Studienarbeit:

Erfahrungen bei einem objektorientierten Softwareprojekt

Boris Fittkau
Matrikel-Nr. 2482945

Universität Hamburg
Fachbereich Informatik
Arbeitsbereich Softwaretechnik
bei Prof. Dr. Züllighoven

Juni 1994

Inhaltsverzeichnis

1. Einleitung

Von April 1993 bis Oktober 1993 war ich als Entwickler an einem objektorientierten Softwareprojekt beteiligt, das die Erstellung eines Hotelreservierungssystems zum Ziel hatte. Nach Auslieferung des Systems habe ich im Hause des Auftraggebers am Arbeitsalltag teilgenommen, mit dem System gearbeitet und mit vielen der Anwender und Betroffenen gesprochen. Dazu habe ich eine Reihe von ausführlichen Einzelgesprächen mit Vertretern sowohl des Auftraggebers als auch des Softwarehauses geführt, wobei durch die Auswahl der Gesprächspartner ein Großteil der hierarchischen Ebenen abgedeckt war.

Die teilweise intensiven Gesprächen ergaben neben einer Reihe übereinstimmender Meinungen auch eine Vielzahl von unterschiedlichen Sichtweisen auf das Projekt. Konsens aller Gesprächspartner war jedoch, daß das Projekt als ein Erfolg zu werten sei. Daß Erfolg nicht objektiv meßbar ist, ist offensichtlich. Wenn man jedoch aus den vielen möglichen Perspektiven eine wesentliche, nämlich die wirtschaftliche Perspektive herausgreift, dann muß das Projekt sowohl für das Softwarehaus als auch für den Auftraggeber als Erfolg angesehen werden. Neben den nicht unerheblichen Erweiterungen am bestehenden System hat das Softwarehaus eine Reihe von Folgeaufträgen erhalten. Zwischen Softwarehaus und Auftraggeber hat sich auf Grund des Projekterfolges eine langfristige Zusammenarbeit etabliert.

1.1. Ziel der Arbeit

Auf Basis der nachträglich geführten Gespräche und meiner persönlichen Erfahrungen während des Projektes will ich in dieser Arbeit versuchen, den Verlauf der Softwareentwicklung zu beschreiben, zu analysieren und unter Berücksichtigung von Erkenntnissen der evolutionären Softwareentwicklung zu bewerten. Da das Projekt nicht explizit unter dem Leitbild einer evolutionären Softwareentwicklung durchgeführt wurde, will ich in dieser Arbeit auch versuchen aufzuzeigen, wo Erkenntnisse einer solchen kundenorientierten Sichtweise für das Projekt hilfreich gewesen wären und für zukünftige Projekte unterstützend wirken können.

1.2. Begriffsbildung

An dieser Stelle will ich einige der im folgenden benutzten Begriffe genauer erläutern. Gegenstand der Arbeit ist die Beschreibung und Bewertung eines Softwareentwicklungsprojektes. Dabei wurde als *Produkt* ein Hotelreservierungssystem entwickelt, das ich im folgenden kurz *Hotelsystem* nennen will.

Am Projekt waren unterschiedliche Gruppen beteiligt: Das Unternehmen, welches die Entwicklung der Software in Auftrag gegeben hatte, wird *Auftraggeber*

genannt, die Firma, die das Hotelsystem entwickelte, wird als *Softwarehaus* bezeichnet. Beide Firmen sind eigenständige Unternehmen.

Beim Auftraggeber war einer der Geschäftsführer am Projekt beteiligt. Hier spreche ich von *Geschäftsführer* oder *Auftraggeber-Management*; beide Begriff gebrauche ich synonym. Weiterhin gibt es die Gruppe der *Anwender* oder *Benutzer,* die heute in ihrer täglichen Arbeit mit dem Hotelsystem arbeiten. Teilweise wird in der Literatur zwischen *Anwendern* und *Benutzern* unterschieden. Benutzer sind dabei diejenigen, die direkt mit dem System arbeiten und Anwender diejenigen, die als Abteilungsleiter oder Schulungspersonal lediglich indirekte Nutzer des Systems sind. Da diese begriffliche Unterscheidung im Projektumfeld nicht getroffen wird, will ich ebenfalls darauf verzichten und beide Begriffe im folgenden synonym verwenden. Zusätzlich spreche ich von *Abteilungsleitern* als solchen, die die Software nicht direkt benutzen, jedoch Anwender als Mitarbeiter unter sich haben. Die Person, die die Anwender im Gebrauch des Hotelsystem geschult hat und maßgeblich für Testen und Qualitätssicherung zuständig war, wird *Schulungsleiter* genannt. Unter *Anwendervertreter* verstehe ich diejenigen, die in den Gesprächen die Interessen der Anwender vertreten haben. Dies waren im vorliegenden Projekt u.a. Geschäftsführer, Abteilungsleiter, Schulungsleiter und ausgewählte Anwender.

Auf Seiten des Softwarehauses wurde die Software von einer Reihe von *Entwicklern* entworfen und programmiert. Die Gruppe der Entwickler wird als *Entwicklerteam* bezeichnet, der Leiter des Entwicklerteams, der selber nicht mitentwickelte, wird *Projektleiter* genannt und zusammen mit dem *Geschäftsführer* des Softwarehauses zum *Softwarehaus-Management* gerechnet.

Zusätzlich zum Hotelsystem als *Produkt* des Projektes, ist für mich besonders der *Prozeß* der Softwareentwicklung wichtig, d.h. die Schritte und Phasen, in denen das System entstanden ist. Von Bedeutung ist in diesem Zusammenhang der Begriff der *evolutionären Softwareentwicklung,* der den prozeßhaften Charakter solcher Entwicklungsprojekte explizit würdigt und die Unterstützung des wechselseitigen Lernprozesses ins Zentrum der Bemühungen rückt. Ein späteres Kapitel wird diesen Begriff und seine Bedeutung ausführlicher erklären.

2. Projektbeschreibung

Im ersten Teil der Arbeit soll die Darstellung des Projektverlaufes im Vordergrund stehen. Zunächst beschreibe ich das Umfeld und die Entstehung des Projektes und daran anschließend die Phasen, in denen das Projekt ablief. Um die Beschreibung abzurunden, werden dann die Perspektiven von wichtigen am Projekt beteiligten Gruppen herausgearbeitet und dabei erste Bewertungen vorgenommen.

2.1. Umfeld

Der Auftraggeber ist ein Unternehmen mit ungefähr 300 Mitarbeitern, das seit 1987 in erster Linie Veranstaltungskarten im telefonischen Verkauf vertreibt. Die Verkäufer arbeiten mit einer sechs Jahre alten, eigens dafür entwickelten *Kartenreservierungssoftware*. Bereits 1992 waren im Unternehmen Bestrebungen erkennbar, die Vertriebsaktivitäten beträchtlich zu erweitern und neben Veranstaltungskarten eine Reihe anderer Freizeitaktivitäten (Hotelübernachtung, gastronomische Dienstleistungen, Souvenirartikel, etc.) anzubieten. Die bestehende Kartenreservierungssoftware konnte diese Expansion nicht mehr mitvollziehen, da sie in den letzten Jahren ständig weiterentwickelt worden war und als nicht mehr erweiterungsfähig angesehen wurde. Es wurde über verschiedene Lösungsmöglichkeiten nachgedacht, wobei die Neuentwicklung einer integrierten Softwarelösung eine denkbare Alternative darstellte.

Das Softwarehaus ist eine Firma mit heute etwa 50 Angestellten, die seit ungefähr sieben Jahren in den Bereichen Warenwirtschaft, Reservierungssysteme und Kreditkartenabwicklung erfolgreich Softwaresysteme entwickelt und Hardware vertreibt.

2.2. Projektentstehung

Der Auftraggeber hatte bereits Mitte 1992 begonnen, neben Veranstaltungskarten auch Hotelübernachtungen zu verkaufen. Zu diesem Zweck waren in den Niederlassungen des Auftraggebers eigene *Hotelabteilungen* entstanden, die Hotelreservierungen zunächst manuell und in geringem Umfang durchführten. Als das Management des Auftraggebers Ende 1992 beschloß, den Verkauf von Hotelübernachtungen beträchtlich auszuweiten, erhöhte sich der Arbeitsaufwand in den Hotelabteilungen und der Buchhaltung so erheblich, daß der Wunsch nach Softwareunterstützung geäußert wurde.

Das Softwarehaus machte dem Auftraggeber Anfang 1993 ein Angebot über die Erstellung eines Prototypen, der grundsätzlich die Abläufe eines integrierten Systems zum Verkauf von Freizeitaktivitäten zeigen und bereits den Verkauf von Hotelübernachtungen in geringem Maße unterstützen sollte. Wichtiger Bestandteil des Angebots waren mehrere Oberflächendarstellungen, die zeigen sollten, welche Vorstellungen das Softwarehaus von einem solchen System hatte.

Gerade diese Oberflächendarstellungen fanden beim Auftraggeber großen Beachtung und man entschloß sich, mit dem Softwarehaus zusammenzuarbeiten.

2.3. **Projektetablierung**

Im April 1993 wurde das Projekt von Auftraggeber und Softwarehaus etabliert. Obwohl die Grundlage immer noch das Angebot des Softwarehauses zur Erstellung eines Prototypen war, hatte sich die eigentliche Zielsetzung des Projektes inzwischen beträchtlich verschoben. Ins Zentrum war die Unterstützung der Hotelreservierung gerückt und es sollte dafür ein produktionsreifes System entwickelt werden. Der im Angebot enthaltende Prototyp wurde zunächst auf später verschoben.

Im Projektverlauf wurde dem Auftraggeber deutlich, daß auf dem Markt kein verfügbares System für den Verkauf von Freizeitaktivitäten existierte, welches die Anforderungen abdeckte und daß deshalb ein integriertes System vermutlich neu entwickelt werden müßte. Mit der Entwicklung des Hotelsystems konnte der Auftraggeber sowohl neue Hard- und Softwareplattformen (Objektorientierte Programmierung, Unix, Client-Server) erproben, als auch die Leistungsfähigkeit des Softwarehauses testen.

Der finanzielle Rahmen, der vom Softwarehaus für die Erstellung des Prototypen veranschlagt war, wurde für das Hotelsystem nicht mehr verändert. Da das Softwarehaus Interesse hatte, den Auftrag über das eventuell zu entwickelnde integrierte System zu erhalten, hatte das Projekt strategischen Charakter bekommen und es wurde ein finanzieller Verlust in Kauf genommen, der allerdings weit höher ausfiel, als erwartet.

Beim Auftraggeber war zunächst der Wunsch entstanden, das Hotelsystem mit dem alten Kartenreservierungssystem zu verbinden. Der engen zeitliche und finanzielle Rahmen machte dies jedoch unmöglich und man entschloß sich, mit zwei Systemen parallel zu arbeiten.

Das Hotelsystem läuft seit Oktober 1993 als Client-Server-Anwendung auf PC's unter Microsoft-Windows und einer RS-6000 (IBM-Unix) als Datenbankserver. Auf Serverseite entstanden im wesentlichen sog. Stored-Procedures, die direkt in der relationalen Datenbank (Sybase) abgelegt werden und dazu C-Programme, die die Batchverarbeitung realisieren. Auf Clientseite wurde objektorientiert mit Microsoft Visual C++ für Windows entwickelt. Die vorliegende Arbeit befaßt sich im wesentlichen mit der Entwicklung auf der Client-Seite.

2.4. **Projektphasen**

Das hier beschriebene Projekt beginnt im April 1993. Nach etwa einem halben Jahr wird im Oktober 1993 der Systembetrieb aufgenommen. Nach mehreren Erweiterungen am System soll das Projekt ungefähr im Juni 1994 beendet werden. Diese Arbeit befaßt sich im wesentlichen mit der Zeit bis zur Systemeinführung, bezieht aber Ereignisse bis Ende Mai 1994 in die Überlegungen mit ein.

Die folgende Tabelle enthält eine Auflistung wichtiger Aktivitäten im Projekt, die ich daran anschließend näher beschreiben will:

Anfang 1993	- Angebot über Erstellung eines Prototypen
Anfang April ‘93	- Projektstart Hotelsystem - Beginn der Anforderungsermittlung
Anfang Mai ‘93	- Erste Präsentation von Oberflächenentwürfen - Erstes “Control-Board-Meeting”
25./26.Mai ‘93	- Gesamtvorstellung des Leistungsumfangs und der Oberflächenentwürfe - Beginn des Entwurfs und der Implementierung
Anfang Juli ‘93	- Erster Test der bis dahin fertigen Programmteile durch den Schulungsleiter
Ende Sept. ‘93	- Auslieferung und Schulung des Systems
ca. 15.Okt. ‘93	- Produktionsbeginn
Dez. ‘93	- Abnahme des Systems und des Leistungsumfangs
ca. März ‘94	- Auftraggeber kündigt die Neuentwicklung eines integrierten Systems in Zusammenarbeit mit dem Softwarehaus an

2.4.1. **Ermittlung der Anforderungen**

In den verschiedenen Hotelabteilungen des Auftraggebers bearbeitete man seit Dezember 1992 manuell Hotelreservierungen. Im März 1993, also kurz vor Projektbeginn, hatte sich der Geschäftsführer des Auftraggebers in einer der Hotelabteilungen über die bis dahin entstandenen Arbeitsabläufe informieren lassen. Eine daraus resultierende grobe textuelle Beschreibung der Abläufe ging zum Softwarehaus und wurde dort vom Projektleiter in einen stellenbezogenen Datenflußplan umgesetzt, der eine Grundlage späterer Gespräche bildete.

Anfang April 1993 begann das Projekt offiziell, wobei zunächst eine Reihe von Treffen in den Räumen des Softwarehauses stattfanden, in denen Vertreter des Auftraggebers und des Softwarehauses die Anforderungen an das Hotelsystem ermittelten. Auf Seiten des Auftraggebers waren der Geschäftsführer und einige Abteilungsleiter (Verkauf, Buchhaltung, EDV) vertreten, sowie ein Mitarbeiter, der leitend in der Kartenreservierung tätig ist und schon bei der Entwicklung der alten Software beteiligt war. Das Softwarehaus wurde in der Regel vom Projektleiter und den beiden Entwicklern vertreten, die die Client-Seite entwerfen und programmieren sollten. Eine Mitarbeiterin, die die Online-Hilfe erstellen sollte, war anfangs ebenfalls bei den Treffen dabei, schied jedoch nach kurzer Zeit aus Kostengründen aus.

Festgehalten wurden die Ergebnisse dieser Besprechungen im *Leistungsumfang*, einem Textdokument, in welchem die Anforderungen an das System beschrieben wurden und das als Grundlage für den Vertrag zwischen Auftraggeber und Softwarehaus dienen sollte. Der Leistungsumfang, der vom Projektleiter geschrieben und aktualisiert wurde, nahm während des Projektes stetig an Umfang zu und hatte am Ende des Projektes eine Größe von annähernd 100 Seiten erreicht (zum ursprünglich geplanten Abschluß der Anforderungsermittlung hatte er lediglich einen Umfang von 40 Seiten). Die Teilnehmer erhielten in diesen Besprechungen Kopien des Leistungsumfangs und wurden aufgefordert, Fehlerberichtigungen und Verbesserungsvorschläge in den folgenden Treffen einzubringen.

Jeweils im Anschluß an diese Besprechungen fanden im Hause des Auftraggebers weitere Gespräche statt. Dabei teilten diejenigen Mitarbeiter, die an den Meetings im Softwarehaus beteiligt waren, den späteren Anwendern den Stand der Anforderungsermittlung mit und diskutierten mit ihnen den Leistungsumfang. Da keine Mitarbeiter des Softwarehauses an diesen Treffen beteiligt waren, wurden die Fragen und Änderungswünsche der Anwender von den ausgewählten Mitarbeitern in die nächsten Gespräche im Softwarehaus eingebracht.

Am 7.Mai zeigten die Entwickler das erste Mal einem Teil der späteren Anwender Oberflächenentwürfe (sog. *Systemvisionen*), die auf Grund der ersten Gespräche entworfen wurden. Diese wurden sehr positiv aufgenommen, da sie eine gute Diskussionsgrundlage bildeten. In späteren Treffen im Softwarehaus und beim Auftraggeber wurden den Teilnehmern jeweils die neuesten Systemvisionen gezeigt und darüber diskutiert.

Ursprünglich sollte die Anforderungsermittlung etwa nach einem Monat mit der “Verabschiedung” des Leistungsumfangs beendet sein. Aufgrund der wachsenden Anforderungen wurde dieser Termin verschoben und es kam mit einmonatiger Verspätung am 25.Mai 1993 zur *Gesamtvorstellung* des Leistungsumfangs und der bis dahin entworfenen Systemvisionen.

An der Gesamtvorstellung, die in den Räumen des Softwarehauses stattfand, nahmen etwa 20 Personen teil, wobei erstmals auch Abteilungsleiter und Anwender aus den anderen Niederlassungen des Auftraggebers anwesend waren, die vorher nur indirekt über das entstehende System informiert waren.

Das ursprünglich für einen Tag angesetzte Treffen wurde auf zwei Tage ausgedehnt. Das gesetzte Ziel, den Leistungsumfang offiziell abzuschließen und zu verabschieden, wurde nicht erreicht.

Die Gesamtvorstellung markierte das Ende einer für beide Seiten arbeitsintensiven Zeit, die in sehr konstruktiver Atmosphäre abgelaufen war. Die Erwartungen an das sich entwickelnde System waren hoch, die Stimmung bei den Beteiligten entsprechend positiv.

2.4.2. Entwurf und Implementierung

Für die Entwickler bedeutete die Gesamtvorstellung einen entscheidenen Einschnitt, da sie als direkte Teilnehmer der Anforderungsermittlung ausschieden und nun mit dem Entwurf und der teilweisen Implementierung des Systems begannen.

In “Control-Board-Meetings” (genannt *CBM*) fanden weiterhin Gespräche zwischen Softwarehaus und Auftraggeber statt, die in erster Linie dazu dienen sollten, den weiteren Projektablauf zu kontrollieren und zu koordinieren. Hier nahm von Seiten des Softwarehauses der Projektleiter teil und auf Seiten des Auftraggebers der Geschäftsführer und verschiedene Abteilungsleiter (inkl. dem *Schulungsleiter*, der das Programm später schulen sollte).

Die Verabschiedung des Leistungsumfangs stellte sich auch weiter als schwierig heraus, da die Ermittlung der Anforderungen noch nicht abgeschlossen war. In den CBM’s wurden weiterhin Erweiterungswünsche geäußert, die teilweise in die laufende Entwicklung mit einbezogen werden mußten. Der zunächst auf Anfang August festgelegte Auslieferungstermin wurde unter dem Eindruck der neuen Anforderungen verschoben.

Als sich beim Softwarehaus abzeichnete, daß der Umfang in der gesetzten Zeit nicht zu bewältigen war, wurde das Entwicklerteam auf der Client-Seite um zwei Entwickler erweitert, wovon einer nach Erfüllung seiner Aufgabe wieder ausschied und der andere bis zum Ende am Projekt mitarbeitete.

2.4.3. Testen der Software

Anfang Juli kam der Schulungsleiter des Auftraggebers ins Softwarehaus, um das System zu testen. Dies war gleichzeitig der erste Kontakt von Seiten des Auftraggebers mit Teilen des “echten” Systems. Es fanden Gespräche mit Entwicklern statt und es kam zu weiteren Änderungswünschen. Der Schulungsleiter testete die Software bis zur Systemeinführung kontinuierlich weiter und diskutierte nicht nur mit den Entwicklern, sondern auch im Hause des Auftraggebers mit den späteren Anwendern über die Software.

Die CBM-Meetings fanden zu der Zeit teilweise in größerer Runde statt (z.B. wurden Abteilungsleiter aus anderen Bereichen mit dazu geholt), um die weiterhin auftretenden Anforderungen mit allen Beteiligten klären und koordinieren zu können.

2.4.4. Systemeinführung und Schulung

Im Hause des Auftraggebers war für die Schulungen ein Termin festgelegt worden. Unter dem Druck dieses Termins wurde Ende September eine erste Version des Systems ausgeliefert, die jedoch instabil und fehlerhaft war. Die Schulungen verliefen dementsprechend unbefriedigend und wurden von den Teilnehmern eher als Testen des Systems denn als Schulung empfunden. Die vorher positive Stimmung beim Auftraggeber kippte ins Gegenteil um.

Während und nach den Schulungen wurde am System weiterentwickelt. Mitte Oktober schließlich wurde mit dem Echtbetrieb begonnen. Nach den Erfahrungen der Schulungsphase rechnete man beim Echtbetrieb ebenfalls mit Problemen, jedoch erwies sich das Programm im Vergleich zur Schulungsphase als wesentlich stabiler. Es waren zwar weiterhin Unzulänglichkeiten vorhanden, doch die Anwender konnten wichtige Arbeitsgänge fehlerfrei erledigen, was sie mit positiver Überraschung aufnahmen. Trotzdem war den Anwendern die Schulungsphase weiterhin sehr präsent, so daß selbst kleinste Fehler des Programms die Stimmung erneut gefährdeten. Auch in den innerbetrieblichen Abläufen beim Auftraggeber gab es in der Anfangsphase Abstimmungsschwierigkeiten, die eine Entspannung der Atmosphäre zusätzlich behinderten: Die Systembetreuung funktionierte nur unzulänglich, einige Niederlassungen waren gegenüber der Hauptniederlassung benachteiligt und es gab keinen offiziell Verantwortlichen für das Projekt, der als Ansprechpartner für die Anwender fungieren konnte.

2.4.5. Systembetrieb

Nachdem die ersten groben Unzulänglichkeiten im Programm verbessert waren und auch ein zweiter, wesentlicher Teil des Systems - allerdings unter erneuter Überziehung von zugesicherten Terminen - nachgeliefert wurde, verbesserte sich die Lage. Das Programm lief bereits in den ersten Wochen überraschend stabil, Abstürze kamen so gut wie nie vor. Die Anwender kamen mit der Zeit besser mit dem Programm zurecht. Für viele Dinge, die das Programm nicht direkt unterstützte, wurden “Workarounds” gefunden. Das anfänglich ins Schwanken geratene Vertrauen in die Software stieg. Allerdings mußten sich die Anwender vieles selber beibringen, da sie nicht an verbesserten und erweiterten Programmversionen nachgeschult wurden.

Auf Seiten des Softwarehauses herrschte zu der Zeit eine sehr gute Stimmung. Insbesondere die Stabilität des ausgelieferten Systems überraschte die Entwickler, die in früheren Projekten anderen Erfahrungen gemacht hatten. Es herrschte Übereinstimmung, daß die Fertigstellung des Systems in so kurzer Zeit eine herausragende Leistung darstellte.

2.4.6. Weiterentwicklung

Bis zum Ende des Jahres 1993 hatte sich die Zusammenarbeit zwischen Auftraggeber und Softwarehaus, aber auch im Hause des Auftraggebers, eingespielt. U.a. hatte sich ein standardisiertes Verfahren zur Fehlermeldung und -beseitigung ergeben. Wesentlichster Aspekt war jedoch, daß nun auch von

Seiten des Auftraggebers ein Verantwortlicher offiziell benannt war, der die Aktivitäten beim Auftraggeber in Bezug auf die Software koordinierte. Das Fehlen dieser Funktion hatte vorher für vielerlei Verwirrung und Unsicherheit gesorgt. Beim Auftraggeber wurde für zukünftige EDV-Projekte eine neue Abteilung gegründet, die sich speziell mit der internen und externen Koordinierung von neuen Softwareprojekten beschäftigen sollte.

