GI, the Gesellschaft für Informatik, publishes this series in order

- to make available to a broad public recent findings in informatics (i.e. computer science and information systems)
- to document conferences that are organized in cooperation with GI and
- to publish the annual GI Award dissertation.

Broken down into the fields of "Seminars", "Proceedings", "Monographs" and "Dissertation Award", current topics are dealt with from the fields of research and development, teaching and further training in theory and practice. The Editorial Committee uses an intensive review process in order to ensure the high level of the contributions.

The volumes are published in German or English.

Information: http://www.gi-ev.de/LNI

ISSN 1617-5468
ISBN 978-3-88579-192-8

The 2006 Software Management conference on „Beratung in der Softwareentwicklung" (Consulting in Software Development) took place in Berlin from November 8th to 10th.

This volume contains the papers of three invited lectures and nine selected lessons as well as short descriptions of three tutorials giving deep insights into the managerial, technical, and social aspects of consulting services for software projects. All papers are in German language.

Hans Brandt-Pook, Werner Simonsmeier, Thorsten Spitta (Hrsg.): WI-MAW

P-98

GI-Edition

Lecture Notes in Informatics

Hans Brandt-Pook, Werner Simonsmeier und Thorsten Spitta (Hrsg.)

Beratung in der Softwareentwicklung Modelle – Methoden – Best Practices

Fachtagung des GI-Fachausschusses „Management der Anwendungsentwicklung und -wartung" im Fachbereich Wirtschaftsinformatik (WI-MAW), Berlin, 2006

Proceedings

Hans Brandt-Pook, Werner Simonsmeier
und Thorsten Spitta (Hrsg.)

Beratung in der Softwareentwicklung
Modelle – Methoden – Best Practices

**Fachtagung des GI-Fachausschusses
Management der Anwendungsentwicklung und -wartung
im Fachbereich Wirtschaftsinformatik (WI-MAW)**

**08. - 10. November 2006
in Berlin**

Gesellschaft für Informatik 2006

Lecture Notes in Informatics (LNI) - Proceedings
Series of the Gesellschaft für Informatik (GI)

Volume P-98

ISBN 978-3-88579-192-8
ISSN 1617-5468

Volume Editors
Prof. Dr.-Ing. Hans Brandt-Pook
Informatik/Wirtschaftsinformatik
Fachbereich Produktion und Wirtschaft
Fachhochschule Lippe und Höxter
Liebigstr. 87
32657 Lemgo
E-Mail: Hans.Brandt-Pook@fh-luh.de
Dipl.-Math. Werner Simonsmeier
sd&m AG
software design & management
Kurfürstendamm 22
10719 Berlin
E-Mail: werner.simonsmeier@sdm.de
Prof. Dr. Thorsten Spitta
Angewandte Informatik/Wirtschaftsinformatik
Fakultät für Wirtschaftswissenschaften
Universität Bielefeld
Postfach 100131
33601 Bielefeld
E-Mail: thspitta@wiwi.uni-bielefeld.de

Series Editorial Board
Heinrich C. Mayr, Universität Klagenfurt, Austria (Chairman, mayr@ifit.uni-klu.ac.at)
Jörg Becker, Universität Münster, Germany
Ulrich Furbach, Universität Koblenz, Germany
Axel Lehmann, Universität der Bundeswehr München, Germany
Peter Liggesmeyer, TU Kaiserslautern und Fraunhofer IESE, Germany
Ernst W. Mayr, Technische Universität München, Germany
Heinrich Müller, Universität Dortmund, Germany
Heinrich Reinermann, Hochschule für Verwaltungswissenschaften Speyer, Germany
Karl-Heinz Rödiger, Universität Bremen, Germany
Sigrid Schubert, Universität Siegen, Germany

Dissertations
Dorothea Wagner, Universität Karlsruhe, Germany
Seminars
Reinhard Wilhelm, Universität des Saarlandes, Germany

© Gesellschaft für Informatik, Bonn 2006
printed by Köllen Druck+Verlag GmbH, Bonn

Vorwort

Beratungstätigkeit als Bindeglied zwischen den Beteiligten spielt im Prozess der Anwendungsentwicklung eine große Rolle. Was in frühen Phasen nicht eindeutig geklärt ist, führt in den späteren Phasen eines Projektes zu Problemen – bis hin zum Scheitern des ganzen Vorhabens. Externe Berater spielen in praktisch allen großen Softwarevorhaben eine entscheidende Rolle. Die Fachtagung *Software Management 2006* macht daher den Aspekt der Beratung im Softwaremanagement zu einem thematischen Schwerpunkt. Die zum sechsten Mal stattfindende Tagung des Fachausschusses *Management der Anwendungsentwicklung und -wartung* in der *Gesellschaft für Informatik* ist das Forum für Fachleute, die in Praxis und Wissenschaft mit dem Management von Softwaresystemen befasst sind. Sie dient der Diskussion des State-of-the-Art sowie dem Gedanken- und Erfahrungsaustausch zu aktuellen Problemstellungen und Forschungsergebnissen. Auf der Konferenz am 9. und 10. November 2006 stellen erfahrene Praktiker und Fachleute aus der Hochschule ihre Konzepte vor und berichten über Beratungsprojekte und andere Themen im Softwaremanagement.

Im ersten Teil der Dokumentation dieser Fachtagung finden sich die eingeladenen Vorträge. Heinrich C. Mayr geht der Frage nach, ob Anforderungsmodellierung und Ontologiebildung das A & O der Softwareentwicklung sind. Den Projektvertrag im Softwaremanagement beleuchtet Benedikt Wemmer aus der juristischen Perspektive. Gerhard Wohland formuliert Anforderungen an die Umgebung von Projektteams, die Höchstleistungen vollbringen.

Der zweite Teil dieser Dokumentation umfasst die Beiträge aus Hochschule und Praxis.

Zunächst geht es um die Beratung im engeren Sinne. Hans Brandt-Pook und Axel R. Raulinat stellen den Ansatz der Wirtschaftsmediation im Kontext des Softwaremanagements vor. Thomas Deelmann beschreibt Rahmenbedingungen und Geschäftsmodelle in der IT-Beratung. Heiner Merz und seine Mitautoren erläutern die Unterstützung bei der Auswahl eines IT-Dienstleisters. Harry Sneed rundet diese Thematik mit einem Beitrag zur Beratung in der Softwareentwicklung und -weiterentwicklung ab.

Es folgen zwei Praxisbeiträge, die auf den Projektprozess fokussieren: Randolph Kappes stellt die Methodik sowie die Erfolgsfaktoren eines Projektaudits vor, und Hubert Surrer betrachtet Nachlässigkeiten, Irrtümer und Methoden in der Projektführung in Kundenprojekten.

Die Tagung *Software Management* bietet Platz für die ganze Themenbreite in der Softwareentwicklung und -wartung. Darum beschließen Beiträge zu aktuellen technologischen Fragen den Teil der Beiträge aus Hochschule und Praxis. Jan Leßner erläutert die Entwicklung von Systemfamilien. Maik Thränert und Andrej Werner zielen auf die Wiederverwendung von Entwicklungsprozessen durch Prozessfamilien. Ulrich Kriegel und Holger Kurrek berichten von der Einführung einer SOA für den Senat des Landes Berlin.

Die Tagung wird traditionsgemäß eröffnet durch halbtägige Tutorials, die einige Aspekte des Schwerpunktthemas vertiefen. Die Beschreibung der Tutorials bilden den dritten Teil dieser Dokumentation: Randolph Kappes erläutert das Projektaudit, Axel Raulinat und Hans Brandt-Pook befassen sich mit der Wirtschaftsmediation und Uwe Vigenschow fokussiert auf notwendige Sozialkompetenzen in Softwareprojekten.

Die Tagung zeichnet sich, wie bereits die bisherigen, durch Rahmenbedingungen aus, die eine Atmosphäre des Diskutierens und Kennenlernens fördern. Deshalb ist die Zahl der Fachvorträge klein gehalten worden, um Raum für Gespräche und Diskussionen zu schaffen.

Hans Brandt-Pook, Werner Simonsmeier und Thorsten Spitta

Lemgo, Berlin und Bielefeld, im September 2006

Programmkomitee

Jens Borchers, Steria Mummert Consulting AG, Hamburg
Prof. Dr. Hans Brandt-Pook, FH Lippe und Höxter (Leiter Programmkomitee)
Prof. Dr. Stefan Eicker, Uni Essen
Karin Beate Elbrechter, Commerzbank AG, Frankfurt am Main
Martin Engstler, Fraunhofer- Institut IAO, Stuttgart
Hans-Joachim Etzel, IFS Consulting, Eschborn
Prof. Dr. Wolfgang Hesse, Uni Marburg
Gerrit Kerber, aragon INTERACTIVE, Fellbach
Dr. Ralf Kneuper, Darmstadt
Dr. Christian Kop, Uni Klagenfurt
Prof. Dr. Franz Lehner, Uni Passau
Prof. Dr. Heinrich C. Mayr, Uni Klagenfurt
Dr. Jörg Noack, Deutsche Leasing AG, Bad Homburg
Prof. Dr. Andreas Oberweis, Uni Frankfurt
Bernd Oestereich, oose.de, Hamburg
Prof. Dr. Reinhard Richter, FH Karlsruhe
Werner Simonsmeier, sd&m, Berlin (Leiter Organisationskomitee)
Harry M. Sneed, AneCon GmbH, Wien
Prof. Dr. Thorsten Spitta, Uni Bielefeld
Prof. Dr. Wolffried Stucky, Uni Karlsruhe
Dr. Karin Vosseberg, pdv.com, Bremen
Manuela Wiemers, Fourth Project Consulting, Wolfsburg

Inhaltsverzeichnis

Teil 1: Konferenz 10

Heinrich C. Mayr
Anforderungsmodellierung und Ontologiebildung:
Das A & O der Softwareentwicklung? 10

Benedikt Wemmer
Typische Krisenthemen beim Projektvertrag aus der Perspektive
des Auftraggebers 11

Gerhard Wohland
Das widerständige Nest 13

Hans Brandt-Pook, Axel R. Raulinat
Neue Chancen im eskalierten Konflikt 22

Thomas Deelmann
Rahmenbedingungen und Geschäftsmodell der IT-Beratung 35

Heiner Merz, Andreas Helferich, Georg Herzwurm, Andreas Platzer
Beratung bei der Auswahl eines IT-Dienstleisters:
Das Projekt Chefplan Online 49

Harry M. Sneed
Beratung in der Softwarewartung und –weiterentwicklung 61

Randolph Kappes
Projektaudit – methodisch Projekte sanieren 74

Hubert Surrer
Projektführung in Kundenprojekten 82

E. Ulrich Kriegel, Holger Kurrek
Erfahrung bei der Einführung einer SOA für den Senat für Bildung,
Jugend und Sport des Landes Berlin 91

Jan Leßner
Systemfamilienentwicklung bei arvato direct services 104

Maik Thränert, Andrej Werner
Wiederverwendung von Entwicklungsprozessen durch Prozessfamilien 118

Teil II Tutorials (Kurzfassungen) 133

Randolph Kappes
Projektaudit: Eine Methode zur kurzfristigen Standortbestimmung 133

Axel R. Raulinat, Hans Brandt-Pook
WirtschaftsMediation im Softwaremanagement: Konflikte produktiv lösen 135

Uwe Vigenschow
Sozialkompetenz in Software-Projekten 138

Anforderungsmodellierung und Ontologiebildung: Das A & O der Softwareentwicklung?

Heinrich C. Mayr

Universität Klagenfurt
Universitätsstr. 65-67
A-9020 Klagenfurt
Heinrich.Mayr@uni-klu.ac.at

Typische Krisenthemen beim Projektvertrag aus der Perspektive des Auftraggebers

Benedikt Wemmer

Brehm & v. Moers, Rechtsanwälte - Steuerberater
Anna-Louisa-Karsch-Str. 2
10178 Berlin
bw@bvm-law.de

Abstract: Softwareprojektverträge sind zumeist sehr komplexe Vertragswerke, die über die reine Erstellung einer Software weit hinausgehen. Gerade bei größeren Eingriffen durch die Software in den unternehmerischen Ablauf steigen die Risiken auf beiden Seiten und somit auch das Konfliktpotential. Darüber hinaus ist mit den meisten Projektverträgen auch eine dauerhafte Wartung verbunden, die einen weiteren Herd für Konflikte darstellen kann. Im Rahmen des Vortrags können diese Punkte nur angeschnitten werden. Die ultimative Lösung für die Ausgestaltung solcher Vertragswerke gibt es leider nicht, so dass es Kern des Vortrags ist, die Hörer zu sensibilisieren und Vorschläge für Lösungen im Interesse des Auftraggebers zu geben.

1. Pflichtenheft – Milestones - Qualität

Häufigster Streitpunkt ist die Frage, ob die Software „wie vereinbart“ erstellt wird oder wurde, ob bestimmte Funktionen geschuldet waren oder nicht, ob der Softwarehersteller berechtigt ist, eine Änderung zusätzlich in Rechnung zu stellen oder ob das Fehlen einer Funktion zur Forderung von Schadensersatz seitens des Auftraggebers berechtigt.

Zwar drängt meist die Zeit, jedoch darf darunter niemals die genaue Bestimmung der Funktionen und der Leistungsfähigkeit leiden. Dies sind oft im Nachhinein die schwierigsten Streitpunkte mit dem Auftragnehmer, die normalerweise verhindert werden können. Denkbar sind auch fortzuschreibende Leistungsverzeichnisse im Projektverlauf. Weiterhin sollten einzelne Leistungsteile in möglichst genauen Abständen (Milestones) begutachtet und teilabgenommen werden. Auch so können Probleme früh erkannt und beseitigt werden.

Auch die Vereinbarung eines Qualitätsstandards oder die frühzeitige Benennung eines unabhängigen Experten schon im Vertragswerk, der während der Entwicklung über fachliche Fragen entscheidet, können durchaus hilfreich sein

2. Fertigstellungsfristen

Ein Projektvertrag muss strikte Regelungen für (Teil-) Abnahmen enthalten. Verzögerungen müssen mit einer Vertragsstrafe bedroht werden und es müssen Mechanismen geschaffen werden, die für eine frühzeitige Erkennung von möglichen Verzögerungen sorgen. Dies kann insbesondere durch eine schematisierte Erfassung von Verzögerungsanzeigen seitens des Auftragnehmers geschehen. Hier ist besonders auf die klare Definition der erlaubten Verzögerungsgründe zu achten.

3. Insolvenzschutz - Vertragsbeendigungen

Wohl das größte Problem für beide Seiten ist das Risiko einer Vertragsbeendigung unabhängig vom Rechtsgrund. Wird der meist auf die Softwareentwicklung folgende Wartungsvertrag beendet, so ist zwar der Auftraggeber im Besitz der Software, doch meist liegen die Quellcodes beim Entwickler. Eine Wartung oder Weiterentwicklung ohne die Quellcodes ist in den allermeisten Fällen aussichtslos. Andererseits kalkulieren Auftragnehmer die Entwicklung meist mit deutlich niedrigeren Werten in der Erwartung der Einnahmen aus der Wartung. Hier gilt es durch Hinterlegung des Quellcodes bei einem Dritten eine für alle Seiten gangbare Lösung zu finden. Diese kann z.B. in einem Eigentumsübergang auf den Auftraggeber bei Kündigung mit einer gewissen Zahlungsverpflichtung an den Entwickler, gestaffelt nach bisheriger Vertragslaufzeit liegen.

4. Angemessene Vergütung

Die Reform des Urheberrechts hat dafür gesorgt, dass ein Urheber nunmehr immer einen Anspruch auf Leistung einer „angemessenen Vergütung“ hat. Aller Voraussicht nach dürfte dies auch für den Entwickler einer Software gelten. Ein solcher Vergütungsanspruch ist unabhängig von der vertraglich vereinbarten Vergütung und die Einschätzung der Angemessenheit liegt nach bisheriger Rechtslage bei den – oft auch nicht erfahrenen – Gerichten. Hier gilt es Mechanismen zu finden, die zumindest das Risiko einer Nachforderung verringert. Denkbare Modelle sind hier Gewinnbeteiligungen, soweit die Software direkt selbst zur Gewinnerzielung verwendet wird oder weiter veräußert wird, aber auch die Kontrolle der Gehälter beim Auftragnehmer kann hier notwendig werden.

Das widerständige Nest

Anforderungen an die Umgebung von Projektteams mit höchster Leistung

Gerhard Wohland

Wohland & Partner
Kürnsteig-Weg 14
72141 Walddorfhäslach
Gerhard.Wohland@t-online.de

Abstract: Das „Widerständige Nest“ ist eine Metapher für die Umgebung von Projekten, die für ihren Erfolg höchste Leistung benötigen. Sie ist entstanden in vielen Sanierungen von Projekt-Havarien und dem Studium von Höchstleistung. Die Detecon-Studie "Dynamik-robuste Höchstleistung"[1] kommt zum Ergebnis: Im Kontext hoher Dynamik stammt der Marktdruck von Unternehmen mit dynamik-robuster Höchstleistung. Diese Form der Wert-Schöpfung ist die schnellste, einfachste und billigste Lösung für moderne Dynamik-Probleme. Sie ist allerdings Kultur-empfindlich und deshalb noch selten anzutreffen. Dies gilt auch und vor allem für das Management von Projekten. Höchstleistung im Projektmanagement ist die Überwindung der durch Methoden gesetzten Grenzen. Wir bezeichnen es deshalb als trans-methodisches Projektmanagement. Dabei ist nicht nur das „Innen" und "Außen" eines Projektes das Thema, sondern vor allem die Kopplung durch Trennung von beidem – kurz: das „Widerständige Nest“.

1. Einleitung

Noch ist die Meinung verbreitet, dass die Beherrschung und Anwendung von Methoden ausreichende Basis für gutes Projektmanagement ist. Methoden sparen Zeit und Geld und sichern den Erfolg - sagt man. Andererseits ist Konsens, dass die Dynamik moderner Wirtschaft ansteigt. Hohe Dynamik bedeutet mehr Überraschendes und damit weniger Wiederholung. Für Überraschendes kann es keine Methode geben – sonst wäre es keine Überraschung. Je höher die Dynamik, umso geringer ist der Nutzen von Methoden - im Grenzfall schaden sie mehr, als sie nutzen. Das gilt auch für Projekte in dynamischer Umgebung.

[1] "Vom Wissen zum Können – Dynamik robuste Höchstleistung" Studie der Detecon. Kostenlos unter: https://www.detecon.com/de/publikationen/bestellen_stud.php?pub_id=1000

Das Management solcher Projekte muss, zumindest für die dynamischen Problem-Anteile, über das gewohnt Methodische hinausgehen. Das Methoden-basierte Management (Verwaltung) benötigt Talent-basierte Führung als Ergänzung. Bleibt diese transmethodische Ergänzung aus, wächst die Havarie-Gefahr. Besonders dann, wenn versucht wird, diesen transmethodischen Mangel durch immer neue und strengere Methoden zu beheben. Nur wenn die transmethodische Ergänzung gelingt, werden die Methoden von unverdaulicher Dynamik entlastet und funktionieren wieder. Dynamik wirkt wie Sand im Getriebe der Methoden.

Notwendig ist also nicht das Methodische zu ersetzen, sondern zu ergänzen und dann für die synergetische Koexistenz beider „Welten“ zu sorgen, denn sie sind aufeinander angewiesen. Wird versucht, die Schwächen in einer Welt durch Stärken in der anderen auszugleichen, entstehen Havarien. "Hohe Dynamik braucht bessere Methoden" ist eine Denkfalle.

Dynamik-robuste Höchstleistung im Projektmanagement erfordert die Überwindung der durch Methoden gesetzten Grenzen. Deshalb wird Höchstleistung im Projektmanagement als transmethodisches Projektmanagement bezeichnet.

2. Das widerständige Nest für dynamische Projekte

Fast jedes Unternehmen hat seine eigene Auffassung von dem, was ein Projekt ist. Oft gibt es sogar im gleichen Unternehmen mehrere Auffassungen. Die Definitionen reichen vom schlichten Thema, welches behandelt werden soll, über besondere Aufgaben für Linieneinheiten bis zur temporären Sonderorganisation außerhalb der Linie.

Wir folgen Letzterer: Ein Projekt ist eine temporär angelegte Sonderorganisation. Sie wird gebildet, wenn Aufgaben zu lösen sind, welche die Institutionen der Linie nicht ohne Beschädigung übernehmen können.

Ein Projekt ist also immer eine temporäre Organisation in einer permanenten Organisation als Umgebung – eine Organisation in einer Organisation. Deswegen hat transmethodisches Projektmanagement nicht nur das Projekt selbst zum Thema, sondern vor allem die Kopplung mit seiner Umgebung.

Wie die Studie gezeigt hat, basiert Höchstleistung nicht auf Wissen, sondern auf Können und Talent. Dieses entfaltet sich nur in einer vertrauensvollen Geborgenheit, welche die unvermeidlichen Niederlagen bis zum Erfolg ertragbar macht. Andererseits wird das Urteil über Leistung letztlich von außen gesprochen. In der Wirtschaft zum Beispiel ist Höchstleistung nur, was Verkaufbares erbringt. Höchstleistung braucht also neben dem "warmen Nest" auch widerständige Bewertung. Beides muss von der Umgebung eines Projektes bereitgestellt werden.

In den vergangenen hundert Jahren hielten sich die Unternehmen, tief in ihrem Innern (Zentrum), kreative Ghettos für ihren Bedarf an talentbasierter Höchstleistung. Heute wird Höchstleistung auch in der marktnahen Wertschöpfung (Peripherie) benötigt. Auch hier widerständige Nester zu bauen gelingt erst wenigen, denn die wichtigste Voraussetzung ist die Entwicklung der Kultur in der eigenen Wertschöpfung.

Dieses Kultur-Problem post-tayloristischer Arbeitsorganisation besser zu verstehen, ist das Ziel dieses Textes.

3. Havarien

Die methodische und die dynamische Welt zu unterscheiden hilft, die Kultur-Bedingungen für Höchstleistung besser zu verstehen: Die Basis der methodischen Welt sind Regeln und Methoden, die der dynamischen sind Prinzipien und Ideen.

In der folgenden Abbildung (Abb. 1) ist versucht, diese beiden Denk-Welten durch Begriffspaare zu unterscheiden. Diese Unterscheidungen werden im Alltag nur unscharf oder gar nicht benutzt, weil beide Seiten immer nur gemeinsam auftreten. Aber erst die präzise Unterscheidung macht es möglich Zusammenhänge und Abhängigkeiten zu beschreiben. Die beiden Welten sind nur begrifflich im Denken zu trennen, nicht aber in der Realität.

Der Charakter eines Projektes wird bestimmt vom Charakter des Problems, welches zu lösen ist. Probleme, die durch hohe Umgebungs-Dynamik entstehen, brauchen dynamik-robuste Projekte zur Lösung.

Allerdings verschwindet auch bei höchster Dynamik der methodische Problem-Anteil nicht. Beide Welten bleiben immer koexistent und aufeinander angewiesen. Ist einer der beiden Teile überfordert, so funktioniert auch der andere nicht. Wenn Schwierigkeiten auftreten, ist nicht ohne Aufwand zu entscheiden, in welchem der beiden Teile die Ursache steckt. Heute steckt die Ursache für fast alle Havarien im dynamischen Teil. Der Sanierungs-Aufwand wird aber meist auf den formalen Teil beschränkt, denn ohne Alternativen zum Methodischen ist nur dieser sichtbar. Havarien entstehen, wenn Problem und Werkzeug nicht zueinander passen. Zur dynamischen Überraschung passt als Werkzeug nur eine Idee, keine Methode. Für Routine ist es umgekehrt. Hier werden Methoden benötigt. Eine Idee als Werkzeug wäre wie die Neuerfindung des Rades.

Im Kontext hoher Dynamik wächst der dynamische Anteil von Projekten.
Wird nur der formale Anteil beachtet, entstehen Havarien

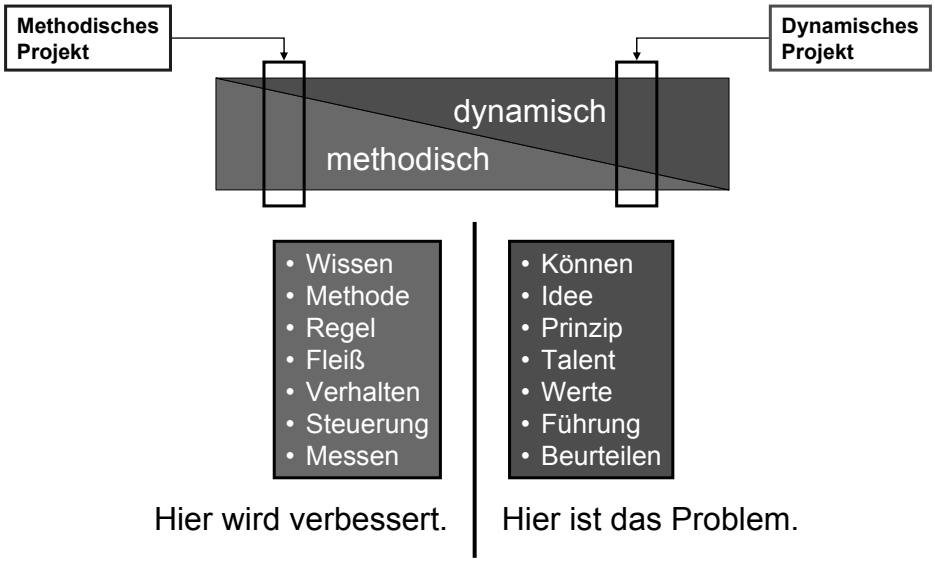


Abbildung 1: Jedes Projekt ist eine sich verändernde "Mischung" aus methodischen und dynamischen Anteilen.

Schutzräume

Wie gesagt, ein Projekt ist eine Organisation in einer Organisation. Dynamische Projekte verhalten sich zur tayloristischen Mutterorganisation wie Feuer zu Wasser. Ihr direkter Kontakt ist destruktiv. Eine Seite gewinnt auf Kosten der anderen. Meist gewinnt der Immunapparat der tayloristischen Umgebung.

Im Alltag des Unternehmens bilden sich deshalb so genannte Schutzräume. Ohne bewusste Gestaltung sind diese meist destruktiv, das heißt, sie reduzieren den möglichen Wirkungsgrad der Arbeit. **Destruktive Schutzräume** entstehen, wenn methodisch dominierte Projekte durch Dynamik überfordert werden. Sie sind eine Form von Notwehr.

Theater-Kommunikation - Zwei Partner, die sich vertragen müssen (zum Beispiel: Projektleitung und externes Controlling), erleichtern sich dieses Problem, indem sie sich mit gegenseitiger Duldung Theater vorspielen. Der Projektleiter schönt die Zahlen, der Controller tut so, als wüsste er von nichts – eine enorme Verschwendung von Ressourcen.

Fleiß - Projekte, die unter Ideen-Mangel leiden, werden fleißig und versuchen sich so gegen Vorwürfe zu schützen. Fleiß als Mittel, eine Idee umzusetzen, ist positiv, Fleiß als Ersatz für Ideen nicht.

Status-Hektik - Hektik ist eine Form von Überforderung. Viele Unternehmens-Kulturen lassen es zu, dass hektische Geschäftigkeit als Status-Symbol gelten kann.

Messung - Messen setzt Trivialisierung voraus. Beschränkung auf Messbares ist Beschränkung auf Triviales. In dynamischem Kontext ist viel Wichtiges nicht trivial und darum nicht messbar. Es muss risikoreich beurteilt werden.

Demokratie - Demokratie ist eine leistungsfähige Organisationsform der Politik. Für den wirtschaftlichen Aspekt eines Unternehmens ist sie unbrauchbar. Sie delegiert Verantwortung an die Mehrheit und ist damit eine "interne Referenz". Höchstleistung benötigt externe Referenzen, also das Urteil des Kunden und der Konkurrenz.

Anreize durch Belohnung - Belohnung als Motivierung reduziert die mögliche Leistung auf Mittelmaß. Höchstleistung braucht maximale Motivation. Diese stellt sich nur dann ein, wenn Gelegenheit gegeben ist, individuelle Talente zu realisieren.

Fettsucht - Projekte haben die Neigung Zuliefer-Probleme durch "Selbermachen" zu lösen. Die Zuständigkeit eines Projektes wird dadurch so lange ausgeweitet, bis kein Überblick mehr gewonnen werden kann (Fettsucht). Die Zuständigkeit eines Projektes ist dann richtig bemessen, wenn nichts mehr weggelassen werden kann.

Schnittmengen - Kommunikation ist schwierig und anstrengend. Werden Schnittstellen durch Schnittmengen ersetzt, wird es leichter. Die Kommunikation kollabiert aber, denn sie setzt Trennung voraus. Ist Kommunikation hoher Qualität nötig, so sind Schnittmengen durch Schnittstellen zu ersetzen. Das heißt zum Beispiel: keine Repräsentanten der Fachbereiche im Projekt.

Ziel des Transmethodischen Projektmanagements ist, die destruktiven Schutzräume durch konstruktive zu ersetzen. **Konstruktive Schutzräume** entstehen aus der Einsicht, dass Projekt und Linie sich nicht belehren, sondern nur tolerieren können. Eine gemeinsam akzeptierte Grenze zwischen Lebensräumen erleichtert die Toleranz. Konstruktive Schutz-Räume sind die Voraussetzung für Dynamik robuste Leistung. Im benutzten Bild vom Feuer und Wasser ist ein konstruktiver Schutzraum ein Kessel, der mit Wasser gefüllt über ein Feuer gehängt wird. Der beste Schutzraum ist die externe Referenz, zum Beispiel der externe Kunde (**Bündnis mit dem Kunden**).

In dynamischer Umgebung versagt die gewohnte Steuerung als Mittel zur Umsetzung von Innovation. Es ist dann leichter, die Innovation als geschütztes Experiment zu gestalten. Das heißt, die gültigen Regeln werden - zeitlich und "örtlich" begrenzt - außer Kraft gesetzt. Dieser Schutzraum ist konstruktiv und wird als **Labor** bezeichnet. Ein Labor ist ein von der Außenwelt anerkannter Ausnahme-Raum. Labore sind zwar scharf und sichtbar abgegrenzt und "dicht", ihre "Wände" sind aber durchsichtig. Es kommt darauf an, dass die Experimente im Labor leicht beobachtet werden können. Das hat den Vorteil, dass im Falle von Erfolg der Wille zur Nachahmung "von alleine" entsteht. Der übliche Widerstand gegen das Neue bleibt aus. Die gewohnten Anstrengungen des Change-Managements sind überflüssig.

Weitere Schutzräume sind die **Konsenswerkstatt** und der **Mentor.** Sie werden nachfolgend als Elemente des Widerständigen Nests erläutert.

4. Struktur und Elemente des Widerständigen Nests

Im Folgenden ist der einfachste Fall eines widerständigen Nests als Struktur für komplexe Probleme dargestellt. Es ist selten möglich, alle Elemente zu realisieren. Meist sind mehr oder weniger große Kompromisse nötig. Wichtig ist nur zu wissen, dass dort, wo die Strukturen verletzt werden müssen, die Risikopunkte des Projektes zu erwarten sind. Verletzte Prinzipien sind Wegweiser für die Aufmerksamkeit der Projektleitung.

