nicht[24].

Im Rahmen eines Pilotprojektes an der Leuphana Universität Lüneburg wurden die Ursachen für die mangelnde Prozessorientierung an Hochschulen am Beispiel eines dezentralen Beschaffungsprozesses untersucht[25]. Als wesentliche Gründe für eine mangelnde Prozessorientierung konnten einerseits der initiale Aufwand der Etablierung einer solchen Organisation und andererseits der im Vergleich zu Unternehmen immer noch als gering empfundene Druck zur Veränderung und Optimierung identifiziert werden. Beides führt zu innerorganisatorischen Widerständen bei der Einführung und Automatisierung von Teilprozessen, deren Wirkung durch die föderalen Strukturen innerhalb öffentlicher Hochschulen noch verstärkt wird.

Nach unserer Einschätzung ist die uneingeschränkte Unterstützung der Hochschulleitung wesentliche Voraussetzung für eine erfolgreiche Prozessimplementierung. Hilfreich sind darüber hinaus einfache Handhabung sowie Integration in bereits anderweitig verwendete und akzeptierte Anwendungen.

Häufig unterschätzt wird die Kommunikation des bereichsübergreifenden Nutzens. Gerade bei den in öffentlichen Hochschulen vorherrschenden dezentralen Organisationsstrukturen stellt sich der Nutzen einer Prozessautomatisierung nicht zwingend bei allen Anwendern direkt ein. Während etwa bei einem Workflow zur Abbildung von Reisekostenerstattungen der Aufwand in der Verwaltung sehr schnell abnimmt, hat der Reisende zunächst einen hohen Umgewöhnungsaufwand. Abhängig von der Reisehäufigkeit bleibt der Erfassungsaufwand nach der Umgewöhnung annähernd unverändert (bei den vielen Wenig-Reisenden) oder nimmt leicht ab (bei den wenigen Viel-Reisenden). Für viele Akteure ist somit lokal kein signifikanter Nutzen erkennbar. Hier ist es wichtig, den bereichsübergreifenden Nutzen transparent zu machen.

Bei der Projektplanung sollte wo möglich versucht werden, über die Realisierung von Quick-Wins den projektinduzierten Nutzen für den Anwender direkt spürbar werden zu lassen.

Trotz der diskutierten Schwierigkeiten gehen die Autoren davon aus, dass der zunehmende Druck durch die Differenzierung und erhöhte Eigenverantwortung der Hochschulen auch in diesem Umfeld mittelfristig zur verstärkten Prozessorientierung und der Nutzung von WfMS führen wird.

[24] An der Leuphana Universität Lüneburg wurde in 2007 ein erster Entwurf eines umfassenden Prozesshandbuchs erarbeitet. Während die Zielsetzung – nämlich die Prozessoptimierung – offensichtlich ist, ist der organisatorische Rahmen bisher unklar geblieben. Es gibt keine klaren, fachlich getriebenen Teilverantwortlichkeiten für die Inhalte bzw. definierte Vorgehensweise zur Aktualisierung. Eine dauerhafte (und damit auch Ressourcen bindende) Unterstützung dieses Themas ist aber zwingend erforderlich, um die Zielsetzung zu erreichen.
[25] Vergleiche dazu: Müller 2007.

Literatur- und Quellenverzeichnis

Van der Aalst, W.; van Hee, K. M.: Workflow Management, MIT Press, 2002

Andriessen, J. H. E.: Working with Groupware: Understanding and Evaluating Collaboration Technology, Berlin, Springer Verlag (2002)

Becker, J.; Kugeler, M.; Rosemann, M.: Prozessmanagement – Ein Leitfaden zur prozessorientierten Organisationsgestaltung, 5. Auflage, Berlin, Springer Verlag (2005)

Funk, B.; Niemeyer, P.: „Führen mit Hife IT-gestützter Workflows“, Boizenburg, vwh-Verlag, 2007

Gadatsch, A.: Grundkurs Geschäftsprozess-Management, Wiesbaden, Vieweg Verlag (2005)

Greif, I.: Computer-Supported Cooperative Work - A Book of Readings, San Mateo CA, Morgan Kaufmann Publischers (1988)

Hasenkamp, U.; Syring, M.: CSCW in Organsationen - Grundlagen und Probleme in: Hasenkamp, Ulrich; Kirn, Stefan; Syring, Michael (Hrsg.): CSCW - Computer Supported Cooperative Work - Informationssysteme für dezentralisierte Unternehmensstrukturen, Bonn, Addison-Wesley Verlag (1994)

Müller, S.: Optimierung eines Beschaffungsprozesses unter Einsatz des Integrationsservers SAP Exchange Infrastructure, Diplomarbeit Universität Lüneburg

Nicolescu, V.; Funk, B.; Heiler, M.; Niemeyer, P.: SAP Exchange Infrastructure for Developers, SAP Press (2007)

Sommerlatte, T.; Wedekind, E.: Leistungsprozesse und Organisationsstruktur, in: Management der Hochleistungsorganisation, Hrsg.: A. D. Little, Wiesbaden 1990, S. 23-41.

Teufel, S. et al.: Computerunterstützung für die Gruppenarbeit, Addison-Wesley, Bonn (1995)

WfMC: Workflow Management Coalition Terminology & Glossary,
http://www.wfmc.org/standards/docs/TC-1011_term_glossary_v3.pdf

White, S. A.: Introduction to BPMN, IBM Coorperation, 2004,
http://www.bptrends.com/publicationfiles/07-04%20WP%20Intro%20to%20BPMN%20-%20White.pdf

Ist D ein ernstzunehmender Nachfolger für C/C++?

Karl Goede

1 Einführung

Ein Blick auf die heutige Informatikwelt zeigt eine bunte Landschaft sehr vieler Programmiersprachen. D (s. [1]) gehört dabei zu jener Gruppe, die nicht von einem Komitee, sondern von einem qualifizierten Einzelkämpfer entwickelt wurde. Der Name Walter Bright, des Erfinders/Entwicklers von D, steht dabei neben Namen wie Niklas Wirth (Pascal) und Yukihiro Matsumoto (Ruby). D zielt auf den Einsatz in der Systemprogrammierung, ein Feld in dem die Programmiersprachen C [2] und C++ [3] dominieren.

Aus ökonomischen Gründen wäre in der IT-Welt die Konzentration auf einige wenige Sprachen sinnvoll. Ein mit einem gewissen religiösen Fanatismus begabter fachfremder Controller könnte sogar noch weiter gehen. Er könnte die Konzentration auf nur eine einzige Sprache fordern, quasi als Gipfel ökonomisch korrekten Verhaltens. Er wäre dabei einig im Glauben mit manchen älteren Vertretern einer jahrzehntelang die IT-Landschaft beherrschenden Firma, die einmal PL/1 für die Erlösung aus der Sprachenvielfalt hielt. Als Quasi-Vereinigung der seinerzeitigen Versionen von FORTRAN, ALGOL und COBOL sollte sie alle anderen Sprachen überflüssig machen. Es wurde jedoch nichts daraus.

Im Gegensatz zu den unvergänglichen „Ursprachen“ FORTRAN und COBOL hatte aus Sicht hartnäckiger Erneuerer PL/1 immerhin den Anstand, zusammen mit ALGOL 60, ALGOL 68 und vielen anderen „Altsprachen“ zügig das Zeitliche zu segnen. Diesen „Anstand“ hat die Programmiersprache C nicht. Dies immerhin hat sie mit FORTRAN und COBOL gemeinsam. Immer wieder als Negativbeispiel genutzt, welches Sprachen wie Pascal oder Ada in besonders hellem Licht zeigt, will C einfach nicht sterben. Auch das „eigene Lager“ in Form von C++ oder - einen Halbton höher - C# (Cis oder C-Sharp, wie Leute mit anglifiziertem Bildungshintergrund sagen) bringt die Ablösung nicht.

> Ein kurzer Versuch, mit einem C++-Programm (oder - schwieriger noch - mit einem Java-Programm) eine DLL (Dynamic Link Library; SO/Shared Objects unter Unix/Linux)[1] zu nutzen, führt bereits zu einem Misserfolg. Beim Versuch, mangelhafte Kontrollmöglichkeiten beim Zusammenbinden (Linkage) von dem, was C/C++ unter Modulen versteht, zu einem Programm, werden in der C++-Welt Funktionsnamen mit Zusätzen versehen, die zum Bindezeitpunkt Typkontrollen der Funktionsparameter erlauben. Damit diese „decorated names“ nicht auftreten, muss innerhalb eines C++-Programms dann auf C „zurückgeschaltet“ werden.

[1] Falls man sich fragt, wo die eine Rolle spielen: Auf dem Laptop mit Windows Vista (c), auf dem dieser Text hier entstand, war die abenteuerlich hohe Anzahl von fast 500 solcher Dynamic Link Libraries gleichzeitig geladen, während nur die Textverarbeitung LYX (ein TEX/LaTEX-Aufsatz) lief. Das Einfügen einer Portabilitätsschicht in Form etwa der Java Virtual Machine erhöht die Gesamtkomplexität weiter. Unterhalb dieser Schicht gibt es dann gleich noch ein paar mehr DLLs.

C++ war der erste in weiten Kreisen wahrgenommene Versuch[2], ein Klassenkonzept zu verwirklichen. C++ wurde angenommen; es bot dafür hinreichend viele Anreize. Es fragt sich, ob D seinerseits so viele Vorteile bringt, dass es zumindest für neue systemnahe Projekte im Vergleich zu C/C++ zur Nutzung motiviert. Es schließt sich die stärkere Frage an, ob die Vorteile sogar soweit reichen, alte C/C++-Programme nach D zu konvertieren. Tatsächlich lassen sich sehr viele Argumente zugunsten von D finden. Sie betreffen vorwiegend die Bereiche Sicherheit/Zuverlässigkeit und Komfort/Programmiereffizienz. Eine Verbesserung der Laufzeiteffizienz gibt es nicht. Bis auf weiteres garantiert D eher etwas längere Laufzeiten und auch längere Übersetzungszeiten als C++.

Eine nochmals weitergehende Frage wäre es, ob D auch dann noch hinreichend Vorteile bietet, wenn die Laufzeiteffizienz nur eine geringe Rolle spielt und Spezifika von Betriebssystemen nicht aufgegriffen zu werden brauchen. Ein Vergleich mit dem laut „Programmiersprachen-Hitliste“ (s. [4]) führenden Java bietet sich dabei nicht an, da man es damit D zu leicht macht. Daher wird im Folgenden unter diesem Aspekt die zwar „langsame“[3], aber hinsichtlich der Programmiereffizienz kaum schlagbare dynamisch typisierende Programmiersprache Ruby (s. [6]) herangezogen. Ruby wird hier als Vergleichsmaßstab gegenüber Perl, Python und PHP der Vorzug gegeben, da sie als perfekt objektorientierter Vertreter dieser Sprachklasse noch weniger als seine Mitbewerber mit dem Attribut „Skriptsprache“ als eigentlich untauglich für „richtige Programmierung“ abgetan werden kann.

2 Eigenschaften von D

Typisch für D ist relativ viel Großzügigkeit in der Syntax, die sich ein wenig jener dynamisch typisierender Sprachen nähert. Für Anhänger des Standpunkts „es soll möglichst nur einen Weg geben, einen Sachverhalt auszudrücken“ ist dies ein Gräuel. Für eine möglichst leichte Übernahme alter C/C++-Quellen (oder Java-Quellen) ist es hingegen ein Segen. Typisch ist gleichfalls, dass es dafür keine „Strafe“ in Form verminderter Effizienz wie bei Ruby gibt.

C++ war eine nahezu perfekt aufwärtskompatible Weiterentwicklung von C. Mit D hat Walter Bright hingegen einen deutlichen Bruch zu C und C++ vollzogen. Zu Recht, wie ein Blick auf nur ein paar Gegebenheiten zeigt:

- Die Präprozessor-Technik (Makros, Include-Dateien, bedingte Kompilation) macht große C/C++-Programme sehr unübersichtlich - selbst bei eiserner Disziplin der Programmierer. Eine automatische Analyse etwa der Querverbindungen der mit Include-Dateien notdürftig „aneinander gekitteten“ Module[4] durch Softwarewerkzeuge ist äußerst schwierig.
- Die von C++ angebotene Mehrfachvererbung ist eher eine Last als Erleichterung, nicht nur wegen des „Diamanten-Problems“. In der Mehrzahl der Fälle ist die u. a.

[2] Andere Sprachen, etwa das um Jahrzehnte ältere Simula 67, verschwanden in der Versenkung.
[3] Das ändert sich gerade. Zum einen ist die Entwicklung der schnell arbeitenden Ruby-spezifischen portablen virtuellen Machine Yarv nahezu abgeschlossen. Zum anderen führt Microsoft IronRuby (s. [5]) ein und erweitert dazu (auch für Python) seine .NET-Umgebung um eine Schicht, welche die Konzepte aufnimmt, die oberhalb von C#, Java etc. liegen. Auch SUN ist rührig und ergänzt seine Java-Welt um ein „schnelles Ruby“.
[4] Allein das Wort „Modul“ ist bereits ein Euphemismus. In vorsichtiger Sprechweise wird daher lieber nur von separat kompilierbaren Einheiten gesprochen.

in Java gepflegte Abbildung der die Zahl 1 überschreitenden Erblasser auf Interfaces genau so gut. Die von Ruby bereitgestellte Mixin-Technik ist sogar eher besser.

- C/C++ sieht ausschließlich eine statische Typprüfung vor. Eine für die objektorientierte Programmierung vielfach notwendige dynamische Typkontrolle muss vom Programmierer „zu Fuß“ realisiert werden[5].

D ist wie Java eine im Prinzip eigene Entwicklung mit dem Ziel, die Gefühle der anzulockenden C/C++-Programmierer möglichst wenig zu verletzen. Im Vergleich zu Java werden dabei auftretende Schmerzen durch sehr viel „syntactic sugar“ gelindert. Wie C++ ist D nicht doktrinär auf eine objektorientierte Programmierung festgelegt. Es lässt alternativ oder parallel dazu auch eine funktional geprägte Arbeitsweise zu[6].

2.1 D-Objekte im Vergleich zu C++-Objekten

C++-Objekte gleichen bezüglich ihres Attributanteils „normalen“ Variablen. Sie sind schlicht entsprechend Abb. 1 darstellbar.

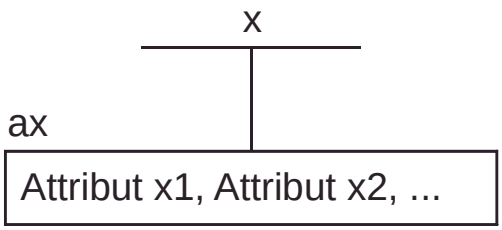


Abbildung 1: Objekt in C++

Insbesondere folgen sie der üblichen Stack-Disziplin. Der von ihnen belegte Speicherplatz wird automatisch mit dem Verlassen ihres Gültigkeitsbereiches durch eine einfache Änderung des Top-of-Stack-Zeigers (standardmäßig ein Register) gleich allen anderen lokalen Variablen und Funktions-/Methodenparametern schlagartig freigegeben. Effizienter geht es nicht.

D schließt sich hingegen der u. a. in Object Pascal und Java bevorzugten Verfahrensweise an und legt den Attributanteil der Objekte auf dem Heap ab (s. Abb. 2).

[5] In großen Rahmenwerken wie etwa der MFC (Microsoft Foundation Class) für C/C++ oder der VCL (Visual Component Library) für Object Pascal wird das dem Anwendungsprogrammierer selbstverständlich abgenommen.

[6] Compilerschreibern bleibt es dabei selbstverständlich unbenommen, ein vollständig objektorientiertes System zu schaffen und durch „syntactic sugar“ dem Programmierer bei Bedarf die Illusion rein funktionalen Arbeitens zu vermitteln, wie es z. B. Ruby eindrucksvoll zeigt.

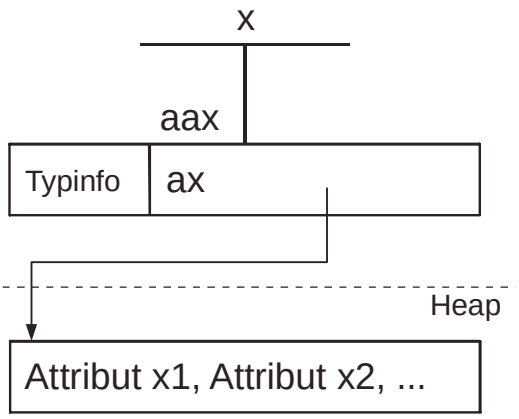


Abbildung 2: Objekt in D

An dieser Stelle wird D daher C++ hinsichtlich der Effizienz des erzeugten Codes stets ein ganz klein wenig unterlegen bleiben, sofern keine dynamische Typisierung benötigt wird. Eine dynamische Typisierung muss in C/C++ ja vom Programmierer durch Instrumentierung des Codes „nachgerüstet“ werden - und das macht dann plötzlich C++ „teurer“.

Hinzu tritt in D eine merkwürdige Art von „Teilobjekten“, nämlich **Slices** (s. Abb. 3). Die im Bild gezeigte Situation wäre wie folgt erreichbar:

```
int[9] a; // oder im C-Postfix-Stil: int a[10];
int[] s = a[4..7];
```

Etwas Vergleichbares ließe sich in C++ (und C) nur mit einem hochriskantem mit zusätzlichem Verwaltungsaufwand behafteten schmutzigen Trick für nullterminierte Strings darstellen. Dabei wäre „mitten“ in einen bestehenden String an der passenden Stelle das dort gerade befindliche Zeichen zwischenzuspeichern und die für den Teilstring benötigte „Brems-Null“ temporär einzusetzen. Außerdem wären Zugriffe auf den längeren „Originalstring“ zu unterbinden bis die temporäre Brems-Null wieder durch das dort original hingehörende Zeichen ersetzt wurde. Eine andere Alternative wäre natürlich die Simulation der Slices durch vom Programmierer explizit einzuführende und explizit zu nutzende Zeiger-Zähler-Kombinationen.

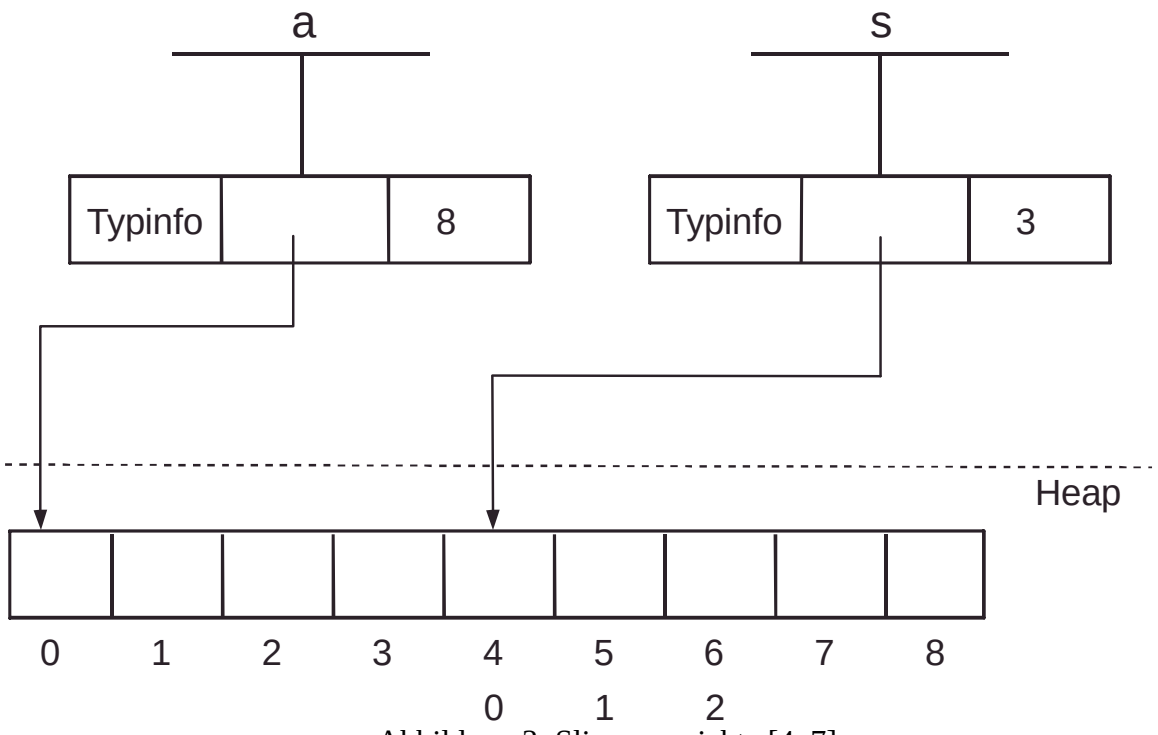


Abbildung 3: Slice s erreicht a[4..7]

2.2 Zuverlässigkeitsfragen

Design by Contract Per assert-Anweisung lässt sich die Prüfung von Vor- und Nachbedingungen sowie Invarianten erzwingen (mit dem ewigen Dilemma: Tests abschalten (geschieht in der Release-Einstellung) oder nicht).

Andockpunkte für Extreme Testing Für D-Klassen gibt es bei Bedarf (den haben XP-Leute [7] erfreulicherweise immer) eine spezielle Methode namens **unittest**.

Beispiel (s. [1], S. 92)

```
class Sum {

        int add(int x, int y) { return x + y; }

        unittest {
            assert(add(3,4) == 7);
            assert(add(-2,0) == -2);
        }
}
```

Definierte Darstellungen einfacher Datentypen C/C++ lässt u. a. offen, wie viele Bits für **int-, long-, real**-Werte etc. verwendet werden und macht die Programmierung in einer dieser Sprachen noch risikoreicher als ohnehin schon. D trifft hingegen klare Festlegungen. Damit kann es zwar nicht mehr „automatische“ Anpassungen an sich ändernde Rechnerarchitekturen[7] geben; immerhin gibt es prophylaktisch schon einen Datentyp **cent** mit einer Breite von 128 Bits (Walter Bright denkt „dollar/cent“-technisch offenbar sehr schwarz in die Zukunft).

[7] C wurde z. B. mal mit 16-Bit breiten Integerzahlen gestartet. Auf PCs sind z. Zt. 32 Bits üblich.

Außerdem könnte man **long** (64 Bits in D) bei Bedarf ohne Weiteres um vielleicht ein **very long** mit einer Breite von 256 Bits aufstocken.

Automatische Garbage Collection Wie Objective C, Ruby, Java etc. sieht D eine automatische garbage collection vor. Allerdings lässt sich bei Bedarf der Heap auch direkt verwalten. Wie sollte man auch sonst z. B. einen garbage collector schreiben? D verlässt sich als Systemprogrammierungssprache nicht wie Java darauf, dass solch eine „Schmutzarbeit“ in C getan und per JNI (Java Native Interface) angebunden wird.

2.3 Kompatibilität zu C/C++

D-Statements und D-Expressions sind weitgehend so gestaltet wie in C/C++. Ein ehemaliger C++-Programmierer findet sich beispielsweise in D-Schleifen, D-switch-Konstrukten und dergleichen leicht zurecht. Analoges kann man von einem C++-Programmierer sagen, der nach Java „konvertiert“. Allerdings lassen sich - anders als bei einer Übertragung von C++ nach Java – auch komplexere C++-Konstrukte in D nachbilden, wie sich im Folgenden zeigt.

C/C++-Typbildner Es ist alles dabei, um (halb-)automatische Umwandlungen von C/C++ nach D zu erleichtern. Es gibt also insbesondere die Typbildner **typedef, enum, union, struct** und **class**.

Überladbare Operatoren Analog zu C++ lassen sich innerhalb von Klassen Methoden auch in Operatorenform definieren. Die Syntax der Operatorendefinition ist allerdings in D nicht besonders gut gelungen[8]. Das folgende Beispiel zeigt - nicht ganz ohne Hintergedanken, wie unten zu sehen sein wird - eine mögliche Implementierung für den Operator += für Geld-Objekte. Wenn hier etwas zu addieren ist, muss die Währung berücksichtigt werden. Normalerweise muss der Kurs einer Datenbank entnommen werden; hier fällt er in Form einer „magic number“ zur Verkürzung des Beispiels „vom Himmel“.

Beispiel für eine += -Überladung (Auszug aus einer „Geld-Klasse“)

Implementierung in C++ (mit der „magic number“ 1.51 versehen)

```
#include <stdio.h>

enum TWaehrung {EURO, USD};

class Geld {
         TWaehrung waehrung;
         float betrag;
   public:
         Geld(TWaehrung w, float b) {waehrung = w; betrag = b;}
         void operator += (Geld b);
         void wert(char *wert);
};

void Geld::operator += (Geld b) {
  float kurs = 1.0;
  switch (b.waehrung) {
```

[8] Beim Test dieser Konversion saß der Autor z. B. einem älteren D-Text auf, bei dem es **addass** anstelle von **opAddAssign** (s. u. im D-Port) hieß. Bei einer Syntax, die sich wie D und Ruby an die echten Operatorennamen hält, wäre solch eine Verwechslung von vornherein unmöglich.

```
          case EURO: if (waehrung == USD) kurs = 1.51; break;
          case USD: if (waehrung == EURO) kurs = 1/1.51; break;
  }
  betrag += kurs * b.betrag;
}

void Geld::wert(char *wert) {
  sprintf(wert, "Wert: %8.2f %s", betrag,
        (waehrung == EURO ? "Euro" : "US-Dollar"));
}

int main() {
  Geld meins(EURO, 2000), seins(USD, 4000);
  char buffer[80];
  meins += seins;
  meins.wert(buffer); printf("%s\n", buffer);
  return 0; // Ausgabe: Wert: 4649.01 Euro
}
```

Analoge Implementierung in D

```
import std.stdio;
import std.string;

enum TWaehrung {EURO, USD};

class Geld {
          TWaehrung waehrung;
          float betrag;
    public:
        this(TWaehrung w, float b) {waehrung = w; betrag = b;}

        void opAddAssign(Geld b) {
                    float kurs = 1.0;
                    switch (b.waehrung) {
              case TWaehrung.EURO:
                 if (waehrung == TWaehrung.USD) kurs = 1.51; break;
            case TWaehrung.USD:
                 if (waehrung == TWaehrung.EURO) kurs = 1/1.51; break;
          }
          betrag += kurs * b.betrag;
       }

       void wert(out string wert) {
           wert = format("Wert: %8.2f %s", betrag,
                 (waehrung == TWaehrung.EURO ? "Euro" : "US-Dollar"));
       }
};

int main() {
  Geld meins = new Geld(TWaehrung.EURO, 2000);
  Geld seins = new Geld(TWaehrung.USD, 4000);
  string buffer;
  meins += seins; // Es ginge auch: meins.opAddAssign(seins);
  meins.wert(buffer); writefln(buffer);
  return 0; // Ausgabe: Wert: 4649.01 Euro
}
```

Wie in Pascal hat D als Ersatz für das nicht ungefährliche **sprintf** (die gibt's für D nur noch im Paket std.c.stdio) durch eine Funktion namens **format** ersetzt, welche einen String als Wert liefert.

Die Abkehr von *sprintf* ist allerdings nichts, was C/C++-Kenner ins Grübeln bringen dürfte.

Die werden sich eher fragen, wieso das Beispiel derart schlecht gestaltet ist, dass die **+=**-Operation keinen Wert liefert. Den Wert der linken Seite eben, so wie man das von einer Zuweisungsoperation in C/C++ gewohnt ist - auch wenn hier nebenbei (oder doch eher als Haupteffekt?!) addiert wird. Es gibt eine Antwort darauf: „Das wird teuer!“. Die Operatordefinition in D müsste dann folgendermaßen aussehen:

```
Geld opAddAssign(Geld b) {
  float kurs = 1.0;
  switch (b.waehrung) {
    case TWaehrung.EURO:
        if (waehrung == TWaehrung.USD) kurs = 1.51; break;
    case TWaehrung.USD:
        if (waehrung == TWaehrung.EURO) kurs = 1/1.51; break;
  }
  betrag += kurs * b.betrag;
  Geld result = new Geld(waehrung, betrag);
  return result;
}
```

Die Return-Anweisung gestattet es nicht, einfach ein Tupel aus **waehrung** und **betrag** zurückzuliefern. Stattdessen muss eigens ein neues Objekt auf dem Heap erzeugt werden. Auf Verdacht; es ist ja unklar, ob es „außen“ eine Zuweisung des Ergebnisses der Operation an ein anderes Objekt geben soll. Tatsächlich ist dieser Fall eher unwahrscheinlich.

Die Entscheidung, Tupel-Returns nicht zuzulassen, wirkt auf den ersten Blick etwas kleinkariert. Ruby erledigt so etwas scheinbar nebenbei in vollkommener Schönheit. Tatsächlich wird „so etwas“ nicht „nebenbei“ gemacht. Die Tupel-Rückgabe frisst Rechenzeit. Der D-Erfinder hat erhebliche Arbeit darin investiert, den resultierenden Effizienzverlust zu vermeiden. Ein Symptom dafür: kanonisch liegen Rückgabewerte auf dem Stack und werden dort vom Aufrufer abgeholt bzw. - wenn möglich - via Stackadresse als Operanden in den erzeugten Maschinenbefehlen verwendet. D ist hingegen so konstruiert und wird von den existierenden Compilern so verwendet, dass der Returnwert in einem (u. U. mehr als einem) passenden Register angeliefert wird, mit dem an der Benutzungsstelle des Resultats sofort weitergearbeitet werden kann. Dies gilt in der Regel auch für die zusammengesetzten Werte des Datentyps **complex** (s. 2.4).

Wenn der Nutzer von einer Funktion/Methode ein Tupel zurückgeliefert haben möchte, wird ihm nahegelegt, dies über OUT-Parameter zu erledigen. Solch ein Ansinnen verstößt allerdings massiv gegen eine Grundregel der reinen Informatik:

dass nämlich eine Funktion/Methode nur einen Wert ermitteln und abliefern soll. Sie soll dabei den Speicherzustand nicht ändern, also keine Seiteneffekte haben.

Die legitime Interpretation der **+=**-Operation als komfortabel nutzbare Funktion weist hier bereits einen Verstoß auf. Natürlich, es ist ja gerade der Zweck einer Zuweisung, den Speicherzustand zu ändern.

D zeigt zu Recht Härte - notfalls auf Kosten der Schönheit[9]. Wenn im Beispiel oben doch wenigstens indirekt ein Tupel per Return zurückgegeben werden soll, lässt D den Programmierer den Preis dafür spüren. Die Abwesenheit von Tupelrückgaben und andere Härten kos-

[9] Den Methodenschreibern ist diese Diskussion ohnehin fremd. Sie finden es völlig normal, wenn eine Methode den Zustand eines Objekts verändert. Insbesondere „Setter“ gelten als „unverzichtbare“ Pfeiler des Software-Engineering.

ten andererseits natürlich Programmiereffizienz. Umgekehrt: Ein Ruby-Programmierer schafft in der Zeiteinheit leicht mehr als doppelt so viel wie ein D- (C++/Java dto.) Programmierer.

2.4 Erweiterungen gegenüber C/C++

Datentyp COMPLEX Etwas kurios für Kenner der FORTRAN-Szene ist die Vorstellung, FORTRANianer könnten für D gewonnen werden. Immerhin gäbe es für die Verweigerung einer Abkehr von FORTRAN eine Ausrede weniger: D unterstützt direkt den Datentyp COMPLEX und benötigt nicht dessen ineffiziente Simulation durch eine COMPLEX-Klasse. Es zeigt auch nicht die Dummheiten (insbesondere eine mangelhafte Exception-Organisation), die in (s. [8]) (*How Java's Floating-Point Hurts Everyone Everywhere*) keineswegs nur für Java angeprangert werden und auch in FORTRAN unangenehme Katastrophen z. B. bei Raketenflügen „erleichtern".

Eingebaute Mengenoperationen Angelehnt an das, was in C vor allem als Zählschleife verwendet wird, nämlich

for *(Vorarbeit ; Vorbedingung; Abschlusshandlung) Anweisung;*

gibt es eine Mengenoperation der Form

foreach *([Index,] Element; Aggregat)* ***Anweisung****;*

Inhaltlich entspricht das der Ruby-Array-Methode *each*.

Dynamische Arrays Aus der Sicht auf effiziente Programme angewiesener Programmierer gibt es eine starke Abneigung gegen Arrays mit veränderlicher oberer Grenze. In der Tat verlängert sich die Laufzeit im Vergleich zu statischen Arrays erheblich - wenn von der Dynamik in starkem Umfang Gebrauch gemacht wird. Nur: Sie müssen ja nicht benutzt werden; D kennt eben auch statische Arrays. Wenn aber eine Dynamik von der Problemstellung her erforderlich ist, ist es gewöhnlich besser, wenn die Programmiersprache sie bereit stellt, als wenn der Programmierer sie sich selbst „zusammenbastelt".

Die Art der Dynamik ist eher moderater Art. Es gibt nicht die starke (und unangenehm viel Rechenzeit verursachende) Dynamik von Ruby, die es beispielsweise gestattet, bei einem Array der aktuellen Länge 1024 „aus heiterem Himmel heraus" einen Wert an ein Element mit dem Index 10000 zuzuweisen - mit der Sicherheit, dass ein Zugriff auf ein Element im Indexbereich 1024..10000) (= [1024, 9999]) dabei garantiert nil liefert. Aber die Verschiebung der oberen Grenze wird leicht gemacht. Der Programmierer braucht nichts umzuspeichern und dabei garbage zu erzeugen, welches noch erkannt werden will, bevor es der garbage collector (so gut es geht) etwas recyceln kann.

Beispiel

```
int[] a;
```

definiert **a** als eindimensionales dynamisches Array. **a** „fasst" dabei analog zur weiter oben dargestellten Situation von Slices ein Zeiger-Längen-Paar an. Der Zeiger seinerseits wird auf

einen Bereich passender Größe auf dem Heap gerichtet. Bei einer Arrayverlängerung ist dann möglicherweise noch ausreichend Platz hinter dem allozierten Bereich. Anderenfalls wird woanders ein entsprechender Bereich auf dem Heap gesucht. Dies ist eine Situation, in der die erwähnte „programmierergebastelte“ Dynamik gefährlich wird, da sämtliche in den bisherigen (aufgegebenen) Bereich weisenden Zeiger (z. B. via Slices) bekannt sein und mit umgesetzt werden müssen.

Verlängerungen bzw. Verkürzungen eines dynamischen Arrays sind durch seine **length**-Methode möglich. Während diese Methode bei statischen Arrays ein reiner **Getter** ist, hat sie hier sowohl **Getter**- als auch **Setter**-Qualität. Aus dem Kontext ist für den Compiler kenntlich, was gerade benötigt wird.

Assoziative Arrays (vulgo _Hash-Tabellen_) *Beispiel* (s. [1], S. 80)

```
int[char[]] b; // associative array b of ints that are
// indexed by an array of characters
b["hello"] = 3; // set value associated with key "hello" to 3
func(b["hello"]); // pass 3 as parameter to func()
delete b["hello"];
```

2.5 Kompatibilität zu Pascal

Eine Portierung von Pascal nach D dürfte wohl eher zu den seltenen Ereignissen zählen. Immerhin lassen sich bereits anhand eines sehr primitiven Beispiels (s. 3.3) dafür relativ günstige Voraussetzungen feststellen, zumindest im Vergleich zu einer Portierung von Pascal nach C++. Unterstellt, alle Bezeichner wären hinsichtlich Groß-/Kleinschreibung bereits konsistent gemacht (also etwa umsatzsteuer, umsatzSteuer und UMsatzsTeuer gleichmäßig zu umsatz-Steuer gewandelt, gibt es für diesen Fall neben den üblichen Problemen zwei besonders schmerzhafte (Namenskonflikte sind aufzulösen) Sachverhalte:

- Pascal lässt lokale Funktionen zu
- Pascal kennt in Form der with-Anweisung die Möglichkeit, nahezu analog zu COBOL direkt mit Feldnamen in Record-Strukturen zu hantieren.

Beides wird von D jedoch unterstützt. Diesbezüglich wäre eine Umsetzung Pascal $\rightarrow$ D ein reines Vergnügen.

3 Fallbeispiele

3.1 Hallo Welt

Dies traditionell erste C-Demonstrationsprogramm sieht nach seinem Transfer nach D nahezu unverändert aus:

```
import std.stdio;

int main() {
   writef("Hallo Welt\n") ;
   return 0;
}
```

Unscheinbar, aber erheblich ist die Zeile

```
import std.stdio;
```

(bzw. *std.c.stdio*, falls die „historischen“ C-I/O-Funktionen wie *printf* benutzt werden sollen) anstelle der üblichen Präprozessor-Anweisung

```
#include <stdio.h>
```

Im- und Exportmechanismen sind fester Bestandteil der Sprache D. Das gleiche gilt für Sprachmittel zur Handhabung plattformspezifischer Codevarianten und Simulationen generischer Funktionen, bei denen C-Leute ebenfalls mit Hilfe des Präprozessors den Hals aus der Schlinge ziehen müssen (allerdings auch *können*).

Die Pascalwelt, so sie einen Seitenblick riskiert, wird irritiert feststellen: Wie in C/C++/Java müssen auch leere Parameterlisten angegeben werden.

3.2 Ein paar Array-Operationen

Es sei wie folgt ein Array unter Nutzung der „magic number“ 4096 deklariert worden.

```
int[4096] a; // oder im C-Stil: int a[4096];
```

Um dies Array auf Null zu setzen, würde ein auf Effizienz bedachter C-Programmierer üblicherweise

```
memset(a, 0, sizeof(a));
```

schreiben. Zwar ließe sich ohne weiteres auch in D unter Verwendung von Inline-Assemblercode eine genauso effiziente *memset*-Funktion realisieren. Es genügt aber bereits ein „Zipfel“ der mächtigen D-Arrayoperationen, um diese Aufgabe zu erledigen:

```
a[] = 0;
```

Wie oben (2.1) angedeutet, lassen sich Teilarrays (Slices) bequem und effizient ansprechen:

```
import std.stdio;

void main ( ) {
   int[9] a;   // standardmäßig mit Nullen initilialisiert

   void a_detailliert() { // lokale Funktion a la Pascal
      writefln("Array a:");
      foreach (i, element; a) writefln("a[%d] = %4d", i, element);
      writefln("-------------------------------");
   }
```

```
    for (int i = 0; i < a.length; i++) a[i] = 10 * i;
       // falls man auf foreach verzichten möchte
    a_detailliert();
    int[] s = a[4..7];
    writefln(a, s);
    s[1] = 4711;
    writefln(a, s);   // ... auch a[5] == 4711

    int zähler = 0;  // deutsche Umlaute zulässig - dank UTF-8
    foreach (ref element; a) {element = zähler; zähler += 100; }
    writefln(a);
}
```

Auf dem Bildschirm wird dabei folgendes ausgegeben:

```
Array a:
a[0] = 0
a[1] = 10
a[2] = 20
a[3] = 30
a[4] = 40
a[5] = 50
a[6] = 60
a[7] = 70
a[8] = 80
------------------------------
[0 10 20 30 40 50 60 70 80][40 50 60]
[0 10 20 30 40 4711 60 70 80][40 4711 60]
[0 100 200 300 400 500 600 700 800]
```

Wie immer irritierend - leider - ist die Verwendung der **..**-Notation. **a..b** bedeutet bei der Deklaration von Pascal-Arrays und in Ruby[10] das Intervall **[a, b]** (also a, a+1, ..., b) und hier in D das Intervall **[a, b)** (also a, a+1, ..., b-1). Das Gefühl des Autors sagt: D macht es richtig. In Internetforen wird aber auch gern mal das Gegengefühl behauptet.