Während des Systembetriebes tauchten neben Fehlern auch “echte” neue Anforderungen auf, die vom Auftraggeber wirklich benötigt wurden, die aber im ausgelieferten System nicht oder zu wenig berücksichtigt waren. Das Softwarehaus erhielt dafür mehrere Folgeaufträge, deren insgesamtes Volumen das des Hotelsystems überstieg.

Im Dezember 1993 wurde das System und der Leistungsumfang offiziell vom Auftraggeber abgenommen. Im März 1994 erhielt das Softwarehaus neben den anderen Folgeaufträgen auch den Auftrag über die Entwicklung des integrierten Systems. Damit war für das Softwarehaus eines der wesentlichen Ziele dieses Projektes erreicht.

2.4.7. **Erkannte Probleme und Verbesserungen**

Im Hinblick auf die Entwicklung des integrierten Systems wurden vom Softwarehaus und vom Auftraggeber die im Projekt begangenen Fehler analysiert und dabei folgende Problemfelder erkannt:

- Die anfängliche Analysephase war zu kurz. Daraus folgte, daß wichtige Anforderungen im Vorfeld nicht erkannt wurden und erst während des Projektverlaufes auftauchten.
- Die Meetings in der Analysephase waren zu groß, so daß die Themen nicht für alle Beteiligten gleichermaßen interessant waren und einige Beteiligte zu kurz kamen.
- Die Phase der Qualitätssicherung und des Testens vor Einführung der Software war zu kurz.
- Die Schulungen waren ungenügend und es wurde nie nachgeschult. Deshalb mußten sich die Anwender vieles selbst beibringen und einige Fehler wurden erst relativ spät entdeckt.
- Es gab beim Auftraggeber erst in der Spätphase des Projektes einen verantwortlichen Ansprechpartner.
- Die EDV-Abteilung des Auftraggebers (genannt *Systembetreuung*) war ungenügend auf das neue System vorbereitet und stand als Ansprechpartner nicht zur Verfügung.

Ausgehend von diesen Überlegungen wurde für die Entwicklung des integrierten Systems eine teilweise veränderte Vorgehensweise gewählt:

- Es wird vom Auftraggeber schon vor Projektbeginn ein Verantwortlicher für die Koordinierung des Softwareprojektes bestimmt. Dazu werden zusätzliche

Mitarbeiter eingestellt und eine neue Abteilung gegründet, deren explizite Aufgabe die Koordinierung und Qualitätssicherung ist.

- Man läßt sich mehr Zeit bei der Ermittlung der Anforderungen. Anfang 1994 erhält das Softwarehaus vom Auftraggeber einen separaten Auftrag, um die Anforderungen an ein integriertes System zu ermitteln.
- Die Ermittlung der Anforderungen findet nicht mehr in "großen Runden" statt, sondern in einer Vielzahl von Einzelgesprächen. Dazu reist der spätere Projektleiter des Softwarehauses durch Deutschland, besucht die einzelnen Niederlassungen des Auftraggebers und führt Einzelgespräche mit einer Vielzahl von späteren Anwendern, Abteilungsleitern und Managementvertretern, wobei die entstehenden Gesprächsprotokolle einmal mit den Betreffenden rückgekoppelt werden. Die Ergebnisse der Analyse werden in einem Leistungsumfang zusammengefaßt, der dann mit ausgewählten Mitarbeitern aus der Reihe der Abteilungsleiter besprochen wird.
- Qualitätssicherung soll bei der Entwicklung des integrierten Systems mehr Zeit in Anspruch nehmen. Das System soll innerhalb eines Jahres fertiggestellt sein und schon während der Designphase sollen Prototypen mit Anwendern diskutiert werden. Die Einführungsphase des Systems wird länger dauern und in mehreren Stufen ablaufen. Nach etwas mehr als der Hälfte der Zeit sollen bereits lauffähige Programmteile beim Auftraggeber getestet werden und zwei Monate vor Inbetriebnahme soll ein auf einzelne Anwendergruppen beschränkter, jedoch echter Produktionsbetrieb aufgenommen werden.

2.5. **Perspektiven auf das Projekt**

Eine der wesentlichsten Herausforderungen in Softwareentwicklungsprojekten scheint in der Vielzahl der vertretenen Interessengruppen zu liegen. Jede dieser Gruppen hat ihre eigene Sichtweise auf ein solches Projekt und verfolgt unterschiedliche Ziele. Um das daraus resultierende Sichtenproblem zu thematisieren und die Beschreibung des Projektverlaufes abzurunden, will ich im folgenden die Perspektiven von Entwicklern, Anwendern und Vertretern des Managements getrennt herausarbeiten.

In den betont subjektiven Darstellungen werden erklärender Text und Zitate gemischt, wobei Zitate lediglich durch kursive Schrift angedeutet werden, um den Lesefluß nicht unnötig zu behindern. Die Zitate entstammen sowohl den Gesprächen nach Projektabschluß als auch meinen Mitschriften während des Projektverlaufes. Sie sind teilweise wörtlich übernommen, teilweise verändert worden, um das Lesen zu erleichtern. Überschneidungen und Wiederholungen von bereits geschilderten Sachverhalten sind beabsichtigt. Die verschiedenen Perspektiven werden jeweils zunächst dargestellt und daran anschließend von mir zusammengefaßt und bewertet.

2.5.1. **Perspektive des Entwicklerteams**

Das Entwicklerteam bestand aus bis zu sieben Entwicklern, von denen vier auf der Client-Seite entwickelten und drei auf der Server-Seite für die Datenbank und den Dokumenten- und Listendruck verantwortlich waren. Die Darstellung konzentriert sich auf die Erfahrungen der Entwickler auf der Client-Seite.

"Es war genial, daß wir so ein System so schnell hingekriegt haben."

Als zum ersten Mal eine Zusammenarbeit mit dem Auftraggeber im Gespräch war, befanden sich die Entwickler noch in anderen Projekten, aber sie verfolgten die sich abzeichnende Zusammenarbeit mit Interesse. Der Projektanfang kam dann relativ schnell. Es hatte bereits Vorgespräche beim Auftraggeber gegeben und daraus waren einige Dokumente entstanden, die sich die Entwickler durchlasen. Dann begannen die ersten Treffen mit Vertretern des Auftraggebers. Ursprünglich war auf Seiten des Softwarehauses nur der Einsatz von einem Entwickler zur Programmierung der Client-Seite geplant gewesen. Ich selber sollte zunächst als Beobachter dienen, da ich über das Projekt meine Studienarbeit schreiben wollte. Relativ schnell jedoch war ich ebenfalls als Entwickler im Projekt beschäftigt. Im Verlaufe des Projektes kamen zwei weitere Entwickler hinzu.

Die ersten Treffen fanden in den Räumen des Softwarehauses statt. Dabei ging es zunächst um die Klärung, welche Funktionalität das System abdecken sollte. *Am Anfang wußte ja auch noch keiner, wo es hingeht*. Die Anforderungen mußten geklärt werden. *Wie kriege ich aus dem Kunden raus, was er will?* Dabei fiel dem Entwicklungsteam die Rolle zu, aus dem Gesagten die Anforderungen herauszufiltern. *Erst als wir die Anforderungen niedergeschrieben hatten, da haben die erst gemerkt und gesagt: 'Ja, das ist es, was wir wollen'.*

Es war nicht nur für die Entwickler ein Lernprozeß. Auch beim Auftraggeber hatte man wenig Erfahrung mit dem zu unterstützenden Gebiet. *Ganz klar, es war völliges Neuland, sie wußten nicht, wo sie sich das abgucken sollten*. Die Gespräche verliefen in ausgezeichneter Atmosphäre und beide Seiten waren engagiert bei der Sache. Die Gespräche fanden in Runden von ungefähr 10 Personen statt, dauerte teilweise lange und waren für alle Beteiligten sowohl interessant als auch sehr anstrengend. Oft wurden so viele verschiedene Alternativen diskutiert, daß unklar schien, für welche man sich im Endeffekt entschieden hatte. *Man glaubt, indem man darüber gesprochen hat, wäre es damit auch erledigt. Aber das ist es ja oftmals gar nicht. Das ist dann der Punkt, wo der Ball wieder an die Softwarefirma geht, wo der Bodensatz ermittelt werden muß.*

Die Ergebnisse der Gespräche sollten zunächst durch die Entwickler festgehalten werden, aber als dies nur unbefriedigende Resultate lieferte, begann der Projektleiter, diese Aufgabe selber auszuführen und den sog. "Leistungsumfang" zu erstellen. *Er hat ein sehr großes Talent dafür, die Sachen kurz, prägnant und gut strukturiert und formuliert zu Papier zu bringen. Und vor allem hat er auch eine schnelle Auffassungsgabe. Das hat auch nicht jeder. Er hat ja auch von früher her enorm viel Erfahrung.*

Der Leistungsumfang war ein wesentliches Dokument für die Entwickler, die *Bibel des Projekts.* Dazu kamen die Systemvisionen (Oberflächendarstellungen), die

die Entwickler schon bald nach Beginn entwarfen und die sie in den Gesprächen den Vertretern des Auftraggebers präsentierten. Dabei machten die Entwickler unterschiedliche Erfahrungen. Das Zeigen der Systemvisionen (z.B. in Form von sog. Dia-Shows) stieß bei den Anwendern auf positives Interesse und regte breite Diskussionen an. Bei den Gesprächen über den Leistungsumfang - einem reinen Textdokument - beteiligten sich nur einige wenige (EDV-erfahrene) Leute; bei den Systemvisionen kamen Anregungen von allen Seiten. Anfangs enthielten die Systemvisionen zusätzlichen erklärenden Text, der den Ablauf verdeutlichen sollte, jedoch erwies sich das Ändern dieser Dokumente als zu aufwendig, so daß die Entwickler dazu übergingen, nur noch Maskenentwürfe vorzustellen. Außerhalb der Treffen scheinen sich die Anwender wenig mit den Systemvisionen beschäftigt zu haben. Ob jemand den erklärenden Text gelesen hatte, ist zweifelhaft. *Es hat niemand gelesen. Es kam kein Feedback und deshalb hätte man sich das Schreiben schenken können. Die haben sich die Visionen angeschaut und fanden das alles ganz toll. Oftmals wußten sie aber nicht, was sich dahinter verbirgt. Aber sie konnten nicht in den 20-30 Minuten, wo sie sich das angeschaut haben, etwas besseres entwickeln. Die sehen das Ganze und haben überhaupt keine Chance. Die müssen das gut finden.*

Dabei spielte wohl auch die Größe der Gruppe ein Rolle. *Ich hatte den Eindruck, oftmals wollten sie sich in der großen Runde nicht die Blöße geben, daß sie es nicht verstanden hatten. Eine gewisse Ehrfurchtshaltung vor den studierten Informatikern hatten die dann doch. Im Einzelgespräch habe ich festgestellt, daß da viel leichter Fragen kamen. Die haben geglaubt, daß das, was sie sehen, ihre Probleme löst. Das war der Punkt.*

Schon zur Zeit der ersten Gespräche machte das Management des Softwarehauses den Entwicklern klar, daß es in erster Linie darum ging, möglichst schnell ein lauffähiges System ohne viele *Schnörkel* zu erstellen. *Das Ding muß stehen, fertig werden und gut laufen. Soviel wie nötig und so wenig wie möglich.* Deshalb wurde am Anfang versucht, die Ermittlung der Anforderungen schnellstmöglich abzuschließen, um mit dem Entwurf und der Programmierung beginnen zu können. Ursprünglich hatte man für die Erhebung der Anforderungen einen Monat angesetzt. Die Gesamtvorstellung des Leistungsumfangs fand dann aber erst nach zwei Monaten statt und auch da wurde kein Ende erreicht. An der Gesamtvorstellung nahmen jetzt auch Anwender und Abteilungsleiter aus anderen Niederlassungen des Auftraggebers teil, die davor nur indirekt über das Projekt informiert waren. *Ich war ein bißchen überrascht, die erst jetzt kennenzulernen. Es wäre besser gewesen, die wären schon vorher dabei gewesen.*

Zu dieser Zeit gab es auch schon einen kleinen Ausschnitt des Systems in ablauffähiger Form, obwohl in so kleinem Ausmaß, daß man nicht von einem Prototyp sprechen konnte. Lediglich dem Geschäftsführer des Auftraggebers wurde er gezeigt und das führte zu der Frage, ob nicht schon zu der bald darauf stattfindenden Messe ein Prototyp fertig sein könnte. Der Entwickler hielt das nicht für ganz unmöglich, es wurde jedoch vereinbart, daß die Entwicklung des Systems Vorrang vor einem Prototyp hatte.

Nach der Gesamtvorstellung am 25./26.Mai *versanken* die Entwickler in die Realisierung des Systems. *Am Anfang lief es zähflüssig, C++ war neu, die Datenbank war neu. Da stand der Chef hinter mir und wollte was sehen. Um 21.50 ist er gegangen und um 22.00 Uhr konnte man was sehen.* Neben der Programmiersprache war auch die

Entwicklungsumgebung teilweise neu gekauft worden und in ihrer Leistungsfähigkeit erst in Ansätzen erprobt. Allerdings erwies sie sich als stabil, was die Entwickler positiv überraschte. Trotzdem brauchte es eine ganze Zeit, bis produktiv gearbeitet werden konnte. *Irgendwann ging es dann sehr flüssig, die Erfahrungen waren gesammelt. Das hat ein bißchen gedauert.* Auch mußten die Entwickler die vorhandene Klassenbibliothek durch eine Reihe von Tool-Klassen erweitern, was zusätzliche Zeit kostete. Lauffähige Teile des Systems zu produzieren, stellte sich besonders in den wesentlichen Bereichen, wie z.B. der Auftragsbearbeitung, als überraschend langwierig heraus. An die Fertigstellung eines Prototypen vor Auslieferung des Gesamtsystems war nicht mehr zu denken.

Überhaupt standen bereits relativ früh die Termine zur Fertigstellung der Software im Raum. Noch Anfang Juli wurde der 1.August als Auslieferungstermin angestrebt. Den Entwicklern erschienen die gesetzten Termine praktisch nicht zu halten. *Der Termin war nicht zu halten, das war allen klar. Man kann sich nur beeilen, daß es nicht zu spät wird. Es geht darum, lediglich die Peinlichkeit in Grenzen zu halten.*

Bereits die Anfangsphase des Projekts war eine arbeitsintensive Zeit gewesen. Daß Termindruck bestand, war von vorne herein klar. Allerdings steigerte sich der Druck und der Arbeitseinsatz gerade zum Ende des Projektes erheblich und bewegte sich mit bis zu 285 Stunden im Monat an der Leistungsgrenze der Entwickler. *Es war ja auch viel Stoff zu bewältigen. Es war sehr anstrengend. Wochenlang keinen Tag frei zu haben und schlechtes Gewissen zu haben, wenn man mal einen Tag frei haben wollte. Das war nicht ganz ohne. Ich glaube, draußen war zu der Zeit gerade Sommer.*

Das Entwicklerteam wurde erweitert, weil auch der inzwischen verschobene Termin sonst nicht zu halten gewesen wäre. Dies kostete zusätzliche Einarbeitungszeit, verlief aber weitgehend unproblematisch.

Im Juli kam das erste Mal in der Person des Schulungsleiters ein Vertreter des Auftraggebers ins Softwarehaus, um die Software zu testen. Das war nach intensiver Programmierzeit der erste Kontakt der Entwickler mit der *Außenwelt*. Es war zwar Zeit für Gespräche und der Schulungsleiter brachte diverse Vorschläge in das System ein, im Prinzip jedoch steckten die Entwickler immer noch *bis über beide Ohren* in der Implementierung. In dieser Zeit hörten die Entwickler hauptsächlich über den Projektleiter von den neuesten Tendenzen beim Auftraggeber. Gelegentlich stand dort die Zusammenarbeit mit einem Reiseveranstalter im Raum, was erheblichen Einfluß auf die Software genommen hätte. *Gott sei Dank hat man das dann abgeblasen.* Der Projektleiter vermittelte den Entwicklern die neuen Anforderungen, die unbedingt noch in das System hinein sollten. Den Entwicklern erschien es, als würden die neuen Anforderungen überhaupt kein Ende nehmen.

Die Zeit vor der Auslieferung verstärkte nochmals den Druck auf die Entwickler. Nach mehreren Terminverschiebungen stand der Zeitpunkt fest, an dem die ersten Anwender mit dem System geschult werden sollten. Erst unmittelbar vor den Schulungen waren die benötigten Programmteile fertiggestellt. *Da baut man natürlich Flüchtigkeitsfehler ein und das System stürzt öfters mal ab.* Die Folgen dieser Instabilitäten bekamen die Entwickler zum größten Teil nur indirekt mit, da im

wesentlichen der Projektleiter Kontakt zum Auftraggeber und den Anwendern hatte. Die Entwickler waren gerade in dieser Phase mit Implementieren und Testen beschäftigt, so daß direkte Kontakte zu den Anwendern eher hinderlich erschienen. *Es ist ganz gut, wenn man die Entwickler in dem Glauben läßt, daß sie die Schönsten und die Größten sind. Weil, das pflegt die Seele. Wenn man hört, daß der Anwender das alles schlecht findet, dann hat man auch keine Lust mehr.*

Trotzdem ließ es sich nicht völlig vermeiden, von der schlechten Stimmung der Anwender während der Schulungen zu erfahren. *Es ging soweit, daß sie an dem einen Tag nahe vorm Platzen waren. Sie sind fast explodiert und es war keiner da, der sie beruhigen konnte. In diesem Fall mußte ich diese Rolle übernehmen und mußte feststellen, daß dort ein ganz schön harter Wind weht. Am nächsten Tag haben sie dann die Schulung abgehalten und da ist das Programm auch nur einmal abgestürzt. Hinterher haben sie sich dann bei mir entschuldigt.*

Auch später war immer wieder festzustellen, wie *traumatisch* diese Zeit für die Anwender war. Übereinstimmend hatten sie erwartet, an einem fertigen System geschult zu werden und empfanden folglich die Schulungen eher als ein Testen des Systems. *Dem Kunden wurde quasi verkauft, daß es fertig ist, obwohl es gar nicht fertig war. Es war ein Test. So schlecht war es gar nicht. Dafür, daß es ein Test war, war es ganz gut.*

Wegen der Schwierigkeiten während der Schulungen verschob man den Termin der Produktionsaufnahme und die Entwickler hatten zusätzliche Zeit, die Fehler im Programm zu korrigieren. Der Produktionsbeginn verlief dann viel unproblematischer, als es die Schulungen hatten erwarten lassen. Das Programm hatte Unhandlichkeiten, lief aber stabil und ohne Abstürze. *Es lag an der Zeit. Wir hatten ja für manche Sachen fast keine Zeit zum Testen. Und dafür lief das Produkt erstaunlich gut. Nur wußten das die Anwender nicht. Die konnten es nicht würdigen, daß es so schnell und stabil lief. Da war ich oft drauf und dran, denen zu sagen: 'Also Leute, jeder andere hätte es wesentlich schlechter gemacht'… Wir waren voll abgeschirmt. Deshalb war es für mich auch so ein Schock zu sehen, daß es Leute gab, die nicht zufrieden waren. Aus meiner Sicht konnte es nicht sein, daß damit jemand nicht zufrieden ist.*

Nach Produktionsaufnahme kam es vereinzelt zu direkten Kontakten zwischen Entwicklern und Anwendern, bei denen die Entwickler den Anwendern einzelne Programmteile erklärten und Fragen beantworteten. *Da war ich öfters mal da und hab ihnen das erklärt. Sie waren doch recht unsicher am Anfang, hatten vor jedem Button Angst gehabt. Hat Spaß gemacht, ihnen das zu erklären.* Diese direkten Kontakte verliefen für beide Seiten überaus produktiv. *Sie hatten unglaublich viele Fragen und ich konnte ihnen Dinge im Programm zeigen, von denen sie keine Ahnung hatten.*

In der Folge waren die Entwickler mit Korrekturen und Fehlerbehebungen beschäftigt. Es gab keine schwerwiegenden Fehler im System, die zu einem Stop des Produktionsbetriebes führten. Insofern konnten die Entwickler mehr in Ruhe arbeiten. Mit der Zeit wurden auch die Folgeaufträge konkreter und damit der wirtschaftliche Erfolg des Projektes sichtbar. Auch für die Entwickler selber war das Projekt ein Erfolg. Die Zusammenarbeit zwischen den Entwicklern hatte sehr gut geklappt. Die Entwickler hatten einen hohen Arbeitseinsatz gebracht und in extrem kurzer Zeit mit einer neue Technologie ein stabiles und optisch ansprechendes System entwickelt. *Es wurde ein System geschaffen, was sie gebraucht haben und das haben sie auch bekommen. Und das in einer Zeit, die rekordverdächtig ist.*

Zusammenfassung der Entwicklerperspektive

Für die Entwickler stellte die Erstellung des Softwaresystems in vielerlei Hinsicht eine Herausforderung dar: Das fachliche Umfeld war unbekannt, die Entwicklungsplattform und Programmiersprache waren neu, das zu realisierende System war umfangreicher als erwartet und die Zeit war begrenzt. Die Entwickler waren zunächst stark an der Ermittlung der Anforderungen beteiligt und begannen früh, ihre Vorstellungen vom System in Form von Systemvisionen darzustellen und diese auch mit den späteren Anwendern zu besprechen. Nach der Gesamtvorstellung konzentrierte sich die Arbeit der Entwickler auf die Fertigstellung des Systems, wobei sie zunächst Anfangsschwierigkeiten im Umgang mit der ungewohnten Programmierumgebung überwinden mußten und im Verlaufe des Projektes durch wachsende Anforderungen und Termindruck zusätzlich gefordert wurden. Besonders die Zeit vor und während der ersten Auslieferungen ist den Entwicklern als strapaziöse Zeit in Erinnerung, da sie dort sehr schnell entwickeln mußten, wenig Zeit zum Testen hatten und die daraus resultierenden Instabilitäten und Unhandlichkeiten im Produkt für große Unzufriedenheit bei den Anwendern sorgten. Bei den Entwicklern hinterließ dies den Eindruck, daß ihre Leistungen vom Auftraggeber nicht in angemessener Form gewürdigt wurden. Insgesamt kann man sagen, daß die Entwickler die Herausforderungen, die das Projekt an sie gestellt hat, in positiver Weise angenommen haben und unter Zeitdruck mit einem erheblichen Arbeitsaufwand ein inzwischen stabil funktionierendes System erfolgreich fertiggestellt haben. Für die Entwickler stellt sich deshalb die Arbeit im Projekt als ein großer Erfolg dar.

Bewertung

In der Anfangsphase versuchte das Softwarehaus, die Vertreter des Auftraggebers durch den Einsatz von Oberflächenentwürfen intensiv an der Softwareentwicklung zu beteiligen. Bemerkenswert ist, daß diese Einbeziehung scheinbar in späteren Phasen nicht mehr aufrechterhalten wurde, weil immer stärker die Fertigstellung des Produktes in den Vordergrund rückte. Es fällt auf, daß auch die Entwickler zu Anfang noch stark an der Ermittlung der Anforderungen mitwirkten, später jedoch vollständig mit der Programmrealisation beschäftigt waren und erst nach Auslieferung wieder teilweise direkten Kontakt zu den Anwendern bekamen. In diesem Zusammenhang erscheint mir die Person des Projektleiters wesentlich, der für die Entwickler als Vermittler neuer Anforderungen eine Schlüsselrolle spielte. Wichtig ist auch, daß eine Art prototypischer Vorgehensweise zumindest greifbar erschien, jedoch wegen der mangelnden Vertrautheit mit der Programmierumgebung und des Termindrucks nicht weiter verfolgt werden konnte.