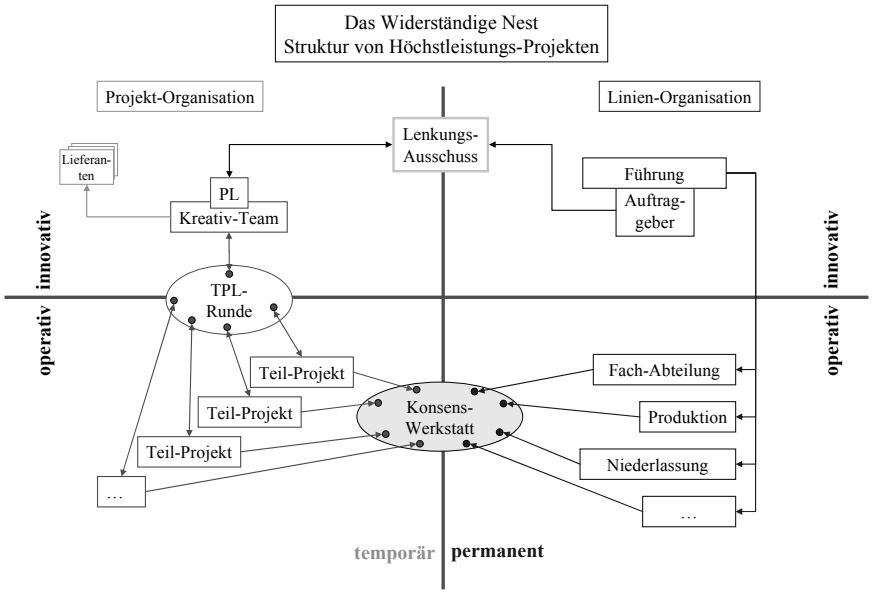


Abbildung 2: Die Elemente des widerständigen Nests ergeben sich aus zwei Unterscheidungen: temporär /permanent und innovativ /operativ

Auch wenn die Bezeichnungen für die Elemente des widerständigen Nests die gewohnten sind, so sind ihre Bedeutungen doch erheblich andere. Nur wenige kommen neu hinzu. Hier einige Beispiele:

Der **Auftraggeber** ist die Person, die das zu lösende Problem als Person hat. Im Kontext hoher Dynamik können Probleme auch durch Änderung der Aufgabe gelöst werden (Problem-Transformation[2]). Dafür braucht das Projekt einen Gesprächspartner, der darüber diskutieren und entscheiden kann. Der **Projektauftrag** ist als schriftliche Beschreibung des zu lösenden Problems, nicht der Lösung zu verstehen. **Teilprojekte**, die aus dem Projektauftrag entstehen**,** sind meist flüchtige projektinterne Strukturen. Ein Teilprojekt ist kein eigenes Projekt, es hat keinen externen Auftraggeber.

Der **Projektleiter** hat weder disziplinarische noch hierarchische Funktion. Er ist der so genannte Erkenntnisknoten des Projekts. In einem Hochleistungs-Projekt werden in vielen Zusammenhängen viele Ideen entwickelt. Es kommt darauf an, diese harmonisch zu halten, damit sie sich nicht destruktiv gegeneinander richten. Deshalb müssen sich alle Ideen, die für das Projekt wichtig sind, "im Kopf" des Projektleiters begegnen. Deswegen kann es nur einen Projektleiter geben. Der **Mentor** ist ein Manager auf höchstem Niveau. Er übernimmt die Aufgabe, unvorhersehbare Schwierigkeiten mit den Mitteln der Macht kreativ zu lösen. Kurz: Er stiftet einen konstruktiven Schutzraum. Andere Namen für diese Funktion sind: Sponsor, Machtpromotor, "Schutzengel".

[2] Ein in der Studie beschriebenes Höchstleister-Werkzeug

Das **Projektteam** hat die Aufgabe durch intensive Kommunikation nach innen und nach außen das Projekt mit Ideen und Irritation zu versorgen. Es besteht aus Personen, die sich der Projektleiter selber und ohne Begründung aussucht. Sie repräsentieren weder ein Fachgebiet noch äußere Interessen.

Mit den **Abnehmern** der Projektleistungen muss intensiv kommuniziert werden. Er darf deshalb keine Repräsentanten im Projektteam haben. Kommunikation ist nur zwischen Getrenntem möglich (Kommunikation durch Trennung).

Lieferanten sind Institutionen, die Ergebnisse liefern können, die das Projekt benötigt. Es gilt das Prinzip, dass alles, was zugeliefert werden kann, zugeliefert werden muss. Das gilt besonders für die umgebende Linie des eigenen Unternehmens. Projekte sind zu teuer, um Schwächen oder Unwillen der Linie auszugleichen.

Der **Lenkungsausschuss (Entscheidungsträger)** koppelt die Linien-Institution des Auftraggebers mit dem Projekt. Er ist aus Repräsentanten zusammengesetzt, die vom Verlauf und Ergebnis des Projektes betroffen sind. Vereinfacht: Alle, die dem Projekt schaden könnten, sollten Mitglied sein. Der Lenkungsausschuss ist ein passives Gremium. Er wird nicht selbst aktiv und schon gar nicht steuernd. Er tagt nur, wenn der Projektleiter Entscheidungen benötigt, die sich aus dem Projektauftrag nicht ableiten lassen.

Die **Konsenswerkstatt** ist ein Gremium aus Repräsentanten des Projekts und der Abnehmer der Projekt-Leistungen. Sie hat die Aufgabe, öffentlich bereits akzeptierte Projektergebnisse einer geheimen Abstimmung auszusetzen. Sie ist der feierliche Abschluss einer intensiven Kommunikation. Eine Gegenstimme genügt, um eine Vorlage abzulehnen. Es ist also explizit erlaubt, öffentlich zuzustimmen und geheim abzulehnen. Die Konsenswerkstatt ist ein empfindlicher Sensor für nicht kommunizierbare Konflikte – eine der Quellen von Havarien. Ein Projekt kann Konsens nur umsetzen, nicht aber erzeugen.

Ein Projekt braucht ein „privates“ Innenleben, eine **"soziale Haut"**. Das Risiko eine innovative Idee zu äußern ist sonst zu hoch. Kann sich ein Projekt diesen Status nicht erobern, entsteht keine Höchstleistung. Konventionelles **Controlling** vermutet in diesem Anspruch den Versuch, sich jeglicher Zielsetzung und Kontrolle zu entziehen. Das Gegenteil ist der Fall. Wird in ein Projekt hineinregiert, wird ihm auch ein Teil der Verantwortung abgenommen. Das Innere eines Projektes „controllen“ zu wollen, hat genauso wenig Sinn wie mit einem Moped durch den Wald zu knattern, um Rehe zu zählen. Die Festlegung und Überwachung von Ressourcen ist dagegen keine Einschränkung der **Projekt-Autonomie**. Ein festgesetzter Rahmen steigert die Handlungsfreiheit eines Projekts. Er ist deshalb eine wichtige Voraussetzung für Autonomie und damit für den Projekterfolg.

Typische Schnittstellen im Unternehmen sind asymmetrisch: eine Seite liefert eine Leistung, die andere liefert nichts Entsprechendes. Diese Schnittstellen benötigen eine externe Steuerung. Bei hoher Dynamik versagt jede Steuerung. Ein dynamisches Projekt benötigt deshalb **symmetrische Schnittstellen** zu seiner Umgebung. Hier wird in beiden Richtungen eine Leistung geliefert. Es wird getauscht. Solche Schnittstellen stabilisieren sich selbst. Sie benötigen keine Steuerung. Höchstleistungs-Projekte benötigen diese Schnittstellen vor allem zu ihren Lieferanten.

5. Zum Schluss

Häufiger als früher haben Unternehmen Probleme, die nur gelöst werden können, wenn Teams von Menschen an die Grenze ihrer Leistung gehen. In der üblichen Umgebung eines Unternehmens sind solche Höchstleistungsprojekte unerträgliche Fremdkörper. Sie können sich nur entfalten, wenn es gelingt, den Immunapparat der Unternehmensorganisation zu neutralisieren (Nest), ohne die Härte des Problems zu dämpfen (Widerständigkeit). Kurz - das "Widerständige Nest" ist die unternehmensinterne Umgebung für Höchstleistung dynamischer Projekte.

Neue Chancen im eskalierten Konflikt

Hans Brandt-Pook

Fachbereich Produktion und Wirtschaft
FH Lippe und Höxter

Liebigstr. 87
D-32657 Lemgo
Hans.Brandt-Pook@FH-LuH.de

Axel R. Raulinat

TENOS Aktiengesellschaft
Konfliktberatung Konfliktlösung
Konfliktprävention
Neuer Wall 35
D-20354 Hamburg
Axel.Raulinat@Tenos.de

Abstract: Bei eskalierten Konflikten im Wirtschaftsleben gibt es mehrere Lösungsstrategien. Häufig wird der Weg der gerichtlichen Auseinandersetzung gewählt. Der vorliegende Artikel stellt als Alternative dazu den erfolgreichen Ansatz der WirtschaftsMediation vor und zeigt seine Anwendbarkeit im Softwaremanagement. In der WirtschaftsMediation verhandeln die Konfliktparteien nach einem strukturierten Phasenmodell unter Mitwirkung eines neutralen Dritten.

1 Einleitung

Ein Konflikt ist nichts Ungewöhnliches - Konflikte gibt es immer und überall. Es gibt offene oder verdeckte Konflikte. Manche eskalieren schnell – andere schwelen jahrelang bevor sie explodieren. Nach der Konflikttheorie von Schwarz [Sc99] gibt es sechs Stufen zur Bewältigung von Konflikten, von der Flucht einer Konfliktpartei (also das Umgehen des Konfliktes) bis hin zum Konsens als der Lösung, die den Interessen beider Parteien voll und ganz entspricht. In der Praxis des wirtschaftlichen Lebens kommen tatsächlich alle Stufen der Konflikttheorie vor – wenn auch nicht immer offen sichtbar.

Auch im Softwaremanagement sind Konflikte an der Tagesordnung. In diesem praxisorientierten Übersichtsbeitrag wollen wir eine moderne Lösungsstrategie für eskalierte Konflikte aufzeigen. Wir stellen mit der WirtschaftsMediation ein Verfahren zur Konfliktlösung vor, das wegen seiner hohen Erfolgsquote zunehmend an Bedeutung gewinnt.

2 Konflikte im Softwaremanagement

Konflikte im Softwaremanagement können zunächst in interne und externe Konflikte strukturiert werden (s. Abbildung 1). Unter internen Konflikten verstehen wir solche Konflikte, die innerhalb des Unternehmens aufbrechen, während bei externen Konflikten Parteien außerhalb des IT-Dienstleisters involviert sind.

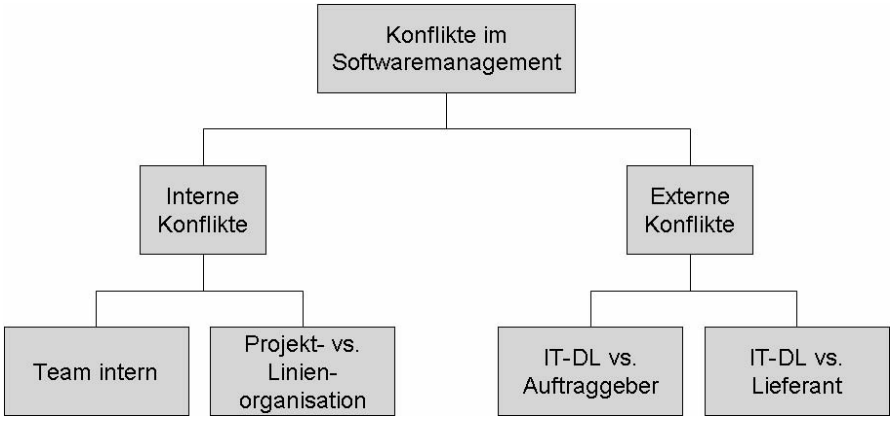


Abbildung 1: Konfliktlinien im Softwaremanagement

Die internen Konflikte können weiter strukturiert werden: Konflikte innerhalb eines IT-Teams unterscheiden sich dem Wesen nach wenig von Konflikten, die auch in anderen Arbeitsgruppen auftreten. Zu den teaminternen Konflikten werden auch Konflikte zwischen Teilteams innerhalb eines Projektes gezählt. In Branchen, die wie die IT-Branche sehr projektorientiert arbeiten, sind Konflikte häufig in der Struktur der Projekt- und Linienorganisation begründet: die unterschiedlichen Aufgaben und Interessen in der Linie und im Projekt beinhalten die Gefahr von Rollenkonflikten, die dann in soziale Konflikte münden können (vgl. z.B. [Fi04], S. 82ff).

An externen Konflikten sind typischerweise entweder die Auftraggeber oder Lieferanten bzw. Subunternehmer des IT-Dienstleisters (IT-DL) beteiligt.

Ein Konflikt hat eine Eigendynamik und durchlebt nach Glasl [Gl02] bis zu neun Konfliktstufen (s. Abbildung 2). Er beginnt mit einer Verhärtung der Positionen – „es knirscht“ zwischen den Beteiligten. In der Debatte ist die Argumentation nur noch vordergründig sachlich. Die dritte Stufe ist gekennzeichnet durch die Strategie der vollendeten Tatsachen. Die nächste Stufe ist durch das Bilden von Koalitionen charakterisiert („wir“ und „die anderen“). In der fünften Stufe werden öffentlich direkte persönliche Angriffe mit dem Ziel des Gesichtsverlusts des Konfliktgegners formuliert. In der nachfolgenden sechsten Stufe gibt es Drohungen und Gegendrohungen, die in Stufe sieben zu einer Strategie der Vernichtungsschläge eskalieren, die als eigene Verteidigung gerechtfertigt werden. Die achte Stufe hat die „Ausschaltung des feindlichen Systems“ zum Ziel. In der neunten und letzten Stufe schließlich gibt es keinen Weg mehr zurück und der gemeinsame Gang in den Abgrund beendet den Konflikt.

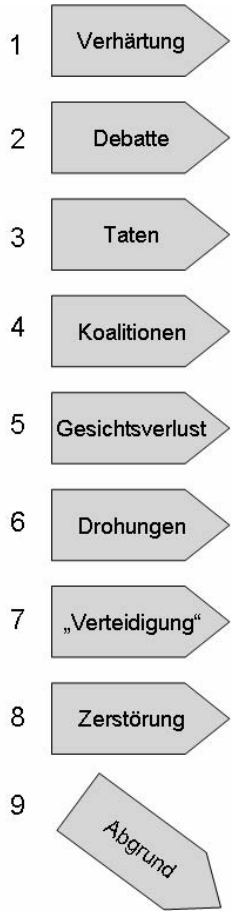


Abbildung 2: Konfliktstufen nach Glasl

Für Konflikte in frühen Phasen (Konfliktstufe eins und zwei) hat das Softwaremanagement sehr brauchbare und bewährte Ansätze und Instrumente entwickelt, die häufig auch mit dem Ziel der Qualitätsverbesserung ohne den Anlass eines Konfliktes eingeführt werden. Bspw. kann durch die Einführung eines wohl definierten Change Request Prozesses der altbekannte Konflikt zwischen IT-Dienstleister und Auftraggeber bezüglich der Auftragsabwicklung entschärft werden. Ebenso gehören Teamentwicklungsmaßnahmen zum Repertoire der Vorbeugung interner Konflikte.

Diese Ansätze greifen aber dann nicht, wenn der Konflikt bereits eskaliert ist und sich bereits in den Stufen fünf oder höher bewegt. Eine Lösung des Konfliktes ist dann nicht ohne weiteres durch die Einführung einer Maßnahme möglich, weil das Gespräch über solche Maßnahmen gar nicht mehr möglich ist. Die Fronten sind so verhärtet, dass eher eine gerichtliche Auseinandersetzung angedacht oder sogar schon angestrebt wird. Für diese Situationen hat sich mit der WirtschaftsMediation in den vergangenen Jahren ein Verfahren etabliert (vgl. z.B. [Ha00], [Di05]), das im Folgenden vorgestellt werden soll.

3 WirtschaftsMediation

Der grundlegende Gedanke der WirtschaftsMediation im Kontext der möglichen Konfliktlösungen besteht darin, die Verhandlung zwischen den Beteiligten als Strategie der Konfliktbeilegung zu wählen. Dass die Parteien den Ansatz der Verhandlung wählen, ist ein freiwilliger Entschluss. Sie haben sich dafür entschieden, die Konfliktbeilegung selbst in die Hand zu nehmen. Eine andere – heute in Deutschland noch weitaus häufigere – Variante besteht darin, ein Gericht einzuschalten, d.h. den Klageweg zu wählen. Nach der Definition von [Sc99] ist das nichts anderes als die Delegation des Konfliktes an eine höhere Instanz.

Im Folgenden wird dieser Verhandlungsansatz der WirtschaftsMediation aus verschiedenen Perspektiven erläutert.

3.1 Herkunft und Status der WirtschaftsMediation

Der Ansatz der Mediation wurde in den 1970er und 1980er Jahren in den USA entwickelt und wird dort sehr erfolgreich praktiziert. Er ist gedanklich eng mit der Verhandlungsstrategie *Havard Konzept* [FUP04] verbunden. In diesem Konzept stehen wie bei der WirtschaftsMediation die Interessen der Verhandlungsparteien und nicht ihre vordergründigen Positionen im Mittelpunkt. Die WirtschaftsMediation kann daher – vereinfachend – als Anwendung des Havard Konzepts unter Beteiligung eines neutralen Dritten verstanden werden.

Seit etwa zehn Jahren gewinnt die WirtschaftsMediation auch in Deutschland immer mehr an Bedeutung [Ha00] – bundesweit laufen an vielen Gerichten Projekte, in denen Richter auch als Mediatoren tätig werden [Vo03].

Neben der *Gesellschaft für Wirtschaftsmediation und Konfliktmanagement*, einem Verein mit Sitz in München[3], sind auf dem Gebiet der WirtschaftsMediation insbesondere die *Handelskammer Hamburg*[4] und die *TENOS AG*[5] aktiv. Sie halten neben notwendigen Verfahrensordnungen auch qualifizierte Mediatorinnen und Mediatoren vor.

In der deutschen Hochschullandschaft ist das Thema noch spärlich vertreten und in der Regel in die juristischen Fachbereiche integriert. An der Europa-Universität Viadrina Frankfurt (Oder) gibt es einen Master-Studiengang „Mediation“, der auch eine Spezialisierung in den Bereich der WirtschaftsMediation vorsieht[6].

3.2 Die Besonderheiten der WirtschaftsMediation

Das Besondere an den Verhandlungen im Rahmen einer WirtschaftsMediation ist zunächst die Hinzuziehung eines neutralen Dritten – des Mediators bzw. der Mediatorin. Sie sind speziell ausgebildet im Verhandlungs- und Konfliktmanagement sowie in Kreativitätstechniken. Der Konflikt ist nach den Glasl’schen Stufen so weit entwickelt, dass eine Entflechtung ohne die Unterstützung durch eine außen stehende Person nicht mehr möglich ist.

Eine weitere Besonderheit besteht darin, dass die moderierte Verhandlung nach einem strukturierten Verfahren abläuft, das in vier Phasen (die „Mediationsuhr“) eingeteilt werden kann (s. Abbildung 3)[7].

---

[3] http://www.gwmk.de
[4] http://www.hk24.de
[5] http://www.tenos.de
[6] http://master-mediation.euv-ffo.de
[7] Der Ansatz ist noch zu jung, als dass sich eine gemeinsame Sichtweise über die Anzahl der Phasen gebildet haben könnte. [Di05] benennt bspw. sieben Phasen, [Ha00] fünf – die beide aber im Wesentlichen jeweils dasselbe abdecken, was hier dargestellt wird.

In der ersten Phase geht es um eine Themensammlung. Oft ist ein Konflikt so verworren, dass zunächst einmal die Themen sortiert werden müssen. Die Leitfrage ist: „Um was geht es überhaupt?“ Nicht selten bringen die Konfliktparteien auch unterschiedliche Themen mit – bedingt durch die unterschiedlichen Sichtweisen auf den Konflikt.

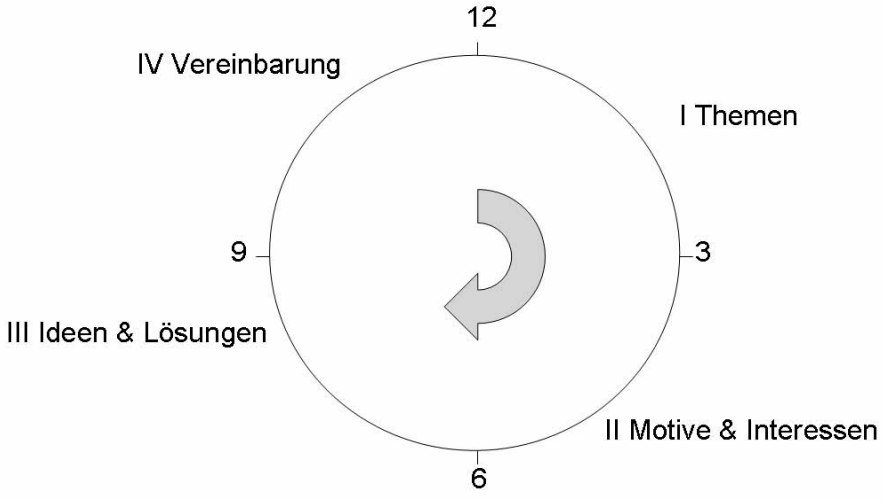


Abbildung 3: Die „Mediationsuhr“

Jedes identifizierte Thema wird in der zweiten Phase danach untersucht, was die Hintergründe der jeweiligen Forderungen sind. „Welche Interessen haben die Konfliktparteien wirklich?“ lautet die in diesem Abschnitt bearbeitete Frage. Nach Auffassung der WirtschaftsMediation ist diese Phase die bedeutendste (und zeitlich längste) Phase. Denn sie legt die Basis für ein gegenseitiges Verständnis (nicht: Einverständnis!) der Konfliktparteien. Nur nach Klärung der jeweiligen Interessen ist der Weg für eine nachhaltige Lösung des Konflikts geebnet.

Die dritte Phase beginnt mit einer möglichst kreativen Ideensammlung – alle Vorschläge zur Lösung des Konflikts sind gestattet und erwünscht. Eine Bewertung der Ideen und die Kombination der besten Ideen zu einer Lösung erfolgt erst in einem zweiten Schritt.

Schließlich wird die favorisierte Lösung in eine konkrete schriftliche Vereinbarung gegossen. Wenn Rechtsanwälte die Parteien bei der Verhandlung beraten (was in der Praxis regelmäßig vorkommt und ratsam ist), wird an Ort und Stelle ein rechtsgültiger Vertrag aufgesetzt.

3.3 Vorteilhafte Kennzeichen und Ausschlusskriterien des Ansatzes

Die WirtschaftsMediation als Form der Konfliktbeilegung bietet handfeste Vorteile:

- Die Verhandlungen sind schnell
  Gerichtsverhandlungen sind häufig langwierig – insbesondere, wenn der Konflikt sich lange entwickelt und damit verkompliziert hat. Die schnelle Erledigung eines Konflikts ist aber häufig ein Wettbewerbsvorteil. Eine Verhandlung der Konfliktparteien nach den Regeln der WirtschaftsMediation kommt typischerweise bereits nach ein bis drei Tagen zu einem für beide Seiten befriedigenden Ergebnis.

- Die Verhandlungen sind selbstbestimmt
  Wählt man im Konflikt den Gang vor das Gericht als Strategie, so gibt man damit nicht nur die Entscheidung über den Konflikt, sondern auch die Gestaltung des Verfahrens ab. Die Konfliktparteien haben nur noch bedingt Einfluss auf die Inhalte dessen, was verhandelt wird. In der WirtschaftsMediation entscheiden die Parteien selbst über Inhalte sowie Art und Weise der Verhandlungen.

- Die Verhandlungen sind vertraulich
  Zu den Grundsätzen der WirtschaftsMediation gehört, dass nichts von den Verhandlungen nach außen dringt, was nicht ausdrücklich von den Konfliktparteien freigegeben wurde. Häufig wird sogar Stillschweigen darüber vereinbart, dass überhaupt Verhandlungen unter Beteiligung eines Mediators stattgefunden haben.

- Die Verhandlungen sind erfolgreich
  Streiten sich Konfliktparteien vor Gericht, gibt es in der Regel nur einen Gewinner und einen Verlierer. WirtschaftsMediation dagegen sucht Win-Win-Lösungen. In [Vo03] wird eine Erfolgsquote, d.h. ein von beiden akzeptiertes Verhandlungsergebnis, von 75 % für 600 analysierte Mediationen durch Richter genannt.

- Die Verhandlungen sind lösungsorientiert
  Bei einer WirtschaftsMediation steht nicht die rückwärtsgerichtete Problemanalyse im Mittelpunkt. Vielmehr wird fokussiert auf eine Lösung in der Zukunft zu beiderseitigem Vorteil.

- Die Verhandlungen sind risikoarm
  Eine WirtschaftsMediation ist – insbesondere im Vergleich zum Klageweg – risikoarm, denn das Verfahren verbaut keine anderen Konfliktlösungsstrategien. Im schlimmsten Fall können die Verhandlungen scheitern, was aber unschädlich für weitere Schritte ist.

- Die Verhandlungen sind kostengünstig
  Die Kosten für eine WirtschaftsMediation hängen *nicht* vom Streitwert ab. Es sind lediglich die Kosten für den Mediator bzw. die Mediatorin zu zahlen, so dass sich ein deutlicher Kostenvorteil gegenüber Gerichts- und Schiedsgerichtsverfahren ergibt[8].

[8] Unter http://www.tenos.de/kostenrechner/ findet sich ein Tool zum direkten Vergleich der Kosten.

Der Ansatz der WirtschaftsMediation eignet sich vor allem dann, wenn ein konkreter Streitfall verhandelt werden kann. Daraus ergibt sich auch ein Ausschlusskriterium: Streitigkeiten von grundsätzlicher Natur, die nicht von den Parteien ergebnisorientiert verhandelbar sind (z.B. zu Auslegungen des Bundesdatenschutzgesetzes), eignen sich für eine WirtschaftsMediation weniger. Eine WirtschaftsMediation ist auch nicht angezeigt, wenn kein prinzipielles Kräftegleichgewicht zwischen den Konfliktparteien besteht.

3.4 Zwei Beispiele

Zur ersten Veranschaulichung des Ansatzes werden nun zwei Beispiele dargestellt.

Umstrukturierung im Betrieb

Nach einer Umstrukturierung in einer Möbelbaufirma war der langjährige Mitarbeiter Heinz F. versetzt worden. Er ist mit seiner neuen Aufgabe unzufrieden. Der Geschäftsführer Hubert S. ärgert sich über Mängel in der Arbeit von Heinz F., der plötzlich auch häufiger erkrankt – was 20 Jahre lang nicht vorgekommen war. Nach einer Abmahnung schaltet Heinz F. seinen Rechtsanwalt ein, der meint, dass die Versetzung von Heinz F. nicht ordnungsgemäß war. Es droht zu einer Auseinandersetzung vor dem Arbeitsgericht zu kommen.

In einem Gespräch unter Beteiligung eines WirtschaftsMediators wird eine Lösung gefunden, welche die Unzufriedenheit von Heinz F. beseitigt und seine langjährige Betriebszugehörigkeit würdigt, ohne dass die unternehmerische Entscheidungskompetenz von Hubert S. in Frage gestellt wird.

Unternehmensnachfolge

Vierzig Jahre lang hatte Gerhard B. sein Bauunternehmen aus dem Nichts aufgebaut. Sein Sohn Martin und seine Tochter Renate sind schon vor einigen Jahren in die Geschäftsführung eingestiegen. Der Senior hat sich offiziell vor einem Jahr zurückgezogen. Nun ist es zum Streit über die Zusammenarbeit gekommen: Martin wirft seinem Vater vor, noch immer alles Wichtige entscheiden zu wollen und Renate glaubt, dass Martin hinter ihrem Rücken Geschäfte betreibt, die nicht der Absprache entsprechen. Sie will einen Notar mit der Prüfung der Angelegenheit beauftragen.

In dem Mediationsverfahren stellt die WirtschaftsMediatorin sicher, dass alle drei ihre Ansprüche und Interessen formulieren können. Nach einer Ideenphase (Brainstorming) wird schließlich eine Vereinbarung geschlossen, welche die Aufgaben und Kompetenzen regelt und somit die Unternehmensnachfolge auf sichere Wege bringt.

3.5 WirtschaftsMediation im Softwaremanagement

Zunächst bietet die WirtschaftsMediation für die Bearbeitung der Konflikte im Softwaremanagement dieselben Vorzüge wie in allen anderen Branchen. Wir sehen zusätzlich einige für die IT-Branche typische Merkmale bzw. Konstellationen, die es besonders lohnenswert machen können, über den Einsatz der WirtschaftsMediation nachzudenken.

Externe Konflikte (also Konflikte mit anderen Unternehmen) können dann nicht mit allen Konfliktlösungsstrategien ausgetragen werden, wenn beide Unternehmen zu einem Firmenverbund gehören. Ein Konzernvorstand kann kein Interesse haben, dass seine IT-Tochter mit einem Schwesterunternehmen eine gerichtliche Auseinandersetzung führt. In diesem Fall bietet die WirtschaftsMediation die Möglichkeit, dennoch ein strukturiertes und geregeltes, gleichzeitig aber offenes und flexibles Verfahren zur Konfliktlösung anzugehen. In den moderierten Verhandlungen können die Interessen der Konzerntöchter abgewogen werden, ohne dass das Ansehen des Konzerns insgesamt in der Öffentlichkeit Schaden leidet.

Viele IT-Vorhaben werden projektorientiert gestaltet. Ein IT-Dienstleister, der schnell und lösungsorientiert IT-Projekte abwickelt, braucht im Konfliktfall eine zeitnahe Lösung. Langwierige und zähe (Gerichts-)Verhandlungen blockieren neue Projekte – zumindest die Akquisition neuer Projekte beim Konfliktgegner.

Die Arbeit in der IT erfolgt in der Regel in Teams. Die WirtschaftsMediation verfügt über einen Baukasten von Methoden, wie eskalierte Team-Konflikte bearbeitet werden können. Diese Methoden können auch dann zum Einsatz kommen, wenn es Konflikte zwischen zwei IT-Teams oder zwischen der Linienorganisation und dem Projektmanagement gibt.

4 Fallbeispiel

In diesem Abschnitt werden die Grundzüge der WirtschaftsMediation an einem ausführlichen Beispiel erläutert. Es wird insbesondere auf den Prozess sowie die in den einzelnen Phasen erzielten Ergebnisse eingegangen.