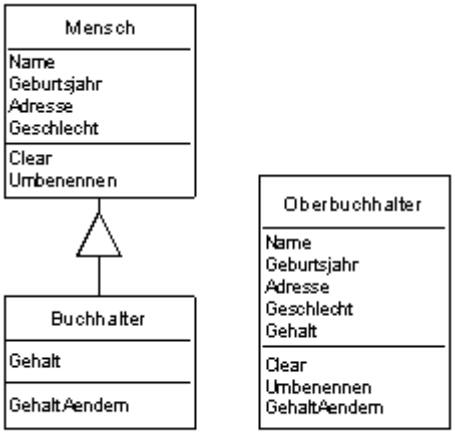
Auffällig ist die hohe Leistungsfähigkeit von **writef** und **write_n**. Zum einen lässt sie die Nutzung analog **write** und **writeln** von Pascal zu, zum anderen erlaubt sie String-Interpolationen mit Hilfe von Formatsteuerzeichen wie die **printf**-Familie von C/C++.

Ein eigener Stringtyp wird in D nicht benötigt, da der String-Effekt bequem durch dynamische Arrays erreicht wird. Gleichwohl gibt es ein String-Paket, um für häufig benötigte Zusatzfunktionen einen Rahmen zu schaffen.

3.3 Ein Klassenbeispiel

Es werden die folgenden einfachen Klassen betrachtet:

[10] In Ruby wird die Verwirrung noch getoppt, da es dort noch den Operator **...** gibt – mit **a...b** gleich dem unten abgeschlossenen und oben offenen Intervall **[a, b).**



Umgesetzt in D wird die im Diagramm dargestellte Struktur anhand einer Konversion aus einem in (Object-) Pascal formulierten Original unter Beschränkung auf die „Rahmenhandlung“. Dies zeigt nebenbei, dass die bei Konversionen von Pascal aus gefürchtete with-Anweisung (neben lokalen Funktionen) beim Ziel D keine Probleme aufwirft - schlicht deswegen, weil D die with-Anweisung auch bereit hält.

Genutzt wird der übernommene Teil des Originals für einen Test, wie sich der jeweilige Compiler bei Zuweisungen von Objekten einer Unter- an solche einer Oberklasse tatsächlich verhält (dies muss erlaubt sein, da dabei keine undefinierten Attribute erzeugt werden), bzw. wie bei der Gegenrichtung verfahren wird (üblicherweise verboten, da undefinierte Attribute entstehen[11]). Außerdem wird geprüft, wie auf Zuweisungen reagiert wird, die von den Attributen her möglich wären, in Programmiersprachen aus doktrinären Gründen aber gern untersagt werden[12] (was dem Compilerschreiber die Arbeit etwas erleichtert).

```
import std.stdio;
import std.string;

enum TGeschlecht {gUndefined, gWeiblich, gMännlich};
struct TAdresse { string Strasse,Plz, Wohnort; }

class TMensch {
 private
  string FName;
  int FGeburtsjahr;
  TAdresse FAdresse;
  TGeschlecht FGeschlecht;

public
 void clear() {
   FName = ""; FGeburtsjahr = 0;
   with (FAdresse) { Strasse = ""; Plz = ""; Wohnort = ""; }
   // FGeschlecht = TGeschlecht.gUndefined;
         // im Pascal-Original stand nur gUndefined statt
            //TGeschlecht.gUndefined
 }

 void Umbenennen(string NameNeu) {FName = NameNeu; }
}
```

[11] Das wäre kein Problem, wenn die Programmiersprache explizite Nil-Werte für alle Datentypen bereithielte und die Undefiniertheit also explizit kenntlich gemacht werden könnte.
[12] Die von C/C++/Java-Menschen gern verfemte Programmiersprache COBOL schurigelt die Programmierer hier nicht unnötig und lässt so etwas in Form des MOVE CORRESPONDING zu.

```
class TBuchhalter : TMensch {
private
  float FGehalt;
public
  void GehaltAendern(float GehaltNeu) { FGehalt = GehaltNeu; };
}

class TOberbuchhalter {
private
  string FName;
  int FGeburtsjahr;
  TAdresse FAdresse;
  TGeschlecht FGeschlecht;
  float FGehalt;
public
  void clear() {
    FName = ""; FGeburtsjahr = 0;
    with (FAdresse) { Strasse = ""; Plz = ""; Wohnort = ""; }
    FGeschlecht = TGeschlecht.gUndefined;
    // im Pascal-Original stand nur gUndefined statt
    // TGeschlecht.gUndefined
  }
  void Umbenennen(string NameNeu) {FName = NameNeu; }
  void GehaltAendern(float GehaltNeu) { FGehalt = GehaltNeu; };
}

void main() {
  int i;
  TMensch m = new TMensch;
  auto b = new TBuchhalter;
    // ok; ein Hauch von Menschenfreundlichkeit; der Compiler kann
    // für b den Datentyp TBuchhalter aus dem Zusammenhang
    // selbstverständlich  erschließen
  TOberbuchhalter ob = new TOberbuchhalter;
  b.GehaltAendern(4000);
  writefln(b); // -> "Buchhalter.TBuchhalter" auf dem Bildschirm
               //    (Reflection)
  m = b;   // ok
  b = m;   // Fehler - wird vom Compiler zu Recht als falsch moniert
  m = ob;  // Fehler - könnte von der Attributmenge her eigentlich
               // erlaubt werden
  ob = b;  // Fehler - könnte von der Attributmenge her eigentlich
               // erlaubt werden
  b = ob;  // Fehler - könnte von der Attributmenge her eigentlich
               // erlaubt werden
}
```

Für Leser mit älterem Java-Hintergrund[13] treten als hauptsächliche Überraschungen ein Enumerationstyp und die *with*-Anweisung auf.

C++-Kenner werden befremdet feststellen, dass das Beispiel nur „Inline-Methoden“ aufweist. Allerdings täuscht deren Eindruck. D kennt ausschließlich die Implementierung der Methoden „an Ort und Stelle“. Ob daraus Inline-Code wird, entscheidet nur noch der Optimierer im D-Compiler[14].

Für alle überraschend ist die Zulassung von Umlauten in Bezeichnern. Allerdings verlangt das dem Quelltext-Editor die Kenntnis des UTF-8-Codes[15] ab. Anstelle von **gMaennlich** im Pascal-Original ist daher hier demonstrativ die Bezeichnung **gMännlich** verwendet worden.

Seltsam dürfte die Zeile

[13] Ab Vs. 5.0 hat auch Java die Enumerationstypen als nützlich wahrgenommen.
[14] Der arbeitet sich allerdings erst langsam an ein Optimum heran. Z. Zt. erreichen D-Programme in der Regel nicht die Laufzeiteffizienz von C++-Programmen.
[15] Falls der SciTE-Editor verwendet wird, ist dazu eine Kopfzeile mit dem Inhalt _// coding: utf-8_ zweckmäßig.

```
auto b = new TBuchhalter;
```

anmuten, jedenfalls für Nutzer statisch typisierter Sprachen. In dieser Situation kann der Compiler für b problemlos den Datentyp **TBuchhalter** aus dem Zusammenhang erschließen. Hier hat Walter Bright großzügig etwas „syntactic sugar“ ausgestreut[16]. Im Beispiel wurde demonstrativ das auch mögliche TBuchhalter zwischen **auto** und **b** fortgelassen.

3.4 Inline-Assembler

In manchen Ausnahmesituationen ist noch die Programmierung in einer Assemblersprache interessant. Kandidaten sind bestimmte Schnittstellen zum Betriebssystem und Funktionen, die extrem die Laufzeiteffizienz beeinflussen. An einigen wenigen Stellen wird z. B. in der Phobos-Library (s. 4.3) davon Gebrauch gemacht. Es handelt sich u. a. um die Thread-Schnittstelle zum Betriebssystem und um ein paar mathematische Funktionen.

Gegenüber einem externen Assembler bietet ein Inline-Assembler ja eine sehr komfortable Schnittstelle zur übergeordneten Programmiersprache. Ein Codebruchstück aus Funktionen zur Berechnung des Message Digest MD5 zeigt das. Es zeigt zugleich, wie sich die Anpassung an die Plattform handhaben lässt. An die Stelle des typischen

```
#ifdef X86
...
#else
...
#endif
```

o. ä. unter Nutzung des C/C++-Präprozessors tritt hier das D-Sprachkonstrukt version. Hier beschränkt es die Realisierung eines assemblergestützten rotierenden Linksshifts („was links herausfällt, wird rechts wieder reingefahren“) auf Intel/AMD-Plattformen (andere Plattformen werden vom D-Compiler dmd (s. %%) auch nicht unterstützt). Falls keine X86-Architektur vorliegt, werden die gewöhnlichen von D (genau wie von C/C++/Java) direkt erreichbaren Links- und Rechts-Schiebeoperationen (<< und >>) benutzt. Auf Nicht-X86-Rechnern sehen Benchmarks, die den MD5 mit heranziehen, dann nicht mehr ganz so gut aus.

Beispiel *(aus der Datei md5.d der Phobos-Library)*

```
/* ROTATE_LEFT rotates x left n bits.
 */
static uint ROTATE_LEFT(uint x, uint n)
{
       version (X86)
       {
              asm
              { naked ;
               mov ECX,EAX ;
```

[16] Im Ruby-Fall ist eine derartige Großzügigkeit selbstverständlich. Aus z. B. der Zeile
b = TBuchhalter.create schließt der Interpreter selbstverständlich, dass b bis auf weiteres den dynamischen Typ TBuchhalter bekommen und ein Objekt erzeugt werden soll. Sinnlose leere Klammern und ein woanders aus ideologischen Gründen erforderliches abschließendes Semikolon werden dem Programmierer beim Schreiben und - sehr viel wichtiger – den Wartungsprogrammierern beim Lesen erspart. In den raren Fällen, in denen der Interpreter mit dergleichen nicht zurechtkommt, bittet er den Programmierer ausnahmsweise doch einmal um ein Semikolon, Klammern, ein Leerzeichen oder einen anderen kleinen syntaktischen Hinweis.

```
                mov EAX,4[ESP] ;
                rol EAX,CL ;
                ret 4 ;
                     }
        }
        else
        {
               return (x < < n) | (x > > (32-n));
        }
}
```

Die Assemblernutzung in der Funktion ROTATE_LEFT setzt selbstverständlich eine genaue Kenntnis der Parameterübergabe und der erforderlichen Lage des Resultats auf dem Stack voraus. Etwas einfacher gestaltet sich die Schnittstelle zwischen Assembler und D-Umgebung, wenn im Assemblerteil auf lokale Variablen Bezug genommen wird oder der Compiler geeignete Prolog- und Epilog-Sequenzen generiert (wird hier vom Schlüsselwort *naked* unterbunden).

3.5 DBMS-Treiber

Als letztes Fallbeispiel wird der Zugang von D aus auf in C/C++ geschriebene Software betrachtet. Dieses Beispiel zeigt zugleich Intermodulbeziehungen. Gezeigt wird der Zugang anhand der Nutzung des einfachen und schnellen Datenbanksystems SQLite. Dieses System präsentiert sich der Anwendungssoftware komplett als Dynamic Link Library oder als statisch anbindbare Funktionssammlung. Die Schnittstelle **sqlite3_imp.d** lässt sich besonders einfach herstellen (ganz im Gegensatz zur Java- oder (härter noch) zur Ruby-Schnittstelle), da C-Funktionen von D aus direkt aufrufbar sind. Ein kleiner Auszug aus **sqlite3_imp.d** zeigt dies (Bemerkung: /+ und +/ schließen in D schachtelbaren Kommentar ein):

```
module sqlite3_imp;
private import std.c.stdarg; // For va_list.
extern(C):
/*
       ** The version of the SQLite library.
*/
const char[] SQLITE_VERSIONl = "3.0.0";
...
int sqlite3_open(
  /+const+/ char *filename, /* Database filename (UTF-8) */
  sqlite3 **ppDb /* OUT: SQLite db handle */
);
int sqlite3_prepare(
  sqlite3 *db, /* Database handle */
  /+const+/ char *zSql, /* SQL statement, UTF-8 encoded */
  int nBytes, /* Length of zSql in bytes. */
  sqlite3_stmt **ppStmt, /* OUT: Statement handle */
  /+const+/ char **pzTail /* OUT: Pointer to unused portion of zSql */
);
...
```

Das Nutzungsprinzip für die Schnittstelle zeigt das Mini-Anwendungsprogramm **buchhandlung.d**

```
import sqlite3_imp;

int main() {module sqlite3_imp;
private import std.c.stdarg; // For va_list.
extern(C):
/*
** The version of the SQLite library.
*/
```

```
const char[] SQLITE_VERSIONl = "3.0.0";
...
int sqlite3_open(
  /+const+/ char *filename, /* Database filename (UTF-8) */
  sqlite3 **ppDb /* OUT: SQLite db handle */
 );
 int sqlite3_prepare(
   sqlite3 *db, /* Database handle */
   /+const+/ char *zSql, /* SQL statement, UTF-8 encoded */
   int nBytes, /* Length of zSql in bytes. */
   sqlite3_stmt **ppStmt, /* OUT: Statement handle */
   /+const+/ char **pzTail /* OUT: Pointer to unused portion of zSql */
);

...
  sqlite3* db;
  int code;

  code = sqlite3_open("Buchbestand.db", &db);
  if(code != SQLITE_OK) {
    printf("DB create error: %s\n", sqlite3_errmsg(db));
    return 1;
  } else printf("DB open!\n");
  // Nutzung der Datenbank Buchbestand
  ...
  ...
  sqlite3_close(db);
  printf("DB closed.\n");

  return 0;
}
```

4 Infrastruktur

4.1 Unterstützte Plattformen

Als grundlegende Plattform für D präsentiert sich Unix/Linux auf X86- und teilweise Power-PCSystemen. Die Art, in welcher der Ursprungs-D-Compiler in C/C++ geschrieben wurde, garantiert nahezu verzögerungsfrei Ports nach Windows (NT/XP/Vista) auf der MinGW- bzw. der Cygwin-Schiene. Die D-Internetpräsenz meldet zur Zeit die folgenden Plattformen:

- Linux (tested on Fedora Core 5 x86, x86_64, and PowerPC)
- Mac OS X 10.3.9, 10.4.x
- FreeBSD 6.x
- Cygwin
- MinGW
- AIX (tested on 5.1)

4.2 Verfügbare Compiler

dmd Es gibt einen D-Compiler namens **dmd** für die genannten Plattformen. Es ist der *Digital Mars* D-Compiler des D-Erfinders Walter Bright. Das Oberteil dieses Compilers (Scanner, Parser und Semantik-Analysator) wurde sowohl unter die *Artistic License* als auch unter die GNU GPL gestellt. Digital Mars bietet die Quellen für das Oberteil und die Phobos-

Bibliothek (s. 4.3) kostenlos zusammen mit dem ausführbaren Compiler (den incl. proprietärem Unterteil) und der notwendigen Umgebung dazu kostenlos zum Download an.

gdc In der aktuellen Version 0.24 (nicht 24/100, sondern Nr. 24 der 0-Serie) vom 22.08.2007 gibt es einen D-Compiler der Open Source-Gemeinde. **gdc** (GNU D-Compiler; s. D.gnu newsgroup) benutzt die Quellen des dmd-Oberteil in der Version DMD 1.020 und schließt als Unterteil den zur jeweiligen Plattform gehörigen Codegenerator der GNU C/C++-Compiler an.

Diese Technik garantiert Sprachkonformität zwischen *dmd* und *gdc*. Der Hauptaufwand auf der GNU-Seite besteht darin, den vom Oberteil erzeugten abstrakten Syntaxbaum (AST – Abstract Syntax Tree) des zu übersetzenden D-Programms auf die etwas andere AST-Vorstellung der GNUCodegeneratoren zu transformieren.

4.3 Bibliotheken

Es existieren zur Zeit zwei miteinander konkurrierende Basisbibliotheken, die Ursprungsbibliothek **Phobos** und eine neuere Bibliothek namens **Tango**. Mit anderen Worten: Wir finden das „gewohnte Bild“ vor, wie wir es beispielsweise in Java (AWT/Swing) oder Object Pascal/Delphi (VCL/VCL_Kylix/Jedi, ...) sehen. Es wäre beunruhigend, gäbe es nur eine Einheitslibrary; man wäre dann versucht, den klinischen Tod von D festzustellen. Darüber hinaus gibt es eine Bibliothek namens DFL (D Forms Library), die direkt zur Realisierung graphischer Benutzerschnittstellen dient und vor allem die Basis für das GUI-Tool Entice (s. u.) bildet.

Phobos Phobos ist die erste - von Walter Bright entwickelte und gewartete - Bibliothek für D. Sie hat Anklänge an C-Bibliotheken.

Tango Tango startete als Bibliotheksprojekt im Jahre 2006. Ziel war (und ist) die Ablösung der Phobos-Bibliothek.

Zitat (s. [9])

> “Tango is a cross-platform open-source software library, written in the D programming language for D programmers. It is structured as a cohesive and comprehensive library for general purpose usage, and is supported by a growing number of recognized D enthusiasts.
> Availability of solid and extensive documentation represents a prime factor in library accessibility, and thus this project is as much about documentation production as it is about top-notch functionality.“

4.4 Testumgebungen

Es existiert ein sehr umfangreiches Testpaket namens DStress (s. [10]). Dieses Paket wird fortlaufend anhand von Fehlermeldungen aus der Benutzergemeinde erweitert. Testcode, der den Fehler beweist, wird Teil von DStress.

Ein gewisses Maß für die momentane Größe des Testpakets ist der Rechenzeitbedarf auf einem leistungsfähigen System des Testpflegers.

Zitat aus Fragen an Thomas Kuehne und seinen Antworten (s. [10]):

> "Another problem is the processing power required to run those tests.
>
> e.g. the DMD-1.009 run contained 6801 test cases. Most of them were run in 32 different configuration ("-O", "-inline", "-O -inline", etc.) resulting in 213324 tests. In total that resulted in ca. 214000 compiler invocations, 138000 linker invocations and ca. 137500 executions of newly generated executables.
>
> How long does it take for all tests to complete? Big difference between GDC/DMD? If the testsuite is the only running application and neglection dead locks:
> ~ 25 hours for DMD, ~ 36 hours for GDC"

4.5 Entwicklungsumgebungen, GUI-Situation

Die simpelste Entwicklungsumgebung ist ein Editor, aus dem heraus der Compiler (bzw. eine Make-Einrichtung) aufgerufen und das erzeugte Programm ausgeführt werden kann. Die Anbindung des Compilers *dmd* gestaltet sich dabei sehr einfach, da sich dessen Parameterversorgung als sehr anspruchslos präsentiert.

SciTE Die Beispiele in diesem Artikel wurden mit Hilfe des Editors SciTE getestet. Mit verhältnismäßig geringen Mitteln hat dort jemand eine sehr gut arbeitende Syntaxkolorierung (Optionsdatei d.properties) beigesteuert. Die schlichten drei Zeilen

```
command.compile.*.d=dmd -O $(FileNameExt)
command.build.*.d=$(make.command)
command.go.*.d=$(FileName)
```

genügen bereits für den Anschluss der fraglichen Software, sofern die Compiler und Linker enthaltenden Verzeichnisse in die Suchpfadliste des Betriebssystems aufgenommen wurden[17] und für die Quelltexte UTF-8 eingestellt wird[18].

[17] Die SciTE-Lösung bietet für kleine Tests (nur ein Modul, kein Makefile erforderlich) nebenbei den Vorteil eines extrem schnellen Wechsels zwischen verschiedenen Programmiersprachen; der einzige „Nachteil": das Auswahlfenster mit der endlos langen Liste unterstützter Programmiersprachen passt bei kleineren Laptops nicht mehr gleichzeitig auf den Bildschirm.

[18] Ansonsten genügen bereits deutsche Umlaute in Kommentaren, um beim dmd-Compiler ärgerliche Fehlermeldungen zu provozieren.

Codeblocks Was bei den einfachen Editorlösungen fehlt, ist naturgemäß der Anschluss des Debuggers. Für Unix/Linux und Windows NT/XP/Vista gibt es dafür in Form der Entwicklungsumgebung *Codeblocks* eine Lösung, die neben Anschlüssen für praktisch alle C- und C++-Compiler auch vorgefertigte Anschlüsse an die D-Compiler dmd (von Digital Mars) und gdc (der GNU-DCompiler) enthält. Allerdings ist dazu eine der neueren Codeblocks-Versionen erforderlich. Beim Verfasser hat sich Build 4545 bewährt.

Eclipse Ein von vielen Seiten gewünschtes Eclipse-Plugin existiert noch nicht. Es gibt leider nur viele begonnene (und wieder aufgegebene) Plugin-Projekte. Im Moment wird ein neuer Anlauf mit einer offenbar realistisch niedrigen Versionsnummer 0.1 gemacht (s. [11]).

Ein mögliches Provisorium wäre vorübergehend die Nutzung des Eclipse-CDT-Plugins und dessen „Verbiegen“ in Richtung D-Compiler und -Debugger. Allerdings ist bereits die Benutzung des CDTPlugins für die C/C++-Entwicklung kein reines Vergnügen. Im Vergleich zur Justierung einer Codeblocks-Konfiguration ist allein schon die Compiler- und Debugger-Anwahl ein Martyrium.

Entice Entice ist ein GUI-Builder auf Basis der oben erwähnten **DFL** (D Forms Library). Es gibt ein Design-Panel, eine Toolbox und einen Objektinspektor (s. Abb. 4). Entice arbeitet bidirektional; Änderungen auf dem Design-Panel werden vom Objektinspektor reflektiert - und umgekehrt. Alle Designs und Änderungen daran werden in D-Quelldateien festgehalten und - zur Warnung für den Programmierer - als Spezialkommentar dargestellt. Insofern geht dieser GUI-Builder ähnlich vor wie Netbeans (von SUN) für Java. Es folgt damit ebenso wie Netbeans nicht der idealen Vorgehensweise von Borlands Delphi, bei dem Designinformationen in separaten Dateien (kein aufgeblasenes XML, sondern klar lesbares „Alt“-Objektformat von Borland) festgehalten sind.

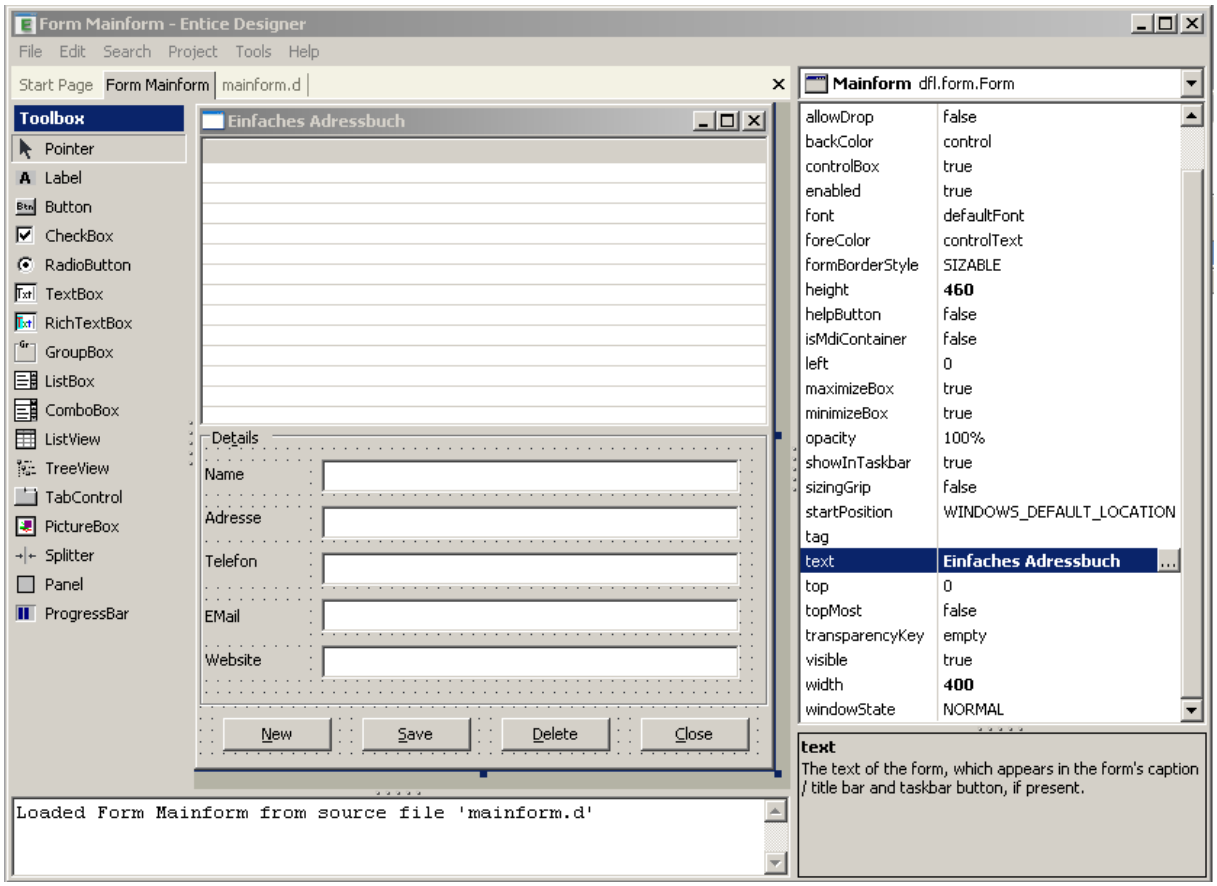


Abbildung 4: Entice-Fenster

5 Zukunftsaussichten

Für jeden, der D als Kandidaten für die Realisierung von Systemsoftware ins Kalkül zieht, stellt sich die Frage, ob D überleben wird. Solch eine Frage ist keine Besonderheit von D. Sie stellt sich bei jeder anderen Programmiersprache auch - nur nicht bis mindestens 2050 bei C, COBOL und FORTRAN.

Es ergibt sich ein ähnlicher Interessenkonflikt wie in vielen anderen Situationen, in denen Neuland betreten wird. Dabei arbeiten Leute, die aus betriebswirtschaftlichen Gründen eigentlich Konkurrenten sind, notgedrungen zusammen, um einer Idee (oder Videorecorder-Norm, DVD-Nachfolger-Norm etc.) zum Durchbruch zu verhelfen. Danach sieht man weiter. Walter Bright hat das berechtigte Interesse, mit D einmal Geld zu verdienen (wohl eher seine Erben) und im Durchbruchsfall den Ruhm zu ernten. Gleichzeitig muss er die Open-Source-Gemeinde(n) unterstützen, damit die D-Kunden ein Auffangnetz für den Fall sehen, dass er sich um D nicht mehr kümmern will (was allerdings eher unwahrscheinlich ist) oder kann.

Es ist das gleiche Dilemma, in dem sich SUN mit seinem „Kind“ Java befindet. Für ein Produkt, in das hoher Aufwand gesteckt wurde, muss die Firma die Konkurrenz ermutigen, mitzumachen und Normungsanstrengungen zu fördern - gleichzeitig aber die Konkurrenz hindern, ihr das Produkt wegzunehmen.

Es gibt hier zusätzlich das Dilemma zwischen einem altruistischen „Geschäftsmodell“ und dem „normalen“ Geschäftsmodell, bei dem Geld verdient werden muss (und hoffentlich auch verdient wird).

Im Sinne von D wäre für einige Zeit exakt ein einziges Compileroberteil ideal, nämlich das oben genannte dmd-Teil von Digital Mars. Sein Inhaber Walter Bright, der D-Erfinder, ist dank „einem sechsten Sinn“ für Programmiersprachen und Compilerfragen der Motor der D-Entwicklung[19].

Die Abhängigkeit von einem Ziehvater wirft natürlich Fragen auf, z. B. jene von Georg Wrede im Internet-Forum (s. [12]) :

„Was ist, wenn ...:

> Painting Devils on the wall
>
> What happens if Walter gets run into by a drunk, and is off-line for 6 months? Or permanently??
>
> “

Auch eine solche Fragestellung ist nicht einzigartig für D. Sie gilt analog für die „Sprachkinder“ von Yukihiro Matsumoto, Larry Wall, James Gosling und anderen. Sie galt vorübergehend auch für „Kommitee-Sprachen“. So gab es z. B. für Ada lange eine hohe Abhängigkeit vom Charisma einer einzigen Person, nämlich Jean Ichbiah.

Inzwischen sind allerdings so viele kompetente Personen in die D-Entwicklung involviert, dass ein Absturz von D nicht zu befürchten ist. Nicht zuletzt die „Adoption“ von D durch die Open-Source-Gemeinde gibt zur Zuversicht Anlass. Sie könnte die gleiche Rolle spielen wie auf kommerzieller Seite die Firma Borland, die sich um die Pflege des „Kindes“ Pascal gekümmert hat, als sein Vater Niklas Wirth in die Modula- und Oberon-Gefilde entschwand.

[19] Er ist dies insbesondere auch dank großer C++-Erfahrung aus seiner Zortech-Zeit (um den noch älteren vorhergehenden Firmennamen zu vermeiden), in der er an der Spitze der Entwicklung von C++-Compilern stand.

6 Fazit

Die Hauptfrage lautete: „Ist D ein ernstzunehmender Nachfolger für C/C++?“ Zunächst sei die Frage einmal reduziert auf C. Geht es um D anstelle von C auch noch um die Nutzung auf dem letzten kleinen Mikrocontroller, ist die Frage sicher zu verneinen. Allerdings nur, falls darauf auch ein D-Compiler laufen soll. Zu groß ist der Anpassungsaufwand, zu klein der zu erwartende Gewinn angesichts sehr begrenzter Aufgabenstellungen[20]. Ansonsten stellt sich die Vorfrage, warum statt C nicht C++ benutzt wird. Es sind allenfalls Winzigkeiten, die ein C++-Compiler an einem C-Programm auszusetzen hat. Sind es mehr als nur leicht umstellbare Winzigkeiten, dürfte es sich um „schmutzigen“ Code (z. B. fehlende Typangaben für Funktionsparameter) ältester Bauart handeln. Es erfordert dann schon einen gefährlichen Hang zum Glücksrittertum, auf ein Refactoring zu verzichten und beim C++-Compiler alles an Kontrollmechanismen abzuschalten, was sich über Compileroptionen erreichen lässt[21].

Im Grunde konzentriert sich die Frage darauf, ob D ein ernstzunehmender Nachfolger für C++ ist. Allein schon die weitaus höheren Sicherheitsvorkehrungen in D legen ein *Ja* nahe. Es genügt im Grunde bereits die Nennung des Stichworts *Präprozessor*. Bereits dessen praktisch unvermeidliche Nutzung stellt ein Sicherheitsrisiko dar. D macht dies Sicherheitsrisiko vermeidbar, ohne die C++-Welt auf den Kopf zu stellen und ohne den Systemprogrammierer in irgendwelche Ineffizienzfallen laufen zu lassen. Auf jeden Fall ist es lohnend, bei einer anstehenden Neuentwicklung (bzw. einem Refactoring) im Systembereich zumindest einen kleinen Teilaspekt herauszugreifen, von einem kleinen Team anstatt in C++ in D realisieren zu lassen und eine eigene Vergleichsbasis zu gewinnen.

Während die Spracheigenschaften von Ruby sehr viel Begeisterung auslösen (bis hin zum Schritt, damit das WEB-Handling per Ruby-on-Rails anzugehen), erregt D keine Jubelstürme. Dies ist aber eher ein gutes Zeichen. Für viele Zwecke ist als Ablösung für C++ einfach nur eine solide sichere Arbeitsumgebung notwendig, die keine Bremseffekte durch eine „vor dem System liegende“ virtuelle Maschine impliziert. Abgesehen davon: Die virtuellen Maschinen fallen nicht vom Himmel, sie müssen ja auch programmiert und gewartet werden. In irgendeiner Programmiersprache muss da wohl oder übel gearbeitet werden. Bisher war das fast immer C oder C++. Die Codes, die man zu sehen bekommt, wirken durchweg solide. Aber selbst die extrem diszipliniert in C realisierte Ruby-VM würde in D noch eine Stufe perfekter sein.

Laufzeiteffizienz und direkte Schnittstellen zum Betriebssystem (soweit ein Supervisor-Call überhaupt das Attribut *direkt* verdient) werden von D abgedeckt - wie von C/C++ allerdings auch. Insofern bietet D noch nichts Besonderes. Erst der Softwareengineering-Aspekt unter Betonung von Sicherheit, Testbarkeit, Zuverlässigkeit bringt D ins Spiel. Auch dabei hat D keinesfalls eine Alleinstellung, wie allein schon ein Blick ins „ALGOL/Pascal-Lager“ (incl. natürlich Ada) zeigt. Das Empfinden eines Ada-Apologeten könnte für ihn durchaus den Schluss nahe legen, dass C++ von Ada abgelöst werden sollte. Schließlich gab es bereits bei der Geburt von Ada die Erkenntnis, diese Sprache nicht auf C, sondern auf Pascal zu gründen. Allerdings wird dies Apologeten-Empfinden kaum wirksam werden. Dem steht die normative

[20] ... was natürlich Cross-Kompilationen mit Code-Generierung für den Winzlings-Befehlssatz nicht ausschließt.
[21] Auch C-Compiler lassen solch einen Code schon seit langem nicht mehr ohne rotglühende Warnlampen durchgehen.

Kraft des Faktischen entgegen. Und die Fakten werden durch C/C++ als „Muttersprache“ für Betriebssysteme wie Unix/Linux und Windows und für die meisten Datenbanksysteme gesetzt. Und diese Fakten zeigen auf D als ernsthaften Ablösungskandidaten und nicht auf ein Pascal-Derivat.

Laufend geführte Analyseinstrumente wie in [4] angegeben spiegeln einen gewissen D-Trend bereits wider. Spitzenreiter in jenem auch die Anwendungsprogrammierung anzeigenden „Barometer“ ist Java, dicht gefolgt von C. C++ verliert in starkem Masse Anhänger. Und D gewinnt am meisten – wenn man von der für die Systemprogrammierung kaum relevanten Phalanx der dynamisch typisierten Sprachen Ruby, PHP, ..., LUA und von Visual Basic absieht.

Literaturverzeichnis

[1] Alexander Klinsky: D Programming Language.
http://www.prowiki.org/upload/HelmutLeitner/D_Programming_Language_030418.pdf;
Schnappschuss der offziellen Online-Documente aus
http://digitalmars.com/d/index.html

[2] Kernighan, B. W., D. Ritchie:Programmieren in C, 2. Auflage, Hanser, München, Wien (1990).

[3] Stroustrup, B.: The Design and Evolution of C++, Addison-Wesley, Bonn (1994).

[4] TIOBE Programming Community Index for February 2008,
http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html

[5] http://www.ironruby.net/

[6] Thomas, David, A. Hunt: Programmieren mit Ruby. Addison-Wesley (2002).

[7] Beck, Kent: eXtreme Programming Explained. Addison-Wesley (2000).

[8] www.cs.berkeley.edu/~wkahan/JAVAhurt.pdf

[9] http://www.dsource.org/projects/tango

[10] http://dstress.kuehne.cn

[11] http://www.digitalmars.com/webnews/newsgroups.php?art_group=
digitalmars.D.announce&article_id=9617

[12] http://www.digitalmars.com/d/archives/digitalmars/D/index.html:

Geschäftsprozesse und IT – eine Bestandsaufnahme

Mathias Groß, Werner Hülsbusch

„Informationstechnologien zur Optimierung von Geschäftsprozessen“ – das ist ein komplexes Thema, das wir in diesem Übersichtsbeitrag weder in seiner Breite noch in seiner Tiefe umfassend behandeln können. Wir können lediglich einige Aspekte und Zusammenhänge herausgreifen, die uns grundsätzlich wichtig erscheinen. Bei weitergehenden Interessen bieten die Literaturhinweise Möglichkeiten für einen vertieften Einstieg in die Materie.

Dieser Beitrag behandelt die Themenfelder:

- Geschäftsprozesse und ihre Bedeutung für das e-Business
- Information und IT-Unterstützung in Geschäftsprozessen – heute und morgen

1 Was sind Geschäftsprozesse?

Seit zwei Jahrzehnten finden prozessorientierte Methoden der Reorganisation großes Interesse sowohl in der betriebswirtschaftlichen Fachliteratur als auch in der Unternehmenspraxis. Jenseits aller Unterschiede im Detail lässt sich als gemeinsames Ziel dieser Ansätze das Streben nach effizienteren Abläufen identifizieren, d. h. die unternehmensinternen und -übergreifenden Prozesse sollen kostengünstiger, schneller und fehlerfreier werden.

In der Literatur hat sich bislang noch keine einheitliche Definition für den Begriff des Geschäftsprozesses herausgebildet. Die Wirtschaftsinformatik versteht unter einem Geschäftsprozess meist die inhaltlich abgeschlossene, zeitlich-sachlogische Abfolge von Funktionen, die zur Bearbeitung eines für die Leistungserbringung des Unternehmens relevanten Objekts erforderlich sind. [Schmelzer/Sesselmann 2004, S. 46] definieren etwa:

> „Geschäftsprozesse sind funktionsübergreifende Verkettungen wertschöpfender Aktivitäten, die von Kunden erwartete Leistungen erzeugen und deren Ergebnisse strategische Bedeutung für das Unternehmen haben. Sie können sich über das Unternehmen hinaus erstrecken und Aktivitäten von Kunden, Lieferanten und Partnern einbinden.“

Geschäftsprozesse bestehen aus Sub-Prozessen, diese wiederum aus Aktivitäten (Abb. 1). Als „Aktivität“ ist dabei eine Arbeitseinheit zu verstehen, die von einer Person in mehreren Arbeitsschritten durchgeführt wird, jedoch als Einzelleistung keinen Wert für den Kunden darstellt. Auf der Mikro-Ebene schließt sich dann das Workflow-Management an, das die technische Unterstützung der operativen Ausführung von Prozessen beinhaltet [Müller 2005].

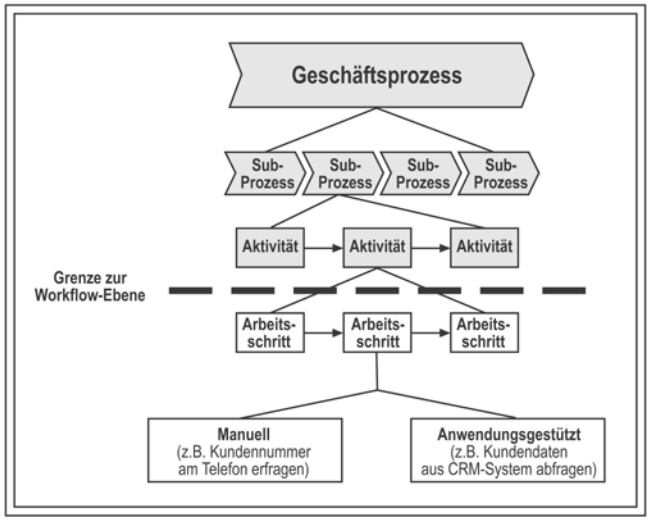


Abb. 1
Vom Geschäftsprozess zum Workflow
[Quelle: in Anlehnung an http://www.ec-management.de]

Geschäftsprozesse haben einen Beginn und ein Ende sowie klar definierte In- und Outputwerte. Die Effizienz von Geschäftsprozessen ist messbar in den Kategorien Kosten, Service und Qualität. Die Gesamtheit der Geschäftsprozesse eines Unternehmens weisen eine hierarchische Struktur auf und ihre Abhängigkeiten lassen sich in einer sogenannten Prozesslandkarte darstellen.

Geschäftsprozesse laufen meist durch mehrere Bereiche/Abteilungen (Abb. 2). Der Geschäftsprozess-Begriff impliziert damit – in praxeologischer Perspektive – vor allem eine bestimmte Betrachtungsweise des Unternehmens, in der nicht die einzelnen vertikalen Funktionen (Aufbauorganisation), sondern der gesamte horizontale Ablauf der Prozesse (Ablauforganisation) im Vordergrund steht [Gaitanides/Ackermann 2004]. Diese Betrachtungsweise der „prozessorientierten Organisation“ begünstigt, das Ziel der Kundenzufriedenheit stets im Blickfeld zu behalten, da Geschäftsprozesse notwendigerweise beim Kunden beginnen (Anforderungen, Erwartungen, Auftrag) und wieder beim Kunden enden (Produkt, Dienstleistung).