Besonders in Bezug auf die Analysephase wird in der Perspektive der Entwickler das zugrundeliegende Rollenverständnis deutlich: die Entwickler sehen sich als diejenigen, die aus den Wünschen und Informationen des Auftraggebers die Anforderungen an das System herleiten, die somit das “Problem” eher für den Auftraggeber als mit ihm gemeinsam lösen sollen. Die Anwender und Anwendervertreter werden tendenziell in der Rolle von Informationslieferanten gesehen,

und weniger als eigentliche Experten der Anwendungswelt, die mit ihrem Fachwissen die Entwickler unterstützen und von den Entwicklern mit deren EDV-Wissen beraten werden.

In der Darstellung der Entwicklerperspektive sind meiner Meinung nach fundamentale Interessenkonflikte innerhalb und zwischen den beteiligten Gruppen erkennbar: Für die Entwickler steht das System und seine Fertigstellung im Vordergrund der Bemühungen. Dabei ist eine schnelle, termingerechte Auslieferung wichtig, gleichzeitig aber auch die Erstellung eines stabilen und performanten Systems. Der inhärente Widerspruch war scheinbar in diesem Projekt nur schwer zu lösen: der Termindruck ließ den Entwicklern zu wenig Zeit zum Testen oder zwang sie, ungetestetete Programmteile auszuliefern; die Anwender sahen während der Schulungen im wesentlichen die Instabilitäten im Programm und deshalb nicht die Leistung der Entwickler. Wichtig erscheint mir, daß beide Gruppen zu sehr mit sich selber beschäftigt waren, um von der Situation der Gegenseite genügend Kenntnis zu nehmen. Die Folge war eine verstärkte Polarisierung und scheinbar unrealistische Erwartungen und Mißtrauen gegenüber der Gegenseite. Beispielsweise hatten die Entwickler enorm viel Zeit in das Gelingen des Projektes investiert und fühlten diese Leistung durch die undankbar anmutenden Reaktionen der Anwender in keinster Weise gewürdigt. Es erscheint mir auch wichtig zu sein, daß diese Kommunikationslücke in den selten stattfindenden direkten Kontakten zwischen Entwicklern und Anwendern geschlossen werden konnte und gerade durch die Förderung gegenseitigen Verständnisses Resultate erbracht werden konnten, die für beide Seiten produktiv waren.

2.5.2. **Perspektive der Anwender**

Der Auftraggeber verfügt in Deutschland über mehrere Niederlassungen, in deren jeweiliger Hotelabteilung seit Oktober 1993 mit dem Hotelsystem gearbeitet wird. Die folgende Darstellung läßt sowohl direkte Benutzer der Anwendung zu Wort kommen, als auch diejenigen, die das System auftraggeberseitig getestet und geschult haben.

“Die Schulungen waren ein Fiasko.”

Bereits vor offiziellem Projektbeginn wurden einige der späteren Anwender durch den Geschäftsführer befragt, um die manuellen Arbeitsabläufe der Hotelreservierung zu klären. Die Anwender hörten danach erst wieder vom Projekt, als die ersten großen Gespräche der Analysephase schon begonnen hatten, bei denen die Anwender durch Verkaufsleitung und einen ausgesuchten Mitarbeiter aus einem anderen Geschäftsbereich bei den Gesprächen vertreten waren. *Die ersten Gespräche fand ich sehr positiv und sehr konstruktiv. Ich hab den Eindruck gehabt, das die Softwarefirma immer sehr detailliert vorbereitet war. Was ich im Nachhinein negativ fand, war, daß man eigentlich niemanden aus der Praxis genommen hat. Man hat sehr hoch angesetzt. Es war nie jemand aus der Hotelabteilung dabei.*

Nach den Gesprächen im Softwarehaus fanden beim Auftraggeber Meetings statt, bei denen die Anwendern vom Stand der Anforderungsermittlung und dem Inhalt des Leistungsumfangs unterrichtet wurden. *Ich glaube, ganz am Anfang waren nur*

die beiden mit dabei. Wir haben immer Teile des Leistungsumfangs bekommen, haben den mit den beiden durchgesprochen und die sind dann mit unserem Input wieder in die Folgemeetings gegangen. Es war meistens so, daß wir abends um fünf Uhr ein Pamphlet [den Leistungsumfang] *von ca. 50 Seiten bekommen haben und uns bis zum nächsten morgen um 10 Uhr darüber Gedanken machen sollten… Die Stimmung war hektisch. Eigentlich immer unter Zeitdruck. Grundsätzlich. Weil immer am nächsten Tag irgendein Treffen war und das ging immer über Stunden.*

Die Diskussionen über den Inhalt des Leistungsumfangs stellten sich als schwierig und zäh heraus. Es schien für die Anwender unmöglich, sich das Aussehen des späteren Systems anhand des Leistungsumfangs vorzustellen. *Die Terminologie in diesem Leistungsumfang war mir auch vollkommen fremd… Was halt eindeutig schlecht daran war: Es war zu theoretisch. Ganz klar. Es war unheimlich schwer, sich das so richtig vor Augen zu führen. Du hast dir einfach zuviel in der Theorie Gedanken darüber gemacht… Zu der Zeit, als wir noch in den CBM's gesessen haben, hatte ich überhaupt keine konkrete Vorstellung davon, wie das in der Praxis wird. Absolut nicht.*

Der Leistungsumfang wurde zwar als wichtig angesehen, jedoch wurde er nur von wenigen Mitarbeitern des Auftraggebers intensiv und kontinuierlich gelesen. Im allgemeinen wurde auf die Richtigkeit des Inhaltes lediglich vertraut. *Da gab es folgende Schwierigkeit: Der Leistungsumfang wurde verteilt und es waren auch die Änderungen zu früheren Versionen markiert und auf dem nächsten CBM sollte immer besprochen werden, ist das alles okay so, ist das beabsichtigt so, habt ihr das verstanden, usw. Und beim nächsten CBM kam man entweder aus Zeitgründen nicht dazu oder, weil es keiner gelesen hatte. Es wurde nicht so richtig empfunden, daß wie eine Hausaufgabe vorzubereiten. Ich kenne zwei Leute in der Firma, die den Leistungsumfang überhaupt jemals ganz gelesen haben. Finde ich erstaunlich.*

Relativ bald wurden einzelnen Anwender die Systemvisionen (z.B. in Form von Dia-Shows) gezeigt, die die Entwickler anhand der ersten Gespräche erstellt hatten. *Diese Dia-Shows fand ich sehr, sehr gut. Das muß man unbedingt beibehalten. Auch, wenn man es noch nicht ausprogrammiert hat, so kann man doch erläutern, welche Bedeutung dieses oder jenes Feld hat. Das vor Augen zu haben ist gut, weil man es dann nicht nur im Kopf hat und jeder unterschiedliche Bilder davon hat. Daß man etwas anzufassen hat.*

Für die späteren Anwender war dann nach der Gesamtvorstellung am 25./26.Mai der Kontakt zur laufenden Entwicklung beendet. Da die Einführung des Systems auch geschäftlich vorbereitet werden mußte, blieb den Anwendern keine Zeit mehr, sich um das Projekt zu kümmern. *Relativ bald kam der Druck von der Geschäftsleitung, zu einem möglichst frühen Termin mit dem Hotelverkauf zu beginnen. Da mußten wir die ganzen Verhandlungen mit den Hotels führen. Anfang Juni hatten wir schon total viel Arbeit. Dementsprechend schrecklich war dann die Zeit zwischen Mitte August und Ende September, weil wir da nur Termine hatten und den Hoteleinkauf gemacht haben. Wir hatten im Grunde genommen gar nicht mehr die Zeit, uns um das Projekt zu kümmern.*

Bei den Anwendern herrschte die Vorstellung, daß die Ermittlung der Anforderungen abgeschlossen sei und nun die Entwickler mit der Implementierung des Systems beginnen würden. *Da hatten wir die Dia-Shows auch schon gesehen. Somit war für uns das Ding schon fast abgeschlossen. Irgend jemand sagte, 'So, Leistungsumfang ist jetzt abgeschlossen, ist verabschiedet worden und jetzt setzen die sich hin und programmieren und werden wohl irgendwann aus ihrem Kämmerlein herauskommen. Da hatte ich nicht mehr das Gefühl, daß man darüber reden müßte. Ich hatte ein total positives Gefühl bei der ganzen Geschichte.*

Während die Anwender erst wieder bei Systemeinführung Kontakt zum Projekt bekamen, kam der Schulungsleiter schon während der laufenden Entwicklung

ins Softwarehaus, um Qualitätssicherung zu betreiben und das Programm, was er hinterher auch schulen sollte, zu testen. Dabei war er zunächst überrascht, daß das Programm nicht in dem Maße fertig war, wie er es erwartet hatte. *Ich bin von der Überlegung ausgegangen, daß da Leute sich wochenlang Gedanken gemacht haben, Abläufe festgeklopft, Strukturen überlegt und daß es jetzt nur noch marginale Änderungen geben wird. Ein bißchen Einfluß von außen muß noch möglich sein, aber im Prinzip steht das Ding. Aber das war überhaupt nicht so.*

Auch war das Aussehen des Programms anders als erwartet. *Ich hab vorher die Systemvisionen auf Papier ausgedruckt gesehen und sah nun plötzlich die ersten Programmschritte. Und fand nichts wieder. Da war ein Unterschied, zwischen dem, was man mir gegeben hatte an Informationen und dem, was ich jetzt plötzlich gesehen hab.*

Es ergaben sich automatisch Gespräche mit den Entwicklern über die jeweiligen Programmteile und über Änderungen daran. *Und ich befand mich schnell in der Diskussion mit den Entwicklern, wo ich mir immer anhören mußte, das ginge aus diesen und jenen Gründen nicht. Ohne daß ich genug Know-how und Kompetenz hatte, um dagegen halten zu können und zu sagen: 'Das machen wir aber so!'*

Das Testen resultierte in einem Textdokument, welches QS[Qualitätssicherungs]-Feststellungen genannt wurde und in dem die einzelnen Fehler und Verbesserungsvorschläge aufgeführt wurden. *Auf jeden Fall hatte ich plötzlich das Gefühl, ich kann hier ein bißchen Einfluß nehmen. Die QS-Feststellungen wurden immer länger und da standen Sachen drin, die dann doch immer wieder abgearbeitet wurden, zum großen Teil. Wo ich also tatsächlich das Gefühl hatte, ich leiste sinnvolle Arbeit. Das Gefühl hatte ich anfangs nicht.*

Der Schulungsleiter gab während dieser Zeit den Anwender den Stand der Entwicklung weiter und es gab neue Anforderungen oder Änderungswünsche. Allerdings tauchten jetzt zwei Bereiche auf, die die Bedeutung zusätzlicher Anforderungen relativierten: Auslieferungstermin und Kosten. *Da wurde mir auf einmal das Thema Kosten bewußt. Daß ich plötzlich feststellte, das ein Angebot abgegeben wurde für ein System, daß Hotelzimmer verwalten und buchen konnte und nicht ein komplettes Hotelreservierungssystem, welches wir jetzt plötzlich als Vision vor uns sehen und was wir gerne haben würden. Ich hatte auch zwischendurch mitgekriegt, daß die Anforderungen immer mehr werden. Also, jetzt halte dich da zurück und bringe nicht noch zusätzliche Anforderungen mit hinein, die schön wären, aber am Leistungsvermögen des Programms nichts ändern würden. Es gibt Leute, die reduzieren das auf eine Design-Frage, aber für mich ist es immer mehr, für mich ist es auch die Qualität des Arbeitsplatzes, das Arbeiten des normalen Anwenders. Und da fing ich an, schon heftige Kompromisse zu machen.*

Der Auslieferungstermin rückte näher und wurde mehrmals verschoben. *Und dann kam dieser Augusttermin immer näher und das waren so meine ersten Erfahrungen damit, daß Termine nicht eingehalten wurden. Das war vielleicht noch nicht mal der Hammer, aber das darüber nicht informiert wurde, daß Termine nicht gehalten werden, das fand ich viel schlimmer.*

Beim Auftraggeber begannen die Vorbereitungen für den Einsatz der Software. Dabei stellten sich diverse Schwierigkeiten ein: Die Niederlassung, die in der selben Stadt wie das Softwarehaus liegt, war gegenüber den anderen Niederlassungen bevorteilt. Außerdem gab es beim Auftraggeber immer noch keinen eindeutigen Ansprechpartner. *Da gab es eine Kommunikationslücke. Hier ging das noch halbwegs, aber unsere anderen Niederlassungen waren vollkommen außen vor. Die haben eine Demonstration mitgekriegt und hörten dann nichts mehr. Es war ein Fehler, die nicht zwischendurch mal*

anzurufen. Wobei noch unklar war, wer denn überhaupt dafür verantwortlich ist. Es ist nie ein Verantwortlicher festgelegt worden, der das bei uns koordiniert.

Nach Auslieferung erster Programmteile begannen beim Auftraggeber die Schulungen. *Wir haben ein überregionales Treffen gemacht für Schulungen, die über drei Tage gehen sollten. Die Schulung endete in einem absoluten Fiasko, daß wir am ersten Tag so gut wie überhaupt nichts machen konnten, 20000mal abgestürzt waren, daß Programmteile nicht funktionierten. Also, es war voll ätzend, ekelhaft. Wir sind dann nur mit Ach und Krach durchgekommen. Drei Tage später sollten die anderen Mitarbeiter geschult werden. Aber das konnte man mit dem Programm, wie es da stand, getrost vergessen.*

Nachdem Fehlerbeseitigungen stattgefunden hatten, begann man mit dem Produktionseinsatz der Software. Dies ging ohne die erwarteten Probleme vor sich. *Und trotzdem: der 1.Tag des neues Systems, es tauchten zwar Fehler auf, aber es konnten Aufträge gebucht werden. Das war überraschend.*

Die internen Probleme beim Auftraggeber hielten in dieser Anfangsphase allerdings weiterhin an. *Plötzlich stellt man fest, daß wir die falschen Printserver gekauft hatten. Es dauerte dann wochenlang, bis die anderen Niederlassungen drucken konnten. Bei uns wurde die Nervosität größer, weil noch immer Fehler auftraten. Die anderen beschwerten sich, daß sie nicht drucken konnten, es gab keine Listen, keine Auswertungen. Die Geschäftsführung hatte ein Verkaufsziel gesetzt und das erschien damals ziemlich aussichtslos. Mittlerweile könnte es klappen. Es sieht ein bißchen besser aus.*

Das Ungleichgewicht zwischen den verschiedenen Niederlassungen blieb auch weiterhin bestehen. *Dann wußten wir nicht, wer unser Ansprechpartner ist bei Problemen. Es wußte niemand Bescheid, also mußten wir direkt in der Softwarefirma anrufen. Wir konnten zwar direkt zu den Entwicklern gehen, aber unsere Kollegen der anderen Niederlassungen konnten das nicht und haben uns mit Fragen genervt bis zum geht-nicht-mehr.*

Nach einer anfänglichen Zeit von Schwierigkeiten normalisierte sich der Systembetrieb. Die Anwender waren inzwischen mit dem Programm und den bestehenden Unhandlichkeiten besser vertraut. *Am Anfang haben nur wir damit gearbeitet. Wir haben unsere Mitarbeiter gar nicht drangelassen. Weil die völlig demotiviert gewesen wären. Mittlerweile finden die das ein ganz tolles Werkzeug und stehen voll dahinter und kommen auch mit den Kleinigkeiten und Schwierigkeiten im Alltag klar. Man hat sich sehr schnell an die positiven Seiten gewöhnt und für die negativen Seiten findest du in den meisten Fällen Workarounds - unser Schlagwort 1993 - die du im nachhinein so selbstverständlich einsetzt, daß du sie gar nicht mehr als Workaround empfindest.*

Viele Problematiken tauchten erst auf, als die Anwender mit dem Programm arbeiteten. *Wir haben soviele Sachen erst hinterher entdeckt. Z.B. daß wir mit Zimmereinheiten und nicht mit verschiedenen Zimmertypen umgehen. Es ist dem Hotel völlig egal, ob wir ein Zimmer im Kontingent als Einzel- oder Doppelzimmer verkaufen. Das war mir auch schon vorher klar, aber ich habe einfach nicht daran gedacht. Wir werden gefragt: "Welche Zimmertypen gibt es?" und wir beantworten die Frage natürlich. Was das für Konsequenzen hat, darüber macht man sich keine Gedanken.*

Nach Produktionsbeginn beginnt die Zeit der Fehlerverbesserungen und Erweiterungen. Dazu werden jetzt auch wieder Anwender zu den Control-Board-Meetings hinzugezogen. *Als wir wieder dazukamen, dienten die CBM's ja schon dazu, auftretende Problematiken zu besprechen. Da haben wir dann einiges an Problemen vorgebracht. Aber da wurde eigentlich nur noch in Manntagen gerechnet. Das war echt frustrierend. Wir versuchten klarzumachen, daß*

dieses oder jenes Problem sehr gravierend ist, weil es uns in unser tagtäglichen Arbeit hemmt. Und dann wurde nur auf die Kosten geschaut. Hauptsache, es ist nicht so teuer.

Das Verfahren, Fehler und Verbesserungen zu melden, wird in der Folgezeit mit Hilfe eines Formulars geregelt, welche die Anwender ausfüllen müssen und die an einer Stelle beim Auftraggeber gesammelt werden, um eventuell an das Softwarehaus weitergegeben zu werden. *Da haben wir dann eine nette Problemmeldung bekommen, ein Formular, was wir auszufüllen hatten. Und niemand wußte, wie wir es auszufüllen hatten. Da hätte man eine Problemmeldung über die Problemmeldung schreiben müssen. Es ist ja im Grunde nicht schlecht, so etwas ein bißchen zu formalisieren. Aber es hätte ein Anwenderformular sein sollen und kein Programmiererformular. Obwohl ich schon seit vier Monaten mit diesem Formular arbeite, kann ich an einigen Stellen nicht sagen, was ich da eintragen muß.*

Zwar hat sich das Verfahren, Fehler zu melden, nach einigen Änderungen in der Anfangszeit stabilisiert, jedoch werden Fehlermeldungen oft überhaupt nicht mehr ausgefüllt. Entweder ist das Ausfüllen zu aufwendig oder es fehlt die Rückmeldung oder es ist unklar, ob Fehler überhaupt noch behoben werden sollen, weil das Hotelsystem vielleicht durch ein integriertes System abgelöst werden soll. *So wichtig ist mir dieses Formular nicht, ich will aktuell loswerden, was mich bedrückt und nicht tausend Felder ausfüllen müssen. Daß die Fehlermeldungen nicht mehr geschrieben werden, ist sicherlich einfach eine Resignation. Weil auch der Fehler gemacht wird, daß keine Rückmeldung kommt. Ich hab bisher von genau einem Formular eine Rückmeldung bekommen, bei unzähligen, die ich ausgefüllt habe.*

Die Anwender haben zunächst hauptsächlich die Softwarefirma für die Schwierigkeiten der Anfangsphase verantwortlich gemacht. *Man kann ja auch nicht erwarten, daß die von der Softwarefirma das richtig machen würden, die kennen ja unsere Sprache, den Hotel-Jargon, gar nicht.* Später wurden auch die *hausgemachten* Probleme in Betracht gezogen. *Irgendwann im Juli hatte ich das Gefühl, daß es absolut chaotisch war. Da wurde überhaupt keine Planung eingehalten. Wenn man da Schuld suchen wollte, dann wohl eher bei uns. Mit der Softwarefirma hat das weniger zu tun. Es waren Fristen da, die eingehalten werden mußten, aber von der Softwarefirma nicht eingehalten werden konnten..*

Das Programm ist inzwischen nicht mehr aus dem Arbeitsalltag der Anwender wegzudenken. Eine Bewertung des Programms fällt bei den Anwendern unterschiedlich aus. Für die einen ist es *ein Werkzeug, das nicht paßt,* andere können relativ gut damit arbeiten. *Es ist schon ein komfortables System im Vergleich zu den Reservierungssystemen in den Hotels.* Übereinstimmung herrscht, daß das Programm Standardfälle gut abdeckt, jedoch bei Sonderfällen oft nur umständliche Lösungswege anbietet. Das Aussehen der vom Programm erzeugten Dokumente, die an Kunden und Hotels verschickt werden (z.B. Auftragsbestätigung), ist bis heute Gegenstand von Kritik. *Womit halt viele Hotels Schwierigkeiten haben, sind diese Dokumente. Mittlerweile ist es soweit, daß wir hoffen, daß uns nie ein Hotel anruft und uns fragt, was wir damit meinen. Weil es völlig verwirrend ist, was auf diesen Dokumenten steht.*

Bei vielen Anwendern scheint eine skeptische Grundhaltung gegenüber dem Softwarehaus geblieben zu sein, obwohl das System unter wirtschaftlichen Aspekten von allen als Erfolg bewertet wird, da es stabil läuft und die Verkaufszahlen gesteigert werden konnten. Insgesamt stellt das System für die meisten einen Schritt in Richtung auf ein integriertes System dar. *Am Anfang habe ich es als Endlösung gesehen und mit der Zeit wurde klar, daß es nur der erste Schritt war. Von daher kann*

ich gar nicht genau sagen, ob es ein Erfolg war. Es ist bestimmt eine gute Grundlage. Es war auch ein Erfolg für mich, mit den ganzen Abstrichen, die ich schon erzählt habe. Es war total interessant, mitzukriegen, wie so ein Programm entwickelt wird. Und daran mitzuwirken, war auch total interessant.

Zusammenfassung der Anwenderperspektive

Aus der Sicht der Anwender sind drei Schritte im Projekt von Bedeutung: Die Anforderungsermittlung, die Schulungen und das Arbeiten mit dem echten System. In der Zeit zwischen Anforderungsermittlung und Auslieferung des Systems gingen die späteren Anwender ihrer normalen Alltagsarbeit nach und hatten wenig Anteil am Projektvorgang. Die Ermittlung der Anforderungen wurde von den Anwendern und Anwendervertretern als sehr fruchtbare Zeit erlebt. Sie bedeutete zwar z.T. viel Arbeit, jedoch empfanden es die Anwender als sehr positiv, in den Prozeß der Softwareentwicklung einbezogen zu sein und ihre Anforderungen an das System direkt oder indirekt nennen zu können. Auch scheinen die Systemvisionen auf viele der Anwender einen sehr positiven Eindruck gemacht zu haben. Dieser ersten Phase steht dann die Zeit der Einführung des Systems stimmungsmäßig diametral gegenüber. Die Anwender waren vorwiegend über die Instabilitäten des Systems aufgebracht und haben diese Zeit bis heute in sehr negativer Erinnerung. Es hat sich bei vielen die Meinung festgesetzt, daß es für Softwareentwicklungen typisch ist, daß ein Programm verspätet ausgeliefert wird und dann relativ unfertig ist und nur unzuverlässig funktioniert und somit viel Ärger und Mühe bereitet.

Bewertung

Für die spätere Analyse des Projektverlaufes ist es meiner Meinung nach wichtig, auf den Bruch in der Vorgehensweise hinzuweisen, der sich bereits in der Darstellung der Entwicklerperspektive angedeutet hatte. Die Ermittlung der Anforderungen und das Zeigen der Systemvisionen hatte bei den Anwendern das Gefühl erzeugt, am Prozeß der Softwareentwicklung gleichberechtigt teilzuhaben. Nach Beendigung der Anforderungsermittlung wurde diese Einbezogenheit, u.a. auf Grund des Termindrucks, nicht weiter aufrechterhalten. Ausgehend von der ersten Phase hatte sich bei den Anwendern eine hohe Erwartungshaltung aufgebaut, die nicht nur auf die Produktqualität bezogen war, sondern auch auf die Chancen einer intensiven Mitarbeit am entstehenden System. Die Schulungen scheinen diese Erwartungen dann in empfindlichen Maße enttäuscht zu haben.

