Inhaltsverzeichnis

1 Einleitung und Motivation 4

1.1 Ziele des Kapitels 4

1.2 Aufgabenstellung 4

1.3 Einleitung 5
1.3.1 Was ist Business Engineering? 5
1.3.2 Gründe für ein Business Engineering Projekt 6
1.3.3 Software 9
1.3.4 Probleme bei Software Projekten 15

1.4 Lösungsidee und Vorgehensmodell 16
1.4.1 Das Business Reengineering-Labor Konzept 16
1.4.2 Das Business Engineering-Labor Konzept 18

1.5 Zusammenfassung des Kapitels 27

2 Grundlagen 28

2.1 Ziele des Kapitels 28

2.2 Modellierung 28
2.2.1 Allgemeines 28
2.2.2 Einsatz von Modellen 29
2.2.3 Modelltypen 30

2.3 Softwaretechnik 32
2.3.1 Vorgehensmodelle in der Softwareentwicklung 32
2.3.2 Techniken des Software-Engineering 35
2.3.3 Qualität aus software-technischer Sicht 36

2.4 Mensch-Maschine Kommunikation 37
2.4.1 Mensch Maschine Kommunikation allgemein 37
2.4.2 Gestaltung von Schnittstellen 38

2.5 Betriebswirtschaftliche Konzepte 40
2.5.1 Organisation 41
2.5.2 Prozeßorientierte Organisation 42

2.6 Ziele und Zielsysteme 43

2.7 Zusammenfassung des Kapitels 45

3 Einsatz von IT im gemeinsamen Gestaltungsprozeß 46

3.1 Ziele des Kapitels 46

3.2 Unternehmensmodellierung 46
3.2.1 Allgemeines 46
3.2.2 Modellierung 47

3.3 Softwareunterstützung allgemein 48

3.4 Unternehmensmodellierungssoftware 50
3.4.1 Grundsätzliches 50
3.4.2 Das verwendete Werkzeug: AENEIS 50

3.4.3 Das ARIS-Toolset der IDS Scheer ... 54
3.4.4 ADONIS von BOC ... 56
3.4.5 Integrationsansätze ... 58
3.5 Zusammenfassung des Kapitels ... 58
4 Fallstudien ... 60
4.1 Ziele des Kapitels ... 60
4.2 Erste Fallstudie mit Firma XXX ... 60
4.2.1 Ausgangssituation ... 60
4.2.2 Vorgehensweise ... 61
4.2.3 Resümee der ersten Fallstudie ... 62
4.3 Zweite Fallstudie: Firma YYY ... 64
4.4 Ausgangssituation ... 65
4.4.1 Szenario ... 65
4.4.2 Das Software Produkt: Applix ... 66
4.4.3 Vorgeschichte ... 67
4.5 Ziele des Fallstudienprojektes ... 68
4.6 Vorgehensweise ... 69
4.6.1 Darstellung des Modellinhaltes ... 71
4.6.2 Modellierung ... 74
4.7 Zusammenfassung des Kapitels ... 77
5 Analyse und Erkenntnisse ... 78
5.1 Ziele des Kapitels ... 78
5.2 Vorgehensmodell ... 78
5.3 Modellierung ... 79
5.3.1 Ist- und Soll-Modellierung ... 79
5.3.2 Darstellung des Modellinhaltes ... 79
5.3.3 Identifikation ... 83
5.3.4 Zustand des Modelles ... 84
5.3.5 Einfügen von Dokumentation ... 85
5.3.6 Integration der Schnittstellen ... 85
5.3.7 Hierarchisierung von Schnittstellen ... 86
5.3.8 Informationen über Eingabeelemente ... 88
5.3.9 Namensgebung ... 88
5.4 Modellierungswerkzeug ... 89
5.5 Resümee ... 89
6 Verweise ... 91
6.1 Literatur ... 91
6.2 Abbildungen ... 95
6.3 Definitionen ... 96
7 Anhang: Fallstudie YYY ... 97
7.1 Strategie-Papier ... 98

7.2 Projekt-Tagebuch ... 100
7.2.1 Koordinations-Treffen (26.04.99) ... 100
7.2.2 1. Treffen (07.05.99) ... 100
7.2.3 2. Treffen (21.05.99) ... 100
7.2.4 3. Treffen (28.05.99) ... 100
7.2.5 4. Treffen (09.06.99) ... 100
7.2.6 5. Treffen (23.06.99) ... 101

7.3 Soll-Modell - Aktiv-Vertrieb (AENEIS-Modell) ... 102

7.4 Soll-Modell - Aktiv-Vertrieb (Gesamtbericht - HTML) ... 103
7.4.1 Berichtsstruktur ... 104
7.4.2 Index ... 105
7.4.3 Ziele ... 105
7.4.4 Geschäftsprozesse ... 105
7.4.5 Aktiv-Vertrieb ... 106
7.4.6 Produktidee einbringen ... 108
7.4.7 Produktidee prüfen ... 109
7.4.8 Verkaufskonzept ausarbeiten ... 111
7.4.9 Aktivmaßnahme durchführen ... 112
7.4.10 Diagramme ... 113
7.4.11 Beschreibung des Informationssystems ... 113
Infrastruktur ... 114
7.4.13 Applix Sales ... 115
7.4.14 Applix Enterprise ... 116
7.4.15 Applix Datenbank ... 116
7.4.16 Unternehmens-DB(en) ... 116
7.4.17 Applix-Sales Formulare ... 117
7.4.18 Applix-Sales Formulare ... 118
7.4.19 Applix-Arbeitsvorrat ... 119
7.4.20 Applix-Kunden-/Produktmatrix ... 120
7.4.21 Applix-Kundendaten ... 121
7.4.22 Applix-Produktidee ... 122
7.4.23 Applix-Produktinformation ... 123
7.4.24 Applix-Terminplaner ... 124
7.4.25 Applix-Verkaufsunterstützung ... 125
7.4.26 Applix-Werbekampagnenmanagement ... 126
7.4.27 Applix-WKM-Kampagnenotizen ... 128
7.4.28 Applix-WKM-Kampagnestatistiken ... 129
7.4.29 Applix-WKM-Tagebuch ... 130
7.4.30 Applix-WKM-Werbeprojekte ... 132
7.4.31 Organigramm ... 133

7.5 Soll-Modell - Aktiv-Vertrieb (Abläufe) ... 134
7.5.1 Produktidee einbringen (BM) ... 134
7.5.2 Produktidee prüfen (KP) ... 134

7.6 Soll-Modell - Aktiv-Vertrieb (Bericht über die Software, Pflichtenheft- Text) ... 137
7.6.1 Beschreibung Gesamtsystem ... 137
7.6.2 Beispiel Hardwarebeschreibung ... 138
7.6.3 Beispiel Schnittstelleninformationen ... 139

1 Einleitung und Motivation

1.1 Ziele des Kapitels

Zuerst soll die Aufgabenstellung und damit verbundene Probleme beschrieben werden. Dabei wird vorwiegend aus der Sicht der Softwareentwicklung vorgegangen, zusätzlich werden betriebswirtschaftliche Gesichtspunkte eingearbeitet.

Der Beschreibung der Aufgabenstellung folgt eine Lösungsidee, die als Vorgehensmodell beschrieben wird. Dieses (Meta-) Vorgehensmodell beschäftigt sich nicht, wie Vorgehensmodelle zur Softwareentwicklung, mit der Frage: „Wie macht man Software richtig?“. Die zentrale Frage lautet vielmehr: „Wie macht man die richtige Software für den richtigen Geschäftsprozeß?“.

1.2 Aufgabenstellung

Der Einsatz von Informationstechnologie in einem Unternehmen hat dort nicht die Erwartungen erfüllt, wo er umfassend entlang der gesamten Wertschöpfung eingesetzt wurde. Punktuelle Erfolge konnten in einigen Bereichen erzielt werden (Lager, Buchhaltung, CAD).

Vermutliche Ursachen sind:

- Die *Arbeitsabläufe*, die durch die Software *unterstützt* werden sollen, wurden nicht in Hinsicht auf die Unterstützung durch Software gestaltet.
- Ein *sequentielles Vorgehen* (Design der Software → Arbeitsablaufgestaltung oder Arbeitsablaufgestaltung → Design der Software) ist kritisch, wenn die Wechselwirkungen zwischen Software und Ablauforganisation unberücksichtigt bleiben.

Um den Aspekt der *gemeinsamen* Gestaltung und Entwicklung von Unternehmenssoftware sowie den dadurch unterstützten Geschäftsprozessen in den Vordergrund zu rücken, wird in dieser Diplomarbeit ein Vorgehen erprobt, bei dem durch die Verwendung eines Unternehmensmodellierungswerkzeuges (das vordergründig die Gestaltung von Geschäftsprozessen zum Ziel hat) sowohl technisches als auch betriebswirtschaftliches Wissen in den Gestaltungsprozeß einfließt.

Das soll im Rahmen eines modifizierten Business Engineering Projektes geschehen, das gemeinsam von einem Betriebswirt und einem Informatiker mit Hilfe eines Unternehmensmodellierungswerkzeuges bearbeitet wird.

1.3 Einleitung

1.3.1 Was ist Business Engineering?

Zuerst eine Beschreibung des Begriffes Business Engineering von James Martin aus [Balzert96-2] (übersetzt und zusammengefaßt):

Definition 1: Business Engineering (synonym Enterprise Engineering)

> ... ist eine ingenieurmäßige Gestaltung eines Unternehmens, unter Verwendung einer Menge von Techniken zur Erreichung von speziellen Zielen.

Wesentliche Elemente des Business Engineering sind [Schröder98]:

- *Kundenorientierte Prozeßgestaltung* (horizontale Integration). Der Kunde rückt bei der Gestaltung der Organisation in den Mittelpunkt des Interesses.
- *Prozeßorientierte Arbeitsorganisation* (vertikale Integration). Der Arbeitsablauf wird nicht strikt vorgeschrieben, dafür gibt es Zielvorgaben, die mit Hilfe eines Handlungsrahmens erreicht werden sollen. Ein Geschäftsprozeß wird je nach Komplexität, Aufwand oder anderen Unterteilungen von einem Case Worker oder Case Team bearbeitet. So entstehen multifunktionale Arbeitsbilder und Veränderungen der Wertesysteme und Führungsmethodik.
- Einsatz der *Informationstechnologie zur Prozeßunterstützung* und zur Schaffung neuer Prozesse.

Die Ziele eines Business Engineering Projektes sind im hier gegebenen Zusammenhang:

- Eine *Umgestaltung der Arbeitsabläufe* eines Unternehmens und die Zuteilung von Ressourcen. Vorwiegend werden Arbeitsabläufe prozeßorientiert gestaltet, funktionale und prozeßorientierte Organisationsformen können aber auch sinnvoll nebeneinander existieren.

- *Verbesserung der Unterstützung* der Arbeitsabläufe durch den Einsatz einer neuen Unternehmenssoftware. Entwicklung, Anpassung oder Auswahl einer Software und Einführung derselben.
- *Gemeinsame* und *ganzheitliche Betrachtung* der beiden oben genannten Ziele, speziell der wechselseitigen Abhängigkeiten.

Das Konzept des Business Engineering baut prinzipiell auf den Charakteristika prozeßorientierter Unternehmensorganisation auf und ist als Erweiterung des Business (Process) Reengineering Konzeptes von Hammer und Champy [Hammer94] zu sehen.

1.3.2 Gründe für ein Business Engineering Projekt

Die Gründe für ein Business Engineering Projekt sind in gesamtwirtschaftlichen Veränderungen zu suchen. Daraus lassen sich zwei Arten der Motivation für ein Unternehmen ableiten:

- *Auftreten von Problemen*
- *Wahrnehmung einer Chance*

Zuerst eine kurze Beschreibung der gesamtwirtschaftlichen Veränderungen.

1.3.2.1 Gesamtwirtschaftliche Veränderungen

Einige dieser Veränderungen seien hier kurz genannt:

- *Globalisierung der Märkte*: Die weltweite Vernetzung (Internet) hebt Entfernungen auf. Wettbewerber sind nun nicht mehr nur in örtlicher Nähe zu suchen. Durch den Eintritt neuer Wettbewerber entsteht eine Intensivierung des Wettbewerbs. Zudem steigt die *Transparenz der Märkte* an: Preisvergleiche können in kürzester Zeit eingeholt werden. Zusätzlich hat sich die Wettbewerbssituation durch den Beitritt zur Europäischen Union verändert. Sinngemäß aus [Schröder98].
- Verstärkter *Einsatz* von *Informationstechnologie*: Immer mehr Unternehmen verwenden neben den oben genannten Kommunikationstechnologien (Internet etc.) Software von großen Anbietern. Firmen wie SAP, BAAN oder PeopleSoft üben einen nicht zu unterschätzenden Gruppenzwang aus, auch deshalb, weil sie die Perspektive eines virtuellen

Marktplatzes anbieten. Das bedeutet, daß der Handel zwischen verschiedenen, diese Software nutzenden Firmen, deutlich vereinfacht und automatisiert werden soll.

- Realität des *Käufermarktes*: Ein Käufermarkt besteht, wenn der Kunde die Möglichkeit hat, aus einer Anzahl von Anbietern auszuwählen. Dadurch entsteht verstärkter Druck auf die Anbieter bezüglich des Preises, des Leistungsangebotes und der Qualitätsanforderungen von Produkten und Dienstleistungen. Sinngemäß aus [Schröder98].

Die Wettbewerbssituation des Unternehmens kann durch die genannten Faktoren gefährdet werden. Zusammengefaßt werden die Anforderungen an ein Unternehmen durch drei Faktoren gekennzeichnet: *Kunden, Wettbewerb und Wandel* [Hammer94].

Informationstechnik spielt als sogenannter „*Enabler*“ eine dominante Rolle bezüglich der genannten Veränderungen. Beispielsweise ist die Globalisierung der Märkte nur durch den Einsatz von Informationstechnologie möglich geworden.

Gleichzeitig ist sie aus der Sicht des Unternehmens ein „*Enabler*“ um auf die Veränderungen des Marktes reagieren und möglicherweise in weiterer Folge zu agieren zu können.

Soviel zu externen Einflußgrößen, jedoch ergeben sich auch innerhalb des Unternehmens wesentliche Änderungen.

In *Arbeitswelt* und *Gesellschaft* ist ein *Wertewandel* zu beobachten. Neben einer zunehmenden Ablehnung von Unterordnung und Hierarchien steht die Forderung nach Handlungsspielraum, Eigenverantwortung und Selbstverwirklichung [Schröder98]. Ein kritischer Faktor ist außerdem die Rolle des Mitarbeiters als Know-how-Träger im Unternehmen [Sauerbrey89]. Dadurch entsteht auch innerhalb des Unternehmens die Notwendigkeit, Veränderungen anzustreben, welche dann meist durch IT-Projekte realisiert werden sollen.

1.3.2.2 Auftreten von Problemen

Wenn bereits konkrete Probleme (eine Gefährdung der Wettbewerbssituation oder geänderte Kundenanforderungen) auftreten, muß nach einer Lösung gesucht werden. Mit Verbesserungen der Software versucht man diese Probleme zu lösen, die zumindest teilweise in der Ablauforganisation liegen:

- Es *gibt keine* durchdachte *Lösung*. Eine abteilungsübergreifende Unterstützung der Geschäftsprozesse eines Unternehmens durch Software wurde nie entwickelt. Schlecht unterstützte und schlecht dokumentierte Arbeitsabläufe, die nur in den Köpfen der Mitarbeiter verankert sind, führen bei Veränderungen zu Problemen. Mängel bestehen also bezüglich Effektivität (Grad der Erreichung der Ziele - Nutzen) und Effizienz (der Aufwand zur Erreichung der Ziele – Kosten).
- Die *Lösung* ist unzureichend: Die Softwareunterstützung ist evolutionär gewachsen. Diese Art von Wachstum entsteht meist ausgehend von konkreten Kristallisationskeimen oder Insellösungen. Ein konkretes Beispiel wäre ein Planungsbüro für Haustechnik. Kristallisationskeimen sind dabei die Buchhaltung und die CAD-Software. In diesen Bereichen waren schon Mitte der achtziger Jahre Produkte am Markt, die einen Quasi-Standard dargestellt haben (z.B. AutoCAD von Autodesk). Ausgehend von diesen Keimen entwickelten sich verschiedene Zusatzwerkzeuge, zum Beispiel zur Angebotserstellung, Erstellung von Stücklisten oder Arbeitszeiterfassung. Diese Werkzeuge hatten dann jeweils Aufgaben zwischen den Kristallisationskeimen zu erfüllen. Dadurch entstanden stark heterogene Systemstrukturen: Verschiedene Softwarepakete, unterschiedliche Datenformate, sowie redundante Daten und Funktionen. Effektiv können die Aufgaben gelöst werden, von Effizienz ist man jedoch weit entfernt.
- Die *Lösung* ist *veraltet*: Die vorhandene Lösung ist zufriedenstellend bezüglich der Effektivität aber nicht bezüglich der Effizienz. Gründe können die Gestaltung der Arbeitsabläufe und/oder eine veraltete Software sein. Manchmal werden Softwareprodukte auch von Seiten des Herstellers oder Anbieters nicht mehr weiterentwickelt – zum Beispiel auf Grund eines Technologiewechsels. Die Anpassung der Software ist für das Unternehmen aus technischen Gründen nicht mehr möglich.

Auswirkungen sind eine starke Beeinträchtigung der Arbeitsabläufe und oft auch ein starkes Unwohlsein der Mitarbeiter, also der Aufgabenträger, was zu einer weiteren Verringerung von Effizienz und Effektivität führt.

Bei einem Projekt, das auf Grund bereits auftretender Probleme initiiert wird, ergeben sich *zusätzliche Schwierigkeiten*: Der Zeitdruck ist groß und die Ressourcen, speziell Human-Ressourcen, sind noch stärker begrenzt.

1.3.2.3 Wahrnehmen einer Chance

Projekte dieser Kategorie werden initiiert, wenn sich ein Unternehmen trotz momentan zufriedenstellender Geschäftssituation in einem oder mehreren Bereichen verbessern will. Visionen zur Veränderung im Unternehmen sind vorhanden. Beispiele dafür sind:

- *Erweiterung des Angebotes*: Neue, verbesserte oder verstärkt individuell zugeschnittene Produkte und/oder Dienstleistungen sollen angeboten werden – Ziele können sein, ein Vergrößern des Marktanteiles oder das Eindringen in neue Marktsegmente.
- *Technologiesprung*: Der Einsatz neuer Technologien kann in verschiedenen Bereichen erfolgen, in der Kommunikation (z.B. Telearbeit), Marketing und Vertrieb (e-commerce) oder in der Fertigung.
- *Einsparungen*: Verstärkte Automatisierung zum Zweck einer Verbesserung der Effizienz oder einer Einsparung an Ressourcen.
- *Verbesserung der Kundenbindung*: Abteilungsübergreifende Sammlung von Wissen über den Kunden zur Verbesserung der Kommunikation und der Antizipation von Kundenwünschen.

Bei einem derartigen Projekt ist der Zeitdruck meist geringer (als bei 1.3.2.2) und auch bei den Ressourcen steht mehr Spielraum zur Verfügung. Besonders eine starke Einbindung von Mitarbeitern (Verstärkte Partizipation in der Gestaltung) ist hier eher möglich, was sich bei der Durchführung des Projektes als vorteilhaft bezüglich der erreichten Qualität des Ergebnisses (Geschäftsprozesse und Softwareunterstützung) erweisen kann.

1.3.3 Software

Softwaretechnische Aspekte und Begriffe wie Qualität oder Vorgehensmodelle zur Softwareentwicklung werden in 2.3 beschrieben. Hier sollen einige grundsätzliche Begriffe geklärt werden.

1.3.3.1 Abgrenzung

Begriffe, die etwas Immaterielles beschreiben, sind oft nicht eindeutig. Auch der Überbegriff Software wird synonym für sehr spezielle Arten von Software verwendet. Wie aus dem

Schichtenmodell in Abb. 1 ersichtlich ist, wird sich diese Arbeit nur mit der „höchststehenden“ Art von Software beschäftigen. Geschäftsprozesse spielen sich auf der Ebene von Anwendungssoftware ab. Die Hierarchie in Abb. 1 zeigt eine mögliche Unterscheidung in Standard- und Individualsoftware.

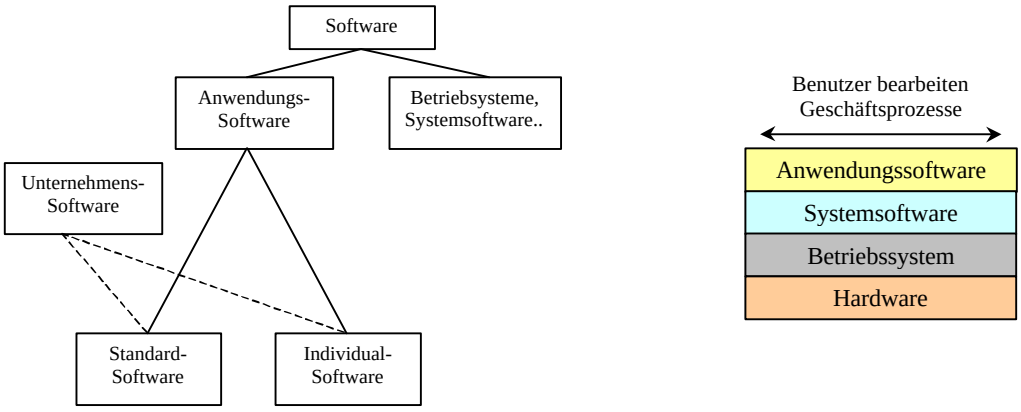

Abb. 1 : Einteilung von Software. Hierarchie (links) und Schichtenmodell (rechts)

Dazu einige Definitionen:

Definition 2: Software [Duden93]

> „Die Gesamtheit aller Programme die auf einer Rechenanlage eingesetzt werden können.“

Prinzipiell steht Software also zwischen dem Menschen und der Maschine. Eine erste und zweckmäßige Untergliederung von Software kann daher erfolgen in *Systemsoftware* (inklusive Betriebssysteme) und *Anwendungssoftware*. Systemsoftware (und das Betriebssystem) ist im hier gegebenen Kontext nicht weiter interessant. Sie dient als Grundlage für die Arbeit mit der Anwendungssoftware. Natürlich ist dabei zu beachten, daß der Übergang teilweise fließend ist: Die dazugehörenden Funktionen werden in weiterer Folge als gegeben angenommen, verschiedene Teile sind in der Anwendungssoftware integriert - wie zum Beispiel Editoren.

Definition 3: Anwendungssoftware [Floyd97]

> „Anwendungssoftware ist ein Mittel zum Zweck, um fachliche *Aufgaben* in einem oder mehreren Anwendungsbereichen zu erledigen. Dazu werden vorhandene oder neue Problem-Lösungsstrategien durch Software realisiert.“

Unternehmenssoftware ist als Standard- und Individualsoftware möglich, siehe Abb. 1. Bei Unternehmenssoftware steht der Aspekt der Zusammenarbeit im Vordergrund – sie dient zur Unterstützung aller im Zusammenhang mit Auftragsgewinnung und -abwicklung durchgeführten Tätigkeiten im Unternehmen. Die Aktivitäten entlang der Wertschöpfungskette stehen dabei im Vordergrund.

Will man nun die Begriffe Standardsoftware (in weiterer Folge abgekürzt als SSW) und Individualsoftware (ISW) definieren, stößt man in der Literatur auf zum Teil recht unterschiedliche Ansichten und Definitionen. Deshalb soll die folgende Unterteilung, anhand der Art und des Grades der Anpassung getroffen werden.

Definition 4: Standardsoftware

> Standardsoftware ist Anwendungssoftware die von einem Softwarehersteller für bestimmte Arbeitsaufgaben gestaltet und entwickelt wurde. Sie soll von mehreren Unternehmen (Kunden des Softwareherstellers) angewandt werden können. Die Anpassung an die gegebenen Anforderungen erfolgt mittels Parametrisierung und nur geringfügigem Programmieraufwand.

Eingangsparameter für die Anpassung und Einführung von Software sind demnach Referenz-Prozesse und die jeweiligen Softwaremodule oder –komponenten (Beispiele SAP [SAP] oder BAAN [BAAN-1 und BAAN-2]). Minimaler Programmieraufwand entsteht bei einer vollkommenen Übernahme der implementierten Geschäftsprozesse; idealisiert betrachtet, müssen in diesem Fall nur die Datenbestände übernommen werden. SSW zeichnet sich durch einen geringeren Entwicklungsaufwand für das jeweilige Unternehmen aus, der „große“ Entwicklungsaufwand wurde schon früher verrichtet.

Charakteristisch ist - durch die einmalige Gestaltung zur mehrfachen Verwendung - eine *begrenzte Anpassungsmöglichkeiten* an Eigenarten des jeweiligen Unternehmens. Aber auch der Aufwand zur Anpassung ist nicht zu unterschätzen.

Beispiel: Ein Beispiel für *Parametrisierung* wird in [Mertens97] beschrieben. Demnach hat das SAP-Modul MM-PP (MM steht für Materialwirtschaft, PP für Produktionsplanung) rund 150 Parameter. 40 davon sind jedem einzelnen Artikel zugeordnet. Bereits für ein kleineres oder mittleres Unternehmen (Rechenbeispiel 2.500 Artikel) ergibt sich eine große Zahl an Parametern (100.000). Diese müssen nicht nur eingestellt, sondern auch gewartet werden.

Der Marktführer im Bereich Standard- Unternehmenssoftware für größere Unternehmen ist die SAP AG (Systeme Anwendungen und Programme), die Zielgruppe umfaßt grundsätzlich größere Unternehmen, wie Banken, Versicherungen und Industriebetriebe, aber auch Non-profit Organisationen [SAP]. Ein weiterer bekannter Anbieter von SSW in Europa ist BAAN [BAAN-1].

Oft werden implizite oder explizite Referenzprozesse oder Referenzmodelle verwendet, um die Software möglichst für bestimmte Geschäftsprozesse vorzubereiten.

- Ein *Referenzprozeß*, oder ein *Referenzmodell*, kann als *implizit* bezeichnet werden, wenn er/es als Grundlage zur Erstellung der Software verwendet wurde, die geplanten Arbeitsschritte aber nur mehr durch die Software – oder durch den Spielraum den die Parameter lassen – repräsentiert werden. Ein Referenzmodell liegt der Software zwar zu Grunde, ist aber nicht in dokumentierter Form vorhanden. Anpassungen oder Erweiterungen werden so erschwert.
- Als *explizit* kann man ein *Referenzmodell* betrachten, wenn das (Prozeß-) Modell, auf dessen Basis die Software entwickelt wurde, vorliegt. Anpassungen oder Erweiterungen können damit leichter vorgenommen werden, da man zur Analyse und Gestaltung das Modell heranziehen kann.

Definition 5: Individualsoftware

> Individualsoftware (ISW) ist Anwendungssoftware die für die Erfüllung der individuellen Arbeitsaufgaben eines speziellen Anwenders mit Software-Entwicklungswerkzeugen entwickelt wurde. ISW wird *einmal* angefertigt und meist auch nur vom Auftraggeber angewandt.