4.1 Ausgangssituation

Die Versandhandelskette *Alpha AG* hat ein Projekt aufgesetzt, um das bestehende host-basierte Warenwirtschaftssystem abzulösen. Das Projekt ist mehrstufig für den Zeitraum von vier Jahren geplant und hat vor gut zwei Jahren begonnen. Für das neue Warenwirtschaftssystem (WaWiS) wurde bei der Konzerntochter *Alpha Systems* ein neues Projektteam von 25 Personen gebildet, welches WaWiS gestalten und entwickeln soll sowie in Zukunft die Verantwortung für den Betrieb des Systems übernehmen wird. Das neue Team setzt sich zusammen aus Experten, die das bisherige System technisch betreuen, aus jungen Softwareentwicklern mit etwas Erfahrung in der objektorientierten Programmierung sowie aus den Fachkollegen, welche die fachlichen Anforderungen an WaWiS aus den verschiedenen Bereichen der *Alpha AG* koordinieren sollen. Die Ziele des Projekts sind also eine Herausforderung: mit einem neu gebildetem Team soll ein fachlich wesentlich erweitertes System basierend auf einer für *Alpha Systems* neuen Technologie zukunftssicher erstellt werden.

Der Bereichsleiter von *Alpha Systems*, Herr Wilfried S., wollte die engagierte Zielsetzung des Projekts absichern. Dazu wandte er sich an seinen alten Weggefährten, Florian W., der vor einiger Zeit eine eigene IT-Beratungsfirma, die *Beta GmbH*, gegründet hatte. So wurden dem Team von Beginn zwei erfahrene Berater zur Seite gestellt, um das Projekt „auf Spur zu halten“. Ihre Aufgaben waren neben dem Coaching des Projektleiters in Fragen der Projektgestaltung die Moderation von Workshops, incl. deren Protokollierung als Grundlage für die Systemspezifikation. Außerdem wurde vereinbart, dass sie sich um die Qualitätssicherung kümmern sollten.

Die erste Phase des Projekts lief sehr gut an. Das erste Modul von WaWiS, die Artikelstammdaten-Pflege, konnte nach zehn Monaten fristgerecht in den Probebetrieb überführt werden. Schon in der zweiten Phase begann es im Projekt aufgrund von Zeitverzug zu kriseln. Die Probleme verschärften sich derart, dass kurz vor Ende der dritten Phase nach 26 Monaten Laufzeit das Projekt vom Bereichsleiter Herrn S. vorerst auf Eis gelegt wird. Die beiden Berater geraten ins Kreuzfeuer der Kritik, denn ein Gutachten weist aus, dass die Architektur von WaWiS grobe Mängel hat und insbesondere für Erweiterungen in den kommenden Projektphasen nur wenig tauglich sein wird.

Herr S. ist nach dem Studium des Gutachtens überaus sauer und möchte am Liebsten das Honorar für die insgesamt 540 Beratertage zurück haben: aus seiner Sicht waren die Berater ja zuständig dafür, dass das Projekt ein Erfolg wird. Der Geschäftsführer von *Beta*, Herr W., sieht das verständlicher Weise anders. Die Fronten verhärten sich drastisch. Um eine gerichtliche Auseinandersetzung zu vermeiden, einigt man sich auf eine WirtschaftsMediation auf der Basis der Mediationsordnung der örtlichen Industrie- und Handelskammer. Als Mediator wird Herr Gerd Schaller engagiert. Zunächst werden zwei Termine à vier Stunden vereinbart – die Kosten dafür werden je zur Hälfte übernommen. Die Verhandlungen finden in den Räumen des Mediators statt.

4.2 Ablauf und Ergebnisse der Phasen in der WirtschaftsMediation

Nachdem der WirtschaftsMediator Herr Schaller noch einmal kurz das Verfahren vorgestellt hat und einige Rahmenbedingungen besprochen worden sind, bekommen beide Parteien zu Beginn die Möglichkeit, den Konflikt ohne Unterbrechung aus der jeweils eigenen Sicht in zehn Minuten zu schildern. Die Stimmung ist anfangs frostig. Es gibt auch vereinzelt gereizte Zwischenrufe bei den jeweiligen Ausführungen, so dass Herr Schaller einige Male eingreift. Herrn S. geht es um seine Forderung, das Honorar erstattet zu bekommen sowie um die weitere Zusammenarbeit. Er fühlt sich von Herrn W. im Stich gelassen, denn er hatte seinem alten Freund voll vertraut und seinem Versprechen geglaubt, dass er die Aufgabe mit seinen Leuten hinbekäme. Herr W. dagegen führt aus, dass der Beratungsauftrag in einigen Details zu klären sei, auch bezüglich der Verantwortlichkeiten. Er ist wütend, weil er jetzt alle Fehler im Projekt ausbaden soll, denn aus seiner Sicht war das Hauptproblem der chaotische technische Projektleiter von *Alpha Systems*. Nach einigen Verständnisfragen zu den Einleitungsstatements hat der Mediator schließlich die jeweiligen Themen auf dem FlipChart notiert (s. Abbildung 4). Es ist zu sehen, dass die Kontrahenten durchaus unterschiedliche Dinge besprechen möchten – wenn sich auch der jeweils zweite Punkt nach dem ersten Eindruck nur in der Formulierung unterscheidet. Die Themen bleiben gut sichtbar aufgehängt und bilden für beide Parteien die Leitlinie für die Verhandlungen. Sie können auch später ggf. noch erweitert werden.

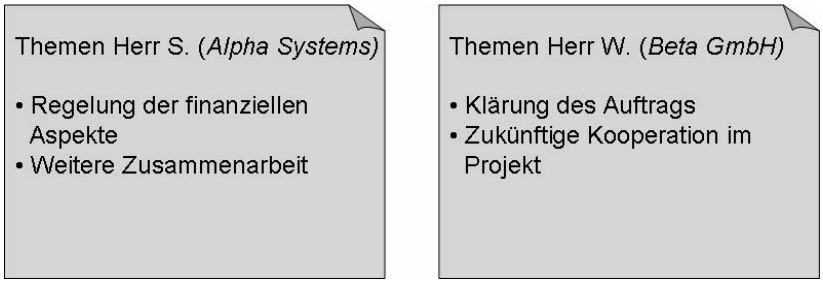


Abbildung 4: Themen der Konfliktparteien

In der zweiten Phase erläutern die Parteien ihre Motive und Interessen zu ihren jeweiligen Forderungen. Herrn S. ist es wichtig, dass *Alpha Systems* – und insbesondere sein eigener Bereich – für zukünftige Anforderungen mit einer modernen Technologie gerüstet ist. Darum liegt ihm der Erfolg dieses Projektes als Pilotprojekt so sehr am Herzen. Er betont die Notwendigkeit, dass *Alpha Systems* auch konzernintern als moderner Dienstleister gelten muss, sonst besteht die Gefahr, dass sich die Konzerntöchter an externe IT-Firmen wenden. Herr W. will unter allen Umständen den Ruf seines noch jungen Beratungsunternehmens in der Branche wahren, weil das der Schlüssel zu neuen Aufträgen ist. Außerdem wäre eine sofortige und vollständige Rückzahlung des Beraterhonorars eine existenzielle Gefährdung für das Unternehmen und für ihn persönlich.

Die Erläuterungen und Hintergrundinformationen werden anfangs nur zögerlich gegeben. Der Mediator fragt daher einfühlsam nach und hilft so, Missverständnisse aufzuklären. Er erinnert auch an die in der Mediationsordnung enthaltene Regelung, dass nichts von dem Verhandelten ohne die ausdrückliche Zustimmung der Parteien nach außen dringt. So entstehen schließlich die beiden Flipcharts, die in Abbildung 5 wiedergegeben sind.

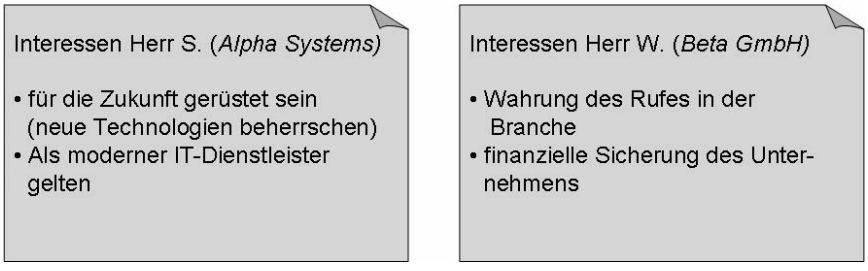


Abbildung 5: Interessen der Konfliktparteien

Mit dem Herausarbeiten der jeweiligen Interessen sind den Konfliktparteien die Motive deutlicher geworden. Das Ende des ersten Sitzungstermins ist erreicht und Herr Schaller bittet die Konfliktparteien, sich auf der Basis der bisher erzielten Ergebnisse erste Gedanken zu möglichen Lösungsideen zu machen.

Zu Beginn des zweiten Sitzungstermins wird der bisherige Verlauf der Verhandlungen rekapituliert. Herr Schaller fragt, ob sich neue Aspekte ergeben haben. Nachdem das verneint wird, beginnt die Phase *Ideen und Lösungen*. Dazu werden zwei Regeln vereinbart: *Jede Idee ist erlaubt* und *Bewertet wird später*. So ergeben sich ganz unterschiedliche mögliche Bausteine für eine Lösung, die auf Karten festgehalten werden (s. Abbildung 6).

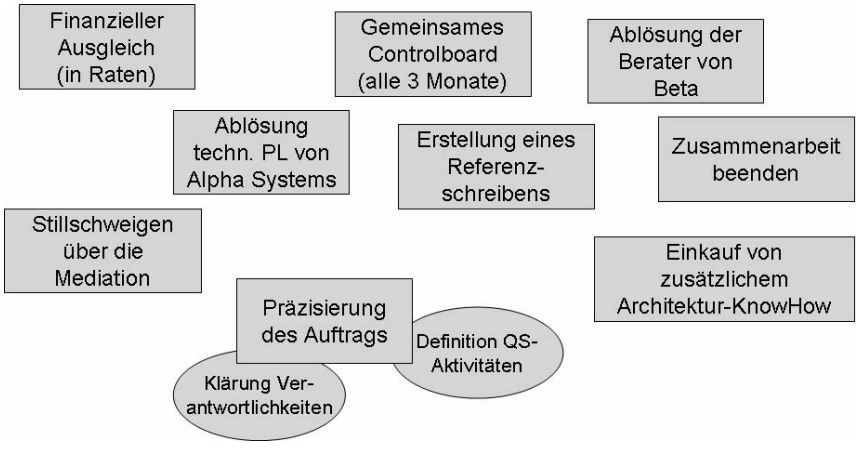


Abbildung 6: Lösungsansätze

Einige Ideen werden während der Bewertung der Ansätze wieder verworfen, bspw. die Beendigung der Zusammenarbeit sowie die Ablösung des technischen Projektleiters. Als zentraler Verhandlungspunkt erweist sich die Präzisierung des Auftrags. Es erfolgt eine Übereinkunft über die Verantwortlichkeiten sowie eine präzise Definition der QS-Aktivitäten. In diesem Zusammenhang wird besprochen, zusätzliches Architektur-KnowHow einzukaufen, damit die Berater von *Beta* sich mehr auf den Management-Prozess konzentrieren können. Eine anteilige Erstattung der Beraterhonorare wird ebenfalls vereinbart – allerdings in Raten, um die Existenz von *Beta* nicht zu gefährden. Ein Referenzschreiben von *Alpha Systems* soll die Leistungen der *Beta*-Berater im Projekt würdigen. Schließlich werden ein gemeinsames Controlboard sowie das Stillschweigen über den Verlauf der Mediation beschlossen.

Die Übereinkünfte werden in der letzten Arbeitsphase schriftlich fixiert. Bei der detaillierten Formulierung an Ort und Stelle geraten noch einmal einige Aspekte in die Diskussion, die eigentlich schon als geklärt gegolten haben.

4.3 Fazit

Die Konfliktparteien haben in achtstündiger Verhandlung unter Beteiligung des WirtschaftsMediators eine schriftliche Vereinbarung erarbeitet, die beiden Parteien in der Zukunft eine konstruktive und zielführende Zusammenarbeit erlaubt.

Die beschlossenen Maßnahmen in diesem Beispiel sind weder spektakulär noch grundsätzlich neu – aber sie waren bisher im Projekt eben noch nicht vereinbart und wären im eskalierten Konflikt ohne Einschaltung eines WirtschaftsMediators auch nicht zu vereinbaren gewesen. Erst das durch ihn geförderte gegenseitige Verständnis der Hintergründe und Interessen hat eine Lösungssuche und letztlich eine Lösung möglich gemacht.

§ 17 Außergerichtliche Streitbeilegung
Sollte es zwischen den Parteien bei der Durchführung dieses Vertrages zu Meinungsverschiedenheiten kommen, so führen die Parteien zur Beilegung dieser Meinungsverschiedenheiten zunächst ein Mediationsverfahren auf der Grundlage der Mediationsordnung der Tenos AG, Hamburg, durch. Für die Dauer des Mediationsverfahrens verzichten die Parteien auf die Einleitung gerichtlicher Schritte. Ausgenommen hiervon sind Maßnahmen des einstweiligen Rechtsschutzes.

Abbildung 7: Beispiel für eine Mediationsklausel

Anders als in dem geschilderten Beispiel ist es regelmäßig schwer, die zerstrittenen Parteien an den Verhandlungstisch zu bekommen. Diese Schwierigkeit ist zu vermeiden, indem vorsorglich in die entsprechenden Verträge Mediationsklauseln aufgenommen werden (s. Abbildung 7).

5 Schluss

Der Ansatz der WirtschaftsMediation bietet die Chance, im eskalierten Konflikt Lösungen zu finden. Die Parteien verhandeln ihre zukünftige Zusammenarbeit unter Vermittlung des WirtschaftsMediators. Dabei spielen die hinter den Forderungen stehenden Interessen eine Schlüsselrolle, deren gegenseitiges Verständnis die Basis für eine gemeinsame Lösungssuche bildet.

Die Erfolgsquoten der WirtschaftsMediation in anderen Ländern und Fachgebieten lassen hoffen, dass dieser vernünftige Ansatz auch im Softwaremanagement mehr Verbreitung findet.

Literaturverzeichnis

[Di05] Diez, H: Werkstattbuch Mediation. Centrale für Mediation, Köln, 2005.
[Fi04] Fischer-Epe, M.: Coaching: Miteinander reden. Rowohlt, Reinbek bei Hamburg, 2004.
[FUP04] Fisher, R. ; Uri, W; Patton, B.: Das Havard-Konzept. Campus, Frankfurt, 2004.
[Gl02] Glasl, F.: Konfliktmanagement. Verlag Freies Geistesleben, Stuttgart, 2002.
[Ha00] Haft, F.: Verhandlung und Mediation. C.H. Beck, München, 2000.
[Sc99] Schwarz, G.: Konfliktmanagement. Konflikte erkennen, analysieren, lösen. Gabler, Wiesbaden, 1999.
[Vo03] Von Olenhusen, G.: Mediation durch Richter – ein Projekt mit Zukunft. In: Deutsche Richterzeitung 2003, S. 396 – 397.

Rahmenbedingungen und Geschäftsmodell der IT-Beratung

Thomas Deelmann

Information Systems & Management
Lehrstuhl Wirtschaftsinformatik und BWL
Johannes Gutenberg-Universität Mainz
deelmann@isym.bwl.uni-mainz.de

Abstract: IT-Beratung vereint einen wesentlichen Teil der Beratungsausgaben von Unternehmen. Die wissenschaftliche Aufarbeitung dieser Beratungsdisziplin bleibt jedoch hinter der praktischen Relevanz zurück. Diese Lücke soll durch den Beitrag in Teilen geschlossen werden. Hierzu werden Rahmenbedingungen, unter denen die IT-Beratung agiert, aufgezeigt, wie auch das generische Geschäftsmodell einer IT-Beratung dargelegt und ausgewählte Details skizziert. Die Ausführungen verbinden Innen- und Außensicht der IT-Beratung und können in ihrer beschreibenden Funktion als Bindeglied zwischen den verschiedenen beteiligten Akteuren dienen.

1 Motivation, Ziel und Aufbau

IT-Beratung stellt, zusammen mit ihren Teildisziplinen der Softwareberatung, Systemintegrationsberatung, Outsourcing etc., einen wesentlichen Teil der Beratungsaktivitäten von Unternehmen dar und vereint einen großen Teil der Beratungsausgaben auf sich. Hierbei ist zu beobachten, dass Anbieter von IT-Beratungsleistungen nicht nur methodische Unterstützung anbieten, sondern häufig auch den Anwender bzw. Kunden funktional-fachlich unterstützen und damit in Teilmärkte anderer Beratungsdomänen eindringen [SKH01, S. 4].

Sowohl innerhalb der Informatikdisziplin wie auch innerhalb der Beratungsforschung nimmt die IT-Beratung eine eher untergeordnete Rolle ein [NB06, S. 7]. Während bspw. zur Softwareentwicklung eine Vielzahl von Literatur existiert und auch rund um die Managementberatung verschiedene Forschungsbemühungen zu identifizieren sind[9], scheint die wissenschaftliche Aufarbeitung der IT-Beratung hinter ihrer praktischen Bedeutung zurückzubleiben. Diese Lücken zwischen praktischer Relevanz und wissenschaftlicher Reflexion will der vorliegende Beitrag in Teilen schließen.

Ziel ist es, sowohl verschiedene Rahmenbedingungen aufzuzeigen, unter denen die IT-Beratung agiert, wie auch das generische Geschäftsmodell einer IT-Beratung darzulegen und ausgewählte Details zu skizzieren.

[9] Vgl. exemplarisch die Lehrstühle und Forschungsschwerpunkte an der Carl von Ossietzky Universität Oldenburg, European Business School in Oestrich-Winkel, FH Ludwigshafen sowie der FH Bonn-Rhein-Sieg.

Nach diesem einleitenden Abschnitt werden zunächst unterschiedliche Rahmenbedingungen diskutiert, unter denen die IT-Beratung handelt. Zu nennen ist hier die Frage der Abgrenzung zu anderen Beratungsdisziplinen, die Größe des verfügbaren Marktes, die Beschreibung verschiedener Möglichkeiten im Umgang mit Kunden und ihren IT-Bereichen, den sog. Kontrahierungsregeln und schließlich die Möglichkeit der Messung von IT-Beratungsqualität. Wesentlich beim letzten Punkt ist, dass die Bewertung der Qualität vom Kunden ausgeht und Beratungsunternehmen sich im Sinne einer Beobachtung zweiter Ordnung an dieser Sichtweise orientieren sollten.

Im dritten Abschnitt werden zunächst an Hand des Geschäftsmodells einer Beratung ihre wesentlichen Charakteristika vorgestellt, ebenso wie verschiedene Arten von IT-Beratungsprojekten. Anschließend werden Senioritätsgrade, Karrierepfade und die entstehende Personalpyramide als Differenzierungsmerkmale der internen Organisation aufgezeigt. Diese Aspekte sind abhängig von den bearbeiteten Projekten und gleichzeitig Voraussetzung hierfür. Sie wirken direkt auf den finanziellen Erfolg der Beratung ein.

Der letzte Abschnitt schließlich fasst die Ergebnisse des vorliegenden Beitrages nochmals zusammen und gibt einen Ausblick auf weitere Forschungsfragen und -aktivitäten.

2 Rahmenbedingungen der IT-Beratung

2.1 Begriffsklärung

Der Begriff *Unternehmensberatung* ist nicht geschützt, so dass Beratungsleistungen von verschiedenen Akteuren in unterschiedlichen Bereichen erbracht werden können. Aus diesem Grund erscheint es sinnvoll, zunächst eine einheitliche Verständnisgrundlage zu schaffen. Zur Begriffsklärung soll folgende Arbeitsdefinition für Beratung zu Grunde gelegt werden: „Als organisationale Beratung wird ein (1) professioneller, (2) vertraglich beauftragter (3) Dienstleistungs- und Transformationsprozess der (4) intervenierenden Begleitung durch ein (5) Beratungssystem bei der (6) Analyse, Beschreibung und Lösung eines (7) Problems des Klientensystems – i. S. einer (8) Arbeit an Entscheidungsprämissen – mit dem Ziel der (9) Transformation verstanden.“ [De06a, S. 6]

2.2 Beratungsinhalt

Die genannte Unterstützungsleistung kann in verschiedenen Bereichen erbracht werden. Der inhaltliche Schwerpunkt der Beratungstätigkeit wird hierbei regelmäßig an Hand von fünf Klassen unterschieden (in Anlehnung an [Fe05, S. 26]):

- *Strategieberatung*: Strategieberatung zielt auf den Erhalt und die Verbesserung der langfristigen und strategischen Lebensfähigkeit einer Organisation. Beispiele hierfür sind die strategische Planung und Organisationsentwicklung, strategische Marketingplanung und Corporate Communication.
- *Organisations- und Prozessberatung*: Diese Beratungstätigkeit zielt auf die Verbesserung der operativen Aspekte einer Organisation. Beispiele hierfür sind Business Process Reengineering, Change Management, Projektmanagement, Supply Chain Management.
- *Finanzberatung*: Die Finanzberatung für Organisationen zielt auf die Verbesserung des Einsatzes der finanziellen Ressourcen. Projektbeispiele kommen aus dem Bereich Prüfungsnahe Beratung, Mergers & Acquisitions, Financial Advisory.
- *Personalberatung*: Personalberatung will das personelle Element von Organisationen sowie den Umgang hiermit verbessern. Die Projektbandbreite reicht von der Personalbeschaffung und dem sog. Headhunting über Benefits & Compensation, Interimsmanagement, Weiterbildung und Training bis hin zur Outplacement-Unterstützung.
- *IT-Beratung*: IT-Beratung will die Aufnahme, Verarbeitung, Speicherung und Verfügbarmachung von Informationen in Organisationen verbessern. Beispiele hierfür sind Systementwicklung und -integration, IT-Outsourcing, IT-Architekturen.

Während sich die deskriptive Unterscheidung reibungslos zu vollziehen scheint, gestaltet sich die tatsächliche Einordnung von Beratern und Beratungsprojekten in diese Klassen in der Praxis als schwierig. Als Beispiel mag die Umstellung von Rechnungslegungsvorschriften in Unternehmen vom deutschen HBG auf die IAS/IFRS- bzw. US GAAP-Standards dienen: Hier haben sich sowohl Operations- und Prozessberater, Finanzberater sowie IT-Berater positioniert, die sich der Problemstellung von unterschiedlichen Seiten nähern. Insbesondere größere IT-Beratungsunternehmen mit historischen Bezügen zu anderen Disziplinen erschweren eine eindeutige Zuordnung. Eine solche Unterscheidung ist jedoch bspw. für die Höhe der vereinbarten Tagessätze von Relevanz: IT-Beratung wird im Allgemeinen geringer, Strategieberatung im Allgemeinen höher vergütet.

Die nachfolgenden Ausführungen beziehen sich auf das Feld der IT-Beratung, lassen sich unbeschadet der genannten Schwierigkeiten im Wesentlichen aber auch auf andere Beratungsfelder übertragen.

2.3 IT-Beratungsmarkt

Der europäische Markt für IT-Beratungsleistungen ist in den vergangenen Jahren relativ konstant geblieben. Innerhalb dieses Marktfeldes ist insbesondere ein Ansteigen von Outsourcingleistungen auf Kosten der übrigen (klassischen) IT-Teildisziplinen zu beobachten (vgl. Abbildung 3).

Der gesamte deutsche Beratungsmarkt weist für das abgelaufene Jahr 2004 in Summe ein Volumen von ca. 13,25 Mrd. Euro auf. Hiervon entfällt ca. ein Drittel auf IT-Beratungsleistungen. Auf Outsourcingleistungen entfallen ca. 7,5%. [Fe05, insb. S. 14]

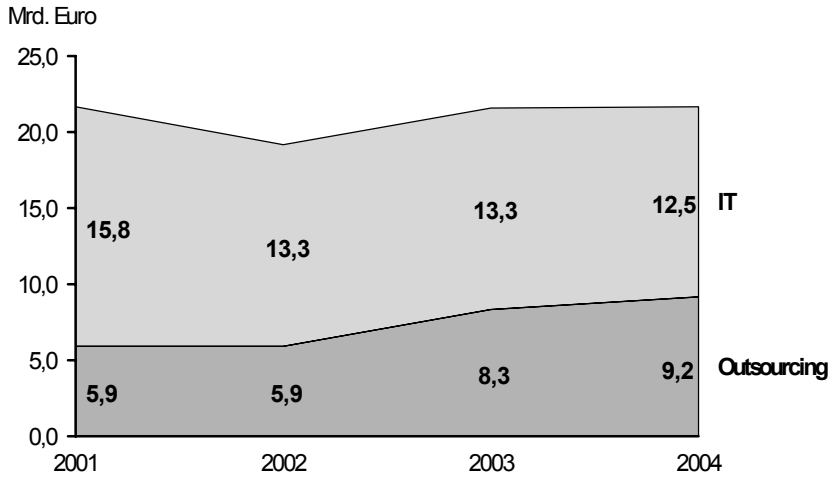


Abbildung 1: Europäischer IT-Beratungsmarkt 2001-2004 [Fe05, S. 8]

2.4 Kontrahierungsregelungen

IT-Beratungsdienstleistungen werden häufig von Unternehmen nachgefragt, die gleichzeitig über interne IT-Ressourcen verfügen. Es besteht an dieser Stelle die Notwendigkeit, eine Zusammenarbeit zwischen beiden Gruppen sowie die Beauftragungsmöglichkeit für externe Dienstleistungen abzustimmen. Die Regelung muss insbesondere die Frage behandeln, ob der interne Lieferant eine Vorzugsbehandlung gegenüber dem externen Lieferanten erfährt und wie diese ggf. ausgestaltet werden kann. Führend in dieser Diskussion sind meist kaufmännische sowie qualitative Argumente. Die Behandlung von internen Lieferanten kann gegenüber ihren externen Kollegen bspw. mit den folgenden Ausprägungen erfolgen:

- *Kontrahierungszwang:* Der interne IT- bzw. Softwareberater (-anbieter) muss genutzt werden.
- *Kontrahierungsgebot:* Der interne IT- bzw. Softwareberater (-anbieter) soll berücksichtigt werden. Bei einem so genannten „First Offer" gibt der interne Berater als erster ein Angebot ab. Gefällt dies dem Kunden nicht, können auch externe Berater um ein Angebot gebeten werden. Eine andere Variante berücksichtigt den so genannten „Last Call". Hierbei darf der interne Anbieter auf Basis der bereits vorliegenden externen Angebote als letzter Anbieter noch selber ein Angebot unterbreiten. Auch die Kombination beider Varianten, also ein „First Offer, Last Call" ist möglich.
- *Wettbewerb:* Es besteht kein Zwang, auf den internen IT- bzw. Softwareberater (-anbieter) bei der Vergabe von Projekten und Aufträgen zurückzugreifen. Der unternehmensinterne Markt entscheidet allein, ob Reputation, Qualität und Preis ausreichend sind, damit der interne Lieferant direkt mit der Projektarbeit beauftragt wird oder sich im Rahmen von Ausschreibungsverfahren, Beauty Contest und Verhandlung für die Projektarbeit qualifiziert.

2.5 Beratungsbewertung durch den Kunden

Eine weitere wesentliche Rahmenbedingung ist die Beratungs- und Projektbewertung durch den Kunden. Kenntnisse an dieser Stelle durch die sog. Beobachtung zweiter Ordnung helfen Beratungsunternehmen, den Anforderungen ihrer Kunden zu entsprechen und ihren Ressourceneinsatz effektiv sowie effizient zu steuern.

Bei der Projektbewertung am Ende eines Projektes werden zwei grundsätzlich verschiedene Facetten unterschieden. Es kann zum einen die Durchführung und der Erfolg des Projektes bewertet werden. Zum anderen kann die Arbeit und der Beitrag des involvierten Beratungsunternehmens sowie der individuellen Berater zum Projektergebnis betrachtet werden.

Typische Fragestellungen und Aspekte bei der Betrachtung eines durchgeführten Projektes am Projektende stammen aus folgenden Bereichen:

- Umsetzung des Projektes
- Projektinduzierte Einsparungen
- Projektinduzierte Investitionen
- ROI (Return on Investment) für Projekteinsatz

Typische Fragestellungen für die Bewertung des involvierten Beratungsunternehmens am Projektende stammen aus folgenden Bereichen:

- Umfang der bereitgestellten Beraterkapazität, Unterlagen, etc.
- Einhaltung vereinbarter Meilensteintermine, etc.
- Qualität der Vertragsabwicklung
- Qualität des Projektmanagements
- Umfang Problem- und Konfliktmanagement
- Innovationskraft des Beraters
- Kompetenz der Mitarbeiter
- Eignung des Methodeneinsatzes
- Genauigkeit der ursprünglichen Aufwandsschätzung
- Transfer von Know-How zum Kundenunternehmen
- Zusammenarbeit zwischen Projektbeteiligten
- Grad der Umsetzungsunterstützung
- Mehrwert durch Beratereinsatz
- Anteil des Beratungsunternehmens an der Ergebniserreichung
- Verhalten bei Preisen und Konditionen

Kundenunternehmen entscheiden für ihre jeweilige Situation, welche Informationen sie erheben wollen und wie viel Aufwand sie hierfür bereit sind zu investieren. Das Ergebnis ist grundsätzlich relevant für eine genauere Beurteilung der jeweiligen Leistung der einzelnen im Unternehmen eingesetzten Berater [HD06]. Eine exemplarische, gewichtete Bewertungsstruktur für IT-Beratungsprojekte zeigt Abbildung 2. Bei den Kriterien werden die für Beratungsleistungen typischen Eigenschaften einer erbrachten Dienstleistung berücksichtigt.

Mit der erfolgten Begriffsklärung, der Beschreibung unterschiedlicher Beratungsinhalte, einer kurzen Marktübersicht, der Darlegung typischer Kontrahierungsregeln für die Zusammenarbeit von internen und externen Ressourcen sowie einer Übersicht zur Kundenbewertung von Beratungsprojekten sind verschiedene Rahmenbedingungen beschrieben worden, unter denen IT-Berater zu operieren haben. In einem nächsten Schritt werden wesentliche interne Strukturen von IT-Beratungsunternehmen betrachtet.