In Bezug auf die Analysephase wird bei den Anwendern und Anwendervertretern ein Rollenverständnis deutlich, daß zu dem der Entwickler “paßt”: Auch die Anwender sehen sich eher in der Rolle von denjenigen, die im Zuge der Anforderungsermittlung ihre Wünsche und Vorstellungen den Entwicklern nennen, damit diese daraus ein Softwaresystem entwickeln. Mit der Erhebung der Anforderungen erscheint die Aufgabe der Anwender erledigt zu sein, was sich im Projektverlauf als Trugschluß herausstellte.

Es erscheint mir charakteristisch, daß viele Probleme erst auftauchten, als das System im Arbeitsalltag eingesetzt wurde. Bemerkenswert ist auch, daß in der Zeit vor Auslieferung ausschließlich Abteilungsleiter in den CBM-Treffen die neu

auftretende Anforderungen besprachen und daß Anwender erst wieder in diese Treffen einbezogen wurden, als das System ausgeliefert war, und es somit konkrete Dinge gab, über die man sprechen konnte. Dies deutet die Bedeutung von ausführbaren Programmteilen für die Evaluierung und Qualitätssicherung durch den Anwender an. Die Anwender wurden zu diesen Treffen hauptsächlich deshalb hinzugezogen, weil es um das Besprechen von Fehlern des Systems ging. D.h. aber, daß die Anwender dabei eindeutig in der Rolle von Kritikern des Systems waren und nicht mehr in der Rolle von am System auf positive Weise mitgestaltenden Teilnehmern des Entwicklungsprozesses. Auch scheinen in den CBM's finanzielle und terminliche Überlegungen mit der Zeit immer stärker in den Vordergrund gerückt zu sein, so daß gegenüber neuen Anforderungen und Änderungswünsche trotz ihrer Berechtigung wenig Offenheit bestand. Dies scheint den anfänglichen Enthusiasmus der Anwender und Anwendervertreter teilweise frustriert und gebremst zu haben.

Die Darstellung der Anwenderperspektive deutet die Bedeutung des Leistungsumfangs als Dokument im Entwicklungsprozeß an. Für die Entwickler war er *die Bibel des Projektes*, für die Anwender zunächst das einziges Dokument, in dem ihre Anforderungen festgehalten wurden, da die Systemvisionen erst ab der Hälfte der Analysephase gezeigt wurden. Gerade für die späteren Anwender scheint der Leistungsumfang nur schwer verständlich und deshalb das Aussehen des späteren Systems nicht antizipierbar gewesen zu sein. Nicht nur die Größe des Dokuments oder der rein textuelle Charakter spielten dabei eine Rolle, sondern auch die Sprache, in der der Leistungsumfang verfaßt war. Diese entsprach nur teilweise der Fachsprache der Anwendungswelt und war somit für den Anwender oft unverständlich, was meiner Ansicht nach mitverantwortlich dafür war, daß sich nicht oder zu wenig mit dem Inhalt des Leistungsumfangs auseinandergesetzt wurde. Damit bestand wenig Aussicht, daß die eigentlichen Experten der Anwendungswelt für die erhobenen Anforderungen frühzeitig Qualitätssicherung durchführen konnten. Die Verständlichkeit der eingesetzten Dokumente scheint für eine möglichst frühzeitige Qualitätssicherung von entscheidender Bedeutung zu sein. In diesem Zusammenhang könnte es auch hilfreich sein, möglichst frühzeitig anhand konkreter ausführbarer Programmteile diskutieren zu können und nicht erst, wenn das fertige System in Produktion arbeitet.

2.5.3. **Perspektive des Auftraggeber-Managements**

Der Geschäftsführer des Auftraggebers war über den gesamten Projektverlauf stark am Projekt beteiligt. Er nahm an allen Gesprächen der Analysephase und den im Verlauf des Projekts stattfindenden Control-Board-Meetings teil.

"Es war ein Erfolg. Ein Kratzer ist jedoch am Softwarehaus geblieben."

Durch die angestrebte Erweiterung des Unternehmens steht man beim Auftraggeber vor der schwierigen Frage, wie eine mögliche Unterstützung der neuen Geschäftsbereiche aussehen könnte. Da zunächst Softwareunterstützung im Bereich der Hotelreservierung benötigt wird, kann der Auftraggeber an einem eigenständigen und überschaubaren Teilbereich die Zusammenarbeit mit dem Softwarehaus und den Einsatz neuer EDV-Techniken erproben.

Vom Auftraggeber wird die Vorgehensweise des Softwarehauses im Projekt und besonders die Rolle des Projektleiters sehr positiv eingeschätzt. *Was mir gut gefällt, daß wirklich zugehört wird, aber auch kritisch hinterfragt wird. Daß es auch im Leistungsumfang sehr strukturiert zusammengestellt wird und daß man sich auf Seiten des Softwarehauses tiefergehende Gedanken macht, was wirklich gemeint ist. Es liegt auch ganz deutlich an der Person des Projektleiters, der das einfach kann.*

Auch der Einsatz der Systemvisionen wurde vom Auftraggeber begrüßt, obwohl auch die Grenzen erkannt werden. *Das hat mir sehr gut gefallen. Wenn man das auch noch nicht ausprogrammiert hat, so kann man doch erläutern, welche Bedeutung dieses oder jenes Feld hat. Man hat es ein bißchen plastischer vor Augen und kann schnell auch konkret diskutieren… Was aber dabei zu kurz gekommen ist, sich nicht nur anhand der Oberfläche damit zu beschäftigen, sondern auch mit Gesamtabläufen zu arbeiten. Man hat sich zu sehr an den Oberflächen festgehalten, weil es das einzige war, was man sehen konnte.*

Nach der Gesamtvorstellung des Leistungsumfang kamen weiterhin neue Anforderungen hinzu. *Den Leistungsumfang haben wir orientiert an den bis dahin vorliegenden Erfahrungen, wie wir manuell Übernachtungen verkauft haben. Wir müssen im nachhinein feststellen, daß wir schon im Laufe des Projektes anreichern mußten mit anderen Funktionen, die notwendig waren. Das gipfelte darin, daß wir den Leistungsumfang, den wir im Mai'93 vorliegen hatten, dann jeden Monat mehr oder weniger überarbeiten mußten. Die letzten wesentlichen Anforderungen hatten wir ca. 1-2 Monate vor Inbetriebnahme des Systems.*

Die neuen Anforderungen mußten in größerer Runde abgestimmt werden. Dazu wurden an den CBM's auch Abteilungsleiter beteiligt, die vorher nur indirekt vom Projekt Kenntnis hatten. *Es gab zu der Zeit aufgeblähte CBM's, wo auch Leute aus den anderen Niederlassungen und verschiedenen Abteilungen mit dabei waren. Und da kamen dann neue Aspekte hinzu. Die haben zwar den Gesamtablauf nicht gekippt, aber doch an manchen Stellen erweitert. Mit der Zeit wurde ich ein wenig nervös, wenn über weitere Funktionalität diskutiert wurde, weil unklar war, welche Konsequenzen das für andere Teile des Produktes haben würde. Der Gesamtzusammenhang geht schnell verloren.*

Anfangs hatte man sich als Termin den 1.August gesetzt. Beim Auftraggeber hatte man intern bereits mit einer Verspätung dieses Termins gerechnet. Im Prinzip sollte das Weihnachtsgeschäft mit dem Programm abwickelt werden und man hatte einen frühen Termin gewählt, um genug Zeit für die Einarbeitung zu

haben. *Und wenn es bis Ende August etwas wird, dann bin ich schon zufrieden. Aber selbst Ende August wurde es nichts, denn es kamen neue Dinge, die eingebaut werden mußten. Sehr häßlich wurde es dann, als die Qualitätssicherungsphase und Einarbeitungsphase dann immer kürzer und gegen Null ging. Den Puffer, den wir hatten, den haben wir immer zu Lasten der Qualitätssicherung und Schulung aufgebraucht.*

Die Schulungsphase wird vom Auftraggeber als negativ gesehen. *Da kamen auch ständig neue Auslieferungen zu uns. Da waren gerade neue Arbeitsabläufe besprochen worden, die erst erprobt werden mußten und da kommt die Software, die nicht ausgetestet ist und nicht funktioniert. Auch, wenn es nur Banalitäten sind, für die Leute funktioniert es nicht. Da haben wir jede Menge Streß gehabt. Es gab auch noch ein Problem: Auf Seiten des Softwarehauses wurde eine neue Version der Datenbank eingesetzt. Man wollte etwas Gutes damit tun, aber das Ding war fehlerhaft und das kam dann auch noch in der Test- und Schulungsphase. Die konnten wir nicht fortsetzen, obwohl wir Leute aus den anderen Niederlassungen da hatten.*

Übereinstimmung herrscht, daß die Einführungsphase dem Softwarehaus besonders bei den Anwendern viel an Image gekostet hat. *Was an Mißerfolg geblieben ist, z.B. die negative Einschätzung bei der breiten Masse der Leute in den unterschiedlichen Niederlassungen. Dieser Kratzer ist an der Softwarefirma dran, auch wenn viele Dinge hausgemacht waren.*

Ein weiterer Grund für die Probleme liegt nach Einschätzung des Auftraggebers an der Tatsache, daß man auf dem Gebiet Hotelreservierung zu wenig Erfahrung hatte. *Wenn man, wie hier, völliges Neuland betritt, dann muß man sich noch mehr einfallen lassen. Daß man es im Vorwege noch tiefer ergründet. Denn ich glaube, wenn wir uns vor 10 Monaten schon über die Dinge klar gewesen wären, die uns jetzt bewußt sind, dann hätten wir vielleicht in der Analysephase noch ein paar Wochen draufgelegt, hätten weitere Gespräche mit Externen geführt und hätten dann in der Implementierungsphase und Einführungsphase etliches an Aufwand gespart. So haben wir zusätzlichen Aufwand gehabt.*

Letztendlich wird das Projekt vom Auftraggeber als Erfolg eingestuft und die Zusammenarbeit mit dem Softwarehaus positiv bewertet. *Insgesamt war es ein Erfolg. Das Volumen, was wir jetzt verkaufen, dazu wären wir ohne das System nicht in der Lage gewesen… Ich kenne aus der jüngeren Vergangenheit viele andere Softwarehäuser. Ich glaube nicht, daß es viele Softwarehäuser gibt, mit denen wir so gut und so schnell zum Ziel gekommen wären. Trotz aller negativer Dinge.*

Zusammenfassung der Perspektive des Auftraggeber-Managements

Mit der Erstellung des Hotelsystems wollte der Auftraggeber die Zusammenarbeit mit dem Softwarehaus und den Einsatz neuer EDV-Techniken testen. Dieser Test ist aus Sicht des Auftraggebers zufriedenstellend verlaufen, da das Hotelsystem zügig entwickelt wurde, akzeptabel arbeitet und die Verkaufszahlen erheblich gesteigert werden konnten. Der Auftraggeber beurteilt sowohl die Vorgehensweise des Softwarehauses als auch die dort für das Projekt verantwortlichen Personen in positiver Weise. Als projektspezifische Schwierigkeiten sieht der Auftraggeber sowohl die mangelnde Erfahrung auf dem Gebiet der Hotelreservierung, als auch die ständig zunehmenden Anforderungen im Verlaufe des Projektes an. Im nachhinein hat man durch die Arbeit mit dem Hotelsystem eine Reihe von neuen Erfahrungen gesammelt, die, hätte man sie schon zu Beginn des Projektes gehabt, zu einer Verlängerung der anfänglichen Analysephase hätten führen können. Neben den nicht einfachen Um-

strukturierungen im Hause des Auftraggebers, wird vorwiegend die Einführungsphase als problematisch gesehen, wo ein nur ungenügend getestetes und instabiles System für viel Ärger bei den Anwendern sorgte und einen Imageverlust für das Softwarehaus zur Folge hatte. Ohne die Schwierigkeiten zu vernachlässigen, wird das Projekt vom Management des Auftraggebers hauptsächlich in seinem strategischen Charakter gesehen und von daher als ein Erfolg eingestuft.

Bewertung

Zunächst erscheint mir wichtig, daß der Auftraggeber mit dem Projekt z.T. strategische Ziele verfolgte. Insofern ist zwar das Auftraggeber-Management mit der Vorgehensweise und dem Resultat des Projektes zufrieden, nicht jedoch der Großteil der Anwender oder Abteilungsleiter. Dies scheint daraufhin zu deuten, daß der strategische Charakter des Projektes entweder den in der Hierarchie tieferstehenden Mitarbeitern zu wenig vermittelt wurde oder die Schwierigkeiten mit dem System auch unter Kenntnis der strategischen Zielsetzung nur schwer erträglich waren.

Auffällig an der Darstellung der Managementperspektive ist auch, daß sich anscheinend die Aufgabe der CBM-Treffen im Projektverlauf erheblich veränderte. Zunächst sollten die Treffen der Projektkontrolle dienen und hatten dementsprechend eine betont kleine Teilnehmerzahl im Vergleich zu den ersten großen Gesprächsrunden. Um neue Anforderungen abzuklären, mußten die CBM's im Verlaufe des Projektes dann aber z.T. erheblich erweitert werden, womit sie einen ähnlichen Charakter bekamen, wie die ersten großen Treffen der Analysephase. Dies deutet auf den Widerspruch hin, daß die Ermittlung der Anforderungen zwar offiziell abgeschlossen war, jedoch faktisch im Verlaufe des Projektes weiterging.

Daß das Auftraggeber-Management aus den Erfahrungen im Projekt z.B. den Schluß zieht, in Zukunft mehr Zeit für die Ermittlung der Anforderungen vorzusehen, ist sehr gut zu verstehen. Die Anforderungen, die noch kurz vor Auslieferung und nach Produktionsbeginn auftauchten, konnten oft nur unter Schwierigkeiten, bzw. finanziellen Belastungen in das System aufgenommen werden. Für das Management stellt sich die Frage, warum diese Anforderungen nicht schon früher von den entsprechenden Personen geäußert wurden. Da wenig Erfahrung mit dem Hotelverkauf vorlag, erscheint mehr Zeit für die Analysephase als eine mögliche Lösung. Ich will bereits an dieser Stelle hierzu eine andere Sichtweise anbieten: Mir erscheinen die vielen, erst im Verlaufe des Projektes geäußerten Anforderungen eher ein Resultat des Entwicklungsprozesses selber zu sein, eines Lernprozesses, der auf beiden Seiten erst mit Beginn des Projektes einsetzte und über das ganze Projekt anhielt. Deshalb vermute ich, daß viele der später geäußerten Anforderungen gerade *nicht* im Vorwege innerhalb einer längeren Analysephase hätten ermittelt werden können, weil zu Anfang des Projektes eine viel zu ungenaue Vorstellung der späteren Arbeitszusammenhänge bestanden hatte. Insbesondere konnten die Anforderungen, die dann später in der täglichen Arbeit mit dem System auftauchten, nicht von vorne herein antizipiert werden.

2.5.4. Perspektive des Softwarehaus-Managements

Vom Softwarehaus-Management waren der Geschäftsführer und der Projektleiter mit dem Projekt befaßt. Der Geschäftsführer hatte wesentlichen Anteil in der Aquirierung des Projektes und hielt während des Projektverlaufes auf der Managementebene Kontakte zum Auftraggeber. Der Projektleiter war auf Seiten des Softwarehauses die maßgebliche Person bei der Durchführung und praktischen Umsetzung des Projektes. Beide Sichtweisen werden in der folgenden Darstellung herausgearbeitet.

"Wir haben eine Menge zusätzlicher Anforderungen dazubekommen."

Das Projekt hatte für das Softwarehaus strategischen Charakter. *Es war ja eine Art "Submarine-Projekt", wie man das so schön nennt.* Das Softwarehaus mußte beweisen, daß es auch unter extremen zeitlichen Beschränkungen ein kommerziell nutzbares System entwickeln konnte. *Wir sind diese Geschichte mit einem völlig bewußten Risiko eingegangen. Es war uns völlig klar, daß wir es zu dem Gelde überhaupt nicht schaffen können, sondern daß es uns wesentlich mehr kosten wird.*

Zusätzlich ging das Softwarehaus das Risiko ein, ein teilweise neues und unerprobtes Tooling (Programmiersprache und Klassenbibliothek) einzusetzen. *Objektorientierung war vertraut, aber das Tool als solches nicht. Es war klar, daß wir damit zunächst einiges an Mehraufwand haben würden… Die fehlende Vertrautheit mit dem Tooling war sicher ein Problem. Das hat uns allerlei Mühe gemacht und zu suboptimalen Dingen geführt. Denn bei den Entwickler war eine Diskrepanz zwischen dem, was sie sich vorstellten und was sie realisieren konnten.*

Das System sollte in kurzer Zeit fertiggestellt werden. Deshalb mußte auch die Analysephase relativ schnell durchgeführt werden. Das Softwarehaus bemühte sich deshalb, viele verschiedene Abteilungen in die ersten Gespräche einzubeziehen, um möglichst schnell die benötigten Informationen sammeln zu können. *Es war insofern gut gegangen, weil vieles in den Gesprächen herauskam. Es ist in anderer Hinsicht schlecht gelaufen, weil zu viele Leute dabei waren, aus zu unterschiedlichen Bereichen. Es gibt sicherlich Situationen, da müssen die Leute zusammen befragt werden, aber für die Gesamterhebung der Anforderungen war das unangemessen. Aber man muß auch sehen, daß die analytische Phase sehr schnell über die Bühne gehen mußte. Man mußte es so gestalten, daß man gerade das herauskriegt, was man braucht, daß man damit nicht weiter in die Tiefe geht. Diesen Charakter hat ja leider Gottes das ganze Projekt gehabt.*

Vom Softwarehaus werden die Gespräche der Analysephase mit einigen Abstrichen als positiv angesehen. *Hat auch den Entwicklern Spaß gemacht. War nicht effektiv, denn es hat dazu geführt, daß wir viel mehr zu tun gekriegt haben, als wir ursprünglich geplant hatten. Es hat seine positiven wie negativen Seiten. Von den Benutzern her war es positiv. Wir haben zum ersten Mal ein Projekt in* ***dieser Tiefe*** *in einer Vorphase im Benutzerdialog gemacht. Das hat dazu geführt, daß eine positive Erwartungshaltung da war, die aber auch sehr hoch gesteckt war.*

Nach Abschluß der ersten Phase wurden weiterhin neue Anforderungen diskutiert, die dem Softwarehaus insofern Schwierigkeiten bereiteten, weil Design und Implementierung bereits begonnen hatten und die Einhaltung des zunächst vereinbarten Termins immer unwahrscheinlicher wurde. *Das müssen wir bei diesem Projekt überhaupt als roten Faden sehen: Daß wir eine Menge zusätzlicher Anforderungen dazubekommen haben.*

Besonders die Zeit unmittelbar vor den ersten Auslieferungen gestaltet sich in höchstem Maße schwierig, weil Auslieferungstermine häufig verschoben werden mußten. *Wir haben ja einen richtigen Eiertanz aufgeführt, zum Ende, zur Auslieferung hin. Es ist nicht mit Sicherheit zu sagen, was noch hätte passieren können. Im Nachhinein kann man wahrscheinlich sagen, daß wir relativ gelassen hätten auftreten können. Also im Juli/August hätten wir sagen können, daß wir im November fertig sind. Nachträglich kann man das sagen, aber nicht zum damaligen Zeitpunkt. In der jeweiligen Situation sieht es ganz anders aus.*

Mit der Einführungs- und Schulungsphase fangen dann aus Sicht des Softwarehauses die eigentlichen Probleme an. *Ich hab das Gefühl, daß es stimmungsmäßig erst ein Problem gab, als wir unsere Auslieferung mit den Schulungen beim Kunden machten. Und alles mögliche dann nicht ging, das Programm abstürzte, usw.. Das war eine heikle Zeit. Da gab es ein paar Momente, da hatte ich das Gefühl, daß wir an der Kippe stehen, daß uns alles um die Ohren fliegt und damit alles vorbei ist.*

Als eine mögliche Erklärung für die Probleme der Einführungsphase wird vom Softwarehaus auch die hohe Erwartungshaltung der Anwender gesehen. *Wir haben die Erwartungshaltung gesteigert durch die Art, wie wir in der Analysephase drangegangen sind, die Benutzer zu involvieren, etwas von Visionen zu erzählen, prototypische Dinge zu bauen. Wir haben dadurch zusätzliche Funktionalität und Komplexität eingebaut. Die Probleme, die wir in der Einführungsphase hatten, werden als absolut negativ gesehen, obwohl wir sie nicht zu vertreten hatten, da die gravierensten Instabilitäten aus einem Sybase-Fehler resultierten.*
Bei den Anwendern war nicht dieser Forschergeist vorhanden wie in früheren Zeiten. Da war schnell Kritik da. D.h. man muß aufpassen, wenn man diesen Weg geht, wenn man Benutzer sehr tief involviert in die Vorphasen eines Projektes, daß man hinterher auch sehr vernünftige Ergebnisse erzielt. Die Erwartungshaltung muß auch erfüllt werden. Für die Zukunft müssen wir vorsichtiger darangehen. Das ist etwas, daran habe ich vorher auch nicht gedacht.

Eine weitere Erklärung für die Probleme liegt für das Softwarehaus im engen Zeitrahmen. *Die Schwierigkeiten bei der Einführungsphase lagen daran, daß - wie immer - die Zeit für das Projekt zu kurz war.* Am Termindruck lag es auch, daß keine prototypische Vorgehensweise gewählt wurde. *Prototyp in dem Sinne war es nie. Die Maßgabe war, ein produktives System zu machen, mit dem man Hotelbuchung abwickeln kann. Für einen Prototyp im Sinne einer evolutionären Entwicklung war die Zeit viel zu kurz. Evolutionäre Entwicklungen dauern. Das ist in meinen Augen wirtschaftlich nicht machbar.*

Die Einführungsphase bedeutete für die Softwarefirma einen erheblichen und nicht erwarteten Imageverlust, der jedoch im nachhinein (auch durch parallele, von diesem Projekt unabhängige Aktivitäten) aufgefangen werden konnte. *Es hat uns Image gekostet, mit Sicherheit. Aber das haben wir zu großen Teilen wieder gutmachen können. Weil es nämlich seit November stabil läuft. Mir kommt es auf ganz bestimmte Stellen dort im Unternehmen an. Vor kurzem hat sich der Wirtschaftsprüfer das System angeschaut und uns bestätigt, daß alles wunderbar und sauber abgewickelt wird. Das ist mir wichtig. Für die Anwender gibt es sicherlich die eine oder andere Unhandlichkeit, aber ich glaube, daß sie im großen und ganzen mit dem Programm zufrieden sind.*

Das System wird vom Softwarehaus als ein strategischer Erfolg bewertet. *Letztlich hat es vom Ziel her gebracht, was wir wollten: Das Ding läuft in Produktion, es verdient Geld, sogar recht gut und es hat in den Folgeauftrag gemündet, der ganz erheblichen Umfang und für uns strategische Bedeutung hat… Alles darauf zurückgeführt, daß wir trotz des ganzen Stresses, der zwischendurch war, ein System geliefert haben, mit dem es möglich war, einen erheblichen Umsatz zu machen. Da ist schon eine ganze Menge bei raus gekommen. Nie werde ich es hinkriegen, es wirklich* ***allen*** *recht zu machen. Dazu sind es zu unterschiedliche Menschen. Wenn du dir viele andere*

Anwendungen anschaust: die sind einfach massiv schlechter als das, was wir gemacht haben. Wir hatten dieses Projekt als Einstieg betrachtet und dann muß man leider auch Zugeständnisse machen.