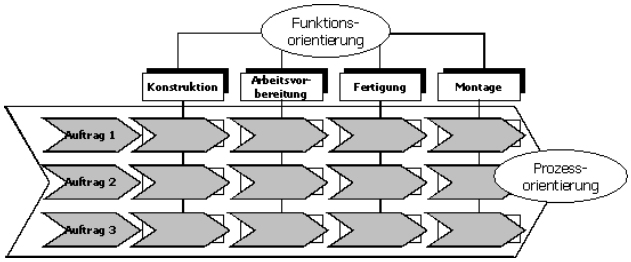


Abb. 2
Unterschied zwischen einer funktions- und prozessorientierten Organisationsgestaltung
[Quelle: Gaitanides/Ackermann 2004]

Geschäftsprozesse können nach unterschiedlichen Kriterien klassifiziert werden, etwa nach der Strukturiertheit oder der Art und Häufigkeit des Auftretens. Wichtig ist auch die Unterscheidung in kundenorientierte Kernprozesse und sie unterstützende Supportprozesse. Ein Kernprozess ist gekennzeichnet durch wahrnehmbaren Kundennutzen – idealerweise (im Sinne der Erzielung und Erhaltung von Wettbewerbsvorteilen) zusätzlich durch Nicht-Imitierbarkeit, Nicht-Substituierbarkeit und Spezifität. Supportprozesse unterstützen die Kernprozesse durch Bereitstellung einer „Infrastruktur“ und stellen keinen unmittelbaren, „sichtbaren“ Kundenvorteil dar. Sie sichern den reibungslosen Ablauf der Geschäftstätigkeit und stehen dabei quasi „unsichtbar“ hinter den Kernprozessen. Geschäftsprozesse, die es in fast jedem Unternehmen gibt, sind beispielsweise der Materialbereitstellungs-, der Produktentwicklungs-, der Qualitätssicherungs- oder der Beschwerdebearbeitungsprozess.

Der Kern der prozessorientierten Organisationsgestaltung liegt in der Festlegung der Prozessstruktur. Diese umfasst zum Beispiel die zeitliche Reihenfolge der Teilprozesse, Art und Methoden der Arbeitsverrichtung, den Ressourceneinsatz etc. Abgestimmt auf die Prozessstruktur werden dann die Organisationseinheiten gebildet und strukturiert.

Das Hauptaugenmerk liegt dabei auf einer durchgängigen Kundenorientierung, der Schaffung einer größtmöglichen Einfachheit des Auftragsdurchlaufes, der Gestaltung klarer, übersichtlicher Wege sowie der Minimierung aufbauorganisatorischer Schnittstellen. Schnittstellen sollen reduziert werden, um die mit ihnen verbundenen Nachteile und Gefahren zu minimieren (Zielkonflikte der verschiedenen Abteilungen, erhöhter Koordinationsaufwand, Kontrollaufwand, Verlängerung der Durchlaufzeiten, mögliche Störungen der Material- und Informationsflüsse, ...). Ferner wird jedem Prozess ein Prozessverantwortlicher („process owner“) zugeordnet, um Unklarheiten der Zuständigkeit zu vermeiden. Zwischen den „process ownern“ werden Leistungsniveaus („service level agreements“) ausgehandelt.

2 Geschäftsprozessorientierung im Wandel

Lange Zeit beschäftigte man sich in der BWL wie in der betrieblichen Praxis hinsichtlich organisatorischer Gestaltungsprobleme vornehmlich mit der *Aufbau*organisation. Dies beförderte mangelnde Kundenorientierung, Flexibilität und Schlagkraft am Markt.

Deshalb setzte eine Gegenbewegung ein. Die *ablauf*organisatorische Dimension und damit die *Prozessorientierung* rückten in den Vordergrund: das Unternehmen wurde als Ort der Verknüpfung von Flüssen und Prozessen betrachtet. Das bedeutet, dass Aktivitäten, Arbeitsschritte und ihre Reihenfolge unabhängig von den aufbauorganisatorischen Gegebenheiten zu modellieren und Stellen erst auf der Basis integrierter Verrichtungskomplexe zu bilden sind. Anstelle der Logik „Ablauforganisation folgt Aufbauorganisation“ etablierte sich die Maxime „Aufbauorganisation folgt Ablauforganisation“.

Dieser Gedanke des Prozessmanagements ist keineswegs neu: wichtige frühe Arbeiten zu diesem Thema wurden in den 80er Jahren im deutschsprachigen Raum etwa von Michael Gaitanides und Wilhelm-August Scheer veröffentlicht. Auch in der betrieblichen Praxis fasste die rechnergestützte Abbildung von Geschäftsprozessen seit Beginn der 80er Jahre Fuß, wenn auch zunächst nur in Großunternehmen und global agierenden Konzernen, etwa der Automobilindustrie. Der Walldorfer Software-Hersteller *SAP* war mit seinem seit 1979 entwickelten System *R/2* hier sicherlich ein Vorreiter – es ermöglichte den Unternehmen erstmals, ihre internen Abläufe, insbesondere in den Bereichen Rechnungswesen, Logistik und Personal, abteilungsübergreifend auf einer einheitlichen Datenbasis in einem integrierten System abbilden zu können. Die Analysten von *Gartner* prägten für diese Softwareklasse 1990 den Begriff „Enterprise Resource Planning“ (ERP).

Seitdem hat die Informations- und Kommunikationstechnologie einen wahren Siegeszug angetreten, der u. a. durch folgende Aspekte schlagwortartig charakterisiert werden kann [Picot/Neuburger 2000]:

- Miniaturisierung
- drastische und fortlaufende Verbesserungen des Preis-/Leistungsverhältnisses
- Standardisierung
- Integration
- Konvergenz im Informations-, Telekommunikations- und Medienbereich
- Aufkommen mobiler Techniken und Anwendungen
- wachsende Akzeptanz und Nutzung bei den „usern“
- umfassenden Vernetzung

Dieser weltweite Ausbau der Netzinfrastruktur und die verbesserten Zugangstechnologien sind vielleicht das herausragendste Merkmal der neueren Entwicklung. Diese und andere Aspekte bilden die Basis für eine immer stärkere Durchdringung der Wirtschaft mit moderner *Informations- und Kommunikationstechnik (IKT)*, insbesondere dem Internet. Sie eröffnen für Nachfrager und Anbieter neuartige, kostengünstigere und effizientere Möglichkeiten, Informationen über Märkte und Unternehmensprozesse zu erhalten, sie versprechen eine Reduktion der fixen und variablen Transaktionskosten, sie erlauben neue Formen der Zusammenarbeit und Arbeitsteilung, neue Möglichkeiten der Prozessabwicklung sowie neue Formen der Arbeit und Zusammenarbeit in und zwischen Unternehmen.

Damit haben wir bereits einige Merkmale des *e-Business* (eB) beschrieben, worunter wir in Übereinstimmung mit [Wirtz 2001] die Anbahnung sowie die teilweise respektive vollständige Unterstützung, Abwicklung und Aufrechterhaltung von Leistungsaustauschprozessen mittels elektronischer Netze fassen können. Das eB hat sich parallel zum Siegeszug moderner IKT seit Ende der 90er Jahre herausgebildet und bekanntlich schon mehrere Entwicklungsphasen durchlaufen – von der Euphorie über die totale Ernüchterung bis zur heutigen Phase einer zunehmend mit Bedacht angegangenen und mit angepassten Geschäftsmodellen abgesicherten Überführung in ein „real business“ (Abb. 3).

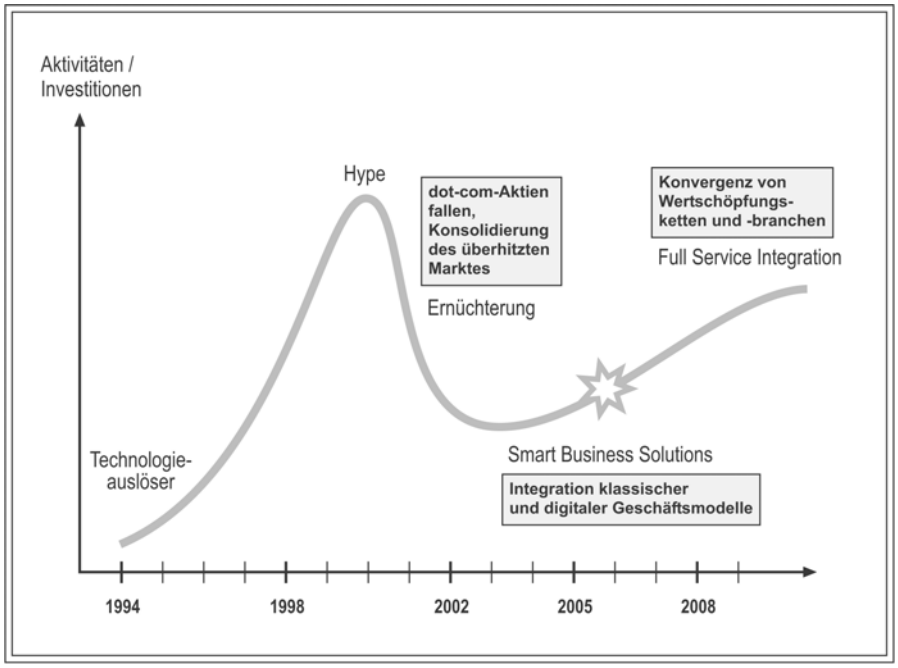


Abb. 3 eB-„Hype-Kurve“ (in Anlehnung an *Gartner Group*)

Im Zuge dieser beiden sich gegenseitig bedingenden und unterstützenden Entwicklungen wurde das Geschäftsprozess-Konzept (zusammen mit der notwendigen Diskussion um realistische Geschäfts*modelle*) zwischenzeitlich eher vernachlässigt, erfuhr jedoch nach dem Abklingen des eB-Hypes eine Renaissance. Denn so manches eB-Projekt ist nicht zuletzt daran gescheitert, dass die Prozesse hinter den schillernden Fassaden der Webseiten nicht oder nicht optimal funktionierten.

Die Geschäftsprozess-Diskussion nahm in den letzten Jahren in immer kürzeren Abständen verschiedene Ausprägungen an, die hier – keineswegs vollständig – nur kurz aufgelistet werden können [Binner 2004, S. 51–97; Kagermann/Zencke 2005]:

- *Lean Management (LM):* Nach diesem Konzept organisiert man einfacher und schlanker, also mit weniger Schnittstellen. Zentrale Komponente ist neben dem kontinuierlichen Verbesserungsprozess (japanisch: „Kaizen“) die Qualitätskultur, die Erweiterung des Technologiemanagements durch das Konzept der Kernkompetenzen und der Kun-

denorientierung mit strikter Ausrichtung an Kundenbedürfnissen sowie die strenge Prozessorientierung.

- *Total Quality Management (TQM):* Die Vertreter dieses Konzepts rücken die Prozess-, Kunden- und Mitarbeiterorientierung als grundlegende Faktoren für die Qualität in den Vordergrund. Hauptziel ist die Verbesserung der Qualitätsfähigkeit von technischen und administrativen Prozessen. Mit ISO 9001:2000 existiert eine Qualitätsnorm zur Umsetzung der geforderten Prozessorientierung.
- *Business Reengineering (BR)* bzw. *Business Process Reengineering (BPR):* Die Weiterentwicklung des Lean Managements zum abteilungsübergreifenden BPR-Konzept brachte den eigentlichen Durchbruch des Prozessmanagements (mit den Bestandteilen Prozessanalyse, -modellierung, -integration, -monitoring). Während die Geschäftsprozessoptimierung im Sinne des „Continuous Process Improvement" (CPI) eher auf Prozessanalyse, Partizipation und inkrementelle Verbesserungen der bestehenden Organisation setzt, verlangt BPR radikale Vereinfachungen des Leitbildes, von Prozessen und Strukturen mit dem Ziel der Kundenzufriedenheit. Allerdings berichteten BPR-Studien von „Flop-Raten" bis zu 75 % – erst mit dem einsetzenden Markterfolg der ERP-Systeme wurde die BPR-Philosophie mit einem praxistauglichen Werkzeug untermauert.
- *Supply Chain Management (SCM):* Auch im SCM-Konzept spielt das Prozessdenken eine wichtige Rolle. Nachdem die unternehmensinternen Rationalisierungpotentiale (von Manufacturing Resource Planning/MRP bis zum ERP) weitgehend ausgeschöpft sind und der Wettbewerb verstärkt zwischen gesamten Wertschöpfungsketten („Supply Chains") stattfindet, gilt es, die gesamte, unternehmensübergreifende Wertkette zu betrachten und zu optimieren.
- *Neue Lösungsansätze in der Logistik:* Die Logistik hat wesentlichen Anteil daran, dass den Kunden Produkte mit kundengerechter Funktionalität und Qualität zu marktgerechten Preisen und zum richtigen Zeitpunkt angeboten werden können. Effiziente Logistikketten werden immer stärker zu einem entscheidenden Differenzierungsfaktor im Wettbewerb, daher sind verschiedene neue Lösungsansätze im Bereich der Logistik entwickelt worden: Efficient Replenishment (ER), Vendor Managed Inventory (VMI), Countinuous Planing Forecasting & Replenishment (CPFR), Just-in-Time (JiT), Quick Response (QR) und Efficient Consumer Response (ECR) sind hier einige Schlagworte.

Egal, in welcher Ausprägung, mit welchem (Mode-) Begriff belegt und ohne in Euphorie zu verfallen: Konsequentes Geschäftsprozessmanagement erlaubt es, Unternehmen insgesamt zielorientierter zu steuern, effizienter zu organisieren und laufend zu verbessern. Die Strategien der Umsetzung eines prozessorientiert geführten Unternehmens fokussieren – bei allen individuellen Unterschieden – grundsätzlich auf Kundenorientierung, Produktqualität und Zeitoptimierung. Prozessmanagement, Organisation, Qualität und Informationstechnik (IT) hängen dabei eng zusammen, wobei das Prozessmanagement eine zentrale Rolle einnimmt [Herterich 2005].

3 Information und Informationstechnik in Geschäftsprozessen

Nachdem Information und Informationstechnik in den letzten Abschnitten eher implizit bei der Beschreibung der Prozessmanagement-Philosophie zum Tragen kamen, soll deren Rolle in den folgenden Abschnitten explizit behandelt werden.

Ganz allgemein muss heute *Information* (pragmatisch verstanden, im Sinne von „zweckorientiertem Wissen" oder „Wissen in Aktion") als ein zunehmend wichtiger Faktor für das Funk-

tionieren von Organisationen (wie Unternehmungen und Verwaltungen) und sozio-ökonomischen Gesamtheiten (Volkswirtschaften, Weltwirtschaft) angesehen werden. Information ist notwendig für das produktive Zusammenwirken der klassischen Ressourcen oder Produktionsfaktoren Arbeit, Boden und Kapital – sie ist eine zunehmend wichtige unternehmerische Ressource von strategischer Bedeutung, ein kritischer Erfolgsfaktor (Abb. 4) für die Sicherung der Wettbewerbsfähigkeit und damit der mittel- und langfristigen Marktstellung eines Unternehmens [Picot/Scheuble 1997; Weiber/McLachlan 2000]. Ob sie damit wirklich als „vierter Produktionsfaktor" im Gutenberg'schen Faktorensystem aufgefasst werden kann, ist allerdings wissenschaftlich umstritten [Seidenberg 1998].

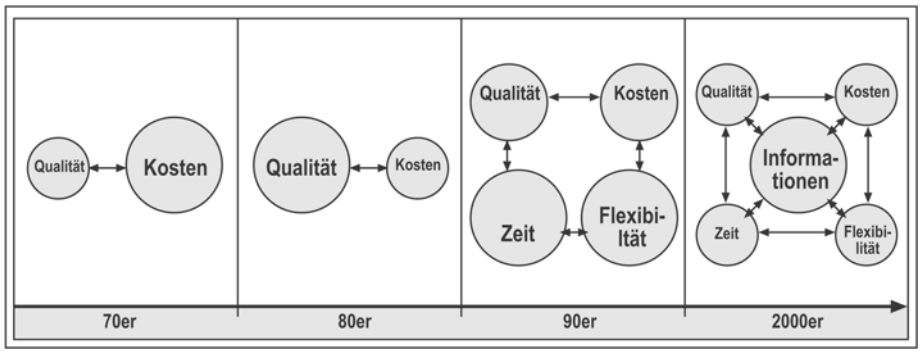


Abb. 4 Bedeutungsveränderung kritischer Erfolgsfaktoren im Zeitablauf
[Quelle: in Anlehnung an Weiber/McLachlan 2000, S. 125]

Jedenfalls gewinnen ökonomische Aspekte von Information und Kommunikation an Bedeutung, und es hat sich dementsprechend eine „Informationsökonomie" als eigenes Forschungsgebiet herausgebildet. Im Makro-Bereich bringt die Entwicklung zur Informations- oder Wissensgesellschaft neue Herausforderungen in ökonomischer, sozialer und ordnungspolitischer Hinsicht hervor – und man kann wohl mit Berechtigung von einem „fünften Kondratieff" mit der Informationstechnologie als zugrunde liegender Basisinnovation sprechen (Abb. 5) [Nefiodow 1990].[1]

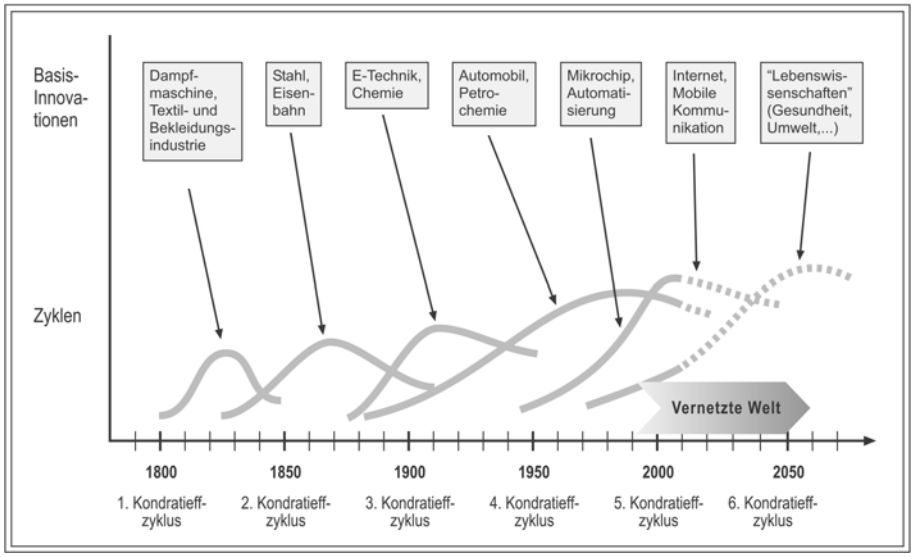


Abb. 5 Die Informations- bzw. Wissensgesellschaft als 5. Kondratieff

[1] Nikolai Kondratieff, ein 1892 geborener russischer Volkswirtschaftler, war Kommunist und gleichzeitig Marktwirtschaftler. Während Lenins Neuer Ökonomischer Politik war er Leiter eines Konjunkturinstituts in Moskau und veröffentlichte 1926 mit seiner Arbeit „Die langen Wellen der Konjunktur" die Theorie der jeweils ca. 50 Jahre andauernden Wirtschaftszyklen, die von bahnbrechenden Erfindungen eingeleitet werden. Als ab 1928 Lenins NÖP durch die Planwirtschaft ersetzt wurde, wurde er von der Institutsleitung entbunden und fand unter Stalin 1938 einen gewaltsamen Tod.

Offensichtlich werden auch und gerade bei der Durchführung von Geschäftsprozessen orientierende, planende und koordinierende Informationen benötigt, erzeugt, verarbeitet, gespeichert und weitergeleitet. Nur wenn diese Informationen bestimmten Qualitätskriterien genügen, können Prozesse erfolgreich generiert, strukturiert und beherrscht werden, so dass der Prozess-Output hinsichtlich Zeit, Kosten und Qualität den Anforderungen entspricht.

Die Gewinnung, Verarbeitung und Interpretation von Information geschieht heute weitgehend durch und mit Unterstützung von IKT. Deshalb ist ein typischer inhaltlicher Schwerpunkt von Beiträgen zur Prozessorganisation die Betonung der Rolle der IT bzw. IKT als *Katalysator* bei der Optimierung von Geschäftsprozessen. Die Integration von IKT in organisationsinterne und -übergreifende Geschäftsprozesse verspricht Effizienzsteigerungen und Kostensenkungen; Kundenorientierung und Rundumbearbeitung verlangen insbesondere dezentralen Datenzugriff. IT wird daher auch als *enabler* („enabler of process innovation“) begriffen: IT ermöglicht es erst, integrierte Geschäftsprozesse zu entwickeln und ganzheitliche Vorgangsbearbeitung zu realisieren. Ihr kommt daher in doppelter Hinsicht eine besondere Bedeutung zu:

- bei der *Gestaltung* (dem Entwurf, dem Management) und
- bei der effizienten technischen *Umsetzung* von Geschäftsprozessen.

4 IT in der Umsetzung von Geschäftsprozessen – heute und morgen

Wenn wir hier eB – etwas verkürzt – als die umfassende Integration von IKT in unternehmensinterne und -übergreifende Geschäftsprozesse auffassen, so ergibt sich damit von selbst, dass IKT eine zunehmend wichtige Rolle in den Geschäftsprozessen des eB-Zeitalters spielt. IKT kann dabei verschiedene Rollen übernehmen, die man zunächst ganz grob folgendermaßen einteilen kann:

- Unterstützung bestehender Prozesse (substitutiver Einsatz) und
- Ermöglichung innovativer Prozesse (innovativer Einsatz).

Zunächst kann IKT zur Unterstützung bereits bestehender Prozesse zur Steigerung der Effizienz gewinnbringend eingesetzt werden – vor allem dann, wenn es um die Rationalisierung weitestgehend standardisierter und gleichförmiger Prozesse geht. Zu denken ist hier etwa an Formen multimedialer Unterstützung bestehender arbeitsorganisatorischer Prozesse (z. B. Büromaterial-Bestellung), an die Unterstützung interner oder externer Kooperation (z. B. Groupware oder Videokonferenzen) oder an die Unterstützung klassischer Werbe- und Vertriebsprozesse (z. B. elektronische Produktkataloge). Die Effizienz-Vorteile sind hier meist offensichtlich (vgl. Abb. 6 a,b am Beispiel des e-Procurement, also der elektronischen Beschaffung).

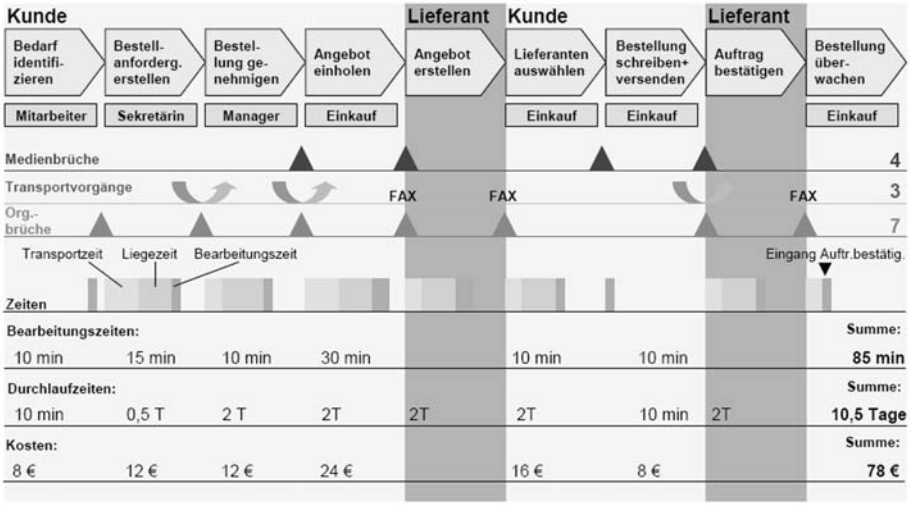


Abb. 6 a Traditionelle Bestellung von Büromaterial [Quelle: Allweyer 2000]

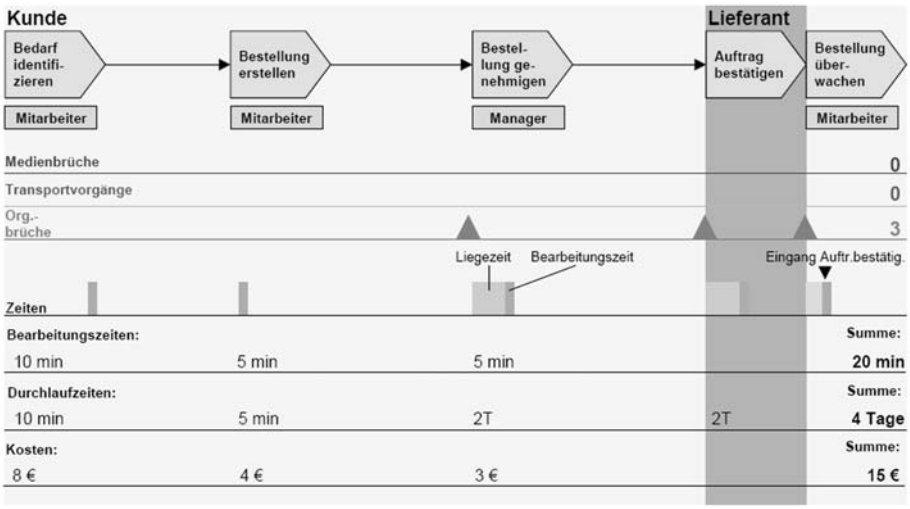


Abb. 6 b e-Procurement: Bestellung von Büromaterial über elektronischen Marktplatz und erzielte Effizienzgewinne [Quelle: Allweyer 2000]

Da jede Aktivität innerhalb der Wertschöpfungskette Informationen verwendet und hervorbringt, verwundert es nicht, moderne IKT heute auf allen Wertschöpfungsstufen des Unternehmens vorzufinden. Abb. 7 verdeutlicht exemplarisch die Vielfalt der zum Einsatz kommenden IT-Systeme.

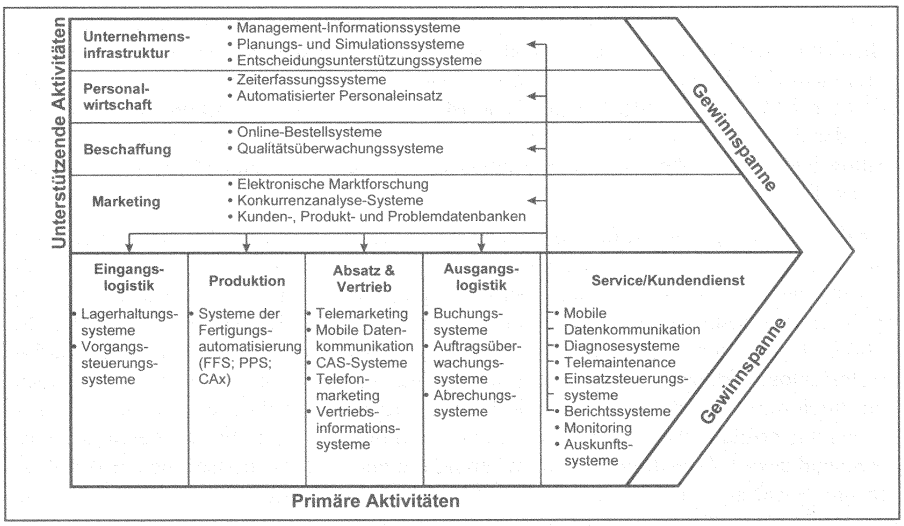


Abb. 7 IT-Systeme im Wertschöpfungsprozess einer Unternehmung (entlang der Porter'schen Wertschöpfungskette) [Quelle: Weiber/McLachlan 2000, S. 128]

Der IKT-Einsatz beschränkt sich nicht auf die Unterstützung bestehender Prozesse und hat auch nicht unbedingt zwangsweise bestimmte organisatorische Konsequenzen zur Folge, sondern eröffnet im Gegenteil erhebliche *Freiheitsgrade* für die organisatorische Gestaltung von Unternehmen. IKT kann daher auch grundlegend neue organisatorische Gestaltungsmöglichkeiten eröffnen, wobei zwei grundlegende Effekte beobachtet werden können: Modularisierung und Prozessorientierung.

IKT macht vor allem dann Sinn, wenn sie aus einer prozessorientierten Sicht eingesetzt wird, da sie gerade die Integration abteilungs- und unternehmsübergreifender Abläufe ermöglicht. Daher hat sich der IKT-Einsatz im Unternehmen in den letzten Jahren deutlich verschoben (Abb. 8): von der Unterstützung einzelner Aufgaben und Funktionsbereichen über die Unterstützung integrierter Unternehmensprozesse (Intranets, unternehmensinternes SCM) hin zu einer weitergehenden Vernetzung über Unternehmensgrenzen hinweg (unternehmensübergreifendes SCM, virtuelle Unternehmen, „Net Economy“).

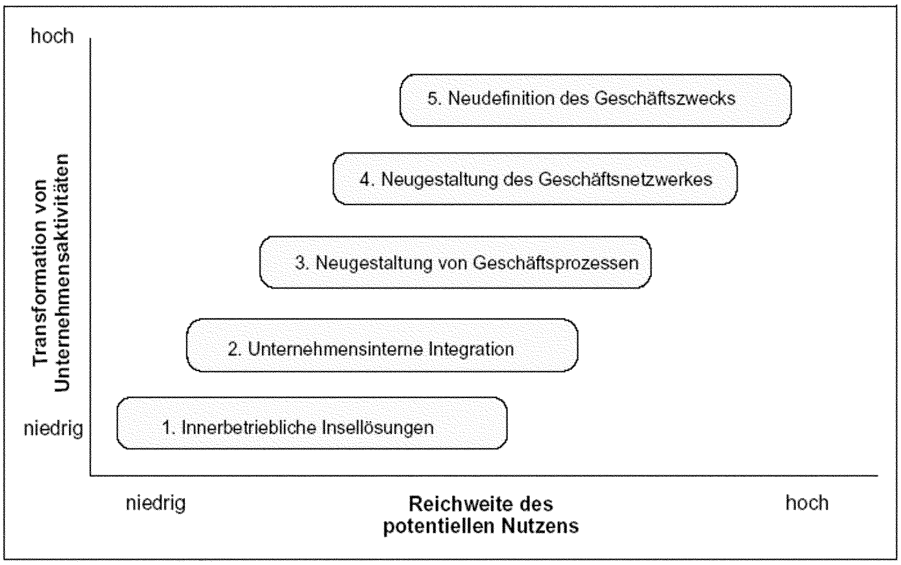


Abb. 8 Entwicklung der IT-Nutzung in Unternehmen [Quelle: Mertens/Faisst 1995]

Eine kleine Auswahl von Techniktrends soll die Betrachtung der IKT in der Umsetzung von Geschäftsprozessen abschließen. Gerade aus der Sicht der zunehmend organisationsübergreifenden Kommunikationsprozesse spielen allgemein die Forderungen nach offenen Systemen, Standards, Client-Server-Konzepten (erweitert um Web Services, s. u.), Open-Source-Software sowie des „Computing aus der Steckdose“ (im Fachjargon „On-demand Computing“ oder „Utility Computing“ genannt) weiterhin eine große Rolle. Einige weitere spezielle Trends und „buzzwords“ lauten:

- *Weitere Integration und Flexibilisierung, etwa durch EAI:* Das „klassische“ ERP wurde schnell um weitere Konzepte wie SCM oder Customer Relationship Management/ CRM (zur Organisation von Kundenbeziehungen) erweitert. Solche Erweiterungen in heterogenen Systemen mit dezentralen Daten und Prozessen um neue Komponenten und Anwendungen werfen verstärkt die Frage nach der Problematik der Systemintegration auf. Ein effizientes Zusammenwirken beispielsweise von ERP-, CRM- und SCM-Konzepten wird daher mit dem Enterprise-Application-Integration-Ansatz (EAI) angestrebt (Abb. 9). EAI scheint ein Schlüsselfaktor für die weitere Integration und Flexibilisierung zu sein. Der Zugang zu diesem komplexen System kann durch ein (Prozess- und Service-) Portal geschaffen werden, durch das jeder User Zugang nicht nur zu den benötigten Daten, sondern auch Prozessen und Services erhält [Grimm 2004].

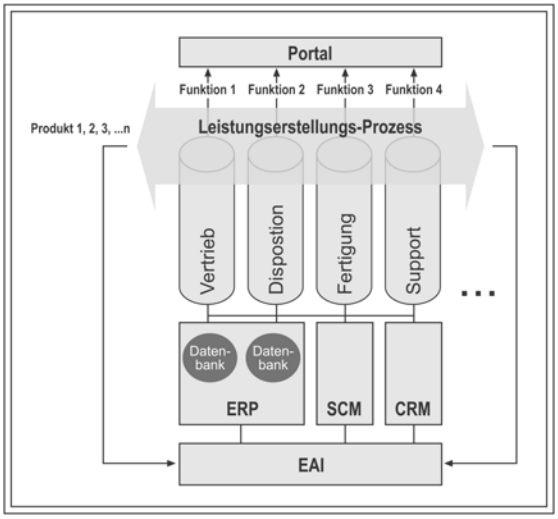


Abb. 9
Von ERP zu EAI
[Quelle: nach http://www.ec-management.de]

- *BPMS und Web Services:* Ein Stück weiter als EAI- gehen BPM- (Business Process Management) Systeme, die zusätzlich Elemente von Workflow-Management-Systemen beinhalten und so die Ablaufsteuerung der Prozesse übernehmen. Der BPMS-Ansatz in Verbindung mit Web Services ermöglicht gänzlich neue, nämlich serviceorientierte Architekturen (SOA) [Hofmann 2003]. Wenn man von Computer-Fachchinesisch und Implementierungsdetails weitgehend absieht, kann man Web Services als wohldefinierte Funktionen auffassen, welche über standardisierte Protokolle auf entfernten Rechnern in offenen Netzen zur Ausführung von Business-Funktionen oder Teilen davon angeboten werden. BPMS rufen die Services der einzelnen Komponenten auf und setzen diese quasi zu einem Gesamtprozess zusammen (man spricht auch von „Orchestrierung"). Heute dominieren zwei „Lager": die plattform-unabhängige Lösung *Sun ONE* (*O*pen *N*etwork *E*nvironment) auf der Basis *J2EE* (*Java 2 Enterprise Edition*) sowie das proprietäre *.NET* von *Microsoft*. Das geplante Haupteinsatzgebiet liegt im B2B-Bereich, etwa im Rahmen von EAI-Software: Geschäftsprozesse sollen problemlos über Unternehmensgrenzen hinweg abgewickelt werden können. Ähnliches gilt für den Einsatz von Web Services als technischer Infrastruktur im e-Government, wo sie es ermöglichen werden, Leistungen und Funktionen unter Wahrung föderaler Strukturen und heterogener Systeme Plattform- und Programmiersprachen-unabhängig z. B. anderen Behörden zugänglich zu machen und so als „Prozessdrehscheibe" oder „Prozessnetzwerk" zu fungieren [Spahni/Meir 2003].

- *Business Intelligence:* Der noch sehr uneinheitlich verwendete Begriff „Business Intelligence" (BI) ist kein neues Produkt oder Konzept, sondern eher eine begriffliche „Klammer", die allgemein die Methoden, Prozesse und Technologien umfasst, um Unternehmens- und Wettbewerbsinformationen zu sammeln, aufzubereiten, zu analysieren und weiterzuleiten und damit Entscheidungsträgern eine verbesserte Unterstützung zu geben. Dazu werden unternehmensinterne und -externe, wohlstrukturierte und schwach strukturierte Daten als Quellen herangezogen. Die Schnittstellen zum Benutzer bilden Informations-, Entscheidungsunterstützungs- sowie Kommunikations- und Koordinationsunterstützungssysteme. Insofern bei einem vornehmlich prozessorientierten Verständnis eine kontinuierliche Anpassung der Datenbasis, der Methoden und Werkzeuge an die Informationsstrategie notwendig ist („strategic alignment"), wird auch bei diesem Ansatz der Schwerpunkt auf die – in diesem Fall informationsorientierte – Prozessgestaltung gelegt. Moderne BI-Werkzeuge beschränken sich nicht auf eine rein vergangenheitsbezogene Analyse von Geschäftsdaten (aufbauend auf Data Warehouses, Online Analytical Processing/OLAP sowie Data Mining), sondern integrieren z. B. auch Planungsunterstützung/Forecasting und Performance Management [Alff/Bungert 2004].

- *Semantic Web:* Die aktuellen Vorstellungen eines „intelligenten“ Semantic Web leiten sich aus langjährigen Forschungsarbeiten unterschiedlichster Disziplinen ab (Künstliche Intelligenz, Wissensmanagement, Kognitionsforschung etc.). Mit der wachsenden Datenflut sowie der zunehmenden Komplexität von Geschäfts- bzw. Verwaltungsprozessen sind heute Fragen nach Effizienzgewinnen durch semantische Technologien hoch aktuell. Das Semantic Web, wie wir es heute verstehen, ist eine Erweiterung des WWW um maschinenlesbare Daten, welche die Semantik der Inhalte formal festlegen. Unter Nutzung dieser Technik können Informationsrecherche-, -verarbeitungs- und -verwaltungsprozesse (teil-) automatisiert werden, etwa durch Agententechnologien. Konkret wäre es z. B. denkbar, dass eine „Suchmaschine“ im „SemWeb“ nicht nur (mehr oder weniger passende) „Treffer“ bietet, also auf möglicherweise „passende“ Dokumente verweist, sondern Anfragen der Art „Wo und wann kann ich einen alten Monitor entsorgen?“ *direkt* beantworten kann. Dies würde völlig neue Möglichkeiten des Wissensmanagements für Unternehmen und Verwaltungen eröffnen.

- *Mobile Geschäftsprozesse (Mobile Business Processes):* Mobile Anwendungen liegen wegen der großen Freiheitsgrade aufgrund der Ortsunabhängigkeit, Lokalisierbarkeit, Erreichbarkeit und Personalisierbarkeit im Trend (vor allem die jüngere Generation wird zu „nomadic usern“) – und es stellt sich die Frage, ob nicht alle Anwendungen früher oder später auch mobil ausgeführt werden können [Lehner/Meier/Stormer 2005]. Bei mobilen Anwendungen geht es nicht nur um die simple Möglichkeit zum mobilen Versand und Empfang von E-Mail oder SMS, sondern weitergehend etwa um die vollständige Einbindung mobiler Arbeitsplätze in inner- und zwischenbetriebliche Prozesse [Khodawandi/Pousttchi/Winnewisser 2003; Köhler/Gruhn 2004] oder die Automatisierung einzelner Teilprozesse durch den Einsatz mobiler Kommunikationstechnologien (GPS, WAP, WLAN, UMTS, etc.) und mobiler Endgeräte (UMTS-Handys, PDAs, etc.). „Mobile Government“ kann hier einen hohen Stellenwert einnehmen und innovative Lösungen bieten – für Verwaltungen liegt der Mehrwert von m-Government in der Möglichkeit zur weiteren Optimierung und Straffung von Geschäftsprozessen (z. B. Unterstützung von Verwaltungsmitarbeitern bei mobilen Kontroll- und Regulierungsaufgaben), für die Bürger in der Möglichkeit zur Bereitstellung innovativer, situationsbezogener Dienste (z. B. WAP-Parkleitsysteme mit Anzeige der aktuellen Belegung, unterschiedlichste „Location Based Services“ im Bereich der Tourismus- und Kultur-Information oder der Sicherheits- und Notfalldienste [Fritsch/ Muntermann 2005] usw.) [Daum 2004; Franz 2005].

- *Trend zu „virtuellen Unternehmen“:* Die Tendenz zur Auflösung von Unternehmens- bzw. Organisationsgrenzen, also von räumlich definierten und hierarchisch organisierten Gebilden hin zu kooperativen, modularen, vernetzten und virtuellen Formen der Leistungserbringung wird anhalten. Auch im staatlichen Sektor versuchen vor allem Städte, Gemeinden und Kreise, mit innovativen Kooperationsmodellen die Effizienz ihrer Leistungserbringung zu steigern [Hofmann 2005]. Die moderne IKT wird weiterhin Katalysator dieser Entwicklungen sein [Grimm/Kozok/Lafos 2001].