Individualsoftware zeichnet sich demnach durch einen wesentlich höheren Entwicklungsaufwand aber auch durch respektive höhere Anpassungsmöglichkeiten aus, da sie ja per Definition für ein bestimmtes Unternehmen maßgeschneidert wird. Der Anwender ist in der Regel der Eigentümer des Quellcodes.

Von speziellem Interesse ist zur Zeit der Bereich *zwischen Individual- und Standardsoftware*, eine Aufstellung wird in [Mertens97, 8-50] beschrieben. Um das Spektrum aufzuzeigen, eine kurze Aufzählung aus software-technischer Sicht:

- *Branchenspezifische* Vorkonfiguration: Nahe an Standardsoftware, Mertens nennt die Branchenlösungen von SAP als Beispiel [Mertens97], dabei werden branchenspezifische Referenzparameter verwendet. Beispiele sind SAP Oil & Gas, SAP Media, SAP Mill Products,...
- *Templates*: Primäre Wiederverwendung auf konzeptioneller Ebene, beispielsweise in einem CASE-Tool. Bereits realisierte Lösung können so übertragen werden.
- *Componentware*: Der Mittelweg, Standardkomponenten werden individuell verknüpft. Komponenten sind Halbfabrikate, die kleiner sind als Applikationen, aber deutlich größer als Klassen [Balzert96-1]. Beispiel: Fabasoft Components [Faba].
- *Frameworks*. Nahe an Individualsoftware, die Wiederverwendung erfolgt auf softwaretechnischer Ebene (Methoden und Klassen). Diese Technik existiert laut Mertens [Mertens97, S. 33ff] zur Zeit nur konzeptionell.
- *Kernprogramme und Kernarchitekturen*: Standardfunktionen einer Unternehmenssoftware (z.B. Buchhaltung) werden um individuelle, oft branchenspezifische Funktionen ergänzt.

1.3.3.2 Eigenschaften von Software

Software ist ein technisches Artefakt – also ein künstlicher, von Menschen geschaffener Gegenstand. Leicht lassen sich über den technischen Kontext hinaus interessante Sichten erschließen (Auszüge aus [Heintz95]):

- Technische Artefakte sind *Ausdruck der Gesellschaft* in der sie entwickelt wurden. Sie sind also nicht rein technisch zu erklären.
- Im Zuge der Technisierung verlagern sich soziale *Gestaltungsmaßnahmen* in die Technik.
- Durch die *Verlagerung des Sozialen* in technische Artefakte wird das Soziale unsichtbar (versteckt!).
- Die Verlagerung der sozialen Strukturen in die Technik *manifestiert* sie, zumindest in einem gewissen Zeitraum, beispielsweise bis zur Überholung der vorliegenden Technologie. Auch die Lebensdauer von Organisationsformen wird so verlängert [Sauerbrey89].

Beispiel: Die funktionale Arbeitsteilung von Abteilungen. Diese Teilung wurde durch den Einsatz von Insellösungen – durch funktional gegliederte Module – manifestiert. Diese Situa-

tion besteht seit den Anfängen der elektronischen Informationsverarbeitung. Übergreifendes Vorgehen ist erst durch die Entwicklungen in den letzten Jahren realisierbar. Zum Beispiel ermöglicht es Workflow Software, daß Arbeitsergebnisse zwischen den funktionalen Einheiten weitergegeben werden.

Ein berühmtes Beispiel, zur Veranschaulichung der Rolle der Technik als sozialen Gestaltungsfaktor, wird in [Heintz95] beschrieben:

> Wenn man von New York City nach Long Island fährt, passiert man eine Reihe von Autobahnbrücken. Sie wurden in den 30´er Jahren von dem damaligen Stararchitekten Robert Moses gebaut und sind die Verbindung zwischen New York und den Stränden von Long Island. Die Autobahnbrücken von Robert Moses sehen aus wie alle anderen, zu dieser Zeit erbauten Brücken. Dasselbe Material, ein ähnlicher Konstruktionsstil. Auffällig ist nur ihre Höhe: sie sind sehr niedrig. Die Durchfahrtshöhe beträgt nur 2,70 m.

Man nimmt an, daß dieser Umstand wohl finanzielle oder technische Gründe hat, doch falsch! Er hatte soziale Gründe. Um der armen (farbigen) Bevölkerung den Zugang zu den Stränden von Long Island zu erschweren, wurden die Brücken so nieder gebaut, daß sie von Bussen (öffentliche Verkehrsmittel) nicht passiert werden konnten. Der Teil der Bevölkerung der kein eigenes Automobil besaß, wurde somit von der Benutzung der Autobahnbrücke ausgeschlossen.

Leicht lassen sich Analogien für eine verschleierte Manifestierung von sozialen Strukturen bei der Gestaltung von großen Softwaresystemen finden, beispielsweise die Organisation von Benutzerrechten (Zugriffe und Nutzung von Ressourcen) als Implementierung von Hierarchien.

Wirtschaftlich gesehen ist eine Steigerung des Wertanteiles von Software an Produkten zu beobachten, diese werden immer stärker durchdrungen [Grünb98]. Als Beispiel die Zunahme des Koordinationsaufwandes, der mit Software bekämpft wird [Sauerbrey89]:

- *Zunahme der Produktkomplexität*: Primär bedingt durch die Marktsituation.
- *Zunahme der Fertigungstiefe eines Produktes*: Steigt proportional zu Produktkomplexität an.

- *Zunahme des Engineering-Anteils am Produkt*: Das Zusammenspiel von mehr Teilen erfordert engere Toleranzen und damit eine höhere Planungsqualität. Die Innovationszyklen sinken stärker als die produzierten Mengen zunehmen [Sauerbrey89].
- *Zunahme der betrieblichen Koordination zwischen Betrieben*: Durch die Komplexität von Produkten (ein einzelnes Unternehmen kann nur selten die gesamten Teile herstellen) steigt der externe Koordinationsaufwand an.

1.3.4 Probleme bei Software Projekten

Es gibt zahllose Untersuchungen und Berichte über vollständig oder teilweise fehlgeschlagene Software-Entwicklungs- und Einführungsprojekte. Tatsächlich ist die Anzahl und das Spektrum von Problemen groß.

Laut dem Change Integration Team der Unternehmensberatung Price Waterhouse [PriceW98] sind 75% der finanziellen Aufwendungen für die Entwicklung und den Einsatz von Anwendungssoftware mehr oder weniger vergeblich: Die Anwendungen erreichen keine zufriedenstellende Funktionalität, die Projektziele (Zeitplan und Kosten) können nicht eingehalten werden.

Als Hauptgründe für das Scheitern dieser Projekte werden vier Gründe genannt, frei nach Price Waterhouse:

- *Übernahme der Software-Altlasten*: Die Integration bestehender Systeme (Hard- und Software) und Datenbestände erweist sich in der Regel als zeitaufwendiger als angenommen. Das Hauptproblem sind dabei meist die Schnittstellen und die unzureichende Dokumentation.
- *Unzureichende Anforderungsanalyse*: Anforderungen wachsen oft während des Projektverlaufs an und können mit den gegebenen Zeit- und „Human-Ressourcen“ nicht mehr bewältigt werden. Oder die Projektziele sind von vornherein unrealistisch. Oft wird bei der Projektierung auf geringe Kosten und nicht auf maximalen Nutzen geachtet.
- *Planungsmängel*: Mangelnde Flexibilität bei der Zeitplanung, meist aufgrund von Druck seitens des Managements. Zusätzliche Anforderungen werden im Zeitplan nicht berücksichtigt.

- *Fehleinschätzung der „weichen" Faktoren*: Die Gestaltung der Benutzerschnittstellen und die Einführung der Software, beziehungsweise der Aufwand zur Einschulung des Personals wird unterschätzt.

Besonders in der Entwicklung großer Systeme, also Unternehmenssoftware, ist die Anzahl der Mißerfolge größer als vor einigen Jahren. Es sind im Gegensatz zur vorherrschenden Meinung nicht technische Probleme, sondern kommunikative Schwierigkeiten und strategische Fehleinschätzungen. Dazu Price Waterhouse [PriceW98]: „*Unrealistische Projekterwartungen entstehen häufig aus dem direkten Widerspruch zwischen den Zielen der IT-Fachleute und den Forderungen ihrer Vorgesetzten...*"

Beispiel [CompW99]: Die beiden größten amerikanischen Abfallentsorgungsunternehmen (Allied Waste Inc.und Waste Managemen Inc) schalten beide ihre SAP-Lösungen ab.

- Allied Waste Inc. übernimmt einen Konkurrenten. Das Unternehmen erreicht damit einen Jahresumsatz von 6 Milliarden US-Dollar. Das SAP R/3 System bei Allied Waste Inc. hat seit 1995 mehr als 130 Millionen US-Dollar gekostet. Das Stoppen des Projektes ist aber trotzdem billiger meint die Geschäftsleitung, denn: „*SAP erwartet, daß man sein Geschäft so organisiert wie deren Software arbeitet – und nicht umgekehrt*". Für die neue Geschäftssituation sei R/3 zu unflexibel.
- Waste Management hat bereits 45 von geplanten 250 Millionen US-Dollar in SAP investiert, auch dieses Projekt wurde gestoppt.

Ein bekanntes und ausführlich dokumentiertes Beispiel für das Scheitern einer Softwareeinführung, die zum Ruin des Unternehmens führte, ist die Causa Fox-Meyer [FoxMeyer95].

Zusätzliche Probleme entstehen durch die schwierige Kontrolle von Produktivität und Qualität während der Entwicklung des immateriellen Produktes Software.

1.4 Lösungsidee und Vorgehensmodell

Das zur Lösung verwendete Business Engineering Labor Konzept ist eine Weiterentwicklung des Business Reengineering Labor Konzeptes das von Langer und Wolff in [Langer94] beschrieben wurde. Zuerst also eine Beschreibung des Ausgangspunktes.

1.4.1 Das Business Reengineering-Labor Konzept

Langer und Wolff beschreiben das Business Reengineering-Labor Konzept zur Realisierung von entwickelten Prozeßmodellen als einen Weg, Ablauforganisation und unterstützende Software quasi simultan zu entwickeln. Dazu ein Auszug aus [Langer94]:

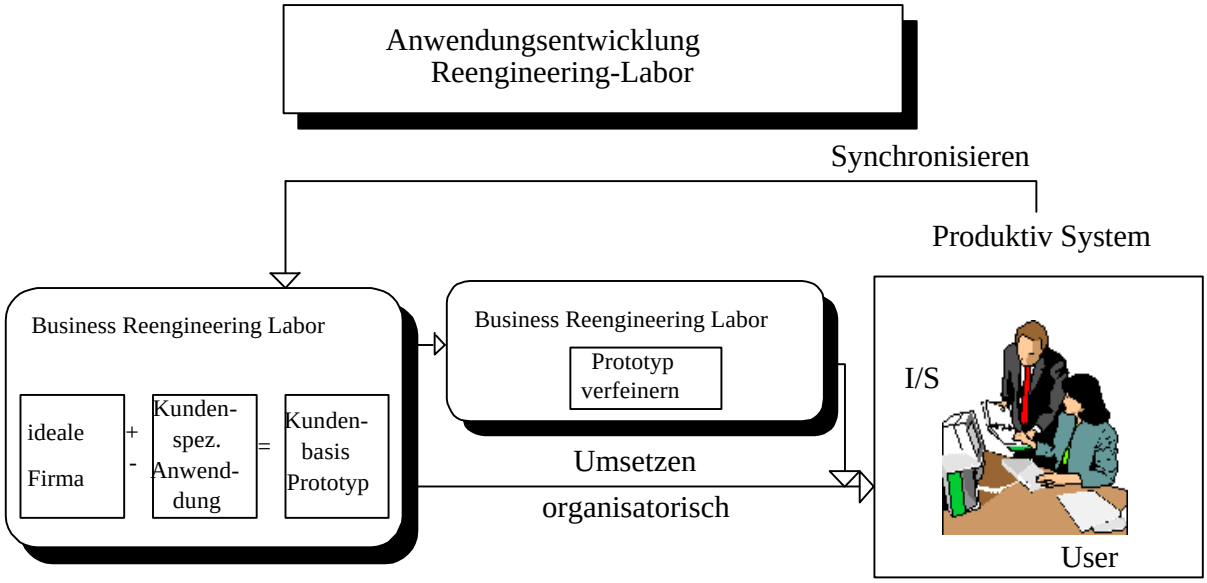

Abb. 2 Das Business Reengineering Labor-Konzept [Langer94]

> *In sogenannten Business Reengineering Labors wird eine Systemumgebung bereitgestellt, in der Geschäftsabläufe auf Basis von Standardsoftware („ideale Firma“) unter Berücksichtigung der unternehmensspezifischen Prozeßmodelle prototypisch entwikkelt werden können. Der Endanwender erhält in diesen Labors die Möglichkeit, nicht nur einzelne Systemfunktionen als Prototypen kennenzulernen, sondern einen Prototypen des neuen (Informations-) Systems inklusive der gesamten prozeßbezogenen Ablauforganisation kennenzulernen. Während die Anpassung der Strukturorganisation bereits begleitend erfolgt, werden die Informationssysteme im Business Reengineering Labor unter Benutzerbeteiligung prozeßorientiert entwickelt und sukzessive zum fertigen System ausgestaltet. Prozeßoptimierung, Entwicklung und Umsetzung der Strukturorganisation sowie die Entwicklung der Informationssysteme werden so in einem evolutionären Prozeß zusammengefaßt.*

Im Zentrum des BR-Labor Konzeptes stehen also eine Entwicklungsumgebung und ein Unternehmensmodellierungswerkzeug.

1.4.2 Das Business Engineering-Labor Konzept

Der Einsatz von Methodologien in Analyse und Entwurf ist meist „zumindest“ mangelhaft [Grünb98]. Diese Defizite der SW-Entwicklung können aber gerade durch die Verwendung eines Unternehmensmodellierungswerkzeuges behoben werden.

Ein Überblick über das Business Engineering Labor-Konzept wie es in Grundzügen in den Fallstudien erprobt wurde, kann am besten mit Hilfe der bekannten W-Fragen gewonnen werden.

- *Was?* (siehe 1.4.2.1): Gestaltung von Geschäftsprozessen und Unternehmenssoftware
- *Warum?* (siehe 1.4.2.1): Verbesserung durch gemeinsame Entwicklung
- *Wer?* (siehe 1.4.2.2): Betriebswirt und Informatiker
- *Womit?* (siehe 1.4.2.3): Modellierungswerkzeug & SW-Entwicklungstools
- *Wie?* (siehe 1.4.2.4): Ein Meta-Vorgehensmodell

Entsprechend dem Titel dieser Diplomarbeit ist bei den folgenden Abschnitten auf zwei Punkte besonderes Augenmerk zu legen: Die Komponente der *gemeinsamen Gestaltung* von Geschäftsprozessen und Unternehmenssoftware und die Unterstützung durch Werkzeuge und Methoden der Informatik.

1.4.2.1 Was und Warum?

In der Praxis ist man von einer zufriedenstellenden Unterstützung der Geschäftsprozesse durch Software meist weit entfernt. Die Abläufe entsprechen oft nicht den Anforderungen, sie sind zu wenig flexibel, zu wenig kundenorientiert und weisen Mängel bezüglich der Flexibilität auf.

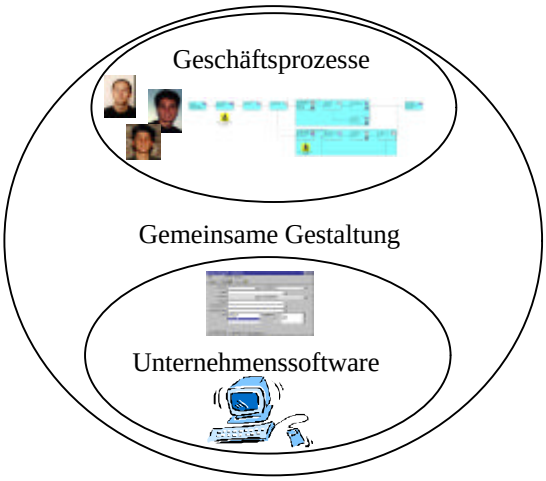

Abb. 3 Gemeinsame Gestaltung von Unternehmenssoftware und Geschäftsprozessen

Um diese Kluft zu schließen, wird nun durch einen gemeinsamen simultanen Gestaltungs- und Entwicklungsprozeß versucht, die Abstimmung zwischen den technischen und betriebswirtschaftlichen Konzepten unter Berücksichtigung der gegenseitigen Abhängigkeiten zu erreichen.

Zur Erreichung dieses Ziels ist ein Werkzeug notwendig, das in der Lage ist, sowohl die technischen (Datenmodell, Informationen über Schnittstellen, IT-Infrastruktur), als auch die betriebswirtschaftlichen Aspekte (Organigramm, Geschäftsprozesse) auf einheitliche Weise zu verbinden und darzustellen.

1.4.2.2 Wer?

Getragen soll dieser gemeinsame Gestaltungsprozeß von einem Team werden. Dieses *BE-* (*Business Engineering-*) *Team* sollte aus Personen bestehen, die über fundierte betriebswirtschaftliche und technische Grundlagen verfügen. Zusätzlich sind betroffene Personen notwendig, die in diesem Gestaltungsprozeß beteiligt werden sollen; sie verfügen über detailliertes Wissen über das Unternehmen, das ein Business Engineering Projekt durchführen will.

Als Schnittstelle zwischen den genannten Personen und Konzepten dient ein (Unternehmens-) Modellierungswerkzeug.

Bei den Fallstudien, die im Rahmen dieser Diplomarbeit durchgeführt wurden, fand eine Kooperationen zwischen einem Diplomanden der Betriebswirtschaft und einem Diplomanden der Informatik statt.

1.4.2.3 Womit?

Um diesen Prozeß, der stark durch Kommunikation geprägt ist, zu unterstützen, ist der Einsatz eines Werkzeuges unerläßlich. Das ist einerseits in der Komplexität der zu bearbeitenden Aufgaben und Daten begründet. Andererseits in der Notwendigkeit, unterschiedliche Standpunkte möglichst einheitlich zu beschreiben, um verschiedenen Personen Zugang zu verschaffen. Ein Softwarewerkzeug ist für die Unterstützung des oben genannten Prozesses geeignet, wenn:

- Alle *Daten*, die das Unternehmen beschreiben, abgespeichert werden können. Man benötigt also ein Werkzeug, das in der Lage ist, Informationen über
  - *Aufgaben* und *Prozesse*,
  - *Informations-* und *Datenmodelle*,
  - die *Unternehmensumwelt*,
  - *Organisationsstrukuren* (die Ordnung und Beziehungen zwischen Organisationseinheiten) und
  - *Sachmittel*

  abzubilden.
- Bei der Darstellung der Daten eine gewisse *Abbildungstreue* erreicht werden kann.
- Die oben genannten Daten auf ansprechende und allgemein verständliche Weise *dargestellt* werden können.
- Das gezielte *extrahieren* von Daten aus dem Modell möglich ist.

Dieses Werkzeug steht also während des gesamten Entwicklungs- und Gestaltungsprozesses im Mittelpunkt. Bei den Fallstudien, die im Rahmen dieser Diplomarbeit durchgeführt wurden, wurde das Unternehmensmodellierungswerkzeug AENEIS verwendet.

1.4.2.4 Wie?

Die folgende Darstellung stellt eine Erweiterung des klassischen Modells des wissenschaftlichen Problem-Lösungsprozesses dar. Im Detail wurden folgende Erweiterungen vorgenommen:

- Die Aufspaltung der Lösung in ein Modell des Soll-Zustandes des Unternehmens und in die neu erstellte Unternehmenssoftware.
- Einführung von *Zyklen*:
  - Der *Zyklus der Weiterentwicklung*. Zwischen der Lösung (organisatorisch und technisch) und der Umsetzung im Unternehmen entstehen Wechselwirkungen.
  - Der Zyklus zwischen dem Modell des *Soll-Zustandes* und der neuen Unternehmenssoftware.

Zur Erklärung von Abb. 4 (Legende):

- *Phasen* werden kursiv dargestellt.
- **Zustände** werden fett dargestellt.

Die Begriffe „Realität“ (Rechteckige Zustände) und „Abstrakte Ebene“ (Ovale Zustände) beschreiben die zwei primären Abstraktionsebenen des Modells. Die folgende Darstellung wurde gewählt, weil sie vertraut ist und relativ neutral bezüglich der Anwendung in einem bekannten Vorgehensmodell zur SW-Entwicklung (Phasenmodell, Spiralmodell, ...). Man kann dieses Vorgehensmodell als eine Art Meta-Vorgehensmodell betrachten.

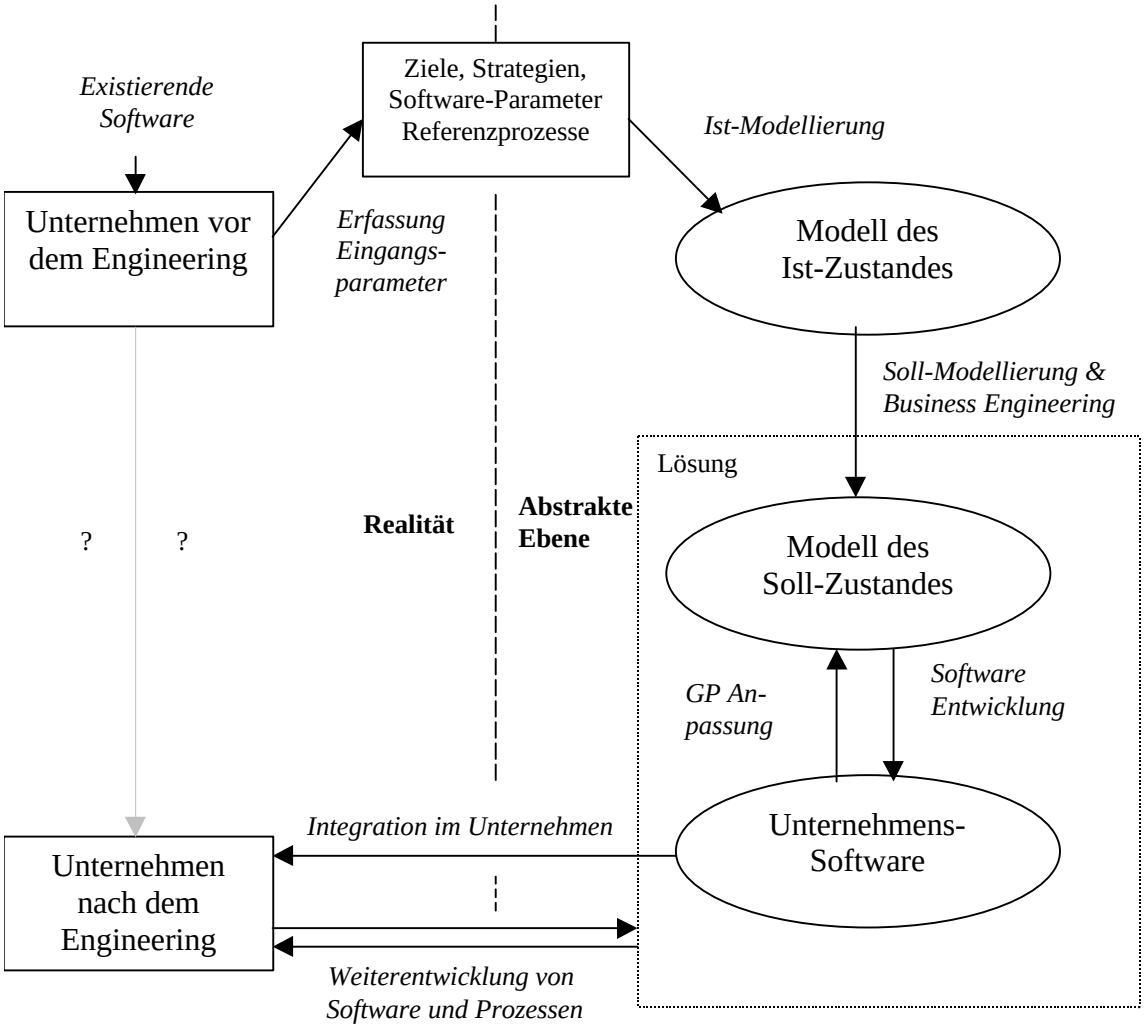

Abb. 4 Vorgehensmodell

Die Zustände *Unternehmen vor dem Engineering* und *Unternehmen nach dem Engineering*, also Ist- und Soll-Zustand, sind in diesem Fall als Zustände der jeweiligen Organisationsstruktur, der Unternehmenssoftware und der Unternehmenskultur zu verstehen. In der Praxis wird meist bereits bei der ersten Modellierung (Modell des Ist-Zustandes) eine gewisse Optimierung (Fokussierung) vorgenommen, die über die natürlichen „Verluste“ bei Modellierungs- und abstrakten Abbildungsvorgängen hinausgehen. Eine möglichst naturgetreue Abbildung, zumindest eine einheitliche Festlegung von Abstraktionen und dem gewählten Grad der Detailliertheit, ist dann vorteilhaft, wenn eine vergleichende Messung zwischen ursprünglichen und neuen Prozessen erfolgen soll (Controlling des Projektes, Benchmarking).

Eine weitere und genauere Beschreibung der einzelnen Phasen (1.4.2.4.1 bis 1.4.2.4.6) folgt, diese Phasen stellen den „Umweg“ dar, zwischen Ist- und Soll-Zustand des Unternehemens.

1.4.2.4.1 Erfassung der Eingangsparameter

Drei Klassen von Eingangsparametern können zunächst erfaßt werden:

- Die *Ziele* werden identifiziert, im Modell abgelegt und eventuell operationalisiert.
- Spezielle *Parameter der Software* werden im Modell abgebildet (Objekte, Eingabemasken, Relationen,...)
- *Referenz*(*prozeß*)*modelle* werden bereitgestellt. Optional, hängt stark von der Art der gewünschten Geschäftsprozesse ab.

Problem: Zum großen Teil sind Arbeitsabläufe im Bewußtsein der Mitarbeiter verankert. Erschöpfende Ablaufbeschreibungen sind selten, da sich die Abläufe im Unternehmen häufig ändern.

Das Werkzeug muß in diesem Schritt die Möglichkeit bieten, Ziele sinnvoll zu strukturieren und abzuspeichern. Klassisch ist dabei die Aufbereitung und Hierarchisierung von Zielen als Zielbaum. Die Darstellung von Geschäftsprozessen ist die Kernkompetenz eines Unternehmensmodellierungswerkzeuges, zusätzlich müssen aber auch Charakteristika der Software sinnvoll integrierbar sein, wie zum Beispiel Beschreibungen von Applikationen, Benutzer-Schnittstellen, usw.

1.4.2.4.2 Ist-Modellierung

Eingabedaten sind:

- *Interviews* mit den Mitarbeitern: Protokollierte Gespräche mit verschiedenen Mitarbeitern, die relevante Aufgabengebiete bearbeiten.
- *Beobachtungen* können mit Hilfe von Videokameras durchgeführt werden, grundsätzlich ist zu beachten, daß sich Arbeitsvorgänge durch, dem Ausführenden bewußte, Beobachtung verändern [REFA84].
- *Dokumentenanalyse* und besonders Fragebögen sind meist nur in Kombination mit anderen Erhebungstechniken sinnvoll.