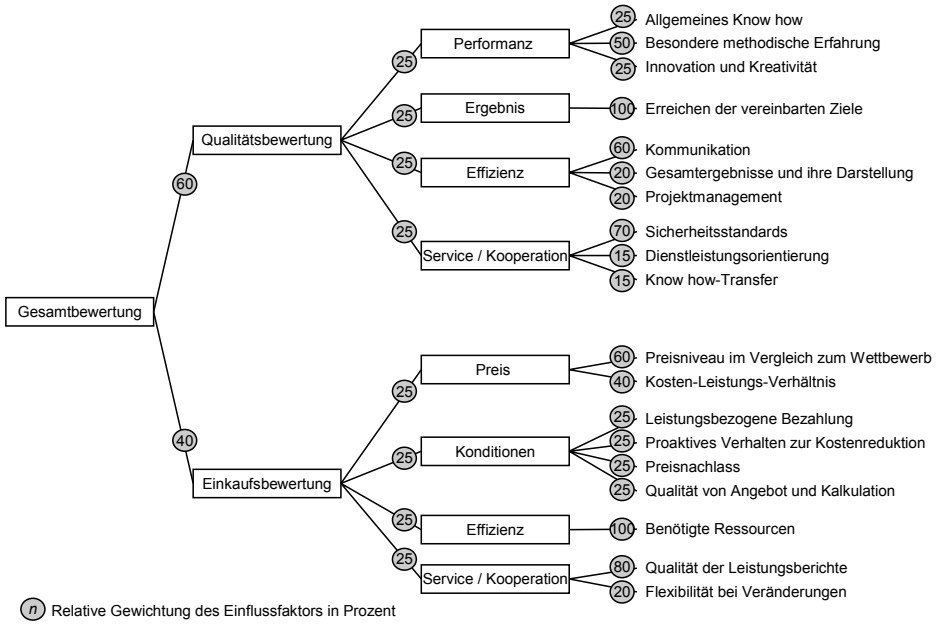


Abbildung 2: Exemplarische Bewertungsstruktur für Beratungsprojekte [De06b]

3 Geschäftsmodell und Charakteristika der IT-Beratung

3.1 Geschäftsmodell

Dem vorliegenden Beitrag liegt folgendes Geschäftsmodellverständnis zu Grunde: „Ein Geschäftsmodell kann als eine abstrahierende Beschreibung der ordentlichen Geschäftstätigkeit einer Organisationseinheit angesehen werden. Diese Abstraktion basiert auf einer Abbildung von Organisationseinheiten, Transformationsprozessen, Transferflüssen, Einflussfaktoren sowie Hilfsmitteln oder einer Auswahl hieraus.“ [SDL03, S. 23]

Das Geschäftsmodell der meisten Unternehmensberatungen sieht vor, dass Berater ihre (Personal-) Ressourcen, ihr Wissen und ihre Fähigkeiten unter Anwendung von Beratungsansätzen, Methodiken etc. zur Problemlösung einsetzen (vgl. Abbildung).[10]

Aus dem Geschäftsmodell lassen sich drei wichtige Elemente herauslösen, die im Rahmen der Unternehmensstrategie betrachtet werden müssen: Mitarbeiter, Wissen und Beratungsansatz. Weitere Merkmale, wie z. B. die Positionierung im Markt, die durch das Verhältnis von Partnern zu Beratern entstehende Hebelwirkung bei der Weitergabe von Wissen oder das Vergütungsmodell sind zur Differenzierung notwendig.

In IT-Beratungen dominieren Projekte gegenüber Regeltätigkeiten. Aufbau- und Ablauforganisation sind also als dynamisch zu betrachten. Beratungsprojekte sind häufig in Phasen unterteilt, die einem wiederkehrenden Schema folgen [Co01, Ni97a, Ni97b]. Aus Sicht einer IT-Beratung besteht ein Projekt im Wesentlichen aus drei Phasen, die ihrerseits weiter unterteilt werden können: Projektanbahnung (mit Anfrage, Angebot, Vertragsgestaltung), Projektdurchführung (mit Analyse, Konzeption, Entwicklung, Einführung) und Projektabschluss (mit direkter Erfolgsbewertung, Dokumentation, Kundenbefragung).

Für die Arbeit in einem IT-Beratungsunternehmen ist es i. d. R. charakteristisch, dass sie an wechselnden sowie verteilten Orten stattfindet, so dass der Zuweisung von Beratern zu Einsatzorten und Projekten eine besondere Bedeutung zukommt. Für das Geschäftsmodell einer Beratung ist es in diesem Zusammenhang wichtig, die Fähigkeiten der Berater richtig einzusetzen, um effizient zu Problemlösungen zu gelangen.

[10] Vgl. für eine Betrachtung des Themenschwerpunktes Personal der Geschäftsmodelle von sog. Professional Service Firms, zu denen auch IT-Beratungen gehören, [Ko04].

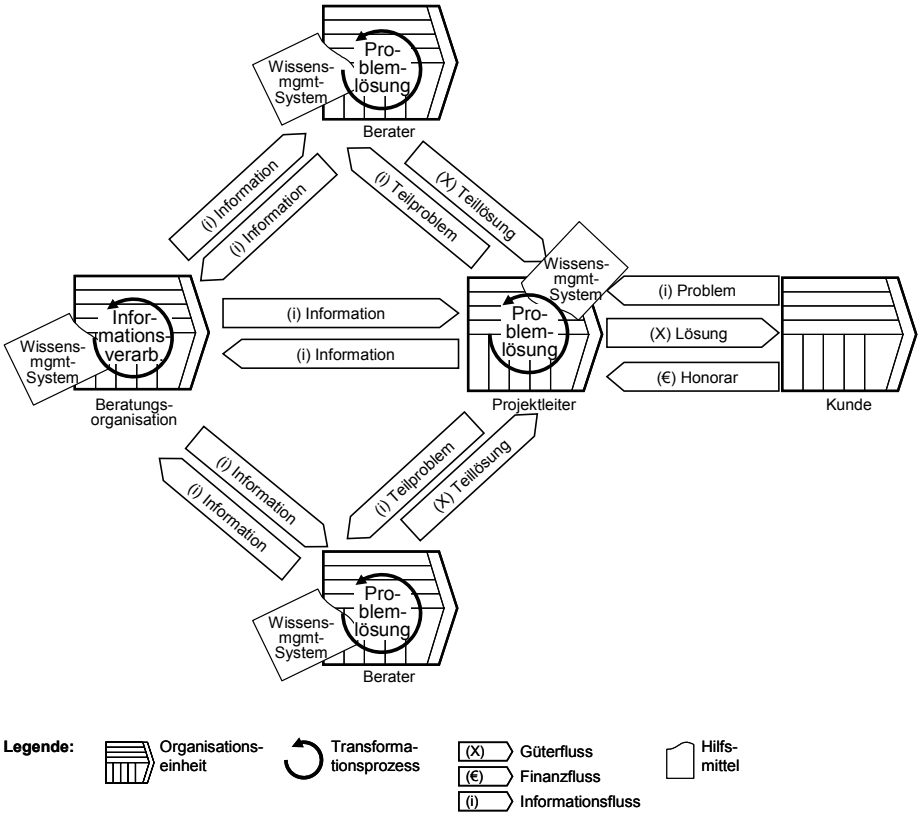


Abbildung 3: Geschäftsmodellbeispiel Beratung (Aufbauend auf [DP05, S. 246])

3.2 Rollen und Projektart

In IT-Beratungsprojekten kann der Berater verschiedene Rollen, Aufgaben und Funktionen einnehmen, z. B. Entwicklungs- und Innovationsfunktion, Durchsetzungsfunktion, Wirtschaftlichkeitsfunktion oder Katalysatorfunktion.[11]

Weiterhin lassen sich IT-Beratungsprojekte nach den bearbeiteten Fragestellungen unterteilen. Die Art der bearbeiteten Projekte kann in drei verschiedene Klassen oder Typen unterschieden werden, wobei die Praxis zeigt, dass es sich hierbei um eine Einordnung an Hand des Projektschwerpunktes handeln muss, da Projekte meist auch die angrenzenden Gebiete behandeln:

- *Insight:* Projekte dieser Gruppe fragen nach dem „Was?“ Sie wecken beim Kunden zum einen das Problembewusstsein, zum anderen weisen sie aber auch den Weg zur Lösung des Problems.

[11] Vgl. exemplarisch [SKH01, S. 2-3 oder Rie04]

- *Concept:* Die sogenannten Concept-Projekte fragen nach dem „Wie?“ Sie greifen die Insights auf und konkretisieren diese bspw. auf organisatorischer oder prozessualer Ebene.
- *Implementation:* Projekte, die in dieser Gruppe zu finden sind, fragen nach dem „Wann?“ Sie führen die entwickelten Konzepte aus und setzen diese bspw. in Lösungen um.

Diese Schwerpunktbildung hat Auswirkungen bspw. auf die benötigten Berater, ausgedrückt in ihren Skills und Senioritätsgraden sowie auf die Personalpyramide, d. h. dem Verhältnis von Partnern zu Beratern in einer Beratung. Implementation-Projekte verlangen nach einer größeren Anzahl von Beratern pro Partner. Weniger Berater benötigen Concept-Projekte, am wenigsten Insight-Projekte. [DP04, S. 179-180]

Eine alternative Art der Projektaufteilung bezieht sich auf sog. „Brain-“, „Grey Hair-“ und „Procedures-Projekte.“ Brain-Projekte sind von hoher Komplexität und meist fachliche oder technologische Vorreiterprojekte, die mit Kreativität, Innovationskraft und neuen Ansätzen neuen Probleme mit neuen Lösungen begegnen. Bei Grey Hair-Projekten werden bekannte Methoden auf kundenspezifische Situationen adaptiert. Berater werden dann für solche Projekte engagiert, wenn sie Erfahrungen auf eben diesem Gebiet vorweisen können. Der Neuigkeitsgrad ist bei Procedure-Projekten am geringsten. Sowohl das Problem, als auch die notwendige Lösung ist meist bereits im Vorfeld des Projektes bekannt. Ein Berater wird ausgewählt, wenn er das Projekt mit einer hohen Effizienz lösen kann. [Ma82]

3.3 Senioritätsgrade

Wie weiter oben bereits angesprochen, sind verschiedene Senioritätsgrade von Relevanz, um Beratungsprojekte effektiv mit Personal auszustatten. Hierbei haben konzeptionelle IT-Beratungsprojekte andere Anforderungen als bspw. Implementierungsprojekte oder Parametrisierungen von Standardsoftware. Die Mitarbeiter einer Beratung lassen sich grundsätzlich in die beiden Gruppen „Partner“ und „Berater“ einteilen. In den meisten Beratungen sind die Partner die Gesellschafter der Unternehmung, während die Berater meist auf diesen Status hinarbeiten. Diese beiden Grundkategorien werden jedoch in der Praxis i. d. R. weiter unterteilt. Obwohl die Bezeichnung der einzelnen Senioritätsgrade, Beraterkategorien, Ranks, Stufen etc. häufig nicht einheitlich ist, d. h. beratungsspezifische Unterschiede aufweist, lassen sich doch gewisse Gemeinsamkeiten identifizieren. Diese beziehen sich zum einen auf die Struktur der ansteigenden Aufgabenkomplexität und zum anderen auf eine steigende Verantwortung und damit einhergehende Kompetenzerweiterungen.

Die folgende Aufstellung präsentiert typische Charakterisierungen mit abnehmender Seniorität. Ihre Kenntnis und Anwendung hilft bspw. bei der Standardisierung von Beratereinsätzen. Während die Bezeichnungen fallweise variieren können, ist die grundsätzliche Unterteilung auf Berater- sowie auf Kundenseite etabliert:

Tabelle 1: Senioritätsgrade und ausgewählte Charakteristika (Aufbauend auf [Kr05])

Senioritätsgrad (und exemplarische Tätigkeits-beschreibungen)	**Ausgewählte Charakteristika**
Senior Partner (Managing Director, Senior IT-Management Coach, Programm Management/-Head, Functional (Senior-) Architect, Program Conceptioner)	- Zentraler Entscheider und Mitglied der Leitungsebene auf Seiten der IT-Beratung - Einsatz i.d.R. nur punktuell in den Projekten bzw. projektbegleitend - Erfahrung in der Leitung von großen IT-Beratungsaufträgen mit hoher Komplexität und strategischer Bedeutung - Verfügt über themenübergreifendes sowie branchenspezifisches Spezialwissen - Verfügt über Spezialkenntnisse im organisatorischen, informationstechnischen und strategischen Umfeld des Kunden - Ansprechpartner bei grundsätzlichen Problemen in der Zusammenarbeit zwischen Beratung und Kunde
Junior Partner (Senior Manager, Project Manager, Technical (Senior-) Architect, Functional Designer, Technical Expert / Conceptioner)	- Erfahrung in der Leitung von großen IT-Beratungsaufträgen mit hoher Komplexität - Ausgezeichnete Kenntnisse in Fragen von IT-Architektur, Aufbau- und Ablauforganisationen - Erfahrung in der Leitung bzw. Steuerung des Projektteams - Verfügt über themenübergreifendes sowie branchenspezifisches Spezialwissen - Verfügt über projektrelevante Spezialkenntnisse im Umfeld des Kunden - Ansprechpartner bei Problemen im Projektteam
Project Manager (Group Leader, Technical Designer, Test Conceptioner)	- Erfahrung in der Leitung von mittleren und kleinen IT-Projekten - Verfügt über projektrelevantes Spezialwissen - Ausgeprägte analytische Fähigkeiten - Entwickelt selbständig Lösungskonzepte - Ausgeprägte Sozialkompetenz
Senior Consultant (Senior Software Developer, Software Analyst)	- Bearbeitet eigenverantwortlich Projektmodule - Führt Arbeitsteam effektiv - Ausgezeichnete Kenntnisse ausgewählter Softwareprogramme/-lösungen - Ausgeprägte analytische Fähigkeiten - Sicherer Umgang mit einschlägigen Tools und Methodiken - Ausgeprägte Sozialkompetenz
Consultant, Junior Consultant (Business Analyst, Software Developer, Software Tester)	- Eigenverantwortliche Erledigung übertragener Projektarbeiten, insbesondere in der Dokumentation - Rechercheaufgaben und Informationsbeschaffung, Pflege von Projektmanagement-Tools - Ausgeprägte analytische Fähigkeiten - Sehr gute Kenntnisse ausgewählter Softwareprogramme/-lösungen - Geübt im Umgang mit einschlägigen Tools und Methodiken - Abgeschlossenes Studium oder vergleichbarer Abschluss - Sozialkompetenz (Kommunikationsfähigkeit, Teamfähigkeit, Flexibilität)

3.4 Karrierepfade

Innerhalb von IT-Beratungen gibt es verschiedene Möglichkeiten, wie die Beraterkategorien zur Karriereentwicklung genutzt werden können. Zunächst kann die durchschnittliche Verweilzeit auf einer Stufe betrachtet werden. Hiermit ist diejenige Zeit gemeint, die ein Mitarbeiter z. B. als Junior Berater im Beratungsunternehmen verbringt, bevor er den Rang eines Senior Beraters erhält. Typisch sind Verweilzeiten von 2-3 Jahren als Junior Consultant, 3 Jahren als Senior Consultant, 2-3 Jahren als Project Manager und 3-4 Jahren als Junior Partner. Die Zugehörigkeit zur Gruppe der Senior Partner ist in der Regel zeitlich nicht begrenzt.

Für die Individualkarrieren in Beratungsunternehmen ist weiterhin die Fluktuationsquote relevant. Um die höheren Karrierestufen als attraktiv und erstrebenswert darzustellen, muss ihre Erreichung als eine Auszeichnung wahrgenommen werden. Dies kann erreicht werden, indem bspw. nicht jedem Mitarbeiter der Sprung auf eine höhere Karrierestufe gewährt wird und die Ergebnisse einer Leistungsevaluation entsprechend zur Personalentwicklung genutzt werden. [EG04]

Beratungsunternehmen können sich neben der Entwicklung ihres Beratungspersonals auch durch die Einstiegsstufen von ihren Wettbewerbern differenzieren. Hier lassen sich zwei grundsätzliche Ausrichtungen unterscheiden. Auf der einen Seite können Beratungen lediglich Hochschulabsolventen einstellen („New Hires“) und Mitarbeiter höherer Karrierestufen nur aus Kandidaten der jeweils darunter liegenden Karrierestufe rekrutieren. Auf der anderen Seite können sie auch Quereinsteigern die Möglichkeit eröffnen, eine höhere Karrierestufe zu bekleiden, ohne alle der niedrigeren Stufen durchlaufen zu haben („Experienced Hires“). [RST05]

Eine Besonderheit dieser Unterscheidung stellen Beratungen dar, die als Mitarbeiter lediglich Personen mit ausgewiesener (z. B. fünfjähriger) Berufserfahrung einstellen. Sie nutzen die außerhalb der Beratung gesammelten Erfahrungen ihrer Mitarbeiter als Verkaufsargument gegenüber dem Kunden. Beratungen, die ein solches Geschäftsmodell verfolgen, können Projekte, die einen personellen Unterbau von junioren Beratern erfordern, c. p. schwerer bearbeiten, als Beratungen mit ebendiesem Unterbau.

Die angeführten Kriterien der Verweilzeit, Umgang mit nicht beförderten Beratern sowie Einstellungsmöglichkeiten haben u. a. Auswirkungen auf das Verhältnis von Partnern zu Beratern und auf die Typen der durchführbaren Projekte.

3.5 Personalpyramide

Aus den gerade diskutierten Charakteristika von Projekttypen, Senioritätsgraden sowie Karrierepfaden ergeben sich verschiedene Möglichkeiten der Ausgestaltung von Personalpyramiden. Drei mögliche Personalverteilungen von junioren und senioren Beratern werden in Abbildung schematisch dargestellt und unterschieden.[12]

[12] Vgl. für eine ähnliche Aufteilung von Projekttypen auch [Ma82, S. 24 f. und Ri04].

Zunächst ist die stark spezialisierte IT-Beratung zu nennen. Sie hat einen sog. Boutique-Charakter und verfügt über eine fast gleichmäßige Verteilung der einzelnen Hierarchiestufen. Im Verhältnis zu den anderen Organisationsmöglichkeiten ist diese Art sehr senior ausgestattet. Dies bedingt die notwendige hohe fachliche Kompetenz der bearbeiteten Projekte. Diese fokussieren i. d. R. auf eine eng umrissene fachliche Fragestellung.

Eine Diamantenform weist die Personalstruktur von sog. Transformationsberatungen auf. Sie übernehmen vornehmlich Beratungsprojekte, welche die langfristige Veränderung von Organisationen zum Ziel haben. Da das Endergebnis der Transformation in weiten Teilen skizziert ist, gleichzeitig aber Projekteinsätze an unterschiedlichen Stellen des Unternehmens notwendig werden, bedarf es einer großen Anzahl von erfahrenen Beratern, die in der Lage sind, komplexe Projektsituationen zu steuern und auch die Einbindung von Kundenmitarbeitern in die Projektteams übernehmen können. Dies ist gleichzeitig ein wesentlicher Grund für den im Gegensatz zur klassischen Pyramide abgeschwächten personellen Unterbau.

Eine sehr breite Basis von junioren Beratern und eine sehr geringe Zahl von senioren Beratern weisen Implementierungsberatungen oder solche, die ganze Projektprogramme betreuen, auf. Hierbei wird eine einmalige, größere Aufgabe in einem abgegrenzten Gebiet behandelt. Das Projektergebnis ist bspw. in Form einer implementierten Software vorhanden und kann den Charakter eines Produktes aufweisen. Die gute Anleitbarkeit der in solchen Projekten auftretenden Aufgaben unterstützt die Delegierbarkeit an juniore Beratungsmitarbeiter und damit auch die flache Pyramidenform.

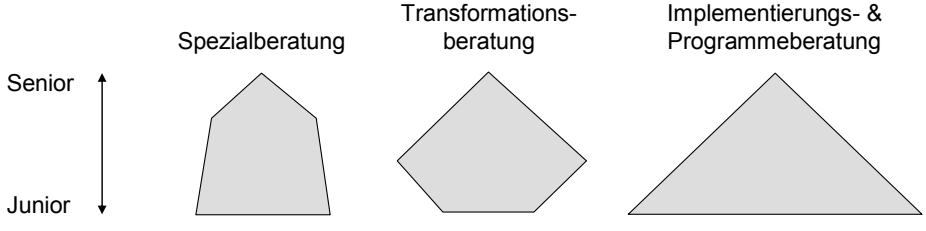


Abbildung 4: Schematische Darstellung Personalverteilung bei unterschiedlichen Projekttypen

4 Zusammenfassung und Ausblick

Der vorliegende Beitrag hat zunächst eine Differenz zwischen der praktischen Relevanz von IT-Beratung und ihrer Reflexion in der Literatur identifiziert. Ziel war es dementsprechend, diese Lücke in Teilen zu schließen. Hierzu sind in einem ersten Schritt im Sinne einer Außensicht verschiedene Rahmenbedingungen expliziert worden, in denen sich Anbieter von IT-Beratungsleistungen bewegen. Zu nennen sind hier die Abgrenzung zu anderen Beratungsleistungen, der zur Verfügung stehende Markt sowie die Zusammenarbeit bzw. Abgrenzung zu kundeneigenen IT-Ressourcen, ausgedrückt durch Kontrahierungsregelungen. Als relevant für Beratungsunternehmen wurde weiterhin die durch den Kunden wahrgenommene Qualität der Beratungsleistung herausgestellt. Hier wurden Parameter der Beratungsbewertung genannt und in eine Bewertungsstruktur überführt.

In einem zweiten Schritt sind im Sinne einer Innensicht das generische Geschäftsmodell von IT-Beratungsunternehmen sowie verschiedene Projekttypen zusammen mit wesentlichen Eigenschaften charakterisiert worden. Zu nennen sind hier insbesondere die verschiedenen Senioritätsgrade, die eine Beratung typischerweise aufweist, die Verweildauer auf den unterschiedlichen Stufen und die sich hieraus ergebene Personalpyramide.

Obige Ausführungen wollen einen Beitrag zur Beschreibung der Beratungstätigkeit als Bindeglied zwischen verschiedenen Beteiligten liefern. Neben diesen eher strukturellen Betrachtungsweisen erscheint es beispielsweise für zukünftige Forschungsaktivitäten notwendig, auch prozessorientierte Überlegungen, z. B. entlang des Beratungsprozesses, zu explizieren. Ebenso mag es interessant sein, eine „Spezielle Betriebswirtschaftslehre für Beratungsunternehmen“ zu diskutieren oder Charakteristika von eher großen IT-Beratungen mit denen von eher kleinen IT-Beratungen gegenüber zu stellen.

Literaturverzeichnis

[Co01] Cope, M.: Consulting mit System. Financial Times Prentice Hall, München, 2001.

[De06a] Deelmann, T.; Huchler, A.; Jansen, St. A.; Petmecky, A.: Internal Corporate Consulting: Thesen, empirische Analysen und theoriegeleitete Prognosen zum Markt für Interne Beratungen. ZU-Schnitte Discussion Paper, Friedrichshafen, 2006.

[De06b] Deelmann, T.: Kennzahlen für die kunden- und beraterseitige Steuerung von Beratungsprojekten. In (Nissen, V.; Schneider, H., Hrsg.): Prozessorientiertes Corporate Performance Measurement – Methoden, Konzepte, Anwendungen. Ilmenau, 2006, S. 145-158.

[DP04] Deelmann, T.; Petmecky, A.: Beratermanagement – Objektive Optimierung des Beratereinsatzes bei der Deutschen Telekom. In (Treichler, Chr.; Wiemann, E.; Morawetz, M., Hrsg.): Corporate Governance und Managementberatung. Gabler Verlag, Wiesbaden, 2004, S. 177-201.

[DP05] Deelmann, T.; Petmecky, A.: Optionenraum für Geschäftsmodell, Strategie, Aufbau- und Ablauforganisation einer Unternehmensberatung. In (Petmecky, A.; Deelmann, T., Hrsg.): Arbeiten mit Managementberatern – Bausteine für eine erfolgreiche Zusammenarbeit. Springer-Verlag, Berlin, Heidelberg, 2005, S. 245-255.

[EG04] Erhard, A.; Gerds, M.: Die Auswahl von Bewerbern und Leistungsevaluation von Beratern in Top-Management-Beratungen. In (Nippa, M.; Schneiderbauer, D., Hrsg.): Erfolgsmechanismen der Beratung. Physica-Verlag, Heidelberg, 2004, S. 209-225

[Fe05] Feaco (Hrsg.): Survey of the European Management Consultancy Market. Brüssel, 2005.

[HD06] Haferkamp, S.; Drescher, S.: Client Professionalization: Proposed Approach for the Knowledge-Centered Management of Consulting Projects. In (Deelmann, T.; Mohe, M., Hrsg.): Selection and Evaluation of Consultants. Hampp-Verlag, Mehringen, 2006, im Erscheinen.

[Ko04] Koehler, M.: Markenmanagement ist Personalmanagement. In (Ringelstetter, M.; Bürger, B.; Kaiser, S., Hrsg.): Strategien und Management für Professional Service Firms. Wiley-VCH, Weinheim, 2004, S. 243-255.

[Kr05] Kraus, A.: Der Einkauf als Intermediär zwischen Berater und Beratenem. In (Petmecky, A.; Deelmann, T., Hrsg.): Arbeiten mit Managementberatern. Springer-Verlag, Berlin, Heidelberg, New York, 2005, S. 65-75.

[Ma82] Maister, D. H.: Balancing the Professional Services Firm. In: Sloan Management Review, Fall 1982, S. 15-29.

[NB06] Niehaves, B.; Becker, J.: Design Sciences Perspectives on IT-Consulting. In (Lehner, F.; Nösekabel, H.; Kleinschmidt, P., Hrsg.): Wirtschaftsinformatik 2006, Tagungsband 2, Gito-Verlag, Berlin, 2006, S. 7-17.

[Ni97a] Niedereichholz, C.: Unternehmensberatung – Band 1: Beratungsmarketing und Auftragsakquisition. Oldenbourg Verlag, München, Wien, 1997.

[Ni97b] Niedereichholz, C.: Unternehmensberatung – Band 2: Auftragsdurchführung und Qualitätssicherung. Oldenbourg Verlag, München, Wien, 1997.

[Ri04a] Richter, A.: Organisation von Beratungsfirmen: Ein Erfolgsfaktor. Vortrag im Rahmen der „ConsultingImpulse 2004“, Oestrich-Winkel, 2004.

[Ri04b] Riedl, M.: Rollen eines Managementberaters. In (Nippa, M.; Schneiderbauer, D., Hrsg.): Erfolgsmechanismen der Beratung. Physica-Verlag, Heidelberg, 2004, S. 137-164.

[RST05] Richter, A.; Schmidt, S.; Treichler, Chr.: Organisation und Mitarbeiterentwicklung als Differenzierungsfaktor. In (Sommerlatte, T.; Mirow, M.; Niedereichholz, C.; Windau, P. G. v., Hrsg.): Handbuch der Unternehmensberatung - Organisationen führen und entwickeln. Erich Schmidt Verlag, Berlin, 2005, S. 1-18.

[SDL03] Scheer, Chr.; Deelmann, T.; Loos, P.: Geschäftsmodelle und internetbasierte Geschäftsmodelle – Begriffsbestimmung und Teilnehmermodell. Working Papers of the Research Group Information Systems & Management, Paper 12, Mainz, 2003.

[SKH01] Scheer, A.-W.; Köppen, A.; Hans, A.: Consulting: Ein Überblick. In (Scheer, A.-W.; Köppen, A., Hrsg.): Consulting – Wissen für die Strategie-, Prozess- und IT-Beratung, 2. Aufl., Springer Verlag, Berlin, Heidelberg, 2001, S. 1-10

Beratung bei der Auswahl eines IT-Dienstleisters:

Das Projekt Chefplan Online

Heiner Merz, Andreas Helferich, Georg Herzwurm, Andreas Platzer

Universität Stuttgart
Lehrstuhl für Wirtschaftsinformatik II
(Unternehmenssoftware)
Breitscheidstrasse 2c
70174 Stuttgart

{merz | helferich | herzwurm}@wi.uni-stuttgart.de

BBE Handelberatung GmbH
Brienner Str. 45
80333 München
platzer@bbe.de

Abstract: Die BBE Handelsberatung GmbH vertreibt seit Jahren erfolgreich ein Excel-basiertes Tool für Unternehmensplanung und Controlling. Im Rahmen eines vom Bayerischen Staatsministeriums für Wirtschaft, Infrastruktur, Verkehr und Technologie geförderten Projekts, sollte dieses Tool zu einer Internet-basierten Lösung weiterentwickelt werden. Dabei war für BBE klar, dass man einen externen Partner brauchen würde, um diese Lösung entwickeln und später betreiben zu können. Der Auswahlprozess eines geeigneten IT-Dienstleisters wurde vom Lehrstuhl für Wirtschaftsinformatik II der Universität Stuttgart beratend begleitet. Dabei wurde die Qualitätsmanagement- und Produktentwicklungsmethode Quality Function Deployment (QFD)[*] zum Vergleich und zur Auswahl der externen Partner verwendet. Einerseits wurden mittels QFD die (Software-) Realisierungskonzepte der potentiellen Partnerfirmen analysiert, verglichen und bewertet bezüglich den Projektanforderungen, welche vorab ebenfalls mit QFD-Methoden aufgestellt wurden. Andererseits wurde QFD angewendet, um die potentiellen, externen Partnerfirmen an sich zu messen, zu vergleichen und zu bewerten – nämlich bezüglich deren Referenzen, deren Arbeitsweisen, Qualitätssicherungs- und Testmethoden, finanzieller Solidität, Sicherheiten, Garantiezusagen usw. Die Auswahl des am besten geeigneten, externen IT-Partners ist somit basiert auf rückverfolgbaren, verifizierten und objektiven Entscheidungsmechanismen – auf QFD.

[*] Quality Function Deployment ist eine zu Beginn der siebziger Jahre in Japan von Professor AKAO u.a. entwickelte Qualitätsmethode zur Ermittlung der Kundenanforderungen und deren direkten Umsetzung in die notwendigen technischen Lösungen. QFD, sinngemäss übersetzt mit: "Aufmarsch der Qualitätsmerkmale" wird QFD als ein vorbeugendes Werkzeug zur Produktdefinition eingesetzt und hat über die USA Eingang nach Europa gefunden. Es ist ein systematischer Weg, der sicherstellt, dass die Festlegung der Produktmerkmale durch die Entwicklung und die anschliessende Auswahl der Produktionsmittel, Methoden und Kontrollmechanismen ausschliesslich von den Anforderungen der zukünftigen Kunden bestimmt werden. *QFD Institut Deutschland e.V.*: http://www.qfd-id.de

1 Einleitung

Im Bereich der Informationstechnologie (IT) spielt die Einbeziehung von und die Zusammenarbeit mit externen Partnern von Jahr zu Jahr eine bedeutendere Rolle. Dies geschieht sowohl im Outsourcing, als auch in der Projektarbeit, sowohl im Customizing, als auch in der Entwicklung neuer Software. Daher wird der Auswahl- und Selektionsprozess von passenden und kompetenten externen Partnerfirmen zunehmend wichtiger und bedeutender für den Geschäftserfolg einer Firma. Mehrere substanzstarke, international bekannte und weltweit bedeutende Unternehmen sowie weitere, in ihrem Bereich jeweils marktführende Firmen, betrachten das Finden, die Auswahl und das Betreuen von externen Partnern mittlerweile als Kernkompetenz in ihrer Firma, [May97].