Zusammenfassung der Perspektive des Softwarehaus-Managements

In der Einschätzung des Softwarehauses war das Projekt ein Risikoprojekt, bei dem das Softwarehaus seine Leistungsfähigkeit unter Beweis stellen wollte. Um eine langfristige und strategisch wichtige Zusammenarbeit mit dem Auftraggeber etablieren zu können, mußte das Softwarehaus zunächst einen beträchtlichen Einsatz erbringen, ohne genau zu wissen, ob sich dies im nachhinein lohnen würde. In der Anfangsphase des Projektes versuchte das Softwarehaus, die Vertreter des Auftraggebers möglichst eng in die Entwicklung einzubeziehen. Aus der Sicht des Managements wird diese Analysephase zwiespältig bewertet, weil zwar durch den Einsatz der gewählten Darstellungsformen eine positive Grundstimmung im Projekt erzeugt wurde, jedoch auf Seiten des Auftraggebers und speziell bei den späteren Anwendern eine hohe Erwartungshaltung an das System entstand, die dann später in der Einführungsphase nicht eingehalten werden konnte. Negativ wirkte sich in der Einführungsphase besonders ein Fehler in der Datenbanksoftware aus, den das Softwarehaus im Prinzip nicht zu vertreten hatte, der ihm jedoch angelastet wurde. Neben den engen Kostenbeschränkungen und dem Einsatz neuer und innovativer Entwicklungswerkzeuge sieht das Softwarehaus hauptsächlich die wachsenden Anforderungen und den zu kurzen Zeitrahmen als Hauptschwierigkeiten im Projektverlauf an. Positiv wird neben Projektabwicklung und -steuerung auch das Engagement und die Motivation der Mitarbeiter gewertet, so daß es trotz der Schwierigkeiten möglich war, ein lauffähiges System mit nicht allzu großer Verspätung auszuliefern und somit das Projekt als Erfolg abzuschließen. Obwohl die Anwender des Systems mit einigen Schwächen im System leben müssen, können sie nach Meinung des Softwarehaus-Managements mit dem Produkt weitestgehend zufrieden sein, auch weil es im Vergleich zu anderen Systemen sehr gut abschneidet. Insbesondere hat sich bei den Entscheidungsträger im Hause des Auftraggebers eine positive Sichtweise des Produktes und damit des Softwarehauses durchgesetzt. Zusammenfassend kann man sagen, daß sich das Wagnis für das Softwarehaus gelohnt hat, obwohl die im vorwege bereits pessimistisch geschätzten Kosten im Projekt beträchtlich überzogen wurden. Mit den Folgeaufträge, die das Softwarehaus vom Auftraggeber erhalten hat, konnten die finanziellen Einbußen und der Imageverlust aus der Einführungsphase ausgeglichen werden.

Bewertung

Das Softwarehaus versuchte trotz des engen Zeitrahmens, die Vertreter des Auftraggebers gerade in der Anfangsphase relativ eng in das Projekt zu integrierten und erprobte mit der Präsentation von Systemvisionen neue, benutzergerechtere Darstellungsformen. Aus der Sicht des Managements führte dies zu einer positiven Gesamtatmosphäre im Projekt, aber auch zu einer Fülle zusätzlicher Anforderungen und sehr hohen Erwartungen an das Produkt, die in der Einführungsphase enttäuscht wurden. Ich will in dieser Arbeit in Bezug auf

diesen Punkt einen anderen Standpunkt vertreten: Meiner Meinung lag die Enttäuschung der Anwender nicht nur daran, daß ihnen auf Grund des Zeitdrucks ein unfertiges System als fertig präsentiert wurde, sondern zum großen Teil auch daran, daß sie nach der ersten Phase nicht weiter an der Softwareentwicklung teilhatten. Eine starke Involvierung der Anwender und Anwendervertreter in den frühen Phasen eines Projektes ist meiner Meinung nach nur dann problematisch, wenn diese nicht kontinuierlich über den Projektverlauf fortgesetzt wird. Die Erwartungshaltung der Anwender war meiner Ansicht nach deshalb besonders groß, weil sie bis zur Auslieferung keinerlei ablauffähige Programmteile gesehen hatten und nicht frühzeitig in den Testbetrieb einbezogen waren, somit wenig Gelegenheit hatten, sich mit dem Produkt zu identifizieren und es als “ihr System” mit allen darin enthaltenen Fehlern zu betrachten.

Daß das Softwarehaus mit dem Projekt wichtige strategische und wirtschaftliche Ziele verfolgte, verdient deshalb Erwähnung, weil sich daraus im Projektverlauf erhebliche Zielkonflikte ergaben: Produktqualität vs. Kosten, Produktstabiltität vs. Auslieferungstermin, um nur einige zu nennen. Insbesondere scheint eine intensive und kontinuierliche Benutzerbeteilung unter einer betont strategischen Zielsetzung und einer daraus resultierenden engen Zeitbegrenzung nicht realisierbar gewesen zu sein.

2.6. **Zusammenfassung der Projektbeschreibung**

Das vorliegende Projekt wurde unter einer primär strategischen Zielsetzung durchgeführt. Der Auftraggeber wollte die Zusammenarbeit mit dem Softwarehaus erproben und das Softwarehaus wollte seine Leistungsfähigkeit unter Beweis stellen, um eine langfristige Zusammenarbeit mit dem Auftraggeber zu etablieren und den Auftrag für die Entwicklung eines integrierten Systems zu erhalten. Um seine Ziele zu erreichen, war das Softwarehaus nicht nur bereit, einen erheblichen finanziellen Verlust hinzunehmen, sondern war auch gezwungen, im Verlaufe des Projektes eine Fülle neuer Anforderungen ins System aufzunehmen. Letztlich wurde zu einem viel zu geringen Preis ein System erstellt, das wesentlich komplexer und leistungsstärker war, als zunächst geplant.

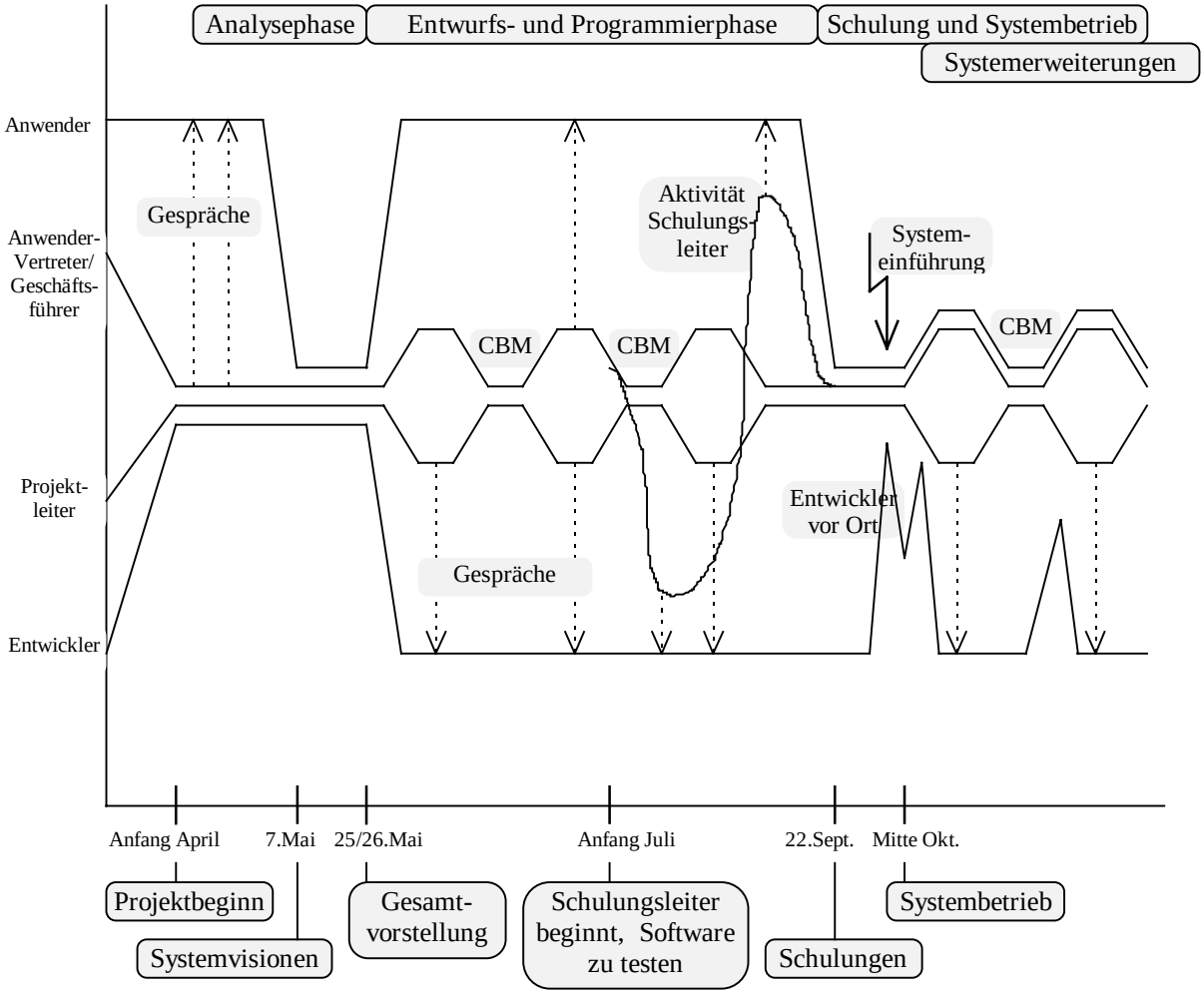
Von vielen der Beteiligten wird das Projekt aus den folgenden Gründen als ein Erfolg gewertet:

- Es wurde ein produktionsreifes System in extrem kurzer Zeit fertiggestellt.
- Das System hat eine hohe Stabilität.
- Der Auftraggeber konnte seine Verkaufszahlen im gewünschten Maße steigern.
- Mit dem Projekt wurden sowohl im Softwarehaus als auch beim Auftraggeber neue Hard- und Softwareplattformen, sowie neue Entwicklungstechniken erfolgreich eingeführt.
- Bis zur Einführungsphase verlief das Projekt in einer arbeitsintensiven und konstruktiven Atmosphäre.

- Die Zusammenarbeit von Softwarehaus und Auftraggeber hat sich durch das Projekt etabliert und gefestigt.
- Mit dem Projekt wurden die strategischen Zielsetzungen erreicht.

Aus der Projektbeschreibung ist deutlich geworden, daß das Projekt speziell auf der Managementebene auf Grund seiner strategischen Bedeutung und bei den Entwicklern auf Grund der technischen Leistung als ein Erfolg eingeschätzt wird. Bei denjenigen, die mit dem System arbeiten, wird die Frage nach dem Projekterfolg zwar überwiegend positiv, aber oft auch zögerlich und zwiespältig beantwortet, weil die Umstände der Systemeinführung und des frühen Systembetriebes immer noch präsent sind.

Wenn man sich fragt, welche Gruppen wann und auf welche Weise am Projekt beteiligt gewesen sind und wie sie untereinander kommuniziert haben, so bietet sich eine Darstellungsform an, die auf graphische Weise den Projektverlauf mit den relevanten Gruppen verdeutlicht. Solche Darstellungen können die Komplexität der realen Ereignisse natürlich nur vereinfachend zeigen, ermöglichen aber gerade dadurch einen veränderten Blickwinkel:



In der obigen Abbildung kann man erkennen, wie sich die Zusammenarbeit zwischen den vier Gruppen über den Projektverlauf hin entwickelte: Die

breiteste und direkteste Zusammenarbeit fand bis zur Gesamtvorstellung des Systems statt, wobei zu erkennen ist, daß die späteren Anwender erst etwa nach der Hälfte der Analysephase direkt mit Hilfe der Systemvisionen hinzugezogen wurden. Der bereits erwähnte Bruch in der Vorgehensweise ist nach der Gesamtvorstellung einerseits in der Entfernung zwischen Anwendern und Entwicklern, aber auch in der Zunahme informationsvermittelnder Aktivitäten (gestrichelte Linien) zu erkennen. In diesem Zusammenhang ist neben der Bedeutung des Projektleiters auch die Wichtigkeit des Schulungsleiters erkennen, der sowohl mit Anwendern als auch mit Entwicklern direkt sprach und damit nicht nur in dieser Grafik aus dem eher strukturierten Rahmen herausfiel. Mit der Systemeinführung ist dann wieder eine relativ enge Verbindung der beteiligten Gruppen erkennbar.

Auch wenn man die zusätzlichen Anforderungen als eine natürliche Folge des einsetzenden Lernprozesses ansieht, so waren sie trotzdem im Projekt ein erheblicher Störfaktor, die zusammen mit den finanziellen Beschränkungen eine schnellstmögliche Inbetriebnahme des Programms erschwerten. Mehrere Terminverschiebungen waren die logische Konsequenz, was zu einer ersten Verschlechterung der bis dahin sehr guten Atmosphäre führte. Die Stimmung scheint sich im Prinzip, mit Annäherung an die Inbetriebnahme stetig verschlechtert zu haben und erreichte in den Schulungen einen absoluten Tiefpunkt. Hier sei wieder auf die Grafik verwiesen: Aus der anfänglich engen Zusammenarbeit war neben einer hohe Erwartung auch eine Vertrauensbasis zwischen den beteiligten Gruppen und besonders auf Seiten der Anwender entstanden. Vor der Systemeinführung als größter Krise im Projekt lag eine lange Zeitspanne, in der gerade die Anwender sehr weit vom Projekt “entfernt” waren. Die anfänglich geschaffene Vertrauensbasis erwies sich als inzwischen geschwächt und war gerade in einer Zeit zu wenig vorhanden, in der es aufgrund des Konfliktreichtums darauf angekommen wäre, Verständnis für die Position der Gegenseite zu haben, anstatt überhöhte Erwartungen. Es war bezeichnend, daß die als sehr negativ empfundenen Instabilitäten des Systems aus einem Datenbankfehler resultierten, den das Softwarehaus im Prinzip gar nicht zu vertreten hatte und der ihm doch vollständig angelastet wurde.

Aus der Zeit der Systemeinführung resultierte bei breiten Teilen des Auftraggebers eine teilweise skeptische Haltung hinsichtlich zukünftiger Softwareentwicklungen, die auch noch ein halbes Jahr später spürbar und wirksam ist.

3. Vergleich zu anderen Softwareprojekten

Weltz und Ortmann haben 1992 eine Veröffentlichung vorgelegt, in der sie die Untersuchung von 46 Softwareentwicklungsprojekte beschreiben und auswerten ([WelOrt92]). Dabei sind im Vergleich zum vorliegenden Projekt eine Reihe von Ähnlichkeiten zu bemerken, die ich im folgenden näher erläutern will, wobei jeweils die Merkmale dieses Projektes den in Zitatform wiedergegebenen Ergebnissen von Weltz und Ortmann gegenübergestellt werden. Ziel dabei ist es, zu zeigen, daß das vorliegenden Projekt zu weiten Teilen in einer für Softwareprojekte durchaus typischen Weise verlaufen ist. In diesem Zusammenhang halte ich folgende Punkte für relevant:

- Phasenorientiertes Vorgehen
- Unterschätzung der Komplexität
- Kontinuierliches Auftreten neuer Anforderungen
- Überziehung der Kosten und der zugesicherten Termine
- Problematische Einführungsphase
- Keine kontinuierliche Anwenderbeteiligung
- Ungelöster Widerspruch zwischen Qualitätssicherung und Kosten
- Herausbildung von Schlüsselfiguren

Der Projektablauf sollte nach einem Phasenmodell abgewickelt werden, in dem zunächst die Anforderungen ermittelt und verabschiedet werden sollten. Es kamen aber während des Projektverlaufes ständig neue Anforderungen dazu, die eine solche Verabschiedung des Leistungsumfangs unmöglich machten. Weltz und Ortmann sagen dazu: *"Für die meisten Software-Projekte, die wir untersuchten, war ein Vorgehen nach dem Phasenschema geplant. Aber nur bei wenigen konnten die vorgesehene Phasenaufteilung und die Termin- und Budgetplanung auch eingehalten werden."*

Der Umfang der zu lösenden Aufgabe wurde im vorliegenden Projekt in erheblichem Maße unterschätzt. Erst im Verlaufe des Projektes wurde immer deutlicher, wieviel verschiedene Teilbereiche realisiert werden müssen, um den Anforderungen gerecht zu werden. Weltz et al. kommen in ihrer Studie zu einem ähnlichen Ergebnis: *"In diesem Zusammenhang ist auch ein Phänomen zu sehen, das sich in vielfacher Weise belastend in der Projektabwicklung auswirkte: die Unterschätzung der Komplexität des Vorhabens, die man erst im Lauf des Entwicklungsprozesses entdeckte."* Im weiteren sprechen die Autoren sogar von einer *"systematischen Unterschätzung der Entwicklungsaufgabe"*.

Abzulesen war die steigende Komplexität vor allem an dem gescheiterten Versuch, den Leistungsumfang am Projektanfang zu definieren und an der stetigen Zunahme neuer Anforderungen. Dazu Weltz et al.: *"Ein Schlüsselproblem … war die Erarbeitung der Anforderungsspezifikationen … Diese zu etablieren, erwies sich in den meisten Projekten als für alle Beteiligten überraschend mühsam… Nicht selten wurde auch die 'endgültige' Vorlage dann im weiteren Projektverlauf noch mehrmals in Frage gestellt, modifiziert und erweitert. In den meisten*

Projekten wurden die Verzögerungen, die schließlich zu der Überschreitung der ursprünglich veranschlagten Projektlaufzeit geführt hatten, nicht zuletzt auf den langwierigen Prozeß der Definition der Anforderungen zurückgeführt."

Beim Softwarehaus hatte man intern mit einer Kostenüberziehung gerechnet. Trotzdem fiel diese mit einem Faktor 2 bis 2_ wesentlich höher als erwartet aus. Die geplanten Auslieferungstermine wurden mehrmals überzogen. Ursprünglich sollte das Hotelsystem innerhalb von 4 Monaten fertiggestellt sein. Es lief schließlich in Produktion nach etwa 6 Monaten. Weltz et al. formulieren es so: "*Projektkosten und Termine - nur selten richtig geschätzt… Angesichts der großen Bedeutung einer realistischen Bestimmung des Projektvolumens überrascht die Nonchalance, mit der diese vielfach behandelt wurde."*

Eigentliche Anwender des Systems waren nicht kontinuierlich über den gesamten Projektverlauf im Entwicklungsprozeß einbezogen, sondern wurden häufig über Vermittler beteiligt (Abteilungsleiter, Geschäftsführung, Schulungsleiter). Auch dies scheint für Softwareprojekte, die heute durchgeführt werden, die Regel zu sein. "*Nur relativ selten begegneten wir wirklich tragfähigen offiziellen und institutionalisierten Ansätzen zur unmittelbaren Einbeziehung der Anwender in den Entwicklungsprozeß; vor allem kaum spürbar war … ein direkter Niederschlag der akademischen Diskussion um die Nutzer-partizipation und der vorliegenden Modellvorhaben… Die negativen Auswirkungen dieser mangelhaften Einbeziehung der eigentlichen Träger der Softwareentwicklung in den Vorphasen der Projekte waren immer wieder spürbar… Zwar kam es dann in der überwiegenden Mehrheit der Projekte zu Präzisierungen und Korrekturen des ursprünglichen Projektauftrags, diese Redefinition war aber häufig ein mühsamer Prozeß, den man unter Umständen durch eine frühere Einbeziehung der eigentlichen 'Experten' an der Basis hätte wesentlich effektiver gestalten können."* ([WelOrt92]).

Unter dem starken zeitlichen Druck gerieten Grundsätze der Qualitätssicherung immer mehr in den Hintergrund. In der Bewertung von Unhandlichkeiten und Fehlern des Programms ging es hauptsächlich darum, ob es bloße "Schönheitsfehler" waren, ob ein "Workaround" existiert oder ob es ein "echter" Fehler ist. Weltz et al. sagen dazu: "*Die Grundsätze der Qualitätssicherung wurden unter dem Druck der Verhältnisse meist sehr rasch recht abstrakt… Letztlich stand die Einhaltung des finanziellen und terminlichen Rahmens im Vordergrund."*

Im Projektverlauf nahmen der Projektleiter des Softwarehauses und in späteren Phasen auch der Schulungsleiter des Auftraggebers Schlüsselrollen ein. Ihre Aufgabe war in zunehmendem Maße, zwischen den Beteiligten zu vermitteln. Insbesondere war nach der Analysephase der Projektleiter der einzige Kontakt der Entwickler zum Auftraggeber und der einzige Vermittler neuer Anforderungen. Weltz et al. kommen zu ähnlichen Ergebnissen, wenn sie sagen: "*In den meisten Softwareprojekten, deren Ablauf wir untersuchten, waren Positionen zu erkennen, denen ganz offensichtlich in dem kooperativen Gefüge des Entwicklungsprozesses eine Schlüsselposition zukam: bei der Übermittlung von Informationen und der Koordination von Tätigkeiten, bei der Abstimmung der Belange verschiedener Bereiche oder der Vorbereitung von Entscheidungen. Diese Positionen waren fast durchweg überfrachtet."*

Auch die Probleme während der ersten Auslieferung des Programms scheinen mir typisch für Softwareprojekte zu sein. Unter Zeitdruck wird das Programm zu wenig getestet und ist somit instabil. Dazu kommt, daß ein Großteil an Unhandlichkeiten und fehlender Funktionalität erst auftaucht, wenn die eigentlichen Benutzer am echten Programm ihrer täglichen Arbeit nachgehen.

Im vorliegenden Fall verlief der erste Kontakt mit dem System für die Anwender ernüchternd und unter großen Schwierigkeiten. Weltz et al. führen dazu aus: "*Dieses Konfliktpotential kam häufig in den frühen Projektphasen nicht zum Tragen, weil die Konturen des neuen Produkts und seiner Auswirkungen noch blaß und undeutlich, mögliche Interessengegensätze damit verdeckt blieben… Unklarheiten, ungelöste Widersprüche wurden so unter Umständen über mehrere Entwicklungsschritte fortgeschleppt, bis sich irgendwann ihre Klärung aufdrängt, häufig erst bei der Konfrontation mit dem fertigen 'Produkt'*".