Ausgabe dieses Arbeitsschrittes ist ein Modell des Unternehmens, das die augenblickliche Struktur und die relevanten Abläufe im Unternehmen dermaßen genau beschreibt, daß eine Analyse möglich ist. Dieses Modell besteht (grob beschrieben) aus verschiedenen Diagrammen und deren Dokumentation.

Das BE-Team wird in diesem Schritt verschiedene, an den jeweiligen Arbeitsabläufen beteiligte, Personen befragen. Die Transformation in diesem Schritt ist die Abbildung der Realität auf ein abstraktes Modell – der Vorgang der im allgemeinen als Unternehmensmodellierung bezeichnet wird. Die Sicht des BE-Teams vom Ist-Zustand wird also im Modell abgebildet.

In der Praxis wird manchmal mit einer Soll-Modellierung begonnen, um eine Beeinflussung durch das „Ist“ und damit Fehlerfortpflanzung bei der Gestaltung zu vermeiden. Das Feststellen des Status quo ist aber dienlich, wenn ein Projektcontrolling oder Assessment durchgeführt werden soll [Grünb98].

1.4.2.4.3 Soll-Modellierung und Business Engineering

Der Kern des Gestaltungsprozesses: Die Eingabedaten ergeben sich aus den bisherigen Schritten. Außerdem kann in dieser Phase bereits eine, noch anzupassende, Softwarelösung bereits ausgewählt sein. Dabei muß bemerkt werden, daß die Auswahl eines Softwareproduktes auch nach diesem Schritt möglich ist, da die Anforderungen damit festgestellt werden.

Aus diesen Parametern wird nun, unter Berücksichtigung gegenseitiger Wechselwirkungen, das neue Unternehmensmodell (Geschäftsprozesse) mit integrierter Softwareunterstützung modelliert. Wenn noch kein SW-Produkt ausgewählt wurde, muß zumindest die technische Machbarkeit, in Verbindung mit den aktuellen Gegebenheiten, berücksichtigt werden.

Die Ausgabe der Transformation der gesammelten Daten ist ein neu gestaltetes Unternehmensmodell, ein Modell des Soll-Zustandes. Die verbesserte (vorwiegend prozeßorientierte) Organisationsstruktur, Prozeßbeschreibungen und die Anforderungen an die Software zur Unterstützung der Prozesse sind im Modellierungs-Werkzeug abgebildet.

Typischerweise ergeben sich daraus neben einer Spezifikation der Software und einem Pflichtenheft auch Ablauf- und Stellenbeschreibungen. Beteiligte Personen sind hauptsächlich die Mitglieder des BE-Teams. Speziell in diesem Schritt ist fundiertes Fachwissen aus Betriebswirtschaft und Informationsverarbeitung notwendig.

1.4.2.4.4 Software Entwicklung und Verfeinerung

Die Entwicklung der Software kann nun beginnen. Eingabedaten sind die erschöpfenden Beschreibungen der Anforderungen an die neue Software (Pflichtenheft) und die gewählte Entwicklungsumgebung. Neben diesem Anforderungsprofil müssen auch bereits vorhandene und in das neue System zu integrierende Soft- und Hardwarekomponenten dokumentiert sein. Beispielsweise spezielle Peripheriegeräte, Schnittstellen zu anderen Anwendungen und bereits vorhandene Datenbanksysteme.

Ausgabe ist die erstellte (entweder entwickelte oder angepaßte) Software mit ausreichender Dokumentation. Über textuell tradiertes Wissen hinaus (Handbücher) sollten auch Schulungsmaßnahmen eingeplant werden. Die Testphase (Probebetrieb vor der Integration) ist hier ebenfalls Teil der Softwareentwicklung, die Software am Ende dieser Phase sollte jedenfalls einen befriedigenden Reifegrad besitzen.

Beteiligte Personen sind die Softwareentwickler und das BE-Team. In Hinblick auf die fortlaufende Entwicklung des Systems und auf die Schwierigkeiten bei der Einführung komplexer Systeme erscheint es wünschenswert, daß bei der Gestaltung der Software schon sehr früh mit den potentiellen Benutzern gesprochen und getestet wird (Prototyping). Dadurch kann man mit größter Wahrscheinlichkeit auch die Reibungsverluste bei der Einführung verkleinern.

Die Transformation ist damit als typisches Softwareengineering unter besonderer Berücksichtigung der notwendigen Weiterentwicklung und der Einbindung der künftigen Benutzer zu sehen.

Besonders wichtig ist Wachsamkeit bezüglich der *Konsistenz* zwischen der Gestaltung der Software und dem Modell: Änderung in der Softwareentwicklung haben stets im Modell geprüft und nachgeführt zu werden. Diese Überwachung sollte vom BE-Team durchgeführt werden.

1.4.2.4.5 Integration

Dieser Schritt kann von einer teilweisen Integration bis zu einer völligen Substitution der bestehenden Unternehmenssoftware reichen. Um das Risiko zu minimieren, wird die neue Software in der Praxis zumindest teilweise parallel zur bestehenden eingesetzt um eine gewis-

se Ausfallsicherheit zu gewährleisten um den Übergang fließender zu gestalten. Naturgemäß tritt eine stärkere Belastung der Mitarbeiter während dieser Phase auf.

Unterstützende Werkzeuge gibt es in dieser Phase nur wenige. Wichtig ist Kommunikationsunterstützung bei der Schulung und Problembehebung und Dokumentation von Mängeln. Speziell zur Kommunikationsunterstützung, beispielsweise der Beschreibung von Tätigkeiten innerhalb eines Prozesses, kann das Modellierungswerkzeug verwendet werden.

1.4.2.4.6 Weiterentwicklung

Der projektartig organisiert Teil des Gestaltungsprozesses ist nun zu Ende: Wartung und Weiterentwicklung und idealerweise eine fortlaufende Evaluierung (Prozeßmessung) müssen aus oben genannten Gründen jedoch weitergeführt werden. Eine Institutionalisierung der prozeßorientierten Arbeitsabläufe und der Weiterentwicklung muß angestrebt werden. Nach Brenner [Brenner95] werden nach der Implementierung ungefähr 70 Prozent des Leistungspotential erreicht. Weiterentwicklung kann zur vollen Ausschöpfung des Potentials führen, keine aktive Weiterentwicklung senkt die Leistung.

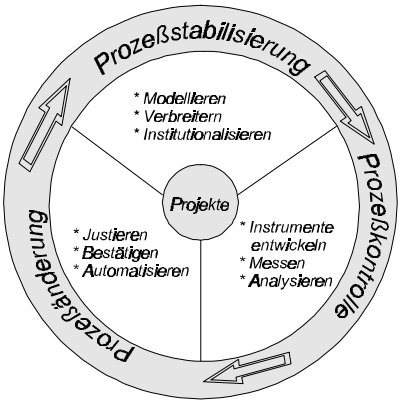

Abb. 5 Kontinuierliche Prozeßverbesserung [Dion93]

Dabei werden die Phasen Prozeßänderung, Prozeßstabilisierung und Prozeßkontrolle durchlaufen. Langer bemerkt dazu: *Die Verbesserung der Softwareentwicklung (und damit des betrachteten Hauptprozesses) ist hier quasi als eigener Unternehmensprozeß identifiziert worden und entsprechend umgesetzt* [Langer94].

Beispielszenario: Die Weiterentwicklung der Geschäftsprozesse liegt primär in den Händen der beteiligten Mitarbeiter des Unternehmens, die Wartung des Softwaresystems in den Hän-

den des Softwareanbieters oder speziell geschulter Mitarbeiter. Auch die Weiterentwicklung der Software sollte bis zu einem gewissen Grad durch interne Personen (EDV-Beauftragter, oder Abteilung) möglich sein.

1.5 Zusammenfassung des Kapitels

Am Beginn des Kapitels wurde die Aufgabenstellung und wesentliche damit verbundene Begriffe, wie Business Engineering, Standard- und Individualsoftware geklärt. Auch Gründe für ein Business Engineering Projekt wurden genannt.

Dann wurde die Lösungsidee zur gemeinsamen Gestaltung und Entwicklung von Unternehmenssoftware und Geschäftsprozessen im Rahmen eines Vorgehensmodelles vorgestellt, dessen zentrale Idee die Zusammenführung von betriebswirtschaftlichem und technischem Wissen mit Hilfe eines Unternehmensmodellierungswerkzeuges ist.

2 Grundlagen

2.1 Ziele des Kapitels

Einerseits sollen wichtige Aspekte der Aufgabenstellung gezeigt werden, andererseits soll eine Aufbereitung verschiedenartiger wissenschaftlicher und technischer Grundlagen zur Analyse erfolgen.

Durch den interdisziplinären Charakter der Arbeit wird besonderer Wert auf die Definitionen von verwendeten Begriffen gelegt, da ein Zugang für Leser mit betriebswirtschaftlichem und technischem Hintergrund ermöglicht werden soll. Das ist besonders deshalb notwendig, weil manche Begriffe im Bewußtsein verschiedener Personen durchaus unterschiedliche Bedeutungen haben können.

Beispiel: Der Begriff Informationssystem steht für Langer [Langer94] synonym zu Anwendungsprogramme. Ein Informatiker assoziiert den Begriff primär mit Datenbanken und Datenbankmanagementsystemen (DBMS). Für Heinrich [Heinrich97] ist ein Informationssystem ein Mensch/Aufgabe/Technik-System zum Beschaffen, Herstellen, Bevorraten und Verwenden von Informationen.

2.2 Modellierung

2.2.1 Allgemeines

Definition 6: Modellbildung [Traunmüller]:

> *Modellbildung ist die Abbildung eines Systems auf ein anderes System, von dem anzunehmen ist, daß es einfacher zu verstehen oder leichter zu handhaben ist.*

Grundsätzliche Techniken sind *Abstraktion* und *Strukturierung*. Das Durchführen von Abstraktionen ist zur Verminderung der abzubildenden Datenmenge notwendig, es bewirkt einen bewußten Verzicht auf Genauigkeit, aber auch eine Erhöhung der Übersichtlichkeit. Strukturelle Gleichheit oder starke Ähnlichkeit ist eine Voraussetzung. Auf abstrakter Ebene wird der wissenschaftliche Problemlösungsprozeß mit Hilfe eines Modells in der folgenden Abbildung dargestellt, die Darstellung ist selbsterklärend:

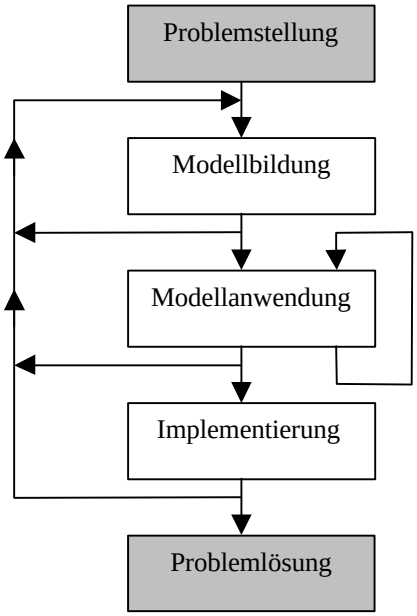

Abb. 6 Phasen der Lösung eines Problems [Chroust97]

Modelle sollen also die Realität repräsentieren, die Auswahl der geeigneten Modellierungstechnik (Darstellungsform) und des verwendeten Modellierungswerkzeuges hat daher eine wesentliche Bedeutung.

2.2.2 Einsatz von Modellen

Die beiden grundsätzlichen Ziele der Modellbildung sind:

- *Analyse*: Erklärung von Sachverhalten, Beziehungen und der Wirkung von Parametern in einem System.
- *Synthese*: Entwicklung, Erweiterung und Veränderung eines Systems.

Bei manchen Arten von computerunterstützten Modellen wird die Analyse mit Hilfe von Simulationsmechanismen der Modellierungsumgebung erreicht. Das dient dem Formulieren und Verifizieren von Vorhersagen und im weiteren Sinne der Validierung der Lösung. Weiters einige Aussagen über Modelle, die später in dieser Arbeit interessant sind:

- Bewältigung von *Komplexität*: Ein Modell ist eine komplexe Systembeschreibung, unterstützt aber das Durchführen von Abstraktionen und dient daher zur Bewältigung der Komplexität des Systems.

- Die *Beschreibung* des Systems als Modell kann als Spezifikation und Dokumentation des realen System verwendet werden.
- *Visualisierung*: Durch den Einsatz *grafischer Gestaltungselemente*, wie zum Beispiel Graphen, Symbolen und verschiedenen Diagrammen, kann man in der Realität schwer durchschaubare Sachverhalte leichter zugänglich machen. Es gibt aber auch Beschreibungsformen für Modelle, die auf Visualisierung verzichten. Beispiel: Beschreibung eines Modells durch Differentialgleichungen.

Man kann Modelle auch nach der *Zustandskontinuität* klassifizieren:

- *Statisches Modell*: Beschreibung von Strukturen oder Kausalitäten, zum Beispiel eine Landkarte oder ein Datenmodell.
- *Dynamisches Modell*: Beschreibung von Abläufen und Wechselwirkungen. Beispiele: Verkehrssimulation, Fertigungsprozeß, ...

2.2.3 Modelltypen

Sowohl im wissenschaftlichen als auch kommerziellen Bereich existiert ein Vielzahl von Modellierungstechniken, die in einer Vielzahl von Darstellungsformen angewandt werden. Die Auswahl, dieser komplexen Parameter der Modellierung, hängt stark vom Zweck der Modellierung und von der gewünschten Sichtweise ab. Dazu seien einige Beispiele für statische und dynamische Modelle genannt.

Spezielle Methoden zur Datenmodellierung (statische Modelle):

- *ER – (Entity Relationship) Modelle*: Die aktuell bekannteste und weitverbreitetste Art der Datenmodellierung, dabei werden die Beziehungen zwischen (Daten-) Einheiten grafisch dargestellt. *Einheiten* (entities) haben Attribute und werden über *Beziehungen* (relations), welche ebenfalls Attribute haben können, miteinander verbunden.
- *Einheitenmodellierung*: Bei der Einheitenmodellierung werden auch Beziehungen als Objekte angesehen. Der Verlauf der Modellierung ist mit dem der Modellierung bei ER-Diagrammen ident [Hawryszkiewycz95].

- *Binäre Beziehungsmodellierung*: Eine Erweiterung der oben beschriebenen Verfahren. Dabei wird die Modellierung vereinheitlicht indem man für die Assoziationen nicht genau ein Attribut verwendet, sondern Attributmengen.
- *Relationenmodelle*: Tabellarische Darstellung, die meist auf obigen Diagrammen aufbaut. Dabei werden konkrete Tabellen dargestellt, die Relationen beschreiben [Dittrich97].

Die bisher genannten Methoden dienen konkret der Gestaltung von *Datenbank-Managementsystemen* (DBMS). Die folgenden Methoden dienen zur Modellierung von Prozessen (dynamische Modelle):

- *Einheitenzyklus Diagramm*: Ein solches Diagramm stellt die Übergänge zwischen den Zuständen von Daten aus ER-Modellen, Einheitenmodellen, usw. dar. Sozusagen eine dynamische Darstellung der Übergänge zwischen statischen (Daten-) Darstellungskonzepten.
- *Petri-Netze*: Vielseitig verwendbare Methode für graphische Modellierung von Prozessen und Ablaufstrukturen [Reisig86]. Wurde auf vielfältige Arten weiterentwickelt und adaptiert.
- *Strukturierte Humansprache*: Verbale Beschreibung von Abläufen, die durch die Verwendung von speziellen Konstrukten starke Ähnlichkeit zu Programmstrukturen haben. Wird zum Beispiel bei der Methode der *schrittweisen Verfeinerung* in der Softwareentwicklung verwendet – dabei wird die humansprachliche Beschreibung einer Lösung bis zu einer, auf einem Rechner, ablauffähigen Version verfeinert. Diese Art der Beschreibung wird später in sehr ähnlicher Form verwendet, daher als Beispiel die Beschreibung der Bearbeitung eines *Kundenauftrages*:

  Kundenauftrag geht ein.

  WENN *der Kunde ein Neukunde ist*

  DANN *Kundendaten erfassen; Auftrag bearbeiten*

  SONST *Auftrag bearbeiten*

- Bekannte Weiterentwicklungen (auch von Petrinetzen) sind Vorgangsketten-Diagramme sowie EPK und eEPK – (Erweiterte) Ereignis-Prozeßketten-Diagramme.

Übergreifende Konzepte sind prinzipiell stark an objektorientierten Konzepten orientiert:

- *OMT – Object Modelling Technique* [Rumbaugh91]: Drei Modellebenen: *Objektmodell* (Modellierung von kommunizierenden Objekten), *dynamisches Modell* (Dynamik der Objekte durch Zustandsautomaten; Use Cases!) und *funktionales Modell* (Datenflußdiagramme).
- *Unified Method* [Jacobson92]: Beim sogenannten *Use Case driven design* werden fünf sequentiell zu bearbeitende Modellebenen unterschieden: *Anforderungsmodell* (Problembeschreibung mit Use Cases), *Analysemodell* (Objektstrukturen auf Klassenebene, wobei in Schnittstellen-, Informations- und Steuerungsobjekte unterschieden wird), *Entwurfsmodell* (Festlegung der Dynamik mit Interaktions- und Zustandsdiagrammen), *Implementierungsmodell* (die Implementierung an sich) und ein *Testmodell* (Testspezifikation und Testergebnisse).
- *UML – Unified Modelling Language* [OMG99]: Eine Weiterentwicklung und Vereinigung von OMT, der Methode von Booch und OOSE (Object Oriented System Engineering). UML befindet sich derzeit noch in der Entwicklung, wird aber zum Teil schon eingesetzt. Ziel von UML ist das Ermöglichen von Spezifikation, Visualisierung, Konstruktion und Dokumentation eines Softwaresystems und der Geschäftsprozeßmodellierung.

Aus erkennbaren Gründen (ganzheitliche Sicht) sieht es zur Zeit so aus, als ob sich UML als Standard etablieren könnte.

2.3 Softwaretechnik

Softwaretechnik (engl. Software Engineering) befaßt sich mit der professionellen Entwicklung großer Softwaresysteme, wobei Anwendungssoftware im Vordergrund steht.

2.3.1 Vorgehensmodelle in der Softwareentwicklung

Definition 7: Vorgehensmodelle

Ein Vorgehensmodell dient zur Benennung und Ordnung von produktbezogenen Aktivitäten bei der Softwareentwicklung.

Die Verwendung eines Vorgehensmodelles bedeutet eine Trennung zwischen dem „Was?“ (dem Produkt: Anwendungssoftware) und dem „Wie?“ (dem Herstellungsprozeß beschrieben

durch das Vorgehensmodell). Man beachte dabei die Frage „Warum?“, die in 1.4 behandelt wird.

Grundsätzlich kann man unterscheiden in *Phasenmodelle*, *Spiralmodelle* und *zyklische Modelle*. Ihre Bedeutung erhalten sie durch das Ermöglichen grundsätzlicher Machbarkeit und aus der Notwendigkeit zur Planung und Steuerung des Prozesses zur Herstellung eines komplexen Produktes (siehe 1.3.3).

- Das *Phasenmodell* oder *Wasserfallmodell* kann prinzipiell als das grundlegende Vorgehensmodell (VM) zur Softwareherstellung bezeichnet werden. Die Softwareentwicklung wird dabei in Phasen zerlegt, wobei jede Phase deklarierte und definierte Entwicklungsschritte und Ergebnisse hat [Boehm76]. Dabei wird das System jeweils in seiner ganzen Breite bearbeitet. Es existiert in zahlreichen weiterentwickelten Variationen. Ein Beispiel für eine Phasengliederung ist: *Analyse und Definition*, *Entwurf*, *Implementation*, *Test*, *Einsatz und Wartung*.
- *Spiralmodelle* sind flexibler als Phasenmodelle, es werden zwar Phasen durchlaufen, aber nicht linear sondern zyklisch. Es wird jeweils ein Ausschnitt des Systems betrachtet, wobei mit kritischen Bereichen begonnen wird. Dabei wird ein Lernprozeß berücksichtigt, der den Herstellungsprozeß betrifft. Weitere Kernpunkte sind die Berücksichtigung von Prototypingschritten (siehe 2.3.2.1) und die Integration einer Risikoanalyse der jeweiligen Phasen [Boehm88].
- Bei z*yklischen Modellen* wird Software nicht als Produkt, sondern als eine Folge von Versionen verstanden, die jeweils zyklisch, auf die vorige Version aufbauend erstellt werden [Floyd97].

Wichtig bei der Betrachtung von Vorgehensmodellen für Softwareentwicklungsprojekte sind weiters die *Rollen der Beteiligten* (Auftraggeber, SW-Hersteller oder Anbieter, Mitarbeiter, Entwickler,...), die Art der *Organisation des Projektes*, die Sicherung der *Prozeßqualität* (zum Beispiel durch *ISO 9000 ff.* [DIN] oder durch das *CMM-Modell* [Paulk93], das von etlichen großen Firmen angewandt wird. Z.B.: Siemens)

Verbunden mit dem Produkt sind die verwendeten *Programmierumgebungen,* die *Produkt-* und *Konfigurationsverwaltung* und die Produkt-*Qualitätssicherung* zu beachten.

2.3.1.1 Beispiel: Einführungsmodell BAAN Target

Ein Beispiel für ein Einführungs-Phasenmodell ist das Phasenkonzept BAAN Target zur Einführung von BAAN IV [BAAN-2]. Drei Phasen auf der obersten Ebene werden jeweils in weitere drei Phasen zerlegt, die durch Meilensteine abgeschlossen werden. Meilensteine sind allgemein als Betrachtungszeitpunkte zur Durchführung einer Validierung und Verifikation zu verstehen.

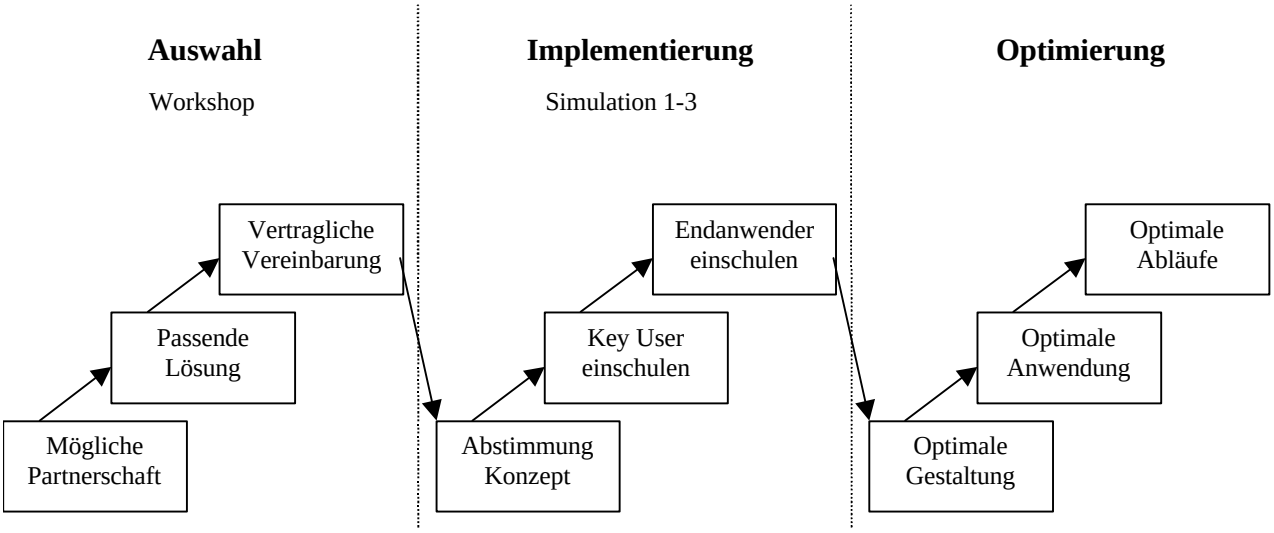

Abb. 7 BAAN Target, Beispiel für ein Phasenkonzept

Die drei top-level Phasen (Auswahl, Implementierung und Optimierung) werden als Projekte betrachtet.

- *Auswahl (Phase 1)*: Die Verkaufsphase ist in der weiteren Betrachtung im Zusammenhang der vorliegenden Diplomarbeit nicht interessant. In dieser Phase wird eine grundsätzliche Analyse der Geschäftsprozesse und Anforderungen vorgenommen.
- *Implementierung (Phase 2)*:
  - *Abstimmung des Konzeptes* bedeutet, daß die Geschäftsprozesse anhand der vorgegebenen Möglichkeiten des BAAN Systems durchgespielt werden. Dann wird entschieden ob sich die Geschäftsprozesse der Firma an das System anpassen, oder ob der möglicherweise teurere Weg gewählt wird – die teilweise Anpassung der Software.
  - *Key User einschulen*: Key User sind Mitarbeiter, die genauestens über die Geschäftsprozesse Bescheid wissen, die von ihnen bearbeitet werden oder für die sie verant-

wortlich sind. Sie haben von seiten des Unternehmens die Befugnis, die Gestaltung der Software mitzubestimmen.

- *Endanwender einschulen*: Die Endanwender werden vorwiegend von den Key Usern eingeschult. Am Ende der zweiten Phase läuft BAAN IV im Echtzeitbetrieb.

- *Optimierung (Phase 3)*: Die endgültige Anpassung wir dann in Phase drei vorgenommen, in der die Software und Prozesse angepaßt (optimiert) werden.

2.3.2 Techniken des Software-Engineering

Eine kurze Beschreibung einiger grundlegender Techniken der Softwareentwicklung, die in weiterer Folge interessant sind.

2.3.2.1 Anforderungsanalyse

In Definitions- und Entwurfsphase sind laut Balzert [Balzert96-1] Structured Analysis (SA) und Real Time Analysis (RT) als Stand der Technik anzusehen. Eine starke Tendenz besteht aber in Richtung der objektorientierten Ansätze, siehe 2.2.3.

Eine interessante Bedeutung kommt dem Lasten- oder Pflichtenheft zu. Inhalt und Gestaltung hängen stark vom Autor ab, auch wenn nach einer formalisierten Methode vorgegangen wird. Bei der Beschreibungsmethode von Gruhn [Gruhn97] besteht die Beschreibung aus einer Liste von Anforderungen, die in Anforderungsname, Anforderungstext, Priorität und Anfordrungsbegründung unterteilt werden. Moderne Konzepte greifen auf Requirement Management Tools zurück, aus denen dann Reports erstellt werden können.

Ein grundsätzliches Problem kann aber auch damit nicht gelöst werden, Blaschek beschreibt es folgendermaßen: *Anforderungen stehen nicht ein für allemal fest, sondern ändern sich dynamisch mit dem Erfahrungsgrad und mit dem Stand der Technik* [Blaschek99].

2.3.2.2 Prototyping

Definition 8: Prototyping

Ein Prototyp repräsentiert bewußt ausgewählte Aspekte eines (Software-) Systems zur Beantwortung von Fragen oder zur Lösung von Aufgaben.

Es gibt verschiedene Klassifikationen für Prototyping. Prototypen kann man unterscheiden in Demonstrationsprototypen und funktionale Prototypen:

- Beispiel für einen *Demonstrationsprototypen*: Eine Eingabemaske eines zu erstellenden Programmes, die mit einem Grafikeditor erstellt wurde, aber über keinerlei Funktionalität verfügt.
- Beispiel für einen *funktionalen Prototypen*: Eine Eingabemaske des Systems, die teilweise mit Funktionalität hinterlegt ist. Performance wird dabei meist nicht beachtet.