Dieser Beitrag behandelt die erfolgreiche Verwendung von QFD zur Auswahl eines externen Softwareentwicklungspartners am Beispiel eines überschaubaren Mittelstands-Projektes; eine exakt vergleichbare Situation und Aufgabenstellung, das gleiche Beraterteam, sowie dieselbe Vorgehensweise, fanden ab Mai 2004 Anwendung zur Auswahl des externen IT-Projektpartners für einen der derzeit weltgrössten KFZ-Zulieferer- und Elektronikkonzerne. Das dortige Projekt ist die extern vergebene Entwicklung eines e-Learning Management System für über 20.000 Anwender im globalen Gesamtkonzern.

Die Co-Autoren dieses Beitrags wurden jeweils und vom jeweiligen Auftraggeber als Berater mit der Auswahl des externen Softwareentwicklungspartners beauftragt.

2 Herkömmliche Lieferantenauswahlmethoden

Im Allgemeinen wird, auch in der IT-Industrie, eine Lieferantenauswahl durch Ausschreibungen durchgeführt. Diese sind idealerweise zweistufig gegliedert in eine Ausschreibung per Lastenheft (Schwerpunkt fachliche/technische Erfordernisse), gefolgt von einer Ausschreibung, welche die Preise betrifft (plus weitere kaufmännische und rechtliche Faktoren). Das Lastenheft beinhaltet das vordefinierte Projektziel, die gewünschte Funktionalität, erwartete Ergebnisse, erforderliche Schnittstellen, Nebenwirkungen (‘Side-Effects’) usw. Falls das IT-Projekt Teil eines übergeordneten Ganzen ist, oder falls die Software des IT-Projektes mit weiteren Geräten und/oder Software zusammenarbeiten muss, so müssen diese Schnittstellen und Anforderungen ebenfalls a priori festgelegt sein. Die Suche und Auswahl möglicher externer Projektpartner sollte erst nachfolgend oder frühestens parallel zur Lastenhefterstellung erfolgen. Falls ein oder mehrere mögliche Lieferanten nämlich bereits im Geschäft mit der ausschreibenden Firma sind, so wird die ausschreibende Stelle sehr stark in Versuchung geführt, die Kenntnisse und die einschlägige Expertise der geschäftlich ja befreundeten Partner zu nutzen im schwierigen (und fachlich die ausschreibende Stelle stark herausfordernden, oft überfordernden) Prozess der Anforderungsfindung und Anforderungsbewertung. Das Produkt würde hierdurch u.U. jedoch am Bedarf oder am Markt vorbeientwickelt werden. Potentielle Lieferanten, welche sich in den Prozess einer Lastenhefterstellung involvieren können, erliegen i.d.R. der Versuchung, das Lastenheft in ideal geeigneter Weise (nur) für sich selbst zu erstellen. Folglich sollte die Suche nach geeigneten Projektpartnern bzw. Lieferanten erst nach der Lastenhefterstellung statt-

finden (frühestens parallel); typischerweise kommen hierzu Messen, Ausstellungen, Kataloge, Internetsuche usw. zur Anwendung. Doch selbst wenn eine Suche und eine Auswahl potentieller Lieferanten völlig unabhängig erfolgt, dann ist die Vorauswahl der in Frage kommenden Lieferanten oft dominiert von subjektiven Werte wie Empfehlungen oder Ruf des Lieferanten. Doch Empfehlungen oder Ruf sind, ebenso wie Erfahrungen, nur von hohem Wert in einem sich unveränderlichem oder nur langsam änderndem Umfeld. Die Situation der sich rasch weiterentwickelnden Welt, besonders in der IT-Branche, macht jedoch den Wert von Erfahrungen oder Reputation fraglich.
Es gibt eine Vielzahl von Publikationen über Lieferantenauswahlstrategien im produzierenden Gewerbe, z.B. [DYY04], oder im Supply Chain Management, z.B. [ESS01]. Diese Strategien habe alle gemeinsam, dass der Lieferant physische Waren liefert – und zur Lieferantenbeurteilung können aus diesen Waren Proben entnommen, geprüft und getestet werden. Software ist jedoch ein immaterielles Gut. Software hat lediglich Funktion, aber kein physisches Dasein. Zudem muss Software üblicherweise einzigartig entwickelt (oder customisiert) werden – denn falls nicht, dann müsste sie ja nicht entwickelt, sondern bräuchte einfach nur kopiert zu werden. Oder marktgängige Standardsoftware könnte (parametrisiert/customisiert) zur Anwendung kommen. Daher ist jeder Herstellungsprozess von Software ebenfalls einzigartiger Natur. Klar können Softwareentwickler schon oft Ähnliches erschaffen haben – aber genau dieses Eine eben noch nicht. Ein Anbieter kann auch nicht Proben der zu entwickelnden Software zur Anschauung liefern, ein Anbieter kann nur auf ähnliche oder vergleichbare Aufgaben verweisen und seine dortigen Erfolge und Fähigkeiten ins Licht stellen.

In herkömmlichen Lieferantenauswahlmethoden werden alle bekannten Fakten und Anforderungen festgehalten und die Ausschreibung wird an eine Vorauswahl einiger potentiell geeigneter Softwareerstellungsunternehmen gesendet. Der Rücklauf wird dann beispielsweise durch Prüflisten und Kriterienkataloge sowie durch die subjektive Meinung der hier tätigen Mitarbeiter beurteilt. Schließlich wird eine Entscheidung für den subjektiv besten Partner getroffen. Dieser nur allzu gängigen Vorgehensweise fehlt es an Objektivität und Nachvollziehbarkeit der Entscheidungen. Ihr fehlt es am Überblick, ihr fehlt es an der Flexibilität im Laufe der Auswertung Kriterien neu aufnehmen, modifizieren oder neu gewichten zu können und ihr fehlt es an der Sicherheit, im Laufe des Prozesses nicht einige Anforderungen zu vergessen oder vernachlässigt zu haben. Auch ist die Handhabung von mehr als nur einer Handvoll Angeboten auf diese Art nahezu unmöglich.

Literatur über die Handhabung von externen (IT-)Projektpartnern oder Outsourcing-Auftragnehmern beschreibt oft detailliert, wie beispielsweise die Vertragsgestaltung oder das Controlling des externen Partners erfolgen kann – d.h. erst die späteren Projektphasen. Die vorhergehenden Schritte, nämlich die Problematik des Findens und der Wahl des externen Partners an sich, diese Schritte werden in der Literatur nur beiläufig erwähnt und beschränken sich i.d.R. auf Ausschreibungen plus Rücklaufprüfung.

In [ESS01] wird konstatiert, dass eine Lieferantenauswahl ein mehrdimensionales Entscheidungsproblem ist [WCB91], [Dic66], [G0B98]. Methoden in einschlägiger Literatur können in zwei Kategorien klassifiziert werden: Mathematische Programme und Gewichtungsmodelle. So basiert beispielsweise in [HJY05] ein Lieferantenauswahlmodell auf einer Support Vector Maschine (SVM). Deren Algorithmus berechnet ein Auswahlmodell durch Berechnung aus gewichteten Anforderungen. Das Ergebnis ist

eine schnelle und effektive Methode zur Lieferantenauswahl, unwissenschaftliche traditionelle Methoden werden vermieden: Wenn erst die Anforderungen gewichtet sind, dann wird der Lieferant mittels eines exakten Rechenweges ermittelt. Die Schwachstelle hier ist jedoch offensichtlich: Experten im Einkauf oder Management müssen vorab die Gewichtung mittels einer Skala vergeben. Egal wie erfahren oder wissend diese Experten sind, der SVM-Input ist nur eine menschliche (Experten-) Schätzung.
Somit sollte das trügerisch exakte, mathematisch ermittelte SVM Resultat gleichwohl nur so gesehen werden: Als eine abgeleitete Schätzung.

Nichtsdestotrotz kann SVM in die richtige Richtung weisen, falls es mit Rückkopplung auf die Eingangsgrößen sowie mehrmaligen Iterationen auf ein Ergebnis hin ausgeführt würde. Die Anwendung von QFD in den frühen Phasen einer Lieferantenauswahl eliminiert diese Schwachpunkte: Die Gewichtung und Auswahl der externen IT-Partner wird methodisiert.

3 Einführung in QFD

Quality Function Deployment (QFD) bezeichnet die integrierte Anwendung verschiedener Qualitätstechniken (vgl. hierzu und im Folgenden [HSM97]). Im engeren Sinne, dann auch Task Deployment genannt, ist damit das "Deployment of Quality Functions" gemeint, welches die Identifikation, Analyse und den schrittweisen, zielgerichteten Einsatz all der Aktivitäten einer Unternehmung beinhaltet, die zur Planung und Sicherung eines gewissen Qualitätsniveaus über Abteilungsgrenzen hinweg dienen (vgl. [MAY94]). Im weiteren Sinne ist QFD sowohl eine prozess- als auch eine produktbezogene Methode. Es fasst daher die Verbesserung der unternehmensinternen Prozesse und die Übersetzung aller (Qualitäts-)Anforderungen der Kunden in Produktcharakteristika ("Comprehensive QFD" genannt) zusammen (vgl. hierzu Abb. 1). Dieses Vorgehen wurde 1966 in Japan in der Industrie und für die Industrie entwickelt. Mittels QFD soll der Entwicklungsprozess, vor allem die Aktivitäten der Produktplanung, strukturiert und systematisch ablaufen. Hierzu wird die "Stimme des Kunden" (engl.: "Voice of the Customer") explizit aufgenommen und in die Sprache der Entwickler transferiert. Des Weiteren wird ein Mechanismus bereitgestellt, der die wichtigsten Kundenbedürfnisse transparent durch den gesamten Entwicklungsprozess trägt. Ziel ist, alle Aktivitäten in der Produkterstellung, zumindest mittelbar, auf Kundenanforderungen zurückführen zu können. In jeder Entwicklungsphase soll der grösste Aufwand auf die Verrichtung der Aufgaben verwendet werden, welche den höchsten Anteil an der Erhöhung des Kundennutzens und der Befriedigung der Kundenbedürfnisse haben. Dabei fordert QFD schon in der Produktplanung eine abteilungsübergreifende Zusammenarbeit innerhalb der Unternehmung sowie mit den Kunden, um die angestrebten Entwicklungsziele jedem Beteiligten frühzeitig vor Augen zu führen und eine Verpflichtung gegenüber dem Produkt zu etablieren. QFD bedingt eine Teambildung, d.h. alle Betroffenen zu Beteiligten zu machen, indem in QFD explizit die Identifikation von Benutzergruppen, Anwendern, Entwicklern, Management etc. durchgeführt werden muss – zur späteren Gewichtung und explizitem, gewichteten Einbeziehen bei weiteren Entscheidungen wie zum Beispiel im Schritt der Bewertung von zu entwickelnden Produkteigenschaften.

Die Merkmale der Methode sind in Abb.1 zusammenfassend dargestellt

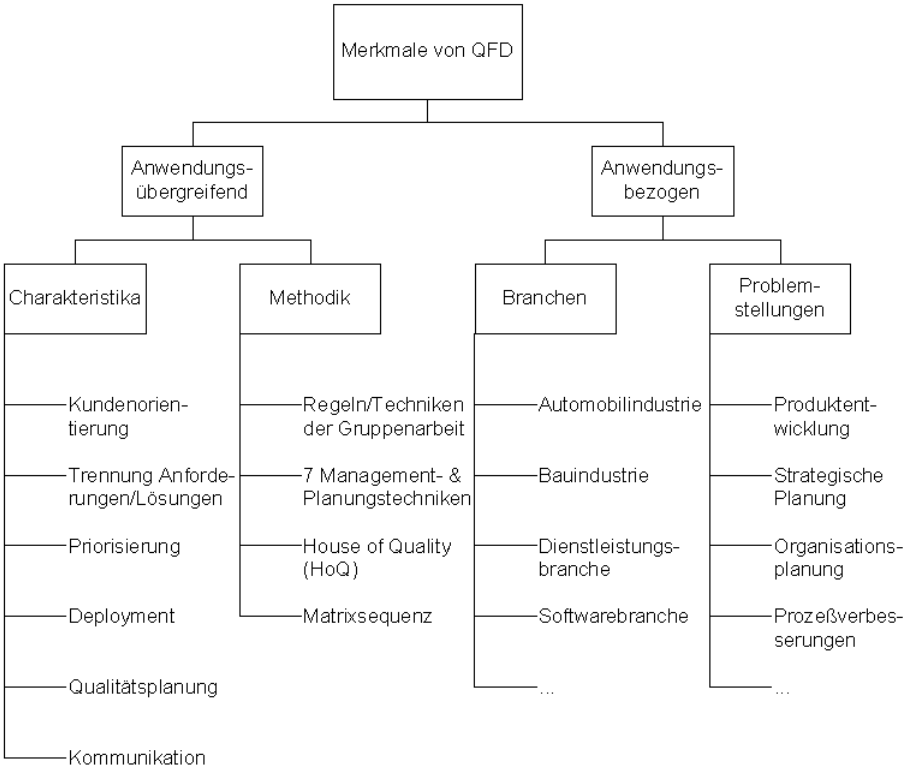


Abbildung 1: Merkmale von QFD (entnommen aus [HSM97], Abb. 2-4, S.37)

4 Die Anwendung von QFD zur Lieferantenauswahl

Quality Function Deployment (QFD) wurde bereits mehrfach in der IT-Industrie angewendet, um wirkliche Kundenbedürfnisse – und nicht beispielsweise nur Rufe nach den neuesten technischen Features – zu identifizieren und im Produkt zu berücksichtigen [CML02]. So wurde beispielsweise mittels QFD bei der Entwicklung des Terminkalenders in SAP R/3 Anforderungen von Lösungsvorschlägen strikt getrennt.

In diesem Abschnitt wird erläutert, wie QFD verwendet wird, um den bestgeeigneten externen IT-Partner an sich zu finden.

Bei Anwendung von QFD können Standard-Kriterienkataloge die, wie deren Name schon sagt, nur undifferenzierte und nicht speziell fallbezogene Standards enthalten, nur äußerst schwierig und mittels eines Systembruchs zur Anwendung kommen. QFD

erzwingt die Entwicklung und Berücksichtigung fallbezogener Kriterien. Dies ist insbesondere im vorliegenden Fall einer Lieferantenauswahl entscheidend; ohne QFD könnte bei der (Vor-)Auswahl externer IT-Partnerfirmen ein unreflektierter Einsatz von zum Beispiel nur Unternehmenskennzahlen wie z.B. Umsatz etc. zum Einsatz kommen – und die entscheidende Eignung der Partnerfirmen für diese Produktentwicklung in den Hintergrund gedrängt werden. Ein Partner mit bestgeeigneter, spezieller technischer Expertise könnte zum Beispiel durch ein hier völlig irrelevantes Standardkriterium (z.B. mehr als zehn einschlägige Referenzen) durch eine Vorauswahl fallen – falls diese Vorauswahl nur stur nach Standard-Einkaufsabteilungs-Kriterien oder Standard-Kriterien einer öffentlichen Ausschreibung getätigt wird – und nicht bereits hier schon die QFD Methode der Einbeziehung/Teambildung aller Betroffenen mitberücksichtigt wird.

Im ersten Schritt werden Entwicklungsziele, funktionelle und technische Anforderungen per QFD verifiziert und gewichtet [HSM00], dann werden Anforderungen über die potentiellen externen IT-Partner gesammelt, sowohl in ihrer Rolle als Unternehmung an sich, als auch betreffend des späteren Supports durch diesen IT-Partner. Insgesamt wurden im IT-Projekt der Entwicklung des Software-Tools 'Chefplan-Online' die folgenden Gruppen von Anforderungen erkannt und priorisiert:

- Funktionelle Anforderungen: Von buchhalterischen und Controlling-Anforderungen über das Benchmarking und die Vergleichsmöglichkeiten innerhalb einer Branche bis hin zur Handhabung der Dateien, z.B. Import-/Export und Druckfunktionen

- Technische Anforderungen: Qualitätsattribute definiert nach ISO-9126 [ISO], aber auch Mindestanforderungen bezüglich der Hardware und Betriebssystemumgebung

- Anforderungen an den IT-Partner an sich: Erfahrungen mit internetbasierten Applikationen, ASP, Referenzen, Unternehmensgrösse, Produkthaftung, Garantien etc.

- Supportanforderungen: Servicezeiten, Dokumentation, Online-Hilfe, Schulung etc.

Diese Anforderungen werden verglichen und gemessen mit den Eigenschaften der potentiellen IT-Partner sowie mit den vorgeschlagenen Realisierungsmöglichkeiten. Hierzu wird eine erste Matrix gebildet, welche die Anforderungen und die vorgeschlagenen Lösungsmöglichkeiten beinhaltet. Im nächsten Schritt werden die möglichen, konkret in Frage kommenden externen IT-Partner untersucht, d.h. die Produktspezifika stehen fest und gehen unabdingbar in die Lieferanten(vor-)auswahl ein. Standard-Kriterienkataloge können zwar auch mit-eingehen, jedoch eben nur 'auch' – und auch nur gewichtet und auch nur in Relation gesetzt zu speziellen Kriterien.
Es werden die vorhandenen, gewichteten Anforderungen und Kriterien dazu verwendet, bereits hier eine Vorauswahl der in Frage kommenden Lieferanten vorzunehmen. Da die Ausschreibungsunterlagen auf eindeutigen, untereinander nicht korrelierten Anforderungen bestehen, hat jede Anforderung bereits eine Gewichtung. Diese Gewichtung wird multipliziert mit dem Erfüllungsgrad jedes Lieferanten. Für jeden potentiellen Lieferanten kann so seine spezifische Summe gebildet werden. Um diese Vorselektion zu verifizieren, wird dann jedem der vorausgewählten Lieferanten eine weitere Ausschreibung bezüglich der Preise usw. zugeleitet. Die dann letztendlich in Frage kommenden Lieferanten werden schließlich zu einem sogenannten 'Beauty Contest' eingeladen. Dort kann jeder Lieferant der Endauswahl seine Lösung etc. präsentieren

und weitergehende Fragen beantworten. Die im 'Beauty Contest' gesammelten Informationen und Eindrücke werden mit zur Schlussevaluation und zur Bestimmung des bestmöglich geeigneten externen IT-Partners verwendet.

Zur Fokussierung und Priorisierung stellt QFD immer die Frage nach dem 'Warum ?'. Eine Anforderungsermittlung und –bewertung mittels QFD hilft, strikt zwischen wirklichen Anforderungen und nur vorgeschlagenen Lösungen zu unterscheiden. So ist, zum Beispiel, eine beispielsweise geforderte Produkteigenschaft, dass eine Software unmodifizierbare Ausdrucke erstellen kann, eine Anforderung. Hingegen wäre der Ruf danach, dass eine Software .pdf-s erstellen kann, nur ein Lösungsvorschlag. Anforderungen sind beständig, Lösungsvorschläge hingegen oft nur Stand der Technik. Ein weiteres Beispiel: Zu betrachten ist die Qualitätssicherung eines Lieferanten: Ist ein ISO 9000-Zertifikat ein Kriterium ? Nein – auch dies wäre nur ein Lösungsvorschlag.

Abb. 2 zeigt den Überblick über die Schritte einer Lieferantenauswahl mittels QFD.

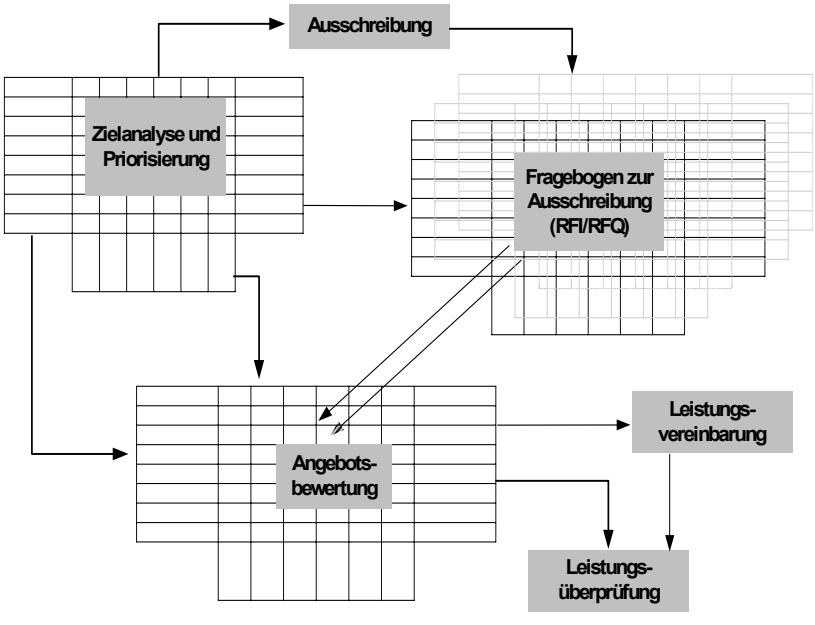


Abbildung 2: Überblick über den QFD Gesamtprozess

5 Fallstudie: 'Chefplan Online'

Das deutsche Beratungsunternehmen BBE Handelsberatung GmbH [http://www.bbe.de] ist spezialisiert auf die Beratung von Einzelhandelsgeschäften. Eine der angebotenen Software-Anwendungen von BBE ist Excel Chefplan, ein Controlling-Tool zur einschlägigen Unterstützung der Entscheidungsträger und zur strategischen Unternehmensplanung. Es basiert als Makro auf Microsoft Excel. Ungefähr 2000 Einzelhandelsgeschäfte setzen Excel Chefplan ein. Bestärkt durch die positiven Rückmeldungen vieler Kunden, entschied BBE eine Internet-basierte Version von Excel Chefplan zu entwickeln und anzubieten: Chefplan Online.

Chefplan Online wird den Anwendern von einem Application Service Provider (ASP)* über das Internet angeboten. Der Anwender ruft das Programm in seinem Browser auf, die Application an sich läuft auf einem Server beim ASP Provider. Diese internetbasierte Version hat die Vorzüge, dass Daten, welche ein Anwender eingibt, statistisch analysiert werden können. Die Analysen können dann als beispielsweise Branchenkennzahlen und 'Benchmarks' allen Anwendern zur Kenntnis gebracht werden. Hierdurch kann der einzelne Anwender sich mit Durchschnitten und dem jeweiligen Optimum vergleichen. Auch können Programmupdates bei ASP-Software sehr einfach durchgeführt werden – was hingegen bei unzähligen, beim Anwender lokal installierten Excel Makros nur äußerst aufwendig durchgeführt werden könnte. Und letztendlich generiert Chefplan Online durch seine monatlichen Benutzungsgebühren regelmässige Rückflüsse.

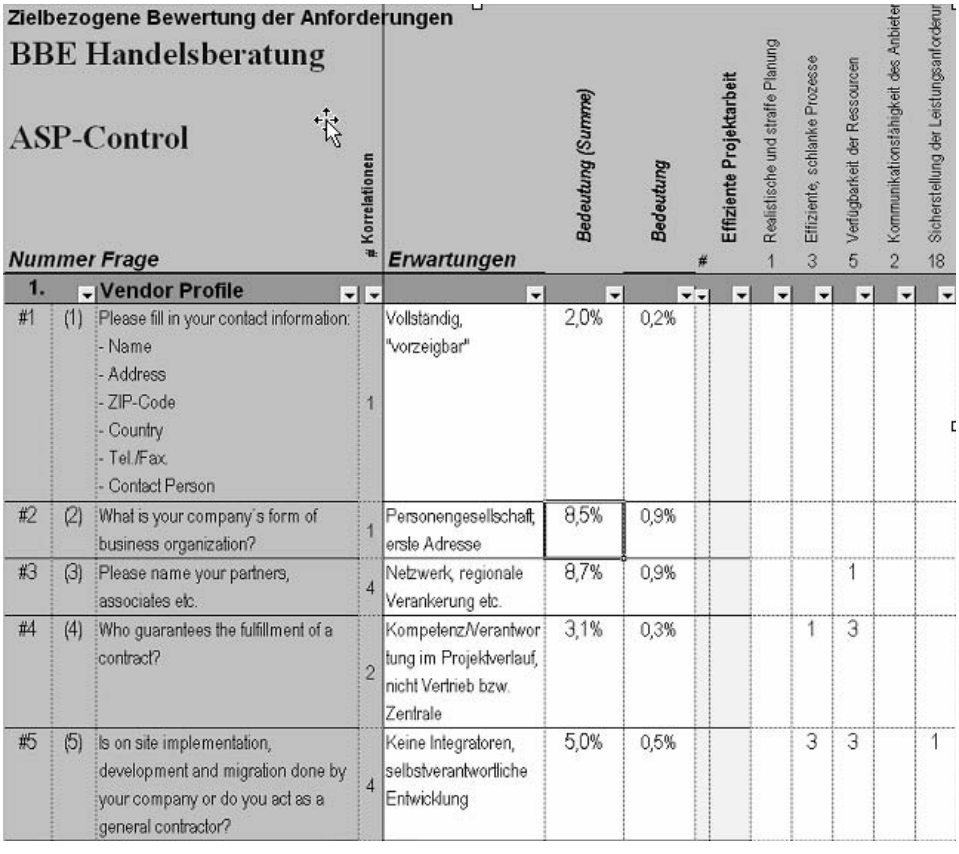
Das Problem war, dass BBE weder in der Softwareentwicklung, noch als Betreiber von Internetanwendungen fundierte Erfahrungen oder Kapazitäten hatte. Folglich wurde entschieden, dass Chefplan Online extern entwickelt und gehostet werden sollte. Um sicherzugehen, dass die Entscheidung für einen externen Partner fundiert und bestmöglich ist, entschied BBE, die Co-Autoren dieses Beitrags mit der Auswahl des externen IT-Partners per QFD zu beauftragen.

Man ging zwar davon aus, dass eine Portierung der Funktionalitäten von Excel Chefpan, plus die Entwicklung des zusätzlichen Benchmarkings in der Online-Version, genügt. Jedoch war BBE sehr positiv offen für weitere Funktionalitäten oder Erweiterungen. Durch die Verwendung von QFD in der Anforderungsanalyse war die Bestandsaufnahme der Funktionalitäten plus die Ergänzung durch Erweiterungsideen, sowie dann alle umfassenden Gewichtungen und Priorisierungen problemlos möglich [HSM00].

Abbildung 3 zeigt einen Ausschnitt aus der QFD-Zielanalyse, welche als Basis zur Ausschreibung diente.

---

* ASP: "Dienstleistungsanbieter ... bei denen Anwendungsprogramme über das Internet durch die Anwender für eine bestimmte Zeit gemietet werden können" [Gab75]



Zielbezogene Bewertung der Anforderungen

BBE Handelsberatung

ASP-Control

Nummer		Frage	# Korrelationen	Erwartungen	Bedeutung (Summe)	Bedeutung	#	Effiziente Projektarbeit	Realistische und straffe Planung	Effiziente, schlanke Prozesse	Verfügbarkeit der Ressourcen	Kommunikationsfähigkeit des Anbieter	Sicherstellung der Leistungsanforderur
									1	3	5	2	18
1.		**Vendor Profile**											
#1	(1)	Please fill in your contact information: - Name - Address - ZIP-Code - Country - Tel./Fax - Contact Person	1	Vollständig, "vorzeigbar"	2,0%	0,2%							
#2	(2)	What is your company's form of business organization?	1	Personengesellschaft, erste Adresse	8,5%	0,9%							
#3	(3)	Please name your partners, associates etc.	4	Netzwerk, regionale Verankerung etc.	8,7%	0,9%					1		
#4	(4)	Who guarantees the fulfillment of a contract?	2	Kompetenz/Verantwortung im Projektverlauf, nicht Vertrieb bzw. Zentrale	3,1%	0,3%				1	3		
#5	(5)	Is on site implementation, development and migration done by your company or do you act as a general contractor?	4	Keine Integratoren, selbstverantwortliche Entwicklung	5,0%	0,5%				3	3		1

Abbildung 3: QFD Zielanalyse

In der Vorauswahl wurden sieben potentielle Lieferanten ausgewählt. Diesen ging dann die erste Ausschreibung ('Request for Information', RfI) zu. Diese erste Ausschreibung bestand aus ungefähr 200 Fragen. Jede Frage korrelierte direkt mit einer Anforderung der abgeschlossenen Anforderungsanalyse. Beispiele dieser Fragen:

- Welche Antwortzeiten können sie für Online-Clients garantieren (>= 2 Sekunden) ?
- Wie werden bei Ihnen Backups durchgeführt (Online, inkrementell, voll, anders) ? Bitte erläutern Sie Ihr Backup-Konzept.
- Bitte benennen Sie Ihre Referenzen, Welche Marktsegmente, wie viele Kunden ?

(Weitere Fragen können auch Abbildung 3 entnommen werden).

Die Analyse der Antworten zu dieser ersten Ausschreibung ist in Abb. 4 dargestellt.

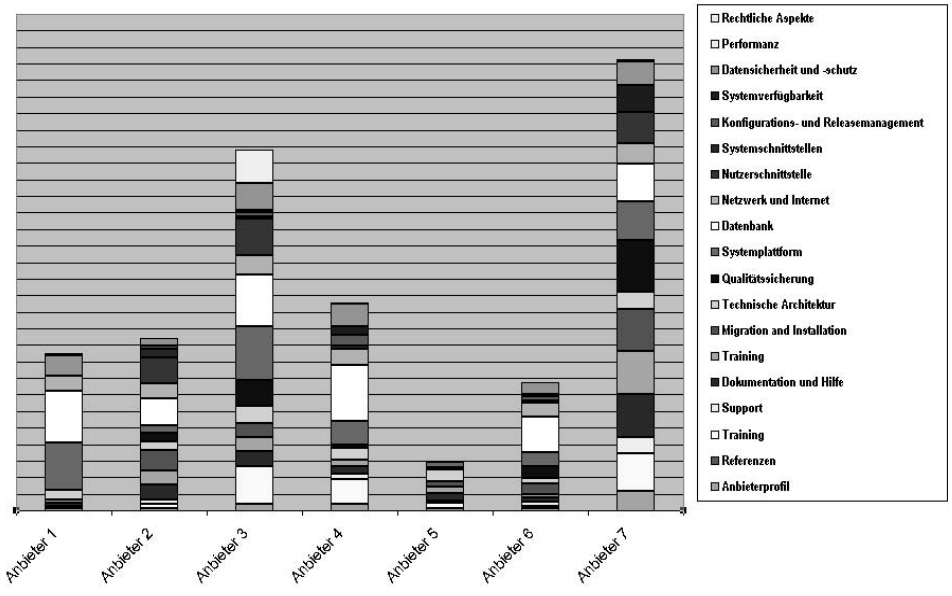


Abbildung 4: Vergleich der aufsummierten Ergebnisse der Anbieter

Im Anschluss an die Analyse der Antworten zur ersten Ausschreibung wurden die potentiellen IT-Partner in einer zweiten Ausschreibung ('Request for Quotations', RfQ) um Preise und kaufmännische/rechtliche Detailklärungen gebeten. Im Anschluss daran wurden die Anbieter zu einem sogenannten 'Beauty Contest' eingeladen. Die Co-Autoren dieses Beitrags evaluierten sowohl die Rückläufe des RfQs, als auch den Eindruck, welchen sie in diesem 'Beauty Contest' gewannen. Dann gaben die Berater Ihre Empfehlung an BBE. BBE wählte den empfohlenen IT-Partner. Alle nun drei Beteiligten (BBE, der externe IT-Partner sowie die Berater) trafen sich zu weiteren Workshops, um dort die endgültige Programmspezifikation abzuschliessen. Das nun gebildete Projektteam war zusammengesetzt aus fachlichen (d.h. was die Funktionalität betrifft) Experten von BBE, die Funktionen und Hintergründe der bisherigen Excel-Realisierung und die Marktanforderungen waren diesen bestens bekannt. Zu diesen fachlichen Experten kamen nun hinzu die Techniker des externen Dienstleisters, die Entwicklung von ASP-Software wurde hier seit Jahren schon sehr oft erfolgreich durchgeführt. Als dritte 'Partei' im Projektteam wurden die Co-Autoren dieses Beitrags damit betreut, die Pflichtenhefterstellung gemäss QFD Methoden zu begleiten: Eine Protokollisierung, Historisierung und stets nachvollziehbare Begründung von Entscheidungen in der Entwicklung war hiermit sichergestellt, die Workshops folgten dem in [HSM00] beschriebenen Vorgehen.