5 Nachdenkliches zum Schluss ... oder: Wider die Technikgläubigkeit

Bisher haben wir die vielfältigen Potenziale der Informationstechnologien zur Optimierung von Geschäftsprozessen herausgestellt. Doch dies ist nur die eine Seite der Medaille – wir wollen die andere nicht verschweigen. Viele Fehlschläge, allerhand enttäuschte Erwartungen und „Flops", Sicherheitsprobleme, allgemeine Technik-Abhängigkeit und nicht zuletzt „leere Kassen" (gerade im öffentlichen Bereich) setzen IT-Hersteller und -Abteilungen unter einen verstärkten Rechtfertigungsdruck: verschafft IKT tatsächlich noch Wettbewerbsvorteile oder hat sie inzwischen den Stellenwert einer generellen Infrastruktur, aus der keine Wettbewerbsvorteile mehr zu erzielen sind – oder ist ist sie gar nur noch „Kostentreiber"?

Die Debatte um „Wettbewerbsvorteile durch IT" ist keinesfalls neu, kommt in den letzten Jahren aber erneut und in verschärfer Form hoch, denn vernetzte und virtualisierte Geschäftsmodelle und -tätigkeiten werden zunehmend Bestandteile von Unternehmens-/Verwaltungsstrategien und die Investitionsvolumina in immer komplexere IT (und damit verbundene Beratungsleistungen) sind – bei gleichzeitiger Standardisierung (etwa im ERP-Markt) – immens gestiegen [Fröschle 2004]. Und vor allem im Dienstleistungssektor und damit auch im eG steht der Beweis noch aus, dass IT-Einsatz einerseits und Steigerung der Arbeitsproduktivität, Kundenorientierung und Akzeptanz andererseits „automatisch" einhergehen (für das eG vgl. [Büllesbach 2005]).

Begründete Zweifel gibt es auch in der wissenschaftlichen Diskussion, die vor allem in der US-amerikanischen Literatur in zwei großen „Wellen" und verbunden mit zwei Namen geführt worden ist – Solow-Paradox und Carr-Debatte:

- Das „Produktivitätsparadox der IT" verweist auf den erstaunlichen Sachverhalt, dass trotz jahrzehntelang anwachsender massiver IT-Investitionen keine merklichen Zuwächse der Arbeitsproduktivität über übliche Produktivitätsfortschritte hinaus festzustellen sind (neuere Untersuchungen bestätigen für viele Industrieländer ab der zweiten Hälfte der 90er Jahre eine positive Korrelation, aber bei starker Streuung). Diese makroökonomischen Erkenntnisse haben R. Solow (Nobelpreisträger für Ökonomie) zu dem berühmten Ausspruch veranlasst: „You can see the computer age everywhere except in the productivity statistics" [Solow 1987] – daher auch der Begriff „Solow-Paradox".
- [Carr 2003] hat dann mit seinem provokant formulierten Artikel „IT Doesn't Matter" im *Harvard Business Review* (später als Buch weiter ausgeführt in [Carr 2004]) zwar keinen originären Beitrag zum IT-Produktivitätsparadox verfasst, aber die Diskussion neu entfacht, indem er nach Produktivitätsvorteilen durch IT für die Unternehmen fragt. IT sei heute eine „commodity"[2], die sich jeder kaufen kann und die nicht mehr automatisch zu Wettbewerbsvorteilen führe. IT sei heute da, wo vor 100 Jahren die Elektrizität war – und ähnlich, wie man heute Strom aus der Steckdose „on demand" beziehe, werde man das auch mit IT tun. Und: Es müsse nicht immer teure Spitzentechnologie, „latest vendor technology", sein – oft genüge „good enough open source", ASP oder gar komplettes IT-Outsourcing mit der Vision des „Utility Computing". Carrs Credo für IT-Investments: "Spend less!" und "Follow, don't lead!"

Es verwundert nicht, dass gerade die IT-Hersteller diese beide Debatten geradezu als ketzerisch angesehen und mit Akribie versucht haben, Solow bzw. Carr Recherche- oder Argumen-

[2] Als „commodities" werden auf offenen Märkten handelbare homogene Güter bezeichnet, die sich durch hohe Verfügbarkeit bei gleicher Qualität auszeichnen – nur der Preis spielt als Auswahlkriterium noch eine Rolle. IT sei als Basistechnologie also auf dem Weg zum Gebrauchsgut – zwar auf jeden Fall nötig, aber ohne strategische Bedeutung für Differenzierung und Kostenvorteile.

tationsfehler nachzuweisen. Neben solchen interessegeleiteten Reaktionen sind auch in der Forschungsliteratur viele Erklärungsansätze [Piller 1997, S. 31–62] ausgebreitet sowie Pro- und Contra-Argumente ausgetauscht worden, was wir hier aus Platzgründen leider nicht weiter nachzeichnen können [Klein 2004; Mieze 2004].

Es ist schon beinahe gleichgültig, ob oder in welchem Maße Solow und Carr „Recht" hatten bzw. behalten werden – jedenfalls lieferten sie wichtige Denkanstöße! [Wigand/Picot/Reichwald 1997] – und später auch [Smith/Fingar 2003] mit ihrer expliziten Carr-Replik schon im Buchtitel *IT Doesn't Matter – Business Processes Do* – haben aus unserer Sicht eine sinnvolle Perspektive vorgezeichnet (*die* allein „richtige" Antwort wird es wohl nicht geben): Zwischen IKT und Unternehmenserfolg besteht – gerade auf dem Hintergrund zunehmender Diffusionsgeschwindigkeit neuer IKT – *kein direkter* Zusammenhang, d. h. rein technologiebasierte Wettbewerbsvorteile sind kaum mehr und vor allem nicht langfristig realisierbar. Es existiert aber ein *indirekter* Zusammenhang – und Organisationen können sich neuen technologischen Entwicklungen daher auch nicht entziehen (IT also gewissermaßen zur Vermeidung von Wettbewerbs-Nachteilen statt zur Erzielung von -Vorteilen). So setzt das moderne „Business Engineering"-Verständnis meist an der Gestaltung von Geschäftsprozessen an, die als *Bindeglied* zwischen IKT einerseits und Unternehmensstrategie (und Geschäftsmodell) andererseits verstanden werden (Abb. 10).

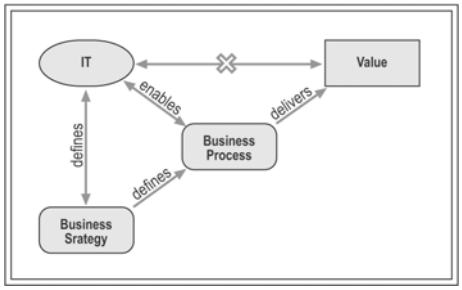


Abb. 10
Zum Zusammenhang zwischen IKT, Unternehmensstrategie, Geschäftsmodellen und Unternehmenserfolg
[Quelle: nach Wigand/Picot/Reichwald 1997]

Die Technologie selbst wird also einerseits zur omnipräsenten „commodity", deren Einsatz als solcher noch keine Wettbewerbsvorteile bringt – andererseits unterstützt der Einsatz von IT nicht nur *bestehende* Prozesse in und zwischen Organisationen, sondern kann in der Wechselwirkung mit innovativen Geschäftsstrategien und -modellen für die Realisierung *innovativer* Prozesse ein ganz wesentlicher „enabler" sein.

Es gibt eine ganze Reihe von Erfolgsfaktoren, die dafür sorgen, dass manche Organisationen durch massiven IT-Einsatz die Konkurrenz abhängen, während für andere mit einer ähnlichen Ausgangslage „große" IT-Projekte zum Desaster werden. Diese Erfolgsfaktoren sind eher organisatorischer Art und liegen meist ausserhalb des IT-Kernbereichs. Stichpunktartig seien etwa genannt [Brynjolfsson 2003]:

- Fokussierung auf die strategischen Unternehmensziele
- hoher Automatisierungsgrad von Routine-Prozessen und Umwandlung von herkömmlichen in automatisierte, digitale, möglichst innovative Geschäftsprozesse
- strategisches Informations-/Wissensmanagement als eine wesentliche Voraussetzung für erfolgreichen IT-Einsatz; nach [Marchand/Kettinger/Rollins 2001] korreliert eine hohe „information orientation" mit dem Unternehmenserfolg
- da IT als Investitionsgut kontextabhängig ist (im Unterschied z. B. zur Elektrizität), kommt es auf eine „passende", genau auf die Wertschöpfungskette abgestimmte Technik an [Schiele 2004], man spricht allgemein von der „Güte" der IT je Wertschöpfungsaktivität („IT adequacy")
- die Zeitdimension, d. h. der Produktivitätsbeitrag der IT ist abhängig von der Entwicklungsphase einer Branche (z. B. sind in der Expansionsphase „early mover advanta-

ges“ möglich, in der Normalisierungsphase ist IT eher „commodity“, bei der es stärker auf eine effektivere Nutzung ankommt) [Schiele 2004]
- das von der IT erzeugte Innovationspotenzial muss mit der Innovations*fähigkeit* einer Organisation zusammenkommen [Bodendorf/Robra-Bissantz/Bauer 2004]
- die starke Rolle des „Humankapitals“ muss berücksichtigt werden (hoch qualifizierte Mitarbeiter, überwiegend dezentralisierte Entscheidungsprozesse, Delegation von Verantwortung, leistungsbezogene Anreizsysteme, Investitionen in Weiterbildung etc.) [Gunnarsson/Mellander/Savvidou 2001].

Kein Anlass also für Technik-Geringschätzung, aber eben auch nicht für Technik-Euphorie, so mag man kurz resümieren! Unter Innovationsgesichtspunkten handelt es sich bei der IKT-Unterstützung von Geschäftsprozesssen eben nicht nur um Produkt- und Verfahrens-Innovationen, sondern immer auch um strukturelle und soziale Innovationen, die bei allen Beteiligten eine Abkehr vom Gewohnten erfordern und letztlich kulturelle Veränderungen bedeuten. Innovationsbereitschaft und -vermögen sowie Change Management sind gefragt.

Bei aller Wichtigkeit von IKT und der Optimierung von Geschäftsprozessen – es geht also auch um „weiche Faktoren“, um Menschen (Mitarbeiter, Bürger, ...), um die wahrgenommene Nützlichkeit des Neuen und um Akzeptanz, auch um eine Verminderung der „digitalen Kluft“. Und Akzeptanz kann man letztlich nicht „erzeugen“ (weder mit dem an sich berechtigten Ruf nach mehr „Medienkompetenz“ noch mit Anreizen) – und bei fehlender Akzeptanz den „schwarzen Peter“ dann den „altmodischen“ oder „störrischen“ (Nicht-) Nutzern zuschieben. Das Neue muss dem Alten schlichtweg überlegen sein und diese Überlegenheit muss transparent, erfahrbar und überzeugend sein – sonst wird es nicht akzeptiert werden.

Literaturverzeichnis

Aus Platzgründen sind die vollständigen URLs von leicht über Suchmaschinen findbaren Online-Dokumenten hier nicht wiedergegeben, stattdessen findet sich folgender Vermerk: ► @

[Alff/Bungert 2004] Alff, S. C./Bungert, W., „Business Intelligence“, in: Scheer, A.-W. et al. (Hrsg.): *Innovation durch Geschäftsprozessmanagement – Jahrbuch Business Process Excellence 2004/2005*. Berlin/Heidelberg 2004, S. 155–167

[Allweyer 2000] Allweyer, T.: *Integration überbetrieblicher Geschäftsprozesse mit Hilfe von Internet-Marktplätzen* (Vorlesung an der TU Chemnitz, Dez. 2000) ► @

[Binner 2004] Binner, H. F.: *Handbuch der prozessorientierten Arbeitsorganisation*. Darmstadt 2004

[Bodendorf/Robra-Bissantz/Bauer 2004] Bodendorf, F./Robra-Bissantz, S./Bauer, C., „There's more to IT – vom Innovationspotenzial zur Innovationsfähigkeit“, in: [Fröschle 2004], S. 7–17

[Brynjolfsson 2003] Brynjolfsson, E., “The IT Productivity GAP”, in: *Optimize* 2003: 21 ► @

[Büllesbach 2005] Büllesbach, R., „eGovernment – Sackgasse oder Erfolgsstory“, in: *Deutsches Verwaltungsblatt* 2005: 10, S. 605–611

[Carr 2003] Carr, N. G., “IT Doesn't Matter”, in: *Harvard Business Review* **81** (2003): 5, S. 41–49

[Carr 2004] Carr, N. G.: *Does IT Matter?* Boston 2004

[Daum 2003] Daum, R., „Die Bedeutung personalisierter Portale im Electronic Government“, in: *Wirtschaftswissenschaftliches Studium* 2003: 4, S. 197–202

[Daum 2004] Daum, R., „Gestaltungsmöglichkeiten von M-Government“, in: [Reichard/ Scheske/Schuppan 2004], S. 140–151

[Franz 2005] Franz, A., „Mobile Kommunikation : Anwendungsbereiche und Implikationen für die öffentliche Verwaltung“, in: *Verwaltung und Management* **11** (2005): 3, S. 123–128

[Fritsch/Muntermann 2005] Fritsch, L./Muntermann, J., „Aktuelle Hinderungsgründe für den kommerziellen Erfolg von Location Based Service-Angeboten“, in: Hampe, J. F. et al. (Hrsg.): *Mobile Business – Processes, Platforms, Payments*. Bonn 2005, S. 143–156

[Fröschle 2004] Fröschle, H.-P. (Hrsg.): *Wettbewerbsvorteile durch IT* (= Praxis der Wirtschaftsinformatik – HMD, Heft 239, Okt. 2004). Heidelberg 2004

[Gaitanides/Ackermann 2004] Gaitanides, M./Ackermann, I.: *Die Geschäftsprozeßperspektive als Schlüssel zu betriebswirtschaftlichem Denken und Handeln*, 2004 ► @

[Grimm 2004] Grimm, S., „Prozessportale als Basis für elektronische Geschäftsprozesse“, in: [Horster 2004], S. 24–43

[Grimm/Kozok/Lafos 2001] Grimm, R./Kozok, B./Lafos, F., „Kommunikationsinfrastruktur für virtuelle Unternehmen“, in: Gora, W./Bauer, H. (Hrsg.): *Virtuelle Organisationen im Zeitalter von E-Business und E-Government*. Berlin 2001, S. 199–209

[Gunnarsson/Mellander/Savvidou 2001] Gunnarsson, G./Mellander, E./Savvidou, E.: *Is Human Capital the Key to the IT Productivity Paradox?* (= The Research Institute of Industrial Economics, Working Paper No. 551). Stockholm 2001 ► @

[Herterich 2005] Herterich, R., „Prozessmanagement zwischen QM und IT“, in: *Information Management & Consulting* **20** (2005): Sonderausgabe, S. 82–88

[Hofmann 2003] Hofmann, O., „Web-Services in serviceorientierten IT-Architekturkonzepten“, in: Fröschle, H.-P. (Hrsg.): *Web-Services* (= Praxis der Wirtschaftsinformatik – HMD, Heft 234, Dez. 2003). Heidelberg 2003, S. 27–33

[Hofmann 2005] Hofmann, G., „Virtuelle Unternehmen – Ein neues Kooperationsmodell für Kommunen?“, in: Brosch, D./Mehlich, H. (Hrsg.): *E-Government und virtuelle Organisation*. Wiesbaden 2005, S. 157–180

[Horster 2004] Horster, P. (Hrsg.): *Elektronische Geschäftsprozesse 2004*. Sauerlach 2004

[Kagermann/Zencke 2005] Kagermann, H./Zencke, P., „Die Renaissance des Geschäftsprozess-Managements : Von Modellierungswerkzeugen zur Prozessplattform für den Wandel“, in: *Information Management & Consulting* **20** (2005): Sonderausgabe, S. 6–11

[Khodawandi, D./Pousttchi, K./Winnewisser, C.: *Mobile Technologie braucht neue Geschäftsprozesse*, 2003 ► @

[Klein 2004] Klein, S., „IT does matter! – Einige Überlegungen zum Produktivitätsparadoxon“, in: Becker, J. et al.: *European Research Center for Information Systems (ERCIS), Gründungsveranstaltung Münster, 12. Okt. 2004* (= ERCIS Working Paper No. 1), S. 91–96 ► @

[Köhler/Gruhn 2004] Köhler, A./Gruhn, V., „Lösungsansätze für verteilte mobile Geschäftsprozesse“, in: [Horster 2004], S. 243–255

[Lehner/Meier/Stormer 2005] Lehner, F./Meier, A./Stormer, H. (Hrsg.): *Mobile Anwendungen* (= Praxis der Wirtschaftsinformatik – HMD, Heft 244, Aug. 2005). Heidelberg 2005

[Lucke 2004] Lucke, J. v., „Portale als zentraler Zugang zu E-Government-Diensten“, in: [Reichard/Scheske/Schuppan 2004], S. 79–94

[Marchand/Kettinger/Rollins 2001] Marchand, D. A./Kettinger, W. J./Rollins, J. D.: *Information Orientation : The Link to Business Performance*. Oxford 2001

[Meier 2002] Meier, A. (Hrsg.): *E-Government* (= Praxis der Wirtschaftsinformatik – HMD, Heft 226, Aug. 2002). Heidelberg 2002

[Mertens/Faisst 1995] Mertens, P./Faisst, W.: *Virtuelle Unternehmen - Einführung und Überblick*, 1995 ► @

[Mieze 2004] Mieze, T., „Beyond Carr – und sie bewegt sich doch“, in: [Fröschle 2004], S. 18–37

[Müller 2005] Müller; J.: *Workflow-based Integration*. Berlin/Heidelberg 2005

[Nefiodow 1990] Nefiodow, L. A.: *Der fünfte Kondratieff*. Frankfurt/Wiesbaden 1990

[Pfeifer 2003] Pfeifer, A.: *Zum Wertbeitrag der Informationstechnologie* (= Diss., Univ. Passau, 2003)

[Picot/Neuburger 2000] Picot, A./Neuburger, R., „Informationsbasierte (Re-) Organisation von Unternehmen“, in: [Weiber 2000], S. 383–401

[Picot/Scheuble 1997] Picot, A./Scheuble, S., „Die Bedeutung der Information für Innovation und Wettbewerbsfähigkeit von Unternehmen“, in: Mantwill, G. J. (Hrsg.): *Informationswirtschaft und Standort Deutschland*. Baden-Baden 1997, S. 15–41

[Piller 1997] Piller, F. T.: *Das Produktivitätsparadoxon der Informationstechnologie*. 2. Aufl., Würzburg 1997 ► @

[Reichard/Scheske/Schuppan 2004] Reichard, C./Scheske, M./Schuppan, T. (Hrsg.): *Das Reformkonzept E-Government : Potenziale – Ansätze – Erfahrungen*. Münster 2004

[Schiele 2004] Schiele, H., „Wettbewerbsvorteile durch IT-Beschaffung: Das Technologieübernahmeparadoxon in der IT“, in: [Fröschle 2004], S. 28–37

[Schmelzer/Sesselmann 2004] Schmelzer, H. J./Sesselmann, W.: *Geschäftsprozessmanagement in der Praxis*. 4. Aufl., München/Wien 2004

[Seidenberg 1998] Seidenberg, U.: *Ist Information als eigenständiger Produktionsfaktor aufzufassen?* Siegen 1998 ► @

[Smith/Fingar 2003] Smith, H./Fingar, P.: *IT Doesn’t Matter – Business Processes Do*. Tampa 2003

[Solow 1987] Solow, R. M., “We’d Better Watch Out”, in: *New York Times*, 1987, July 12th, S. 36

[Spahni/Meir 2003] Spahni, D./Meir, J., „Web Services im eGovernment: Vision und Konzept veränderter Wertschöpfungsketten der staatlichen Leistungserbringung“, in: *Tagungsband IRIS 2003* (Intern. Rechtsinformatik-Symposium, Salzburg, 20.–22. Feb. 2003) ► @

[Weiber 2000] Weiber, R. (Hrsg.): *Handbuch Electronic Business*. Wiesbaden 2000

[Weiber/Krämer 2000] Weiber, R./Krämer, T., „Paradoxien des Electronic Business“, in: [Weiber 2000], S. 149–177

[Weiber/McLachlan 2000] Weiber, R./McLachlan, C., „Wettbewerbsvorteile im Electronic Business“, in: [Weiber 2000], S. 117–148

[Wigand/Picot/Reichwald 1997] Wigand, R./Picot, A./Reichwald, R.: *Information, Organization and Management*. Chichester 1997

[Wirtz 2001] Wirtz, B. W.: *Electronic Business*. 2. Aufl., Wiesbaden 2001

Modellierung und Prognose autoregressiver und Vektor-autoregressiver Zeitreihen

Jürgen Jacobs
Fakultät III - Umwelt und Technik, Leuphana Universität Lüneburg

1 Einleitung und Themenabgrenzung

Zeitreihenprognosen versuchen die künftige Entwicklung von interessierenden Größen wie bspw. Zinsen, Inflationsraten oder Aktienkurse aus Kenntnis historischer Werte zu extrapolieren. Im univariaten Fall wird nur eine Größe betrachtet, im multivariaten Fall werden Interdependenzen zwischen mehreren Größen berücksichtigt. Während im univariaten Fall künftige Werte allein aus den historischen Werten extrapoliert werden, könnten im multivariaten Fall auch kausale Modelle wie z. B. Zinstheorien Berücksichtigung finden. Wir beschränken uns hier auf reine Zeitreihenmodelle, d. h., die Interdependenzen zwischen den betrachteten Größen werden allein mit Hilfe historischer Werte ermittelt. Ebenso sei angemerkt, dass Zeitreihen nicht der einzig denkbare Gegenstand einer Prognose sind. Ebenso könnte man Prognosen von Ereignissen betrachten, wie das Eintreten eines Ereignisses mit ungewissem Ausgang (z. B. Ausgang einer Wahl) oder den Zeitpunkt des Eintretens eines sicheren Ereignisses (z. B. Zinsänderung).

In der Zeitreihenprognose lassen sich drei Prognosetypen unterscheiden: Punkt-, Intervall- und Dichteprognose. Bei der Punktprognose wird nur ein einzelner Wert, in der Regel der bedingte Erwartungswert bei gegebenen historischen Werten, bei der Intervallprognose wird dagegen ein Wertebereich geschätzt, in welchem der Prognosewert mit einer bestimmten Wahrscheinlichkeit liegen wird. Bei der Dichteprognose wird die gesamte Wahrscheinlichkeitsverteilung des Prognosewerts geschätzt. In der vorliegenden Arbeit werden Dichteprognosen auf der Basis von Simulationsrechnungen betrachtet.

2 Univariate autoregressive Zeitreihenmodelle

Eine wichtige Klasse der univariaten Zeitreihenmodelle bilden die sog. autoregressiven Modelle. Sie beschreiben die aktuelle Beobachtung als gewichtete Summe zurückliegender Realisationen. Ein autoregressiver Prozess der Ordnung p (AR(p)-Prozess) folgt der Beziehung:

$$(1) \quad x_t = \alpha + \varphi_1 \cdot x_{t-1} + \varphi_2 \cdot x_{t-2} + \ldots + \varphi_p \cdot x_{t-p} + u_t, \quad t = p,\, p+1,\, p+2, \ldots .$$

Die Störgrößen u_t werden als unabhängige Zufallsvariablen angenommen, die alle dieselbe Verteilung (independent and identically distributed (IID)) um den Mittelwert 0 haben: $u_t \sim IID(0, \sigma^2)$. Gleichung (1) beschreibt eine Regression der Variablen x_t auf ihre eigenen verzögerten Werte. Dies erklärt die Namensgebung.

Eine wichtige Voraussetzung für die Parameterschätzung ist die Stationarität der Zeitreihe. Eine Zeitreihe heißt *streng stationär*, falls die gemeinsame Verteilung von

$(x_{t+l1}, x_{t+l2}, ..., x_{t+lk})$ für beliebig gewählte *l1, l2, ..., lk* nur von den Verschiebungen *l1, l2, ..., lk* und nicht vom Zeitpunkt *t* abhängt. Dies ist in der Praxis selten gegeben. Daher wird in der Zeitreihenanalyse meist nur eine schwache Stationarität angenommen. Eine Zeitreihe heißt *schwach stationär*, falls gilt:

$(a) \quad E(x_t) = \mu,$

$(b) \quad C(x_t, x_{t-l}) = \gamma_l.$

Das heißt, der Erwartungswert (E) ist zeitunabhängig, und die Autokovarianz (C) hängt nur von der Verschiebung l, nicht aber vom Zeitpunkt t ab.

Ein AR(1)-Modell ist gegeben durch:

$$(2) \quad x_t = \alpha + \varphi_1 \cdot x_{t-1} + u_t, \quad t = 1, 2, 3, \ldots .$$

Nehmen wir die Gültigkeit von (2) für beliebig weit zurück liegende Zeitpunkte t = -1, -2, -3, ... an, so ergibt die sukzessive Anwendung von Gleichung (2) für x_{t-1}, x_{t-2}, ...:

$$(3) \quad x_t = \alpha + u_t + \varphi_1(\alpha + u_{t-1}) + \varphi_1^2(\alpha + u_{t-2}) + \ldots$$

Falls $|\varphi_1| < 1,$ beschreibt (3) einen schwach stationären Prozess. Für den Erwartungswert erhält man:

$$\mu = E(x_t) = \alpha + \varphi_1\alpha + \varphi_1^2\alpha + \ldots = \alpha\sum_{i=0}^{\infty}\varphi_1^i = \frac{\alpha}{1-\varphi_1}.$$

Dabei wurde $E(u_t) = 0$ verwendet. Die Beziehung $|\varphi_1| < 1$ wurde für die Konvergenz der Reihe benötigt. Für die Autokovarianz erhält man wegen $E(u_i \cdot u_i) = \sigma^2$ und $E(u_i \cdot u_j) = 0$ für $i \neq j$:

$$C(x_t, x_{t-l}) = E[(x_t - \mu)(x_{t-l} - \mu)]$$

$$= E[(u_t + \varphi_1 u_{t-1} + \varphi_1^2 u_{t-2} + \ldots + \varphi_1^l u_{t-l} + \varphi_1^{l+1} u_{t-l-1} + \varphi_1^{l+2} u_{t-l-2} + \ldots)(u_{t-l} + \varphi_1 u_{t-l-1} + \varphi_1^2 u_{t-l-2} + \ldots)]$$

$$= (\varphi_1^l + \varphi_1^{l+2} + \varphi_1^{l+4} + \ldots)\sigma^2 = \varphi_1^l(1 + \varphi_1^2 + \varphi_1^4 + \ldots)\sigma^2 = \frac{\varphi_1^l}{1-\varphi_1^2}\sigma^2.$$

Im Falle von $\varphi_1 = 1$ spricht man von einem sog. *Random-Walk-Prozess*. Die sukzessive Anwendung von (2) auf x_{t-1}, x_{t-2}, ..., x_1 liefert:

$$(4) \quad x_t = x_0 + \alpha \cdot t + \sum_{i=1}^{t} u_i.$$

Erwartungswert und Autokovarianz sind nun zeitabhängig:

$$E(x_t) = x_0 + \alpha \cdot t, \quad C(x_t, x_{t-l}) = (t-l) \cdot \sigma^2.$$

Der Teil $x_0 + \alpha \cdot t$ beschreibt ein deterministisches Wachstumsverhalten. Man spricht von daher von einem *stochastischen Trend*. Charakteristisch für Random-Walk-Prozesse mit oder ohne Trend (α = 0) ist ein aufgrund der kumulierten Störgrößen unbeschränktes Wachstum mit der Zeit: $Var(x_t) = t \cdot \sigma^2$.

Die kumulierten Störgrößen können zu längeren Phasen wachsender und fallender Werte von x_t führen, die ein trendartiges Verhalten vortäuschen und daher bei Regressionsanalysen zu fehlerhaften Modellen führen. Bereits Granger und Newbold (1974) haben auf dieses sog. *Spurious-Regression-Problem* hingewiesen. Um Random-Walk-Prozesse im Rahmen der klassischen Zeitreihenanalyse behandeln zu können, werden Differenzen gebildet. Ist beispielsweise eine Zeitreihe x_t ein Random-Walk-Prozess ohne Trend, so ist die erste Differenz $\Delta x_t = x_t - x_{t-1} = u_t$ eine Zeitreihe, die durch die Störgröße beschrieben wird. Nun ist denkbar, dass die ersten Differenzen ebenfalls ein Random-Walk-Verhalten aufweisen. In diesem Fall werden zweite Differenzen gebildet: $\Delta^2 x_t = \Delta(x_t - x_{t-1}) = x_t - 2 \cdot x_{t-1} + x_{t-2}$. Dieses Verfahren wird so lange fortgesetzt, bis man kein Random-Walk-Verhalten mehr antrifft. Ein Prozess, bei dem d Differenzen zu bilden sind, heißt *integriert* von der Ordnung d oder kurz I(d)-Prozess.

Betrachten wir einen AR(1)-Prozess ohne Interzept α: $x_t = \varphi_1 x_{t-1} + u_t$. Die gewöhnliche Kleinste-Quadrate-Schätzung (ordinary least squares (OLS)-Schätzung) liefert:

$$\hat{\varphi}_1 = \frac{\sum_{t=1}^{n} x_t \cdot x_{t-1}}{\sum_{t=1}^{n} x_{t-1}^2}$$

Zum Testen der Nullhypothese $\varphi_1 = 1$ bietet sich die Statistik

$$\tau = \frac{\hat{\varphi}_1 - 1}{std(\hat{\varphi}_1)}$$

an; $std(\hat{\varphi}_1)$ bezeichnet den Standardfehler des Schätzers $\hat{\varphi}_1$. Die Verteilung von τ hängt vom wahren Wert φ_1 ab. Für $|\varphi_1| < 1$ folgt τ der t-Verteilung mit n Freiheitsgraden. Im Falle eines Random Walks – also bei Zutreffen der Nullhypothese - folgt τ nicht der t-Verteilung. Kritische Schranken können mit Hilfe von Monte-Carlo-Simulationen ermittelt werden. Sie liegen unterhalb der Schranken der t-Verteilung mit unendlich vielen Freiheitsgraden. Dies zeigt, dass die Nullhypothese bei Entscheidung auf Basis einer t-Verteilung viel zu oft verworfen würde. Der Test auf Basis der τ-Statisik wird *Unit-root-Test* oder nach den Entdeckern *Dickey-Fuller-Test* genannt.

Das Modell $x_t = \varphi_1 x_{t-1} + u_t$ lässt sich umschreiben zu $\Delta x_t = \delta x_{t-1} + u_t$ mit $\delta = \varphi_1 - 1$. Ein Random Walk ist demnach nicht auszuschließen, falls die Hypothese $\delta = 0$ nicht abgelehnt werden kann. Im Fall eines AR(p)-Prozesses mit p > 1 lässt sich das von Dickey und Fuller (1979) vorgeschlagene Testverfahren in seiner verallgemeinerten Form verwenden. Zusätzlich können noch ein Interzept und ein deterministischer Trend in die Modellgleichung aufgenommen werden:

(5) $\Delta x_t = \delta x_{t-1} + \beta_1 \Delta x_{t-1} + ... + \beta_{p-1} \Delta x_{t-p-1} + u_t$,

(6) $\Delta x_t = \alpha + \delta x_{t-1} + \beta_1 \Delta x_{t-1} + ... + \beta_{p-1} \Delta x_{t-p-1} + u_t$,

(7) $\Delta x_t = \alpha + \beta \cdot t + \delta x_{t-1} + \beta_1 \Delta x_{t-1} + ... + \beta_{p-1} \Delta x_{t-p-1} + u_t$.

Gleichung (6) geht aus Gleichung (1) hervor, indem man $\delta = \varphi_1 + \varphi_2 + ... + \varphi_p - 1$ und $\beta_i = -\sum_{j=i+1}^{p} \varphi_j$ setzt. Die Dickey-Fuller t-Statistiken zur Überprüfung der Nullhypothese $\delta = \varphi_1 + \varphi_2 + ... + \varphi_p - 1 = 0$ zeigt, ob ein Random-Walk für die Modelle (5), (6) bzw. (7) anzunehmen ist, werden mit τ, τ_μ bzw. τ_τ bezeichnet. Mit Hilfe des F-Tests nach Dickey und Fuller (1981) kann die Signifikanz der Parameter α bzw. β überprüft werden:

$$F = \frac{S_0 - S}{S} \cdot \frac{n+1-l}{r}.$$

S_0 bezeichnet die Summe der quadrierten Residuen bei Zutreffen der Nullhypothese, S die Summe der quadrierten Residuen ohne Restriktion, r die Anzahl der Parameterrestriktionen bei Zutreffen der Nullhypothese, $n + 1$ den Stichprobenumfang und l die Anzahl der Parameter. Die entsprechenden Teststatistiken nennt man Φ_1 (Modell (6), Nullhypothese: $\alpha = \delta = 0$), Φ_2 (Modell (7), Nullhypothese: $\alpha = \beta = \delta = 0$) bzw. Φ_3 (Modell (7), Nullhypothese: $\beta = \delta = 0$).

Die Anzahl p der zu berücksichtigenden Verzögerungen wird üblicherweise mit Hilfe von sog. Informationskriterien ermittelt, die den Schätzfehler mit der Anzahl l der verwendeten Parameter zu einer Größe verknüpfen. Je mehr Parameter verwendet werden, desto „leichter" ist es, den Schätzfehler klein zu halten. Daher wird jede Hinzunahme von Parametern „bestraft". Häufig verwendet werden die *Informationskriterien* von Akaike (1973) (*AIC*) und Schwarz (1978) (*SIC*). Bis auf konstante Terme sind die Kriterien wie folgt definiert:

$$AIC = \log\left(\frac{1}{n-p+1}\sum_{i=p}^{n}\hat{u}_i^2\right) + \frac{2}{n-p+1}l,$$

$$SIC = \log\left(\frac{1}{n-p+1}\sum_{i=p}^{n}\hat{u}_i^2\right) + \frac{\log(n-p+1)}{n-p+1}l,$$

wobei $\hat{u}_i$ die durch die Residuen geschätzten Störgrößen bezeichnen. Die Ordnung p wird nun so bestimmt, dass das gewählte Kriterium einen minimalen Wert annimmt. Für $n - p+1 > 7$ gilt $\log(n - p+1) > 2$; daher führt die Verwendung des Schwarzschen Informationskriteriums im Allgemeinen zu einer niedrigeren Ordnung als die Verwendung des Informationskriteriums von Akaike.

3 Vektor-autoregressive Zeitreihenmodelle

Zur Berücksichtigung von kontemporären und intertemporären Abhängigkeiten zwischen den untersuchten Zeitreihen bieten sich Vektor-autoregressive (VAR) Modelle an. Bei k Zeitreihen und der Berücksichtigung von p Verzögerungen erhält man folgendes Gleichungssystem:

$$\begin{aligned}
(8)\quad & y_{1,t} = \alpha_1 + \delta_{11,1} \cdot y_{1,t-1} + ... + \delta_{1k,1} \cdot y_{k,t-1} + ... + \delta_{11,p} \cdot y_{1,t-p} + ... + \delta_{1k,p} \cdot y_{k,t-p} + u_{1,t}, \\
& y_{2,t} = \alpha_2 + \delta_{21,1} \cdot y_{1,t-1} + ... + \delta_{2k,1} \cdot y_{k,t-1} + ... + \delta_{21,p} \cdot y_{1,t-p} + ... + \delta_{2k,p} \cdot y_{k,t-p} + u_{2,t}, \\
& ... \\
& y_{k,t} = \alpha_k + \delta_{k1,1} \cdot y_{1,t-1} + ... + \delta_{kk,1} \cdot y_{k,t-1} + ... + \delta_{k1,p} \cdot y_{1,t-p} + ... + \delta_{kk,p} \cdot y_{k,t-p} + u_{k,t}.
\end{aligned}$$

Falls einzelne Zeitreihen ein Random-Walk-Verhalten aufweisen, ist es naheliegend, diese zunächst durch Differenzenbildung in stationäre Zeitreihen zu überführen und anschließend in das Gleichungssystem (8) aufzunehmen. Dadurch werden jedoch möglicherweise zu viele Differenzen gebildet – nämlich dann, wenn Kointegrationsbeziehungen zwischen einzelnen Zeitreihen bestehen. *Kointegration* bedeutet, dass eine Linearkombination nicht-stationärer Zeitreihen gefunden werden kann, die stationär ist. Das Konzept der Kointegration geht auf Granger (1981) und Engle und Granger (1987) zurück. Tiao, Tsay und Wang (1993) zeigen an einem Beispiel auf, dass es problematisch sein kann, existierende Gleichgewichtsbeziehungen mit einem Kointegrationstest zu erkennen. Insbesondere wird von ihnen kritisiert, dass die Größe der Streuung der einzelnen Zeitreihen vom Kointegrationstest nicht berücksichtigt wird. Die Bedeutung von Kointegrationsbeziehungen für langfristige Prognosen wurde u. a. von Engle und Yoo (1987) herausgestellt. Christoffersen und Diebold (1997) bemängeln allerdings zu Recht, dass die Ergebnisse schwer zu interpretieren sind, da nicht das VAR-Modell der stationären Zeitreihen sondern das VAR-Modell der nicht-stationären Niveauwerte mit dem Kointegrationsmodell verglichen wird. In einer eigenen Untersuchung kommen sie zu dem Resultat, dass die Berücksichtigung von Kointegrationsbeziehungen für langfristige Prognosen zu schlechteren Ergebnissen führt als die Verwendung univariater integrierter Variablen. Im Folgenden soll daher auf eine Betrachtung von Kointegrationsmodellen verzichtet werden. Die Bedeutung von Unit-Root-Tests und Kointegrationstests für die Spezifikation von VAR-Modellen ist in der Literatur umstritten. Eine aktuelle Studie zur Modellauswahl und einen Literaturüberblick findet man bei Allen und Fildes (2007).

Ein weiteres Problem von VAR-Modellen ist, dass die Anzahl der zu schätzenden Parameter rasch mit der Dimension k und der Ordnung p anwächst. Beispielsweise sind bei 5 Zeitreihen und 3 Verzögerungen bereits $k \cdot (1 + k \cdot p) = 5 \cdot (1 + 5 \cdot 3) = 80$ Parameter zu schätzen. Eine hohe Anzahl von Parametern erschwert nicht nur die Interpretation des Modells, sondern führt auch zu großen Ungenauigkeiten bei der Parameterschätzung, falls die Anzahl der verfügbaren historischen Zeitreihenwerte nicht erheblich größer als die Anzahl der Parameter ist. In der Praxis ist häufig eine große Anzahl der geschätzten Parameter nicht signifikant von Null verschieden. Zur Elimination nicht signifikanter Parameter bietet sich das sequentielle Verfahren nach Brüggemann und Lütkepohl (2001) an, welches nacheinander die Regressoren eliminiert, die zur größten Reduktion des Informationskriterium führen, bis keine weitere Reduktion mehr möglich ist.

Für eine Stichprobe mit Indexmenge $I = \{0, 1, 2, ..., n\}$ ergibt sich in Matrixform folgendes Gleichungssystem:

(9) $Y = \delta \cdot \mathrm{Z} + U,$

wobei δ die vollständige $k \times (1 + k \cdot p)$ - Parametermatrix

$$\delta = \begin{pmatrix} \alpha_1 & \delta_{11,1} & \cdots & \delta_{1k,1} & \cdots & \delta_{11,p} & \cdots & \delta_{1k,p} \\ \alpha_2 & \delta_{21,1} & \cdots & \delta_{2k,1} & \cdots & \delta_{21,p} & \cdots & \delta_{2k,p} \\ & & & \ddots & & & & \\ \alpha_k & \delta_{k1,1} & \cdots & \delta_{kk,1} & \cdots & \delta_{k1,p} & \cdots & \delta_{kk,p} \end{pmatrix}$$

bezeichnet und

$$Y = [Y_p \cdots Y_n],\ Y_t = \begin{bmatrix} y_{1,t} \\ y_{2,t} \\ \vdots \\ y_{k,t} \end{bmatrix},\ Z = [Z_{p-1} \cdots Z_{n-1}],\ Z_t = \begin{bmatrix} 1 \\ Y_t \\ Y_{t-1} \\ \cdots \\ Y_{t-p+1} \end{bmatrix},\ U = [U_p \cdots U_n],\ U_t = \begin{bmatrix} u_{1,t} \\ u_{2,t} \\ \cdots \\ u_{k,t} \end{bmatrix}$$

sind.