Grundsätzlich kann man Prototyping anhand der Zielsetzung in drei Arten unterteilen [Floyd84]:

- *Exploratives Prototyping*: Hilfe zur Erklärung der Problemstellung aus der Sicht des Anwenders.
- *Experimentelles Prototyping*: Unterstützung der konstruktiven Umsetzung an ein System.
- *Evolutionäres Prototyping*: Teil eines inkrementellen Verfahrens zur schrittweise Entwicklung von Anwendungssoftware.

2.3.3 Qualität aus software-technischer Sicht

Definition 9: Qualität (nach DIN 55350)

> Softwarequalität ist die Gesamtheit der Eigenschaften oder Merkmale, die Software in Verwendung und (Weiter-) Entwicklung aufweist, um die an sie gestellten Qualitätsmerkmale zu erfüllen [Floyd97].

Es gibt verschiedene Annäherungen an den Begriff Qualität aus technischer Sicht, als Beispiel die Unterteilung in *interne* und *externe Qualitätsmerkmale*:

Als *externe Qualitätsmerkmale* werden dabei genannt [Chroust97]:

- *Effektivität*: Die Korrektheit und Konsistenz bei der Erfüllung der Aufgabe.
- *Effizienz*: Der Wirkungsgrad bei der Erfüllung der Aufgabe (Z.B.: Laufzeit).
- *Zuverlässigkeit* : Wiederherstellbarkeit, Kontrollierbarkeit, Sicherheit und Redundanz.

Interne Qualitätsmerkmale sind:

- *Wartungsfreundlichkeit*: Dokumentation, Transparenz, Normengerechtheit.
- *Anpassungsfähigkeit*: Flexibilität.
- *Maschinenabhängigkeit*: Portabilität und Kompatibilität.

2.4 Mensch-Maschine Kommunikation

George Bernard Shaw soll einmal gesagt haben, daß die Fachsprachen eine Verschwörung der Berufsstände gegen die Allgemeinheit sind. Genau das ist der Ansatzpunkt der Mensch-Maschine Kommunikation (MMK engl. HCI-Human Computer Interaction): Die Barriere die durch unterschiedliche Begriffswelten von Entwickler und Anwender (natürlich aber auch durch technische Zwänge und Paradigmen) zwischen Benutzern und elektronischen Systemen besteht, soll durchbrochen werden. Die Gesichtspunkte der Mensch-Maschine Kommunikation sind in weiterer Folge aus zwei Gründen interessant:

- *Gestaltung der Software*: Das in 1.4 vorgestellte Vorgehensmodell soll zur Entwicklung und Gestaltung eines qualitativ hochwertigen Software-Produktes dienen.
- *Darstellung des Modelles*: Um das Modell als Kommunikationsunterstützung nutzen zu können, muß die Darstellung des Modellinhaltes für Personengruppen mit verschiedenen Ausbildungshintergründen verständlich sein.

2.4.1 Mensch Maschine Kommunikation allgemein

Mensch-Maschine Kommunikation ist ein interdisziplinäres Forschungsgebiet, das sich mit der Gestaltung, Implementierung und Evaluierung interaktiver Computersysteme beschäftigt. Dabei werden unter diesem Gesichtspunkt unter anderem Beiträge aus Informatik, Psychologie, Soziologie und den Wirtschaftswissenschaften berücksichtigt.

2.4.1.1 Qualität in der MMK

Qualität in der MMK ist etwas sehr subjektives: *Die empfundene Qualität einer Benutzerschnittstelle ist wichtiger als die tatsächliche* [Blaschek99].

Beispiel: Ein Fortschrittsbalken, wie er bei zum Beispiel bei Lade- oder Kopiervorgängen angezeigt wird, verlangsamt das System. Der Benutzer empfindet das System aber als schneller, weil er ständig über den aktuellen Fortschritt informiert ist.

Ein zusätzliche Differenzierung kann auch getroffen werden, wenn man das Design des Produktes vom Design des Nutzens trennt.

- *Design des Produktes*: Unternehmenssoftware.
- *Design des Nutzens*: Unterstützung der Geschäftsprozesse, Teilautomatisierung, Unterstützung beim Treffen von Entscheidungen.

Tatsächlich muß die Betrachtung von Qualität auch von unterschiedlichen Standpunkten durchgeführt werden: Ein Mitarbeiter (eine einzelne Person) hat andere Vorstellungen von Qualität und damit auch von den Anforderungen (Beispiel: Benutzerfreundlichkeit), als das Unternehmen (Beispiel: Sicherheit).

2.4.2 Gestaltung von Schnittstellen

Wesentliche Bedeutung kommt der Gestaltung von Eingabefenstern zu. Durch die verbesserten technischen Möglichkeiten, aber auch durch die gestiegene Betrachtung des Benutzers haben sich deutliche Veränderungen ergeben.

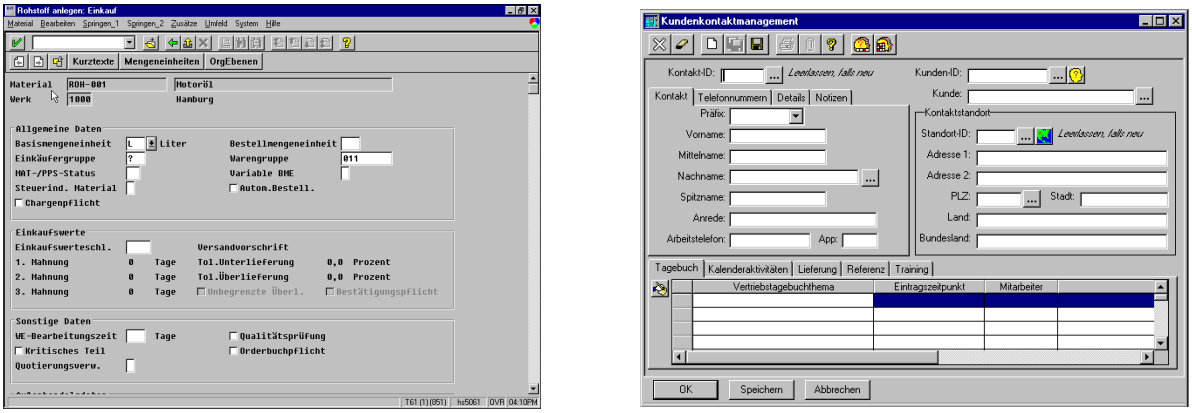

Abb. 8 Eine SAP und eine Applix-Schnittstelle

Ältere Softwareprodukte wie zum Beispiel SAP und BAAN verwendeten bis vor kurzer Zeit nur wenige Strukturierungshilfsmittel, wie zum Beispiel Register- oder Karteikarten. Diese

Schnittstellen haben starke Ähnlichkeiten zu (papierenen) Formularen. Neuere Software-Produkte (Applix [Applix], Fabasoft Components [Fabasoft] oder CSE Workflow [CSE]) verwenden diese dagegen sehr häufig.

Gerade zur Zeit werden die graphischen Benutzerschnittstellen von SAP R/3 zur Gänze umgestaltet, Version 4.6 stellt einen Umbruch in der Repräsentation von R/3 dar.

2.4.2.1 Schnittstellenprototyping

Schnittstellenprototyping ist ein wesentliches Hilfsmittel bei der Anforderungsanalyse und zur Erhöhung der Partizipation der betroffenen Personen. Beim *User centered design* [Gould85] werden die Benutzer und damit auch deren Aufgaben schon sehr früh in das Zentrum der Aufmerksamkeit gerückt. Mit Hilfe von Prototypen, Skizzen und Szenarien werden empirische Studien angefertigt, die dann iterativ weiterentwickelt werden.

User centered design zeichnet sich also durch integriertes, benutzerorientiertes Design und einen ebensolchen Systementwurf aus.

2.4.2.2 Prinzip der geringsten Überraschung

Dieses Prinzip kann zum Beispiel durch sogenannte *natural mappings* erreicht werden, dabei steht bei der Gestaltung des technischen Artefaktes die Erreichung einer hohen Ähnlichkeit mit einem realen Objekt, oder einem bekannten Konzept im Mittelpunkt. Durch diese geringe Überraschung soll die Einarbeitung erleichtert und die Fehlerhäufigkeit verringert werden. *Beispiel:* Die Verwendung eines Zahlschein-Bildes in einer Telebanking-Software.

Ein anderes Beispiel ist die Verwendung von *User Interface Style Guides (UISG)*. Microsoft legt fest, daß das erste Pull-down-Menü links oben in einer Applikation „Datei“ heißt. Windows-Applikationen die nach den MS-Style Guides erstellt werden, zeichnen sich demnach durch einen hohen Wiedererkennungswert aus.

2.4.2.3 Zustände

Im Sinne der MMK ist ein Zustand der „modus operandi“. Durch die Aufteilung von Befehlsmengen und Datenelementen auf verschiedene Eingabemasken entstehen zwei Arten der Benutzeraktionen:

- *Schritte zur Navigation*: Wechseln zwischen Zuständen.
- *Eingabeoperationen*: Arbeiten in einem bestimmten Zustand.

In vielen, meist älteren Programmen wurde versucht die Bearbeitung der Aufgabe nach einem Algorithmus des Entwicklers zu lösen [Blaschek97]. Der Benutzer hat dabei die Rolle eines Datenlieferanten. Dabei entstehen Zustandsräume die von einem Benutzer leicht als einengend empfunden werden, hier als Graphen dargestellt:

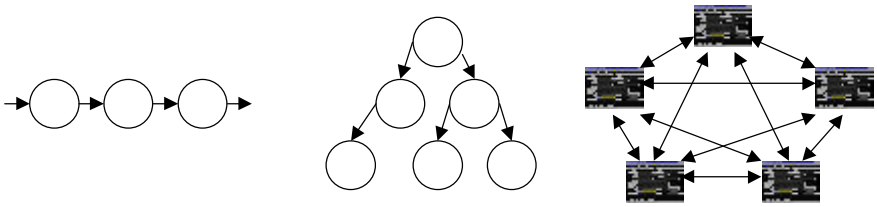

Abb. 9 Lineare und baumartige Zustandsübergänge. Übergänge zwischen Fenstern.

Durch den Einsatz von mehreren Fenstern (jedes repräsentiert einen komplexen Zustand), die eine möglichst logische Segmentierung von Daten und Funktionen darstellen, ist der Benutzer relativ frei in der Wahl der Zustandswechsel.

2.4.2.4 Der Faktor Mensch

MMK stellt den Menschen gezielt in den Mittelpunkt, bei der Einführung von organisatorischen Änderungen und einer neuen Software sind schwerwiegende Änderungen für den einzelnen erkennbar. Die Ängste, die einzelne Personen betreffen können, lassen sich in vier Kategorien unterteilen [Grünb98]:

- Allgemeiner *Widerstand* gegen *Änderungen*: „Das war schon immer so...“.
- *Ich-bezogene* Ängste: Furcht vor Statusverlust, Lernen und höherer Komplexität durch die Veränderungen.
- *Vorgehensbezogene* Ängste: Furcht vor Verlust von Freiheiten und Flexibilität oder auch Zweifel an der Zweckmäßigkeit des Vorgehens.
- *Technik-bezogene* Ängste: Scheu vor neuen Werkzeugen und ungewohnten Paradigmen.

2.5 Betriebswirtschaftliche Konzepte

Anhand einer kurzen Beschreibung einiger betriebswirtschaftlicher Grundlagen soll ein Überblick geschaffen werden. Dazu soll folgende Unterteilung dienen:

- *Organisation*: Beschreibung von Begriffen traditioneller Organisationsformen
- *Prozeßorientierte Organisation*: Beschreibung von Begriffen prozeßorientierter Organisation und Abgrenzung zu traditionellen Organisationsformen.

2.5.1 Organisation

Grundsätzlich müssen zur Erreichung des Unternehmensziels die einzelnen Aufgaben formuliert und die zur Erfüllung dieser Aufgaben benötigten Ressourcen (Human, Sachmittel und Werkzeuge) in einen sinnvollen Ordnungszusammenhang gebracht werden. Man kann zuerst in Aufbau- und Ablauforganisation unterteilen:

Definition 10: Aufbauorganisation [REFA84]

In der Aufbauorganisation werden die Aufgaben auf verschiedene Stellen aufgeteilt und die Zusammenarbeit dieser Stellen geregelt.

Die verschiedenen Ausprägungen von Aufbauorganisation, also die (*Weisungs*-) *Beziehungen* zwischen Stellen, können nach folgenden hierarchischen Leitungssystemen unterschieden werden: Liniensysteme (Einliniensystem, Mehrliniensystem, Matrixorganisation) und Stabliniensysteme [Frese93].

Definition 11: Ablauforganisation (nach [REFA84])

Die Ablauforganisation befaßt sich mit dem räumlichen und zeitlichen Zusammenwirken von Menschen, Betriebsmitteln, Arbeitsgegenständen und Informationen zur Erfüllung von Arbeitsaufgaben, also der *Planung, Gestaltung und Steuerung von Arbeitsabläufen*.

Kennzeichen Funktionaler Organisationen:

- Stark *hierarchische* Kommunikations- und Ordnungsbeziehungen.
- Strikte *Trennung von Fertigung und Disposition.*
- Ausrichtung von Innovation auf den Fertigungsbereich.

- Konzentration der Arbeitsmethodik auf Aufgabenzerlegung.

Der Zusammenhang zwischen Unternehmenssoftware und Organisationsform ist einfach aber schwerwiegend: Die *Aufbau- und Ablauforganisation des Informationsystems* stellt eine Projektion auf diejenigen Stellen und Beziehungen des Unternehmens dar, die informationsverarbeitende Aufgaben durchführen.

2.5.2 Prozeßorientierte Organisation

Der Unterschied zwischen einer klassisch und einer prozeßorientiert gestalteten Organisation läßt sich am einfachsten mit der folgenden Tabelle erklären. Die grundsätzliche Gliederung wurde [Langer94] entnommen.

	Klassisches Organisationsverständnis	*Prozeßorientiertes Organisationsverständnis*
Primäre Gestaltungs-elemente	• Aufgaben • Informationen und Daten • Personal und Sachmittel	• Geschäftsprozesse • Informationen und Daten • Ressourcen
Stufe 1	*Aufgabenanlayse* Ausgehend von Zielen und Hauptaufgaben werden die Aufgaben top-down zerlegt und konkretisiert Die Daten werden aus den Aufgaben abgeleitet.	*Prozeßfindung* Ausgehend von zu erzielenden Ergebnissen werden Geschäftsprozesse identifiziert.
Stufe 2	*Aufgabensynthese* Gruppierung von Aufgaben und Zuordnung zu Personal- und IT Ressourcen. Bildung von Stellen und Festlegung der Beziehungen. Festlegung von Anwendungsprogramm-Moduln.	Prozeßgestaltung Detaillierung der Prozesse in Aktivitäten Zuordnung zu Personalressourcen (Case-Worker, Case-Team, Prozeßverantwortliche) Festlegung von prozeßübergreifenden Zuständigkeiten Analyse und Festlegung der Anforderungen an ASW
Stufe 3	*Ablaufgestaltung* • Festlegen der Abläufe nach der Aufbauorganisation. • Beschreibung durch Arbeitsanweisungen innerhalb der Aufgaben.	

Abb. 10 Vergleich zwischen klassischem und prozeßorientiertem Organisationsverständnis

2.5.2.1 Arten von Prozessen

Zuerst die Definitionen einiger, in weiterer Folge, häufig verwendeter Begriffe:

Definition 12: Geschäftsprozeß [Langer94]

> Geschäftsprozesse sind Folgen von Aufgaben, die als Reaktionen eines Unternehmens auf externe, zeitliche oder interne Ereignisse ausgelöst werden, und die in einen unmittelbaren Zusammenhang mit dem Zielsystem der Unternehmung gebracht werden können.

Geschäftsprozesse können in Kern- und Supportprozesse unterschieden werden [Osterloh97]:

Definition 13: Kernprozeß

> Kernprozesse erzeugen wahrnehmbaren Kundennutzen, sind durch eine spezifische Nutzung von Unternehmensressourcen einmalig, nicht imitierbar und nicht substituierbar.

Definition 14: Supportprozeß

> Supportprozesse dienen dazu, Kernprozesse in ihrem Ablauf zu unterstützen (die Kernprozesse sind die „Prozeßkunden“), sie sind nicht unternehmensspezifisch und daher imitier- und substituierbar.

Bei der *Unterstützung der Kernprozesse* wird ein starker Individualcharakter determinierend sein, dieser Individualcharakter ist ein wichtiger Parameter bei der Auswahl der Software Unterstützung. Kernprozesse sind schließlich das Charakteristikum des Unternehmens, eben per Definition nicht substituierbar. Wenn Kernprozesse durch Standardsoftware unterstützt werden sollen, können sich durch diesen Individualcharakter Probleme ergeben.

Supportprozesse hingegen können meist von Standardsoftware, beziehungsweise von standardisierten Komponenten, in befriedigendem Ausmaß erfüllt werden. Hier ist dann auf die Integration in die Gesamtstruktur der IT-Landschaft des Unternehmens zu achten. Softwarelösungen für Supportprozesse sind mit größter Wahrscheinlichkeit bereits am Markt vorhanden.

2.6 Ziele und Zielsysteme

Eine grobe Unterteilung der Ziele kann man treffen in:

- *Hard-Factor Ziele*: Quantifizierbare Ziele, zum Beispiel Kosten, Umsatz,...
- *Soft-Factor Ziele*: Nicht quantifizierbare Ziele, zum Beispiel Zufriedenheit, Arbeitsatmposphäre,...

Zielstrukturen werden meist mit Hilfe eines baumartigen Graphen dargestellt. Wichtig scheint die Operationalisierung von Zielen, wobei man naturgemäß Hard-Factor Ziele leichter operationalisieren kann. Auch Langer weist auf die Identifikation und Operationalisierung [Langer94] hin.

Es gibt auch Werkzeuge die eine gezielte Betrachtung von Ziele ermöglichen sollen, ein Beispiel wäre das Zielfindungstool von Reichwald [Reichwald96].

2.7 Zusammenfassung des Kapitels

Als erstes wurden einige Grundlagen über *Modellierung* beschrieben, wobei schon ein gewisses Augenmerk auf die spätere Verfeinerung dieses Themas (Unternehmensmodellierung und dazu vorgesehene Werkzeuge) gelegt wurde.

Grundsätzliche Begriffe de sind *Softwaretechnik* in dieser Diplomarbeit besonders deshalb interessant, weil der Gedanke der gemeinsamen Gestaltung von Unternehmenssoftware und Geschäftsprozessen konkret in ein Vorgehensmodell, wie es zur Softwareentwicklung angewandt wird (oder werden soll), integriert worden ist. Diese Integration wird im Kapitel über die Fallstudien näher beschrieben. Speziell in der Anforderungsanalyse ergeben sich daraus neue Gesichtspunkte.

Der genannte Aspekt der gemeinsamen Gestaltung bietet sich an, um die Rolle des *Menschen* (des betroffenen Mitarbeiters) besonders zu betrachten: Der Mensch soll den Geschäftsprozeß mit Hilfe der Softwarelösung ausführen. Zentrales Element der weiteren Betrachtung ist daher die Gestaltung der Schnittstellen.

Bei den beschriebenen *betriebswirtschaftlichen Konzepten* ist besonders die Unterscheidung zwischen klassischem und prozeßorientiertem Organisationsverständnis wichtig.

Interessant sind auch die verschiedenen Ansichten von Qualität (vgl. 2.3.3 und 2.4.1.1) in den verschiedenen Wissensgebieten.

3 Einsatz von IT im gemeinsamen Gestaltungsprozeß

3.1 Ziele des Kapitels

Der Einsatz eines Unternehmensmodellierungswerkzeuges ist zentraler Punkt dieser Diplomarbeit. Deshalb wird Unternehmensmodellierung zunächst allgemein charakterisiert.

Zur Gestaltung und Entwicklung von Softwaresystemen werden traditionell verschiedene Werkzeuge verwendet. Die Gestaltung und Entwicklung von Software – und natürlich den Geschäftsprozessen die dadurch unterstützt werden - soll in dieser Arbeit mit Hilfe eines Unternehmensmodellierungswerkzeuges erreicht werden. Deshalb erfolgt auch eine grundsätzliche Betrachtung von konventionellen Software-Entwicklungswerkzeugen (CASE-Tools).

Abschließend werden das verwendete Werkzeug (AENEIS) sowie zwei andere Werkzeuge grundsätzlich beschrieben.

3.2 Unternehmensmodellierung

3.2.1 Allgemeines

Unternehmensmodellierungswerkzeuge (UM-Werkzeuge) sind unter verschiedenen Bezeichnungen bekannt, eine einheitliche Klassifikation gibt es nicht, Synonyme sind unter anderem:

- Geschäftsprozeß Optimierungs-Tools (GPO-Tools)
- Tools (Werkzeuge) zur Geschäftsprozeßorganisation
- Organisations-Engineering-Tools (OE-Tools)
- Business Modelling Tools (BMS-Tools)
- Business Process Management Systems (BPMS)
- ...

Diese Werkzeuge sind für „ähnliche“ Aufgaben konzipiert. Ähnlich bedeutet, daß sie zum großen Teil Realisierungen ähnlicher Konzepte darstellen, allerdings mit leicht unterschiedlichen Zielsetzungen. Gemeinsam ist die Fähigkeit wichtige Informationen über Unternehmen

zu modellieren. Die Palette der Aufgaben ist relativ breit gefächert. Aktivitäten die mit Hilfe von UM-Werkzeugen in der Praxis ausgeführt werden, sind beispielsweise [Ronaghi99]:

- Modellierung von Geschäftsprozessen aus fachlicher Sicht (sehr wichtig für 76 % der Unternehmen, die UM-Werkzeuge einsetzen)
- Permanentes Geschäftsprozeßmanagement (65%)
- Durchführung von Mitarbeiterschulung und Fortbildung (63%)
- ...
- Einführung von Standard-Software (42%)

Die Einführung von Standard-Software wird von 42% als Inhalt eines Projektes angegeben, das mit der Unterstützung eines UM-Werkzeuges durchgeführt wurde. Das Vorhandensein einer Schnittstelle zu Standard-Software ist aber für 72% der Anwender eine sehr wichtige Funktionalität [Ronaghi99]. Damit läßt sich auch leicht erklären, warum die später genannten Werkzeuge alle über SAP R/3 Schnittstellen verfügen.

Die Durchführung von Analyse wird bei UM-Werkzeugen prinzipiell durch Simulation unterstützt. Als problematisch ist dabei die Datenerfassung (Zeiten, Häufigkeiten, Wahrscheinlichkeiten) zu betrachten, die im Modell hinterlegt wird.

3.2.2 Modellierung

Gemeinsam sind den verschiedenen UM-Werkzeugen die Möglichkeiten zur Abbildung organisatorischer Sachverhalte [Buresch97]:

- Funktionenmodell
- Prozeßmodell
- Ressourcenmodell
- Informationsmodell
- Stellenmodell

Prozesse und Abläufe sollen computergestützt erfaßt und dargestellt werden. Wesentlich ist dabei die Zuordnung von Daten und Personen zu den Arbeitsschritten. Die Beschreibung von

Prozessen (Abläufen) kann man in graphische Diagramme und sprachliche Darstellungen unterteilen [Grünb98]:

- *Graphische Diagramme*: Petri-Netze und Weiterentwicklungen (siehe 2.2), Structured Analysis and Design Technique, siehe auch 2.3.2.1 und [Balzert96-1].
- *Sprachliche Darstellungen*: Beschreibung durch strukturierte Humansprache, regelbasierte Formulierung und ähnliches. Ein Beispiel sind Produktflußtabellen aus dem V-Modell [Broehl93]. Auch bei verschiedenen objektorientierten Modellierungsansätzen werden sprachliche Darstellungen verwendet.

Laut James Martin [Balzert96-2] sind ER-Diagramme (siehe 2.2.3) die weitverbreitetste Darstellungsform für (relationale) Datenmodelle im deutschen Sprachraum. Objektorientierte Datenbanken setzen sich erst in letzter Zeit durch. Diese Aussage ist speziell in Hinblick auf die notwendige Unterstützung durch ein Modellierungswerkzeug wichtig.

3.3 Softwareunterstützung allgemein

Man kann in dem Vorgehensmodell, das in 1.4.2.4 beschrieben wurde, leicht die verschiedenen Anwendungsgebiete für Softwareunterstützung erkennen. Deshalb bietet sich zuerst die Betrachtung des Begriffes CASE-Tool an.

Als CASE-Tools (*Computer Aided System Engineering* - Tools) werden alle Software-Werkzeuge bezeichnet, die zum Zweck der Software-Entwicklung eingesetzt werden. Sie bilden gemeinsam eine CASE-Umgebung. Aufgaben dieser Umgebung sind Projekt- und Prozeßmanagement, Objektmanagement, Werkzeugintegration und Unterstützung der Kooperation der beteiligten Personen sowie der Kommunikation [Grünb98].

Man kann also auch Unternehmensmodellierungswerkzeuge, wenn sie im hier gegebenen Kontext verwendet werden, als CASE-Tools bezeichnen – genauer gesagt wird das UM-Werkzeug als CASE-Tool eingesetzt. Allerdings ist die Integration (beispielsweise mit der Entwicklungsumgebung) nicht derart ausgeprägt, wie sie bei einem CASE-Tools sein sollte.

Eine Integration einer Unternehmensmodellierungssoftware in einer CASE-Umgebung (Schaffung von Schnittstellen) ist grundsätzlich wünschenswert, um die Reibungsverluste mit anderen Werkzeugen, einschließlich der Software-Entwicklungsumgebung, gering zu halten.

Allerdings scheint das Konzept, das in 1.4.2.4 beschrieben wurde, nicht ganz in die klassische Aufteilung von CASE-Tools zu passen, denn grundsätzlich wird dabei in *Front-end* und *Back-end CASE-Werkzeuge* unterschieden.

Definition 15: Front-end CASE-Werkzeuge

> ...sind jene Werkzeuge, die die ersten Phasen einer Software-Entwicklung (Planung, Definition, Entwurf) unterstützen [Balzert96-2].

Definition 16: Back-end CASE-Werkzeuge

> ... sind jene Werkzeuge, die die späten Phasen einer Software-Entwicklung (Implementierung, Abnahme und Einführung, Wartung und Pflege) unterstützen [Balzert96-2].

Vielmehr könnte eine Zuordnung zur übergreifenden Kategorie der *cross life cycle tools* erfolgen. Folgende Punkte sprechen gegen die Einordnung eines Unternehmensmodellierungswerkzeuges als CASE Tool:

- Unternehmensmodellierungswerkzeuge dienen nicht nur als Werkzeug zur Entwicklung von Software, sondern (im gegebenen Kontext) als Werkzeug zur gemeinsamen Gestaltung von Geschäftsprozessen und Unternehmenssoftware.
- Zielgruppe der Benutzer (1): Im Gegensatz zu CASE-Systemen soll auch ein Fachanwender, also ein beliebiger Mitarbeiter eines Unternehmens, in der Lage sein, das Werkzeug zu bedienen. Später soll man Informationen aus dem Unternehmensmodell gewinnen können (Dokumentation). Dadurch kann die Benutzerakzeptanz des Unternehmensmodellierungs-Tools und damit des gesamten Projektes verbessert werden, wie in einem internen Bewertungskatalog über ein Unternehmensmodellierungswerkzeug zu lesen war [BEKO99]. Dies kann die Partizipation der Endanwender an der Gestaltung des Anwendungssystems verbessern, was mit CASE-Tools sehr schwierig ist.
- Zielgruppe der Benutzer (2): CASE-System werden von *allen Mitgliedern* eines Software-Entwicklungsteams verwendet, Unternehmensmodellierungswerkzeuge nicht.