Der Dienstleister entwickelte Chefplan Online und hostet die Applikation in Kooperation mit einem Rechenzentrumsbetreiber. BBE vermarktet die Software (teilweise in Zusammenarbeit mit Handelsverbundgruppen).
Ein Testzugang zu Chefplan Online findet sich unter: http://www.fit-for-business.org.

6 Fazit

QFD erwies sich als überaus geeignete Methode, die Entscheidung für einen externen IT-Dienstleister verifizierbar, nachvollziehbar und belegbar zu dokumentieren. Ebenso wie beispielsweise im Requirements Engineering [HSM00], belegen die erstellten Matrizen getroffene Entscheidungen, Gründe sowie die hierzu führenden Faktoren. Bei neu hinzukommenden Informationen, z.B. zusätzlicher Lieferant oder neue/modifizierte Anforderungen/Fakten, können die hierdurch nun beeinflussten Ergebnisse wieder leicht gezogen werden: Bestehende Matrizen werden einfach erweitert. Die Objektivität ist gewährleistet und nichts kann unbeabsichtigt unberücksichtigt bleiben oder verloren gehen.

Das Besondere bei der Lieferantenauswahl mittels QFD ist, dass methodenbedingt sämtliche relevante Kriterien bei der Lieferantenauswahl miteinbezogen und miteinander (bzw. gegeneinander) ausgewertet werden können. Nicht nur fachliche oder technische Anforderungen können miteinander gewichtet werden, sondern insbesondere Projektspezifika und auch Anforderungen bezüglich der auszuwählenden Fremdfirma an sich: Solidität, Arbeitsweise, Qualitätssicherungsverfahren, rechtliche Rahmenbedingungen, Haftung, Zukunftsperspektiven und Investitionssicherheit – um nur einige zu nennen. Selbst sogenannte 'Soft-Factors' [UNC04] (Subjektives wie z.B. Eindrücke oder Zu-/Vertrauen) können, falls gewünscht, zusammen mit 'objektiven' Kriterien explizit und gewichtet in Relation gesetzt, methodisch dokumentationssicher mitausgewertet werden.

Literaturverzeichnis

[CML02] *Chan, L.-W. and M.-L.Wu*: "Quality function deployment - A literature review." In: European Journal of Operational Research 143 (2002) 463–497

[Dic66] *G. W. Dickson*: "An analysis of vendor selection systems and decisions"; J. Purchas., vol. 2, no. 1, pp. 5–17, 1966

[DYY04] *Dan Wang; Yezhuang Tian; Yunquan Hu*: "Empirical study of supplier selection strategies across the supply chain management in manufacturing companies**"**; Engineering Management Conference, 2004. Proceedings. 2004 IEEE International, Volume 1, 18-21 Oct. 2004 Page(s):85 - 89 Vol.1

[ESS01] *Eon-Kyung Lee; Sungdo Ha; Sheung-Kown Kim*: "Supplier selection and management system considering relationships in supply chain management"; Engineering Management, IEEE Transactions , Vol. 48, Issue 3, Aug.2001 pp 307 – 318

[Gab75] Gablers Wirtschaftslexikon, Band: 1 A – K, 9. Aufl. - Wiesbaden: Gabler, 1975

[G0B98] *S. H. Ghodsypour and C. O'Brien*: "A decision support system for supplier selection using an integrated analytic hierarchy process and linear programming"; Int'l Journal of Production Economics, Vol. 56-57 (1998), pp. 199-212

[HJY05] *Hua-Li Sun; Jian-Ying Xie; Yao-Feng Xue*: "An SVM-based model for supplier selection using fuzzy and pairwise comparison"; Machine Learning and Cybernetics, 2005. Proceedings of 2005 International Conference on, Volume 6, 18-21 Aug. 2005 Page(s):3629 - 3633 Vol. 6

[HSM97] *Herzwurm, Georg; Schockert, Sixten; Mellis, Werner*: Qualitätssoftware durch Kundenorientierung. Die Methode Quality Function Deployment (QFD): Grundlagen, Praxis und SAP R/3 Fallbeispiel. Braunschweig/Wiesbaden 1997

[HSM00] *Herzwurm, G.; Schockert, S.; Mellis, W.*: "Joint Requirements Engineering - QFD for Rapid Customer-Focused Software and Internet-Development"; Vieweg, Wiesbaden, 2000

[ISO] *ISO 9126*: ISO (2001). International Standard ISO/IEC 9126. Information technology - Software product evaluation - Quality characteristics and guidelines for their use, Int'l. Organization for Standardization, International Electrotechnical Commission, Geneva

[MAY94] *Mizuno, Shigeru; Akao, Yoji (Hrsg.)*: QFD, the customer-driven approach to quality planning and development; Tokio 1994

[May97] *Eberhard Mayer, Hewlett-Packard, Germany*: "Vendor Selection Process for Outsourcing"; Proceedings of the 3rd Annual International QFD Symposium, Linköping (1997), E2. Case Studies

[UNC04] United Nations Conference on Trade and Development http://www.unctad.org/Templates/Webflyer.asp?docID=4865&intItemID=1397&lang=1

[WCB91] *C. A. Weber, J. R. Current, and W. C. Benton*: "Vendor selection criteria and methods"; Eur. J. Oper. Res., vol. 50, pp. 2–18, 1991

Beratung in der Softwarewartung und -weiterentwicklung

Harry M. Sneed (MPA)

ANECON, GmbH, Wien (Berater)

Universitäten Passau, Regensburg und Koblenz (Lehrbeauftragter)

Abstract: Dieser Beitrag befasst sich mit der Rolle des Beraters in einem Wartungsprojekt. Wartung wird hier definiert als alle Arbeiten an einem System, nachdem es in Produktion gegangen ist. Dazu gehören die Behebung von Fehlern, die Änderung der bestehenden Funktionalität und die Ergänzung zusätzlicher Funktionalität sowie qualitätssteigernde Aktivitäten wie Nachdokumentation, Reengineering und Optimierungsmaßnahmen. In diesem Umfeld ist guter Rat besonders wertvoll. Ein Berater hat hier viele Möglichkeiten sowohl den Prozess als auch das Produkt zu beeinflussen, muss aber besonders viel Wissen und Erfahrung mitbringen. Es geht in dem Beitrag darum, die potenziellen Aufgaben eines Beraters in der Software-Wartung aufzuzeichnen und zu schildern, wie sie wahrgenommen werden können. Der Autor hat schon mehr als 20 Jahre Erfahrung auf diesem Gebiet. Seine Vorschläge für andere Berater spiegeln diese Erfahrung wider. Keywords: Software-Wartung, Softwareevolution, Change-Management, Fehler-Management, Release-Management, Configuration-Management, Reverse- und Re-Engineering, Software-Messung.

1. Das Zielgebiet – Softwarewartung und -weiterentwicklung

Eigentlich ist der Begriff Software-Wartung eine Fehlübersetzung des englischen „Maintenance“. Besser wäre der Begriff „Software-Erhaltung“, weil Erhaltung eher den Kern der Sache trifft, aber was sich einmal etabliert hat, ist schwer zu ändern. Darum wird der Begriff Softwarewartung auch hier verwendet, um alle Aktivitäten nach dem ersten produktiven Einsatz eines Systems zu bezeichnen [MO83]. Diese Aktivitäten sind äußerst vielfältig. Was sie zusammenhält, ist allein die Tatsache, dass sie sich alle auf das gleiche Objekt beziehen, nämlich das Anwendungssystem bzw. das Produkt. Somit sind aus der Sicht der Objektorientierung das Anwendungssystem als das Objekt und die diversen Aktivitäten der Softwarewartung als die Methoden zu betrachten. Die Attribute des Produktes sind die einzelnen Software-Elemente – die Dokumente, Spezifikationen, Source-Dateien, Testfälle und Fehlerberichte. Sie sind alle Gegenstand der Wartungsaktivitäten. Die Methoden sind u.a. die Analyse, Korrektur, Änderung, Optimierung, Reengineering, Reverse Engineering und Kapselung der Software-Komponente. [LS80]

Das Zielgebiet ist wegen der Undurchsichtigkeit der existierenden Software-Elemente sowie der vielen möglichen Aktivitäten um einiges diffiziler als das der Software-Entwicklung. In der Software-Entwicklung steht nur die momentane Version an. In der Software-Wartung gibt es oft viele Versionen, die gleichzeitig gepflegt und weiterentwickelt werden müssen. Während die nächste Version entwickelt wird, muss die letzte Version korrigiert und nachgebessert werden. Die Probleme, die mit bisherigen Versionen vorkommen, können auch nicht vorausgesagt werden. Wenn sie auftreten, muss sofort darauf reagiert werden. Insofern ist die Software-Wartung Ereignis-gesteuert im Gegensatz zur Entwicklung, die nach einem Plan abläuft. Das macht die Organisation der Wartungsarbeit besonders schwierig und unplanbar. Lientz und Swanson haben die Wartungsaktivitäten in die vier Kategorien eingeteilt: corrective, adaptive, enhansive und preventive maintenance. [Ra00]

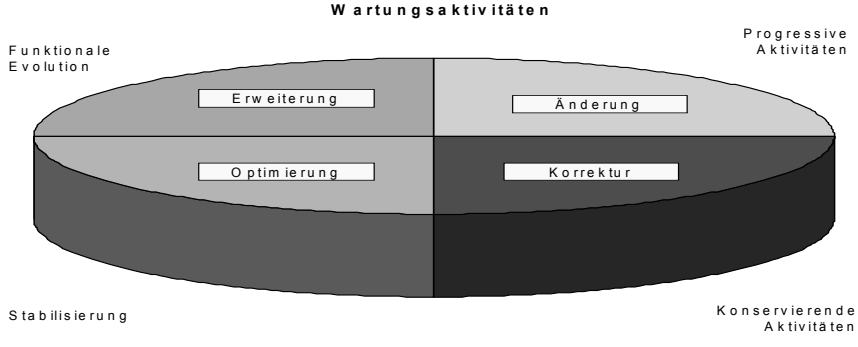


Abb.1: Wartungskategorien nach Lientz und Swanson

2. Die Diagnose der Wartungsprobleme

Wenn Berater wegen Problemen mit der Wartung bestehender Systeme gerufen werden, könnte das eine der folgenden Ursachen haben:

- Der Wartungsbetrieb ist chaotisch organisiert, es gibt keine definierten Wartungsprozesse
- Die zu wartende Software ist sehr wartungsunfreundlich

- Das Wartungspersonal ist überfordert
- Die erforderliche Information für eine systematische Wartung fehlt
- Die neuen Releases werden nicht ausreichend getestet [Sh97]

Nicht selten treten diese Faktoren in Kombination miteinander auf. Die erste Aufgabe des Beraters ist es daher, die Ursachen der Probleme festzustellen. Er muss wie ein Arzt den Patient, bzw. hier den Wartungsbetrieb, diagnostizieren. Dies geschieht am besten über Interviews mit den betroffenen Personen. Dazu gehören neben dem Wartungspersonal auch die zuständigen IT-Manager und die Endanwender in der Fachabteilung. Ein guter Berater wird einen Bewertungsbogen haben, den seine Gesprächspartner ausfüllen können. Bewertet werden u.a.

- die Dauer der Änderungen und Korrekturen,
- die Häufigkeit und Schwere der Softwarefehler,
- die Kosten neuer Releases,
- die Abstände zwischen Releases,
- das Verhältnis mit dem Wartungspersonal,
- die Kontinuität des Wartungsbetriebes [Sn96]

Daraus wird ersichtlich, welche Probleme als vorrangig angesehen werden, z.B. Fehlerkorrekturen und Änderungen dauern viel zu lange, Erweiterungen sind viel zu teuer, neue Releases stecken voller Fehler, Releaseabstände sind zu lange, das Wartungspersonal ist unkooperativ, es gibt eine hohe Fluktuation beim Wartungspersonal. Sind die Probleme einmal erkannt, kann der Berater beginnen, sie den möglichen Ursachen zuzuordnen. Oft wird ein Problem mehrere Ursachen haben. Zum Beispiel können zu hohe Kosten für neue Releases das Ergebnis mangelhafter Prozesse, fehlender Werkzeuge und/oder überfordertes Personal sein.

Jedenfalls muss es dem erfahrenen Berater gelingen, in kurzer Zeit die Hauptursachen der Wartungsprobleme zu identifizieren. Danach kann der Berater beginnen, Rezepte zu schreiben. Falls die Wartungsabwicklung zu chaotisch ist, schreibt er einen neuen Prozess vor. Wenn der Prozess nur Mängel aufweist, schreibt er Verbesserungen vor. Sollte die Software sich als total wartungsunfreundlich erweisen, schreibt er ein Reengineering-Projekt vor. In dem Fall, dass Information über das System fehlt, schreibt er eine Nachdokumention gekoppelt mit dem Aufbau eines Repository vor. Ist das Wartungspersonal überfordert, schreibt er Schulungsmaßnahmen vor. Bei übermäßiger Fehlerhaftigkeit in den neuen Releases kann er schließlich ein Regressionstestverfahren vorschreiben. Es gehört dazu, auch die Kosten für die Umsetzung der Verbesserungsmaßnahmen zu schätzen und alles anschließend seinem Auftraggeber zu präsentieren.

Damit ist der erste Teil des Beratungsauftrages abgeschlossen. Er hat die Probleme aufgedeckt und dokumentiert. Er hat die Ursachen der Probleme ermittelt und er hat alternative Vorschläge gemacht, wie diese Ursachen zu beseitigen sind. Zu guter Letzt hat er berechnet, was die Umsetzung der Vorschläge kosten würde. Jetzt ist der Kunde aufgerufen zu entscheiden, welche Maßnahmen er wann einleiten will. Damit beginnt die zweite Phase der Beratungsarbeit – die Umsetzung.

3. Die Rolle eines Beraters bei der Umsetzung der Verbesserungsmaßnahmen

Es fragt sich, was ein Berater auf diesem schwierigen Gebiet nutzbringend anzubieten hat. Für die Planung der Releases ist der Produktmanager, für die Verwaltung der Softwareelemente der Configuration Manager, für die Korrektur und Fortschreibung der Software der zuständige Entwickler, für den Regressionstest der neuen Versionen der Testmanager und für die Auslieferung der neuen Versionen der Release Manager zuständig. Somit sind die Rollen in einem ordentlichen Wartungsbetrieb alle schon festgelegt und zugeteilt. Was könnte ein externer Berater in so einer Situation noch einbringen?

Hier sind zwei verschiedene Situationen zu unterscheiden. In der ersten Situation gibt es noch keinen eingefahrenen Wartungsprozess. Die Rollen sind zwar verteilt, aber den Rolleninhabern ist nicht klar, was sie leisten sollten und in welcher Reihenfolge. In dieser Situation hat der Berater die Aufgabe, die Wartungs- und Weiterentwicklungsprozesse selber zu definieren. Damit werden auch die Aufgaben der einzelnen Rolleninhaber festgelegt. Diese Aufgabe dürfte nicht so schwer sein. Es gibt mittlerweile einige Normen, auf die er sich beziehen kann. Dazu gehört die ANSI/IEEE-1219 [ANSI92] und die ISO Norm 12250 [ISO97] sowie zahlreiche Bücher zum Thema Softwarewartung und Wartungsprozesse. [Sn89] [Le90] [SHT04]

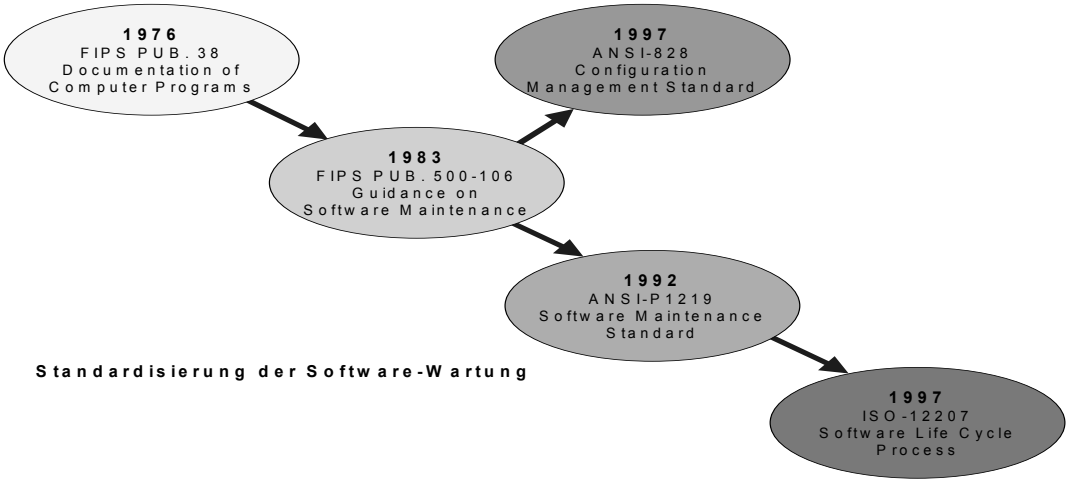


Abb. 2: Entwicklung der Wartungsprozesse

Viel schwieriger ist die Umsetzung des Prozesses. Denn anschließend muss der Berater die von ihm festgelegten Aufgaben den zuständigen Rolleninhabern erklären und sie davon überzeugen, sie auszuführen. Der Berater ist nicht befugt, die Rolleninhaber zu einer bestimmten Arbeitsweise zu zwingen. Er kann nur empfehlen. Es braucht auf der Kundenseite einen starken Produktmanager, um die neue Arbeitsweise durchzusetzen. Der Berater braucht seine Unterstützung für die Umsetzung seiner Ideen. Falls diese starke Führungskraft fehlt, muss der Berater sich auf seine Überzeugungskunst verlassen. In mühseliger Kleinarbeit muss er die Rolleninhaber dazu bewegen, ihre vorgesehene Rolle im neuen Prozess wahrzunehmen. Ein Berater braucht hier sehr viel Geduld und noch mehr Menschenkenntnisse.

In der zweiten Situation gibt es bereits einen eingefahrenen Prozess. Die Wartungsanträge werden schlecht oder recht erledigt und neue Releases ausgeliefert. In diesem Falle ist es die Aufgabe des Beraters, den Prozess zu verbessern. Einen Prozess zu verbessern heißt die Produktivität und/oder die Qualität zu steigern und die Kosten zu senken. Um das nachzuweisen, muss der Berater damit beginnen, den gegenwärtigen Stand der Wartungsproduktivität, der Produktqualität und der Prozesskosten zu messen. Die Kosten sind am leichtesten zu ermitteln. In den meisten Betrieben werden die Personalstunden gegen die Projekte gebucht. Ein Release ist ein Wartungsprojekt. Möglicherweise werden die Kosten sogar gegen die einzelnen Wartungsaufträge gebucht. Dann wüsste man genau, was eine Wartungsmaßnahme - Korrektur, Änderung, Erweiterung, Sanierung, usw. - kostet. Es genügt aber auch, nur die gesamten Releasekosten zu kennen.

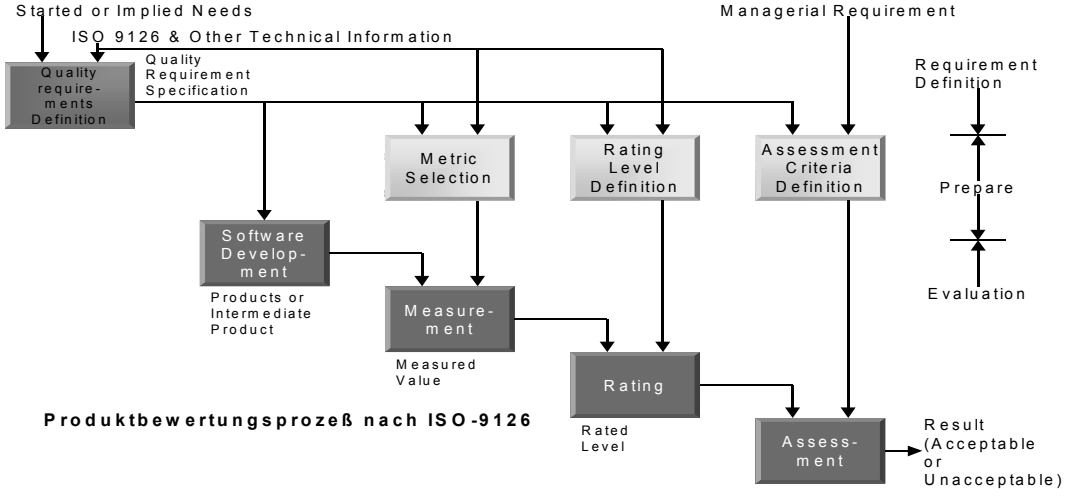


Abb. 3: Software-Produktbewertung

Schwieriger ist es, die Qualität des Produktes und die Produktivität des Projektes zu messen. Software Produktqualität teilt sich nach ISO-9126 in interne und externe Qualitätseigenschaften. [ISO94] Externe Qualitätseigenschaften sind diejenigen, die von außen sichtbar sind – die Zuverlässigkeit, die Effizienz, die Sicherheit und die Benutzbarkeit. Sie sind Maße für das Systemverhalten. Die internen Qualitätseigenschaften sind diejenigen, die nur durch eine Analyse des Codes gemessen werden können. Dazu gehören die Wartbarkeit, die Wiederverwendbarkeit, die Ausbaufähigkeit und die Interoperabilität. Sie sind Maße für die Systemkonstruktion. Die ersten Messungen kann der Berater aus der Betriebsstatistik entnehmen. Fehler, Fehlzeiten, Sicherheitsvorfälle, Performanzkennzahlen, usw. werden von jedem ordentlichen IT-Operationsbetrieb gezählt. Wenn nicht, muss der Berater dafür sorgen, dass sie gezählt werden. Für die Messung der internen Qualitätseigenschaften braucht der Berater Werkzeuge, nämlich statische Analysatoren, mit denen er die verschiedenen Software-Elemente – Datenbanken, Benutzeroberflächen, Programme, Kontrollprozeduren und Testfälle – analysieren, prüfen und messen kann. Mit solchen Werkzeugen kann er recht schnell die Qualität auch von sehr großen Systemen messen. Ohne sie kann er nur zu Stichproben und subjektiven Beurteilungen greifen. Damit wird aber die Analyse teuer und doch nicht zuverlässig, so dass es sich kaum lohnt. Also sind Werkzeuge die Voraussetzung für die Messung der Softwarequalität. Wenn keine vorhanden sind, muss der Berater sie besorgen. [CLO94]

Zur Messung der Wartungsproduktivität sind ebenfalls Werkzeuge erforderlich. In der Regel sind es die gleichen Werkzeuge wie für die Messung der Qualität. Gemessen wird hier die Größe der Software nach verschiedenen Größenmaßen wie Codezeilen, Anweisungen, Object-Points und Function-Points. Es kommt darauf an, die Größe der Veränderungen, bzw. das Delta der Releases, festzustellen und dies dem Aufwand gegenüberzustellen. Daraus geht die Produktivität der Wartungsmannschaft hervor. [Gr87]

Wartungsproduktivität = geänderte & hinzugefügte Anweisungen / Personentage

Neben der Produktivität der Wartung muss der Berater auch die Qualität der Wartung messen. Die Qualität der Wartungsarbeit hängt davon ab, inwiefern es dem Wartungspersonal gelingt die Software zu verändern, ohne weitere Fehler zu verursachen. Hier müssen also die Fehlermeldungen gezählt werden, die nach der ersten Freigabe eingegangen sind. Dabei müssen die Fehler natürlich gewichtet werden. Kritische Fehler wiegen mehr als schwere Fehler und schwere Fehler wiegen mehr als mittlere Fehler. Es kann sein, dass einige dieser Fehler von Anfang an vorhanden waren. Nichtsdestotrotz werden sie gegen den jeweiligen Wartungsvertrag gebucht, weil ein Fehler nur dort gezählt wird, wo er in Erscheinung tritt. Die Zahl der sogenannten Second Level Fehler, also jener Fehler, die als Folge der Wartungsmaßnahmen verursacht sind, werden hier durch die Größe des Systems dividiert, um die Fehlerdichte zu errechnen. Die Fehlerdichte ist letztendlich das Maß für die Qualität der Wartung. [Dr95]

Wartungsqualität = gewichtete Fehler / Anweisungen

Die Fehlerrate ist nur eines von vielen möglichen Maßen für die Qualität des Wartungsbetriebes. Ebenfalls zu messen ist der mittlere Aufwand für die Fehlerkorrektur, der Aufwand pro Änderungsantrag und die Reaktionszeit, bis ein Problem behoben ist. Das Verhältnis der Wartungsorganisation zu den Anwendern ist auch nicht zu vergessen. Es muss über eine Umfrage der Anwender ermittelt werden.

Erst nachdem er diese Vergleichsbasis geschaffen hat, kann der Berater beginnen, Empfehlungen für die Verbesserung des Wartungsprozesses abzugeben. Eine Empfehlung könnte die Einführung eines neuen Wartungswerkzeuges oder die Refaktorierung einiger Programme sein. Für jede empfohlene Maßnahme muss der Berater ein messbares Ziel setzen, d.h. um wie viel Prozent soll diese Maßnahme die Wartungsproduktivität und/oder -qualität steigern. Nach der Umsetzung der Maßnahme muss der Berater dessen Nutzen an Hand der Messwerte nachweisen. Eine signifikante Verbesserung des Wartungsbetriebes wird erst nach Jahren sichtbar. Bis dahin könnte der Berater schon längst woanders sein. Deshalb kommt es hier vor allem darauf an, ein dauerhaftes Messverfahren einzuführen, das der Kunde von sich aus übernehmen und weiter betreiben kann. Der Anwender holt einen Wartungsberater, um seinen Wartungsbetrieb zu verbessern, aber Verbesserungen in der Wartung sind nur langfristig zu erzielen. Dies muss der Berater dem Kunden von Anfang an klar machen. Es gibt keine Wundermittel für die Softwarewartung. [NV00]

Maße für den Wartungsbetrieb

1) ZEIT zur Erfüllung der Wartungsaufträge in Kalendertagen relativ zum Umfang der Wartungsaufträge in betroffenen Code-Anweisungen bzw. Function-Points, Data-Points oder Object-Points.

2) AUFWAND zur Erfüllung der Wartungsaufträge in Personentagen relativ zum Umfang der Wartungsaufträge.

3) FEHLER, die durch die Erfüllung der Wartungsaufträge entstehen relativ zum Umfang der Wartungsaufträge.

4) QUALITÄT und KOMPLEXITÄT der Programme nach der Erfüllung der Wartungsaufträge relativ zur Qualität und Komplexität der Programme vorher.

5) QUALITÄT der Dokumentation bezüglich der Verständlichkeit, der Genauigkeit und der Übereinstimmung mit den Programmen.

6) VERHÄLTNIS der Wartungsprogrammierer zur Software-Größe in LOC, Anweisungen, Function-Points, Data-Points, Object-Points und/oder Komponenten relativ zur jährlichen Änderungsrate in % Code-Zeilen geändert.

7) GRAD DER BENUTZERZUFRIEDENHEIT mit der Software-Wartung bezüglich

- der Reaktionszeit
- der Kosten
- des Nutzens
- der Termintreue
- der Fehlerrate
- der Hilfsbereitschaft
- der Verfügbarkeit

auf einer Skala von 1 bis 10.

4. Software Maintenance und ITIL

Einige Aktivitäten der Softwarewartung wie Change Management, Problem Management, Configuration Management und Release Management werden heutzutage immer mehr aus der Sicht des IT Service Managements betrachtet. [VG05] In der Tat ist der ITIL-Standard aus der Welt der Software-Wartung hervorgegangen. Die Vorschläge für die Organisation der IT-Service-Prozesse basieren auf der langjährigen Arbeit des British Centre for Software Maintenance in Durham, die bei British Telecom umgesetzt und von dort aus zum British National Standard wurden. Was die Erhaltung des Operations- und Service-Betriebs anbetrifft, sind die ITIL-Richtlinien zu berücksichtigen. Da das Management für den Begriff ITIL sensibilisiert ist, ist es sogar von Vorteil, die Optimierung des Wartungsbetriebes unter diesem Vorzeichen zu verkaufen. Zwar wird die Software-Evolution vom ITIL-Standard nicht behandelt, aber die Trennung von Wartung und Weiterentwicklung ist sehr schwierig. Die Überlappung der beiden Themen ist viel zu stark, als dass man sie getrennt einführen könnte. Deshalb empfiehlt es sich, die Einführung von ITIL-Prozessen mit der Verbesserung der Wartungsprozesse zu verbinden, und dazu gehört eben die Weiterentwicklung. Der Software-Wartungsberater soll auf jeden Fall mit ITIL gut vertraut und möglichst ITIL-zertifiziert sein.

5. Vergütungsmöglichkeiten für Beratung auf dem Gebiet der Softwarewartung

Die Vergütung für die erste Phase der Beratung – die Diagnose – muss sich nach dem Aufwand richten. Allerdings sollte der Berater diesen Aufwand vorher vereinbaren. Je nachdem wie groß der Betrieb ist und wie viele Personen involviert sind, schätzt der Berater eine bestimmte Anzahl Tage. Zum Schluss wird er nur für die geleisteten Tage bezahlt, aber er darf die vereinbarte Höchstgrenze nicht überschreiten. Falls er diese Phase innerhalb der vereinbarten Zeit nicht abschließen kann, muss er mit dem Kunden über eine Verlängerung verhandeln. Es kann sein, dass der Kunde damit nicht einverstanden ist. In dem Fall muss der Berater die Diagnose auf eigene Kosten abschließen oder abbrechen.