Es gab zwar in der Einführungsphase gelegentlich direkte Kontakte zwischen Anwendern und Entwicklern, jedoch wissen die Entwickler nur sehr wenig über die tägliche Praxis der Anwender und deren Probleme mit dem System. Hier hat das Fehlerformular einen direkten Austausch zwischen Entwicklern und Anwendern weitgehend ersetzt. Weltz et al. sagen dazu: "*In den meisten Projekten fehlte ein systematisches Feedback aus der Anwendungspraxis und bestand auch ein erstaunliches Desinteresse vieler Entwickler und vor allem Projektverantwortlicher gegenüber dem weiteren Schicksal der Produkte ihrer Arbeit… Überraschend für uns war das weitgehende Fehlen von systematischen Versuchen zu einer differenzierten Evaluierung des Erfolgs von Projekten - vor allem unter anwendungsbezogenen Aspekten - und einer gründlichen Auseinandersetzung mit dessen Ursachen… Ohne Zweifel besteht hier eines der schwerwiegendsten Defizite in der gegenwärtigen Praxis der Softwareentwicklung.*"

Zusammenfassend kann man feststellen, daß das vorliegende Projekt sowohl in seinen Randbedingungen und seinem Verlauf, als auch in seinen Schwierigkeiten durchaus vergleichbar zu anderen heute durchgeführten Softwareprojekten ist. Es liegt deshalb nahe, anzunehmen, daß es sich bei den Ursachen um typische, dem Entwicklungsprozeß innewohnende Problematiken handelt. In den folgenden Kapiteln werde ich versuchen, diese Ursachen näher zu beleuchten.

4. Einordnung des Projektes

Nachdem gezeigt wurde, daß das vorliegende Projekt einen für Softwareprojekte typischen Verlauf genommen hat, soll nun der Projektverlauf und die Vorgehensweise aus wissenschaftlicher Sicht eingeordnet werden. Dabei werden zunächst die traditionelle und die evolutionäre Sichtweise als relevante Modelle der Softwareentwicklung vorgestellt, um daran anschließend das vorliegende Projekt einordnen zu können.

4.1. Traditionelles Software Engineering

Das traditionelle *Software Engineering* entstand Ende der sechziger Jahre unter dem Eindruck zunehmender Unkontrollierbarkeit softwaretechnischer Entwicklungsprojekte. Merkmale dieser sog. "Softwarekrise" waren im wesentlichen die Überziehung von Kosten und Terminen, die Fehlerhaftigkeit und schlechte Wartbarkeit, sowie die mangelhafte Erfüllung von Anforderungen.

Das sich entwickelnde Software Engineering versuchte, diese Probleme durch verstärkte Strukturierung und Kontrolle des Entwicklungsprozesses zu bewältigen, wobei das Phasenmodell die entscheidende Rolle spielte. Beim Phasenmodell wird davon ausgegangen, daß sich ein Softwareentwicklungsprozeß nach einem Modell zeitlich aufeinander folgender Phasen durchführen läßt. Zunächst werden die Anforderungen an das System möglichst vollständig erhoben, um dann von den Entwicklern in mehreren Phasen in ein EDV-System umgesetzt zu werden. *"Die Realität zeigte jedoch, daß eine rein sequentielle Vorgangsweise in den seltensten Fällen durchführbar ist."* ([PomBla93]). Die Phasenmodelle waren deshalb in der Folge Gegenstand berechtigter Kritik und es gab eine Reihe von Anpassungen des Modells, wobei im wesentlichen die Rückkehr zu früheren Phasen erlaubt wurde. Als wichtigstes Kennzeichen der traditionellen Vorgehensweise bleibt jedoch ein Phasendenken erhalten.

Vordringliches Ziel bei der Entwicklung der traditionellen Vorgehensweise war es, den Softwareentwicklern Kontrolle über den scheinbar chaotischen Prozeß der Softwareentwicklung zu ermöglichen. Floyd sagt dazu: *"Maßgebliche Ziele sind dabei die Transparenz und Kontrollierbarkeit des Herstellungsprozesses, die Gewährleistung vorgegebener Anforderungen und Qualitätsmerkmale durch die Produkte und die Einhaltung eines fest kalkulierten finanziellen und terminlichen Rahmens."* ([Floyd93])

Die traditionelle Sichtweise ist somit eine primär *produktbezogene* Sichtweise, denn sie macht das Produkt der Entwicklung - das lauffähige Programm - zum Hauptgegenstand der Überlegungen. Traditionelle Softwareentwicklung kann als *"Produktion im Sinne einer Fertigung"* verstanden werden ([Floyd93]). Durch die Überbewertung des Produktes findet zu wenig explizite Unterstützung des Prozesses der Softwareentwicklung statt, da der Prozeß durch das Phasenmodell in seinem Verlauf hinreichend festgelegt erscheint. Das Vorhandensein von Phasen scheint den Entwicklungsfortgang speziell für das Management transparenter zu gestalten und man versucht mit solchen Modellen, dem

Idealbild vollständig kontrollierbarer Prozesse möglichst nahe zu kommen. Pomberger et al. sagen dazu: *"Das Modell beruht auf der (falschen) Annahme, daß der Entwicklungsprozeß in der Regel sequentiell ausgeführt werden kann und Iterationen zwischen den Phasen nur ausnahmsweise notwendig sind."* ([PomBla93]).

In der Praxis findet man heute nur selten ein Vorgehen nach einem reinen Phasenmodell, jedoch sehr häufig organisatorische Maßnahmen, die eine Annäherung an das "Idealbild" ermöglichen und damit letztlich ein Phasenkonzept aufrecht erhalten sollen. Einige dieser organisatorischen Maßnahmen, die auch im vorliegenden Projekt zu finden sind, werden von Floyd genannt: *"Dies führt zu Denk- und Arbeitsformen, mit Hilfe derer man das Idealbild zu simulieren versucht, etwa durch organisatorische Maßnahmen wie die Abschirmung des Entwicklungsteams von der Kommunikation mit dem Kunden, die Zulassung von Rückgriffen auf bereits abgeschlossene Phasen oder die Einführung von formalisierten Verfahren zur Berücksichtigung geänderter Anforderungen."* ([Floyd93]).

Das traditionelle Software Engineering konnte in Teilbereichen der Softwareentwicklung (Programmierung, Spezifikation) wirksame Lösungen anbieten, jedoch *"wurde die Softwarekrise im eigentlichen Sinne noch nicht bewältigt. Die ungelösten Probleme haben sich vielmehr vom technischen Kern der Softwareentwicklung in das Vorfeld der Anforderungsermittlung, in die Gestaltung der Mensch-Rechner-Interaktion und in die Einbettung der Software in menschliches Arbeitshandeln verschoben."* ([Floyd93])

Wesentliche Kritik an traditionellen Vorgehensweise kann in Bezug auf das Phasendenken formuliert werden, das mit seinem grundsätzlich statischen Charakter den dynamischen Erkenntnisprozessen, die in der Realität stattfinden, nicht gerecht wird. Die Bedeutung einer breiten und kontinuierlichen Kommunikation wird in traditionellen Softwareprojekten unterschätzt und Kommunikation auf bestimmte Phasen oder bestimmte Personen begrenzt. *"Kooperative Erkenntnisprozesse finden so nicht statt. Wo Phasenmodelle angewendet werden, sind Entwickler und Benutzer gezwungen, um das Modell herumzuarbeiten. Wird die Kommunikation zwischen Entwicklern und Benutzern, wie im Modell vorgesehen, auf die Anfangsphasen beschränkt, so ergeben sich häufig erhebliche sogenannte Akzeptanzprobleme bei der Auslieferung des Systems, weil es sich als nicht relevant erweist."* ([Floyd93]).

Da die Anwender erst zur Auslieferung Kontakt mit dem System bekommen, werden viele Fehler erst in der täglichen Arbeit entdeckt. Eine schwierige und konfliktreiche Einführungsphase kann deshalb als Folge einer traditionellen Vorgehensweise verstanden werden. Zusätzlich kann man erkennen, daß im traditionellen Software Engineering versucht wird, den evolutionären Charakter von Softwareentwicklung in die Zeit nach Systemauslieferung zu verlagern, wo es dann typischerweise zu einer Vielzahl neuer Versionen, aber auch aufwendiger Erweiterungen kommt. Dem liegt gerade bei Festpreisaufträgen der verständliche Wunsch des Softwareherstellers zugrunde, den Kunden nachträglich an den anfänglichen Kosten zu beteiligen, die in der Regel höher als erwartet ausgefallen sind.

4.2. **Softwareentwicklung als evolutionärer Prozeß**

Im Gegensatz zur traditionellen Sichtweise setzt sich seit einiger Zeit immer mehr eine evolutionäre Sicht der Softwareentwicklung durch (vgl. [Floyd93], [BuKaKuZü92], [KilGryZü93]). Softwareentwicklung wird hier als ein kontinuierlicher, wechselseitiger Lernprozeß aller Beteiligten über die gesamte Dauer des Projektes angesehen, woraus sich wichtige Rückschlüsse für den Entwicklungsprozeß ergeben:

- Der Lernprozeß findet bei allen Beteiligten statt: bei Anwendern, Abteilungsleitern, Geschäftsführung, Entwickler und Projektleitung.
- Es lassen sich nicht zu Beginn vollständige Anforderungen an das System bestimmen, da beide Seiten erst am Anfang des Lernprozesses stehen. Auch die Hoffnung, zumindest die wichtigsten Anforderungen am Anfang zu finden, ist unbegründet.
- Es ist natürlich und erwünscht, daß im Projektverlauf neue Anforderungen hinzukommen. Dies ist Beweis dafür, daß auf beiden Seiten ein kreativer Lernprozeß in Gang gekommen ist. Der Entwicklungsprozeß selber führt zu Modifikationen des untersuchten Anwendungsgebietes. Alle diese Prozesse müssen im Projektverlauf kontrolliert und von der Vorgehensweise explizit unterstützt werden, damit sie produktiv wirken können.

Die Anwender lernen während eines solchen Prozesses die Möglichkeiten der programmtechnischen Unterstützung kennen, und erst aus der wachsenden Kenntnis ergeben sich neue Ideen, was das System noch unterstützen könnte. Indem sich die Anwender Gedanken über eine programmtechnische Unterstützung ihrer Arbeit machen, werden die gezwungen, ihre stark verinnerlichten, unbewußten Routinearbeiten neu zu reflektieren. Dabei entstehen neue Anforderungen. Dies ist in der evolutionären Sichtweise ausdrücklich erwünscht und deutet nicht auf eine Gier der Anwender nach immer neuen “Features” hin oder auf das bei Entwicklern verbreitete Vorurteil, daß Anwender nicht wissen, was sie wollen.

Die Entwickler lernen das Arbeitsumfeld der Anwender erst im Entwicklungsprozeß kennen. Auch dieses Lernen läuft prozeßhaft ab. Es ist für den Entwickler unmöglich, das Arbeitsumfeld des Anwenders im vorweg komplett kennenzulernen oder es während der Implementierung aus einem Anforderungskatalog erschließen zu wollen.

Der wechselseitige Lernprozeß hält über das gesamte Projekt hinweg an und beschränkt sich nicht auf einzelne Phasen, wie z.B. einer anfänglichen Analysephase. Wesentliche Fragen und Unklarheiten treten oft erst während des Entwurfs und der Implementierung auf. Der Anforderungskatalog, der auf Basis früherer und damit u.U. veralteter Erfahrungen und Kenntnissen geschrieben wurde, kann oft genau diese Fragen nicht beantworten.

Der Lernprozeß spiegelt sich auch in der Vorgehensweise selber wieder, die sich oft erst im Projektverlauf ergibt oder modifiziert werden muß, weil jedes Projekt

eigene Gesetzmäßigkeiten hat. Darüber hinaus finden sowohl beim Auftraggeber als auch beim Softwarehaus projektübergreifende Lernprozesse statt. Es ist beispielsweise bekannt, daß Softwarehäuser besonders dann gute Software entwickeln, wenn sie für ein und dasselbe Anwendungsgebiet zum zweiten oder wiederholten Mal ein Softwaresystem neu entwickeln. Die Erfahrungen früherer Entwicklungen werden in neue Projekte übernommen. Auch hierbei entwickeln sich Vorgehensweise, Problembewußtsein und Lösungsfindung evolutionär.

Viele der genannten Punkte (z.B. wachsende oder ungenaue Anforderungen) werden aus der traditionellen Sichtweise heraus als Probleme und Schwierigkeiten angesehen, die es zu beseitigen gilt. Die evolutionäre Sichtweise nimmt hier einen anderen Standpunkt ein: Diese sogenannten *Schwierigkeiten* stellen kein Übel dar, sondern gehören zur grundlegenden Natur von Entwicklungsprojekten allgemein, wobei der Begriff *Entwicklung* diese Richtung bereits andeutet. Es kommt nun darauf an, diese Tatsachen anzuerkennen und sie innerhalb einer veränderten Vorgehensweise explizit und damit kontrollierbar zu gestalten.

Damit ist deutlich, daß der Begriff “evolutionäre Softwareentwicklung” nicht andeuten soll, daß es Softwareentwicklungen gibt, die nicht evolutionär verlaufen. Dem ist nicht so. Jede Software entwickelt sich evolutionär und unter Einbeziehung *aller* Beteiligten. Werden einzelne Gruppen von bestimmten Phasen des Projektes ausgeschlossen, so kommen deren Anforderungen an das System in den späten Phasen hinzu, speziell bei Systemeinführung, wenn nämlich das konkrete Programm zu sehen ist. Die Kosten für Änderungen fallen dann jedoch erheblich höher aus.

Es scheint von wesentlicher Bedeutung zu sein, besonders frühzeitig im Entwicklungsprozeß eine effektive Qualitätssicherung der Anforderungen an das entstehende System durchführen zu können. In einer evolutionären Vorgehensweise ist deshalb die Verständlichkeit der benutzten Dokumente von entscheidender Bedeutung. Die gängige Erfahrung in Softwareentwicklungen ist, daß oft erst im ablauffähigen System Fehler und Unzulänglichkeiten des Entwurfs und der Anforderungsermittlung erkannt werden können. Somit kommt es darauf an, solche Fehler bereits frühzeitig erkennbar zu machen. Der Gebrauch verständlicher Dokumente und der frühzeitige Einsatz von Prototypen, ist deshalb für eine evolutionäre und kundenorientierte Vorgehensweise unablässig.

4.3. **Einordnung des Projektes**

Im vorliegenden Projekt hat sich neben dem Produkt auch die Vorgehensweise selber im Verlaufe des Projektes weiterentwickelt. Bei den Beteiligten setzten durch die Beschäftigung mit dem Projekt und durch die Arbeit mit dem System Lernprozesse ein, die Rückwirkungen auf das Projekt selbst hatten. Insofern ist der evolutionäre Charakter deutlich erkennbar.

Es ist bemerkenswert, daß besonders in den Vorphasen und in der Anfangsphase des Projektes auch in der Vorgehensweise des Softwarehauses evolutionäre

Elemente enthalten waren. Beispielsweise waren für die Aquirierung prototypische und visionäre Darstellungsformen von entscheidender Bedeutung. Auch in der Anfangsphase des Projektes läßt sich das Ziel des Softwarehauses erkennen, die Vertreter des Auftraggebers gleichberechtigt am Entwicklungsprozeß zu beteiligen und durch den Einsatz von Systemvisionen frühzeitige Rückkopplungen zu ermöglichen. Außerdem wurden sowohl die Entwickler als auch die Anwender an dieser ersten Phase teilweise direkt beteiligt.

Im Projektverlauf fand dann aber, bedingt durch den engen Zeitrahmen, eine immer stärkere Verlagerung hin zu traditionelleren Vorgehensweisen statt, was aus folgenden Punkten sichtbar wird:

- Die Anwender und Entwickler wurden nicht mehr direkt an der Erhebung neuer Anforderungen beteiligt.
- Die Entwickler wurden bis zur Auslieferung und nach Produktionsbeginn durch den Projektleiter abgeschirmt.
- Die Kommunikation lief hauptsächlich über den Projektleiter, den Schulungsleiter und Abteilungsleiter beim Auftraggeber.
- Neue Anforderungen wurden nicht mehr in Form von Systemvisionen, sondern ausschließlich über den Leistungsumfang rückgekoppelt, wobei der Leistungsumfang nicht ausreichend gelesen oder verstanden wurde und somit im Prinzip keine effektive und rechtzeitige Qualitätssicherung stattfinden konnte.
- Bei Systemeinführung kam es zu massiven Problemen. Neben den Instabilitäten tauchten diverse Unhandlichkeiten und fehlende Funktionalität im Programm auf. Es entstanden in der Folgezeit erhebliche Kosten durch Verbesserungen und Folgeaufträge, die vom Umfang her in etwa die gleiche Größenordung hatten, wie das System selber.

Zusammenfassend kann man sagen, daß im Projekt anfangs evolutionäre Elemente enthalten waren, die Vorgehensweise an sich jedoch traditionellen Charakter hatte, bzw. durch den Termindruck in traditionelle Bahnen gezwungen wurde. Die Vorgehensweise orientierte sich am Phasenmodell und beteiligte Anwender und Anwendervertreter schwerpunktmäßig während der Anforderungsermittlung. Die Aufgabe der Entwickler war im wesentlichen die technische Realisierung und weniger das Erheben neuer Anforderungen.

Für die Entwicklung des integrierten Systems ist eine teilweise veränderte Vorgehensweise geplant, die u.a. Prototypen und schrittweise Auslieferung produktionsreifer Programmteile anstrebt. Es kann davon ausgegangen werden, daß dies im Projektverlauf positive Auswirkungen haben wird.

5. Bewertung des Projektes

Es wäre naheliegend, in einer Bewertung des Projekts in der Hauptsache auf den strategischen Charakter hinzuweisen und die daraus resultierenden zeitlichen und finanziellen Beschränkungen zu Alleinverantwortlichen der offensichtlichen Schwierigkeiten im Projekt zu erklären. Obwohl das Projekt sicherlich stark von seinen Randbedingungen geprägt war, muß man trotzdem konstatieren, daß nahezu alle industriellen Projekte, nicht nur Softwareprojekte, mit ähnlichen Bedingungen zu kämpfen haben.

Es gibt unabhängig von den beschriebenen Randbedingungen einige Themenkomplexe, die einer genaueren Untersuchung bedürfen. Ich will in dieser Arbeit zum einen die Vorgehensweise selber und zum anderen den Stellenwert der Kommunikation im Projekt näher untersuchen. Folgende Punkte halte ich hierbei für relevant:

- Das phasenorientierte Vorgehen wirkte sich im Projektverlauf negativ aus.
- Es wurde versucht, Kommunikationsbeziehungen zwischen den Beteiligten zu minimieren und zu formalisieren. Die Bedeutung direkter Kommunikation zwischen Anwendern und Entwicklern wurde unterschätzt. Als Folge konnte das Sichtenproblem nicht gelöst werden.
- Das Softwarehaus besaß über den gesamten Projektverlauf ein starkes *Modellmonopol*.
- Die eingesetzte hochinnovative Technologie führte zu einer hohen *technischen Stabilität*, die jedoch eine *fachliche Instabilität* nicht verhindern konnte.

5.1. Vorgehensweise

Bereits aus der Beschreibung des Projektverlaufes ist deutlich geworden, daß ein Vorgehen nach einem Phasenmodell für dieses Projekt zumindest angestrebt war. Die Gründe für ein solches Vorgehen, die im wesentlichen in der Kontrolle des Projektablaufes liegen, sind bereits erwähnt worden. Ebenso konnte gezeigt werden, daß sich eine solche phasenorientierte Vorgehensweise im vorliegenden Projekt praktisch nicht durchhalten ließ, weil die Phase der Anforderungsermittlung nicht beendet werden konnte.

Die Bereitschaft des Softwarehauses, überhaupt neue Anforderungen zuzulassen, wird vom Auftraggeber im nachhinein als sehr positiv gewertet. Warum muß nun trotzdem eine phasenorientierte Vorgehensweise kritisch beurteilt werden? Selbstverständlich lassen sich in Softwareprojekten Tätigkeiten wie Anforderungsermittlung, Entwurf, Implementierung und Testen erkennen. Gleichzeitig ist es verlockend, diese Tätigkeiten in eine zeitliche Reihenfolge zueinander zu setzen, wie es die traditionelle Vorgehensweise anstrebt. Das widerspricht jedoch dem fundamentalen Charakter von Softwareentwicklungsprojekten. Floyd sagt dazu: *"Softwareentwicklung wird als Prozeß*

wechselseitigen Lernens verstanden und die Einlösbarkeit einer zeitlichen Phasenaufteilung grundsätzlich angezweifelt." ([Floyd94]).

Wird an einem traditionellen Phasendenken in wie auch immer gearteter Weise festgehalten, so müssen sich daraus zwangsläufig Probleme ergeben. Im vorliegenden Projekt können diese Probleme anhand der Anforderungsermittlung sehr deutlich gezeigt werden. Zu Beginn des Projektes bestand eine große Offenheit gegenüber den Wünschen und Anforderungen des Auftraggebers. Man befand sich in der Phase der Anforderungsermittlung und das Ziel dieser Phase war gerade die Extrahierung der benötigten Programmfunktionalität. Insofern waren alle Anregungen willkommen. Als dann jedoch das Abschließen dieser ersten Phase nicht möglich war, weil immer neue Anforderungen von Seiten der Anwender geäußert wurden, entstanden für das Softwarehaus Probleme in der Entwurfs- und Implementierungsphase, die aus zeitlichen Gründen begonnen werden mußten. Deshalb wurden neue Anforderungen mit zunehmender Zeit immer lästiger. Für die Anwender und deren Vertreter wirkte es zusätzlich demotivierend, daß neue Anforderungen stets vor dem Hintergrund der zeitlichen und finanziellen Beschränkungen abgewägt wurden.

Meiner Meinung nach lassen sich diese Probleme unmittelbar auf eine am Phasenmodell orientierte Vorgehensweise zurückführen. Mambrey et al. bringen es auf den Punkt: *"Durch diese Vorgehensweise wird ein Lernprozeß der Beteiligten unterbunden. Vor allem die Betroffenen erhalten keine Möglichkeit, aus eigener Anschauung die Konzeptbildung zu beeinflussen."* ([MamOppTep86]).

Im vorliegenden Projekt sind die Symptome (die wachsenden Anforderungen) natürlich von allen Beteiligten gesehen worden, jedoch sind sie unterschiedlich interpretiert worden. Für das Softwarehaus bedeuteten die neuen Anforderungen erhöhten Aufwand, der nicht bezahlt wurde und für den Auftraggeber Verzögerungen und Schwierigkeiten in der Einführungsphase. Bei Auftraggeber und Softwarehaus besteht die Hoffnung, daß eine längere Analysephase schon im Vorwege viele der späteren Anforderungen entdeckt hätte. Von vielen Autoren wird dies bezweifelt; Weltz und Ortmann sprechen in diesem Zusammenhang von der *"Fiktion eines Vorlaufes von Wissen, Konsens und Anforderungsdefinition"* ([WelOrt92]).

Anwenderbeteiligung spielt in traditionellen Vorgehensweisen hauptsächlich in den frühen Phasen eine Rolle. Von den Realisierungsphasen sind sie normalerweise ausgeschlossen, weil es für die weniger technisch Ausgebildeten dann anscheinend "nichts mehr zu tun" gibt. Es erscheint mir charakteristisch, daß die Anwender als eigentliche "Experten der Anwendungswelt" lediglich als Informationslieferanten in frühen Phasen gesehen werden und somit prinzipiell, wie Floyd es ausdrückt, *"instrumentalisiert"* werden. Daß man sie danach nicht mehr am Projektfortgang beteiligt, muß deshalb zu Problemen führen, weil die Anforderungsermittlung lediglich auf dem Papier abgeschlossen ist, faktisch jedoch weitergeht. Erst bei Systemeinführung bekommen die Anwender wieder direkten Kontakt zum Projekt und so verläuft dieser *"moment of truth"* ([BuKaKuZü92]) zumeist unter großen Schwierigkeiten.