3.4 Unternehmensmodellierungssoftware

3.4.1 Grundsätzliches

Im Sinne des Software Engineering übt das UM-Werkzeug eine Funktion als Repository für die Organisations- und Softwareentwicklung aus, es fungiert als Verwahrungsort für Informationen.

Die folgenden Beschreibungen der genannten Werkzeuge ist keinesfalls als vollständige Analyse oder Beschreibung gedacht: Im gegebenen Kontext interessante Merkmale (Beispiele: Berichtsgenerierung, Notation, Modellierungstechnik,...) sollten dabei vorrangig behandelt werden.

3.4.2 Das verwendete Werkzeug: AENEIS

Wie auch die beiden anderen vorgestellten Modellierungswerkzeuge wurde das Werkzeug AENEIS (*A*nalyse, *E*ntwicklung und *N*utzung *e*ingebetteter *I*nformationssysteme) an einer Universität (Hannover) entwickelt, später aber kommerziell weiterentwickelt und angeboten [Ipro]. Für unsere Arbeiten stand die Version 4.0 von AENEIS zur Verfügung. Laut Auskunft der Firma Ipro wurden bisher über 1000 Lizenzen verkauft, zusätzlich soll es einen nicht genannten Großkonzern geben, der 700 Kopien mit reduziertem Funktionsumfang einsetzt (OEM-Version).

Neben den für UM-Werkzeuge typischen Funktionalitäten kann AENEIS auch zur Unterstützung der Ausführung von Unternehmensprozessen (Workflowsteuerung) verwendet werden. Diese Möglichkeit ist in der Ausführungskomponente implementiert.

Weitere Komponenten von AENEIS sind Modellierungs-, Analyse-, Prozeßkostenrechnungs- und Dokumentationskomponente.

3.4.2.1 Grundsätzlicher Aufbau

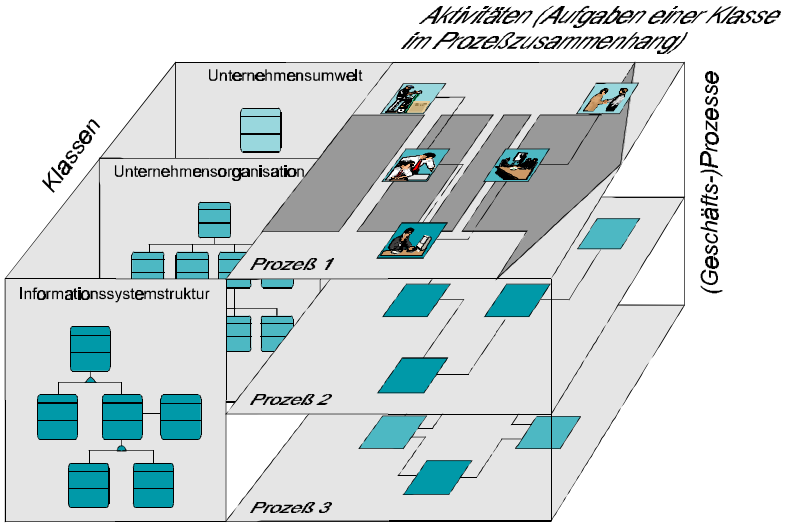

Abb. 11 Das AENEIS-Konzept [Langer94]

Man kann in Abb. 11 erkennen, daß eine Unterteilung in Unternehmensumwelt, Unternehmensorganisation, Informationsstruktur und Geschäftsprozesse vorgenommen wurde. Dabei wird grundsätzlich in s*tatische* und *dynamische* Diagramme unterschieden.

- *Statische Diagramme* beinhalten beispielsweise Organigramm (Personen, Rollen und Beziehungen) , Sachmittel, Informationssysteme (Klassenebene, Modulebene, Datenstruktur,...) oder Daten.
- *Dynamische Diagramme* dienen zur Beschreibung von Geschäftsprozessen.

Zusätzlich kann man noch *generische Strukturen* erzeugen (baumartig), die sich beispielsweise zum Generieren von Qualitätshandbüchern (nach ISO 900x) und von Zielbäumen einsetzen lassen.

Grundsätzlich wird objektorientiert modelliert, wobei auf eine strenge Klassifizierung verzichtet wurde. Man ist dadurch an keine strenge Vorgehensweise gebunden, was sehr hilfreich sein kann. Abgespeichert wird in einer objektorientierten Datenbank (POET); die abgespeicherte Struktur entspricht weitestgehend der sichtbaren Struktur an der Oberfläche, wie man mit Hilfe des mitgelieferten Datenbankadministrator-Tools überprüfen kann.

3.4.2.2 Notation

AENEIS verwendet eine *eigene Notation* zur Darstellung von Arbeitsabläufen, man unterscheidet in die Begriffe Geschäftsprozeß, Aktivität und Arbeitsschritt.

- Ein *Geschäftsprozeß* besteht aus einer Menge von Aktivitäten und ihren Beziehungen zueinander. Jeder Geschäftsprozeß hat mindestens eine *Startaktivität*. Diese ist in Abb. 12 strichliert umrandet. Das Beispiel in der unten eingefügten Abbildung zeigt den Geschäftsprozeß Aktivvertrieb auf der Aktivitätenebene.

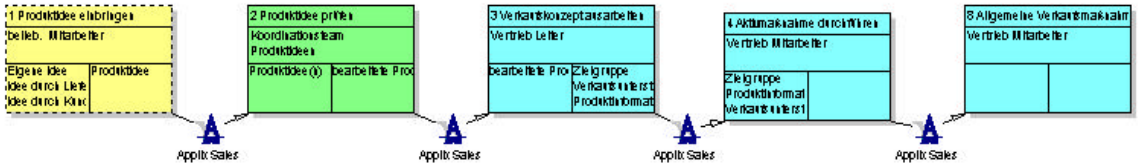

Abb. 12 Ein Geschäftsprozeß in AENEIS

- Eine *Aktivität* besteht aus einer Eingangschnittstelle, einer Ausgangsschnittstelle und einer Menge von Arbeitsschritten. Startaktivitäten sind Aktivitäten die keinen Vorgänger haben.
- *Arbeitschritte* können sequentiell, parallel, in Schleifen (do while, repeat until) und in Verzweigungen (einfach und mehrfach) angelegt werden. Arbeitsschritte können auch verschachtelt verwendet werden. In der Abb. 13 wurde der Geschäftsprozeß aus Abb. 12 dargestellt, wobei diesmal die Detailansicht der dritten Aktivität geöffnet wurde, die Arbeitsschritte sind nun sichtbar.

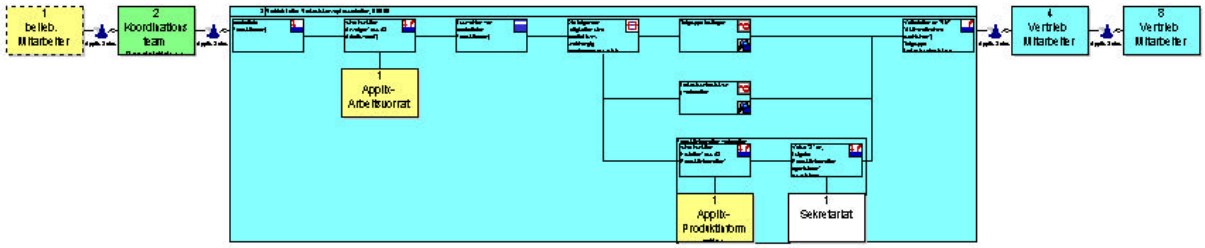

Abb. 13 GP in AENEIS mit Detailansicht einer Aktivität

Eine Umschaltung von AENEIS-Notation zu EPK-Notation (siehe 2.2.3) ist auf Top-Level Ebene (Geschäftsprozeßebene) möglich. Allerdings wird dann auf Aktivitätsebene wieder die AENEIS-Notation angezeigt.

Zur Darstellung von Datenmodellen können Objektdiagramme, Klassendiagramme, Moduldiagramme und Transaktionssysteme verwendet werden. Generell wird vorwiegend objektorientiertes Design unterstützt.

3.4.2.3 Besondere Eigenschaften

Als Hilfsmittel gibt es *generische Referenzlisten.* Da generell Objekte abgebildet werden, können zwischen beliebigen Objekten (also Objekten verschiedener Klassen) Referenzlisten eingefügt werden. Diese Listenobjekte können beliebig benannt werden und speichern eine Liste von anderen Objekten ab, wobei erwähnt werden muß, daß man in einer Referenzliste Objekte beliebigen Typs speichern kann.

Beispiel: Eine Person, ein Arbeitsschritt und ein Geschäftsprozeß können von einer Referenzliste, die zu einem Informationssystem-Objekt gehört, referenziert werden..

AENEIS verfügt über einen *Berichtsgenerator*, der sich über eine eigene Berichtssprache „theoretisch“ beliebig anpassen läßt. Man kann damit Berichte in einem Textverarbeitungsprogramm (Beispiel Word für Dokumentation, ISO Handbücher, Pflichtenhefte,...), einem Tabellenkalkulationsprogramm (Bsp. Excel für Kostenrechnung) oder im HTML-Format generieren. Diese Anpaßbarkeit des Berichtsgenerators ermöglicht es AENEIS, eine Schnittstellenfunktion einzunehmen und zur Kommunikationsunterstützung dienen zu können.

In der neuesten Version (4.5) wurde eine Integration von Microsoft Visual-Basic for Applications (VBA) durchgeführt, um die Anpaßbarkeit der Benutzerschnittstelle zu verbessern. Außerdem ist es in der neuesten Version möglich, JAVA-Code zu generieren.

Laut [Fank98] ist die objektweise Versionierung ein hervorragendes Merkmal von AENEIS im Vergleich zu anderen UM-Werkzeugen. Damit kann besonders bei gemeinsamem Bearbeiten eines Modelles protokolliert werden, wer, wann, welche Änderungen an einem bestimmten Objekt (mit aktivierter Versionierung) vorgenommen hat.

Weitere Informationen werden im Zusammenhang mit den Arbeiten an den Fallstudien präsentiert.

3.4.3 Das ARIS-Toolset der IDS Scheer

ARIS (Architektur Integrierter Systeme) ist das weitverbreitetste UM-Werkzeug. Weltweit gibt es über 10000 Lizenznehmer [BEKO99]. In Österreich ist ARIS mit 23% Marktführer [Ronaghi99]. Zum Vergleich wurde die Testversion (Version 4.0) von der Homepage der IDS Prof. Scheer GmbH [IDS] verwendet.

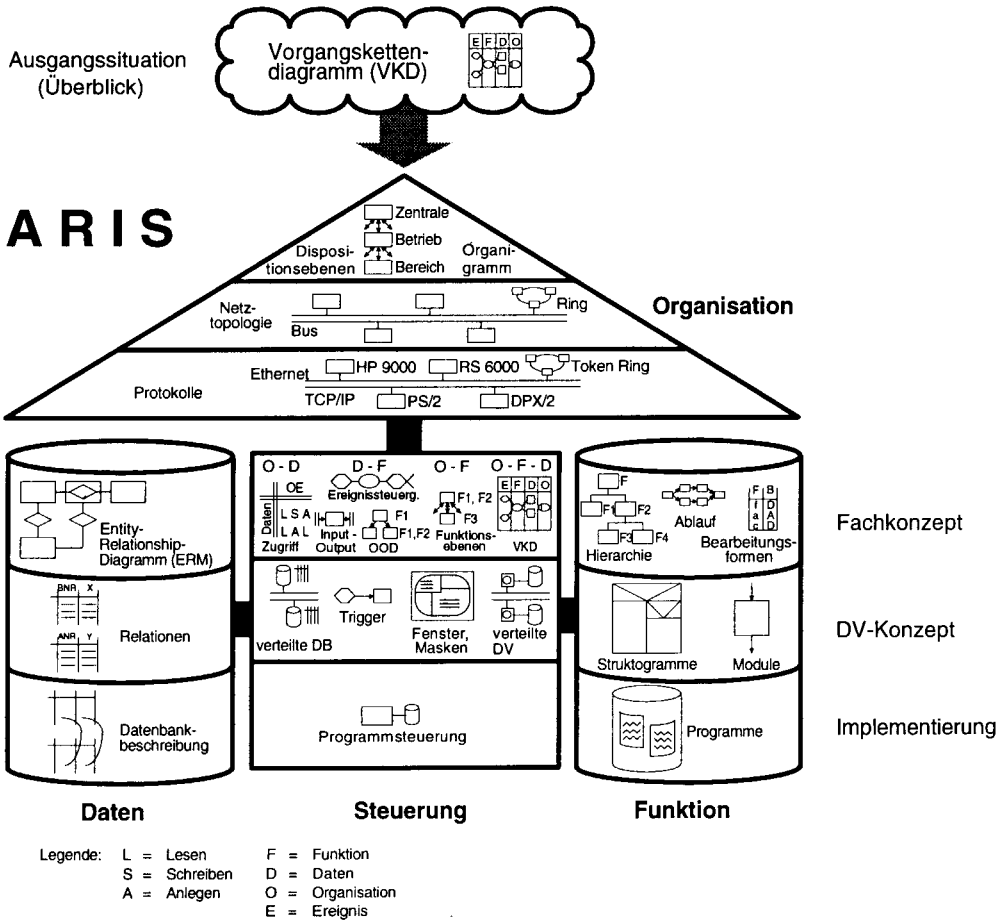

Abb. 14 Das ARIS-Konzept [Scheer94]

Grundsätzlich Zielsetzung ist die Entwicklung eines Informationssystems, ausgehend von den betriebswirtschaftlichen Erfordernissen.

3.4.3.1 Grundsätzlicher Aufbau

Die ARIS-Architektur unterscheidet folgende vier Beschreibungssichten:

- *Funktionssicht*: Die Funktionssicht beschreibt die auszuführenden Funktionen (Vorgänge) eines Unternehmens sowie ihre hierarchischen Zusammenhänge. In ARIS wird diese Sicht Funktionen genannt.
- *Datensicht*: Die Datensicht beschreibt die Ereignisse und Zustände des Bezugsumfelds von Unternehmen. In ARIS wird diese Sicht Daten genannt.
- *Organisationssicht*: Die Organisationssicht beschreibt die Organisationseinheiten und Bearbeiter eines Unternehmens sowie ihre Beziehungen und Strukturen. In ARIS wird diese Sicht Organisation genannt.
- *Steuerungssicht*: In der Steuerungssicht werden die Verbindungen zwischen den drei Einzelsichten beschrieben. Im Zentrum der Steuerungssicht stehen die Geschäftsprozesse. In ARIS wird diese Sicht Prozesse genannt.

Es existiert eine große Anzahl von Zusatzmodulen und Schnittstellen (Beispiel: MS-Project).

3.4.3.2 Notation

Die Darstellung von Prozessen erfolgt in der Steuerungssicht, diese enthält unter anderem folgende Modelltypen [Fank98]:

- Erweiterte ereignisgesteuerte Prozeßkette (eEPK), mit und ohne Materialfluß
- Ereignisdiagramm
- Funktionsausführungsdiagramm
- Vorgangskettendiagramme
- Informationsflußdiagramme
- ...

Ein typisches Prozeßdiagramm (eEPK) sieht so aus:

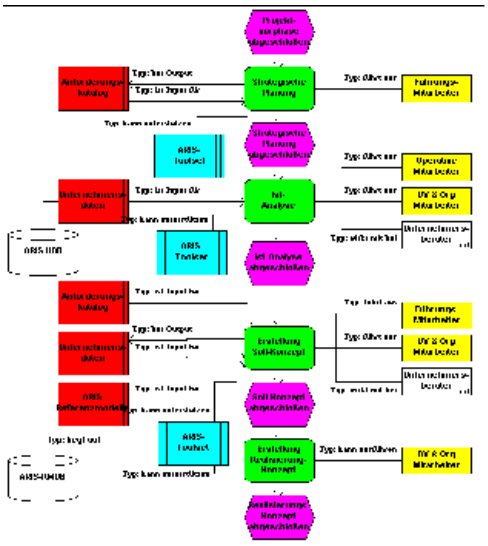

Abb. 15 eEPK Notation

Man unterscheidet in Objekt- und Beziehungstypen [IDS]. Objekttypen in eEPK sind zum Beispiel Ereignis, Funktion und Regel. Wenn man Objekttypen und Beziehungstypen zu einem Satz verbindet, so ergibt sich:

- Ereignis aktiviert Funktion (in der Aktivform),
- Funktion wird aktiviert durch Ereignis (in der Passivform).

3.4.3.3 Besondere Eigenschaften

Der Berichtsgenerator von ARIS kann mit Hilfe von Visual Basic Programmierung verändert werden, die Standardvariante wird allerdings als „sehr gewöhnungsbedürftig“ bezeichnet [BEKO99]. Die Erstellung von HTML-Dokumenten und die verteilte Verwendung von ARIS-Modellen ist mit dem Zusatzprodukt ARIS-Weblink möglich.

Darüber hinaus verfügt ARIS über ein Vielzahl von zusätzlichen Komponenten. Beispiel Zusatzmodul zur objektorientierten Modellierung [Fank99].

3.4.4 ADONIS von BOC

Laut [Ronaghi99] ist ADONIS mit einem Marktanteil von 13% an zweiter Stelle in Österreich. Es wird von der BOC GmbH Wien [BOC-1] angeboten die aus der BPMS-Gruppe der Universität Wien hervorgegangen ist.

Das Haupeinsatzgebiet für ADONIS sieht die BOC GmbH in der Unterstützung von Reengineering Prozessen und der Evaluierung der Prozeß-Performance [BOC-2]. Die Evaluierung einer Testversion war nicht möglich, dafür konnte eine ADONIS-Basisschulung besucht werden.

3.4.4.1 Grundsätzlicher Aufbau

ADONIS verfügt über mehrere Komponenten: Import/Export, Modellierung, Analyse, Transformation, Simulation, Evaluation und Erhebung. Zusätzlich gibt es ein Dokument-Toolkit.

Durch die Einführung einer Metamodellebene ist es möglich die Art der Modellierung in ADONIS selbst zu bestimmen. Auf dieser Ebene setzten Methodenbibliotheken auf, die „frei“ definierbar sind. Man ist daher weit weniger an eine bestimmte Modellierungstechnik gebunden wie in den anderen Modellierungsumgebungen. Auf der Metamodellebene sind grundsätzliche Eigenschaften implementiert: Ein Kunde kann entweder die ADONIS-Notation verwenden, EPK, oder eine selbst entwickelte Notation.

3.4.4.2 Notation

Ein Geschäftsprozeß besteht aus einer Menge von Aktivitäten. Anders als bei AENEIS sind diese als atomare Arbeitsschritte zu betrachten. Verschachtelung und damit Strukturierung wird durch den Aufruf von Subprozessen, ähnlich wie in ARIS, realisiert. Jeder Geschäftsprozeß verfügt über genau einen Startpunkt sowie über mindestens einen Endpunkt.

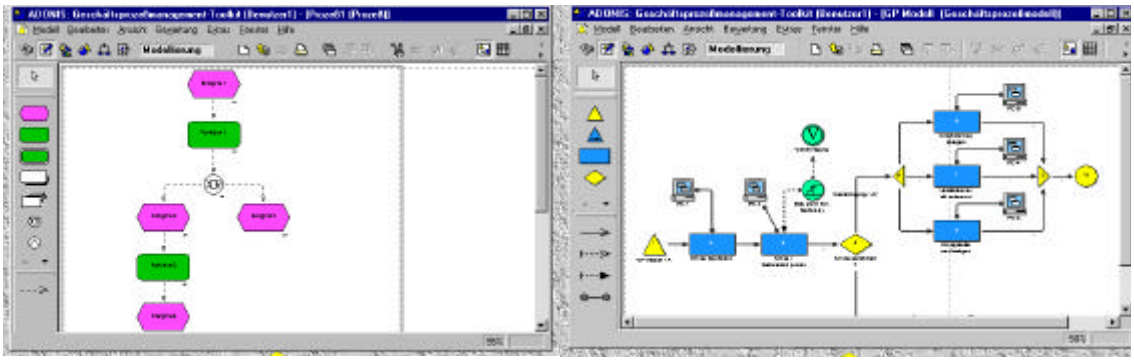

Abb. 16 EPK-Notation und ADONIS-Notation

In der rechten Abbildung sieht man ganz links einen *Startpunkt* (Dreieck) und ganz rechts einen *Endpunkt* (Kreis).

3.4.4.3 Besondere Eigenschaften

Zusätzlich zur Verbreitung von Modellinformationen über WWW (Inter- und Intranet), kann man mit ADONIS auch eine Anbindung an Lotus Notes durchführen.

3.4.5 Integrationsansätze

Es gibt auch Beispiele für die Integration eines Unternehmensmodellierungswerkzeuges in der Softwareentwicklung. Naturgemäß sind es größere Anbieter von Workflowsystemen die sich intensiv mit der Integration von Unternehmensmodellierungssoftware in die Softwareentwicklung beschäftigen.

Beispiel: Die Version 3.0 von Fabasoft Components [Faba] verfügt über eine Schnittstelle zu ARIS (3.4.3). Modellierung und Simulation werden in ARIS durchgeführt, das Prozeßmodell (Abläufe, Organisation und Geschäftsobjekte) wird von Fabasoft Components (FSC) aus dem ARIS-Modell übernommen. Die Geschäftsprozesse können in FSC ausgeführt werden, die Ist-Daten aus dem Prozeß werden zur Kostenrechnung und Simulation wieder in ARIS übernommen. Zu FSC gibt es aber auch eine Anbindung an ADONIS (siehe 3.4.4).

Weitere Beispiele gibt es unter anderem von CSE Systems [CSE], wo CSE-Workflow ebenfalls mit ARIS kombiniert wurde.

3.5 Zusammenfassung des Kapitels

Um die Aufgaben und Fähigkeiten eines Unternehmensmodellierungswerkzeuges genauer zu beschreiben wurde der Begriff Unternehmensmodellierung betrachtet, dann wurde eine Positionierung von einem UM-Werkzeug wie AENEIS zu CASE-Tools vorgenommen.

Der daraus gezogen Schluß war, daß UM-Werkzeuge nicht als CASE-Tools bezeichnet werden können, weil ihr Primärziel die Organisationsgestaltung ist.

Dann folgte eine kurze Beschreibung des verwendeten Werkzeuges AENEIS, sowie zweier weiterer Produkte mit ähnlicher Zielsetzung, ARIS und ADONIS. Dabei wurden jeweils der grundsätzliche Aufbau und die Notationen der Modellierung besonders betrachtet.

Besondere Aufmerksamkeit haben wir den Berichtsgeneratoren der verschiedenen Werkzeuge geschenkt, da diesen bei der Kommunikationsunterstützung eine wesentliche Bedeutung zukommt.

Die genannten Tools bieten alle die Möglichkeit, in die Generierung von Berichten einzugreifen (HTML und Text). Modifikationen sind meist nur mit einem größerem Aufwand durchführbar, alle Hersteller bieten Modifikationen im Rahmen von Consulting-Tagen an.

Während unserer Recherche über das Thema Modellierung haben wir festgestellt, daß sich praktisch sämtliche Literatur im deutschen Sprachraum, die sich mit direkt oder indirekt mit Unternehmensmodellierung beschäftigt, ARIS als Beispiel verwendet. Beispiele dafür sind [Becker99], [Staud99] oder [Brenner95].

Ein weiteres Indiz für die Bedeutung sind die oben genannten Integrationsprojekte der Anbieter von Workflowprodukten (Fabasoft und CSE). Für ein Workflowprodukt scheint die ARIS-Schnittstelle ebenso obligat zu sein, wie die SAP/R3-Schnittstelle für ein Unternehmensmodellierungswerkzeug.

4 Fallstudien

4.1 Ziele des Kapitels

Grundsätzliche Ziele der folgenden Ausführungen über die beiden Fallstudien, die im Rahmen dieser Diplomarbeit durchgeführt wurden, sind:

- Die *Überprüfung* der grundsätzlichen Machbar- und Wirksamkeit des beschriebenen Vorgehensmodelles, im besonderen der Zusammenarbeit von Betriebswirt und Informatiker bei der Gestaltung von Geschäftsprozessen und Softwaresystemen.
- Die *Gewinnung* neuer Erkenntnisse über den Einsatz eines konkreten Unternehmensmodellierungswerkzeuges bei einem Business Engineering Projekt. Speziell die Art der Integration und Representation der Software im und durch das Modell sollte dabei verbessert werden.

4.2 Erste Fallstudie mit Firma XXX

Ziel dieser Fallstudie war es, eine Bewertung der umgestalteten Geschäftsprozesse und der neu entwickelten Software durchzuführen. Das sollte durch einen Vergleich mit der alten Software (und den alten Geschäftsprozessen) geschehen. Dieses Ziel sollte auf Basis der Modelle von Ist- und Sollprozessen und dem alten und neuen Softwaresystem geschehen. Nach dem Abschluß der Modellierung sollte eine geeignete Art der Analyse gefunden werden, um die Erreichung der Projektziele zu kontrollieren.

Leider konnte dieses Ziel aufgrund des Projektverlaufs der Anpassung und Einführung der neuen Software nicht erreicht werden, aus dem Projektverlauf konnten aber Schlußfolgerungen für die zweite Fallstudie gezogen werden. Aus diesem Grund ist eine kurze Beschreibung der einige Monate dauernden Zusammenarbeit sinnvoll.

4.2.1 Ausgangssituation

Die Firma *XXX Tresor* ist eine Handelsfirma mit Sitz in St. Georgen im Attergau. Die vertriebenen Waren sind hauptsächlich Tresore und Briefkästen, der Marktanteil in diesem Bereich

lag zum Zeitpunkt der Fallstudie bei über 90 Prozent (Österreich). Zusätzlich werden noch Accessoires (Schlüsselanhänger etc.) vertrieben.

Die ursprüngliche softwaretechnische Unterstützung war durch ein speziell für die Firma XXX adaptiertes Softwaresystem gegeben. Durch massive Umstellungsschwierigkeiten von Version 3 auf Version 4 – zusätzlich konnten die spezifizierten Anforderungen nicht erfüllt werden – entstand der Wunsch nach einer neuen Lösung.

Ziel des Projektes war die Gestaltung und Anpassung einer vollständigen Unternehmenssoftware. Dadurch sollten Auslieferungsfehler minimiert werden (laut Schätzungen lagen diese bei 10-20%), sowie in weiterer Folge Personaleinsparungen möglich sein.

Begonnen wurde das Projekt mit einer Modellierung des Soll-Zustandes. Bei dieser Modellierung wurden von Dr. Schröder mehrere Mitarbeiter der Firma XXX herangezogen. Das Modell wurde dann als Spezifikation für die Anforderungen an die Software betrachtet und sollte in weiterer Folge gemeinsam mit der Software weiterentwickelt werden.