Mit der Vergütung der zweiten Phase – die Umsetzung - sieht es anders aus. Am besten ist es, wenn sie an dem Erfolg der Maßnahmen gekoppelt ist. Ein Mindestbetrag wird aufgrund der Schätzung des Beraters für seinen Anteil an der Umsetzung der Maßnahmen vereinbart. Darüber hinaus ist der Berater an den Kosten der Ersparnisse beteiligt. Für jeden Euro, der in dem Wartungsbetrieb gespart wird, könnte der Berater einen Anteil von 25 Cent haben. Dies würde die Berater motivieren, nur jene Maßnahmen umzusetzen, die wirklich Ersparnisse bringen. Natürlich müssen hier alle Kostenarten erfasst werden, die Kosten der Fehlerbehebung und die Kosten der neuen Releases sowie die Lost Opportunity Costs aufgrund verzögerter Releases. Es gibt leider auch sehr viele unqualifizierte Softwareberater und es geht darum, die Kunden vor solchen Beratern zu schützen.

6. Zur Qualifikation eines Softwarewartungsberaters

Wer sich als Berater für die Softwarewartung anbietet, braucht eine besondere Qualifikation. Als Erstes muss er sich mit alten Systemen auskennen, denn die meisten Systeme in der Wartung sind alt. Am besten ist es, wenn er viele Jahre selbst Programme entwickelt und gewartet hat. Darüber hinaus soll er mit Wartungsprozessen und Wartungswerkzeugen vertraut sein. Vor allem soll er die internationalen und nationalen Standards kennen. Außerdem muss er mit Software-Metrik umgehen können, weil er nicht weiter kommt, ohne die Wartungsproduktivität und Wartungsqualität zu messen. Dazu braucht er viel Geschick und Geduld im Umgang mit Menschen, da er von den Menschen erfahren muss, wo die Probleme liegen. Schließlich muss er wie alle Berater gut schreiben und präsentieren können.

Es ist offensichtlich, dass nur wenige Menschen als Wartungsberater in Frage kommen. Dies ist aber richtig so. Denn gar keine Beratung ist besser als eine schlechte Beratung. Es kommt darauf an, die Spreu vom Weizen zu trennen. Zu viele Beratungshäuser schicken unqualifizierte Berater ins Feld, die mehr schaden als nutzen. Damit muss Schluss gemacht werden. Es geht hier um Qualität vor Quantität. Auf diesem sehr undurchsichtigen und technisch anspruchsvollen Gebiet der Softwarewartung muss der Kunde mit der Auswahl eines Beraters sehr sorgfältig umgehen.

7. Zusammenfassung

In diesem Beitrag zum Thema Beratung für die Softwarewartung wurde geschildert, um was es bei der Wartungsberatung geht. Softwarewartung ist ein sehr diffuses und unzulänglich definiertes Gebiet. Schon allein der Begriff „Wartung“ selbst ist umstritten. Es geht hier viel mehr um eine System- und Dienstleistungserhaltung. Die Abgrenzung der Wartung von der Entwicklung ist auch schwierig, obwohl Wartung eigene Gesetze hat und einen eigenen Prozess voraussetzt. Dennoch leiden viele Anwenderbetriebe unter der Ineffizienz ihrer Softwareerhaltung und suchen nach Wegen, die Kosten der Wartung zu senken und die Qualität der Wartung zu steigern. Manche greifen nach dem Mittel des Outsourcing. Sie lagern die Verantwortung für die Erhaltung ihrer Altsysteme aus an fremde Partner. Andere versuchen den internen Wartungsbetrieb selbst auf Vordermann zu bringen. Dazu sind sie auf Hilfe von außen angewiesen. Sie brauchen gute Beratung und geeignete Werkzeuge.

Der Berater für Softwarewartung braucht eine besonders hohe Qualifikation. Er muss sowohl die Technik der vorhandenen Systeme als auch die Organisation und den Umgang mit Menschen beherrschen. Seine Arbeit teilt sich in zwei Phasen. Erstens muss er die Probleme mit der Systemwartung untersuchen, um auf die Ursachen zu kommen und Lösungen für deren Beseitigung vorzuschlagen. Zweitens muss er bei der Umsetzung seiner Vorschläge mitwirken. Entweder hilft er, einen neuen Wartungsprozess einzuführen, oder er arbeitet mit an der Verbesserung des bestehenden Prozesses. Im letzteren Fall besteht seine Hauptaufgabe darin, die Produktivität und Qualität vor und nach den Verbesserungsmaßnahmen zu messen. Dazu braucht er spezielle Messwerkzeuge und gute Metrikkenntnisse.

Es soll auf jeden Fall versucht werden, die Kosten der Beratung mit dem Erfolg der empfohlenen Verbesserungsmaßnahmen zu koppeln. Guter Rat soll belohnt werden, schlechter Rat soll bestraft werden. Es obliegt dem Berater, die für den jeweiligen Betrieb geeigneten Maßnahmen zu finden und durchzusetzen. Er soll deshalb an dem damit verbundenen Risiko persönlich beteiligt sein. Dies wird helfen, falsche Berater von diesem äußerst schwierigen und risikobehafteten Geschäft fern zu halten.

8. Referenzen

[ANSI92] ANSI/IEEE Standard 1219/D16, Standard for Software Maintenance, IEEE Press, New York, 1992

[CLO94] Coleman, D., Lowther, B., Oman, P.: “Using Metrics to evaluate Software System Maintainability”, IEEE Computer, August 1994, p. 42-57

[Dr95] Dromey, D.: “A Model for evaluating Software Product Quality”, IEEE Trans. on S.E., Vol. 21, No. 2, Feb., 1995, p. 146-152

[Gr87] Grady, R. B: “Measuring and Managing Software Maintenance”, IEEE Software, Sept. 1987, p.35-49

[ISO94] ISO: ISO/IEC Std. 9126, Software Product Evaluation", International Standards Organization, Geneva, 1994

[ISO97] ISO: ISO/IEC Std: 12207, Standard for Software Life Cycle Management, ISO Press, Geneva, 1997

[Le90] Lehner, F.: Softwarewartung, Hanser Verlag, München, 1990

[LS80] Lientz, B., Swanson, E.B.: Software Maintenance Management, Addison-Wesley, Reading, 1980, p. 105

[MO83] Martin, R.J., Osborne, W.: “Guidance of Software Maintenance”, U.S. National Bureau of Standards, NBS Pub. 500-129, Dec. 1983

[NV00] Niessink, F., van Vliet, H., “Software Maintenance from a Service Perspective”, Journal of Software Maint., Vol. 12, No. 2, April, 2000, p. 103-120.

[Ra00] Ramaswamy, R. : “How to staff business critical Maintenance Projects”, IEEE Software, June, 2000, p. 90-94

[Sh97] Sherer, S.: “Using Risk Analysis to Manage Software Maintenance”, Journal of Software Maint.”, Vol. 9, No. 6, Dec. 1997, p. 345-264

[SHT04] Sneed, H., Hasitschka, M., Teichmann, M.-T.: Software Produktmanagement, dpunkt Verlag, Heidelberg, 2004

[Sn89] Sneed, H.: Softwarewartung, Rudolf Müller Verlag, Köln, 1989

[Sn96] Sneed, H.: “Evaluating the Maintenance Process at the Zurich Life Insurance”, proc. of Int. Conf. on Software Maintenance, IEEE Computer Society Press, Monterey, 1996, p. 217-227

[VG05] Victor, F./ Günther, H.: Optimiertes IT-Management mit ITIL, Vieweg-Bertelsmann Verlag, Wiesbaden, 2005

Projektaudit – methodisch Projekte sanieren

Randolph Kappes

Geschäftsbereich IT-Beratung
sd&m AG
software design & management
Mülheimer Straße 9a
53840 Troisdorf
kappes@sdm.de

Abstract: Die Professionalisierung der Softwareentwicklung schreitet immer weiter fort. Gleichzeitig wachsen die Anforderungen an die Anwendungsentwicklung – insbesondere aufgrund neuer Trends in der Software-Industrie. Gerade innovative Projekte geraten leicht in schwierige Situationen: Sie überschreiten die veranschlagten Zeit- und Kostenrahmen oder liefern nicht die gewünschte Funktionalität. Eine Beratung in Form eines Projektaudits durch IT-Experten, die nicht in das Projekt involviert sind, ermöglicht die Bewertung einer solchen Projektsituation, verbunden mit der unvoreingenommenen Empfehlung der notwendigen Maßnahmen. Jedes Softwareentwicklungsprojekt folgt einem methodischen, strukturierten Vorgehen. Ein methodisch strukturiertes Vorgehen auch im Projektaudit ist neben der fundierten Erfahrung der projektexternen Experten die Basis für eine erfolgreiche Untersuchung. Nur so können die Handlungsempfehlungen zur Sanierung in der zur Verfügung stehenden Zeit geliefert werden. Der Beitrag stellt die Methodik sowie die Erfolgsfaktoren eines Projektaudits vor.

1 Standortbestimmung mithilfe eines Projektaudits

Für einen IT-Manager eine alltägliche Situation: Einer seiner Projektleiter meldet Probleme im Projekt. Ohne interne Kenntnisse der Projektsituation kann der IT-Manager die wirkliche Lage nicht beurteilen und keine Handlungsempfehlungen aussprechen. Auch der Projektleiter ist aufgrund der intensiven operativen Steuerung meist schon betriebsblind und kann die notwendigen Maßnahmen nicht mehr erkennen.

Die Beratung kann einen Beitrag zur Sanierung eines solchen Projektes leisten in Form eines Projektaudits. Ein Projektaudit durch einen unabhängigen Berater liefert kurzfristig eine Standortbestimmung. Die Auditoren müssen dabei eine vertrauensvolle Zusammenarbeit mit der Softwareentwicklung, also den Projektmitarbeitern auf Auftraggeberseite, pflegen. Nur über die Mitarbeiter aus der Softwareentwicklung können die Auditoren in der Kürze der zur Verfügung stehenden Zeit die für die Bewertung erforderlichen Informationen erhalten.

Neben der Zusammenarbeit mit der Softwareentwicklung und der fundierten Erfahrung der Berater ist – genau wie bei der Softwareentwicklung – ein methodisch strukturiertes Vorgehen der wesentliche Erfolgsfaktor für ein Projektaudit. Im Folgenden stellen wir eine Methodik vor, die sich in der Praxis bewährt hat.

2 Vorgehen bei einem Projektaudit

sd&m hat ein Vorgehen entwickelt, um methodisch strukturiert die Situation in einem Projekt zu analysieren und Sanierungsmaßnahmen abzuleiten. Die Erfahrungen sind in diesem Vorgehen verdichtet.

Das Vorgehen gliedert sich in fünf Schritte:

- Schlüssel zum Erfolg ist es, die Ziele bereits im Vorfeld des Audits klar zu formulieren. Nur dies stellt sicher, dass die Fragen des Auftraggebers konkret beantwortet werden.
- Zu Beginn des Projektaudits wird aus diesen Zielen der Untersuchungsgegenstand abgeleitet.
- Im zweiten Schritt werden etablierte Normen ausgewählt, die eine objektive Bewertung des Projekts ermöglichen.
- Die Untersuchung selbst erfolgt hypothesengetrieben. Dieser Ansatz stellt sicher, dass früh Indikatoren über das Audit-Ergebnis vorliegen.
- Im letzten Drittel des Audits werden die Handlungsempfehlungen erarbeitet und auf Kosten und Nutzen bewertet.

2.1 Schlüssel zum Erfolg: Ziele das Audits klar formulieren

Anfragen, ein Projekt zu auditieren, kommen meist, wenn das Topmanagement aufgrund von Problemmeldungen kurzfristig Auskunft über die wirkliche Situation des Projekts fordert. Diese Anfragen sind oft offen formuliert: Mit dem Projektaudit soll eine unabhängige Meinung zur Projektsituation eingeholt werden.

Die in der Regel vage Formulierung des Auftrags an die Auditoren ist darauf zurückzuführen, dass derjenige, der den Auftrag erteilt, meist eine gewisse inhaltliche Distanz zum Projektgeschehen hat. Es ist für ihn schwierig, die im Audit zu untersuchenden Aspekte präzise zu benennen. Es liegt daher in der Verantwortung der Auditoren, die Ziele des Audits im Dialog mit dem Auftraggeber zu präzisieren und festzulegen, welche Fragestellungen zu untersuchen sind.

Mögliche Fragestellungen sind beispielsweise:

- Ist das Projektmanagement und die Projektorganisation adäquat aufgestellt, um weitere Anforderungen der Kunden umsetzen und parallel die Bearbeitung von Weiterentwicklung, Wartung und Support gewährleisten zu können?

- Trägt die technische Architektur der Anwendung, um eine Erweiterung der Datenmengen und der Anzahl Anwender im Rahmen eines Mergers zu verkraften?

- Sind die bereits intern identifizierten Schwachstellen des Projekts richtig erkannt und bewertet? Gibt es noch weitere Schwachstellen?

- Ist der Betrieb der neuen Anwendung wirtschaftlicher als der Betrieb der vorhandenen Systeme?

Ziele und Fragestellungen für das Audit müssen Bestandteil des Audit-Auftrags sein. Nur dies stellt sicher, dass die Fragen des Auftraggebers konkret beantwortet werden. Darüber hinaus kann die Arbeit des Projektteams an der Beantwortung der Fragen und der Definition von adäquaten Maßnahmen in Bezug auf die Ziele gemessen werden.

Die klare Formulierung hilft darüber hinaus dem Auditorenteam, seine Arbeit explizit auf diese Ziele zu fokussieren und somit in kurzer Zeit zu einer Gesamteinschätzung und zu Handlungsempfehlungen zu kommen.

2.2 Untersuchungsgegenstand festlegen

Nachdem die Ziele und Fragestellungen für das Audit mit dem Auftraggeber vereinbart sind, müssen sie durch die Auditoren operationalisiert werden. Dazu wird aus den Zielen und Fragestellungen abgeleitet, welche Aspekte des Projekts untersucht werden müssen. Als Untersuchungsgegenstand kommen dabei DV-technische Aspekte wie Architektur und Qualität des Quellcode ebenso wie Aspekte der Softwareentwicklungs- oder Projektmanagementprozesse oder aber auch Wirtschaftlichkeitsaspekte in Frage. Welcher Untersuchungsgegenstand sich für die jeweilige Fragestellung im Einzelnen ergibt, können die im Audit eingesetzten Experten auf Basis ihrer Erfahrung zügig festlegen (Abbildung 2).

Abbildung 2 Auswahl möglicher Untersuchungsgegenstände eines Projektaudits

Der Einsatz von erfahrenen Beratern, die fundiertes Wissen in den zu bearbeitenden Untersuchungsgegenständen mitbringen, ist ein wesentlicher Erfolgsfaktor für das Audit. Für den Einsatz von Juniorberatern ist in einem Projektaudit kein Platz, da weder die zur Verfügung stehende Zeit noch die Kritikalität der Auditergebnisse eine Einarbeitung oder umfangreiche Recherche zulässt.

2.3 Objektivierung durch Vergleich mit etablierten Normen

Die Untersuchungsgegenstände müssen durch das Projektteam analysiert und anschließend bewertet werden. Im Vorfeld der Untersuchung müssen dafür die Bewertungskriterien festgelegt werden. Für eine möglichst objektive, nachvollziehbare Bewertung hat sich die Nutzung von etablierten Standards bewährt.

Soll beispielsweise das Projektmanagement untersucht werden, dann empfiehlt es sich, als Vergleichsbasis die Projektmanagementprozesse des PMBOK® Guide [PMI04] oder gemäß PRINCE2® [PR05] zu nutzen. Für die Bewertung des Softwareentwicklungsprozesses können CMMI®-SE/SW [CM05] oder unternehmenseigene, nach ISO9001:2000 zertifizierte Qualitätsmanagementsysteme [ISO9001] herangezogen werden. Für die Bewertung von Softwarearchitekturen bieten sich die Nutzung von ISO-Normen wie ISO/IEC 9126 (Framework für die Bewertung von Softwarequalität) [ISO9126] oder Standardarchitekturen wie Quasar (Quality Software Architecture, Standardarchitektur der sd&m AG für betriebliche Informationssysteme) [SI04] an.

Der Untersuchungsgegenstand wird gegen die einzelnen Teilaspekte des jeweiligen Standards gehalten. Es wird festgelegt, welche dieser Teilaspekte zur Beantwortung der Fragestellung beitragen. Dies konkretisiert erneut den Fokus der Untersuchung. Darüber hinaus liefern die Standards meist auch Kriterien, gegen die die Umsetzung im Projekt gespiegelt und bewertet werden kann.

Die Bewertungskriterien, gegen die die Projektsituation bewertet wird und gegen die die Abweichungen von der Zielerreichung gemessen werden, ergeben sich damit aus den aus der ursprünglichen Fragestellung konkret heruntergebrochenen Teilaspekten dieser Standards. Dies bedeutet,

- dass die Projektbewertung klar gegen eine Zielgröße (die definierten Bewertungskriterien) bewertet wird und

- dass sich der Untersuchungsgegenstand aus unterschiedlichsten Bereichen zusammensetzt, wie der Bewertung des Quellcodes, der Bewertung des Anforderungsmanagements bis hin zu Wirtschaftlichkeitsbewertungen.

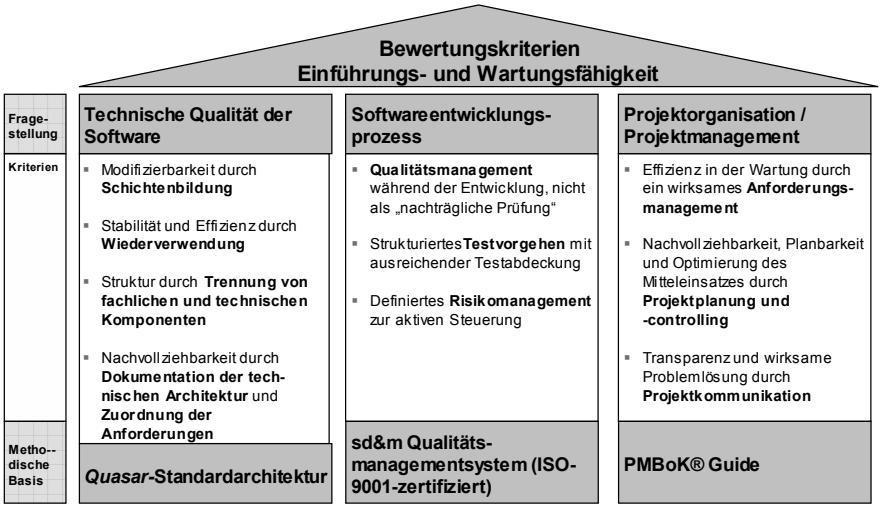


Abbildung 3 Bewertungskriterien ergeben sich aus den Teilaspekten aller Kategorien

Abbildung 3 stellt die Bewertungskriterien der Einführungs- und Wartungsfähigkeit einer Anwendung dar. Diese wurden aus der konkreten Fragestellung, über die Spiegelung gegen die relevanten Standards und durch Festlegung konkreter Kriterien nach den ersten Eindrücken in Gesprächen mit dem Auftraggeber und dem Projektteam festgelegt.

2.4 Hypothesengetriebener Ansatz liefert früh Indikatoren

Als Vorgehensmodell für die Einschätzung der Projektsituation hat sich der hypothesengetriebene Ansatz bewährt. Bei dieser Vorgehensweise stellen die Auditoren bereits zu einem frühen Zeitpunkt im Projektverlauf Hypothesen über die erwartete Projekteinschätzung auf. Experten mit einem fundiert fachlichen Hintergrund können solche Hypothesen erfahrungsgemäß nach wenigen Tagen aus der Sichtung der vorhandenen Dokumente und den ersten Interviews ableiten.

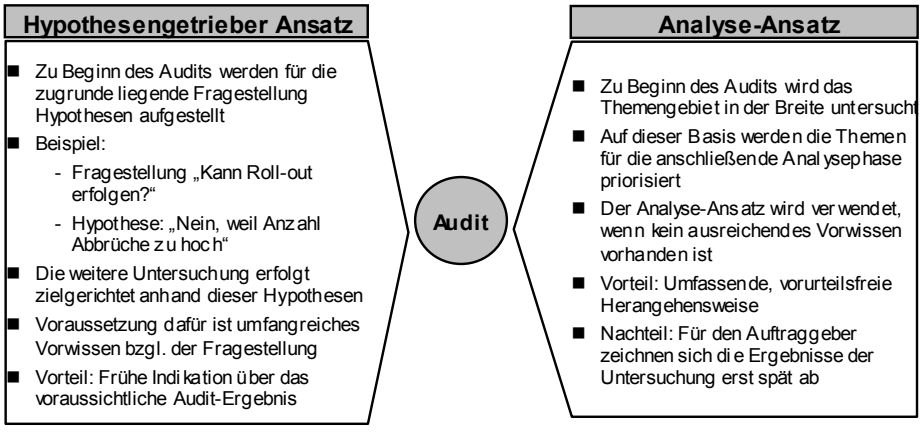


Abbildung 4 Hypothesengetriebener Ansatz und Analyse-Ansatz sind die beiden Grundmuster einer Auditdurchführung

Als nächsten Schritt verifizieren oder widerlegen die Auditoren die aufgestellten Hypothesen. Dazu werden zu jeder Hypothese Schlüsselfragen aufgestellt und im weiteren Projektverlauf geklärt. Diese Schlüsselfragen konkretisieren die Einschätzung und machen die Bewertung greifbar. Bei der Erstellung der Schlüsselfragen helfen wiederum die als Absprungbasis genutzten Standards, da diese teilweise Checklisten enthalten oder einen abprüfbaren Prozess beschreiben.

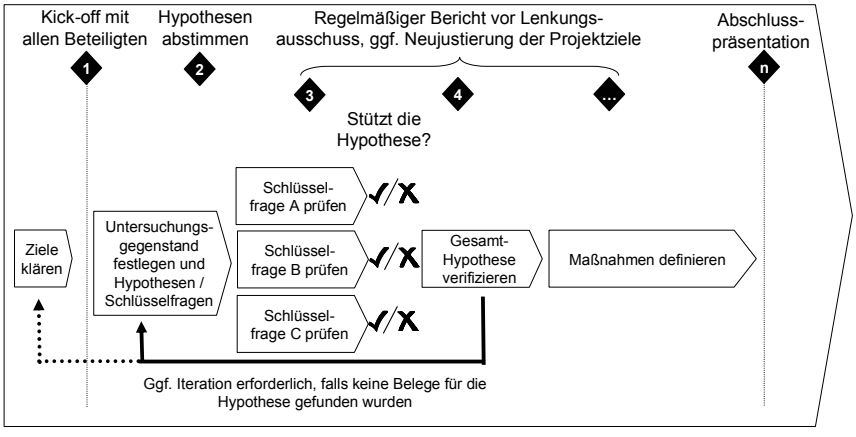


Abbildung 5 Laufzeit eines Audits ist in der Regel ein bis drei Monate, davon entfallen ca. 2/3 der Zeit auf Aufstellen und Verifizieren der Hypothesen und ca. 1/3 auf die Maßnahmen-Definition

Der Vorteil eines hypothesengetriebenen Ansatzes liegt vor allem darin, dass bereits nach wenigen Tagen eine Einschätzung über das voraussichtliche Audit-Ergebnis vorliegt. Hiermit können dem Auftraggeber frühzeitig vorsichtige Signale zur Projektsituation gegeben werden. Die Gesamteinschätzung und der Umgang mit den Projektergebnissen werden so bereits parallel zur weiteren Untersuchung zwischen Auftraggeber und Auditoren besprochen. Gegebenenfalls wird in diesem Rahmen auch der Auftrag nachjustiert. Diese Diskussion hilft, die Handlungsempfehlungen zielgerichtet auf die strategischen Überlegungen des Auftragnehmers zu entwickeln.

2.5 Handlungsempfehlungen auf Kosten und Nutzen analysieren

Ein Auditergebnis enthält immer Handlungsempfehlungen für den Auftraggeber. Diese bestehen neben der generellen Einschätzung des Projekts und einer Lösungsstrategie zur Erreichung eines stabilen Projektzustands aus einer Liste der vorgeschlagenen Einzelmaßnahmen. Die Einzelmaßnahmen werden so definiert, dass ihre Umsetzung hilft, den definierten Zielzustand – gemäß den Bewertungskriterien für den jeweiligen Untersuchungsgegenstand – zu erreichen.

Die Definition der Einzelmaßnahmen darf dabei nicht losgelöst vom laufenden Projekt erfolgen, da insbesondere die benötigten Mitarbeiterressourcen meist aus dem laufenden Projekt gestellt werden müssen. Die Maßnahmen werden daher analog zu jeder anderen Aktivität im Projekt geplant. Dies beinhaltet Aufwandschätzung, Ressourcenzuordnung, Festlegung von Verantwortlichkeiten und Endterminen. Darüber hinaus muss die Integration in das laufende Projekt geplant und sichergestellt werden.

Da keine Maßnahme aus reinem Selbstzweck erfolgen darf, empfiehlt sich eine Nutzenbewertung der Maßnahmen, um ihren Nutzen den durch die Umsetzung entstehenden Kosten gegenüberzustellen. Über eine Kosten-Nutzen-Betrachtung kann so der Wertbeitrag für das Projekt gerechnet werden.

3 Die Arbeit beginnt mit dem Ende des Audits

Mit dem Abschluss der Projektaudits ist das Projekt jedoch noch nicht saniert. Erst die Umsetzung der Maßnahmen stellt dies sicher. Dies hat vor allem dann Aussichten auf Erfolg, wenn das Topmanagement über das Projektaudit hinaus das Projekt begleitet und der Wille zur Veränderung im Projekt und im Projektumfeld vorhanden ist.

Des Weiteren empfiehlt es sich, im Nachgang eines Projektaudits auch die Wirtschaftlichkeitsbetrachtung des Gesamtprojekts auf Basis der neuen Erkenntnisse zu überarbeiten. Dabei muss auch die Frage gestellt werden, ob sich das Projekt für das Unternehmen in der geplanten Form noch rechnet.

Literaturverzeichnis

[CM05] CMMI-SE/SW, http://www.sei.cmu.edu/cmmi, Stand: 22. August 2005

[ISO9001] ISO9001:2000, http://www.iso.org, Stand: 22. August 2005

[ISO9126] ISO9126, http://www.iso.org, Stand: 22.August 2005

[PMI04] Project Management Institute, A Guide to the Project Management Body of Knowledge, A (PMBOK® Guide), (Third Edition) Projekt Management Institure, Ort 2004, ISBN 193069945X

[PR05] PRINCE2, http://www.ogc.gov.uk/prince2, Stand: 22. August 2005

[SI04] Siedersleben, Johannes: Moderne Softwarearchitektur: Umsichtig planen, robust bauen mit Quasar. (1. Auflage) dpunkt-Verlag, 2004, ISBN 3-89864-292-5

Projektführung in Kundenprojekten

Nachlässigkeiten-Irrtümer-Methoden

Hubert Surrer

sd&m AG software design & management
Carl-Wery-Str. 42
81739 München
hubert.surrer@sdm.de

Abstract: In der Vergabe und Abwicklung von Kundenprojekten beobachtet man oft, dass Erkenntnisse aus der Informatik den Weg in die Köpfe der Kunden nicht finden - oder sich sogar als Irrtümer manifestieren.

Gleichzeitig beobachten wir immer wieder Nachlässigkeiten in der Projektführung. Beide Erkenntnisse zusammen bringen uns dazu, bei der Führung von Projekten nach immer mehr Professionalität zu streben.

Der Beitrag vergleicht Paradigmen aus dem Software-Engineering mit Software-Management und skizziert unter den oben genannten Einflüssen, wie man Software-Projekte professionell plant und steuert. Der Beitrag möchte zum Nachdenken anregen und die Fachwelt etwas näher an die IT-Welt rücken.

1 Motivation

Dass die Entwicklung von Individualsoftware ein schwieriges Geschäft ist, ist weiß Gott keine neue Erkenntnis. Nur zur Erinnerung: die CHAOS-Studie der Standish Group [Stan01] bescheinigt uns zwar eine Verbesserung in Termin- und Budgetüberschreitungen in den letzten zehn Jahren, trotzdem scheitern immer noch 23% aller untersuchten Projekte, d. h. diese Projekte werden nie produktiv! Die Ursachen dafür sind vielfältig. Unvollständige Anforderungen, fehlende Anwenderbeteiligung und "moving targets" werden als Top Ursachen genannt. McKinsey hat 1999 die Geheimnisse erfolgreicher IT-Projekte in ihrem Buch „Secrets of Software Success" veröffentlicht. Daran haben in den letzten 10 Jahren auch Werkzeuge, Prozesse und Vorgehensmodelle wenig geändert. Ich erinnere an 4GL-Sprachen, CASE-Tools, das AD/Cycle-Konzept der IBM oder Programmiersprachen wie Smalltalk. Es bleibt dabei: den Stein der Weisen hat die Informatikbranche noch nicht gefunden. Laut Dirk Berensmann [Ber05] hat die IT seit Anfang der Neunziger sogar wieder einen Rückschritt in Richtung „Handwerk" gemacht - ausgelöst durch Technologiehype, New Economy und nachfolgende Budgetstopps.

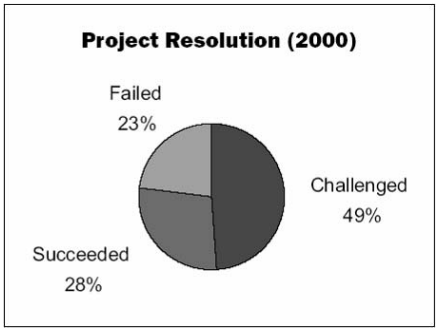


Abbildung 1: Chaos Studie 2001 an der Standish Group (aus [Stan01])

Zwei Beobachtungen scheinen dem zu widersprechen:

IT does not matter [Car04]

Informationstechnologie ist zur commodity geworden, zur Ware also, die man einkaufen kann wie Strom aus der Steckdose. Von einem gewissen Standpunkt aus hat Nicholas Carr sogar Recht: IT ist kein Alleinstellungsmerkmal mehr, sondern „Business enabler". Allein daraus zu schließen, dass jetzt alles ganz einfach wird, dass es keine Budget- oder Terminüberschreitungen mehr gibt, ist eine gefährliche Vereinfachung der Aussage „It is commodity". Wer hat schon von Termin- oder Budgetproblemen bei der Lieferung von Strom gehört? Unser Handwerkzeug wird nicht einfacher, sondern komplexer: SUN veröffentlicht ständig neue Java-Releases, wir werden beglückt mit neuen Versionen und Upgrades von Datenbanken, Middleware, Application-Servern usw. Einfacher wird es dadurch nicht.

Zwang zur Reduktion von Kosten

Märkte werden globaler, Unternehmen stehen im internationalen Wettbewerb, der Zwang zum Sparen wächst und wächst. Kostensparprogramme begegnen uns überall, manchmal haben sie wohlklingende Namen wie CORE, save-for-growth oder First Choice. Die Kostensparprogramme werden intern umgesetzt, schlagen aber auch auf die Lieferanten durch, was sich letztlich am Einkaufsverhalten widerspiegelt.