Es wurde bereits erwähnt, daß Softwareprojekte grundsätzlich evolutionär verlaufen und daß das traditionelle, phasenorientierte Vorgehen die evolutionären Elemente nicht genügend ins Zentrum der Überlegungen stellt, sondern versucht, sie in die Zeit nach Systemeinführung hineinzuverlagern. Auch im vorliegenden Projekt traten nach Systemeinführung und während des Systembetriebes ein Fülle von Veränderungswünschen auf, die zu einer Reihe von Nachbesserungen, neuen Versionen und Folgeaufträgen führten. Betrachtet man dies auf einer finanziellen Ebene, so kann man speziell eine Verlagerung der eigentlichen Entwickungskosten zu Lasten sogenannter Wartungs- und Änderungskosten erkennen.

In der Zeit nach Systemeinführung nimmt die Vorgehensweise deutlich evolutionäre Züge an, wenn beispielsweise Anwender wieder direkt einbezogen werden, Verbesserungen anhand des konkreten Systems diskutiert werden und fehlerhafte Programmteile relativ zügig und unbürokratisch nachgeliefert werden. Obwohl eine solche Vorgehensweise einen deutlich experimentellen Charakter hat, wird dies nicht offiziell zugegeben, weil u.U. daraus ein unprofessioneller Eindruck entstehen würde. Dabei wäre es meiner Ansicht nach durchaus wirkungsvoll, eine ausgelieferte Version zunächst auch offiziell als Testversion (oder gar Prototyp) zu bezeichnen und diese ausgewählten späteren Anwendern zur Verfügung zu stellen, um dadurch eine viel positivere Einstellung gegenüber Fehlern und Instabilitäten und eine verstärkte Identifikation mit dem System herbeizuführen.

5.2. **Kommunikation**

Zunächst muß man sich fragen, worin die Bedeutung von Kommunikation in einem Softwareprojektes überhaupt besteht. Zum einen besteht Notwendigkeit für eine Übermittlung relevanter Informationen zwischen den beteiligten Gruppen und Personen. Ob eine Software für einen Bereich entwickelt wird, in dem der Kunde bereits über umfassende Erfahrung verfügt oder ob mit der Software teilweise Neuland betreten wird, in jedem Fall muß Anwendungswissen zu den Entwicklern gelangen und umgekehrt EDV-Wissen zu den Vertretern des Kunden. Hier sind besonders solche Dokumente wichtig, die für die Beteiligten gleichermaßen verständlich sind, somit eine wirkungsvolle Kommunikationsbasis schaffen. Dokumente, die für bestimmte Gruppen nur schwer verständlich sind, schließen diese vom Qualitätssicherungsprozeß und dem Projektfortgang aus.

Neben der Übermittlung wichtiger Sachinformationen, die man in Anlehnung an Schulz von Thun als *Sachebene* bezeichnen könnte, dient Kommunikation aber auch der Unterstützung der *Beziehungsebene.* Eine stabile Beziehungsebene ist Grundlage für die Etablierung und Absicherung effektiver Zusammenarbeit und wichtiger Bestandteil für einen gut funktionierenden Austausch auf der Sachebene. Für die Festigung der Beziehungsebene ist vorwiegend ein Verständnis der jeweils anderen Seite und somit direkte und kontinuierliche Kommunikation notwendig. Die daraus entstehende Vertrauensbasis ist gerade

in Konfliktzeiten unentbehrlich (vgl. [SchulzvonThun81], [SchulzvonThun91], [WatzBeaJa67]).

Die zu Beginn dieser Arbeit geschilderten Perspektiven der am Projekt Beteiligten machen deutlich, daß die zugrundeliegenden Erwartungen und Interessen nur selten in Harmonie zueinander gestanden haben. Das daraus resultierende Konfliktpotential kann meiner Meinung nach nur durch mehr an Stelle von weniger Kommunikation entschärft werden. Im Gegensatz dazu fand im vorliegenden Projekt mit zunehmendem Zeitdruck eine erhebliche Beschränkung von Kommunikation statt, die meiner Meinung nach Ursache für viele Schwierigkeiten in der Folgezeit war. Wie in Softwareprojekten üblich, war auch im vorliegenden Projekt die Zeit der Systemeinführung eine der konfliktreichsten Phasen, weil gerade dort die verschiedenen Interessen am deutlichsten hervortraten und sich oftmals konträr gegenüberstanden. In einer solchen Zeit ist eine gesicherte Basis, d.h. eine stabile Beziehungsebene für gemeinsame Zusammenarbeit nötig, um Konflikte sachlich wirkungsvoll bewältigen zu können. Diese Basis war zur entscheidenen Zeit im Projekt nicht in ausreichendem Maße vorhanden, da über weite Strecken im Projektverlauf zu wenig direkter Kontakt zwischen Anwendern und Softwarehaus ermöglicht wurde. Daraus resultierte meiner Meinung nach ein bis heute wirksames Mißtrauen vieler Anwender gegenüber weiteren Softwareentwicklungen.

Im folgenden will ich die Kommunikationsproblematik anhand folgender Punkte thematisieren:

- Einsatz von schwer verständlichen Dokumenten
- Minimierung von Kommunikationsbeziehungen
- Formalisierung von Kommunikation
- Modellmonopol auf Seiten des Softwarehauses

5.2.1. **Verständlichkeit von Dokumenten**

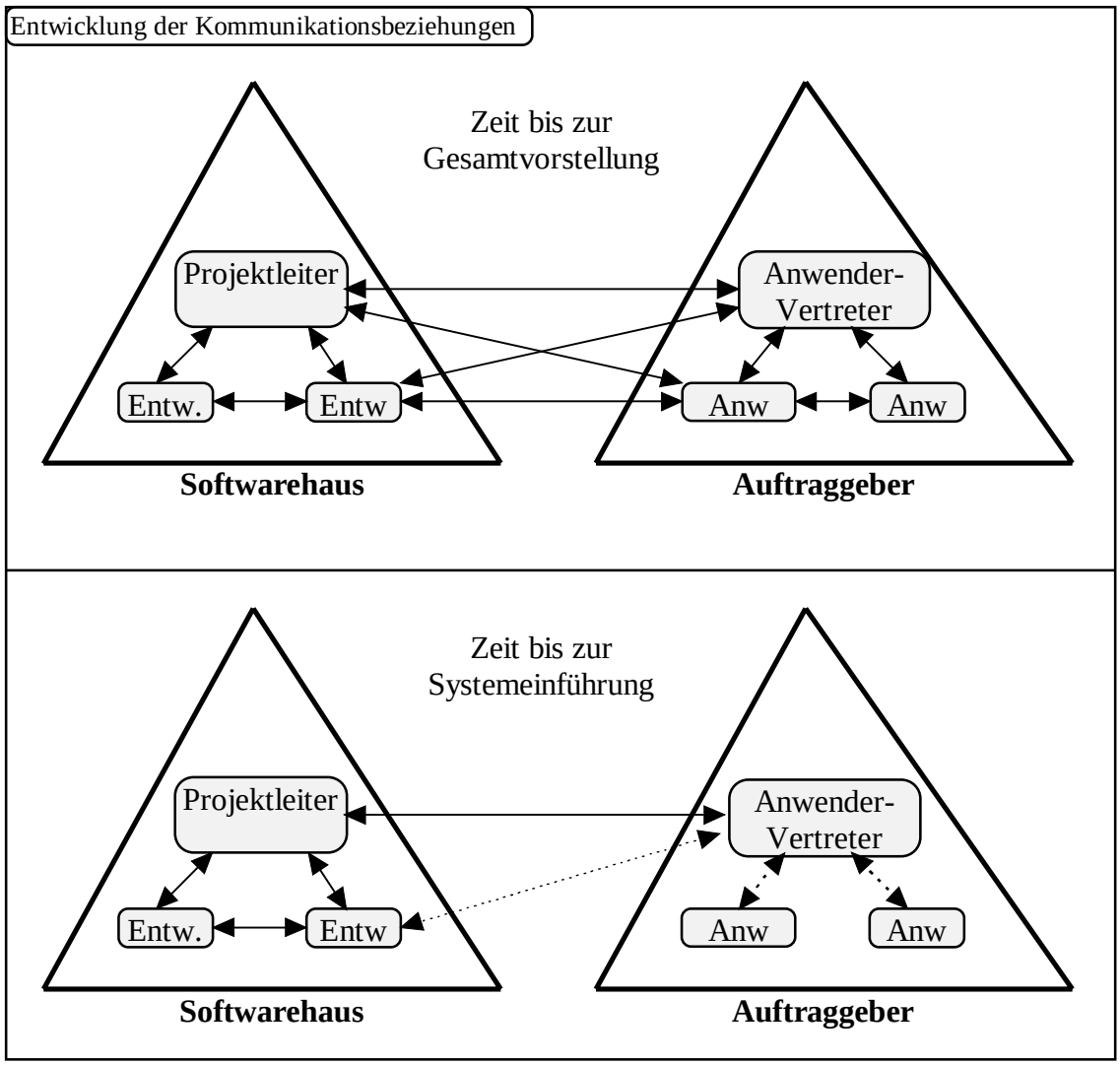
Im vorliegenden Projekt kam dem Leistungsumfang entscheidende Bedeutung zu, da dieses Dokument im Prinzip über weite Strecken einziges Kommunikationsmittel zwischen Anwenderseite und Softwarehaus war. Auch in der Analysephase diente er als Informationsübermittlung, da spätere Anwender zunächst indirekt über den Leitungsumfang vom Projektfortgang informiert wurden und erst später über Systemvisionen direkt beteiligt waren. Der Leistungsumfang war für die überwiegende Anzahl der Anwender und Anwendervertreter nur schwer verständlich und wurde als Konsequenz nur von wenigen überhaupt gelesen. Dies ist ein erhebliches Manko, da eine frühzeitige Qualitätssicherung nur stattfinden kann, wenn diejenigen, die Dokumente bewerten sollen, diese auch verstehen.

In Bezug auf den Einsatz von Systemvisionen habe ich beobachtet, daß gerade durch deren Verständlichkeit ein viel größerer Kreis von Personen erreicht werden konnte, der sich in der Lage fühlte, Anmerkungen dazu zu machen. Im vorliegenden Projekt hätten Systemvisionen durchaus Teil des Leistungsumfangs werden können, jedoch wurden sie von den Beteiligten meiner Meinung nach gar nicht als Dokumente im eigentlichen Sinne verstanden. Es ist deutlich geworden, daß Systemvisionen in die richtige Richtung weisen, jedoch ebenfalls inhärente Probleme mit sich bringen und nicht die alleinige Lösung sein können. In einem späteren Abschnitt wird genauer auf die Grenzen solcher Oberflächendarstellungen eingegangen.

Grundsätzlich stellt sich die Frage, in welcher Form die Ergebnisse der Anforderungsermittlung festzuhalten sind. Es erscheint naheliegend, hierfür genau ein Dokument zu benutzen und dieses zur Grundlage eines gegenseitigen Vertrages zu machen. Problematisch hierbei ist der evolutionäre Charakter von Softwareentwicklungen, der es unmöglich macht, die Anforderungsermittlung frühzeitig abzuschließen und somit eine Vertragsgrundlage zu haben. Zusätzlich muß angemerkt sein, daß die Erstellung nur eines Dokumentes meistens auch dazu führt, daß nur genau eine Person die Fertigstellung übernimmt. Der Lernprozeß, der gerade beim Schreiben eines solchen Dokumentes einsetzt, bleibt so auf eine Person beschränkt, was mir als nicht ausreichend erscheint. Beispielsweise ist mir bezüglich der Erstellung von Systemvisionen klar geworden, wie stark alleine die Gestaltung von solchen Oberflächenentwürfen für den Entwickler ein Lernprozeß in Bezug auf das entstehende System ist.

5.2.2. **Minimierung von Kommunikationsbeziehungen**

Die folgende Abbildung zeigt in vereinfachter Form, wie sich die Kommunikationsbeziehungen im Projektverlauf entwickelten:



In der Darstellung fällt besonders auf, daß sich mit fortschreitender Projektdauer und damit zunehmenden Termindruck eine Verengung der Kommunikationswege stattfand. Wo zu Anfang die Entwickler aber auch die Anwender teilweise direkt beteiligt waren, bildeten sich in späterer Zeit bestimmte Schlüsselpositionen aus, die die Kommunikation übernahmen und den übrigen Beteiligten als Vermittler dienten. Kilberth et al. bemerken dazu: "*Das traditionelle Software Engineering hat für die Softwareentwicklung eine Minimierung der Kommunikationsbeziehungen zwischen den beteiligten Gruppen angestrebt.*" ([KilGryZü93]).

Als Grundlage hierfür kann wieder das traditionelle Phasenmodell genannt werden, das breite Kommunikation und Anwenderbeteiligung hauptsächlich in der Phase der Anforderungsermittlung vorsieht. In späteren Phasen wird weniger, bzw. minimale Kommunikation angestrebt, weil die Anforderungsermittlung abgeschlossen erscheint und "nur" noch Koordinationsfragen zu lösen sind. Anzeichen dafür, daß diese Vorstellung nicht mit der Realität übereinstimmt, sind beispielsweise zunächst "kleine Runden", die im Laufe der Zeit immer mehr erweitert werden müssen. Auch die Schaffung neuer Arbeitshierarchien, die den Informationsfluß strukturieren sollen, weist auf tieferliegende Problematiken hin.

Eine Minimierung von Kommunikationsbeziehungen führt dazu, daß sowohl Anwender als auch Entwickler immer nur ein vermitteltes Bild dessen bekommen, was die jeweils andere Seite für Vorstellungen und Wünsche hat. So findet besonders bei den Entwicklern ein wirklichen Kennenlernen der Anwendungssituation nicht statt. Ich kann aus eigener Erfahrung sagen, daß für mich während der Programmrealisierung besonders die Zeit wichtig war, als ich direkt an der Anforderungsermittlung beteiligt war und beim Präsentieren der Systemvisionen unmittelbaren Kontakt zu den späteren Anwendern hatte. Für die Programmierung war dies entscheidend, da ich so besser in der Lage war, spätere Anwendungssituationen zu antizipieren. Wesentliche Anforderungen wurden den Entwickler im Projektverlauf über den Projektleiter vermittelt. Es war auf Grund der "Sachzwänge" sehr verständlich, daß die Entwickler abgeschirmt wurden und ich kann aus eigener Erfahrung sagen, daß ich so sehr mit implementieren beschäftigt war, daß ich im Prinzip auf neue Anforderungen nur schlecht ansprechbar war. Letztlich war ich froh, daß der Projektleiter die Gespräche mit dem Auftraggeber führte. Trotzdem sehe ich es aus heutiger Sicht als Manko an, in dieser Zeit keinen direkten Kontakt zum Auftraggeber gehabt zu haben, weil viele wichtige Anforderungen eben erst in dieser Zeit entstanden sind.

Die Bedeutung, die Anwendungswelt direkt kennenzulernen, bringen Kilberth et al. auf den Punkt: *"Die Voraussetzung für die Entwicklung eines solchen Systems ist, daß alle Beteiligten, insbesondere die Entwickler, verstehen, was die Anforderungen im Anwendungsbereich sind und wie die dort anstehenden Aufgaben erledigt werden. Diese Notwendigkeit, eine Anwendungssituation zu verstehen,… steht im Widerspruch zu konventionellen Sichtweisen, bei denen es Aufgabe der Entwickler ist, aus einem vorgegebenen Pflichtenheft oder einer anderen vorgegebenen Beschreibung in schematischen, möglichst formalen Schritten ein Softwaresystem abzuleiten."* ([KilGryZü93])

Der direkten Entwickler-Anwender-Kommunikation wird heute von vielen Autoren die entscheidende Rolle in der Entstehung qualitativ hochwertiger Software eingeräumt. In den von mir geführten Gesprächen trat immer auch die Bedeutung direkter Kommunikation deutlich zutage und die Probleme, die eine Fehlen mit sich bringt:

- Entwickler können Systemanforderungen und -wünsche des Anwenders nur verstehen und würdigen, wenn sie die "echten" Arbeitszusammenhänge kennen und verstehen.
- Wechselseitige Vorurteile können nur im direkten Kontakt abgebaut werden.

- Entwickler tendieren dazu, sich auf technische Probleme zu konzentrieren und sich u.U. von realen Problemen zu entfernen. Beispielsweise kommt dies bei der Bewertung sogenannter “Schönheitsfehler” zum Tragen. Für die Anwender können diese Schönheitsfehler eine beträchtliche Verschlechterung der Arbeitssituation bedeuten, wobei es für den Entwickler eher um die Frage geht, ob nicht vielleicht doch ein “Workaround” vorhanden ist, dessen Existenz die Dringlichkeit der Fehlerverbesserung sofort herabsetzt. Das zugrundeliegende Sichtenproblem ist bereits angesprochen worden.

Die Bedeutung direkter Gespräche zwischen Entwicklern und Anwendern kann auch daran erkannt werden, daß solche Kontakte auch im vorliegenden Projekt nach Systemeinführung stattfanden und durchweg positive Resultate für beide Seiten brachten. Letztlich können meiner Ansicht nach nur direkte Gespräche eine dauerhafte und stabile Vertrauensbasis etablieren, die für die Bewältigung von Krisenzeiten nötig ist. Deshalb halte ich es für notwendig, daß solche Kontakte Teil der offiziellen Vorgehensweise werden und somit explizit gewürdigt und gefördert werden.

5.2.3. **Formalisierung der Kommunikation**

Wie sich Formalisierung im vorliegenden Projekt einschränkend auf wechselseitige Kommunikation ausgewirkt hat, soll exemplarisch am Beispiel des *Fehlermeldungsformulars* erläutert werden. Während des Systembetriebes wurden von den Anwendern notwendigerweise Fehler entdeckt und diese mußten gemeldet und behoben werden. Dies stellte sich zunächst als schwieriger Prozeß heraus, weil es keine kompetenten Ansprechpartner im Hause des Auftraggebers gab. Nachdem der offizielle Ansprechpartner schließlich benannt war, führte das Softwarehaus ein sog. Fehlermeldungsformular ein, daß die Anwender ausfüllen mußten, wenn sie Fehler und Änderungswünsche entdeckten. Diese Fehlermeldungen wurde zuerst im Hause des Auftraggebers gesammelt und überprüft. Viele Fehlermeldungen wurden gar nicht erst weitergeleitet, weil es sich dabei um Dinge handelte, die der Benutzer nicht verstanden hatte, die schon gemeldet waren oder die nicht verändert werden sollten, weil es zuviel kosten würde. Die Benutzer füllten zu Anfang eine Vielzahl solcher Formulare aus, jedoch mit der Zeit immer weniger. Das Formular selber war schwer verständlich, es fehlte die Rückmeldung zu den einzelnen Fehlern und es war unklar, ob sie überhaupt behoben werden sollten. Auch wurden viele Unhandlichkeiten des Programms gar nicht als Fehler gemeldet, weil das Ausfüllen zu aufwendig erschien oder inzwischen “Workarounds” gefunden waren und die Unhandlichkeiten nicht mehr als solche empfunden wurden.

Kritisch an der Maßnahme, ein Fehlerformular einzuführen, ist nicht das Formular als solches, obwohl dieses für die Anwender nur schwer verständlich erschien und von ihnen eher als Programmiererformular gesehen wurde. Kritisch muß hier in erster Linie die Abschirmung der Entwickler von den Resultaten ihrer Arbeit gesehen werden und die Behinderung der Anwender, ihre Wünsche in angemessener Form zu äußern. Insbesondere erreichen die Entwickler nur noch “wirkliche” Fehler, also Dinge, die unbedingt behoben werden müssen. Die vielen Kleinigkeiten und Unhandlichkeiten, die den Benutzer ihre

tägliche Arbeit erschweren, erreichen das Softwarehaus oft gar nicht mehr oder werden herausgefiltert. Dies kann bei den Entwickler u.U. zu einer falschen Einschätzung des wirklichen Arbeitsumgangs und der Zufriedenheit der Benutzer mit dem Programm führen. Ich kann dies aus eigener Erfahrung bestätigen, als ich nach Projektende das erste Mal im Hause des Auftraggeber war, um mit den Vorbereitungen dieser Arbeit zu beginnen. Die Realität, die ich dort antraf, war komplett anders, als ich es mir vorgestellt hatte. Die Stimmung dort und die Einstellung zum Programm war wesentlich negativer, als ich aus den Informationen, die mich im Entwicklerteam erreichten, geschlossen hätte. In der Tat hatte ich eher erwartet, die Anwender total glücklich über das System vorzufinden.

Wie bereits erwähnt, hat die Abschirmung den wichtigen Grund, die Entwickler ungestört arbeiten zu lassen. Um den Termin im vorliegenden Projekt zu halten, waren "überflüssige" Gespräche im Prinzip ab einem bestimmten Punkt nicht mehr möglich. Die Abschirmung der Entwickler wurde allerdings auch aufrechterhalten, als die wirklich "heißen" Phasen vorbei waren. Die Entwickler bleiben so in dem Glauben, daß es "das Richtige" war, was sie entwickelt haben, obwohl auch im vorliegenden System eine Vielzahl von Umständlichkeiten und Uneinheitlichkeiten gerade in der Oberfläche zu bemerken sind. Durch die fehlende Rückmeldung entgeht den Entwicklern häufig die Chance, aus den gemachten Fehlern zu lernen.

Organisatorische Maßnahmen, wie die Einführung eines Fehlerformulars, sind notwendig und sicherlich ein wichtiger Teil der Projektkontrolle und -abwicklung. Jedoch müssen die Auswirkungen solcher Maßnahmen offen diskutiert und kritisch reflektiert werden. Eine *"Überbewertung des Formalisierbaren"* ([Wendt92]) kann dazu führen, daß die inzwischen allgemein als wesentlich erkannten informellen Kommunikationsbeziehungen zwischen den Projektbeteiligten in erheblicher Weise behindert werden. Formalisierte Abläufe sollen nicht abgeschafft werden, jedoch sollte ihr bürokratischer Charakter gesehen und ihre Bedeutung relativiert werden. Es muß klar sein, daß eine gut funktionierende Prozedur von Fehlermeldung und Fehlerbehebung lediglich an den Symptomen (den Fehlern) kuriert und nicht an den Ursachen, die zu solchen Fehlern geführt haben. Oftmals wird in die Perfektionierung solcher Symtombehandlung viel mehr Zeit investiert, als in tiefergehende Ursachenforschung.

5.2.4. **Modellmonopol**

Um das bisher gesagte zu vertiefen und besser einordnen zu können sei hier der Begriff des "*Modellmonopols*" eingeführt. Dieser wurde zunächst von Bråten in die Diskussion eingebracht und von Pasch speziell für die Softwareentwickung ausgearbeitet (vgl. [Bråten73], [Pasch92]). Zunächst will ich die Bedeutung des Begriffs "Modellmonopol" näher erläutern:

Am Anfang des Projektes steht die Aufgabe (das scheinbare Ziel), ein Hotelreservierungssystem zu erstellen. Es gibt dazu auf keiner der beteiligten Seiten eine genaue Vorstellung von einem solchen System. Entwickler und Anwender

haben nur unscharfe Bilder einer möglichen Realisierung. Keine Gruppe hat von der Vorgehensweise oder vom Aussehen des späteren Systems ein *Modell*.