Erwähnt werden muß, daß die ursprüngliche Spezifikation (Anforderungen an die SW) während des Projektes erweitert wurde, indem ein Lagerverwaltungsystem integriert werden sollte, das auf dem Einsatz von EAN-Lesegeräten basieren sollte.

4.2.2 Vorgehensweise

Ebenso wie bei der Erstellung des Soll-Modelles wurden wieder die selben Mitarbeiter zur Modellierung herangezogen. Modelliert wurde ein ausgewählter Geschäftsprozeß (Abwicklung Handelsauftrag).

Aufgrund der Erfahrung aus den Gesprächen wurde der Einsatz einer Videoanalyse zur Zeiterfassung eingeplant. Sowohl mit dem alten als auch mit dem neuen System sollten zwei Vertriebsmitarbeiterinnen eine vorgegebene Menge von Aufträgen bearbeiten. Diese Aufträge hätten von einer dritten Person ausgewählt werden sollen.

Durch Probleme bei der Implementierung und Integration der neuen Software konnte die Analyse leider nicht mehr durchgeführt werden.

4.2.3 Resümee der ersten Fallstudie

4.2.3.1 Erfahrungen

Folgende Erfahrungen wurden von Dr. Schröder in diesem Projekt gemacht:

- *Aeneis-Diagramme* haben sich in der Phase der Modellierung mit den Mitarbeitern *bewährt*, es gab weder Einarbeitungs- noch Verständnisprobleme. Im Kernteam wurden die Abläufe gemeinsam am Bildschirm erarbeitet. In weiterer Folge wurden dann Ausdrucke erstellt und mit allen Betroffenen abgestimmt. Mit Mitarbeitern aus dem Lagerbereich waren hier Verständnisprobleme festzustellen (kann aber auch in der Methodik liegen).
- Der *Neuigkeitsgrad der Lösung* wurde von den Betroffenen höher empfunden als von mir. Der Eindruck der Neuigkeit entstand für die betroffenen und beteiligten Mitarbeiter offensichtlich aus der erstmaligen theoretischen Auseinandersetzung mit den Arbeitsabläufen, im speziellen durch die Verwendung eines Modellierungswerkzeuges.
- Bei diesem Projekt wurde vorwiegend versucht, die Schwachstellen im Ist-Ablauf zu beheben, eine *Vision* für grundlegende Veränderungen gab es nur im Lagerbereich. Diese Vision stellte sich später als unter den gegebenen Umständen nicht realisierbar heraus.
- Die Verwendung der Diagramme hat deutlich zu einer „*Disziplinierung*“ *der Anforderungen* bzw. zur Beschränkung auf Wesentliches und logisch Machbares geführt. Dieser Sachverhalt wurde ganz deutlich, als in weiterer Folge Änderungsanforderungen nur mehr verbal spezifiziert wurden.
- Parallel zu den Abläufen wurden *Klassendiagramme* für Artikel, Partner (Kunden, Lieferanten, sonstige) und Konditionen in der Entwicklungsumgebung entwickelt. Grund: Diese Entwicklungsumgebung verfügt über einen Klasseneditor, der zur Software-Entwicklung verwendet wird. Dieser Klasseneditor ist mächtiger als in AENEIS, außerdem war bereits eine Klassenbibliothek in der Entwicklungsumgebung verfügbar.
- Als logischer Schritt hätten die *Aeneis-Prozesse* direkt mit der Entwicklungsumgebung *umgesetzt* werden sollen. Bei der Entwicklung der Software wurden die Inhalte des Modells aber nicht berücksichtigt, die Software für XXX wurde nach dem Ermessen der Entwickler und den Gegebenheiten der Entwicklungsumgebungen gestaltet. Faktisch

wurde dann versucht, den Benutzern die Prozesse der verwendeten Entwicklungsumgebung „schmackhaft“ zu machen, was natürlich zu einer Verwirrung bei den Benutzern geführt hat, da diese ja vorher bei der Modellierung integriert waren.

- Es wurden dann *Abläufe aus einem anderen Projekt* des Softwareunternehmens auf das XXX-Objektmodell *angepaßt* – mit erheblichem Aufwand und Zeitbedarf, das Ergebnis waren dann prototypische Abläufe, die außerordentlich fehlerhaft waren – zumindest zu Beginn.
- Um diese Prototypen den XXX-Anforderungen anzupassen, wurden die *Aeneis-Diagramme in verbale Beschreibungen* „übersetzt“ und den Entwicklern zur Verfügung gestellt. Das war aufwendig und durch den „verbalen Puffer“ konnten jetzt beliebige andere Anforderungen definiert werden und der Bezug zum Modell ging verloren.
- Ein weiteres Problem war ein Spezifikum der Firma XXX: Die *Bestpreisfindung*. Für verschiedene Kunden (oder Kundengruppen, wie Genossenschaften, Großhändler, Baumärkte etc.) gibt es verschiedene Preisschemen. XXX muß im Falle eines Auftrages jedem Kunden die jeweils für ihn beste Kondition/Preise ermitteln und verrechnen. Da der einzelne Kunde eine gute Übersicht über die für ihn geltenden Konditionen hat, konnte der Kunde Fehler sofort feststellen, nicht aber XXX. Deshalb sollte die Bestpreisfindung in der neuen Software automatisch erfolgen, die Implementierung erwies sich aber als sehr aufwendig.
- Trotz dieser Zwischenphase waren die implementierten Abläufe *weitgehend ähnlich* den modellierten Abläufen in AENEIS.

4.2.3.2 Schlußfolgerungen

Aus den oben genannten Erkenntnissen zogen wir folgende Schlußfolgerungen:

- Die Prozeßmodellierung mit den Benutzern hat in erster Instanz wesentliche Vorteile gebracht. Zu einem guten Teil wird das durch die anschauliche AENEIS-Notation erreicht, die sich von anderen Modellierungswerkzeugen unterscheidet – EPK ist für Betroffene beispielsweise eher schwer zugänglich.
- Für den Aufbau des Strukturmodelles (Formulare, Schnittstellen) sollte mehr Zeit in die Gestaltung investiert werden, hier kommt auch die Abhängigkeit von der Softwareumge-

bung zum Tragen. Eine Verstärkung der Integration von Charakteristika der Software in die Modellierung wäre wünschenswert.

- Das Vorhandensein von *Referenzmodellen* ist sowohl aus Prozeß, als auch aus Software-Sicht wünschenswert:
  - *Referenzprozesse*: Diese können als Ausgangspunkt ein wesentliches Hilfsmittel bei der Gestaltung darstellen, zumindest kann man Vergleiche mit einer „idealen Firma“ anstellen um Abweichungen festzustellen und gegebenenfalls zu bewerten.
  - *Referenzmodell der Software*: Wenn das Softwaresystem auf Objektorientierung beruht, sollte das Objektmodell in AENEIS zur Verfügung stehen. Andernfalls sind relevante und abbildbare Informationen über das Softwaresystem zu identifizieren und abzubilden. Beispiel: Relationen, Eingabemasken, ...
- Schließlich sollte in *AENEIS* folgende *Möglichkeiten* bestehen:
  - *Benutzer-Schnittstellen*, die über die EA-Schnittstellen (Daten!) der Aktivitäten hinaus gehen, sollten im Modell integriert werden können.
  - Entsprechende *Benutzerschnittstellen des Software- Zielsystems* sollten bei der Modellierung der Geschäftsprozesse verwendet werden können, beziehungsweise sollten Änderungsanforderungen protokolliert werden können.
- *Rasches Prototyping* sollte bereits zu einem frühen Zeitpunkt möglich sein, um die Gestaltungsphase befruchten zu können und die Qualität des Ergebnisses von Anforderungsanalyse und Gestaltung zu erhöhen.

Nach Inbetriebnahme des Systems hat XXX das AENEIS-Modell verwendet, um es zur ISO-Zertifizierung zu nutzen (ISO-9000).

4.3 Zweite Fallstudie: Firma YYY

Aufgrund des Abbruchs der Fallstudie mit der Firma XXX, war es notwendig, eine zweite Fallstudie zu beginnen. Erkenntnisse und Schlußfolgerungen der ersten Fallstudie wurden dabei berücksichtigt. Im Rahmen dieser Fallstudie wurden die folgenden Schritte des Vorgehensmodells, das in 1.4 beschrieben wurde, durchlaufen:

- Zielfindung und Erfassung der Eingangsparameter (1.4.2.4.1).
- Business Engineering, Soll-Modellierung von Geschäftsprozessen und Software (1.4.2.4.3).
- Bearbeiten und Verfeinern der Schnittstellen zur Software. Grundsätzlich die erste Phase des Schrittes der in (1.4.2.4.4) beschrieben wurde.

4.4 Ausgangssituation

4.4.1 Szenario

Die Firma *YYY* [YYY] ist ein etablierter Anbieter von Hardware mit Sitz in Linz. Aufgrund der Notwendigkeit einer Verbesserung der Softwareunterstützung des Vertriebes wurde nach einiger Marktanalyse (u.a. SAP) die CI-Software (Customer Interaction) Applix eingesetzt.

Die bis zu diesem Zeitpunkt bestehende Eigenentwicklung unter dem Tool Progress sollte nicht weiter entwickelt (technische Grenzen erreicht), aber auch nicht abgelöst werden (zu großer Aufwand). Die technischen Grenzen waren vor allem durch die mangelhafte Unterstützung von Vertriebsprozessen erreicht. Daraus ergaben sich eindeutige Anforderungen an die neue Lösung:

- Die Entwicklung einer befriedigenden Unterstützung für den Verkauf und das Management der Kundenbeziehungen sollte möglich sein, künftig *verbessert* und *weiter angepaßt* werden können.
- Die *bestehenden Daten* sollten weiter verwendet werden können.

Aufgrund der positiven Erfahrungen mit Applix entschloß sich die Geschäftsleitung der Firma YYY 1998 zur Gründung der Tochterfirma ZZZ Datenverarbeitung [ZZZ]. Ziel dieses Unternehmens ist unter anderem das Anbieten von Unternehmenslösungen auf Basis von Applix. ZZZ ist inzwischen als Applix Competence Center zertifiziert.

4.4.2 Das Software Produkt: Applix

Applix [Applix] ist ein amerikanisches Produkt und kann grundsätzlich als CRM-Software (Customer Relationship Management) kategorisiert werden. Es stellt eine Front-Office Management-Lösung dar, der Kunde wird in den Mittelpunkt gestellt.

Applix bietet die Möglichkeit einer ganzheitlichen Kundenbetrachtung; alle Daten die mit einem Kunden in Verbindung stehen (Aufträge, Abrechnungen, Anforderungen) können verwaltet und zugänglich gemacht werden. Dadurch soll das Weiterreichen von Kunden im Falle von Anfragen vermieden werden und eine wesentliche Erhöhung der Kundenzufriedenheit erreicht werden.

Dabei werden „neue“ Daten die durch die Einführung des Kundenbeziehungsmanagement entstehen, mit bereits bestehenden Daten verknüpft. Daraus ergibt sich eine abteilungsübergreifende Informationsstruktur, Vertriebsmitarbeiter können nicht nur Aufträge sondern auch historische Daten (Reklamationen, Anfragen, Kontostände, etc.) abrufen und so besser auf den Kunden eingehen.

Die Problematik der Übernahme von Altlasten (Datenbeständen) wird dadurch maßgeblich entschärft. Applix kann daher als eine Art Sanierungsvariante gesehen werden, da die Datenbestände übernommen werden können und trotzdem in relativ kurzer Zeit die Softwareunterstützung neue Geschäftsprozesse ermöglicht wird. Eine Besonderheit stellt der branchenweit (Customer Interaction Software-Lösungen) einzige voll Java-taugliche Client dar. Weitere Information: [Applix], [ZZZ].

Grundsätzlich ist Applix in die Module *Sales*, *Service* und *Helpdesk* unterteilt:

- *Applix-Sales*: Das Sales-Modul beschäftigt sich grundsätzlich mit der Abwicklung von Verkaufsaktivitäten. Es stellt eine Wissensdatenbank für den Verkäufer sowie eine Marketing-Enzyklopädie für die Verkaufsleitung dar. Die Unterstützung des Verkaufs erfolgt durch verbessertes Kundenmanagement, Kontaktmanagement, Bereitstellung von Gesprächsleitfäden für den Verkauf, Integration von Außendienstmitarbeitern (JAVA-Client) und mannigfaltige Möglichkeiten zur Datenanalyse. Die Datenanalyse wird durch eine integrierte Version des Tools Crystal Reports von Seagate [Seagate] unterstützt, das die verschiedensten Daten auswerten und visualisieren kann.

- *Applix-Service*: Das Service-Modul verfügt über automatische Zuweisung von von Aufträgen, Termin- und Auslastungskontrollen, Protokollierung von Kundenzufriedenheit und –bedürfnissen in einer Wissensdatenbank und dadurch eine mögliche Nutzung dieser Daten für den Verkauf (Übergang zu *Applix-Sales* durch gemeinsame Datenbasis oder Nutzung der Informationen in der Produktweiterentwicklung).
- *Applix-Helpdesk*: Das Helpdesk-Modul verfügt über eine Ressourcenverwaltung. Produktkonfigurationen sind abgespeichert (Daten aus dem Verkauf und Kundeninformationen), zur Fehlerbehebung steht ein Lösungsdatenbaum zur Verfügung. Grundsätzlich werden jene Funktionen bereitgestellt, die in modernen Call-Centern üblich und notwendig sind.

In unserer Fallstudie haben wir uns mit dem Modul *Applix-Sales* beschäftigt.

4.4.3 Vorgeschichte

In 10 bis 15 Mann-Tagen war von einem Team ein AENEIS-Modell des gesamten Unternehmens angefertigt worden. Das Team bestand aus den Geschäftsführern der Firma YYY und einem externen Berater. Die Geschäftsführer sind aktiv im Tagesgeschäft tätig.

Ziele der Modellierungsphase waren die Gewinnung eines Überblicks, das Aufspüren von Defiziten und die Gewinnung von Erkenntnissen über neue Prozesse. Ausgangspunkt war, daß Applix im Vertrieb eingesetzt werden sollte. Da aber bei der Geschäftsleitung der Firma YYY das Gefühl bestand, daß auch in anderen Bereichen Abläufe mit Verbesserungspotential aufzufinden sind, wurde eine umfassende Ist-Analyse begonnen.

Während der Ist-Analyse wurden Schwachstellen aufgedeckt und als Verbesserungsziele in den Zielbaum (in AENEIS) übertragen. Ist-Analyse und Zielfindung haben gezeigt, daß die größten Verbesserungspotentiale im Bereich Kundengewinnung und Kundenauftragsabwicklung liegen. Beispielsweise konnte durch eine Analyse der Kundenumsätze festgestellt werden, daß nur ca. 20 Prozent der Kunden aktiv sind, das heißt in letzter Zeit gekauft haben. Um die Kundenbindung zu erhöhen und passive Kunden zu aktivieren, wurde die Einführung eines Aktivvertriebs-Prozesses beschlossen.

Aktivvertrieb ist als Gegenteil von *Passivvertrieb* zu sehen. Man wartet (passiv) nicht bis eine Anfrage eines Kunden eintrifft, sondern versucht Kundenbedürfnisse zu antizipieren und von

sich aus Angebote zu stellen (aktiv), beziehungsweise den Kunden auf seine Bedürfnisse aufmerksam zu machen.

Um den Kunden gezielt ansprechen zu können, wird eine Analyse von Bedürfnissen, Kaufgewohnheiten, Unternehmensanforderungen, oder auch mögliche Nutznießer technischer Weiterentwicklungen als potentielle Kunden betrachtet und direkt auf ihre Bedürfnisse angesprochen. Die Daten für diese Auswertung sollten aus den bestehenden Datenbanken und aus der Applix-Datenbanken kommen. Die Auswertung sollte mit einer Kunden-Produkt-Matrix erfolgen, die eine Vereinfachung eines OLAP-Systems sein soll.

Bei einem OLAP-System (Online Analytical Processing) werden aus verschiedenen Datenbanken mehrdimensionale Datenstrukturen erzeugt, mit den derartig aggregierten Daten werden dann Berechnungen durchgeführt. Aufgrund der Zweidimensionalität der Kunden-Produkt-Matrix (im Gegensatz zu mehrdimensionalen Datenstrukturen in der Analyse) und der identen Aufgabenstellung kann man daher von einer Vereinfachung eines OLAP-Systems sprechen.

Aufgrund des Verhältnisses zwischen investierten Mann-Tagen und der Komplexität der Aufgabe, waren verschiedene Bereiche des Modelles unterschiedlich detailliert ausformuliert. Bereiche mit größerem Verbesserungspotential wurden genauer behandelt.

4.5 Ziele des Fallstudienprojektes

Primärziel war die Gestaltung und Detaillierung des Geschäftsprozesses *Aktivvertrieb* für die Firma YYY (intern) - unter Berücksichtigung der Zielvorgaben und der Möglichkeiten von Applix. Die Grundstruktur und notwendige Inhalte des Aktivvertriebes waren grundsätzlich gegeben.

Aus diesem Modell sollten nicht nur Informationen über die Arbeitsabläufe sondern auch über die Anforderungen extrahierbar sein, die eine Anpassung und/oder Erweiterung von Applix erfordern. Es sollte eine Art Pflichtenheft entstehen, welches zusätzlich in künftigen Applix-Projekten als Implementierungshilfe verwendet werden kann.

Gleichzeitig sollte das Modell des Aktivvertriebes in AENEIS als Referenzmodell zur Darstellung der möglichen Unterstützung des Prozesses durch Applix dienen können.

Durch die Schlußfolgerungen aus der ersten Fallstudie konnten zusätzlich mehrere Ziele identifiziert werden. Die Ziele des Fallstudienprojektes sind zusammengefaßt:

- *Gestaltung und Detaillierung des Aktiv-Vertriebsprozesses*: Vereinfachung und Konkretisierung des bestehenden Modelles in AENEIS und Verbesserung der Beschreibung der Softwareunterstützung zur späteren Umsetzung.
- *Entwicklung eines Referenzprozesses*: Grundidee ist es, einem Kunden nicht nur die softwaretechnische sonder auch die organisatorische Lösung für ein Problem anbieten zu können und damit den Verkauf von Know-how in den Vordergrund rücken zu können. In Anlehnung an die Schlußfolgerungen aus der ersten Fallstudie sollte der Aktivvertrieb als fragmentarisches Referenzmodell betrachtet werden, in dem versucht wird, sowohl die Arbeitsabläufe als auch die Unterstützung durch die Software abzubilden.
- *Verbesserung der Prototypingmöglichkeiten*: Um zu einem frühen Zeitpunkt des Projektes Prototypingtechniken anwenden zu können, sollte das Modellierungswerkzeug nach Integrationsmöglichkeiten untersucht werden.
- *Verfeinerung des Zielsystems*: Bei der ursprünglichen Modellierung waren die Ziele großteils nur als Liste und schwach strukturiert vorhanden. Sie sollten übersichtlicher strukturiert und hierarchisiert werden.

4.6 Vorgehensweise

In Anlehnung an das Vorgehen in Abschnitt 1.4 läßt sich die Vorgehensweise nun folgendermaßen darstellen:

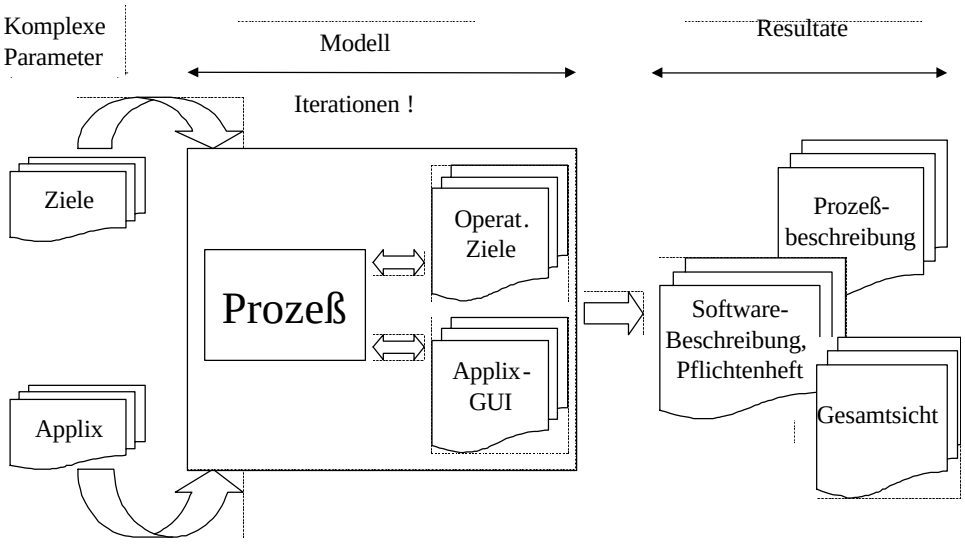

Abb. 17 Vorgehen bei der Firma YYY

Eingabe in den Modellierungsprozeß sind die vorgegebenen Ziele (im speziellen des Aktiv-Vertriebsprozesses, siehe **Fehler! Verweisquelle konnte nicht gefunden werden.**), die in einem Strategiepapier festgelegt wurden, und die Software Applix [Applix].

Im Modell sollten die Ziele zugeordnet werden und die Integration der Software in das Modell verbessert werden. Die Zuordnung von Zielen zu Geschäftsprozessen, Aktivitäten und Arbeitsschritten kann als Maß der Wirksamkeit von verschieden gestalteten Geschäftsprozessen mit dem selben Ziel angesehen werden. So kann die zu konkretisierende Lösung aus einem Pool von Lösungsideen ausgewählt werden, wobei die Überdeckung von Zielen und Tätigkeiten sowie der Aufwand der durch den Geschäftsprozeß verursacht wird zur Bewertung herangezogen werden können. Konzepte zur Wirtschaftlichkeitsbewertung von Geschäftsprozessen sind allerdings nicht Teil dieser Diplomarbeit und werden daher nicht weiter behandelt.

Als Resultate waren operable und wiederverwendbare Dokumente gewünscht. Änderungen sollten mit Hilfe der automatischen Generierung von Berichten durch den Berichtsgenerator des Unternehmensmodellierungswerkzeuges mit geringem Aufwand möglich sein.

Ein Tagebuch des Fallstudienprojektes mit der Firma YYY ist im Anhang **Fehler! Verweisquelle konnte nicht gefunden werden.** zu finden.

4.6.1 Darstellung des Modellinhaltes

Aufgrund der Erfahrungen und Schlußfolgerungen aus der ersten Fallstudie (siehe 4.2) und der vorgegebenen Ziele, wurden die ersten Anstrengungen in die Art der visuellen Aufbereitung des Modells gesteckt. Konkret sollten Berichte – also Repräsentationen des Modellinhaltes - geschaffen werden, welche als Kommunikationsunterstützung des Diskurses über Inhalte des Modells während der Modellentwicklung oder –anpassung dienen können, wobei die Kommunikation sowohl innerhalb des BE-Teams, als auch mit Betroffenen unterstützt werden soll.

Grundsätzlich werden bei AENEIS 4.0 folgende Berichtsformate mitgeliefert:

Berichtsformat	*Bezeichnung des Berichtes*	*Ziel-Plattform des Berichtes*
AENEIS.ARL	Kostentreiber	Microsoft Word
	Markierte Einträge (Details)	Microsoft Word
Standard-Berichte	Markierte Einträge (Standard)	Textfenster
	Markierte Einträge (Winhelp)	Microsoft Word
	OQL-Beispielbericht	Textfenster
	Prozeßbericht ohne Bilder	Microsoft Word
	Prozeßbericht/detail. Nummern	Microsoft Word
	Standardbericht	Microsoft Word
	Standardbericht (Windows-Hilfe)	Microsoft Word
	Verwendungsnachweis TP	Microsoft Excel
HTML.ARL	Dokumentverzeichnis	Alle HTML-Berichte werden im Modus Textfenster generiert und in HTML im Verzeichnis des jeweiligen Modells abgelegt. Grafiken (Prozeßdiagramme, Organigramm, etc.) werden als Grafikdatei abgelegt (Bsp. JPG).
	Einzelseite	
HTML-Berichte	Gesamtindex	
	Gesamtmodell	
	Hauptseite	
	Hauptseite, erweitert	
	Mit Unterpunkten	
	Vorgangsstati	
HTML_001.ARL	Wie HTML.ARL, in englisch	Wie HTML.ARL
	...	
QMH1.ARL und QMSYSTEM.ARL	Verschiedene Qualitätsmanagementhandbücher. Hier nicht weiter wichtig	Microsoft Word
TEMPLATE.ARL	HTML(markierte Einträge)	Lotus AmiPro 3.0

Abb. 18 AENEIS Berichtsformate

Die Dateibezeichnung ARL steht für *AENEIS Report Language*. Die Ziel-Plattform des Berichtes repräsentiert die Applikation/Schnittstelle des jeweiligen Berichtes.

Um die Ziele des Projektes zu erreichen wurden drei Berichte ausgewählt, die jeweils unterschiedliche Sichten auf die Daten aus dem Modell darstellen:

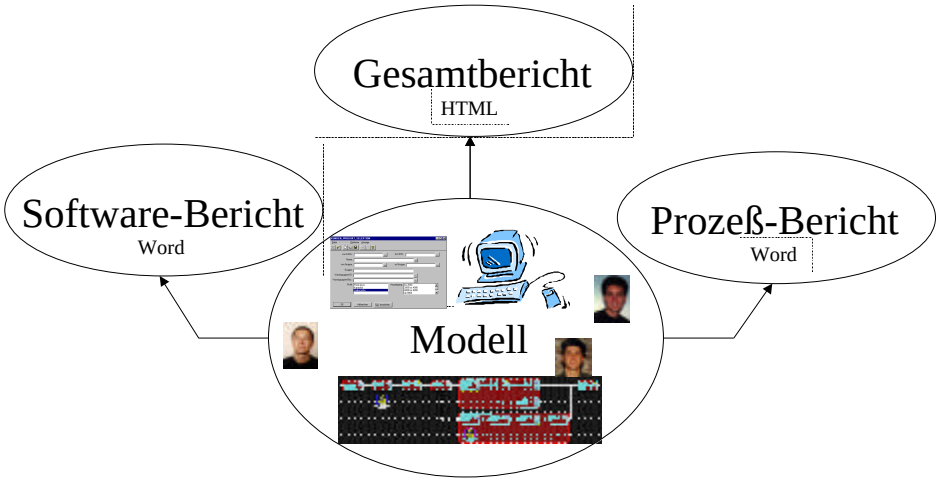

Abb. 19 Berichtsstrukturen

- *Der Gesamtbericht*: Aufgrund der Mehrdimensionalität des Modells bietet sich die Möglichkeit der Generierung von HTML-Strukturen an. Speziell der Gesamtbericht scheint auf andere Art und Weise kaum lesbar zu gestalten zu sein. Dazu wurden die Berichte „Hauptseite“ und „Gesamtmodell“ des Berichtsformats HTML.ARL aus der oben angeführten Tabelle verwendet. Dieser Bericht enthält alle im Modell gespeicherten Informationen: Ablaufbeschreibungen, Diagramme (Informationsystem, Organigramm, ..), Ziele, etc. Um das Ergebnis der Generierung zu verbessern, wurden mit Hilfe der Sprache des Berichtsgenerators kleine Änderungen vorgenommen. Das Ergebnis ist auszugsweise im Anhang abgebildet (**Fehler! Verweisquelle konnte nicht gefunden werden.**). Neben der Integration der Schnittstellen wurden zum Beispiel folgende Änderungen in der Berichtssprache durchgeführt:
  - Einfügen des Firmenlogos der Firma YYY
  - Entfernen von HTML-Elementen wie Zeilenvorschüben, Linien etc., um eine kompaktere Repräsentation zu erreichen.
  - Anpassung der Textformatierung zur besseren Lesbarkeit.