Die Restriktionen lassen sich mit Hilfe einer geeigneten Matrix R in der Form $vec(\delta) = R\gamma$ notieren, wobei γ den Vektor der von Null verschiedenen Parameter bezeichnet. Der vec-Operator überführt eine Matrix in einen Spaltenvektor, welcher – beginnend mit der ersten Spalte – aus den untereinander geschriebenen Spalten der Matrix besteht. Sei zum Beispiel $k = 2$ und $p = 2$, so wird aus dem Gleichungssystem (8):

$$y_{1,t} = \alpha_1 + \delta_{11,1} \cdot y_{1,t-1} + \delta_{12,1} \cdot y_{2,t-1} + \delta_{11,2} \cdot y_{1,t-2} + \delta_{12,2} \cdot y_{2,t-2} + u_{1,t},$$
$$y_{2,t} = \alpha_2 + \delta_{21,1} \cdot y_{1,t-1} + \delta_{22,1} \cdot y_{2,t-1} + \delta_{21,2} \cdot y_{1,t-2} + \delta_{22,2} \cdot y_{2,t-2} + u_{2,t}.$$

Die Restriktion $\delta_{21,1} = 0$ lässt sich dann in der Form

$$vec(\delta) = \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \delta_{11,1} \\ 0 \\ \delta_{12,1} \\ \vdots \\ \delta_{22,2} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 1 & \cdots & 0 \\ & & \ddots & & & \\ 0 & 0 & 0 & 0 & \ldots & 1 \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \delta_{11,1} \\ \delta_{12,1} \\ \vdots \\ \delta_{22,2} \end{pmatrix}$$

schreiben. Für die nachfolgenden Umformungen werden folgende Rechenregeln für passende Matrizen verwendet:

$(R1)$ $(A \otimes B)' = A' \otimes B'$.

$(R2)$ $(A \otimes B)(C \otimes D) = AC \otimes BD$.

$(R3)$ $(A \otimes B)^{-1} = A^{-1} \otimes B^{-1}$.

$(R4)$ $vec(AB) = (B' \otimes I)vec(A)$.

Hierbei bezeichnen I die Einheitsmatrix, $\otimes$ das Kronecker-Produkt und A' die transponierte Matrix A. Das Kronecker-Produkt einer $(m \times n)$-Matrix $A = (a_{ij})$ und $(p \times q)$–Matrix $B = (b_{ij})$ ergibt eine $(mp \times nq)$-Matrix, die wie folgt definiert ist:

$$A \otimes B = \begin{pmatrix} a_{11}B & \cdots & a_{1n}B \\ & \ddots & \\ a_{m1}B & \cdots & a_{mn}B \end{pmatrix}.$$

Aus (9) und $vec(\delta) = R\gamma$ folgt:

$$(10)\quad vec(Y) = vec(\delta \cdot Z) + vec(U) \overset{(R4)}{=} (Z' \otimes I_k)vec(\delta) + vec(U) = (Z' \otimes I_k)R\gamma + vec(U).$$

Für das obige Beispiel mit $k = 2$, $p = 2$ und der Restriktion $\delta_{21,1} = 0$ ergibt sich:

$$vec(Y) = \begin{pmatrix} y_{1,2} \\ y_{2,2} \\ \vdots \\ y_{1,n} \\ y_{2,n} \end{pmatrix} = \left[\begin{pmatrix} 1 & y_{1,1} & y_{2,1} & y_{1,0} & y_{2,0} \\ 1 & y_{1,2} & y_{2,2} & y_{1,1} & y_{2,1} \\ & & \ddots & & \\ 1 & y_{1,n-1} & y_{2,n-1} & y_{1,n-2} & y_{2,n-2} \end{pmatrix} \otimes \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \right] \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \delta_{11,1} \\ 0 \\ \delta_{12,1} \\ \vdots \\ \delta_{22,2} \end{pmatrix} + \begin{pmatrix} u_{1,2} \\ u_{2,2} \\ \vdots \\ u_{1,n} \\ u_{2,n} \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 0 & y_{1,1} & 0 & \cdots & y_{2,0} & 0 \\ 0 & 1 & 0 & y_{1,1} & \cdots & 0 & y_{2,0} \\ & & & \ddots & & & \\ 1 & 0 & y_{1,n-1} & 0 & \cdots & y_{2,n-2} & 0 \\ 0 & 1 & 0 & y_{1,n-1} & \cdots & 0 & y_{2,n-2} \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \delta_{11,1} \\ 0 \\ \delta_{12,1} \\ \vdots \\ \delta_{22,2} \end{pmatrix} + \begin{pmatrix} u_{1,2} \\ u_{2,2} \\ \vdots \\ u_{1,n} \\ u_{2,n} \end{pmatrix}.$$

Zur Parameterschätzung sollte eine verallgemeinerte Kleinste-Quadrate-Anpassung (generalized least squares (GLS)-Schätzung) durchgeführt werden, da diese bei restringierten Modellen effizienter als eine OLS-Schätzung ist (vgl. Lütkepohl, 2005, S. 71, 199). Für die Kovarianzmatrix von *vec(U)* gilt:

$$C_{vec(U)} = I_{n-p+1} \otimes C_u,$$

wobei C_u die zeitlich konstante Kovarianzmatrix der Störgrößen u_t bezeichnet. Nach dem GLS-Verfahren ist

$$S(\gamma) = vec(U)' C_{vec(U)}^{-1} vec(U) = [vec(Y) - (Z' \otimes I_k)R\gamma]' (I_{n-p+1} \otimes C_u)^{-1} [vec(Y) - (Z' \otimes I_k)R\gamma]$$

zu minimieren. Mit obigen Rechenregeln ergeben sich folgende Vereinfachungen beim Ausmultiplizieren von $S(\gamma)$:

$$(I_{n-p+1} \otimes C_u)^{-1} \overset{(R3)}{=} I_{n-p+1} \otimes C_u^{-1},$$

$$[(Z' \otimes I_k)R\gamma]' = [R\gamma]'(Z' \otimes I_k)' \overset{(R1)}{=} \gamma' R'(Z \otimes I_k),$$

$$(I_{n-p+1} \otimes C_u^{-1})(Z' \otimes I_k) \overset{(R2)}{=} I_{n-p+1} Z' \otimes C_u^{-1} I_k = Z' \otimes C_u^{-1},$$

$$(Z \otimes I_k)(I_{n-p+1} \otimes C_u^{-1}) \overset{(R2)}{=} Z I_{n-p+1} \otimes I_k C_u^{-1} = Z \otimes C_u^{-1},$$

$$(Z \otimes I_k)(I_{n-p+1} \otimes C_u^{-1})(Z' \otimes I_k) = (Z \otimes C_u^{-1}) \otimes (Z' \otimes I_k) \overset{(R2)}{=} ZZ' \otimes C_u^{-1}.$$

Man erhält:

$$
\begin{aligned}
&S(\gamma) = [vec(Y) - (Z' \otimes I_k)R\gamma]'(I_{n-p+1} \otimes C_u^{-1})[vec(Y) - (Z' \otimes I_k)R\gamma] \\
&= vec(Y)'(I_{n-p+1} \otimes C_u^{-1})vec(Y) - vec(Y)'(I_{n-p+1} \otimes C_u^{-1})(Z' \otimes I_k)R\gamma \\
&- [(Z' \otimes I_k)R\gamma]'(I_{n-p+1} \otimes C_u^{-1})vec(Y) + [(Z' \otimes I_k)R\gamma]'(I_{n-p+1} \otimes C_u^{-1})(Z' \otimes I_k)R\gamma \\
&= vec(Y)'(I_{n-p+1} \otimes C_u^{-1})vec(Y) - vec(Y)'(I_{n-p+1} \otimes C_u^{-1})(Z' \otimes I_k)R\gamma \\
&- \gamma' R'(Z \otimes I_k)(I_{n-p+1} \otimes C_u^{-1})vec(Y) + \gamma' R'(Z \otimes I_k)(I_{n-p+1} \otimes C_u^{-1})(Z' \otimes I_k)R\gamma \\
&= vec(Y)'(I_{n-p+1} \otimes C_u^{-1})vec(Y) - vec(Y)'(Z' \otimes C_u^{-1})R\gamma \\
&- \gamma' R'(Z \otimes C_u^{-1})vec(Y) + \gamma' R'(ZZ' \otimes C_u^{-1})R\gamma.
\end{aligned}
$$

Ein Minimum liegt vor, wenn $\frac{\partial S}{\partial \gamma'} = (0,\ldots,0)$ und die Hessische Matrix $\frac{\partial^2 S}{\partial \gamma \partial \gamma'}$ positiv definit ist. Unter Verwendung der Produktregel $\frac{\partial [a(\gamma)' Ab(\gamma)]}{\partial \gamma'} = b' A' \frac{\partial a}{\partial \gamma'} + a' A \frac{\partial b}{\partial \gamma'}$ erhält man die Gleichung:

$$
\frac{\partial S}{\partial \gamma'} = -vec(Y)'(Z' \otimes C_u^{-1})R - [R'(Z \otimes C_u^{-1})vec(Y)]' + \gamma'[R'(ZZ' \otimes C_u^{-1})R]' + \gamma'[R'(ZZ' \otimes C_u^{-1})R].
$$

Da C_u^{-1} symmetrisch ist und $(ZZ')' = ZZ'$, gilt wegen Regel *(R1)*:

$$
[R'(ZZ' \otimes C_u^{-1})R]' = [R'(ZZ' \otimes C_u^{-1})'R] = [R'(ZZ' \otimes C_u^{-1})R],
$$

$$
[R'(Z \otimes C_u^{-1})vec(Y)]' = vec(Y)'(Z' \otimes C_u^{-1})R.
$$

Daraus folgt:

$$
\frac{\partial S}{\partial \gamma'} = -2vec(Y)'(Z' \otimes C_u^{-1})R + 2\gamma'[R'(ZZ' \otimes C_u^{-1})R]'.
$$

Damit gilt $\frac{\partial S}{\partial \gamma'} = (0,\ldots,0)$, falls

$$
\gamma'[R'(ZZ' \otimes C_u^{-1})R]' = vec(Y)'(Z' \otimes C_u^{-1})R,
$$

$$
\gamma' = vec(Y)'(Z' \otimes C_u^{-1})R\left([R'(ZZ' \otimes C_u^{-1})R]'\right)^{-1},
$$

$$
\gamma = [R'(ZZ' \otimes C_u^{-1})R]^{-1} R'(Z \otimes C_u^{-1})vec(Y).
$$

Bei der letzten Gleichung wurde die Beziehung $(A^{-1})' = (A')^{-1}$ verwendet.

Da $[R'(ZZ' \otimes C_u^{-1})R]$ symmetrisch ist (s. o.: der Ausdruck ist gleich dem transponierten Ausdruck), folgt $\frac{\partial^2 S}{\partial \gamma \partial \gamma'} = 2 \cdot [R'(ZZ' \otimes C_u^{-1})R]$. Diese Matrix ist positiv definit, falls die Zeilen von *Z* linear unabhängig sind:

Zunächst ist C_u^{-1} positiv definit, da C_u positiv definit ist. $ZZ' = ZI_{n-p+1}Z'$ ist positiv definit, da I_{n-p+1} positiv definit ist und ein voller Spaltenrang von Z' vorausgesetzt wird. Damit ist auch $ZZ' \otimes C_u^{-1}$ positiv definit. Schließlich ist $R'(ZZ' \otimes C_u^{-1})R$ positiv definit, da *R* den vollen Spaltenrang hat.

Damit ist der GLS-Schätzer für γ durch

$$\hat{\gamma} = [R'(ZZ' \otimes \hat{C}_u^{-1})R]^{-1} R'(Z \otimes \hat{C}_u^{-1})vec(Y)$$

gegeben. Als Schätzer für die unbekannte Kovarianzmatrix lässt sich

$$\hat{C}_u = \frac{1}{n-p+1-(k \cdot p+1)}(Y-\hat{\delta}Z)(Y-\hat{\delta}Z)'$$

mit dem OLS-Schätzer $\hat{\delta} = YZ'(ZZ')^{-1}$ des nicht-restringierten Gleichungssystems (8) verwenden. Bei dem Schätzer der Kovarianzmatrix zum berechneten restringierten System ist dagegen auf die Subtraktion von $k*p+1$ zu verzichten, da bei jeder der k Gleichungen deutlich weniger als $k*p+1$ Parameter verwendet werden.

Wie im univariaten Fall kann die Anzahl der Verzögerungen mit Hilfe eines Informationskriteriums ermittelt werden. Bis auf konstante Terme sind die Kriterien wie folgt definiert:

$$AIC = \log\left(\det\left(\frac{1}{n-p+1}\sum_{i=p}^{n}\hat{U}_i\hat{U}'_i\right)\right) + \frac{2}{n-p+1}l,$$

$$SIC = \log\left(\det\left(\frac{1}{n-p+1}\sum_{i=p}^{n}\hat{U}_i\hat{U}'_i\right)\right) + \frac{\log(n-p+1)}{n-p+1}l,$$

wobei $\hat{U}_i$ die $(k \times 1)$-Spaltenvektoren der Residuen und l die Anzahl der geschätzten Parameter bezeichnen. Lütkepohl (2005, S. 153 ff.) konnte für ein bivariates Modell die Vorteilhaftigkeit des Schwarzschen Informationskriteriums gegenüber anderen Informationskriterien mit Hilfe einer Simulationsrechnung nachweisen.

4 Prognose

Mit Hilfe der geschätzten Parameter können künftige Werte durch sukzessive Anwendung der Regressionsgleichung ermittelt werden. Im Falle von Gleichung (1) erhält man:

$$\begin{aligned}
(11)\quad & \hat{x}_{n+1} = \hat{\alpha} + \hat{\varphi}_1 \cdot x_n + \hat{\varphi}_2 \cdot x_{n-1} + \ldots + \hat{\varphi}_p \cdot x_{n-p} + \hat{u}_{n+1}, \\
& \hat{x}_{n+2} = \hat{\alpha} + \hat{\varphi}_1 \cdot \hat{x}_{n+1} + \hat{\varphi}_2 \cdot x_n + \ldots + \hat{\varphi}_p \cdot x_{n+1-p} + \hat{u}_{n+2}, \\
& \ldots \\
& \hat{x}_{n+h} = \hat{\alpha} + \hat{\varphi}_1 \cdot \hat{x}_{n+h-1} + \hat{\varphi}_2 \cdot \hat{x}_{n+h-2} + \ldots + \hat{\varphi}_p \cdot \hat{x}_{n+h-p} + \hat{u}_{n+h}, \text{ für } h > p.
\end{aligned}$$

Die Wahrscheinlichkeitsverteilung der Prognosewerte lässt sich über eine Simulation der stochastischen Störgrößen u_i ermitteln. Dabei werden die statistischen Eigenschaften der unbeobachteten Störgrößen üblicherweise mit Hilfe der Residuen geschätzt. Bei Finanzzeitreihen weisen die ermittelten Residuen häufig eine im Vergleich zur Normalverteilung größere Kurtosis (Wölbung) auf. Dies bedeutet ein häufigeres Auftreten von kleinen Residuen in der Mitte und großen Residuen an den Rändern der Verteilung. Krämer und Runde (2000) zeigen, dass bei der Normalisierung von leptokurtischen Verteilungen eine Anpassung mit empirischen Interquantilbereichen wesentlich robuster ist als eine Anpassung mit der empirischen

Standardabweichung. Um diese Effekte zu berücksichtigen, lässt sich mit Hilfe eines Normalverteilungstests prüfen, ob die einzelnen Residuen durch eine an das Maximum aus der empirischen Standardabweichung und dem empirischen Interdezilbereich (Streubreite der mittleren 80% der Fälle) angepasste Normalverteilung mit Mittelwert 0 modelliert werden können:

$$\hat{u}_t \sim \hat{\sigma} \cdot N(0,1); \hat{\sigma} = \sqrt{\frac{n-p+1}{n-p+1-l}} \cdot Max(\hat{s}, \frac{I_{80}}{2{,}564}).$$

Der Quotient 2,564 ist der Interdezilbereich der Standard-Normalverteilung $N(0,1)$. I_{80} bezeichnet den empirisch ermittelten Interdezilbereich, $n + 1$ den Stichprobenumfang, l die Anzahl der Parameter und $\hat{s}$ die empirische Standardabweichung.

Für nicht normalverteilte Residuen lässt sich ein symmetrischer *Kerndichteschätzer* mit Gauß-Kern der Form

$$\hat{f}_s(x) = \frac{1}{2}(\hat{f}(x) + \hat{f}(-x)), \text{ mit } \hat{f}(x) = \frac{1}{(n-p+1) \cdot h} \sum_{i=p}^{n} K\left(\frac{x - \hat{u}_i}{h}\right), K(z) = \frac{1}{\sqrt{2\pi}} \exp(-z^2/2)$$

mit der Bandbreite wird $h = 1{,}06 \cdot \hat{s} \cdot (n-p+1)^{-1/5}$ (vgl. Silverman, 2003) verwenden, falls mit einem Chi-Quadrat-Test die Nullhypothese (Übereinstimmung des Kerndichteschätzers mit der empirischen Verteilung) nicht abgelehnt werden kann. Alternativ kann ein *Bootstrap*-Verfahren verwendet werden, bei dem die Verteilungsfunktion der Störgröße durch zufällige Ziehung der Residuen (mit Zurücklegen) simuliert wird. In diesem Fall wird allerdings die Größe der Störung durch den maximalen Wert der Residuen beschränkt.

Neben der korrekten Ermittlung der Verteilungsfunktion der Störgrößen ist für die Prognose wesentlich, dass keine Autokorrelationen vorliegen. Ansonsten führt die sukzessive Anwendung der Regressionsgleichung gemäß (11) zu fehlerhaften Prognosewerten. Autokorrelationen $\rho_i = Corr(u_t, u_{t-i})$ bis zu einer bestimmten Ordnung m können mit der *Q-Statistik* von Ljung und Box (1978)

$$Q_m = (n-p+1)(n-p+3) \sum_{i=1}^{m} \frac{\hat{\rho}_i^2}{n-p+1-i} \text{ mit } \hat{\rho}_i = Corr(\hat{u}_t, \hat{u}_{t-i}).$$

überprüft werden. Die Nullhypothese besagt, dass bis zu einer vorgegebenen Verzögerung keine Autokorrelationen vorliegen: $\rho_1 = \rho_2 = ... = \rho_m = 0$.

Zum Test der Stabilität der Zeitreihenmodelle bietet sich der *Prognosetest nach Chow* (1960) an. Dazu schätzt man ein neues Regressionsmodell für eine bis zu einem vorgegebenen Zeitpunkt verkürzte Zeitreihe mit denselben Regressoren wie das zu prüfende Modell. Die Nullhypothese besagt, dass die Parameter beider Modelle gleich sind. Die Teststatistik hat die Form:

$$F = \frac{S - S_1}{S_1} \cdot \frac{n_1 - l}{n + 1 - n_1},$$

wobei n_1 die Länge der verkürzten Zeitreihe, S die Summe der quadrierten Residuen, S_1 die Summe der quadrierten Residuen des an die verkürzte Zeitreihe angepassten Modells und l die Anzahl der Regressoren bezeichnen.

Bei den multivariaten Modellen sind die Prognosewerte analog zu (11) zu berechnen. Da in die Berechnung der Prognosewerte die Störgrößen vorheriger Perioden eingehen, dürfen wie im univariaten Fall keine Autokorrelationen und zusätzlich keine Kreuzkorrelationen mit verzögerten Störgrößen auftreten. Die Verallgemeinerung der Q-Statistik von Ljung und Box für VAR-Modelle ist für die Ordnung m durch

$$Q_m = (n+1)^2 \sum_{i=1}^{m} \frac{1}{n+1-i} \mathrm{tr}(\hat{C}'_i \hat{C}_0^{-1} \hat{C}'_i \hat{C}_0^{-1}) \text{ mit } \hat{C}_i = \frac{1}{n+1} \sum_{j=i+1}^{n+1} \hat{U}_j \hat{U}'_{j-i}$$

gegeben (Lütkepohl, 2004, S. 127). Allerdings sind kontemporäre Abhängigkeitsstrukturen der Störgrößen zu berücksichtigen. Im Falle einer multivariaten Normalverteilung können die Störgrößen über eine *Cholesky-Faktorisierung* der Kovarianzmatrix $\hat{C}_u = GG'$ berechnet werden: $\hat{U}_t = G\varepsilon_t$. Dabei bezeichnen G eine untere Dreiecksmatrix und ε_t einen $(k \times 1)$-Vektor mit k standard-normalverteilten Zufallsvariablen. Eine solche Faktorisierung existiert für alle positiv definiten Matrizen, also auch für die Kovarianzmatrix. Man überzeugt sich leicht, dass $\hat{U}_t = G\varepsilon_t$ die gewünschte Kovarianzmatrix besitzt:

$$E(\hat{U}_t \hat{U}'_t) = E(G\varepsilon_t (G\varepsilon_t)') = E(G\varepsilon_t \varepsilon'_t G') = GE(\varepsilon_t \varepsilon'_t) G' = GI_k G' = GG' = \hat{C}_u .$$

Falls keine multivariate Normalverteilung vorliegt, ist die Verwendung der Kovarianzmatrix zur Beschreibung der Abhängigkeitsstrukturen nicht geeignet. Stattdessen bietet sich die *Copula-Methode* an (Embrechts et al. 2005). Alternativ kann wie im univariaten Fall das Bootstrap-Verfahren verwendet werden, wobei bei jeder Ziehung der Residuen derselbe Zeitpunkt für die verschiedenen Zeitreihen zu wählen ist, damit die Abhängigkeitsstrukturen erhalten bleiben.

Eine Verallgemeinerung des Chow-Prognosetests für VAR-Modelle findet man in Lütkepohl (2004, S. 136).

5 Anwendungsbeispiel

Nachfolgend sollen univariate und multivariate autoregressive Modelle für die in Tabelle 1 beschriebenen Zeitreihen ermittelt werden.

Tabelle 1 Zeitreihen

Zeitreihe	Bedeutung	Quelle	Zeitraum
cpir_de	Änderungsrate pro Quartal des Verbraucherpreisindex in Deutschland	Eurostat bis November 2002, Statistisches Bundesamt ab Dezember 2002	Q1 1984 – Q4 2004
cpir_uk	Änderungsrate pro Quartal des Verbraucherpreisindex in Großbritannien (vor 1997 durch das Office for National Statistics geschätzt)	Office for National Statistics	Q1 1984 – Q4 2004

In Tabelle 2 sind die p-Werte der Dickey-Fuller-Tests aufgeführt.

Tabelle 2 p-Werte der Dickey-Fuller-Tests

Zeitreihe x	□□ (x)	$\Phi_3(x)$	□□(x)	$\Phi_1(x)$	$\tau(x)$	τ (Δx)
cpir_de	0,61	> 0,1	0,29	> 0,1	0,29	0,000
cpir_uk	0,49	> 0,1	0,54	> 0,1	0,27	0,000

Die nicht signifikanten p-Werte bezüglich $\tau_\tau(x)$ und $\Phi_3(x)$ besagen, dass die Nullhypothese eines Random Walks mit $\beta = 0$ für Modell (7) nicht zu verwerfen ist. Wegen der nicht signifikanten p-Werte bezüglich $\tau_\mu(x)$ und $\Phi_1(x)$ kann die Nullhypothese eines Random Walks mit $\alpha = 0$ für Modell (6) nicht abgelehnt werden. Daher bleibt Modell (5) zu prüfen. Wegen der nicht signifikanten p-Werte bezüglich $\tau(x)$ und der hoch-signifikanten p-Werte bezüglich $\tau(\Delta x)$ können alle Zeitreihen als integriert von der Ordnung 1 ohne linearen Trend β und ohne Interzept α modelliert werden. Tabelle 3 zeigt die Ergebnisse einer OLS-Schätzung.

Tabelle 3 AR-Modelle (Standardabweichungen der Parameter in Klammern)

$\Delta\text{cpir\_de}_t = -0{,}729\,(0{,}100)\,\Delta\text{cpir\_de}_{t-1} - 0{,}835\,(0{,}087)\,\Delta\text{cpir\_de}_{t-2} - 0{,}472\,(0{,}099)\,\Delta\text{cpir\_de}_{t-3}$
$\Delta\text{cpir\_uk}_t = -0{,}925\,(0{,}082)\,\Delta\text{cpir\_uk}_{t-1} - 0{,}839\,(0{,}092)\,\Delta\text{cpir\_uk}_{t-2} - 0{,}687\,(0{,}080)\,\Delta\text{cpir\_uk}_{t-3}$

Tabelle 4 fasst die Ergebnisse der Residuenanalyse zusammen. Der Ljung-Box-Test wurde für bis zu 16 Verzögerungen durchgeführt. Der Chow-Prognosetest wurde für alle Zeitpunkte ab dem 1. Quartal 1995 durchgeführt. Für beide Tests sind die minimalen p-Werte angegeben. Wie man sieht, sind auf einem Signifikanzniveau von 5 % keine Autokorrelationen für die Residuen anzunehmen, und die Modelle können nach dem Chow-Prognosetest als stabil betrachtet werden. Zur Simulation der Störgrößen können nach dem Anderson-Darling-Test (Stephens, 1974) Normalverteilungen verwendet werden.

Tabelle 4 p-Werte der Residuen-Tests

	Anderson-Darling-Test	Ljung-Box-Test	Chow-Prognosetest	$\hat{\sigma}$
cpir_de	0,77	> 0,44	0,32	0,0041
cpir_uk	0,21	> 0,23	0,54	0,0060

Tabelle 5 zeigt die Ergebnisse der GLS-Schätzungen für das VAR-Modell.

Tabelle 5 VAR-Modell (Standardabweichungen der Parameter in Klammern)

$\Delta\text{cpir\_de}_t = -0{,}776\,(0{,}087)\,\Delta\text{cpir\_de}_{t\text{-}1} - 0{,}801\,(0{,}080)\,\Delta\text{cpir\_de}_{t\text{-}2} - 0{,}137\,(0{,}045)\,\Delta\text{cpir\_uk}_{t\text{-}2} - 0{,}428\,(0{,}090)\,\Delta\text{cpir\_de}_{t\text{-}3}$
$\Delta\text{cpir\_uk}_t = -0{,}908\,(0{,}076)\,\Delta\text{cpir\_uk}_{t\text{-}1} - 0{,}830\,(0{,}088)\,\Delta\text{cpir\_uk}_{t\text{-}2} - 0{,}687\,(0{,}075)\,\Delta\text{cpir\_uk}_{t\text{-}3}$

Tabelle 6 fasst die Ergebnisse der Residuenanalyse zusammen. Wie im univariaten Fall wurde der Ljung-Box-Test für bis zu 16 Verzögerungen und der Chow-Prognosetest für alle Zeitpunkte ab dem 1. Quartal 1995 durchgeführt. Für beide Tests sind die minimalen p-Werte angegeben. Wie man sieht, muss die Hypothese einer multivariaten Normalverteilung der Residuen nach dem Test von Doornik and Hansen (1994) verworfen werden. Nach dem multivariaten Ljung-Box-Test sind keine Auto- und Kreuzkorrelationen der Residuen für bis zu 16 Verzögerungen anzunehmen, und nach dem Chow-Prognosetest kann das VAR-Modell als stabil betrachtet werden.

Tabelle 6 p-Werte der Residuen-Tests

	Doornik-Hansen-Test	Ljung-Box-Test	Chow-Prognosetest
(cpir_de, cpir_uk)	0	> 0,34	0,35

Nach 10.000 Simulationsläufen ergeben sich für die univariaten Modelle folgende Werte (Werte in eckigen Klammern beziehen sich auf die mit dem Bootstrap-Verfahren simulierten Störgrößen, die übrigen Werte wurden mit normalverteilten Störgrößen ermittelt):

Tabelle 7 cpir_de

	nach einem Quartal	nach 16 Quartalen
□	0,66 % [0,65 %]	0,58 % [0,60 %]
□	0,41 % [0,39 %]	0,68 % [0,65 %]

Tabelle 8 cpir_uk

	nach einem Quartal	nach 16 Quartalen
□	0,11 % [0,10 %]	0,51 % [0,41 %]
□	0,60 % [0,56 %]	0,94 % [0,90 %]

Die Abbildungen 1 und 2 verdeutlichen die „Verbreiterung“ der Häufigkeitsverteilung (10.000 Simulationsläufe mit normalverteilten Störgrößen) mit zunehmendem Prognosehorizont.

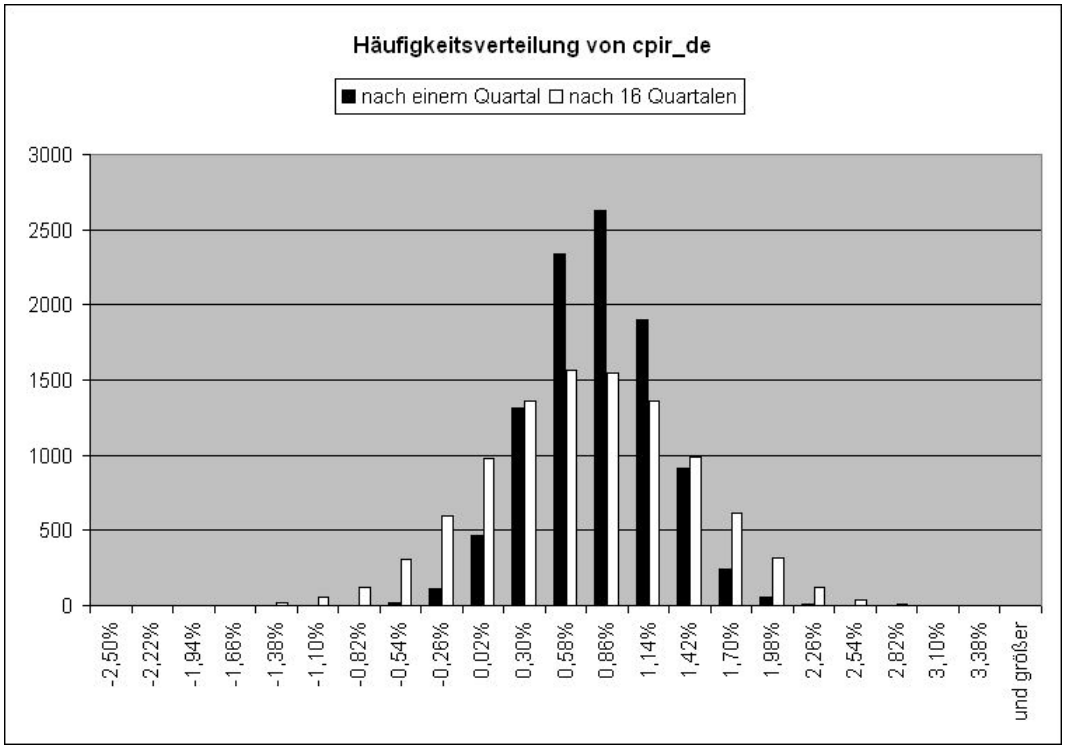


Abbildung 1

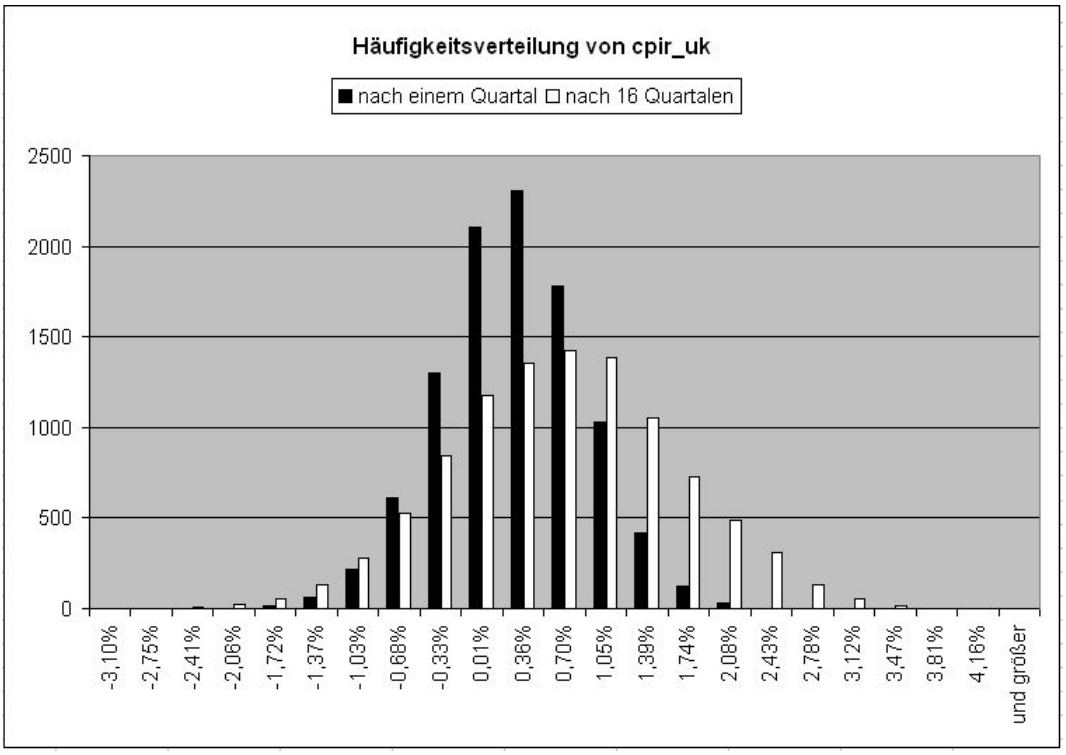


Abbildung 2

Die Abbildungen 3 und 4 zeigen den Einfluss des angewandten Verfahrens zur Simulation der Störgrößen nach 16 Quartalen. Die Normalverteilung betont die Ränder etwas stärker als das Bootstrap-Verfahren.

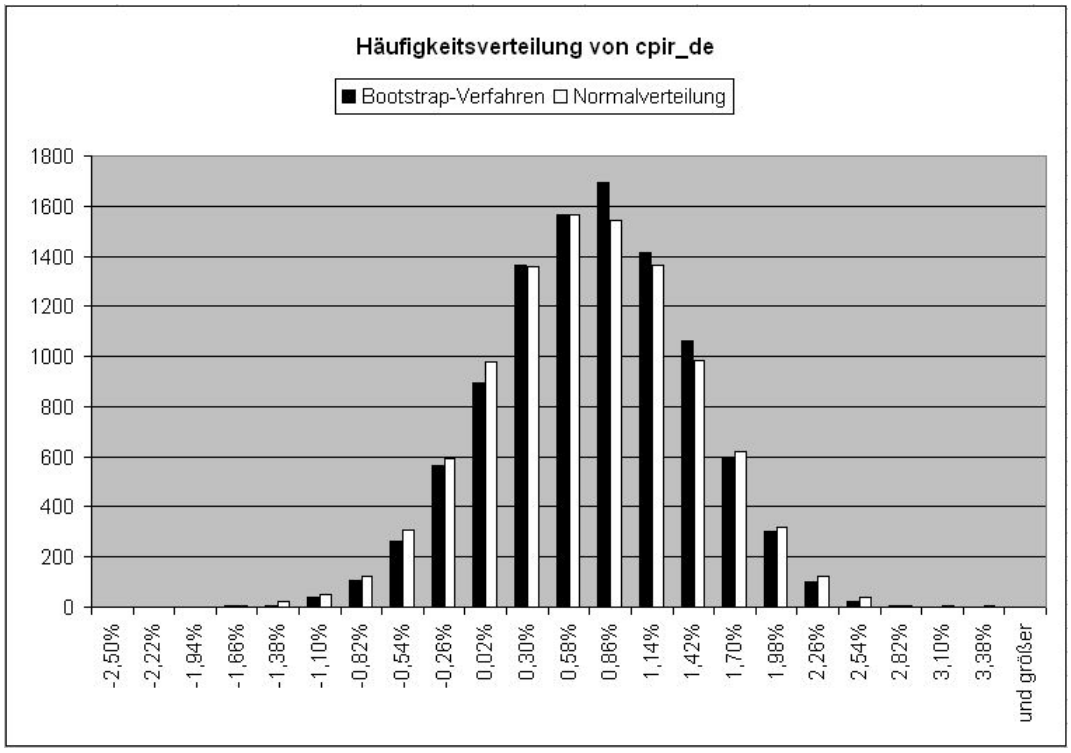


Abbildung 3

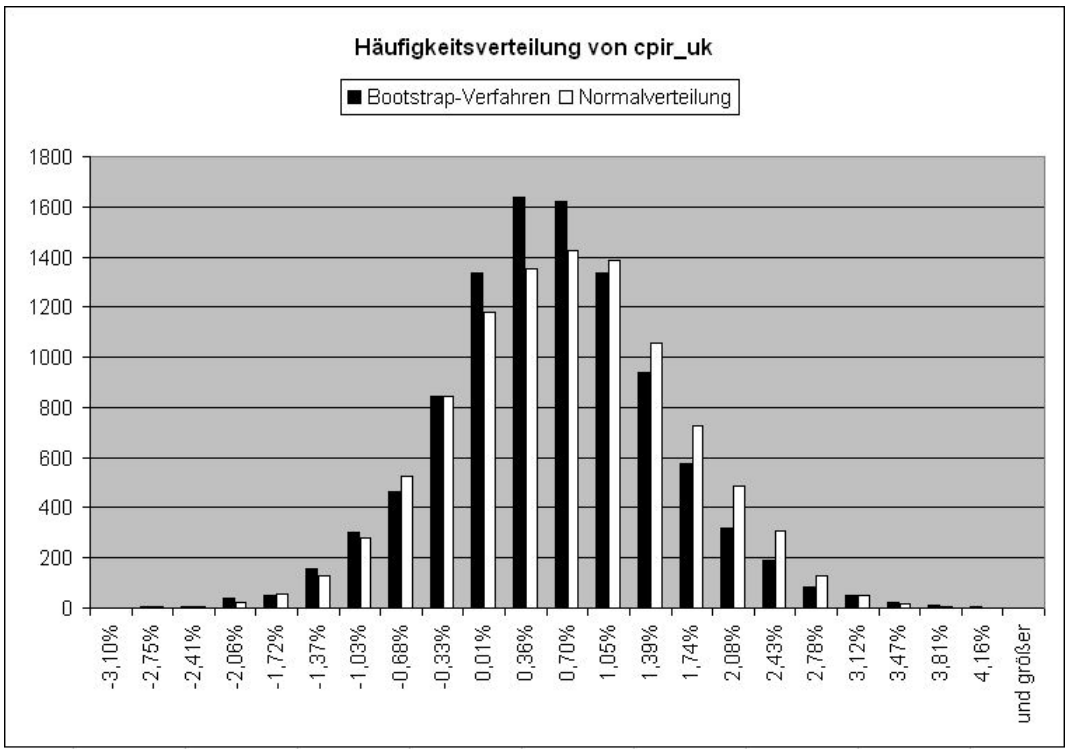


Abbildung 4

Nach 10.000 Simulationsläufen ergeben sich für das VAR-Modell folgende Werte (Störgrößen wurden mit dem Bootstrap-Verfahren simuliert):

Tabelle 9 cpir_de

	nach einem Quartal	nach 16 Quartalen
□	0,70 %	0,56 %
□	0,36 %	0,64 %

Tabelle 10 cpir_uk

	nach einem Quartal	nach 16 Quartalen
□	0,11 %	0,41 %
□	0,58 %	0,92 %

Beim Vergleich mit den Tabellen 7 und 8 stellt man nur kleine Abweichungen fest, was nicht verwundert, da die Regressionsgleichungen der univariaten Modelle gemäß den Tabellen 3 und 5 eine große Übereinstimmung mit dem VAR-Modell aufweisen.