Das Fehlen eines Modells ist nun gerade deshalb lästig, weil es ja um die "Lösung eines Problems" geht und um eine möglichst baldige. Dieses Modellvakuum ist nur schwer erträglich und es findet sich bald eine Gruppe, die ein Modell für die Vorgehensweise und das Aussehen des zukünftigen Systems entwickelt. Diese Gruppe ist in diesem Projekt (wie in fast allen Softwareprojekten) die Entwicklergruppe. Es ist scheinbar auch ihre Aufgabe, denn sie muß das "Problem lösen". Die Anwender nennen ihre Bedürfnisse und die Entwickler "bauen" das passende System. Es entsteht das, was Bråten einen *"asymetrischen Dialog"* oder sogar *"Monolog"* nennt. Die Entwickler bilden die *modellstarke* Gruppe, insofern sie (scheinbar) über ein Modell zur Lösung des Problems verfügen. Dabei gerät der Auftraggeber und besonders die weniger technisch orientierten Anwender als *modellschwache* Gruppe in eine relativ passive und reaktive Rolle

Nach außen scheinen beide Gruppen miteinander zu kommunizieren, jedoch findet in Wirklichkeit eine immer stärkere Anpassung der schwachen Gruppe an die starke statt. Pasch führt dazu aus: *"Will 'B' den Dialog aufrechterhalten, ist er gezwungen, sich im Modell von 'A' auszudrücken, was er aber vorher verstehen muß; dabei ist er auf 'A' angewiesen. Ironischerweise gerät nun 'B' bei seinen Versuchen, seine schwache Position durch Verständnis des Modells von 'A' zu stärken, immer mehr unter die Kontrolle von 'A', je erfolgreicher 'B' bei seiner Anpassung an das Modell von 'A' ist."* ([Pasch92]).

Die Folge ist, daß es nicht zu einer gemeinsamen, kooperativen Lösungsfindung kommt. Die Anwender tragen nicht mehr selbst zur Modellbildung bei, sondern bewegen sich in der Modellwelt der Entwickler. Das, was Naur *"Theoriebildung"* ([Naur85]) nennt, findet nur einseitig statt. Wenn Entwickler ihr Modell, z.B. in Form eines schriftlichen Leistungsumfangs oder durch Systemvisionen präsentieren, dann kommt es von Anwenderseite im besten Fall zu kritischen Anmerkungen und Verbesserungsvorschlägen. Dadurch jedoch findet lediglich eine Verhärtung des vorhandenen Modells statt. Das Entwicklerteam besitzt weiterhin das *"Modellmonopol"*. Es bleibt unklar, ob das bei den Entwicklern vorhandene Modell überhaupt eine passende Lösung für die Probleme der Anwender darstellt. Dies ist nicht mehr von Interesse, wenn sich das Modellmonopol erst etabliert hat. Der Zeitdruck treibt die Beteiligten in der Folge dazu, auf keinen Fall von dem einmal gewählten Modell loszulassen.

Der Begriff *Modellmonopol* wird im folgenden anhand der Funktion des Leitungsumfangs und der Systemvisionen näher erläutert.

Leistungsumfang

Der Leistungsumfang ist zunächst einziges greifbares Resultat der ersten Anforderungsermittlungen (somit einzig vorhandenes *Modell*). Den Anwender wird der Stand der Anforderungsermittlung über den Leistungsumfang mitgeteilt. Wie sich gezeigt hat, haben sich die Anwender bemüht, den Inhalt des Leistungsumfanges zu verstehen, hatten jedoch einen schweren Stand. Um den Inhalt zu verstehen, mußten sich die Anwender in der Welt des Modells

bewegen, das im Entwicklerteam bereits am Entstehen war und Ausdruck im Leistungsumfang fand. Beispielsweise fand während der Gesamtvorstellung am 25./26.Mai auch eine "abschließende" Besprechung des Leistungsumfangs statt. Von den anwesenden Anwendern kamen so gut wie keine Anmerkungen zum Inhalt des Leistungsumfangs. Nur der anwesende Leiter der EDV-Abteilung des Auftraggebers konnte Hinweise und Fragen beisteuern. Er war der einzige, der sich im *akzeptierten Modell* überhaupt begrifflich bewegen konnte. Die Anwender befanden sich in einer eher schwachen Position und somit konnte nicht erwartet werden, daß die im Leistungsumfang vorgestellten Anforderungen auf ihre Richtigkeit und Plausibilität überprüfen werden konnten, geschweige denn, daß fehlende Anforderungen entdeckt würden.

Systemvisionen

Auch die Systemvisionen müssen unter dem Gesichtspunkt des Modellmonopols noch einmal kritisch hinterfragt werden. Die ersten Vorführungen der Dia-Shows verliefen meiner Meinung nach positiv, denn jetzt beteiligten sich auch die nicht EDV-erfahrenen Personen stärker am Gespräch. Aber auch hier ordneten sich die Anwender und Anwendervertreter unter das Modell der Entwickler. Die Entwickler stellten z.B. bei der Demonstration der Dia-Shows fest, daß nur selten eine Rückmeldung von den Anwendern kam. Es wurde viel über das Layout geredet, über die Bedeutung einzelner Felder. Man *"hielt sich an dem fest, was da war"*, wie es ein Beteiligter im nachhinein treffend formulierte. Das jedoch liegt gerade im Kern der Theorie über Modellmonopole. Die Systemvisionen sahen optisch sehr ansprechend aus. Sie erschienen die Lösung für die Probleme zu repräsentieren. Aber es war eben "nur" eine mögliche Lösung (ein Modell) der Entwickler, entstanden aus einem (natürlich zu dem Zeitpunkt) begrenzten Verständnis der Anwendungswelt. Viele der Visionen enthielten bei ihrer Präsentation offensichtliche Fehler, die erst nachträglich von den Entwicklern entdeckt und korrigiert wurden. Einige Anwendern äußerten sich später vom Aussehen des Programms überrascht, obwohl die vorher die System-visionen gesehen hatten.

Mir ist deutlich geworden, daß Systemvisionen einen Schritt in die richtige Richtung bedeuten, weil sie für alle Beteiligten gleichermaßen verständlich waren. Jedoch konnten sie keinen Eindruck vom Gesamtablauf des Systems geben. Auch die Autoren, die den Begriff "Systemvisionen" eingeführt haben, sehen das ähnlich: *"Die Bedeutung von Systemvisionen ergibt sich aus ihrer Stellung zwischen Analyse und Konstruktion. Primär sollen sie unter den Entwicklern ein Einverständnis über das zu entwickelnde System - eine Vision - herstellen."* (KilGryZü93]. Die Systemvisionen waren tatsächlich hauptsächlich für die Entwickler wichtig und stellten eine z.T. hervorragende Unterstützung beim Entwerfen des Systems dar. Ihr Wert für die Kommunikation mit den Anwendern muß jedoch unter den erwähnten Einschränkungen beurteilt werden.

5.3. **Stellenwert der Objektorientiertheit**

Die bisherige Analyse zielte deshalb stark auf kommunikative Aspekte hin, weil ich dort die wesentlichen Schwierigkeiten im Projekt vermute. Es erscheint merkwürdig, daß im Titel dieser Arbeit der Begriff *Objektorientierung* benutzt wird, der jedoch in der Arbeit selber bisher praktisch nicht vorkam. Dies weist auf einen scheinbaren Widerspruch hin: Neben einer eher traditionellen Vorgehensweise im Projektablauf, wurden für die Entwicklung des Hotelsystems vom Softwarehaus zweifelsohne hochinnovative Technologien eingesetzt. Diese sollen hier nochmals aufgelistet werden:

- Objektorientierte Analyse, Entwurf und Programmierung
- Client-Server-Architektur
- UNIX-Rechner als Server mit schneller relationaler Datenbank (Sybase)
- C++ als Programmiersprache
- MS-Visual-C++ als Teil der Entwicklungsumgebung
- Moderne interaktive Fensteroberfläche (MS-Windows)

Wenn man sich fragt, welchen Stellenwert der Einsatz der gewählten Technik, besonders der Objektorientierung, auf den Projektverlauf hatte, so stellt man zunächst fest, daß hier hauptsächlich die Entwickler profitierten. Der Einsatz objektorientierter Vorgehensweisen beschränkte sich auf die Seite der technischen Realisierung und hatte dort allerdings höchst positive Wirkungen hinsichtlich Entwicklungsgeschwindigkeit und Programmstabilität. Man könnte sagen, daß die Wahl der technischen Mittel eine hohe *technische Stabilität* des Produktes bewirkte, gleichzeitig aber eine *fachliche Instabilität* des Systems nicht verhindern konnte, die sich in einer Fülle zusätzlicher Anforderungen im Projektverlauf und kostspieliger Erweiterungen nach Aufnahme des Produktionsbetriebes zeigte.

Die Tatsache, daß Objektorientierung noch immer “Sache der Entwickler” ist, kann historisch sehr leicht erklärt werden, weil die objektorientierte Sichtweise aus der Programmierung heraus entstanden ist und von daher zunächst technisch orientiert gewesen ist ([Henders92]). Der Einzug objektorientierter Verfahren in Entwurf und Analyse ist erst vor relativ kurzer Zeit erfolgt und bleibt auch dort in jedem Fall auf der Seite des Softwareherstellers. Allerdings scheint die Objektorientierung auch Möglichkeiten hinsichtlich einer direkten Anbindung von technisch weniger versierten Anwendern zu bieten. Die Idee, Anwender auch am Entwurf von Teilen der Klassenhierarchie zu beteiligen, mag vielen Entwicklern als utopisch erscheinen, könnte jedoch meiner Meinung nach genau dann möglich sein, wenn ins besonders der fachliche Klassenentwurf eng an den Begriffen der realen Arbeitszusammenhänge entsteht und mit diesen, sich verändernden Arbeitsformen, wachsen kann. Genau dies würde meiner Meinung nach zu einer erhöhten *fachlichen Stabilität* führen und spätere Änderungen kostengünstiger gestalten. Eine genauer Diskussion dieses Aspektes kann an dieser Stelle nicht mehr geführt werden, jedoch wäre es eine lohnenswerte Aufgabe, hier tiefergehende Forschung zu betreiben.

6. Zusammenfassung und Ausblick

Diese Arbeit ist im Zuge ihrer Entstehung in viel stärkerem Maße gewachsen, als ich es erwartet hatte. Nach dem bisher gesagten erscheint dies nicht allzu verwunderlich, weil nicht nur Software evolutionär entsteht, sondern sich selbstverständlich auch wissenschaftliche Arbeiten in dieser Weise entwickeln. Genauso wie es gewissermaßen erfreulich wäre, wenn man schon zu Beginn eines Softwareprojektes genau wüßte, was für ein System entstehen soll, genauso wäre ich froh gewesen, von Beginn an die Richtung und Zielsetzung dieser Arbeit, inkl. *aller* wichtiger Aspekte zu kennen. So stellte sich auch für mich die Entstehung dieser Arbeit als ein Lernprozeß dar.

Eine abschließende Bewertung des Projektverlaufes fällt nicht leicht, weil je nach Rolle des Betrachters unterschiedliche Sichtweisen und Interessen vorhanden sind. Auch ich selber habe während der Entstehung dieser Arbeit verschiedene Perspektiven zum Projekt eingenommen. Als Entwickler habe ich eine sehr gut funktionierende Teamarbeit erlebt und mich vollständig mit dem technischen Erfolg meiner Arbeit und der meiner Entwicklerkollegen identifiziert. Konfrontiert mit der realen Anwendungspraxis war ich zunächst überrascht über teilweise komplett andere Bewertungen des Projektergebnisses. Daraus resultierte eine gewisse Relativierung des Projekterfolges. Während der Untersuchung der den Schwierigkeiten zugrundeliegenden Ursachen habe ich u.a. durch wachsende Erkenntnisse gängiger Probleme von Softwareprojekten und dem Verstehen der Theorie über Modellmonopole eine teilweise veränderte Sicht auf Softwareentwicklung bekommen.

Ich sehe im vorliegenden Projekt den grundlegenden Konflikt zwischen Kosten und Terminen auf der einen Seite und Kundenorientierung auf der anderen Seite. Ich sehe es als kritisch an, daß diesem Spannungsverhältnis im Projektverlauf nicht durch ein Mehr an Kommunikation begegnet wurde, sondern direkte Kommunikation zu wenig Unterstützung in der Vorgehensweise fand. Daraus entstand meiner Meinung nach eine verstärkte Interessen-polarisierung zwischen Softwarehaus und Auftraggeber, zu der es keine adäquate Vertrauensbasis gab, die zur Lösung solcher Interessengegensätze nötig gewesen wäre. Daraus resultiert u.a. die skeptischen Grundhaltung vieler Mitarbeiter des Auftraggebers bezüglich einer weiteren Zusammenarbeit mit dem Softwarehaus und damit ein noch höherer Druck auf zukünftige Projekte, der in einem *Teufelskreis* enden könnte, wenn in der Zukunft mit noch stärkerer Kommunikationsverengung reagiert würde (vgl. [ThomSchulz90]).

Lösungen für diese grundsätzlichen Schwierigkeiten können nur von einer Anpassung der gewählten Vorgehensweise kommen. Thesenartig lassen sich mögliche Richtungen für Veränderungen folgendermaßen formulieren:

- *Betroffene* zu *Beteiligte* machen. *Teambildung* und *Teamentwicklung* unterstützen.
- Aufhebung einer strikten, auch räumlichen Trennung von Anwendern und Entwicklern, hin zum Arbeiten in *Teams* gleichberechtigter Experten.

- *Rollenverständnisse* ändern: Aus Softwareentwicklern werden Berater des Kunden in Fragen der Softwareunterstützung, aus Anwendern und Anwendervertretern werden Berater der Entwickler in Fragen der fachlichen Zusammenhänge.

Mein Eindruck ist, daß eine enge Einbeziehung der Anwender im Projektverlauf nicht nur wünschenswert, sondern für die Entstehung hochwertiger Softwaresysteme unerläßlich ist. Im vorliegenden Projekt führten die vornehmlich strategischen Zielsetzungen jedoch zu einem immer stärkeren Ausschluß gerade der Anwender aus dem Prozeß der Softwareentstehung. Da aber Benutzerinvolvierung sowohl vom Auftraggeber als auch vom Softwarehaus durchaus erwünscht sind, sehe ich als Hauptschwierigkeit das Fehlen von erprobten alternativen Vorgehensweisen, was dazu führt, daß man an bestehender Methodik festhält oder versucht, bestehende Methodik in sich zu ändern, ohne dabei grundsätzliche Probleme zu beseitigen.

Für zukünftige Projekte müssen Maßnahmen im Vordergrund stehen, die Kommunikation und Teambildung unterstützen und speziell ein wirkliches Verständnis des Problembereiches im Entwicklerteam fördern. Hierzu kommen Ansätze der evolutionären Softwareentwicklung in Frage.

Die Ideen der evolutionären Softwareentwicklung sind auch im Softwarehaus nicht unbekannt, jedoch sind sie mit starken Zweifeln hinsichtlich Projektkontrollierbarkeit überhaupt und der Durchführbarkeit unter engem Zeit- und Geldbudget belegt. Gerade deshalb ist es notwendig, anhand praktischer Beispiele den Beweis anzutreten, daß genau das Umgekehrte wahr ist. Um mit Kilberth et al. zu sprechen: *"Der hier vertretende Ansatz, der evolutionären Systementwicklung in der Kombination mit einer objektorientierten Methodik, die eine maximale Änderbarkeit zu möglichst geringen Kosten verspricht, sollte vor dem Hintergrund dieser Überlegungen gesehen werden…Im weiteren zeigen wir, daß eine Abkehr vom Phasenmodell nicht eine Beliebigkeit und Unkontrollierbarkeit eines Softwareprojekts zur Folge hat, sondern sogar verbesserte Voraussetzungen zur Projektsteuerung und -kontrolle bietet."* ([KilGryZü93])

Damit ist gesagt worden, daß Systeme, die explizit evolutionär und partizipartiv entwickelt werden, nicht nur den Anforderungen der Anwender besser gerecht werden, sondern auch leichter und kostengünstiger geändert und erweitert werden können, d.h. daß sie einen gleichmäßigeren und letztlich günstigeren Kostenverlauf haben und nicht, wie in der traditionellen Vorgehensweise, eine Kostenverlagerung in die Zukunft stattfindet. Es gibt konkrete Beispiele für den erfolgreichen Einsatz evolutionärer Vorgehensweisen, maßgeblich im skandinavischen, aber auch im deutschsprachigen Raum (vgl. z.B. [BjeEhmKyn87], [KieLiSchZü92], [BüGryZü92]).

Ich will darauf hinweisen, daß auch die vom Softwarehaus und Auftraggeber vereinbarte geänderte Vorgehensweise für zukünftige Projekte teilweise evolutionäre Züge aufweist. Hier wurde u.a. die relativ frühe Auslieferung eines produktionsreifen Teilsystems beschlossen, der dann von einzelnen Anwendern erprobt werden soll. Hier liegt die schon oben beschriebene Erfahrung zugrunde, daß viele Fehler erst entdeckt werden, wenn die Anwender mit dem "echten" System arbeiten. Wenn man nur einen Schritt weitergeht und noch früher Teil-

systeme ausliefert, die u.U. sogar produktionsreif sind, dann hat man eine prototypische Vorgehensweise bereits erreicht. Zentral ist dann allerdings, daß die ausgelieferten Teilsysteme wirklich *Prototyp* oder *Pilotsystem* genannt werden, um damit eine viel größere Offenheit gegenüber Kritik und Verbesserungsvorschlägen zu erzeugen.

In zukünftigen Projekte sollte meiner Meinung nach besonders die direkte und kontinuierliche Kommunikation unter den Beteiligten in den Mittelpunkt rücken und von der Methodik explizit gewürdigt und unterstützt werden. Dafür ist es vor allem wichtig, Dokumenttypen einzuführen, die von allen Parteien gleich gut verstanden werden und somit das Entstehen eines gemeinsamen Verständnisses (z.B. einer gemeinsamen *"Projektsprache"*) über das zu entwickelnde System fördern. Das Verständnis der Entwicklungsdokumente speziell durch die Anwender ist entscheidend, weil so die eigentlichen "Experten" der Anwendungswelt den Entwicklungsstand bewerten und produktiv beeinflussen können. Nur so ist eine frühzeitige und kontinuierliche Qualitätssicherung möglich. Passende Dokumenttypen, wie Glossar, Szenarios und Prototypen sind ausführlich z.B. von Kilberth et al. vorgestellt und diskutiert worden ([KilGryZü93]). Für zukünftige Projekte kann meiner Meinung nach nicht auf den Einsatz von Prototypen verzichtet werden, da Prototypen als lauffähige Programmteile besonders gut für eine frühzeitige Qualitätssicherung geeignet erscheinen. *Prototyping* als Vorgehensweise wurde u.a. von Budde et al. und Kieback et al. mit einiger Gründlichkeit erörtert (vgl. [BüGryZü92], [KieLiSchZü92]). Praktische Erfahrungen mit dem Erfolg prototypischer Vorgehensweise liegen vor (vgl. z.B. [Bäum93]).

Literaturverzeichnis

[Bäum93] D.Bäumer: *Gebos - Ein Praxisbeispiel zu objektorientierter Analyse und objektorientiertem Entwurf*, HMD 170, 1993

[Boehm76] B.W.Boehm: *Software engineering,* IEEE Transactions on Computers, 25(12) 1976

[Boehm88] B.W.Boehm: *A Spiral Model of Software development and Enhancement,* in Computer 5/1988

[Booch91] G.Booch: *Object-Oriented Design with Applications,* Benjamin/Cummings 1991

[Bråten73] S.Bråten: *Model Monopoly,* in Acta Sociologica, 16(2), 1973

[Brooks75] F.Brooks: *The Mythical Man-Month,* Addison-Wesley 1975

[Brooks87] F.Brooks: *No Silver Bullet,* in IEEE Computer 20(4) 1987

[BüGryZü92] U.Bürkle, G.Gryczan, H.Züllighoven: *Erfahrungen mit der objektorientierten Vorgehensweise bei einem Bankenprojekt*, Informatik-Spektrum, Band 15, Heft 5, Okt.1992

[BuKaKuZü92] R.Budde, K.Kautz, K.Kuhlenkamp, H.Züllighoven: *Prototyping - An Approach to Evolutionary System Development*, Springer-Verlag 1992

[BuZü90] R.Budde, H.Züllighoven: *Software-Werkzeuge in einer Programmierwerkstatt*, Oldenbourg Verlag 1990

[Dahl92] B.Dahlbom: *The idea that Reality is Socially Constructed,* in [FloZüBuKe92]

[DahlMath93] B.Dahlbom, L.Mathiassen: *Computers in Context,* Blackwell Publishers 1993

[FloZüBuKe92] C.Floyd, H.Züllighoven, R.Budde, R. Keil-Slavik: *Software Development and Reality Construction*, Springer-Verlag 1992

[Floyd92] C.Floyd: *Software Development as Reality Construction,* in [FloZüBuKe92]

[Floyd93] C.Floyd: *Einführung in die Softwaretechnik*, Script zur gleichnamigen Vorlesung an der Universität Hamburg, 1993

[Floyd94] C.Floyd: *Software-Engineering - und dann?,* Informatik-Spektrum(1994)17, Springer-Verlag 1994

[GryZü92] G.Gryczan, H.Züllighoven: *Objektorientierte Systementwicklung. Leitbild und Entwicklungsdokumente*, Informatik-Spektrum, Band 15, Heft 5, Okt.1992

[Gold90] A.Goldberg: *Informations, Models, Views and Controllers*, in Dr.Dobb's Journal, Juli 1990

[Henders92] B.Henderson-Sellers: *A Book of Object-Oriented Knowledge*, Prentice Hall 1992

[KieLiSchZü92] A.Kieback, H.Lichter, M.Schneider-Hufschmidt, H.Züllighoven: *Prototyping in industriellen Software-Projekten*, Informatik-Spektrum (1992) 15: 65-77, Springer Verlag 1992

[KilGryZü93] K.Kilberth, G.Gryczan, H.Züllighoven: *Objektorientierte Anwendungsentwicklung*, Verlag Vieweg 1993

[MamOppTep86] P.Mambrey, R.Oppermann, A.Tepper: *Computer und Partizipation*, Westdeutscher Verlag 1986

[Naur85] P.Naur: *Programming as Theory Building*, In: Microprocessing and Microprogramming, Vol. 15, 1985

[Pasch92] J.Pasch: *Dialogischer Software-Entwurf*. Dissertation, Technische Universität Berlin, 1992

[PomBla93] G.Pomberger, G.Blaschek: *Grundlagen der Software engineering: Prototyping und objektorientierte Software-Entwicklung*, Hanser Verlag 1993

[Reisin91] F.-M.Reisin: *Kooperative Gestaltung in partizipativen Softwareprojekten*, Dissertation, Technische Universität Berlin, 1991

[Salzm89] H.Salzman, S.Rosenthal: *Organizational Dimensions of the Software Design Process*, Center for Applied Social Science, Bosten, 1989 (zitiert nach [WelOrt92])

[SchulzvonThun81] F.Schulz von Thun: *Miteinander reden 1 - Allgemeine Psychologie der Kommunikation*, Rowohlt 1981

[SchulzvonThun91] F.Schulz von Thun: *Miteinander reden 2 - Differenzielle Psychologie der Kommunikation*, Rowohlt 1991

[ThomSchulz90] C.Thoman, F.Schulz von Thun: *Klärungshilfe - Theorien, Methoden, Beispiele*, Rowohlt 1990

[WatzBeaJa67] P.Watzlawick, J.H.Beavin, D.D.Jackson: *Pragmatics of Human Communication*, Norton & Company 1967

[WelOrt92] F.Weltz, R.G.Ortmann: *Das Softwareprojekt*, Campus Verlag 1992

[Wendt92] S.Wendt: *Defizite im Software Engineering*

[WirfWiWi90] R.Wirfs-Brock, B.Wilkerson, L.Wiener: *Designing Object-Oriented Software*, Prentice Hall 1990