- *Der Software-Bericht*: Dieser Bericht soll möglichst viele Informationen zur Entwicklung des Informationssystems enthalten. Dazu wurde der Bericht „Markierte Einträge (Details)“ des Berichtsformats AENEIS.ARL verwendet, wobei der Unterpunkt Informationssystem im Modell markiert wurde. Dieser Bericht enthält die Eingabemasken und Informationen über das Softwaresystem, (**Fehler! Verweisquelle konnte nicht gefunden werden.**).
- *Der Prozeß-Bericht*: Dieser Bericht soll eine verbale Beschreibung der Arbeitsabläufe inklusive Hinweisen auf die verwendeten Schnittstellen beinhalten. Dazu wurde der Bericht „Prozeßbericht ohne Bilder “ des Berichtsformats AENEIS.ARL gewählt. Um diesen Bericht zu generieren wurde der Prozeß Aktivvertrieb im Modell markiert. Bilder (Prozeßdiagramme) wurden nicht verwendet, da die Darstellung von Prozeßdiagrammen in Textdokumenten aufgrund ihrer Größe sehr problematisch ist: Diagramme werden zerteilt und oft entstehen durch die automatische Zerteilung der Diagramme durch den Berichtsgenerator Ausschnitte, die wenig aussagekräftige sind. Der Prozeßbericht enthält die Beschreibung der Geschäftsprozesse in strukturierter Humansprache und nur andeutungsweise Informationen über Aufgabenträger, Softwaresystem etc. (**Fehler! Verweisquelle konnte nicht gefunden werden.**).

4.6.2 Modellierung

Prozeßmanager ist der Vertriebsleiter, diese Rolle wurde zur Verankerung des neuen Geschäftsprozesse im Vertrieb geschaffen. Bei der Modellierung des Geschäftsprozesses wurde eine Segmentierung anhand des jeweiligen Aufgabenträgers durchgeführt, wie in Abb. 22 leicht erkennbar ist. Nachfolgend seien einige grundsätzliche Strategien erläutert, die bei der Modellierung angewandt wurden.

4.6.2.1 Abstraktion

Als erstes wurde eine sinnvolle Abstraktion gesucht, um eine effiziente Erfüllung der Ziele zu erreichen. Die gewählte Abstraktion kann so dargestellt werden:

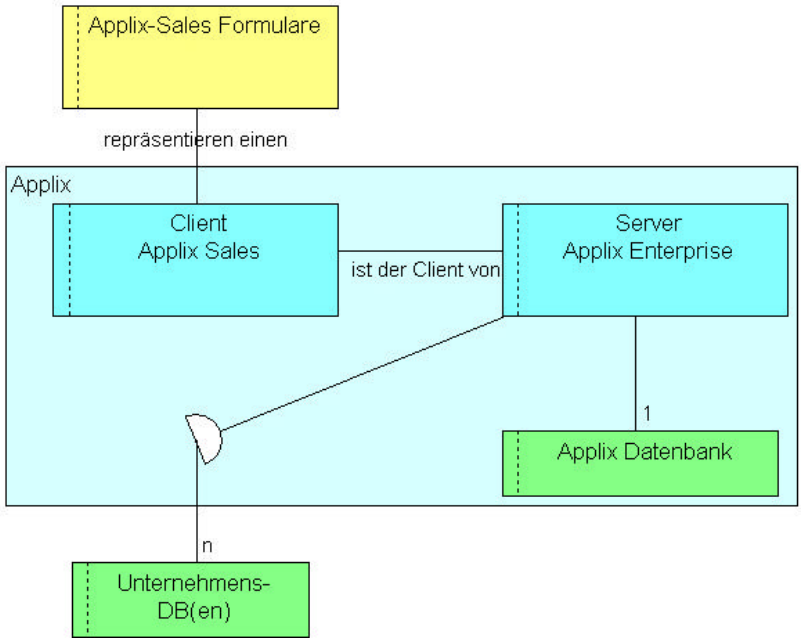

Abb. 20 Abstraktion zur Modellierung des Aktiv-Vertriebs

Wichtig ist die *Modellierung des Geschäftsprozesses*. Die Aktivitäten und Arbeitsschritte des Geschäftsprozesses Aktiv-Vertrieb (Definitionen wie in AENEIS, siehe 3.4.2) werden mit den *Formularen* (eine Menge von Eingabemasken; Formulare ist ein Menüpunkt im Hauptmenü von Applix-Sales) eines Applix-Sales Clients abgearbeitet. Diese Formulare sind also der gemeinsame Nenner von Software und Geschäftsprozeß.

Diese Eingabemasken gehören zum jeweiligen Client, in diesem Fall ein Applix-Sales Client. Dieser bildet (abstrahiert betrachtet) gemeinsam mit dem Applix Enterprise Server und der Applix-Datenbank das Applix System.

Der Applix-Sales Client leitet alle Transaktionen an den *Applix Enterprise Server* weiter, der wiederum für Transparenz bezüglich des Ortes und Formates des jeweiligen Datums sorgt. Durch die Integration der Schnittstellen (genaueres siehe 5.3.6) können diese Daten jedoch mit den dargestellten Daten in den verschiedenen Eingabemasken abgeleitet werden.

In der *Applix-Datenbank* werden alle für den Workflow notwendigen Daten gespeichert, sowie alle Daten, die bisher nicht gespeichert waren. In den *Unternehmensdatenbanken* sind alle bisherigen Daten gespeichert.

Durch diese Abstraktion kann erreicht werden, daß der Prozeß von einem konkreten Projekt unabhängig bleibt, weil:

- Die *Daten* (speziell im Fall Aktivvertrieb) nur eine relativ geringe *Varianz* aufweisen sollten, beispielsweise die Artikeldaten.
- *Zuordnung zu Datenbanken* ohnehin immer unterschiedlich ist. Die Zuordnung der Transaktionen zu den genannten Datenbanken wird bei der Konfiguration des Applix-Enterprise Servers vorgenommen.

4.6.2.2 Faktorisierung und Strukturierung

Um die Komplexität des Prozesses zu vermindern, wurde zuerst eine Faktorisierung (Zusammenfassung anhand von gemeinsamen Eigenschaften) durchgeführt.

Beispiel Startaktivitäten: Im ursprünglichen Modell waren mehrere verschiedene Startaktivitäten eingeplant, welche als Vorschläge oder Produktideen betrachtet werden können, die in weiterer Folge zu einer Kontaktaufnahme mit einem Kunden führen können.

- Information durch einen Lieferanten
- Informationen durch das Koordinationsteam Produktideen
- Information durch Technologierecherche
- Information durch Kundenanfragen

Daraus wurde *eine Startaktivität* modelliert - mit der Bezeichnung „Produktidee erfassen“. Die Art der Informationsquelle (Kunde, Recherche, Lieferant,..) spielt aus der Sicht der Software keine Rolle, Ziel ist die Erfassung einer Produktidee und die Initiierung der Weiterbearbeitung (siehe **Fehler! Verweisquelle konnte nicht gefunden werden.**). Die Faktorisierung wurde sozusagen anhand der Betrachtung der notwendigen Softwareunterstützung vorgenommen.

Ein anderes Beispiel für Strukturierung wäre die Zusammenfassung der Prüfung einer Produktidee (siehe HTML-Seite **Fehler! Verweisquelle konnte nicht gefunden werden.**), die im ursprünglichen Modell noch in eine technische und wirtschaftliche Prüfung unterteilt war, aber von den selben Personen durchgeführt wurde. Da diese Prüfung im Rahmen von Meetings erfolgen soll, ist eine Zusammenfassung daher auch im Modell sinnvoll. Die Prüfung der Produktidee wurde daher im Soll-Modell als eine Aktivität dargestellt.

4.7 Zusammenfassung des Kapitels

In diesem Kapitel wurden die beiden Fallstudien, die im Rahmen dieser Diplomarbeit durchgeführt wurden, beschrieben. Dabei wurden jeweils die Ausgangsituation, die Vorgeschichte, durchgeführte Arbeiten sowie Erkenntnisse und Schlußfolgerungen beschrieben.

Die genannten Erkenntnisse und Schlußfolgerungen beziehen sich konkret auf die Vorgehensweise und die Verwendung des Modellierungswerkzeuges.

Bei der ersten Fallstudie (Firma XXX) wurde auch beschrieben, warum ein Abbruch notwendig wurde. Grundsätzlicher Inhalt wäre ein Vergleich von Geschäftsprozessen und Unternehmenssoftwarelösungen im Soll- und Ist-Zustand gewesen.

Bei der zweiten Fallstudie (Firma YYY/ZZZ) wurde die Modellierung eines bisher nicht realisierten Prozesses durchgeführt. Zusätzlich wurde durch verstärkte Anpassung des Modellierungswerkzeuges die Integration der Software in das Modell verbessert, um die Aspekte der Gemeinsamkeit zwischen Geschäftsprozeß und Unternehmenssoftware deutlicher darstellen zu können.

Grundsätzlich muß natürlich bemerkt werden, daß aufgrund des Fallstudiencharakters des Projektes nicht alle Möglichkeiten im Modell ausgeschöpft wurden. Die Detailliertheit verschiedener Informationen sollte noch in einigen Punkten vergrößert werden.

Beispiel: Die Darstellung der Informationen über Eingabeelemente (siehe 5.3.8) sollte noch verfeinert werden um als Unterstützung der Implementierung herangezogen werden zu können. Die Machbarkeit der Hinterlegung dieser Art von Informationen ist jedoch in oben genanntem Beispiel aufgezeigt.

Trotz der doch unterschiedlichen Einsatzgebiete der beiden Fallstudien konnten in beiden Erkenntnisse über Methodik und Nutzung des Werkzeuges gewonnen werden.

5 Analyse und Erkenntnisse

5.1 Ziele des Kapitels

Hier sollen nun einige Erkenntnisse über die Lösungsidee und die damit gewählte Strategie aufbereitet werden. Der Schwerpunkt liegt auf den Erkenntnissen über die Anpassung des Modells sowie der Ausgabe und Darstellung des Modellinhaltes zur Unterstützung des Vorgehens bei der Entwicklung und Gestaltung von Geschäftsprozessen und Unternehmenssoftware.

5.2 Vorgehensmodell

Erkenntnisse bezüglich des Business Engineering Konzeptes:

- Die Zusammenarbeit zwischen einem Betriebswirt und Informatiker erwies sich als anfänglich schwierig, da stark unterschiedliche Begriffswelten und Denkstrategien bestanden. Nach einiger Zeit wandelte sich dieser Umstand jedoch zu einem Vorteil: Für den einzelnen nur schwer lösbare Problemstellungen konnten im Team gelöst werden. Ein Bestätigung für die folgende Aussage: Laut Becker [Becker99] sind derartige Projekte generell nur von interdisziplinär zusammengesetzten Teams zu lösen.
- Das permanente Betrachten der Software (Schnittstellen, siehe 5.3.6) im Kontext des Geschäftsprozesses steigert die Qualität der Anforderungsanalyse und auch der Prozeßgestaltung. Durch die Wahl einer geeignete Darstellung (HTML) kann man schon sehr früh wichtige Informationen über das Soll-Modell gewinnen. Natürlich sowohl bezüglich der Geschäftsprozesse als auch deren Softwareunterstützung. Konkret wurde das HTML-Modell mit einem Beamer an die Wand projiziert, so konnten mehrere Personen über verschiedene Gesichtspunkte diskutieren, eine Person (aus unserem Team) hatte die Aufgabe des Moderators am PC zu übernehmen. Das gemeinsame Betrachten von Informationen auf einem Bildschirm ist bei einer Gruppe, die mehr als zwei Personen umfaßt, mehr als unangenehm. Ein Nachteil dieser Methode ist sicher der Bedarf eines ruhigen abgeschlossenen Raums, mit angemessener Infrastruktur (Beamer, unter Umständen Leinwand, ...).

5.3 Modellierung

5.3.1 Ist- und Soll-Modellierung

Wichtiger Punkt bei der Erstellung von Ist- und Sollmodellen ist es, betroffene Personen zu Beteiligten zu machen [Osterloh97].

Grundsätzlich gehen wir davon aus, daß es sinnvoll ist, ein Istmodell abzubilden, um aus den festgestellten Stärken und Schwächen die Ziele für die Umgestaltung ableiten zu können. Diese Ziele sind wesentliche Eingangsparameter für das Sollmodell, das dann realisiert werden soll. Daraus ergibt sich unserer Ansicht nach ein wesentlicher Zusammenhang zwischen dem Ist- und Sollmodell. Modellierung ist immer von Abstraktionen geprägt, es ist unwahrscheinlich, daß mehrere Personen die selben Abstraktionen anwenden, um Sachverhalte abzubilden. Deshalb sollte das Ist- und das Sollmodell von den selben Personen abgebildet werden, um eine gewisse „Abstraktionskonsistenz“ zu erreichen.

Dabei soll aber unbestritten bleiben, daß es sinnvoll ist, unterschiedliche, aber exakt zu bestimmende Grade der Detailliertheit zwischen Ist- und Sollmodellierung anzuwenden [Bekker99].

Beispiel: Ein Team erstellt mit Hilfe der Betroffenen (Erhebungstechnik: Interviews) ein Ist-Modell. Die Geschäftsprozesse werden aber vorwiegend strukturell abgebildet, das heißt, die tatsächlichen Zusammenhänge zwischen Aktivitäten, Arbeitsschritten etc. werden abgebildet, Details über Arbeitsschritte werden aber nicht im Modell aufgenommen. Bei der Sollmodellierung muß dann eine höhere Detailliertheit angestrebt werden um die Verbesserungen realisieren zu können.

5.3.2 Darstellung des Modellinhaltes

Um das Modell nutzen zu können ist die Art der Darstellung als kritischer Faktor zu betrachten.

Beispielsweise wurde in beiden Fallstudien-Projekten zur Softwareentwicklung eine Übersetzung des AENEIS-Modells in eine verbale Beschreibung durchgeführt. Diese Beobachtung hat uns eine neue Sicht auf das Modell vermittelt.

Dazu die verschiedenen Darstellungsarten, wie sie von AENEIS angeboten werden und unsere Schlußfolgerungen.

- *Plotten von Diagrammen*
- *HTML-Darstellung*
- *Darstellung im Textformat*
- *Winhelp Dateien*

5.3.2.1 Plotten von Diagrammen

Die Ausgabe von Ablaufdiagrammen mit Hilfe eines Plotters, also großflächig (größer als A4), ist nur sehr schwer zur Kommunikation nutzbar. Es ist denkbar geplottete Diagramme als eine Art Flip-Chart zu verwenden, um den betroffenen Personen einen Einblick in das Modell zu geben. So kann die Partizipation der Beteiligten und Transparenz des Modells verbessert werden.

Vorteil: Man kann in jedem Büroraum jene Abläufe publizieren, welche die jeweiligen, dort arbeitenden Mitarbeiter betreffen.

Der Platzbedarf ist dabei ein wesentlicher *Nachteil*, ebenso wie das Fehlen von beschreibenden Texten, es wird lediglich strukturelle Information dargestellt. Man müßte die Diagramme mit Beschreibungen in Textform ergänzen.

5.3.2.2 HTML-Darstellung

Unser besonderes Augenmerk fiel besonders im zweiten Projekt auf die Aufbereitung des Modellinhaltes mit HTML. Grundsätzlich kann man zwei mögliche Szenarien unterscheiden, die einen effizienten Einsatz von HTML-Berichten unterstützen:

- *Präsentation mit Beamer*: Eine Gruppe von Personen diskutiert den mit einem Beamer dargestellten HTML-Bericht.
- *Präsentation im Intranet*: Der HTML-Bericht wird im Intranet bereitgestellt, alle Mitarbeiter des Unternehmens deren Arbeitsplatz im Intranet zugänglich ist und mit einem Browser ausgestattet ist, können den Bericht ansehen.

Ebenso wie durch Plots kann die Transparenz des Modells und die Partizipation der Beteiligten gefördert werden, wenn man den Bericht im Intranet des Unternehmens veröffentlicht. Feedback kann über e-mail oder ein BBS (Bulletin Board System) erfolgen.

Folgende *Vorteile* sprechen unserer Ansicht nach für den Einsatz von HTML-Berichten:

- Das *Navigieren* („Surfen“) auf HTML-Seiten gehört heute schon beinahe zur Allgemeinbildung: Es ist keine weitere Erklärung notwendig, das spricht für die Publikation des HTML-Berichtes im Intranet.
- Die *Größe der Grafiken* fällt nicht so ins Gewicht wie bei Texten (A4). Durch Scrolling-Mechanismen kann der interessierende Ausschnitt gewählt werden.
- Die *Visualisierung* des Modellinhaltes kann mit der Hilfe von HTML-Seiten unserer Ansicht nach am besten durchgeführt werden, weil durch die Abbildung der Objekte in der Datenbank auf einzelne HTML-Seiten die grundlegende Strukturierung des Modells vermittelt wird. Die grundsätzliche Struktur, die durch die automatische Berichtsgenerierung von AENEIS gegeben ist (auszugsweise in **Fehler! Verweisquelle konnte nicht gefunden werden.** beschrieben), ist durchaus befriedigend. Sie stellt eine Abbildung der Datenobjekte in der Datenbank dar, was grundsätzlich für die Abbildungstreue des Programmes spricht. Besonders während der Modellierung ist dieser Umstand wichtig.
- Die *Aktualität* des Entwurfs (wenn man über einen adäquat angepaßten Berichtsgenerator verfügt und keine händischen Änderungen durchführen muß) ist leichter zu gewährleisten, als bei gedruckten und geplotteten Ausgaben.
- *Grafiken und Texte* können problemlos eingebunden werden, die gemeinsame graphische und textuelle Darstellung eines Geschäftsprozesse stellt eine wesentliche Unterstützung bei der Verbesserung der Transparenz des Modells dar. Jene Benutzer, die Informationsgewinnung durch Texte bevorzugen, können ebenso angesprochen werden, wie Benutzer die leichter graphische Informationen (Ein Bild sagt mehr, als tausend Worte) aufnehmen. Diese Vereinigung der Darstellungsarten ist nur im HTML-Format möglich.

Diese *Nachteile* können beim Einsatz von HTML-Berichten auftreten:

- Ein möglicher Nachteil ist, daß überall (im Intranet) das *gesamte Modell* angeboten wird: Der Benutzer muß die für ihn wichtigen Ausschnitte selbst finden.

- Das Ausdrucken von HTML-Dokumenten ist praktisch nicht möglich.
- Das mitgelieferte Berichtsformat in AENEIS sollte auf jeweilige Anforderungen angepaßt werden. Die Anpassung ist aber relativ mühsam (AENEIS 4.0 und 4.5) da die Benutzung des Editors des Berichtsgenerators sehr gewöhnungsbedürftig und die Dokumentation unserer Ansicht nach mangelhaft ist.

Ein HTML-Bericht kann das gesamte Modell abbilden, dadurch ergibt sich eine interessante Vielseitigkeit des Einsatzes:

- *Qualitätssicherung*: Der HTML-Bericht empfiehlt sich auch in weiterer Folge zur Qualitätssicherung. Verschiedene Firmen verwenden ihr Intranet zur Publikation von Prozeß- und Ablaufbeschreibungen. Natürlich ist das als Ergänzung zu Qualitätshandbüchern (Texte) zu sehen, die ebenfalls mit AENEIS generiert werden können (siehe 4.6.1).
- *Handbuch*: Handbücher zur Erklärung der Software werden heute zum größten Teil in elektronischer Form ausgeliefert. Der Einsatz des HTML-Berichtes als Handbuch für die Benutzung eines Softwareprodukt wäre ebenso denkbar, zudem dabei eine ungewöhnliche Strategie angewandt wird: Die Beschreibung einer Anwendungssoftware im Kontext des Geschäftsprozesses.

5.3.2.3 Darstellung im Textformat

Die Darstellung im Textformat ist unserer Ansicht nach nur für Ausschnitte des Modells geeignet, speziell für solche Ausschnitte, bei denen graphische Informationen eine gewisse kritische Komplexität nicht überschreiten. Beispiele sind Ablaufbeschreibungen (siehe **Fehler! Verweisquelle konnte nicht gefunden werden.**) oder Pflichtenhefte (siehe **Fehler! Verweisquelle konnte nicht gefunden werden.**).

Positives Beispiel: Eine Eingabemaske mit den dazugehörenden Informationen (Datenelemente, ...) ist durchaus sinnvoll. Damit können wesentliche Informationen zur Implementierung dargestellt werden.

Negatives Beispiel: Graphische Darstellungen komplexer Geschäftsprozesse sind im DIN A4-Format nicht verwertbar.

5.3.2.4 Winhelp-Dateien

AENEIS kann auch Hilfedateien wie sie in Windows verwendet werden generieren. Diese Möglichkeit haben wir aber nicht in Anspruch genommen.

5.3.2.5 Resümee

Unsere Schlußfolgerung ist: Das Modell muß als Medium dienen, nicht nur als technisches Hilfsmittel. Es hat zentrale Bedeutung bei der Unterstützung der Kommunikation zwischen den beteiligten Personen. Die Möglichkeiten zur Aufbereitung der Modelldaten durch einen Berichtsgenerator sind als zentrales Qualitätselement eines Unternehmensmodellierungswerkzeuges zu betrachten.

5.3.3 Identifikation

Eines der Probleme der Neugestaltung und Einführung von Prozessen ist die Identifikation der Prozeßträger mit dem neuen Arbeitsablauf. In Unternehmensmodellierungstools ist es üblich, Aktivitäten, aber auch Stellen durch Symbole darzustellen. Dazu einige Beispiele:

Abb. 21 Mitarbeiter-, Team-, und Arbeitsplatzsymbol

Diese Art der Anwendung von Symbolik ist zwar durchaus üblich, fördert die Identifikation aber nur bedingt. Wir haben daher versucht die Repräsentation des Prozeßmodelles zu personalisieren:

Abb. 22 Steigerung der Identifikation durch Personalisierung

Wie man sieht, wurden Bilder der jeweiligen Mitarbeiter als Symbole für die Aktivitäten verwendet. In diesem Fall nicht immer die direkt im Prozeß betroffenen Personen, sondern zum

Teil unsere jeweiligen Gesprächspartner. Abb. 22 ist dem Gesamtbericht (siehe **Fehler! Verweisquelle konnte nicht gefunden werden.**) entnommen.

Beispielszenario: Bei einer Prozeßbesprechung mit einer Gruppe von Betroffenen werden deren Bilder im HTML-Bericht eingesetzt. Diese Mitarbeiter können sich so schneller mit dem Geschäftsprozeß identifizieren, sie *finden sich* wieder. Wenn man andere Personen, mit denselben Aufgaben, zu einer Besprechung des Geschäftsprozesses lädt, kann man deren Bilder einbinden. Der Änderungsaufwand ist gering die Akzeptanz kann unserer Erfahrung nach erhöht werden.

5.3.4 Zustand des Modelles

Wie bereits erwähnt können Arbeitsschritte zusammengefaßt werden. Optisch sieht das so aus:

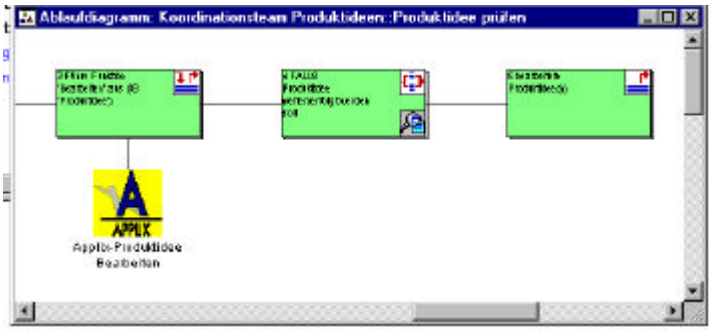

Abb. 23 Verborgene Arbeitsschritte

Der mittlere der drei Arbeitsschritte in obiger Abbildung verfügt im rechten unteren Eck über ein Lupensymbol. Das bedeute, daß er verborgene Arbeitsschritte enthält. In der unten stehenden Abbildung wurden diese Arbeitsschritte sichtbar gemacht:

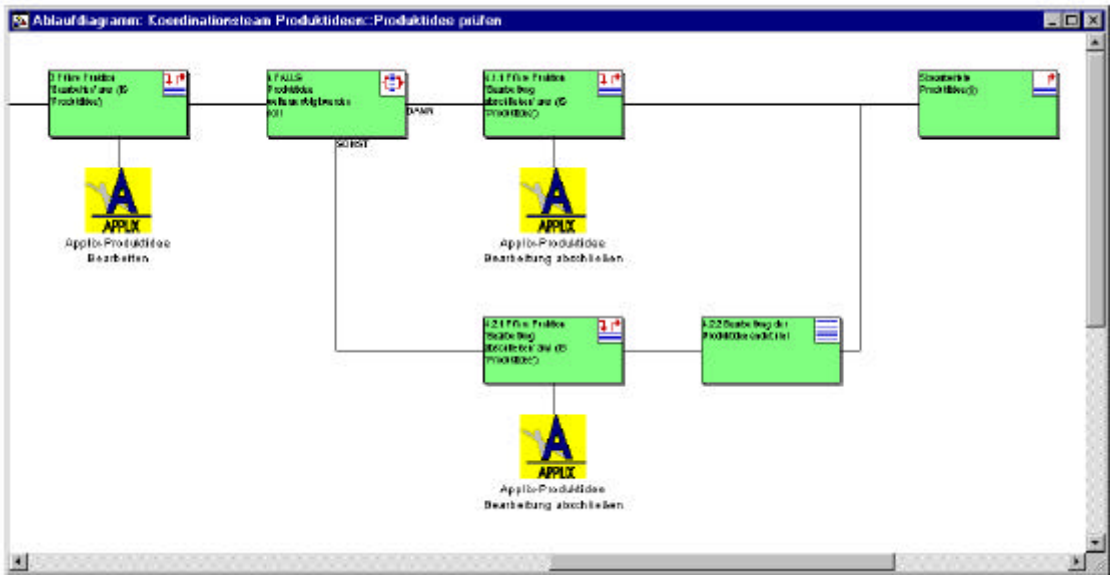

Abb. 24 Arbeitschritte detailliert

Interessant ist, daß der aktuelle Zustand (Details verborgen oder nicht) im Modell abgespeichert wird. Wenn man einen Bericht (HTML) erzeugt, werden verborgene Teile als nur ein Arbeitsschritt angezeigt, siehe Abb. 23.

Wie man in der Abb. 24 sieht, ist die dadurch entstandene Zusatzinformation nicht sehr groß, da im HTML-Bericht zusätzlich zum Ablauf-Diagramm eine textuelle Beschreibung generiert wird, welche die versteckten Informationen ohnehin enthält.