6 Schlusswort

Bei der Interpretation der Zeitreihenprognosen ist darauf zu achten, dass die gewonnenen Simulationsmodelle lediglich eine Approximation der Realität darstellen können. Zum einen basieren sie auf Vergangenheitsdaten und unterstellen implizit, dass sich die Zukunft aus Vergangenheitsdaten extrapolieren lässt und innerhalb des Prognosezeitraums keine Strukturbrüche auftreten werden. Zum anderen wird bei den autoregressiven und Vektor-autoregressiven Zeitreihenmodellen ein linearer Zusammenhang der zukünftigen Werte mit Werten der Vergangenheit unterstellt.

Literaturverzeichnis

Akaike, H. (1973): Information theory and an extension of the maximum likelihood principle. In: Petrov, B. N./Csáki (Hrsg.): Proceeding of the Second International Symposium on Information Theory, Akadémiai Kiadó, Budapest, S. 267-281.

Allen, P. G./Fildes, R. (2007): Levels, differences and ECMs - Principles for Improved Econometric Forecasting. In: Oxford Bulletin of Economics and Statistics, erscheint demnächst, http://www.umass.edu/resec/faculty/allen/obes.pdf, Abruf am 2007-07-12

Brüggemann, R./Lütkepohl, H.: Lag Selection in Subset VAR Models with an Application to a U.S. Monetary System. In: Friedmann, R./Knüppel, L./Lütkepohl, H. (Hrsg.): Econometric Studies: A Festschrift in Honour of Joachim Frohn. LIT, Münster, S. 107-128.

Chow, G. C. (1960): Tests of equality between sets of coefficients in two linear regressions. In: Econometrica, Vol. 28, S. 591-605.

Christoffersen, P. F./Diebold, F. X. (1997): Cointegration and Long-Horizon Forecasting. NBER Working Paper No. T0217, http://ssrn.com/abstract=226625.

Dickey, D. A./Fuller, W. A. (1979): Distribution of the Estimators for Autoregressive Time Series with a Unit Root. In: Journal of the American Statistical Association, Vol. 74, S. 427-431.

Dickey, D. A./Fuller, W. A. (1981): Likelihood ratio statistics for autoregressive time series with a unit root. In: Econometrica, Vol. 49, S. 1057-1072.

Doornik, J. A./Hansen, H. (1994): A practical test of multivariate normality. Unveröffentlichtes Manuskript, Nuffield College, University of Oxford.

Embrechts, P./McNeil, A./Straumann, D. (2002): Correlation and dependence in risk management: properties and pitfalls. In: Dempster, M. A. H. (Hrsg.): Risk Management: Value at Risk and Beyond, Cambridge University Press, Cambridge, S. 176 - 223.

Engle, R. F./Granger, C. W. J. (1987): Co-integration and Error Correction: Representation, Estimation and Testing. In: Econometria, Vol. 55, S. 251-276.

Engle, R. F./Yoo, B. S. (1987): Forecasting and testing in cointegrated systems. In: Journal of Econometrics Vol. 35, S. 143-159.

Granger, C. W. J. (1981): Some properties of time series data and their use in econometric model specification. In: Journal of Econometrics, supplement 16, S. 121-130.

Granger, C. W. J./Newbold, P. (1974): Spurious Regression in Econometrics. In: Journal of Econometrics, Vol. 2, S. 111-120.

Krämer, W./Runde, R. (2000): Peaks or tails - What distinguishes financial data? In: Empirical Economics, Vol. 25, No. 4, S. 665-671.

Ljung, G. M./Box, G. E. P.(1978): On a Measure of Lack of Fit in Time Series Models. In: Biometrika, Vol. 65, No. 2, S. 297-303.

Lütkepohl, H. (2004): Vector Autoregressive and Vector Error Correction Models. In: Lütkepohl, H./Krätzig, M. (Hrsg.): Applied Time Series Econometrics, Cambridge University Press, Cambridge, S. 86 – 158.

Lütkepohl, H. (2005): New Introduction to Multiple Time Series Analysis. Springer, Berlin, Heidelberg, New York.

Schwarz, G. (1978): Estimating the Dimension of a Model. In: Annals of Statistics, Vol. 6, S. 461-464.

Silverman, B. W. (2003): Density Estimation for Statistics an Data Analysis, Chapman-Hall, London, S. 45.

Stephens, M. A. (1974): EDF Statistics for Goodness of Fit and Some Comparisons. In: Journal of the American Statistical Association, Vol. 69, S. 730-737.

Tiao, G. C./Tsay, R. S./Wang, T. (1993): Usefulness of Linear Transformations in Multivariate Time-Series Analysis. In: Empirical Economics, Vol. 18, S. 567-593.

Quality Assurance Methods and the Open Source Development Model

Heinz D. Knoell
Leuphana University of Lueneburg
Lueneburg, Germany
Knoell@uni.leuphana.de

Tobias Otte
University of Wolverhampton
School of Computing and Information Technology
Wolverhampton, UK
Tobias.Otte@wlv.ac.uk

Robert Moreton
University of Wolverhampton
School of Computing and Information Technology,
Wolverhampton, UK
R.Moreton@wlv.ac.uk

Abstract

Open Source Software (OSS) reached a remarkable popularity within the last years not least because of renowned products such as Linux, the Apache Web Server or the Mozilla project. Under the Open Source Software Development (OSSD) model products are launched in rapid succession and with high quality, without following traditional quality practices of accepted software development models (Raymond, 1999). Furthermore, some OSSD projects challenge established Quality Assurance (QA) approaches, claiming to be successful through partially contrary techniques.

The aim of this research is to improve the understanding of Quality Assurance practices under the OSSD model. A survey research method is used to gain empirical evidence about applied QA practices in mid-size to large OSS projects. A further evaluation of "successful" projects results that well structured and organized development processes are applied in the OSSD. The findings provide evidence for Raymond's lifecycle and show that OSS projects leverage their communities effectively.

1 Introduction

Open Source Software (OSS) is developed by freely participating programmers, who distribute source code in a collaborative, virtually and geographically distributed environment, communicating over the Internet (Raymond, 1999, Feller and Fitzgerald, 2000). In contrast to proprietary software development, which is classified by plan, schedules, resources and deliverables, the OSS model starts with an idea, followed by a more prototypical approach with frequent release cycles. The OSS momentum is driven by the participant's motivation, the free availability of the source code, ongoing interactive tasks and

a continuous feedback by the community. The Open Source Software Development (OSSD) model uses unconventional methods, such as the involvement of large development communities for coding. This contradicts Brooks' law (Brooks, 1995), which faces an increased complexity due to exponential growth of communication and co-operation with an increasing project size. However, the OSSD model delivers successful products, such as Linux, Apache Tomcat or Mozilla, which appear to be high quality. The lasting success of the OSSD model delivering superior products makes it important to draw further attention on this phenomenon.

The research explores the following questions: How is Quality Assurance (QA) under the OSSD model approached? What practices and methods are applied in mid- to large-sized OSS projects and what key practices do we learn from successful approaches? In the recent years, research about QA in OSS evolved, but only a few empirical studies exist. This research paper explores applied QA practices and provides empirical evidence about software quality assurance methods in OSS projects. Furthermore, it analyses successful projects in order to find a common pattern, which distinguishes these projects from the sample and indicates key processes that contribute to their project success.

2 Quality Assurance under the OSSD model

Software Quality Assurance (SQA) is defined as a set of systematic activities providing evidence of the ability of the software process to produce a software product that is fit to use (Schulmeyer and McManus, 1999). It is assumed that software quality is built into the product through the use of a process that has quality built in. Thus, SQA must be an aspect of all software development activities (Dunn, 1990).

Fundamental empirical studies about QA activities under the OSSD model can be found in the literature by Zhao and Elbaum (2000, 2003), Halloran and Scherlis (2002), Koru and Tian (2004) or Michlmayr (2005). The recent studies confirmed the uniqueness of the OSS model and provided evidence for Raymond's lifecycle. They show that user participation is extremely high, defect-handling processes follow mainly structured approaches, testing takes a significant portion of the software life cycle and there is a high usage of configuration and bug tracking tools. However, project documentation is often rare, design documents are lacking, source code may remain unmaintained when developers leave the project or testing faces complexity issues, in case developers may have limited access to diverse platform configurations. Michlmayr (2005) examined some OSSD processes in relation to project success and showed that projects that are more successful make more use of version control tools, systematic testing and effective communication through the deployment of mailing lists.

The studies provide useful information and constitute the basis for this research. An empirical view of QA practices within the development lifecycle in conjunction with project success measures is missing. Our survey was conducted by combining quality characteristics with process success measures to identify key practices in OSSD projects.

3 Research Method

Our survey research is undertaken following an interpretative tradition using a social relativism paradigm. The major objective of the survey is to find quality criteria regarding

development processes, defect handling and testing techniques, documentation as well as infrastructure and quality assurance criteria. The questionnaire was enriched with project success measures as suggested by Crowston (2006). The target group of the survey consists of individual developers or project managers contributing to mid- or large-sized OSS projects.

A multimode approach is selected, which combines e-mail based communication with the participants and a web-based survey for data collection. The survey was executed in the period from 13th of June until 10th of July 2007. In total 427 participants responded to the survey, 11 records were incomplete or invalid, which results a response rate of 8.2%. The main statistical analysis was done with SPSS 15.0, while some minor analyses were done with Excel.

4 Results

The multimode approach turns out as efficient. It follows the collaborative approach in a distributed development environment, using mailings and freely accessible online tools. Open-ended questions provided a detailed feedback about applied practices and further recommendations. In the following, the research results are discussed and the key aspects in conjunction to project success are shown.

4.1 General Information

The projects are grouped according to their size in lines of code (LOC) into mini 3.1% (<1000), small 26.2% (1,000-10,000), medium 36.8% (10,000-100,000) and large 23.3% (>100,000). Around 10% of the respondents could not classify their project size for any reason. The main proportion of the projects belong to “Software Development” (20.7%) or “Internet” (17.8%), while other application types, such as Office, Database, ERP/Financials, Education, Networking, Entertainment or Communications are equally represented with a proportion of around 5% for each type.

The majority of projects (55.8%) are developed by small groups of 2-5 core developers, 10.3% have only one developer, 17.8% of the sample have groups of 6-10 developer and 8.7% have more than twenty developers. These finding show evidence of much larger development groups as observed by Zhao and Elbaum (2003).

Only 26.2% projects of the sample responded to have less than ten users, 33.4% have 10-50 users, 9.1% have 50-100 user and 31.2% argue to have more than 100 users. These figures base upon the participant’s evaluation and represent only estimations. This study shows the existence of much smaller user groups compared to Zhao and Elbaum (2003) who reported that 59% of the projects had more than 50 people, while this survey results 40.3%.

Evidence for a growth of project community with the project size could be observed. 86.5% of the mini projects stated to have less than 50 users. But there is a significant growth of the community with an increasing project size, as 77.7% of projects with more than 20 developers stated to have a large community (<100 users) as shown in Figure 1.

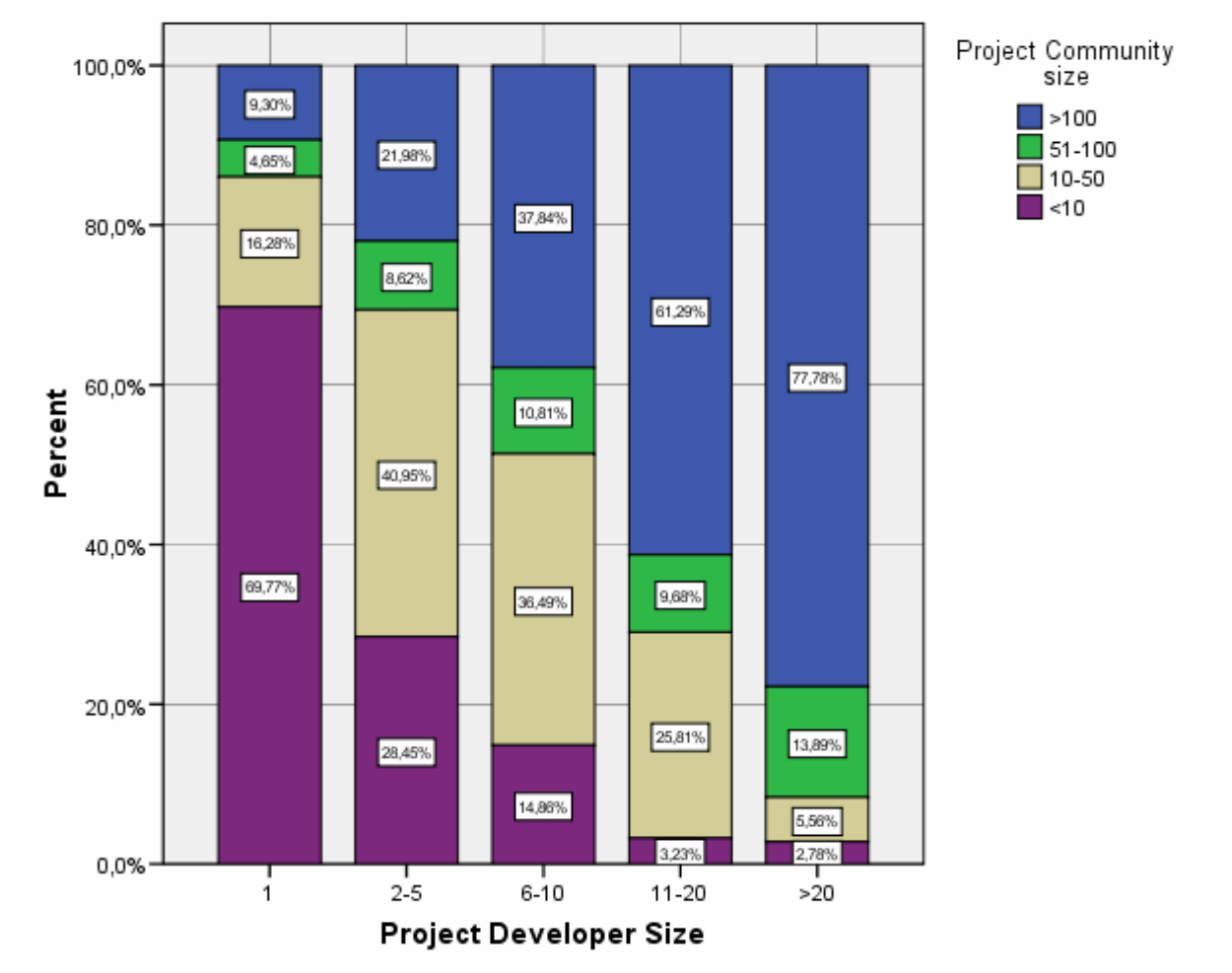


Figure 1. Community Size to Developer Team Size

Furthermore, projects have a major growth of their community size with the time in the market. Around 55% of the projects have less than 10 users in their first year, while 58.02% have more than 50 users in their fourth year.

The completion of project status from “beta” toward a productive status reaches 57.8% of the projects after the first year on the market.

The participant’s level of knowledge in software development is extremely high, as more than 75% state to have above five years development experience. Massey (2003) emphasizes the importance of an experienced team, as professional developers provide accurate feedback. The knowledge transfer to new participants lasts averagely 2-3 weeks. Only mini projects report an “On boarding” time of about one week, while especially large projects need 4-5 weeks and more. The knowledge transfer processes are more common in larger projects. However, only 36% in large projects do claim to have them.

In small development teams (2-5 developers), participants adopt multiple roles. Even in larger teams or projects participants have got multiple roles, which confirms the findings of Jensen and Scacchi (2005) that versatile and fluid roles are specific in the OSSD. Developers are mainly motivated by personal needs (36.5%), 27.2% by company needs and 22.1% by community needs. However, in relation to project size company needs become more important with project growth. This shows a higher commercial interest in larger projects.

More than 54% of the participants work part-time. Nevertheless, large projects show a high proportion of full-time participants, which also corresponds to an increased company motivation in this area.

The survey results that developer fluctuation seems to be lower than expected, although OSS developers are freely participating or leaving. Mature projects, which are more than four years in the market, report to have less than 1% of new participants with minor development experience (less one year). It can be assumed that participants attend projects over a long period, while participants experience matures with the project or simply projects attract only professionals.

4.2 Development

The average proportions of the development activities in the lifecycle, such as design (23.6%), coding (49%) and testing (28.9%) show an almost uniformly distributed picture independent to the project size. However, the proportion of coding efforts increases to the disadvantage of testing with project growth.

Around 38.9% of the projects stated to have code changes of about 10-20% between major releases as depicted in Figure 2. Averagely 24.3% of the code is reused. Branches, which are up to 100% clones, might deform the result upwards. Roughly, 55% of the large projects considered modularity already during design. Vice versa 58.3% of the mini projects reconsidered their code modularity during development and 8.3% had even no modular development at all. That leads to the conclusion that mini projects start from scratch and may improve their modularity with product growth.

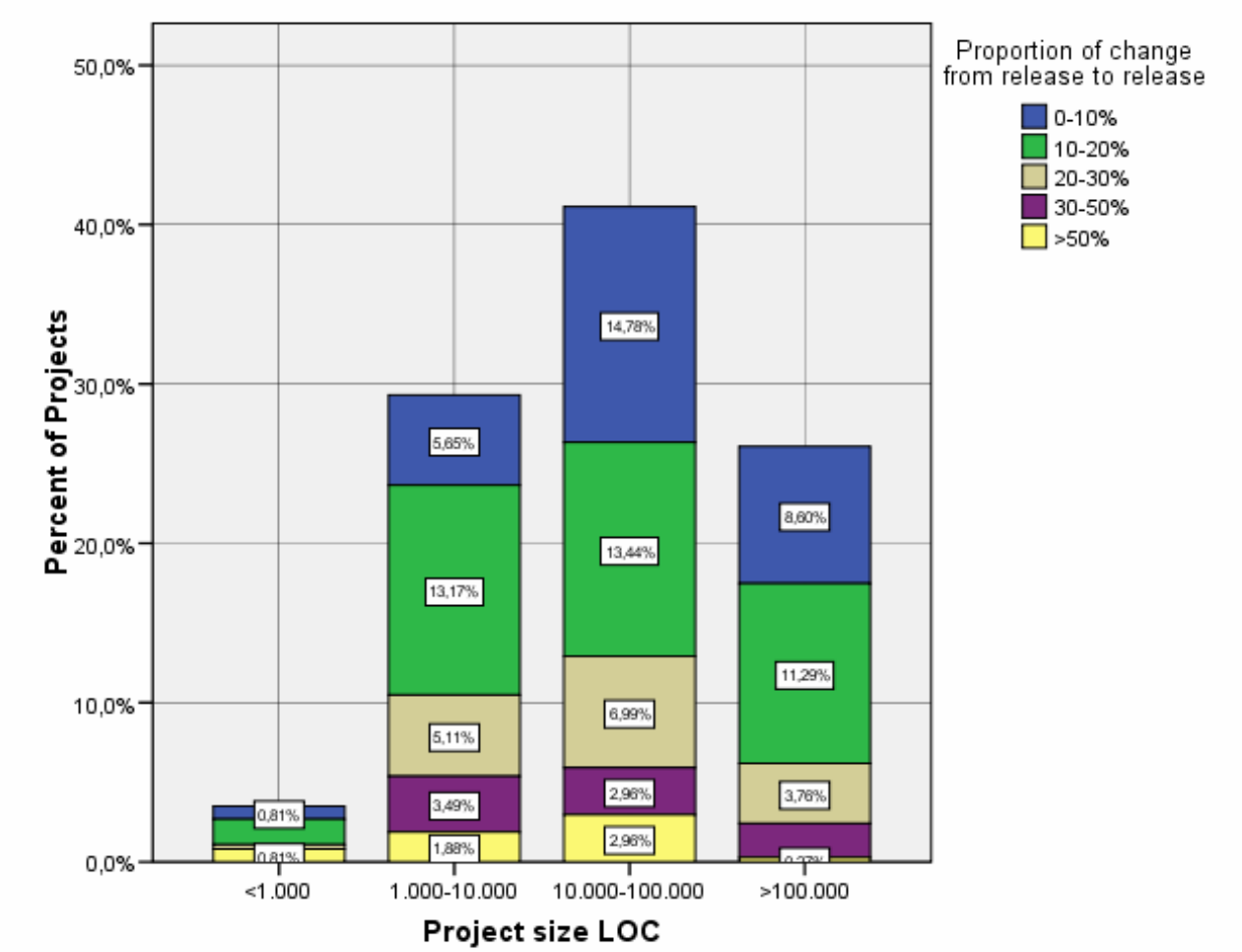


Figure 2. Code change between major releases in relation to project size

The estimated level of abandoned code averages about 12% per project. Projects with an increasing level of abandoned code reported more quality issues and side effects than others, as depicted in Figure 3. These results confirm Michlmayr (2005), who observed certain quality issues due to unsupported code.

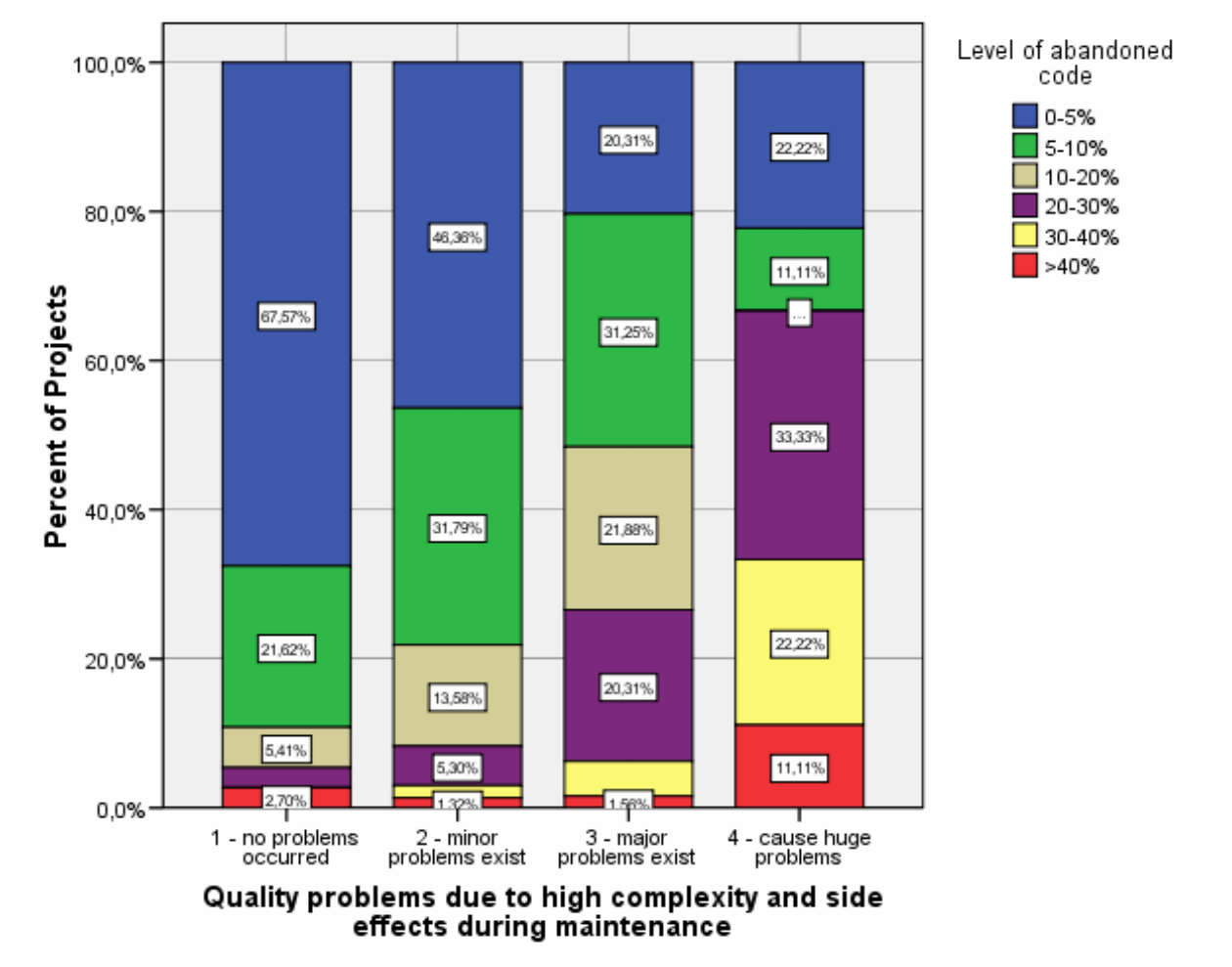


Figure 3. Code complexity to level of abandoned code

Most of the projects (72.1%) follow a feature based release strategy, which is triggered by readiness of the code. However, there is a growing tendency of hybrid approaches as combination of time and feature based strategies. In general, the observed release frequency is lower as expected, as 5.3% of the projects once or twice every fortnight, 14.4% once a month, 29.6% once a quarter, 32.7% releases once half a year and roughly 18% have even longer intervals. In contrast to Raymond's (Raymond, 2001) often-cited statement, "release early and release often" the observed release frequency is much lower, compared to the study of Zhao and Elbaum (2003) in which 43% of the projects release every month. On the other hand, too short release cycles could affect the software quality and frustrate end-users due to less stability (Porter *et al.*, 2006).

4.3 Testing

The average testing time compared to the whole development time averages 38.6%. This reflects the findings of Zhang and Pham (2000) who argued that projects spend 20-40% in testing. More than half of the projects (52.3%) follow a structured testing approach.

Around 57% of the projects reported a high user testing efficiency, as the users found major defects or hard bugs. The remarkable positive team communication may contribute to this effect, as 72.9% of the participants reported to have a direct and efficient feedback between

developer and user. Especially larger projects rate their communication as more effective as smaller ones. This contradicts Michlmayr (2005) findings of communication problems in mid-size to large projects.

In comparison to code reviews by developers (47.2%), user testing (55.7%) seems to have the same or even a higher importance for projects. OSS projects frequently apply code readings (69.9%) or walkthroughs (61.4%). Peer reviews (50.5%) are not as frequently applied. However, Raymond (2001) emphasized its importance, as "the high level of quality is partly due to the high degree of peer reviews and user involvement".

An important part of the user involvement are user suggestions. 48.5% of all respondents mention that user suggestions bring the design forward, while 49.2% feel that they are only sometimes useful. It shows that OSS projects work as claimed by Raymond (2001) and that they listen to their customers.

Around 46% of the projects perform corrective actions before a contributed code is committed to the repository. Especially large projects have strict quality checks, as 75% of them report to rework or even reject inappropriate code.

4.4 Defect Handling

Projects either introduce defect-handling processes with project start (31.7%) or with start of the development activities (32.5%). The majority of projects (88.9%) track source code defects, while tracking of requirements (41.8%), design (43.0%) or documentation issues (48.5%) are underpart. More than 50% of the projects receive useful bug descriptions and only 13.8% complain about insufficient information. These findings show a contrary picture to Michlmayr (2005) who argues that developers see an increase of useless bug reporting with less technical skilled users. More than 77% of the projects classify their defects for status tracking. This shows an improved situation in contrast to Villa (2003), who observed absent information about priorities or severities for defect handling.

4.5 Documentation

The majority of projects (66.6%) use coding and development style guidelines, while 58% of the projects have common process documentation. However, process descriptions gain importance with project growth. For instance, 71% of large projects claim to have at least minor process documentation. More than 68% of the projects state to have product documentation or at least a draft. This study results a far improved position compared to Michlmayr (2005), who observes a lack of user and developer documentation.

4.6 Infrastructure

The observed tool usage in OSS projects is relatively high. For instance, 87.2% of the projects use source code control tools, 76.2% bug-tracking tools and 73.5% mailing lists, 51.9% instant messaging, while only 36.5% apply test support tools. Projects that apply bug-tracking tools report an averagely higher defect reporting quality. For example, 50% of the projects that report not to get any useful information at all from their participants do not make use of a supporting tool. A further analysis of tool usage in relation to project size shows that large

projects in general make more use of supporting tools, than smaller ones. For instance, 60% of the mini projects do not benefit from using bug-tracking tools.

4.7 Quality Assurance

Only one-third of the projects apply QA practices although they become more important with project growth. An analysis of applied QA practices result that several projects discuss quality issues within the core development team or their community. A number of projects conduct frequently different kind of code reviews and implement structured test approaches, defect handling as well as test management. Quite a few projects established live sessions, smoke testing as well as automated nightly testing to improve software quality. Some projects restructure their organization and setup QA teams, who contribute full-time to the project.

Only 20% of the projects reply that their QA team performs actions, resulting from interviews or reviews. For instance, projects increase their communication activities, adopt their development approach and apply refactoring processes, rework flawed code, introduce additional code reviews, improve their defect handling and bug reporting processes, increase testing prior to a release or even reconsider the release scope, such as the definition of exit criteria or stopping the release.

The respondents suggest the following quality improvements of the OSSD model: The enhancement of the community due to the attraction of knowledgeable users, professional full-time developers, high integration of users or developers and even paid contributors. Moreover, the project management needs to be experienced and has to provide leadership and guidance for their projects. The whole community must be conscious about quality to achieve significant improvements. However, large projects require an independent QA team. The QA team needs to perform checks and verify that processes or guidelines are kept, which can be supported by quality assessment tools. Projects should have detailed process documentation and development guidelines, comprising style or coding standards. A technical design phase prior to the development could reduce shortcomings in architecture or development approach. The release approach must be appropriate to the projects, regarding strategy, frequency and scoping. The respondents see room for improvement in testing processes, such as test efficiency, improvements of code reviews or the use of tools for automated testing. The tools should be OSS products themselves to lower barriers for communities. Furthermore, tools should focus on a higher integration, while there is another demand for easy handling and simplicity.

5 Project Success Examination

The measurement of project success is an elusive target and depends not least of the beholders position. To gain reasonable results for the project success examination, more mature projects are an ideal object of study, as it is assumed they abolish shortcomings and improve their processes over the time. Independent of their size (in LOC), in this study projects are considered as successful, which have a productive release version, are more than two years in the market, whose developer teams consist of more than five developers and which have a community above fifty participants. In this research, the term success describes the group of selected projects. However, this assumption does not reflect the actual project success.

These success selection criteria apply to one-fifth of the surveyed projects. These projects are object to the following analysis. It results that a higher motivation of the participants due to company needs (35.7%) exists. It can be concluded that the development of mature projects is often continued for commercial reasons. An increased release frequency, compared to the previous findings can be observed, as 39.2% releases once a quarter, while 32.1% release once half a year. The projects have increased testing activities in relation to the development time (40.7%) and three-fourth of the sample follows a structured testing approach. Users test averagely 60.2% of the code. Moreover, 77.3% reported that user found hard bugs. Code reviews (84.5%), inspections (59%), walkthroughs (71.4%) and even peer review techniques (69.5%) are more frequently applied. An increased number of projects (55.9%) perform corrective actions before code commit and rework code or reject inappropriate code. It is remarkable that 42.8% of the projects introduce defect handling from project start, while 32.1% started this in the development phase. These projects track more frequently code defects (95.2%), documentation (63.1%), requirement (52.4%) and design issues (50.0%). Their defect reporting quality is noticeable high as three-fourth reported to have useful bug descriptions as well as rules for defect classification and status tracking. In this sample, around half of the projects have a knowledge transfer procedure. Communication between user and developer assess 84.5% of the respondents as direct and efficient. Documentation has a high significance, for instance, 73.8% have process descriptions, 85.7% development documentation and 94.0% mention to have detailed product documentation or at least some drafts. The tool usage is conspicuously high, as more than 90% use source code control tools, bug-tracking tools or mailing lists and roughly 60% apply instant messaging and test support tools. Projects have a slightly higher interest in the execution of QA practices (33.3%). However, an increased number of 42.8% report that through QA activities the adherence to standards and requirements is checked. Also 29.7% reply that QA triggers actions, resulting from interviews or inspections.

6 Conclusion

This research analyses quality assurance practices by surveying open source projects and puts a special focus on key practices in mature projects. The survey results provide evidence for the OSSD lifecycle described by Raymond (1999). Projects follow frequent releases, have a high user involvement and benefit largely from user testing and peer reviews. There is a significant growth of communities when projects mature and the existence of large user communities and developer groups could be observed. Developers mainly contribute due to personal or community needs, however a higher commercial motivation exists in large projects. In general, a superior level of development knowledge can be observed. This leads to the assumption, that professionals know how to approach thing ‘right’, which could explain the large amount of successful projects collaborating in a loose manner sometimes with informal processes and fewer rules.

An investigation into more mature projects gains important insight into applied practices and may indicate reasons for the success of the OSSD model. The study results, that modularity of code is considered predominantly already during design, which shows the effort to base the development on a solid architecture. Quality control activities before code commit have a higher importance. More time is spent into testing and testing approach is better structured. These projects leverage efficiently their community, benefiting from an efficient user testing. Internal communication is rated as remarkable, which contributes positively to these processes. Defect handling processes seem better structured and comprise beside source code defects, also requirements and documentation issues. Documentation has a higher

significance. These projects emphase on widespread documentation and avail these information to support the knowledge transfer to developers and users. Tool usage is high, making additional use of instant messaging, bug tracking tools and test support tools. At large, the establishment of QA processes seems to play a more important role in larger projects but is underdeveloped in smaller projects.

The research findings contribute to an understanding of quality assurance practices and provide empirical evidence about applied processes under the OSSD model. The relations to project success criteria indicate some correlations between quality and succession but no causalities. Further research is required to explore QA practices and to determine their relation to the project success.

References

BROOKS, F.P. (1995) *The mythical man-month: essays on software engineering*. Reading, Mass: Addison-Wesley Pub. Co.

CROWSTON, K., HOWISON, J., ANNAB, H. (2006) Information systems success in free and open source software development: theory and measures. *Software Process: Improvement and Practice - Special Issue on Free/Open Source Software Processes,* **11(2)**, pp. 123–148.

FELLER, J. and FITZGERALD, B. (2000) A framework analysis of the open source software development paradigm. *in Proceedings of the twenty first international conference on Information systems ICIS 2000, Brisbane, Queensland, Australia*, pp.58–69.

HALLORAN, T.J. and SCHERLIS, W.L. (2002) High Quality and Open Source Software Practices. *in* ICSE 2002, *Proceeding of the 2nd Workshop on Open Source Software Engineering, Orlando, FL, USA*.

JENSEN, C. and SCACCHI, W. (2005) Modeling Recruitment and Role Migration Processes in OSSD Projects. Institute for Software Research, University of California, [cited 19th April 2007].
http://www.ics.uci.edu/~wscacchi/Papers/New/Jensen-Scacchi-ProSim05.pdf

KORU, G. and TIAN, J. (2004) Defect Handling in Medium and Large Open Source Projects. *IEEE Software,* **4**, pp. 54–61.

MASSEY, B. (2003) Why OSS folks think SE folks are clue-impaired. *in* ICSE 2003, *Proceedings of the 3rd Workshop on Open Source Software Engineering*, pp. 91–97.

MICHLMAYR, M. (2005) Software Process Maturity and the Success of Free Software Projects. *in* ZIELIŃSKI, K. and SZMUC, T. (eds.) *Software engineering: evolution and emerging technologies*. Amsterdam: IOS Press, pp. 3–14.

PORTER, A, YILMAZ, C., MENON, A.M., KRISHNA, A.S., SCHMIDT, D.C. and GOKHALE, A. (2006) Techniques and processes for improving the quality and performance of open-source software. *Software Process: Improvement and Practice - Special Issue on Free/Open Source Software Processes*, **11(2)**, pp. 163–176.

RAYMOND, E.S. (1999) Linux and open-source success. *IEEE Software*, **16(1)**, pp. 85–89.

RAYMOND, E.S. (2001) *The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary.* revised ed., O'Reilly.

SCHULMEYER, G., MCMANUS, J. (1999) Handbook of Software Quality Assurance. Upper Saddle River, NJ: Prentice Hall, p.9.

VILLA, L. (2003) Large free software projects and Bugzilla. *in Proceedings of the Linux Symposium, 23-26 July 2003, Ottawa, Canada.*

ZHANG, X. and PHAM, H., (2000) An analysis of factors affecting software reliability. *Journal of Systems and Software,* **50(1)**, pp. 43–56.

ZHAO, L., ELBAUM, S. (2000) A survey on software quality related activities in open source. *ACM SIGSOFT Software Engineering Notes*, **25 (3)**, pp. 54–57.

ZHAO, L., ELBAUM, S. (2003) Quality assurance under the open source development model. *The Journal of Systems and Software* **(66)**, pp. 65-75.

Clusteranalyse als Methode zur Strukturierung großer Datenmodelle

Dieter Riebesehl

Abstract

Datenmodelle sind ein zentraler Bestandteil betrieblicher Informationssysteme. In der Praxis eingesetzte Datenmodelle erreichen oft eine erhebliche Komplexität. Die sinnvolle Anordung der Entitäten und Relationen im Datenmodell ist nicht einfach. Durch Verwendung eines Maßes, das die Ähnlichkeit von Relationen innerhalb eines Datenmodells misst, können ähnliche Relationen nahe beieinander angeordnet werden. Die Zerlegung eines großen Datenmodells in Teilmodelle ist dann mit Methoden der Clusteranalyse möglich. Die vorgestellten Methoden werden auf die Datenmodelle aus [SC97] angewendet.

1 Einleitung

Ausgangspunkt für die Feststellung von Ähnlichkeiten oder besser Strukturanalogien zwischen den Relationen eines Datenmodells ist die Definition einer Maßzahl für die Ähnlichkeit aus [FE05]. Als Wert für die Ähnlichkeit zweier Relationen wird die normierte symmetrische Differenz der Entitätenmengen, auf denen die Relationen definiert sind, genommen, Details folgen weiter unten. Die Interpretation dieses Ähnlichkeitsmaßes ist naheliegend, wenn sich als Ähnlichkeitswert 1 ergibt, welches bedeutet, dass die beiden verglichenen Relationen strukturidentisch sind. Strukturidentische Relationen können formal zu einer Relation vereinigt werden. Das Datenmodell gewinnt schon dadurch erheblich an Übersichtlichkeit. Ist der Ähnlichkeitswert kleiner als 1, so kann er genutzt werden, um das Datenmodell durch nahe Anordnung ähnlicher Relationen übersichtlicher zu gestalten.

2 Ähnlichkeitswerte

Es soll zunächst einmal die Definition des Ähnlichkeitswertes aus [FE05] wiederholt werden. Dazu sei ein Datenmodell mit Entitäten und Relationen gegeben, $\mathcal{R}$ sei die Menge der Relationen, und $\mathcal{E}$ die Menge der Entitäten. Die Behandlung von als Entitäten reinterpretierten Relationen sei zunächst zurückgestellt. Dann gibt es zu einer Relation $R \in \mathcal{R}$

die Menge $\hat{E}(R) \subseteq \mathcal{E}$ der Entitäten, über denen R definiert ist. $\hat{E}(R)$ ist im Allgemeinen eine Multimenge, in der Elemente mit Vielfachheit (mehrfach) vorkommen können.

Bezeichnet man für eine Multimenge $\hat{M}$ und ein Element $a \in \hat{M}$ mit $r_{\hat{M}}(a)$ die Vielfachheit von a in $\hat{M}$, dann kann man Durchschnitt und Vereinigung von Multimengen wie folgt definieren:

$$\begin{aligned} r_{\hat{M}\cap\hat{N}}(a) &:= \min(r_{\hat{M}}(a), r_{\hat{N}}(a)), \\ r_{\hat{M}\cup\hat{N}}(a) &:= \max(r_{\hat{M}}(a), r_{\hat{N}}(a)). \end{aligned}$$

Definiert man die Mächtigkeit für eine Multimenge durch

$$|\hat{M}| := \sum_{a\in\hat{M}} r_{\hat{M}}(a),$$

dann ist der Ähnlichkeitswert zweier Relationen einfach gegeben durch [1]

$$a(R_1, R_2) = \frac{|\hat{E}(R_1) \cap \hat{E}(R_2)|}{|\hat{E}(R_1) \cup \hat{E}(R_2)|}.$$

Bei der Definition des Ähnlichkeitswertes ist natürlich die Definition von $\hat{E}(R)$ entscheidend, und diese wiederum hängt vom Typ des verwendeten Datenmodells ab. Drei Modelle sind recht weit verbreitet:

ERM: Das Standard-ERM nach Chen ([CH76]), welches nur Entitäten, dargestellt durch Rechtecke, und Relationen, dargestellt durch Rauten, unterscheidet und insbesondere keine als Entitäten reinterpretierten Relationen kennt.