Diesen Umstand kann man sehr gut zur Verringerung der Komplexität der Darstellung verwenden, Inhalte gehen dadurch nicht verloren.

5.3.5 Einfügen von Dokumentation

Die verschiedenen Register- oder Karteikarten die dem Dialog eines Objektes in AENEIS zugeordnet sind, bieten die Möglichkeit Texte (Dokumentation) in das Modell zu integrieren. Beispielsweise wurden einige Passagen der Applix-Dokumentation in das Modell übernommen, um später die Darstellung in den verschiedenen Berichten zu überprüfen.

Ausgabe im HTML-Format: Der Applix-Text im HTML-Format (**Fehler! Verweisquelle konnte nicht gefunden werden.**).

Ausgabe im Text-Format: Der Applix-Text im Word-Format (**Fehler! Verweisquelle konnte nicht gefunden werden.**).

Die Übernahme ist prinzipiell problemlos, allerdings muß auf die Formatierungen geachtet werden. Speziell für die Erstellung eines Pflichtenheftes kann diese Möglichkeit genutzt werden.

5.3.6 Integration der Schnittstellen

Die Repräsentation von Arbeitsschritten, bei denen der Aufgabenträger eine Interaktion mit einem Softwaresystem durchführt, ist in der Standarddarstellung wenig aussagekräftig. Zur Gestaltung des Informationssystems notwendige Informationen werden im Normalfall nur auf sehr abstrakte Art und Weise dargestellt (Beispiel: Kundendaten eingeben); deshalb wurden bei diesen Arbeitsschritten die aufgerufenen Applix-Schnittstellen in das Modell integriert.

Insbesondere eröffnet sich dadurch die Möglichkeit von Schnittstellenprototyping: Man kann mit einem beliebigen Editor (Beispiel: Visual Basic for Applications) Schnittstellen definieren

und im Geschäftsprozeßmodell verankern. *Walkthroughs* können damit erheblich unterstützt werden.

Die Integration erfolgte durch Angabe des Pfades und Namens der Grafikdatei, in der die betreffende Schnittstelle abgespeichert war.

Realisierung im Modell: Im Modell wurden die Filenamen von Screenshots abgespeichert.

Ausgabe im HTML-Format: Jene Seiten, welche Informationen über Applix-Formulare enthalten (siehe **Fehler! Verweisquelle konnte nicht gefunden werden.**), sind direkt aus der Ablaufbeschreibung (siehe **Fehler! Verweisquelle konnte nicht gefunden werden.**) heraus anwählbar.

Ausgabe im Text-Format: Als Beispiel kann die Beschreibung der Software in **Fehler! Verweisquelle konnte nicht gefunden werden.** ff. betrachtet werde.

5.3.7 Hierarchisierung von Schnittstellen

Ein Problem der Integration der Schnittstellen war die große Menge an versteckten Eingabeelementen, da praktische jedes Applix-Formular über sogenannte Karteikarten verfügte. Wenn nur ein Screenshot einer Eingabemaske die über Karteikarten verfügt verwendet wird, gehen wesentliche Informationen verloren.

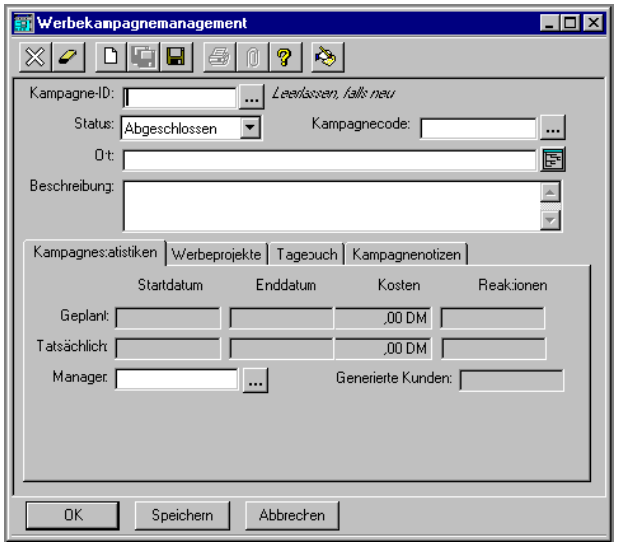

Abb. 25 Eingabemaske mit Karteikarten: Applix Werbekampagnemanagement

Bei obigem Beispiel sind die Inhalte der Karteikarten Werbeprojekte, Tagebuch und Kampagnenotizen verborgen. Um alle Informationen die in dieser Maske repräsentiert und bearbeitet werden darzustellen, müssen vier Screenshots verwendet werden.

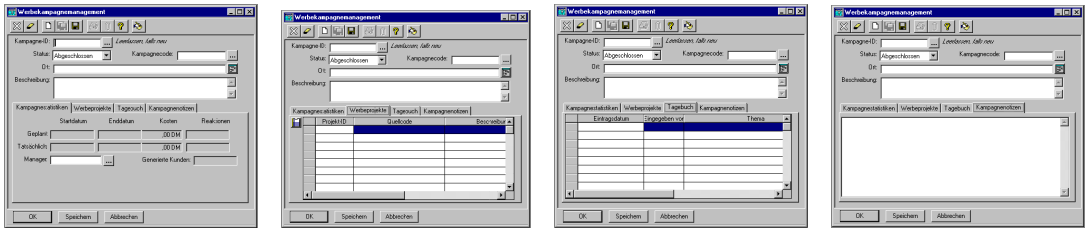

Abb. 26 Inhalte aller Karteikarten von Applix Werbekampagnemanagement

Die Daten des Dialoges stehen also viermal in einem unterschiedlichen Kontext – zu den jeweiligen Daten der aktiven Karteikarte. Um diesen Sachverhalt im Modellierungswerkzeug darstellen zu können, wurde eine Hierarchie eingeführt:

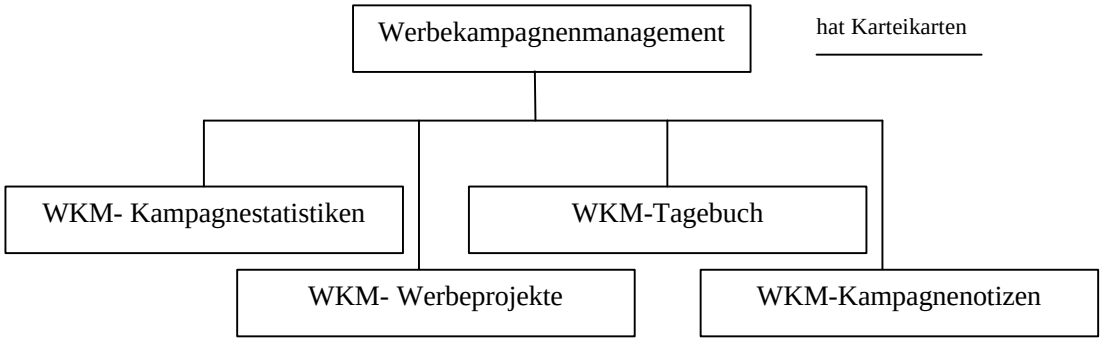

Abb. 27 Hierarchie von Eingabemasken

Realisierung im AENEIS-Modell: AENEIS speichert die Formulare prinzipiell als Liste ab, um die strukturelle Information über die logische Hierarchie darstellen zu können, wurden generische Referenzlisten (siehe 3.4.2.3) verwendet. Das Formular Werbekampagnemanagement verfügt über eine generische Referenzliste mit der Bezeichnung Karteikarte(n) welche die Beziehung „*hat Karteikarte*(*n*)“ ausdrückt. Die Karteikarte Werbekampagnenmanagement ist also fünfmal abgespeichert, um alle relevanten Informationen sichtbar zu machen, die Hierarchie wird durch generische Referenzlisten dargestellt.

Um die Beziehung im Namen zu kennzeichnen, beginnen die Bezeichnungen der Karteikarten mit Applix-WKM. WKM steht dabei für Werbekampagnemanagement.

Realisierung im HTML-Berichtsformat: Die Darstellung der Beziehung war im HTML-Format einfach, auf der Seite mit der Schnittstelle Werbekampagnenmanagement (**Fehler! Verweisquelle konnte nicht gefunden werden.**) gibt es eine Liste von Links zu allen Karteikarten. Diese sind auf den Seiten (**Fehler! Verweisquelle konnte nicht gefunden werden.** bis **Fehler! Verweisquelle konnte nicht gefunden werden.**) gespeichert und werden durch das Vorhandensein einer generische Referenzliste automatisch generiert.

Realisierung im Word Berichtsformat: Ein Beispiel für die Darstellung im Word-Bericht ist im Abschnitt **Fehler! Verweisquelle konnte nicht gefunden werden.** zu finden. Die Schnittstelle hat eine Liste aller Karteikarten.

5.3.8 Informationen über Eingabeelemente

Um die Beschreibung der Benutzerschnittstelle zu verbessern wurden dem Screenshot des Prototypen einige Daten über die Eingabeelement angefügt. Diese Daten dienen zum besseren Verständnis der Schnittstelle, außerdem kann man bei der Implementierung der Software auf diese Informationen zurückgreifen.

Ein Beispiel ist dafür das Attribut *Notizen* das zur Karteikarte *Kampagnenotizen* des Formulares *Werbekampagnenmanagement* gehört (siehe **Fehler! Verweisquelle konnte nicht gefunden werden.** im HTML-Bericht und **Fehler! Verweisquelle konnte nicht gefunden werden.** im Word-Bericht).

Notizen

- Anmerkung: Notizen zur Kampagne
- Typ: Text
- Bereich: ---

Weiters könnte zur weiteren Verfeinerung der Spezifikation ein Hinweis auf die Art des Eingabeelementes angefügt werden. Als zusätzlicher Implementierungshinweis kann aber, nach dem hier gezeigten Prinzip, auch die Datenbank, die betreffende Relation, oder das Datenobjekt eingefügt werden.

5.3.9 Namensgebung

Besonders bei der Modellierung des Softwaresystem (besonders bei den Schnittstellen) wurde auf Analogien zwischen der Notation des Softwareproduktes und der Namensgebung im Modell geachtet. Beispiele:

Applix	*AENEIS*
Beziehung zwischen einem Dialog und seinen *Karteikarten*	Referenzlistenobjekte mit der Bezeichnung *Karteikarte(n)*
Menüpunkt „*Formulare*“ (Pull-down) -> Enthält die Formulare	Statisches Diagramm „*Formulare*“ -> Enthält die Objekte die Formulare darstellen

Abb. 28 Applix- und AENEIS-Notation

5.4 Modellierungswerkzeug

Ein klarer Vorteil von AENEIS liegt in seiner konsequenten Objektorientierung. Die Objekthierarchie auf der Modellierungsoberfläche entspricht der Objekthierarchie in der Datenbank. Diese Hierarchie wird wiederum bei der Generierung von HTML-Berichten abgebildet.

Wie schon erwähnt, ist der größte Nachteil die umständliche Anpassung des Berichtsgenerators, der nach unserer Ansicht einen wesentlichen Qualitätsfaktor eines Unternehmensmodellierungswerkzeuges darstellt. Aber auch andere Modellierungswerkzeuge sind in diesem Punkt nicht wesentlich bedienungsfreundlicher.

Das ist unserer Ansicht nach gerade deshalb schade, weil gerade die Anpassung der Berichtsgeneratoren eine Vielseitigkeit der Nutzung des Modellinhaltes ergeben würde, die bisher nicht gegeben ist. Dazu Becker [Becker99] über verschiedene Zwecke von Unternehmensmodellierung (Dokumentation, Benchmarking, Zertifizierung nach ISO 900x, Customizing von integrierter Standardsoftware, ...):

> *Die Vielfalt möglicher Modellierungszwecke macht deutlich, daß nicht ein Modell alle diese Zwecke erfüllen kann (aber möglicherweise doch mehr als einen Zweck)*

Die Transparenz des Berichtsgenerators ist in AENEIS unzureichend, da kaum Dokumentation vorhanden ist und Änderungen nur sehr umständlich durchgeführt werden können.

Der Anbieter von AENEIS [Ipro] bietet das Anpassen von Berichtsformaten als Dienstleistung an.

5.5 Resümee

Leider wäre der Rahmen einer Diplomarbeit gesprengt worden, wenn die vorliegenden Konzepte, deren Machbarkeit weitgehend gezeigt wurde, zur Gänze umgesetzt worden wären. Dazu wäre vor allem notwendig gewesen:

- Eine genauere Abbildung des Applix-Systems im Modellierungswerkzeug um ein Referenzmodell der Software zu erhalten.
- Die Mitarbeit an mehreren konkreten Projekten (Applix Entwurf, Entwicklung und Einführung), um Referenzprozesse abbilden und später wiederverwenden zu können.
- Eine vollständige Anpassung des Berichtsgenerators an die vorgelegten Konzepte um die gewünschten Berichte automatisch generieren zu können.

Das beschriebene Vorgehensmodell, insbesondere die interdisziplinäre Zusammenarbeit und die gemeinsame Verwendung eines Unternehmensmodellierungswerkzeuges, hat sich unserer Ansicht nach durchaus als operabel herausgestellt. Leider kam die neueste Version von AENEIS (4.5) zu einem zu späten Zeitpunkt für unsere Fallstudien auf den Markt. Diese Version hat eine Schnittstelle zu Microsoft Visual Basic for Applications und die Möglichkeit zur Generierung von Web-Seiten mit Java-Code. Dadurch wären einige zusätzliche Veränderungen und Anpassungen möglich gewesen.

Dabei kam dem gewählten Unternehmensmodellierungswerkzeug eine wichtige Rolle zu. Durch das Bestreben, möglichst viel Nutzen aus dem Modell zu ziehen, haben sich unsere Konzepte stark verändert. Sie haben sich sozusagen in evolutionärem Sinne in Richtung des Möglichen (von Seiten des Modellierungswerkzeuges) entwickelt, wobei bemerkt werden muß, daß wir im Endeffekt mehr erreicht haben, als wir am Anfang unserer Fallstudien erwartet haben.

6 Verweise

6.1 Literatur

Bemerkung: Um Inkonsistenzen auszuschließen, wurden Referenzen auf Informationen im World Wide Web mit dem Datum (Monat und Jahr) des letzten Zugriffs versehen.

[Applix]	Homepage von Applix Inc.: http://www.applix.com, Juli 1999
[BAAN-1]	Homepage der BAAN Company: http://www.baan.com, Juli 1999
[BAAN-2]	BAAN Company: „*Standardsoftware*“, BAAN Deutschland GmbH, 1996
[Balzert96-1]	H. Balzert et al: „*Lehrbuch der Software-Technik*“ Band 1, Spektrum Akademischer Verlag 1996
[Balzert96-2]	H. Balzert et al: „*Lehrbuch der Software-Technik*“ Band 2, Spektrum Akademischer Verlag 1996
[Becker99]	J.Becker et al: „Prozessmanagement – Ein Leitfaden zur prozessorientierten Organisationsgestaltung“, Springer Verlag Berlin Heidleberg New York 1999
[BEKO99]	W. Hausmann: „*Bewertungskatalog ARIS Toolset Version 4.0*“, interne Publikation der BEKO AG, Österreich
[Blaschek97]	G. Blaschek: „*Mensch-Maschine-Kommunikation*“ erschienen in: P. Rechenberg/G. Pomberger: „*Informatik-Handbuch*“, 1. Auflage, Hanser 1997
[Blaschek99]	G. Blaschek: Skriptum zur Vorlesung „*Mensch-Maschine-Kommunikation*“, Sommersemester 1999, Johannes Kepler Universität, Abteilung für Praktische Informatik, 1999
[BOC-1]	Homepage der BOC Information Technologies Consulting GmbH Wien: http://www.boc-eu.com, Juli 1999
[BOC-2]	Schulungsunterlagen: Adonis Basisschulung, 6.8.1999
[Boehm76]	B. Boehm: „*Software engineering*“, erschienen in: „*IEEE Transactions on Computer*“, Nr. 25,, 1976
[Brenner95]	W. Brenner et al.: „*Business Process Reengineering mit Standardsoftware*“, Campus Verlag, Frankfurt am Main 1995
[Broehl93]	A. Broehl et al: „*Das V-Modell – der Standard für die Softwareentwicklung mit Praxisleitfaden*“, Oldenbourg, 1993

[Buresch97] M. Buresch etal.: „*Auswahl von Organisations-Engineering-Tools*“ erschienen in: „*Zeitschrift Führung + Organisation*“, Ausgabe 6, 1997

[Chroust97] G. Chroust: Skriptum zur Vorlesung „*Systemtechnik*“, Wintersemester 97/98, Johannes Kepler Universität Linz, Abteilung für Systemtechnik und Automation 1998

[CompW99] Computerwoche: „*Löst Oracle das marode SAP-Projekt bei der SAP ab?*“ erschienen in: Computerwoche Nr. 24, Computerwoche Verlag GmbH, 18.06.99

[CSE] Homepage der CSE Systems GmbH: http://www.csesystems.com, Juli 1999

[DIN] Homepage der Beuth Verlag GmbH Berlin: http://www.din.de, Juli 1999

[Dion93] R. Dion: „*Process Improvement and the Corporate Balance Sheet*“ erschienen in: IEEE Software, July 1993, pp. 28-35

[Dittrich97] K. Dittrich: „*Datenbanksysteme*“, erschienen in: P. Rechenberg/G. Pomberger: „*Informatik-Handbuch*“, 1. Auflage, Hanser 1997

[Duden93] H. Engesser (Hrsg.): „*Informatik*“, Duden Verlag 1993

[Faba] Homepage der FABAplus Software GmbH: http://www.fabasoft.com, Juli 1999

[Fank98] M. Fank: „*Tools zur Geschäftsprozeßorganisation*“, Vieweg Verlag 1998

[Floyd84] C. Floyd: „*A systematic look at prototyping*“ erschienen in R. Budde et. al.: „*Approaches to prototyping*“, Springer, 1984

[Floyd97] C. Floyd, H. Züllighoven: „*Softwaretechnik*“ erschienen in: P. Rechenberg/G. Pomberger: „*Informatik-Handbuch*“, 1. Auflage, Hanser 1997

[FoxMeyer95] http://ccwf.cc.utexas.edu/~cutty/foxmeyer/index.html, Juli 1999

[Frese93] E. Frese: „*Organisation*“ erschienen in: W. Lück (Hrsg.): „*Lexikon der Betriebswirtschaftslehre*“, 5. Auflage, Verlag Moderne Industrie, 1993

[Gould85] Gould et al.: „User centered design“, IBM 1985

[Gruhn97] V. Gruhn: Vorlesungsunterlagen zu „*Software Engineering – Rollenbasierte Softwareentwicklung*“ Universität Dortmund , 1997

[Grünb98] P. Grünbacher: Skriptum zur Vorlesung „*Vorgehensmodelle und Softwareentwicklungsumgebungen*“, Sommersemester 1998, Johannes Kepler Universität Linz, Abteilung für Systemtechnik und Automation 1998

[Hammer94] M. Hammer, J. Champy: „*Reengineering the Corporation*“, Allen & Unwin, 1994

[Hasted93] H. Hasted: „*Anstatt eines Vorwortes. Bemerkungen zur Interdisziplinarität*“ erschienen in: P. Schefe et. al.: „*Informatik und Philosophie*“, BI-Wissenschaftsverlag, 1993

[Hawryszkiewycz 95] I. Hawryszkiewycz: „*Systemanalyse und –design*“, Prentice Hall, 1995

[Heintz95] B. Heintz: „*Die Gesellschaft in der Maschine - Überlegungen zum Verhältnis von Informatik und Soziologie*“ erschienen in: H.-J. Kreowski: „Realität und Utopien der Informatik“, agenda Verlag, 1995

[IDS] Homepage der IDS Prof. Scheer GmbH: http://www.ids-scheer.de/, Juli 1999

[Ipro] Homepage der Ipro Consulting Gmbh: http://ipro-consulting.de, Juli 1999

[Jacobson92] Jacobson et al.: „Object Oriented Software Engineering“, Addison Wesley 1992

[Langer94] G. Langer: „*Objektorientierte Unternehmensmodellierung mit Aeneis*“, Ipro-ToolGmbh, interne Veröffentlichung, Stuttgart 1994

[Mertens97] P. Mertens, et al.: „*Formen integrierter Anwendungssysteme zwischen Individual- und Standardsoftware*“ veröffentlicht von: FORWISS (Bayrisches Forschungszentrum für Wissensbasierte Systeme) 1997

[Mössenböck97] H. Mössenböck: „*Systemsoftware*“ erschienen in: P. Rechenberg/G. Pomberger: „*Informatik-Handbuch*“, 1. Auflage, Hanser 1997

[Mühlbacher98] J. Mühlbacher, M. Schröder: „*Der Weg zur Telearbeit führt über Business Reengineering*“ erschienen in: SYSPRO 64/98

[OMG99] Object Management Group Inc.: http://www.omg.org „*Unified Modelling Language Specification*“, Version 1.3 alpha R5:, http://www.omg.org/news/pr97/umlprimer.html, Juli 1999

[Orange97] G. Orange, P. Dixon: „*Business Process Redesign*“ erschienen in: Informatik Forum,Band 11, Nr. 1, März 1997

[Osterloh97] M. Osterloh: „*Prozessmanagement als Kernkompetenz*“, Gabler Verlag 1997

[Paulk93] M- Paulk et al.: „*Capability Maturity Model. Version 1.1*“ erschienen in: „*IEEE Software*“ 10/44, 1993

[PriceW98] Price Waterhouse Change Integration Team: „*Das Management Paradox*“, Campus Verlag 98

[REFA84] REFA: „*Methodenlehre des Arbeitsstudiums*“, Hanser Verlag, 1984, 7. Auflage

[Reichwald96] R. Reichwald, Technische Universität München, Lehrstuhl für Allgemeine und industrielle Betriebswirtschaftslehre, 1996

[Reisig86] W. Reisig: „*Petrinetze – Eine Einführung*“, Springer Berlin, 1986, 2. Auflage

[Ronaghi99] F. Ronaghi, S. Junginger: „Geschäftsprozeßmanagement in Österreich: State-of-the-Art“ erschienen am Institut für angewandte Informatik und Informationssysteme, Abteilung Knowledge Engineering der Universität Wien, Februar 1999

[Rumbaugh 91] Rumbaugh et al.: „*Object Oriented Modelling and Design*“, Prentice Hall 1991

[SAP] Homepage der SAP AG Walldorf: http://www.sap-ag.com , Juli 1999

[Sauerbrey89] G. Sauerbrey: „Betriebliche Organisation im Informationszeitalter“, Dr. Alfred Hüthig VerlaggmbH Heidelberg, 1989

[Scheer94] A.-W. Scheer: „*Wirtschaftsinformatik – Referenz-Modelle für industrielle Geschäftsprozesse*“, 4. Auflage, 1994

[Schröder98] M. Schröder: Skriptum zur Vorlesung „*Planung und Durchführung/Implementierung von Business Engineering Projekten*“ Wintersemester 1998/99, Johannes Kepler Universität Linz, Institut für Informationsverarbeitung und Mikroprozessortechnik 1998

[Schröder99] M. Schröder: Skriptum zur Vorlesung „*Informatik im Business Reengineering*“ Sommersemester 1999, Johannes Kepler Universität Linz, Institut für Informationsverarbeitung und Mikroprozessortechnik 1999

[Seagate] Homepage: http://www.seagate.com , September 1999

[ZZZ] Homepage der ZZZ GmbH & CoKg Linz: http://www.ZZZ.at, Juli 1999

[Sinz99] E. Sinz: „*Konstruktion von Informationssystemen*“ erschienen in: Peter Rechenberg/Gustav Pomberger: „*Informatik-Handbuch*“, 2. Auflage, Hanser 1999

[YYY] Homepage der YYY electronic GmbH & CoKg Linz: http://www.YYY.com, Juli 1999

[Staud99] J. Staud: „*Geschäftsprozeßanalyse mit Ereignisgesteuerten Prozeßketten*“; Springer Verlag Berlin Heidelberg New York; 1999

[Traunmüller] R. Traunmüller: Skriptum zur Vorlesung „*Angewandte Informatik in Wirtschaft und Verwaltung*“ Sommersemester 1994-1999, Johannes Kepler Universität Linz, Abteilung für Angewandte Informatik

6.2 Abbildungen

Abb. 1 : Einteilung von Software. Hierarchie (links) und Schi chtenmodell (rechts) 10
Abb. 2 Das Business Reengineering Labor-Konzept [Langer9 4] 17
Abb. 3 Gemeinsame Gestaltung von Unternehmenssoftware u n d Geschäftsprozessen 19
Abb. 4 Vorgehensmodell 22
Abb. 5 Kontinuierliche Prozeßverbesserung [Dion93] 26
Abb. 6 Phasen der Lösung eines Problems [Chroust97] 29
Abb. 7 BAAN Target, Beispiel für ein Phasenkonzept 34
Abb. 8 Eine SAP und eine Applix-Schnittstelle 38
Abb. 9 Lineare und baumartige Zustandsübergänge. Übergäng e zwischen Fenstern. 40
Abb. 10 Vergleich zwischen klassischem und prozeßorientiert em Organisationsverständnis 43
Abb. 11 Das AENEIS-Konzept [Langer94] 52
Ein Geschäftsprozeß in AENEIS 53
Abb. 13 GP in AENEIS mit Detailansicht einer Aktivität 53
Abb. 14 Das ARIS-Konzept [Scheer94] 55
Abb. 15 eEPK Notation 57
EPK-Notation und ADONIS-Notation 58
Abb. 17 Vorgehen bei der Firma YYY 71
Abb. 18 AENEIS Berichtsformate 73
Abb. 19 Berichtsstrukturen 74
Abstraktion zur Modellierung des Aktiv-Vertriebs 76
Abb. 21 Mitarbeiter-, Team-, und Arbeitsplatzsymbol 84
Abb. 22 Steigerung der Identifikation durch Personalisierung 84
Abb. 23 Verborgene Arbeitsschritte 86
Abb. 24 Arbeitschritte detailliert 86
Abb. 25 Eingabemaske mit Karteikarten: Applix Werbekampa gnemanagement 88
Abb. 26 Inhalte aller Karteikarten von Applix Werbekampagn emanagement 89
Abb. 27 Hierarchie von Eingabemasken 89
Abb. 28 Applix- und AENEIS-Notation 91

6.3 Definitionen

Definition 1: Business Engineering (synonym Enterprise Engi n eering) 5
Definition 2: Software [Duden93] 10
Definition 3: Anwendungssoftware [Floyd97] 10
Definition 4: Standardsoftware 11
Definition 5: Individualsoftware 12
Definition 6: Modellbildung [Traunmüller]: 28
Definition 7: Vorgehensmodelle 32
Definition 8: Prototyping 35
Definition 9: Qualität (nach DIN 55350) 36
Definition 10: Aufbauorganisation [REFA84] 41
Definition 11: Ablauforganisation (nach [REFA84]) 41
Definition 12: Geschäftsprozeß [Langer94] 44
Definition 13: Kernprozeß 44
Definition 14: Supportprozeß 44
Definition 15: Front-end CASE-Werkzeuge 50
Definition 16: Back-end CASE-Werkzeuge 50