PERM: Das erweiterte ERM nach Loos ([LO97]), welches reinterpretierte Relationen kennt. Es enthält noch weitere Modellierungsmöglichkeiten, auf die hier aber nicht eingegangen wird.

SERM: Das strukturierte ERM nach Sinz ([SI93]), welches zusätzlich eine hierarchische Anordnung der Relationen vorsieht.

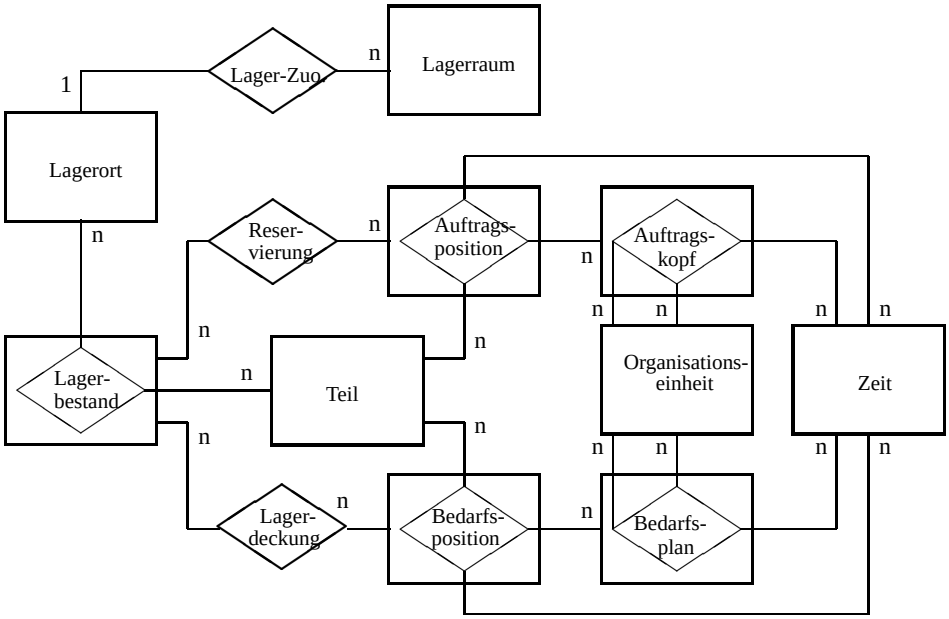
Die Verwendung der Abkürzung ERM für ein Modell ohne reinterpretierte Relationen entspricht nicht unbedingt dem Standard, wird aber von manchen Autoren von in die Modellierung einführender Literatur und auch in mehreren Modellierungswerkzeugen so genutzt, siehe z.B. Jarosch in [JA02].

Die als Beispiele benutzten Modelle liegen alle als PERM vor, weshalb nur noch dieser Modelltyp betrachtet werden soll.

Der Ähnlichkeitskoeffizient wird zunächst am Beispiel des folgenden Modellausschnitts der Vertriebsabwicklung aus [SC97] diskutiert:[2]

[1] Abweichend von [FE05] wird der Ähnlichkeitswert mit a statt d bezeichnet, da noch eine auf a basierende Metrik d eingeführt werden wird.

[2] Das vollständige Modell der Vertriebsabwicklung ist im Anhang, Seite 123, abgebildet.



2.1 Definition von $\hat{E}(R)$

Die Mengen $\mathcal{E}$ und $\mathcal{R}$ haben hier einen nichtleeren Durchschnitt bestehend aus den Relationen, die als Entitäten reinterpretiert werden.

Die Bestimmung der Mengen $\hat{E}(R)$ geschieht rekursiv:

- Für $E \in \mathcal{E} \setminus \mathcal{R}$ ist

  $$\hat{E}(E) := \{E\}.$$

- Für $R = E_1 \times E_2 \times \cdots \times E_n \in \mathcal{R}$ ist

  $$\hat{E}(R) := \bigoplus_{1}^{n} \hat{E}(E_i).$$

Dabei ist die Summe zweier Multimengen definiert durch

$$r_{\hat{M} \oplus \hat{N}}(a) := r_{\hat{M}}(a) + r_{\hat{N}}(a).$$

Wegen des Fehlens rekursiver Bezüge muss die Rekursion enden, siehe dazu auch den nächsten Abschnitt.

Aus den damit berechneten Ähnlichkeitskoeffizienten ergeben sich für den Modellausschnitt folgende Gruppen strukturidentischer Relationen, also solcher mit Ähnlichkeitswert 1:

```
Auftragskopf = Bedarfsplan
Auftragsposition = Bedarfsposition
Lagerdeckung = Reservierung
```

Die Modellierung im PERM gestattet es, durch geschickte Anordnung der Objekte im Diagramm diese Analogien direkt über Symmetrien zu erkennen - so wie im obigen Bild.

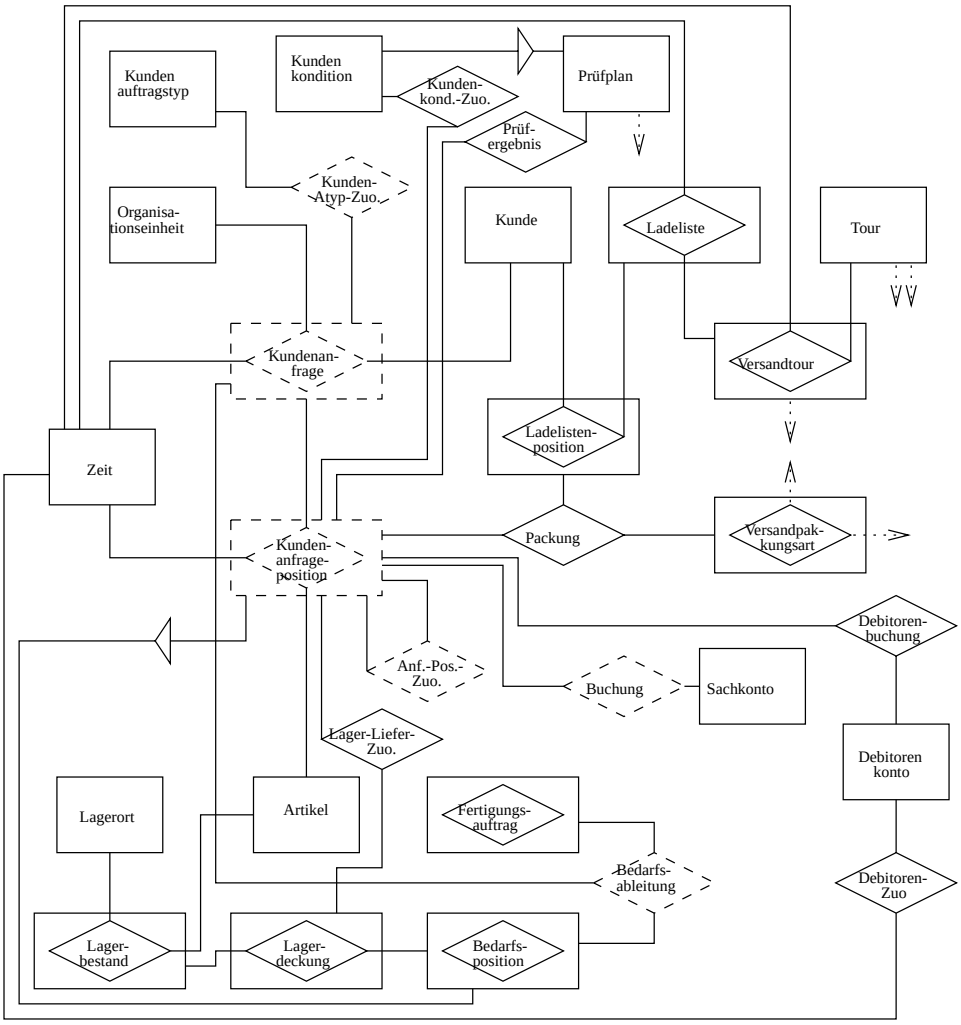
2.2 Anwendung auf die Vertriebsabwicklung

Aus der bereits genannten Quelle [SC97] soll nun die gesamte Vertriebsabwicklung (siehe daselbst S. 458f, Abb. B.II.25), sowie die Abbildung im Anhang (Seite 123), betrachtet werden.

Der Ähnlichkeitswert ist 1 für folgende Relationen:

```
    Kundenanfrage = Kundenangebot = Kundenauftrag
= AbgesKundenauftrag = Lieferschein = Kundenrechnung,
    Kundenanfrageposition = Kundenangebotsposition
  = Kundenauftragsposition = AbgesKundenauftragsposition
  = Lieferscheinposition = Kundenrechnungsposition,
    Kundenauftrag_Angebot_Zuo = Folgeauftrag
= Teillieferung = Kundenanfrage_Angebot_Zuo
= Lieferschein_Rechnung_Zuo,
    Kundenauftragstyp_Zuo = KundenAbgesauftragstyp_Zuo,
    Bedarfsableitung = Bedarfsdeckung,
    Buchung = Sachbuchung,
```

Werden die strukturidentischen Relationen identifiziert, so reduziert sich das PER-Modell ganz erheblich:



Gestrichelte Relationen stehen dabei für Gruppen von strukturidentischen Relationen. Als Name der Gruppe wurde eine darin enthaltene Relation zufällig ausgewählt. Gestrichelte Pfeile weisen auf nicht abgebildete Beziehungen zu weiteren Entitäten und Relationen hin, die weggelassen wurden, da sie nicht von Strukturidentäten betroffen sind.

Der Gewinn an Übersichtlichkeit, der allein durch die graphische Vereinigung strukturidentischer Relationen erreicht wird, ist offensichtlich.

3 Clusteranalyse

Die Clusteranalyse dient zur Zusammenfassung einer Menge von Objekten in möglichst homogene oder eng zusammenliegende Gruppen. Bei den Objekten handelt es hier natürlich

um die Relationen (und evtl. auch Entitäten) eines Datenmodells. Unter Homogenität wird eine weitgehend ähnliche Struktur der Clustermitglieder verstanden. Es ist klar, dass die Strukturähnlichkeit von Relationen mittels des Ähnlichkeitswertes zu messen sein wird. Die Nähe von Objekten zueinander wird durch ein Abstandsmaß, also eine Metrik beschrieben.

In diesem Abschnitt sollen Methoden der Clusteranalyse eingesetzt werden, um die Relationen eines Datenmodells zu Gruppen zusammenzufassen. Ziel dabei ist es, graphische ER-Modelle besser zu strukturieren, übersichtlicher zu gestalten und vereinfachte Sichten zu erstellen.

In [RA92] wird durch O. Rauh und E. Stickel eine andere Methode zur Clusterung von ER-Modellen vorgestellt, die ebenfalls das Ziel hat, ER-Modelle klarer und übersichtlicher zu machen. Die beiden Autoren bilden Cluster von *Entitäten*, während in dieser Arbeit bei der Clusterung von den *Relationen* ausgegangen wird.

Vorstellung der Verfahren

Für die Clusterbildung müssen vorab zwei Auswahlen getroffen werden:

1. Wahl eines Proximitätsmaßes
2. Wahl eines Algorithmus für die Clusterbildung

Ein Proximitätsmaß misst die Ähnlichkeit zweier Objekte bzw. deren Nähe zueinander. Da im Laufe des Verfahrens Objekte zu Clustern zusammengefasst werden, muss das Proximitätsmaß auch für zwei Cluster bestimmt werden. Aus dem Ähnlichkeitswert für Objekte muss also ein Ähnlichkeitswert für Mengen von Objekten abgeleitet werden. Gleiches gilt für den Einsatz einer Metrik: aus dem Abstand zwischen Objekten muss ein Abstand zwischen Mengen von Objekten abgeleitet werden.

Für die Clusterbildung gibt es zwei grundsätzlich verschiedene Vorgehensweisen:

1. Austauschverfahren
2. Agglomeratives, hierarchisches Verfahren

Beim Austauschverfahren geht man von der gewünschten Clusterzahl aus und zerlegt die Objektmenge auf eine geeignete heuristische Weise in so viele vorläufige Cluster, wie man am Ende haben möchte. Dann wechselt man in mehreren iterativen Schritten jeweils die Clusterzugehörigkeit eines Objektes mit dem Ziel, ein Gütemaß für die Clusterung zu maximieren. Das Verfahren endet, wenn eine gewünschte Güte erreicht ist.

Da die Clusteranzahl aber erst bestimmt werden soll, scheidet das Austauschverfahren aus. Beim agglomerativen Verfahren geht man stattdessen von der maximalen Clusterzahl aus, in dem jedes Objekt R zu einem ein-elementigen Cluster $\{R\}$ erklärt wird. In mehreren iterativen Schritten werden dann jeweils die zwei Cluster mit der größten Ähnlichkeit bzw. geringsten Distanz vereinigt. Dieser Schritt wird so lange wiederholt, bis alle Objekte in einem Cluster vereinigt sind.

Das Ergebnis der Clusterung hat die Struktur eines Binärbaumes, genannt *Dendogramm*. Seine Blätter sind die einzelnen Objekte. Jeder Knoten steht für die Zusammenfassung zweier Cluster, die Wurzel ist der Cluster, der alle Objekte enthält.

Bei den hier betrachteten Relationen wird die Proximität über nominale Merkmale mit binärer Merkmalstruktur definiert: Relation R ist oder ist nicht über der Entität E definiert. Für solche Nominalskalen werden in [BA86] verschiedene Proximitätsmaße definiert. Das in dieser Arbeit bisher benutzte Ähnlichkeitsmaß a stellt sich als identisch mit dem Tanimoto-Koeffizienten heraus.

Die Fortsetzung des Proximitätsmaßes auf Cluster kann nach verschiedenen Kriterien erfolgen. Eine Auswahl solcher Fortsetzungen findet sich in [BA86] (Tabelle 6.17, S. 287), wo auch ihre Eigenschaften diskutiert werden. Dabei ist zu beachten, dass von den sieben dort genannten Fortsetzungen drei zwingend eine Metrik zwischen den Objekten voraussetzen.

3.1 Metriken

Wenn man strukturäquivalente Relationen identifiziert, dann kann man den Ähnlichkeitswert nutzen, um eine Metrik d auf der Menge der Äquivalenzklassen von Relationen zu definieren:

$$d(R_1, R_2) := 1 - a(R_1, R_2)$$

ist nämlich automatisch symmetrisch und erfüllt

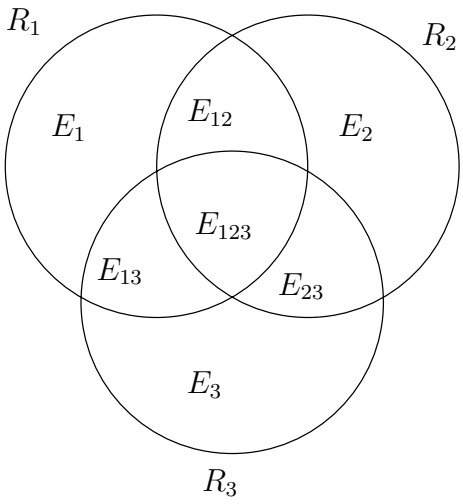
$$d(R_1, R_2) = 0 \iff R_1 \cong R_2.$$

($\cong$ steht für „strukturidentisch“)

Die Dreiecksungleichung

$$d(R_1, R_2) \leq d(R_1, R_3) + d(R_3, R_2)$$

ergibt sich aus dem folgenden Venn-Diagramm für die Entitätenmengen zu den Relationen:



In diesem Venn-Diagramm sind die Multimengen von Entitäten, auf denen die Relationen R_1, R_2 und R_3 definiert sind, in disjunkte Teile zerlegt:

$$\begin{aligned} E_{123} &:= \hat{E}(R_1) \cap \hat{E}(R_2) \cap \hat{E}(R_3), \\ E_{12} &:= \hat{E}(R_1) \cap \hat{E}(R_2) \setminus E_{123}, \\ &\dots \\ E_1 &:= \hat{E}(R_1) \setminus (\hat{E}_2 \cup \hat{E}_3), \\ &\dots \end{aligned}$$

Wenn man die Mächtigkeiten dieser Teilmengen mit $e_{\dots} := |E_{\dots}|$ bezeichnet - wobei die Punkte für einander entsprechende gleiche Indizes stehen, dann ist

$$a_{ij} := a(R_i, R_j) = \frac{e_{123} + e_{ij}}{G - e_k},$$

hier ist $G = |\hat{E}(R_1) \cup \hat{E}(R_2) \cup \hat{E}(R_3)|$, und $\{i, j, k\} = \{1, 2, 3\}$.

Daraus ergibt sich

$$d_{ij} := d(R_i, R_j) = \frac{e_{ik} + e_{jk} + e_i + e_j}{e_{ij} + e_{ik} + e_{jk} + e_i + e_j + e_{123}}.$$

Die Dreiecksungleichung ist erfüllt, wenn stets

$$d_{ij} + d_{jk} - d_{ik} \geq 0$$

ist.

Bringt man die linke Seite auf einen Nenner, so errechnet sich der Zähler zu

$$
\begin{aligned}
&e_{123}\,e_i\,e_{ij} + {e_i}^2\,e_{ij} + e_i\,{e_{ij}}^2 + 2\,{e_{123}}^2\,e_{ik} + 3\,e_{123}\,e_i\,e_{ik} + {e_i}^2\,e_{ik} + \\
&4\,e_{123}\,e_{ij}\,e_{ik} + 4\,e_i\,e_{ij}\,e_{ik} + 2\,{e_{ij}}^2\,e_{ik} + 4\,e_{123}\,{e_{ik}}^2 + 3\,e_i\,{e_{ik}}^2 + \\
&4\,e_{ij}\,{e_{ik}}^2 + 2\,{e_{ik}}^3 + 2\,{e_{123}}^2\,e_j + 2\,e_{123}\,e_i\,e_j + {e_i}^2\,e_j + 3\,e_{123}\,e_{ij}\,e_j + \\
&2\,e_i\,e_{ij}\,e_j + {e_{ij}}^2\,e_j + 6\,e_{123}\,e_{ik}\,e_j + 4\,e_i\,e_{ik}\,e_j + 5\,e_{ij}\,e_{ik}\,e_j + 4\,{e_{ik}}^2\,e_j + \\
&2\,e_{123}\,{e_j}^2 + e_i\,{e_j}^2 + e_{ij}\,{e_j}^2 + 2\,e_{ik}\,{e_j}^2 + e_i\,e_{ij}\,e_{jk} + 4\,e_{123}\,e_{ik}\,e_{jk} + \\
&3\,e_i\,e_{ik}\,e_{jk} + 4\,e_{ij}\,e_{ik}\,e_{jk} + 4\,{e_{ik}}^2\,e_{jk} + 3\,e_{123}\,e_j\,e_{jk} + 2\,e_i\,e_j\,e_{jk} + \\
&2\,e_{ij}\,e_j\,e_{jk} + 5\,e_{ik}\,e_j\,e_{jk} + {e_j}^2\,e_{jk} + 2\,e_{ik}\,{e_{jk}}^2 + e_j\,{e_{jk}}^2 + 2\,e_{123}\,e_i\,e_k + \\
&{e_i}^2\,e_k + 2\,e_i\,e_{ij}\,e_k + 3\,e_{123}\,e_{ik}\,e_k + 4\,e_i\,e_{ik}\,e_k + 3\,e_{ij}\,e_{ik}\,e_k + \\
&3\,{e_{ik}}^2\,e_k + 2\,e_{123}\,e_j\,e_k + 2\,e_i\,e_j\,e_k + 2\,e_{ij}\,e_j\,e_k + 4\,e_{ik}\,e_j\,e_k + {e_j}^2\,e_k + \\
&e_{123}\,e_{jk}\,e_k + 2\,e_i\,e_{jk}\,e_k + e_{ij}\,e_{jk}\,e_k + 4\,e_{ik}\,e_{jk}\,e_k + 2\,e_j\,e_{jk}\,e_k + {e_{jk}}^2\,e_k + \\
&e_i\,{e_k}^2 + e_{ik}\,{e_k}^2 + e_j\,{e_k}^2 + e_{jk}\,{e_k}^2,
\end{aligned}
$$

und das ist offensichtlich nichtnegativ! Außer in Trivialfällen ist die Dreiecksungleichung also sogar eine echte Ungleichung.

Der Sonderfall $e_2 = e_3 = e_{13} = e_{23} = 0$, $e_{123} = 1$ ergibt

$$d_{12} + d_{23} - d_{13} = \frac{e_1\,e_{12}}{(1+e_{12})\,(1+e_1+e_{12})} < \frac{e_1}{e_1+e_{12}},$$

welches beliebig klein werden kann. Ebenso kann

$$\frac{d_{13}}{d_{12}+d_{23}} = 1 - \frac{e_1\,e_{12}}{e_1+e_{12}+2\,e_1\,e_{12}+{e_{12}}^2} > 1 - \frac{e_1}{e_1+2e_{12}}$$

beliebig nahe an 1 liegen. Die Ungleichung läßt sich also weder absolut noch relativ verschärfen.

Für die Ähnlichkeitswerte ergibt sich aus der Dreiecksungleichung die Beziehung

$$a_{12} \geq a_{13} + a_{23} - 1.$$

Dies ist eine intuitiv einleuchtende Eigenschaft von Ähnlichkeitswerten, die, wie es hier der Fall ist, einen relativen Grad von Ähnlichkeit messen. In Worten ausgedrückt heißt es nämlich, dass zwei Relationen, die mit einer dritten Relation beide gemeinsame Entitäten haben, und deren relative Anteile in der Summe 100% überschreiten, untereinander auch mindestens eine Entität gemein haben müssen.

3.2 Ein anderes Ähnlichkeitsmaß

Schon in [FE05] wird darauf hingewiesen, dass die Ähnlichkeitswerte auch anders berechnet werden könnten. Interessanterweise führt auch das in [BA86], S. 267 beschriebene *Simple-Matching*-Maß (M-Maß) auf eine Metrik.

Angewendet auf unsere Situation berechnet sich der Ähnlichkeitswert mit dem M-Maß a_M durch

$$a_M(R_1, R_2) = \frac{m - |\hat{E}(R_1) \triangle \hat{E}(R_2)|}{m},$$

dabei ist $\triangle$ die symmetrische Differenz und

$$m = \left| \bigcup_R \hat{E}(R) \right|$$

die Gesamtzahl aller beteiligten Entitäten mit Vielfachheit.

Setzt man wieder für die Metrik $d_M = 1 - a_M$ und berücksichtigt, dass m eine feste Konstante ist, kann auch einfach

$$d_M(R_1, R_2) = |\hat{E}(R_1) \triangle \hat{E}(R_2)|$$

als Metrik verwendet werden, was im Folgenden geschehen soll (M-Metrik).

d_M hat die gewünschten Eigenschaften:

1. $d_M = 0$ genau für strukturidentische Relationen,
2. d_M ist symmetrisch,
3. d_M erfüllt die Dreiecksungleichung, wie man sich sofort am Venn-Diagramm für drei Relationen klarmacht:

$$\begin{aligned} d_M(R_1, R_2) &= e_1 + e_{13} + e_2 + e_{23} \\ &\leq e_1 + e_{13} + e_2 + e_{23} + 2e_{12} + 2e_3 \\ &= d_M(R_1, R_3) + d_M(R_2, R_3). \end{aligned}$$

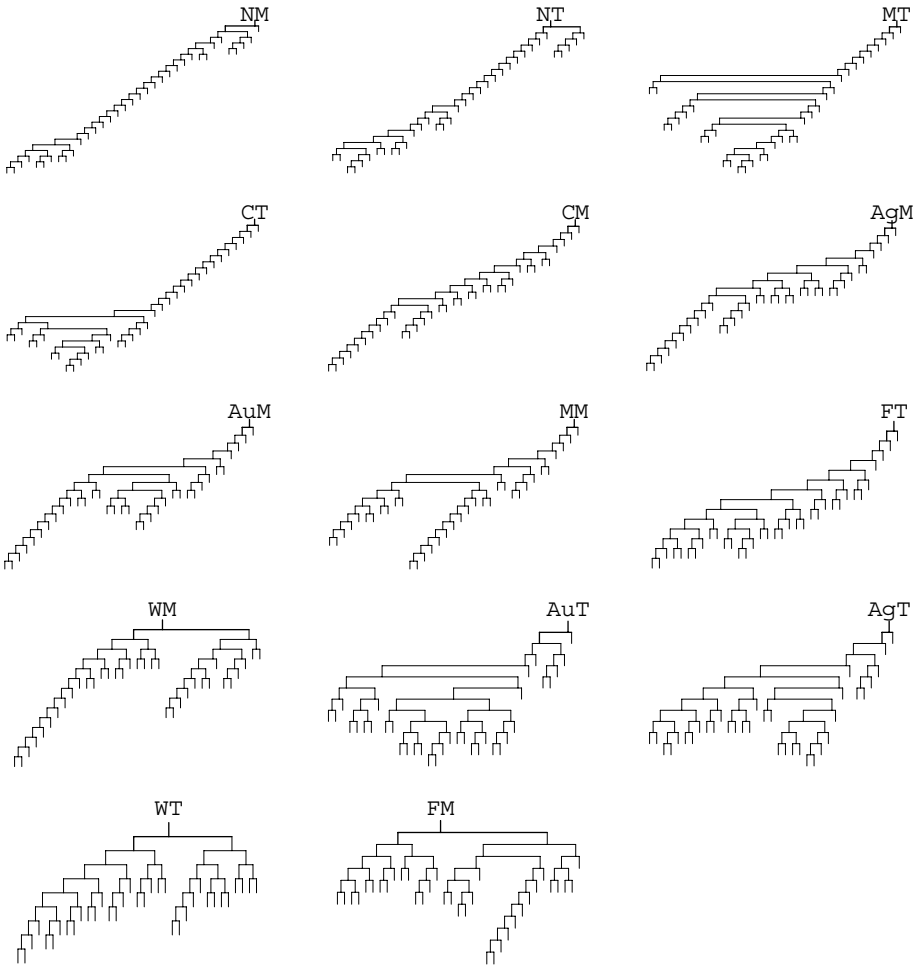
3.3 Anwendung auf Datenmodelle

Im Folgenden sollen ausführlich die Cluster von Relationen diskutiert werden, die sich bei verschiedenen Verfahren der Clusteranalyse ergeben. Es gibt 14 verschiedene Kombinationen, die mit Abkürzungen bezeichnet werden, siehe die folgende Tabelle.

Metrik / Proximitätsmaß	Tanimoto	M-Metrik
Nearest Neighbour (Single Linkage)	NT	NM
Furthest Neighbour (Complete Linkage)	FT	FM
Average Linkage ungewichtet	AuT	AuM
Average Linkage gewichtet	AgT	AgM
Centroid	CT	CM
Median	MT	MM
Ward	WT	WM

Dabei kommen als Metriken die nach Tanimoto und die M-Metrik zum Einsatz, als Proximitätsmaße für Gruppen alle sieben in [BA86] genannten.

In der folgenden Grafik sind die Dendogramme gezeigt, die sich für das Teilmodell der Bedarfs-, Zeit- und Kapazitätsplanung aus [SC97][3]ergeben.



Je nach Metrik und Proximitätsmaß erkennt man eine mehr oder weniger starke Neigung zur Bildung langer Ketten. Die Bilder sind absteigend sortiert nach der Neigung zur Bildung von Ketten, von links nach rechts und von oben nach unten.

Aus jedem bei der Clusterung entstehenden Dendogramm lässt sich durch Augenschein eine kleine Anzahl von Hauptclustern (im folgenden kurz Cluster genannt) auswählen.

[3]S. 176, Abb. B.I.66, S. 254f, Abb. B.I.133, auch dargestellt im Anhang, Seite 122,

Die Anzahl und genaue Zusammensetzung der Cluster unterliegt dabei einer gewissen Willkür.

Es zeigt sich, dass die ausgewählten Cluster über alle Methoden und bei der „richtigen“ Metrik sehr stabil sind, bei der „falschen“ Metrik aber so gut wie gar nicht erkennbar. Daraus kann man vorsichtig schließen, dass das Proximitätsmaß einen vergleichsweise geringen Einfluss auf die Clusterung hat, diese aber stark von der Metrik abhängen kann.

In den folgenden Abschnitten sollen nur noch drei Proximitätsmaße als repräsentative Auswahl weiterverfolgt werden, nämlich **Nearest Neighbour**, **Furthest Neighbour** und **Ward**. Damit ist je ein stark kettenbildendes, ein eher viele etwa gleich große Gruppen bildendes und ein zu zwei etwa gleich großen Gruppen neigendes Maß ausgewählt. Versuche mit mehreren Datenmodellen zeigten, dass bei allen diesen Maßen die grobe Clusterstruktur recht stabil ist.

Die folgenden Abschnitte untersuchen für die Vertriebsabwicklung und das Gesamtmodell aus [SC97] die Clusterbildung näher.

3.3.1 Vertriebsabwicklung

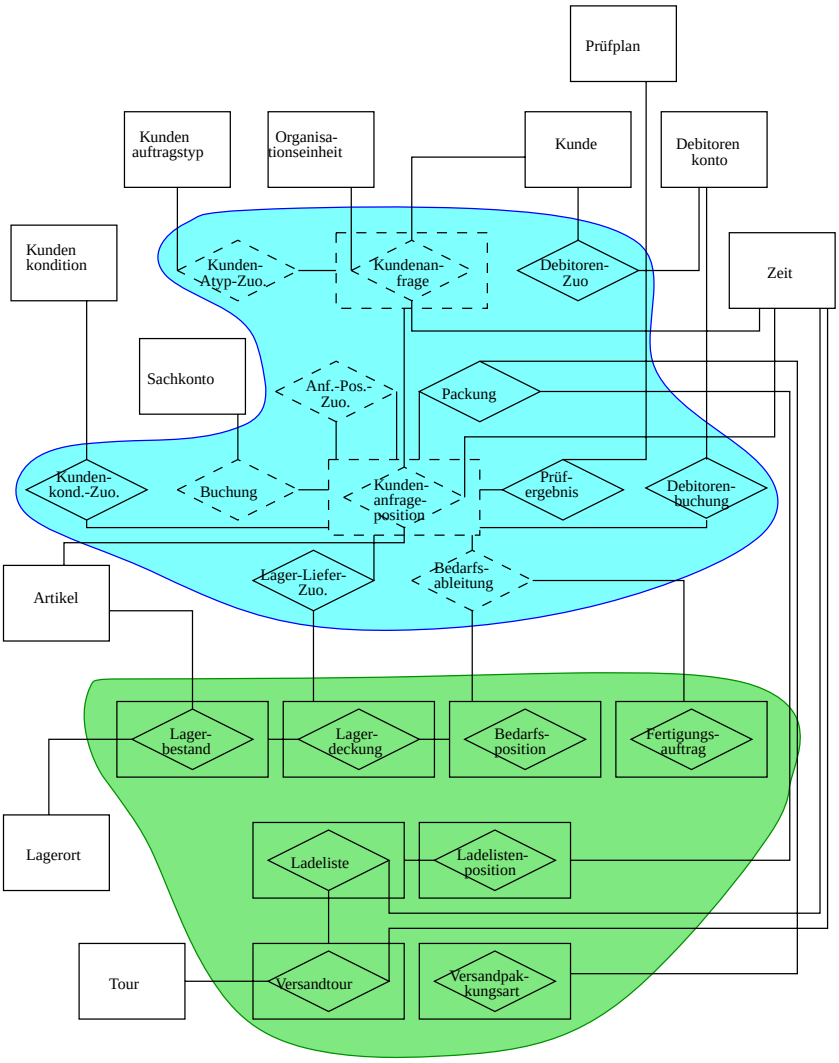
Die Methode **WT** soll diesmal im Detail gezeigt werden. Man erhält eine Zerlegung der Relationenmenge in zwei große farbige Cluster (Abbildung Seite 116). Der untere Cluster enthält dabei noch weitere hier nicht gezeigte, aber im Originalmodell vorhandene Relationen. Jede Zeile in den Clustern stellt ein Untercluster dar. Im Vergleich mit dem Originalmodell[4] wird der Gewinn an Übersichtlichkeit deutlich, die um die Cluster angeordneten Entitäten machen die Verwandtschaft der Relationen gut sichtbar, auch lassen sich klare Schnittstellen zwischen den beiden Clustern erkennen.

3.3.2 Gesamtmodell

Bei [SC97] werden neben der Vertriebsabwicklung und der Bedarfsplanung noch weitere Datenmodelle betrachtet, die ein Gesamtmodell ergeben. Dieses wird von Scheer in im wesentlichen vier Teilmodelle zerlegt:

1. für Bedarfs-, Zeit- und Kapazitätsplanung, ([SC97], S. 176, Abb. B.I.66, S. 254f, Abb. B.I.133)
2. für Beschaffungslogistik, ([SC97], S. 418f, Abb. B.II.07)
3. für CAM ([SC97], S. 366f, Abb. B.I.221) und
4. für Vertriebsabwicklung ([SC97], S. 458f, Abb. B.II.25).

[4] [SC97], S. 458f, Abb. B.II.25, oder Seite 123,



Wenn man die Clusteranalyse auf das Gesamtmodelle anwendet, erhofft man sich natürlich, die vier Teilmodelle als Cluster wiederzufinden. Falls nicht dies nicht geschieht, erhält man evtl. Aufschluss über andere Gliederungsmöglichkeiten des Gesamtmodells.

Die folgende Abbildung zeigt das komplette Dendogramm für alle 129 Relationen bei Verwendung der Tanimoto-Metrik und des Proximitätsmaßes von Ward. Es sind dabei der Übersicht halber nicht die Relationsnamen gezeigt. Stattdessen steht für jede Relation nur die Nummer des Teilmodells nach obiger Liste, in dem sie vorkommt[5].

Der Übersichtlichkeit halber ist die Darstellung zweidimensional, dabei wurde ein Dendogramm

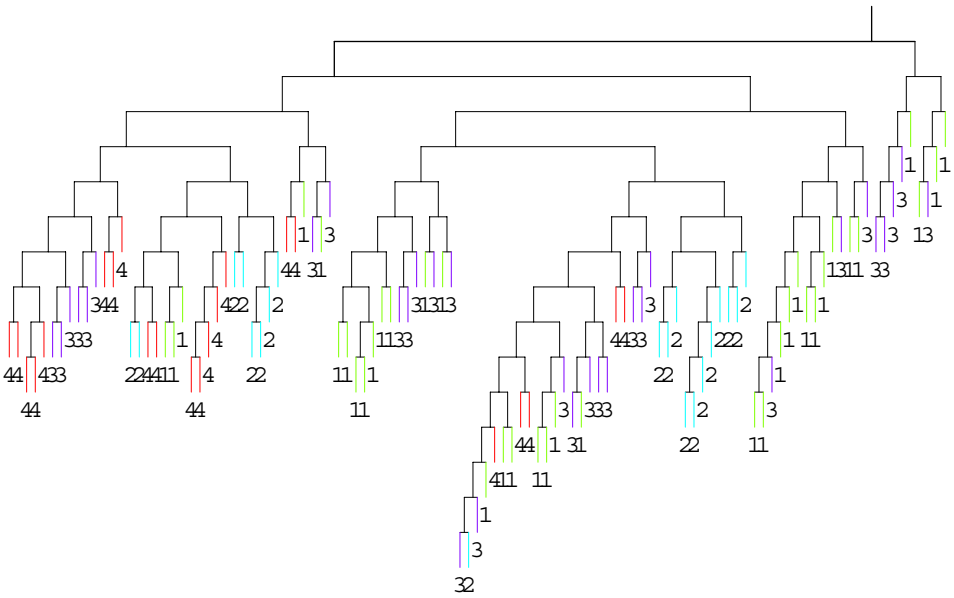
[5]Wegen Überschneidungen der Teilmodelle kommen einige Relationen in mehreren Teilmodellen vor, sie wurden willkürlich einem Datenmodell zugeordnet.

als Matrix $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$ dargestellt.

$$\left(\left(\left(\left(\left(\begin{pmatrix}\begin{pmatrix}(2,3,4,2)\\(3,3)\\4\end{pmatrix} & \begin{pmatrix}(1,1,1)\\3\end{pmatrix}\\ & \begin{pmatrix}\begin{pmatrix}(1,1,1)\\(1,1,1)\\1\end{pmatrix} & (1,1,1)\\ & 1\end{pmatrix}\end{pmatrix} & \begin{pmatrix}(3)\\4\end{pmatrix}\\ & \begin{pmatrix}(2,2)\\(2,2,2)\end{pmatrix}\end{pmatrix} & \begin{pmatrix}(3,1,3)\\(1,3,3)\end{pmatrix}\\ (3,3,3,3)\end{pmatrix} & \begin{pmatrix}(3,3,3) & (4,4,4)\\(4,4,4,4) & (4,4)\end{pmatrix}\\ & \begin{pmatrix}(2,2,2)\\(2,2,2)\end{pmatrix}\end{pmatrix} & \begin{pmatrix}\begin{pmatrix}(1,1,1,3)\\(3,3,3)\end{pmatrix} & (1,3,1)\\ \begin{pmatrix}(1,1,3,1)\\(1,3)\end{pmatrix} & 1\end{pmatrix}\\ & (1,1,3,1)\end{pmatrix}\\ (3,3,3,3,1)\end{pmatrix} & \begin{pmatrix}\begin{pmatrix}\begin{pmatrix}2&2\\4&4\end{pmatrix} & (1,1,1)\\(4,4,4,4) & 4\end{pmatrix} & (2,2,2,2,2)\\(1,4,4) & \begin{pmatrix}(1,4)\\(3,3,1)\end{pmatrix}\end{pmatrix}\end{pmatrix}\right)$$

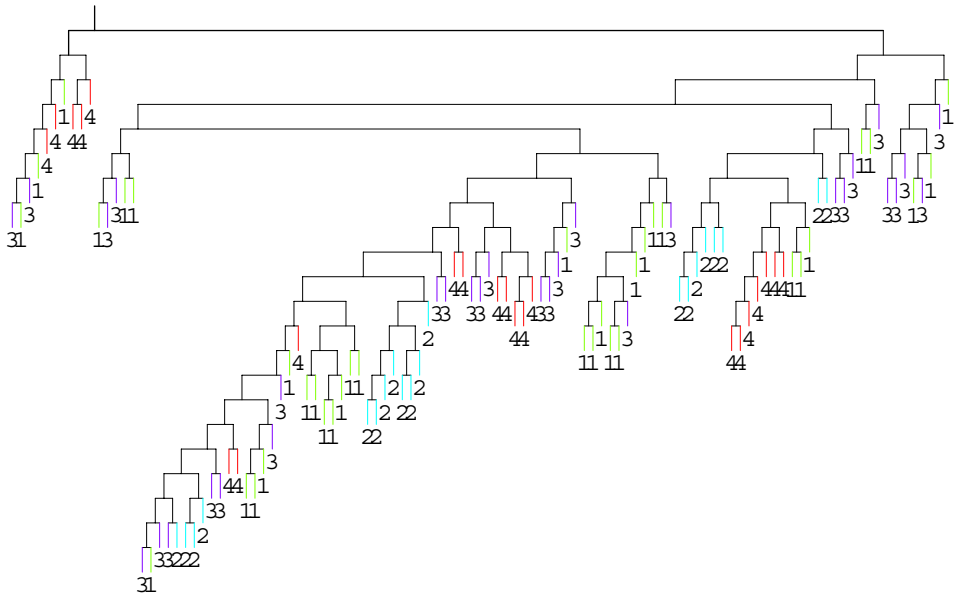
Zum Vergleich werden noch die Dendogramme zu den Methoden **WM** und **FM** gezeigt [6]:

WM:



[6]Die anderen Varianten eignen sich nicht für diese Darstellung, da sie sehr lange Ketten enthalten.

FM:



Man erkennt deutlich, dass sich die Relationen eines Modells auch in Clustern zusammenfügen, dass aber auch häufig Relationen verschiedener Teilmodelle zusammengefasst werden. Dies ist klar der Fall für die Modelle 1 und 3, also Bedarfsplanung etc. und CAM. Die Clusterung mit den anderen Proximitätsmaßen und Metriken zeigt ein ähnliches Verhalten, auch ohne Relationsnamen kann man per Augenschein fast identische Cluster in den verschiedenen Bildern wiedererkennen.

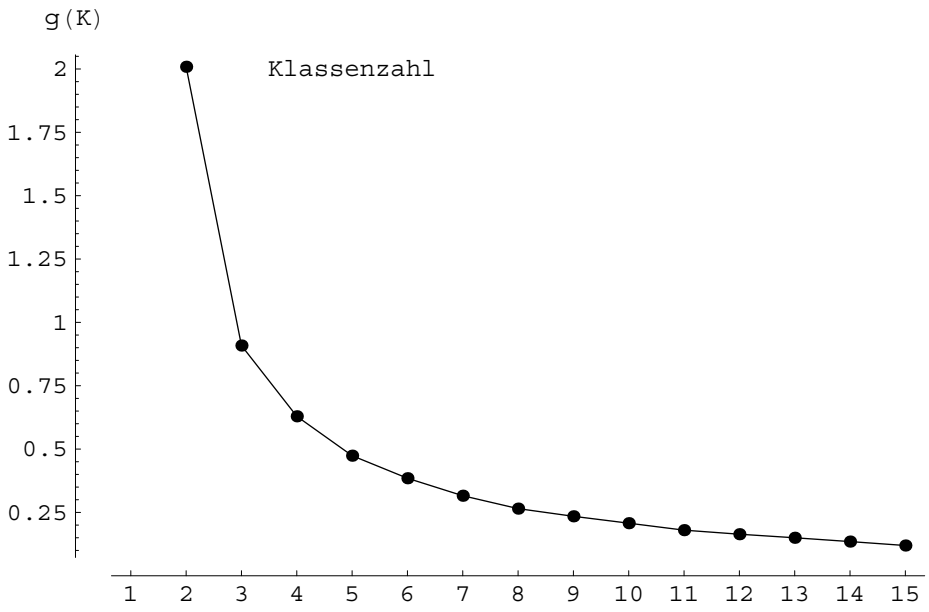
Betrachtet man die einzelnen Relationen, so findet man einige, die bei allen Clusterungsvarianten ein und demselben *anderen* Teilmodell zugeordnet werden. Die Clusteranalyse nach Ähnlichkeitsmetriken kann also konkrete Hinweise zur Umorganisation der Teilmodelle geben.

Es wird aber auch deutlich, dass nicht die gesamten Teilmodelle – auch nicht im Wesentlichen – als Cluster auftauchen, sondern stets Untermodelle, die dann wechselnde Beziehungen eingehen.

3.4 Bestimmung der optimalen Anzahl von Clustern

Das Verfahren der Clusteranalyse geht von einelementigen Clustern aus, die so lange miteinander vereinigt werden, bis nur noch ein Cluster übrig ist. Zu jedem Zeitpunkt liegt damit eine disjunkte Zerlegung der Objekte in Cluster vor. Die Cluster einer solchen Zerlegung heißen Klassen. Deren Anzahl liegt also zwischen der Anzahl der Objekte und 1.

Die bestgeeignete Gruppierung der Objekte in Klassen ist gesucht. Um diese zu finden, gibt es verschiedene Verfahren, die, ausgehend von einer Heterogenitätsfunktion für Klassen, einen Gütewert für eine Klassifikation bestimmen. Die Einteilung in Klassen versucht, eine gute Kombination von kleiner Klassenzahl und hoher Güte der Klassifikation zu finden, siehe z.B. [EV01]. Da hier ein großer Interpretationsspielraum besteht, soll nur ein Beispiel gegeben werden, nämlich die aus dem Dendogramm zur Ward-Proximität und zur Tanimoto-Metrik (WT) entstehende Graphik, die die Güte der Klassenbildung gegen die Anzahl Klassen aufträgt:



Hierbei wurden verwendet: die maximale Distanz $h(K)$ innerhalb einer Klasse als Heterogenitätsmaß und die gewichtete Summe der Heterogenitäten über alle Klassen $g(\mathcal{K})$ als Maßzahl der Güte der Klassenzerlegung, in Formeln:

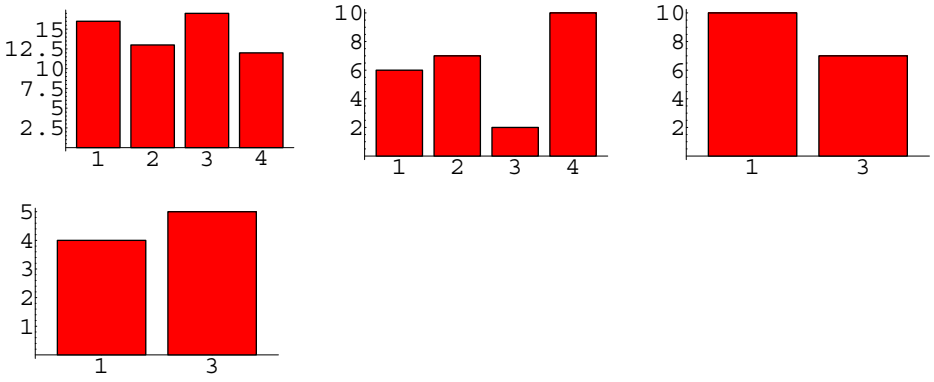
$$
\begin{aligned}
h(K) &:= \max_{R_1,R_2\in K} d(R_1,R_2), \\
g(\mathcal{K}) &:= \frac{1}{w(\mathcal{K})}\sum_{K\in\mathcal{K}} h(K), \text{ mit} \\
w(\mathcal{K}) &:= \sum_{K_1,K_2\in\mathcal{K}} v(K_1,K_2), \text{ wobei} \\
v(K_1,K_2) &:= \frac{1}{|K_1||K_2|}\sum_{\substack{R_1\in K_1\\ R_2\in K_2}} d(R_1,R_2) \text{ ist.}
\end{aligned}
$$

Einer hohen Güte entsprechen kleine Werte von $g(\mathcal{K})$. $g(\mathcal{K})$ nimmt mit zunehmender Klassenzahl ab und erreicht den Wert Null, wenn jede Relation eine eigene Klasse bildet. Man

sucht deshalb eine geeignete nicht zu große Klassenzahl mit befriedigender Güte aus.

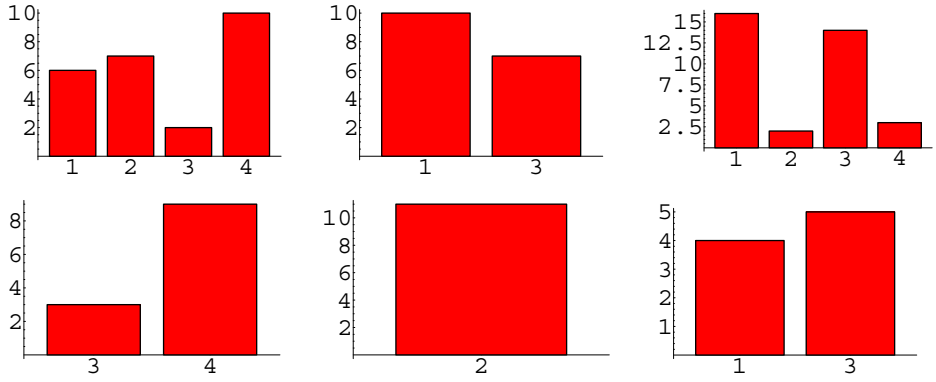
Die entstehende Grafik weist evtl. einen deutlichen Knick, den „Ellenbogen“ auf, hinter welchem eine Erhöhung der Klassenzahl nur noch eine geringe Zunahme der Güte bewirkt. Dieser Knick kann als Ort der optimalen Klasseneinteilung gesehen werden.

Freundlicherweise findet sich ein einigermaßen deutlicher Ellbogen bei der Klassenzahl vier, was gerade die Anzahl Teilmodelle ist, die Gegenstand der Clusteranalyse waren! Die folgenden Bilder zeigen, wie sich die Relationen der originalen vier Teilmodelle auf die 4 neuen Klassen verteilen:



Insofern hat die Clusteranalyse die richtige Anzahl wiedergefunden. Allerdings bleibt es bei dem Ergebnis, dass die Zuordnung der Relationen zu den Klassen nicht mit der Zerlegung in die vorgegebenen Teilmodelle identisch ist. Es zeigt sich also, dass diese Art der Clusteranalyse anhand der Ähnlichkeitswerte zu vernünftigen Ergebnissen führen und gleichzeitig Hinweise auf eine Umorganisation der ER-Modelle geben kann.

Man könnte sich allerdings auch dafür entscheiden, den Ellbogen bei der Klassenzahl 6 zu sehen. Dann erhält man folgende Verteilung auf die sechs Klassen:



Man erkennt, dass die Teilmodelle 1 und 3 nicht einzeln vorkommen, sondern gemeinsam den Hauptbestandteil von dreien der gebildeten Klassen stellen, dass Teilmodell 4 in verschiedenen Kombinationen vorkommt, insbesondere zusammen mit Relationen aus Teilmodell 3, und Teilmodell 2 stellt alle Mitglieder einer der neuen Klassen.

Im Anhang A, Seite 121, sind zwei Teilmodelle farblich nach Klassenzugehörigkeit eingefärbt, so dass man beurteilen kann, ob und wie gut die Klassenbildung mit fachlichen Gliederungen in Einklang steht.

Der Ellbogen bei der Klassenzahl 4 taucht auch auf, wenn zwei alternative Heterogenitätsmaße verwendet werden, nämlich

$$h_1(K) := \frac{1}{2|K|} \sum_{\substack{R_1,R_2 \in K \\ R_1 \neq R_2}} d(R_1,R_2), \quad \text{(BStandard)}$$

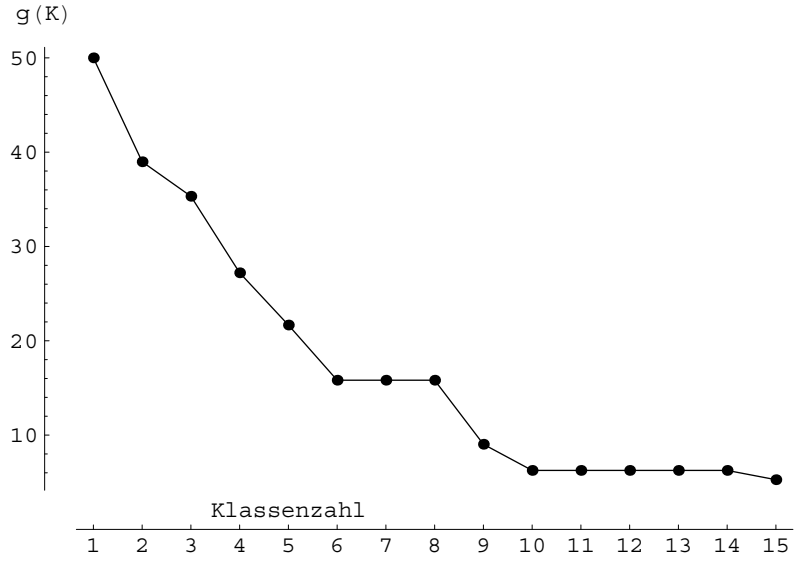
$$h_2(K) := \frac{1}{|K|(|K|+1)} \sum_{\substack{R_1,R_2 \in K \\ R_1 \neq R_2}} d(R_1,R_2), \quad \text{(durchschn. Abstand)}$$

Als andere Gütemaße kommen in Frage

$$g_1(\mathcal{K}) := \sum_{K \in \mathcal{K}} h(K), \quad \text{(Summe der Heterog.)}$$

$$g_2(\mathcal{K}) := \max_{K \in \mathcal{K}} h(K), \quad \text{(Maximum der Heterog.)}$$

Beide erwiesen sich nur in Verbindung mit dem *BStandard* als Heterogenitätsmaß sinnvoll. g_2 ergab dabei einen stufenförmigen Verlauf:



Hier kommen als Klassenzahlen 6 oder 10 in Frage. g_1 hingegen zeigte keinen ausgeprägten Ellbogen im Bereich unter 15 Klassen.

3.5 Zusammenfassung

Die in den vorigen Abschnitten diskutierten Beispiele zeigen, dass man die Clusterbildung nach Ähnlichkeitswerten von Relationen zur Verbesserung der Anordnung von Datenmodellen sinnvoll einsetzen kann. Es lässt sich auch eine Empfehlung geben, welches die günstigste Kombination von Verfahren ist:

Metrik: Tanimoto-Metrik

Proximitätsmaß: Wardsches Maß

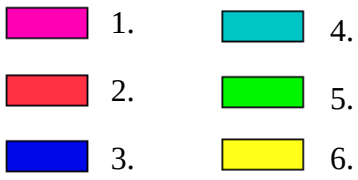
Heterogenitätsmaß: maximale Distanz innerhalb einer Klasse

Gütemaß: gewichtete Summe der Heterogenitäten

Die Ergebnisse einer Clusteranalyse sind im Kern weitgehend stabil gegen eine Änderung der Methode. Durch Vergleich der Ergebnisse bei Einsatz verschiedener Varianten und durch manuelle Nacharbeit lassen sich noch weitere Verbesserungen erreichen.

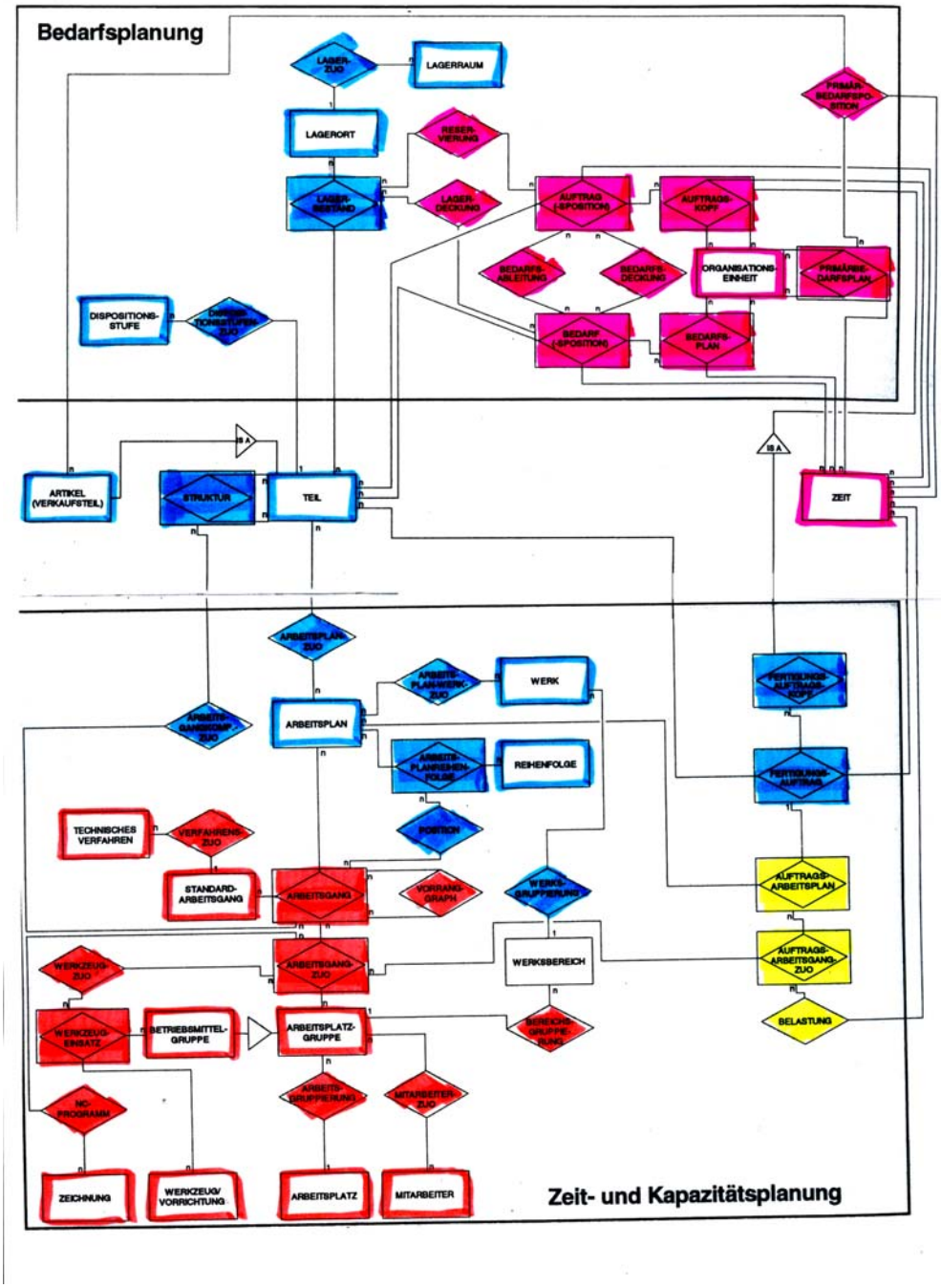
A Zuordnung Teilmodelle - Klassen

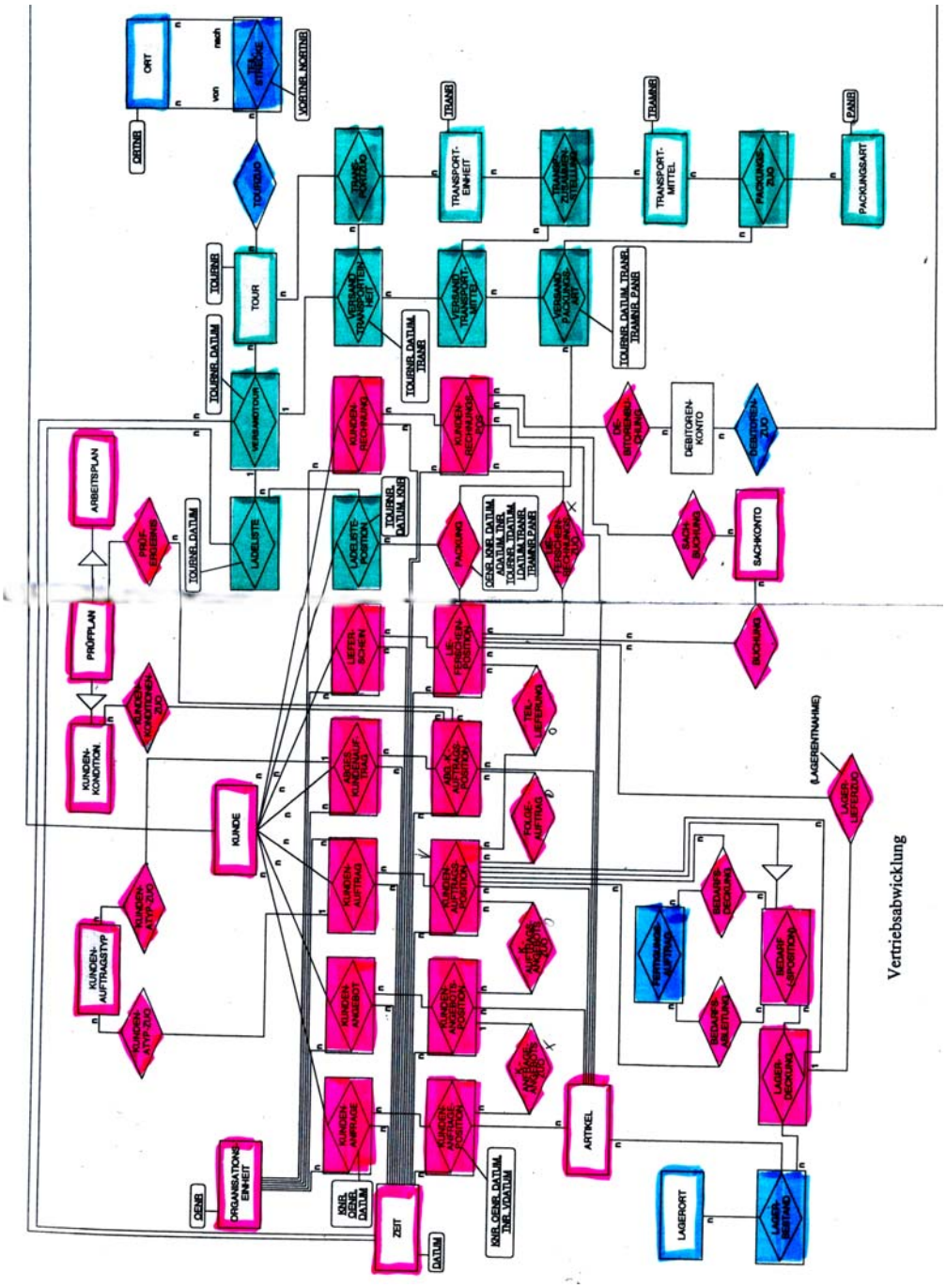
Auf den folgenden Seiten finden sich Abbildungen zweier Teilmodelle aus [SC97], in denen die Relationen und die Entitäten den 6 Klassen aus der Clusteranalyse durch eine farbliche Kodierung zugeordnet werden. Relationen sind ausgefüllt, Entitäten farbig umrandet. Die Entitäten wurden dabei derjenigen Klasse zugeordnet, zu der sie die meisten Beziehungen haben. Einige wenige Entitäten wurden nicht eingefärbt, da sich die Zuordnung nicht eindeutig bestimmen ließ. Die Zuordnung von Farben zu Klassen ist der folgenden Abbildung zu entnehmen:



Wenn nur 4 Klassen gebildet werden, dann fallen die Nummern 3, 4 und 5 in eine Klasse zusammen, das sind die blau-grünen Farben.

Die beiden Abbildungen stellen die Teilmodelle Bedarfs-, Zeit- und Kapazitätsplanung und anschließend die Vertriebsabwicklung dar.


Bedarfsplanung
LAGER-ZUO
LAGERRAUM
LAGERORT
RESER-VIERUNG
LAGER-BESTAND
LAGER-DECKUNG
AUFTRAG (-SPOSITION)
AUFTRAGS-KOPF
PRIMÄR-BEDARFSPO-SITION
BEDARFS-ABLEITUNG
BEDARFS-DECKUNG
ORGANISATIONS-EINHEIT
PRIMÄRBE-DARFSPLAN
DISPOSITIONS-STUFE
DISPOSITIONSSTUFE-ZUO
BEDARF (-SPOSITION)
BEDARFS-PLAN
ISA
ARTIKEL (VERKAUFSTEIL)
STRUKTUR
TEIL
ZEIT
ARBEITSPLAN-ZUO
ARBEITS-PLANWERK-ZUO
WERK
ARBEITS-GANGSORT-ZUO
ARBEITSPLAN
FERTIGUNGS-AUFTRAGS-KOPF
ARBEITS-PLANREIHEN-FOLGE
REIHENFOLGE
FERTIGUNGS-AUFTRAG
TECHNISCHES VERFAHREN
VERFAHRENS-ZUO
POSITION
STANDARD-ARBEITSGANG
ARBEITSGANG
VORRANG-GRAPH
WERKS-GRUPPIERUNG
AUFTRAGS-ARBEITSPLAN
WERKZEUG-ZUO
ARBEITSGANG-ZUO
WERKSBEREICH
AUFTRAGS-ARBEITSGANG-ZUO
WERKZEUG-EINSATZ
BETRIEBSMITTEL-GRUPPE
ARBEITSPLATZ-GRUPPE
BEREICHS-GRUPPE-RUNG
BELASTUNG
NC-PROGRAMM
ARBEITS-GRUPPIERUNG
MITARBEITER-ZUO
ZEICHNUNG
WERKZEUG/ VORRICHTUNG
ARBEITSPLATZ
MITARBEITER
Zeit- und Kapazitätsplanung



Vertriebsabwicklung

Literaturverzeichnis

[BA96] Backhaus et al.: Multivariate Analyse-Methoden. Springer 1996

[BA86] Batini, Carlo; Lenzerini, Maurizio; Navathe, Shamkant B.: A Comparitive Analysis of Methodologies for Database Schema Integration. ACM Computing Surveys 18 (1986) 4, 323-364

[BE81] Siegfried Bergs: Optimalität bei Cluster-Analysen. Diss. Münster 1981

[CH76] Chen, Peter Pin-Shan: The Entity-Relationship Model - Toward a Unified View of Data. ACM Transactions on Database Systems 1 (1976) 1, S.9-36

[EV01] Brian S. Everitt, Sabine Landau, Morven Leese: Cluster analysis. London [u.a.] : Arnold [u.a.], 2001

[FE05] Fettke, Peter; Loos, Peter: Zur Idenfikation von Strukturanalogien in Datenmodellen. Wirtschaftsinformatik 47 (2005) 2, S. 89-100

[JA02] Helmut Jarosch: Datenbankentwurf - Eine beispielorientierte Einführung für Studenten und Praktiker. Vieweg 2002

[LO97] P. Loos: Capture More Data Semantic Through The Expanded Entity-Relationship Model, Arbeitsbericht des Instituts für Wirtschaftsinformatik, Nr. 53, Münster, März 1997

[RA92] Otto Rauh, Eberhard Stickel: Entity Tree Clustering - A Method for Simplifying ER Designs. In: Günther Permul, A. Min Tjoa (Eds.): Entity-Relationship Approach - ER'92, 11^{th} International Conference on the Entity-Relationship Approach, Karlsruhe, Germany, October 7-9, 1992. Lecture Notes in Computer Science 645, Springer 1992, S. 62–78

[SI93] Sinz, E.: Datenmodellierung im Strukturierten Entity-Relationship-Modell (SERM), in: Müller-Ettrich, G. (Hrsg.): Fachliche Modellierung von Informationssystemen - Methoden, Vorgehen, Werkzeuge. Bonn - Paris 1993, S. 63–126

[SC97] Scheer, August-Wilhelm: Wirtschaftsinformatik - Referenzmodelle für industrielle Geschäftsprozesse. Berlin et al. 1997

Wir danken unseren Sponsoren:

EXCONSO

Förderverein Netzwerk Wirtschaft
Universität Lüneburg e.V.

MANZKE®

NIELSEN+PARTNER

NundP ES GmbH

werum
SOFTWARE & SYSTEMS

In der Reihe FINAL sind bisher erschienen:

1. Jahrgang 1991:

1. Hinrich E. G. Bonin; Softwaretechnik, Heft 1, 1991 (ersetzt durch Heft 2, 1992).
2. Hinrich E. G. Bonin (Herausgeber); Konturen der Verwaltungsinformatik, Heft 2, 1991 (überarbeitet und erschienen im Wissenschaftsverlag, Bibliographisches Institut & F. A. Brockhaus AG, Mannheim 1992, ISBN 3-411-15671-6).

2. Jahrgang 1992:

1. Hinrich E. G. Bonin; Produktionshilfen zur Softwaretechnik --- Computer-Aided Software Engineering --- CASE, Materialien zum Seminar 1992, Heft 1, 1992.
2. Hinrich E. G. Bonin; Arbeitstechniken für die Softwareentwicklung, Heft 2, 1992 (3. überarbeitete Auflage Februar 1994), PDF-Format.
3. Hinrich E. G. Bonin; Object-Orientedness --- a New Boxologie, Heft 3, 1992.
4. Hinrich E. G. Bonin; Objekt-orientierte Analyse, Entwurf und Programmierung, Materialien zum Seminar 1992, Heft 4, 1992.
5. Hinrich E. G. Bonin; Kooperative Produktion von Dokumenten, Materialien zum Seminar 1992, Heft 5, 1992.

3. Jahrgang 1993:

1. Hinrich E. G. Bonin; Systems Engineering in Public Administration, Proceedings IFIP TC8/ WG8.5: Governmental and Municipal Information Systems, March 3--5, 1993, Lüneburg, Heft 1, 1993 (überarbeitet und erschienen bei North-Holland, IFIP Transactions A-36, ISSN 0926-5473).
2. Antje Binder, Ralf Linhart, Jürgen Schultz, Frank Sperschneider, Thomas True, Bernd Willenbockel; COTEXT --- ein Prototyp für die kooperative Produktion von Dokumenten, 19. März 1993, Heft 2, 1993.
3. Gareth Harries; An Introduction to Artificial Intelligence, April 1993, Heft 3, 1993.
4. Jens Benecke, Jürgen Grothmann, Mark Hilmer, Manfred Hölzen, Heiko Köster, Peter Mattfeld, Andre Peters, Harald Weiss; ConFusion --- Das Produkt des AWÖ-Projektes 1992/93, 1. August 1993, Heft 4, 1993.
5. Hinrich E. G. Bonin; The Joy of Computer Science --- Skript zur Vorlesung EDV ---, September 1993, Heft 5, 1993 (4. ergänzte Auflage März 1995).
6. Hans-Joachim Blanke; UNIX to UNIX Copy --- Interactive application for installation and configuration of UUCP ---, Oktober 1993, Heft 6, 1993.

4. Jahrgang 1994:

1. Andre Peters, Harald Weiss; COMO 1.0 --- Programmierumgebung für die Sprache COBOL --- Benutzerhandbuch, Februar 1994, Heft 1, 1994.
2. Manfred Hölzen; UNIX-Mail --- Schnelleinstieg und Handbuch ---, März 1994, Heft 2, 1994.
3. Norbert Kröger, Roland Seen; EBrain --- Documentation of the 1994 AWÖ-Project Prototype ---, June 11, 1994, Heft 3, 1994.
4. Dirk Mayer, Rainer Saalfeld; ADLATUS --- Documentation of the 1994 AWÖ-Project Prototype -- -, July 26, 1994, Heft 4, 1994.
5. Ulrich Hoffmann; Datenverarbeitungssystem 1, September 1994, Heft 5, 1994. (2. überarbeitete Auflage Dezember 1994).
6. Karl Goede; EDV-gestützte Kommunikation und Hochschulorganisation, Oktober 1994, Heft 6 (Teil 1), 1994.
7. Ulrich Hoffmann; Zur Situation der Informatik, Oktober 1994, Heft 6 (Teil 2), 1994.

5. Jahrgang 1995:

1. Horst Meyer-Wachsmuth; Systemprogrammierung 1, Januar 1995, Heft 1, 1995.
2. Ulrich Hoffmann; Datenverarbeitungssystem 2, Februar 1995, Heft 2, 1995.
3. Michael Guder / Kersten Kalischefski / Jörg Meier / Ralf Stöver / Cheikh Zeine; OFFICE-LINK --- Das Produkt des AWÖ-Projektes 1994/95, März 1995, Heft 3, 1995.
4. Dieter Riebesehl; Lineare Optimierung und Operations Research, März 1995, Heft 4, 1995.
5. Jürgen Mattern / Mark Hilmer; Sicherheitsrahmen einer UTM-Anwendung, April 1995, Heft 5, 1995.

6. Hinrich E. G. Bonin; Publizieren im World-Wide Web --- HyperText Markup Language und die Kunst der Programmierung ---, Mai 1995, Heft 6, 1995.
7. Dieter Riebesehl; Einführung in Grundlagen der theoretischen Informatik, Juli 1995, Heft 7, 1995.
8. Jürgen Jacobs; Anwendungsprogrammierung mit Embedded-SQL, August 1995, Heft 8, 1995.
9. Ulrich Hoffmann; Systemnahe Programmierung, September 1995, Heft 9, 1995 (ersetzt durch Heft 1, 1999).
10. Klaus Lindner; Neuere statistische Ergebnisse, Dezember 1995, Heft 10, 1995.

6. Jahrgang 1996:

1. Jürgen Jacobs / Dieter Riebesehl; Computergestütztes Repetitorium der Elementarmathematik, Februar 1996, Heft 1, 1996.
2. Hinrich E. G. Bonin; "Schlanker Staat" & Informatik, März 1996, Heft 2, 1996.
3. Jürgen Jacobs; Datenmodellierung mit dem Entity-Relationship-Ansatz, Mai 1996, Heft 3, 1996.
4. Ulrich Hoffmann; Systemnahe Programmierung, (2. überarbeitete Auflage von Heft 9, 1995), September 1996, Heft 4, 1996 (ersetzt durch Heft 1, 1999).
5. Dieter Riebesehl; Prolog und relationale Datenbanken als Grundlagen zur Implementierung einer NF2-Datenbank (Sommer 1995), November 1996, Heft 5, 1996.

7. Jahrgang 1997:

1. Jan Binge, Hinrich E. G. Bonin, Volker Neumann, Ingo Stadtsholte, Jürgen Utz; Intranet-/Internet-Technologie für die Öffentliche Verwaltung --- Das AWÖ-Projekt im WS96/97 --- (Anwendungen in der Öffentlichen Verwaltung), Februar 1997, Heft 1, 1997.
2. Hinrich E. G. Bonin; Auswirkungen des Java-Konzeptes für Verwaltungen, FTVI'97, Oktober 1997, Heft 2, 1997.

8. Jahrgang 1998:

1. Hinrich E. G. Bonin; Der Java-Coach, Heft 1, Oktober 1998, (CD-ROM, PDF-Format; aktuelle Fassung).
2. Hinrich E. G. Bonin (Hrsg.); Anwendungsentwicklung WS 1997/98 --- Programmierbeispiele in COBOL & Java mit Oracle, Dokumentation in HTML und tcl/tk, September 1998, Heft 2, 1998 (CD-ROM).
3. Hinrich E. G. Bonin (Hrsg); Anwendungsentwicklung SS 1998 --- Innovator, SNiFF+, Java, Tools, Oktober 1998, Heft 3, 1998 (CD-ROM).
4. Hinrich E. G. Bonin (Hrsg); Anwendungsentwicklung WS 1998 --- Innovator, SNiFF+, Java, Mail und andere Tools, November 1998, Heft 4, 1998 (CD-ROM).
5. Hinrich E. G. Bonin; Persistente Objekte --- Der Elchtest für ein Java-Programm, Dezember 1998, Heft 5, 1998 (CD-ROM).

9. Jahrgang 1999:

1. Ulrich Hoffmann; Systemnahe Programmierung (3. überarbeitete Auflage von Heft 9, 1995), Juli 1999, Heft 1, 1999 (CD-ROM und Papierform), Postscript-Format, zip-Postscript-Format, PDF-Format und zip-PDF-Format.

10. Jahrgang 2000:

1. Hinrich E. G. Bonin; Citizen Relationship Management, September 2000, Heft 1, 2000 (CD-ROM und Papierform), PDF-Format.
2. Hinrich E. G. Bonin; WI>DATA --- Eine Einführung in die Wirtschaftsinformatik auf der Basis der Web_Technologie, September 2000, Heft 2, 2000 (CD-ROM und Papierform), PDF-Format.
3. Ulrich Hoffmann; Angewandte Komplexitätstheorie, November 2000, Heft 3, 2000 (CD-ROM und Papierform), PDF-Format.
4. Hinrich E. G. Bonin; Der kleine XMLer, Dezember 2000, Heft 4, 2000 (CD-ROM und Papierform), PDF-Format, aktuelle Fassung.

11. Jahrgang 2001:

1. Hinrich E. G. Bonin (Hrsg.): 4. SAP-Anwenderforum der FHNON, März 2001, (CD-ROM und Papierform), Downloads & Videos.
2. J. Jacobs / G. Weinrich; Bonitätsklassifikation kleiner Unternehmen mit multivariater linear Diskriminanzanalyse und Neuronalen Netzen; Mai 2001, Heft 2, 2001, (CD-ROM und Papierform), PDF-Format und MS Word DOC-Format

3. K. Lindner; Simultanttestprozedur für globale Nullhypothesen bei beliebiger Abhängigkeitsstruktur der Einzeltests, September 2001, Heft 3, 2001 (CD-ROM und Papierform).

12. Jahrgang 2002:

1. Hinrich E. G. Bonin: Aspect-Oriented Software Development. März 2002, Heft 1, 2002 (CD-ROM und Papierform), PDF-Format.
2. Hinrich E. G. Bonin: WAP & WML --- Das Projekt Jagdzeit ---. April 2002, Heft 2, 2002 (CD-ROM und Papierform), PDF-Format.
3. Ulrich Hoffmann: Ausgewählte Kapitel der Theoretischen Informatik (CD-ROM und Papierform), PDF-Format.
4. Jürgen Jacobs / Dieter Riebesehl; Computergestütztes Repetitorium der Elementarmathematik, September 2002, Heft 4, 2002 (CD-ROM und Papierform), PDF-Format.
5. Verschiedene Referenten; 3. Praxisforum "Systemintegration", 18.10.2002, Oktober 2002, Heft 5, 2002 (CD-ROM und Papierform), Praxisforum.html (Web-Site).

13. Jahrgang 2003:

1. Ulrich Hoffmann; Ausgewählte Kapitel der Theoretischen Informatik; Heft 1, 2003, (CD-ROM und Papierform) PDF-Format.
2. Dieter Riebesehl; Mathematik 1, Heft 2, 2003, (CD-ROM und Papierform) PDF-Format.
3. Ulrich Hoffmann; Mathematik 1, Heft 3, 2003, (CD-ROM und Papierform) PDF-Format und Übungen.
4. Verschiedene Autoren; Zukunft von Verwaltung und Informatik, Festschrift für Heinrich Reinermann, Heft 4, 2003, (CD-ROM und Papierform) PDF-Format.

14. Jahrgang 2004:

1. Jürgen Jacobs; Multilayer Neural Networks; Heft 1, 2004, (CD-ROM und Papierform) PDF-Format.

15. Jahrgang 2005:

1. Ulrich Hoffmann; Mathematik für Wirtschaftsinformatiker; Heft 1, 2005, (CD-ROM und Papierform) PDF-Format.
2. Ulrich Hoffmann; Übungen & Lösungen zur Mathematik für Wirtschaftsinformatiker; Heft 1, 2005, (CD-ROM und Papierform) PDF-Format.
3. Ulrich Hoffmann; Datenstrukturen & Algorithmen; Heft 2, 2005, (CD-ROM und Papierform) PDF-Format.

16. Jahrgang 2006:

1. Hinrich E. G. Bonin; Systemanalyse für Softwaresysteme; Heft 1, August 2006, (CD-ROM und Papierform) PDF-Format.
2. Hinrich E. G. Bonin; Faszination Programmierung; Heft 2, August 2006, (CD-ROM und Papierform) PDF-Format.
3. Dieter Riebesehl; Strukturanalogien in Datenmodellen, Heft 3, Dezember 2006, (CD-ROM und Papierform) PDF-Format.

17. Jahrgang 2007:

1. Ulrich Hoffmann; Ausgewählte Kapitel der Theoretischen Informatik; Heft 1, August 2007, (CD-ROM und Papierform) PDF-Format.
2. Ulrich Hoffmann; Mathematik für Wirtschaftsinformatiker und Informatiker; Heft 2, August 2007, (CD-ROM und Papierform) PDF-Format.
3. Hinrich E. G. Bonin; Der Java-Coach, Heft 3, September 2007, (CD-ROM und Papierform) PDF-Format.
4. Jürgen Jacobs; Dichteprognose autoregressiver Zeitreihen, Heft 4, September 2007, (CD-ROM und Papierform) PDF-Format.

Herausgeber:
Prof. Dr. Dipl.-Ing. Dipl.-Wirtsch.-Ing. Hinrich E. G. Bonin
Leuphana Universität Lüneburg, Volgershall 1, D-21339 Lüneburg, Germany
email: bonin@uni.leuphana.de

Lektorat für diese Ausgabe:
Maja Irmhild Schütte-Hoof MA
Lektorin für Deutsch im Fremdsprachenzentrum der Leuphana Universität Lüneburg

Verlag:
Eigenverlag (Fotographische Vervielfältigung), Leuphana Universität Lüneburg (vormals Fachhochschule Nordostniedersachsen)

Erscheinungsweise:
ca. 4 Hefte pro Jahr.
Für unverlangt eingesendete Manuskripte wird nicht gehaftet. Sie sind aber willkommen.

LEUPHANA
UNIVERSITÄT LÜNEBURG

Prof. Dr. Horst Meyer-Wachsmuth