Wie wird heute IT-Dienstleistung eingekauft:

- *Skill Level soll in großen Unternehmen oftmals Mitarbeiter normieren*

Natürlich ist es erstrebenswert, Preise von Dienstleistern miteinander zu vergleichen. Alle Berater werden in ein Raster so genannter Skill-Level´s gesteckt, die versuchen, eine vermeintliche Normierung herzustellen. Dabei lehrt uns schon DeMarco, [DeMar, Lis91] dass die Performance zwischen dem besten und dem schlechtesten Mitarbeiter einen Faktor von 1:10 betragen kann, um nur einen Aspekt zu nennen, der diese Praxis fragwürdig erscheinen lässt.

- *Internetauktionen als Ersatz zu persönlichen Vertragsverhandlungen*

Nachdem IT ja offensichtlich commodity ist, und Berater über Skill-Level normiert sind, kann man auch den nächsten Schritt tun, und Stundensätze in einer online Auktion verhandeln. So zu sagen "ebay anders herum". Bleiben da die Qualität und Alleinstellungsmerkmale der Lieferanten auf der Strecke?

- *Risiko der Schätzunsicherheit beim Anbieter*

Große Vorhaben werden oft über alle Phasen von der Spezifikation bis zur Einführung und Hardwarebeschaffung ausgeschrieben. Dass ohne Vorliegen einer Spezifikation eine seriöse Aufwandsschätzung nicht machbar ist, bleibt das Risiko des Lieferanten (siehe dazu auch Kapitel 2). Oft werden solche Vorhaben unterschätzt. Das wird dann auch ein Problem für den Auftraggeber, wenn Mehraufwände nicht mehr im geplanten Terminrahmen abzuleisten sind.

- *Offshore Entwicklung*

Wir befinden uns in einem globalen Markt, weshalb also nicht auch die niedrigen Löhne und Kosten anderer Industrieländer wie China, Indien, Philippinen oder auch der Nearshore Länder wie Weißrussland, Polen und Ukraine ausnutzen? Wir satteln also auf die ohnehin schon komplexe Software-Entwicklung noch die Komplexität wie fremde Muttersprache, andere Kultur und Zeitverschiebung oben drauf. Bemerkenswert ist, dass es keine veröffentlichte Studien über gescheiterte Offshore Projekte gibt.

Die Kausalkette: „Software ist commodity, muss also immer günstiger einzukaufen sein" ist nur bis zu einem bestimmten Punkt richtig. Möglicherweise lohnt es sich, ein paar Euro mehr für Software auszugeben und Folgeschäden zu reduzieren. Gescheiterte Softwareprojekte haben US-Amerikanischen Firmen im Jahr 1995 immerhin 81 Mrd. USD [Stan01] gekostet.

2 Das Unglück beginnt mit der Schätzung

Prognosen sind besonders dann schwierig, wenn sie sich auf die Zukunft beziehen. Schätzung von Individualsoftware ist in der Informatik leider noch immer unterentwickelt. Es gilt noch immer: „Our techniques of estimating are poorly developed" [Bro75]. Alle Schätzmethoden basieren letztlich auf unscharfen Basisgrößen wie z. B. die Einordnung in leichte, mittlere und schwere Usecases, in Funktionspunkte oder in Lines of Code. Je früher die Projektphase, umso höher die Schwankungsbreite der Schätzung. Ein Faktor 4 zwischen zwei Schätzergebnissen ist in frühen Phasen normal [DeMar94], wenn eine Schätzung überhaupt möglich ist: „The cone of uncertainly implies that it is not only difficult to estimate a project accuractly in the early stages, it is theoretically impossible" [McCo98]

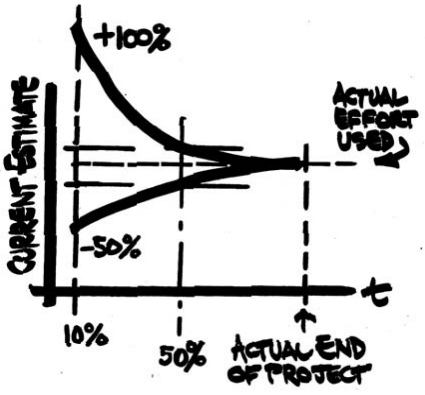


Abbildung 2: Schätzunsicherheit nach Projektverlauf (aus [DeMa94])

Versetzen wir uns in die Lage eines Auftraggebers. Die Ausschreibungsunterlagen sind verschickt. Vier Softwarelieferanten geben die Angebote ab: der Teuerste 3,4 Mio €, der Billigste 950 T€, dazwischen zwei mit 1,9 Mio € und 2,5 Mio €.

Ist die Aussage dieser Angebote, dass der Billigste die fachlichen Anforderungen nicht verstanden hat oder der Teuerste nicht wettbewerbsfähig ist, „sich zu warm angezogen hat“?

Hat wirklich jeder alles verstanden? Kennt jeder die Komplexität der eigenen Betriebsumgebung (deployment, Betrieb, Rechenzentrum etc.)? In dem Bewusstsein, dass eine Schwankung um Faktor 4 normal ist, macht man einen Fehler, wenn man sowohl den billigsten wie auch den teuersten Anbieter nicht ernst nimmt.

Aber was ist die Lösung?

Schätzgrundlagen schaffen

Ein Umsetzungsprojekt sollte ausschließlich auf Basis einer ordentlichen (!) und vollständigen (!) Spezifikation erfolgen. Wir beobachten häufig Ausschreibungen auf Basis von Spezifikationen, die diese Kriterien nicht erfüllen oder auf Basis von Requirements durchgeführt werden.

Machen Sie sich selbst ein Bild:

Auftraggeber sollten unabhängig von den Softwarelieferanten eine eigene Schätzung durchführen oder von Experten durchführen lassen.

Ist möglicherweise das eine oder andere Projekt von vorn herein zum Scheitern verurteilt, weil das Fundament einer ordentlichen Schätzung und damit Projektplanung gefehlt hat?

3 Struktur? Struktur!

Die Randbedingungen für Dienstleiter sind also strenger geworden: Dienstleistung wird als austauschbare Ware angesehen, der Preis steht im Vordergrund, Risiken aus großen Schätzunsicherheiten trägt häufig der Lieferant. Was bleibt also? Professionalität in Schätzung und Durchführung!

Im Folgenden gehe ich auf das Thema Durchführung ein und beginne mit einem Plädoyer für Struktur. Dies gilt sowohl für Kundenprojekte, als auch für interne Projekte.

Softwaresysteme sind die komplexesten Gebilde, die Menschen erschaffen haben. Für die Umsetzung solch komplexer Gebilde streben wir in der Informatik spätestens seit dem Nato-Gipfel 1968 nach Software-Engineering: Friedrich L. Bauer: „There is so much tinkering with software…..what we need is software engineering“ [Nato68]

Seit dem versuchen wir uns mit Struktogrammen, Nassy-Shneidermann, endlichen Automaten und vielem mehr. Letztlich aber haben wir nur “Boxes and arrows“, Prosa und Sourcecode.

Einige wenige ewige Wahrheiten des Software-Engineering sind unbestritten und allgemein anerkannt: Über Datenkapselung (zu Beginn als ADT, heute als Objekte) oder Nutzung von Interfaces wird heute niemand mehr streiten. Standardarchitekturen (SOA, Quasar) [Sid04] haben ihren Platz gefunden, und dass „goto´s“ gefährlich sind, wissen wir auch spätestens seit Edsger W. Dijkstra [Dijk68].

Die gleichen Paradigmen, die wir für die Umsetzung von komplexen Softwaresystemen wie selbstverständlich verwenden, sollten auch für die Projektplanung gelten:

Kapselung

Gute Pläne fassen verschiedene Arbeitspakete zu größeren Einheiten zusammen und kapseln den jeweils darunter liegenden Inhalt. Eine geordnete Darstellung dieser Kapselung sollte in Form von Projektstrukturplänen (work breakdown structures) erfolgen.

Interfaces

Was für die Software selbst Interfaces oder Schnittstellen sind, sind für Pläne Abhängigkeiten zwischen Arbeitspaketen. Die Planung sollte so geschnitten sein, dass sich möglichst einfach darstellbare Abhängigkeiten ergeben.

Architektur

Architektur hat etwas mit Ästhetik zu tun, genau wie ein Plan eine gewisse Ästhetik ausdrücken sollte.

goto

Haben Sie schon einmal einen Plan mit einem goto gesehen? Kennen Sie Pläne, die eine Abhängigkeit des Arbeitspaketes 563 zum Arbeitspaket 2 und vom Arbeitspaket 17 zum Arbeitspaket 623 darstellen? Na bitte!

Für die Steuerung von großen Softwareprojekten ist also professionelle Planung unerlässlich, ich gehe deshalb kurz auf die wichtigsten Themen ein.

4 Steuerung von Softwareprojekten

Es scheint so einfach, viele Bücher sind darüber geschrieben worden. Und trotzdem scheitern noch immer 23% aller Softwareprojekte. Gute Planung ist notwendige Grundlage, aber nicht hinreichend für die erfolgreiche Steuerung von Software Projekten. Trotzdem muss man mit den Grundlagen beginnen, denn nicht einmal da sprechen wir die gleiche Sprache.

Gleiche Sprache

Im Maschinenbau ist eine M6 Schraube eine M6 Schraube. Die ist genormt, darüber gibt es keine Diskussion. In der Informatik ist die Definition eines Arbeitspaketes zwar auch genormt,[13] die Norm hat aber in der Industrie leider noch keinen Einzug gefunden.

Ebenso verhält es sich mit den Begriffen wie: Aktivität, Aufgabe, Vorgang, Stufe und Release.

Weitere Beispiele sind Begriffe wie: Modultest, Integrationstest, Entwicklertest und Systemtest. Auch hier beobachten wir in verschiedenen Projekten verschiedene Interpretationen.

Selbst Rollen und Titel sind nicht normiert. Hat ein Projektleiter oder ein AP-Leiter mehr Verantwortung, gibt es einen Projektmanager und was macht ein Teilprojektleiter?

Wir begegnen bei jedem Kunden in jedem Projekt neuen Begriffen. Nun ist es Gott sei Dank nicht wie beim Turmbau zu Babel, dass daran Projekte scheitern, aber mindestens die ersten Wochen sind schwierig und führen zu Missverständnissen.

Handwerkliche Fehler bei der Projektplanung

Es gibt einige Mindestanforderungen, die wir an gute Pläne stellen. Eine fehlende Umsetzung davon ist manchmal möglicherweise durch die Auftraggeber-Auftragnehmerbeziehung verursacht. Die folgende unsortierte Liste soll einen Eindruck über handwerkliche Fehler vermitteln, die uns in der Projektplanung häufig begegnen.

[13] [DIN69901 „Projekt management systeme“]

- Arbeitspakete werden zu klein (ein oder zwei Tage) oder zu groß (zwei oder drei Monate) geplant

- Planung ohne Puffer: ein Plan mit Puffer ist kein Zeichen von Unsicherheit oder Schwäche sondern von Professionalität

- Nicht jeder Termin ist ein Meilenstein (Milestone oder Millstone) [Bro75]

- Projektteams mit viel Teilzeitkräften (trau keinem unter 70)[14]

- Fehlendes Change Management

- Fehlende Definition von "was ist fertig": kennen Sie Statusberichte, bei denen Arbeitspakete 4 Wochen lang den Status zu 90% fertig haben?

- Fehlende Projektstrukturpläne

- Endtermine die nicht geplant, sondern vom Management vorgegeben sind

- Das Einfachste wird zuerst eingeplant (weil es am Einfachsten ist): Ein Projekt gerät nicht in Zeitverzug, weil die letzte Stammdatenpflege nicht fertig ist, sondern weil die kritischen Funktionen nicht funktionieren. Diese plant man zu erst ein.

- Teamaufbau zu stark. Man kann es nur immer wieder betonen: das „Chinesenprinzip" funktioniert nicht ewig, und „Adding manpower to a late software project makes it later". [Bro75]

Der Projektplan

Wir schlagen vor, die Projektplanung in Anlehnung an die PMI Arbeitsgebiete 5, 6 und 9 [PMBOK04] als die Summe folgender Einzelprojekte zu definieren:

- 1. Projektstrukturplan (der zeigt das „was")

- 2. der Aufwandsplan (der zeigt, „wie aufwändig")

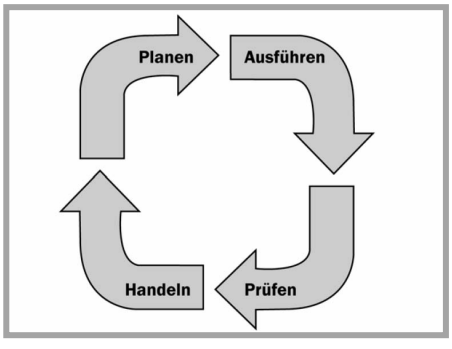
- 3. der Terminplan (der zeigt, das „wann" und die Abhängigkeiten)

- 4. der Personalplan (der zeigt das „wer")

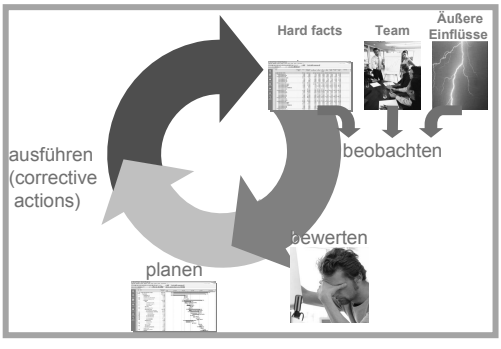
[14] 70 meint in diesem Zusammenhang weniger als 70 % Kapazität

Softfacts

Neben diesen Hardfacts gibt es eine Reihe von Softfacts, die für die erfolgreiche Steuerung von Softwareprojekten zusätzlich berücksichtigt werden müssen. In Erweiterung zu PMBOK ® plädieren wir hier für einen erweiterten Regelkreislauf, der die Schritte „beobachten“ und „bewerten“ als explizite Prozessschritte definiert. Wir trennen in Anlehnung an die Medizin den *Befund* (Fieber = beobachten) vom Schritt *Diagnose* (Grippe = bewerten) und *Therapie* (Bettruhe = Planung). Das Ausführen könnte man als *Anamnese* des Folgeschritts betrachten.



PMBOK Regelkreislauf



sd&m Regelkreislauf

Literaturverzeichnis

[Ber05] Dirk Berensmann, Deutsche Postbank AG, Vortrag auf der sd&m-Konferenz Software-Industrialisierung am 16.06.2005, IT matters – but who cares?

[Bro75] Frederick P. Brooks: The Mythical Man Month, Addison Wesley 1975

[Car04] Nicholas G. Carr: Does IT matter? Harvad Business School Publishing, Boston 2004

[DeMa94] Tom de Marco: Peopleware and beyond: Workshop mit sd&m, München 1994

[DeMa,Lis91] Tom DeMarco, Timothy Lister: Wien wartet auf Dich!

[Dijk68] Edsger W. Dijkstra: Letters to the editor: go to statement considered harmful, Communications of the ACM, v.11 n.3, p.147-148, 1968

[McCo98] Steve Mc Connell: Software Project Survival Guide, Microsoft Press, Redmond, 1998

[Nato68] P. Naur and B. Randell (Eds.): Report on a Conference sponsored by the NATO SCIENCE COMMITTEE, Garmisch 7.-11.10.1968, NATO, Brüssel 1969

[PMBOK04] Projekt Management Institute: A guide to the Projekt Management of Knowledge (PMBOK ® Guide) Dritte Ausgabe, Pennsylvania 2004

[Sid04] Johannes Siedersleben: Moderne Software-Architektur. Umsichtig planen, robust bauen mit Quasar, Dpunkt Verlag 2004

[Stan01] Standish Group www.standishgroup.com

Erfahrung bei der Einführung einer SOA für den Senat für Bildung, Jugend und Sport des Landes Berlin

E. Ulrich Kriegel, Holger Kurrek

Abteilung Sichere Business IT Infrastrukturen
Fraunhofer Institut für Software- und Systemtechnik
Mollstraße 1
D-10178 Berlin
ulrich.kriegel@isst.fraunhofer.de
holger.kurrek@isst.fraunhofer.de

Abstract: Im Rahmen des Projektes „Integrierte Software Berliner Jugendhilfe“ wurde eine diensteorientierte und komponentenbasierte Architektur für die Senatsverwaltung für Bildung, Jugend und Sport von Berlin realisiert. Dabei zeigte sich, dass die Einführung und Realisierung moderner Architekturkonzepte keine hinreichende Bedingung für den Erfolg eines Projektes sind. Gerade in Projekten, in denen mehrere unabhängige Hersteller Realisierungsarbeiten durchführen, ist die Sicherung der konzeptionellen Integrität sowohl im Entwurf als auch später im Betrieb relevant.

1 Einleitung

Seit 2002 führt die Senatsverwaltung für Bildung, Jugend und Sport von Berlin das mehrstufige Projekt „Integrierte Berliner Jugendverwaltung (ISBJ)“ durch, in dem bestehende IT-Fachverfahren (Kinder in Tagesbetreuung, Wirtschaftliche Jugendhilfe, Zentrale Vormundschafts- und Unterhaltsvorschusskasse) modernisiert werden. Hauptziele des Projektes sind ein fachübergreifendes Controlling, die Verbesserung der Arbeitsprozesse der Sachbearbeiter, die Auslagerung von Arbeitsprozessen als Self-Services an externe Partner, z. B. an die Betreiber von Kinderbetreuungseinrichtungen und das Anbieten innovativer Dienstleistungen im Rahmen der E-Government-Initiative für die Bürger von Berlin. Die einzelnen Teilprojekte werden entweder ausgeschrieben oder im Rahmen bestehender Dienstleistungsverträge entwickelt, so dass unterschiedliche Hersteller an den Lösungen arbeiten.

Das Fraunhofer Institut für Software- und Systemtechnik begleitet das Projekt seit Ende 2002 als Qualitätssicherer [KNS04]. Qualitätssicherung beinhaltet hierbei sowohl die analytische Qualitätssicherung der gelieferten Artefakte als auch die Sicherung der konzeptionellen Integrität der entwickelten Softwaresysteme. Zusätzlich werden auch die Anforderungen der IT-Sicherheit und der wirtschaftlichen Betriebsführung geprüft.

Während der Laufzeit des Projekts zeigte sich, dass die Spezifikation und anschließende Realisierung einer dienstezentrierten Architektur zwar eine notwendige, aber keine hinreichende Bedingung dafür ist, dass die realisierten Fachverfahren erfolgreich in den produktiven Betrieb überführt werden können. Gerade in Projekten, in denen mehrere Anbieter Komponenten und Dienste realisieren, die dann in ein fertiges Gesamtsystem integriert werden müssen, ist die strikte Einhaltung von Prozessen in Entwicklung und Betrieb unumgänglich. Aber selbst dann existieren mit heutigen Technologien noch Probleme, für die es derzeit keine befriedigende Lösung gibt.

2 Das Projekt ISBJ

Von der Ausgangslage her ist das Projekt „ISBJ“ ein typisches Projekt für die öffentliche Verwaltung: Die erstellten Lösungen müssen konform zu den gesetzlichen Vorgaben sein. Diese sind über einen gewissen Zeitraum stabil, nach einer Änderung im Rahmen des Gesetzgebungsprozesses muss die Lösung jedoch sehr schnell an die neuen gesetzlichen Regelungen angepasst werden. Wartbarkeit und Modifikationsfähigkeit sind deshalb primäre Anforderungen an die Softwaresysteme der öffentlichen Verwaltung.

Hinsichtlich der IT-Sicherheit sind die Vorgaben des BSI für E-Government-Anwendungen [EGHB] zu berücksichtigen, die auch den IT-Grundschutz beinhalten. Hinzu kommen noch Vorgaben des Landes Berlin sowie des Betreibers bzw. zentralen Dienstleisters der IT-Infrastruktur im Land Berlin. Der Betrieb der im Rahmen des Projekts entwickelten Anwendungen ist nach den Best Practices aus der IT-Infrastructure Library [ITIL] zu organisieren und in die bestehenden Betriebsprozesse der vorhandenen IT-Infrastruktur einzugliedern.

Softwaresysteme für die öffentliche Verwaltung werden z. Zt. meist als Individualsysteme in einem Nischenmarkt realisiert, der von wenigen kleinen Anbietern dominiert wird. Die am Markt angebotenen Softwaresysteme entsprechen meist technologisch nicht dem aktuellen Stand der Softwaretechnik, auf Grund der benötigten Kapazität sind nicht alle angebotenen Lösungen in Großstädten wie Berlin einsetzbar. Aus den damit verbundenen Problemen resultiert die zusätzliche Anforderung, möglichst Software auf einer skalierbaren „Standardplattform“ zu realisieren, so dass die Abhängigkeit von bestimmten Herstellern reduziert und ggf. eine Migration zu einem anderen Hersteller, d. h. ein flexibles Sourcing, erleichtert wird.

Neben diesen speziellen Rahmenbedingungen gelten all die Rahmenbedingungen bzgl. Leistungsfähigkeit, Ausfallsicherheit und Wartbarkeit, wie sie auch für Softwaresysteme in Unternehmen gelten: Von den Mengengerüsten her sind die nichtfunktionalen Anforderungen der einzelnen Fachverfahren mit denen kleinerer Finanzdienstleister vergleichbar. So muss das Teilprojekt „KiTa (Kinder in Tagesbetreuung)“ Informationen zur Betreuung von jährlich etwa 150 000 Kindern verwalten und den verteilten Zugriff von etwa 300 Sachbearbeitern aus 12 Bezirksverwaltungen und aus der Senatsverwaltung über das Intranet und von etwa 1000 Trägern von Kindertagesstätten über das Internet ermöglichen.

Die Mitarbeiter der öffentlichen Verwaltung besitzen eine hohe fachliche Kompetenz. Erfahrungen auf dem Gebiet der Softwaretechnik sind jedoch nur begrenzt vorhanden, so dass teilweise die Auswirkungen bestimmter fachlicher Anforderungen nicht abgeschätzt werden können.

Für den Erfolg großer IT-Projekte in der öffentlichen Verwaltung ist deshalb eine herstellerunabhängige Unterstützung in Form einer externen Qualitätssicherung essentiell, wie sie z. B. von öffentlichen Forschungseinrichtungen erbracht werden kann. Neben den üblichen Aufgaben der analytischen Qualitätssicherung besteht eine wesentliche Aufgabe der Qualitätssicherung in der Sicherung der konzeptionellen Integrität der gesamten Entwicklung, in der Abschätzung der Auswirkungen fachlicher Anforderungen auf die Komplexität der Realisierung, in der Gewährleistung der IT-Sicherheit sowie in der Definition der Rahmenbedingungen für einen sicheren Betrieb.

2.1 Sicherung der konzeptionellen Integrität

Projekte der Größenordnung von ISBJ werden in mehreren Phasen realisiert, so dass unterschiedliche Anbieter mit der Realisierung von Teilprojekten beauftragt werden können.

Um die konzeptionelle Integrität der Arbeiten der einzelnen Teilprojekte zu sichern, wurde seitens der Qualitätssicherung für das Projekt ein verantwortlicher Software- und Systemarchitekt benannt, der den Auftraggeber in allen architekturrelevanten Fragen berät. Er ist für alle Fragen zuständig, die die Plattform, Software- und Systemarchitektur betreffen. Soweit nicht globale Vorgaben betroffen sind, liegt die Realisierung der einzelnen Komponenten in der Hoheit der sie erzeugenden Auftragnehmer. Regelmäßige Absprachen zwischen den Anbietern und dem Architekten haben sich jedoch als günstig erwiesen, da hierdurch auch der Aufwand in der analytischen Qualitätssicherung reduziert werden konnte.

Zu Beginn des Projektes wurde die unten näher erläuterte Referenzarchitektur spezifiziert und später erweitert, in die sich alle Realisierungen einzuordnen haben. Die Arbeiten gingen dann in die Spezifikation der Systemarchitektur über, die in enger Zusammenarbeit mit den für den Betrieb Verantwortlichen erfolgte.

Um eine bessere Wartbarkeit und Wiederverwendung der realisierten Komponenten zu ermöglichen, wurden verbindliche Programmierkonventionen vorgegeben, an die sich alle Realisierungen zu halten haben.

2.2 Abschätzung der Auswirkungen fachlicher Anforderungen

Ein Großteil der fachlichen Anforderungen können auf unterschiedliche Art und Weise realisiert werden. Eine wesentliche Aufgabe der Qualitätssicherung besteht deshalb darin, bei der Formulierung fachlicher Anforderungen durch den Auftraggeber auf die Konsequenzen für die Realisierung sowie für die Geschäftsprozesse hinzuweisen.

Im Teilprojekt „KiTa“ wurde deshalb ein Jour Fixe etabliert, an dem neben den fachlich Verantwortlichen und der realisierenden Firma auch der Architekt teilnimmt. In diesem Kreis werden alle fachlichen Anforderungen diskutiert und einer Lösung zugeführt.

2.3 Gewährleistung der IT-Sicherheit

Im Projekt „ISBJ“ wird auch mit personenbezogenen Daten gearbeitet, deren Vertraulichkeit unbedingt gesichert werden muss. Weiterhin sind die in den Zahlungen eingesetzten Mittel von mehr als einer Milliarde Euro im Jahr sicher zu verwahren. Daher wurde die Bedeutung der IT-Sicherheit von Anfang an auch durch die Projektverantwortlichen hoch eingeschätzt.

Die Gewährleistung der IT-Sicherheit wurde auf Empfehlung des Fraunhofer ISST durch ein ganzheitliches IT-Sicherheitskonzept angegangen. Entsprechend den Empfehlungen des BSI [EGHB] wurde nach dem BSI-Standard [BSIS2] vorgegangen. Wichtig für die Wirtschaftlichkeit der IT-Sicherheitsmaßnahmen war die frühzeitige Begleitung des Projekts durch das IT-Sicherheitskonzept, so dass auch schon in der Konzeptphase der jeweiligen Komponenten Anforderungen der IT-Sicherheit berücksichtigt werden konnten und damit später ein teureres Nachbessern vermieden wurde.

Zuerst wurde aus den geplanten Geschäftsprozessen eine Schutzbedarfsfeststellung abgeleitet, welche die schützenswerten Objekte erfasst und qualitativ den Schutzbedarf bestimmt, aus dem sich der Umfang der notwendigen IT-Sicherheitsmaßnahmen ableitet. Dabei konnte in Iterationen durch eine Modifikation von Geschäftsprozessen der Schutzbedarf und damit der spätere Aufwand teilweise reduziert werden.

Die begleitende Weiterentwicklung des IT-Sicherheitskonzepts unterstützt auch die nachfolgenden Design-Entscheidungen, indem Sicherheitsrisken minimiert und damit im späteren Betrieb aufwändige Schutzmaßnahmen begrenzt bleiben bzw. Sicherheitsvorfälle vermieden werden. Dabei kann der Einsatz von IT-Sicherheitsmaßnahmen wie SSL-Verschlüsselungshardware auch zu Einsparungen in der Implementierung führen, indem dort diese Funktionen nicht eingebaut werden müssen.

Zur Gewährleistung der IT-Sicherheit werden im IT-Sicherheitskonzept Anforderungen definiert, die in den realisierten Komponenten umgesetzt und deren Wirksamkeit auch geprüft werden muss. Ebenso gelten diese Anforderungen für die Entwicklung und die Betriebsführung der Anwendungen. Daher werden entsprechende Fragestellungen in einer Arbeitsgruppe, dem IT-Sicherheitsteam, zusammen mit Anwendungsentwicklern, fachlichen Experten und den Betreibern regelmäßig diskutiert.

2.4 Definition der Rahmenbedingungen für einen sicheren Betrieb

Das Projekt „ISBJ“ ist das erste große Projekt, in dem komponentenbasierte Software auf der Basis einer J2EE-Plattform verfahrensübergreifend eingesetzt wird. Es existieren im Land Berlin weder Erfahrungen mit der Technologie noch mit der komponentenbasierten Software-Entwicklung. Unter Leitung des Fraunhofer ISST wurde deshalb eine Arbeitsgruppe etabliert, die sich mit Fragen des Betriebs der ISBJ-Plattform befasst. Ziel ist es, den sicheren Betrieb der Plattform und der Anwendungen durch Mitarbeiter der Senatsverwaltung zu ermöglichen. Zu Beginn des Projektes erfolgte der provisorische Betrieb durch einen externen Anbieter, Mitte 2006 konnte der Betrieb durch Mitarbeiter des Rechenzentrums von SenBJS übernommen werden. Spätestens Ende 2007 ist mit dem Umzug in ein Sicherheitsrechenzentrum der endgültige Ausbau der Systemarchitektur geplant.

Die Software wird durch verschiedene Auftragnehmer, die nach unterschiedlichen Zeitplänen arbeiten, in Form von Komponenten realisiert. Deshalb muss dafür gesorgt werden, dass die einzelnen Komponenten getestet und in Betrieb genommen werden können, ohne dass die sie nutzende Fachverfahren in Mitleidenschaft gezogen werden. Die Systemarchitektur wurde deshalb so ausgelegt, dass eine Hierarchie von Plattformen für unterschiedliche Tests zur Verfügung steht. Neben dem Produktionssystem steht ein baugleiches Abnahmetestsystem zur Verfügung. Komponenten wie die Personenstammverwaltung greifen auf andere Systeme des Landes Berlin zu und können deshalb nur im Intranet des Landes Berlin und nicht bei einem Auftragnehmer getestet werden. Deshalb wurde ein zusätzliches Testsystem etabliert, auf dem die Hersteller von Komponenten ihre Software im Zusammenspiel mit anderen Komponenten testen können. Dieses System enthält mehrere Instanzen der ISBJ-Plattform, so dass es neben den Tests auch für Schulungen genutzt werden kann.

Die Abstimmung der Entwicklung und die Bereitstellung einer neuen Version der jeweiligen Komponenten mit ihren Abhängigkeiten auf den verschiedenen Systemen wurde durch die Definition von ITIL-konformen Prozessen für das Change- und Releasemanagement [ITIL] und deren technische Unterstützung durch ein Trouble Ticket System gesichert. Nur so können die unterschiedlichen Anforderungen von Entwicklern, Fachlichkeit und Betreibern erfolgreich koordiniert werden. Dabei ist die Integration der Qualitätssicherung ein Erfolgsfaktor.

Eine der größten Herausforderungen bei der Vorbereitung des Betriebes war die Etablierung eines einheitlichen Deployment-Prozesses, der auch den Anforderungen eines Produktionsbetriebes genügt. Ursprünglich verwendeten die beteiligten Unternehmen eigene Deployment-Szenarien, die z. T. dedizierte Rechner benötigten. Sensible Informationen wie Passworte oder Ressource-URLs wurden an unterschiedlichen Stellen verwaltet, so dass der Aufwand, diese zu ändern, erheblich war.

Erst durch die Vorgabe eines einheitlichen, zur verwendeten J2EE-Plattform (s. u.) konformen Deployment-Konzeptes für alle Produzenten von Komponenten und Anwendungen konnte dieses Problem gelöst werden.

2.5 Wissenstransfer und Schulungen

Um eine gemeinsame Kommunikationsbasis zu schaffen und um durch den Aufbau eigenen Wissens die Abhängigkeit vom Qualitätssicherer zu reduzieren, wurden parallel zu o. g. Aktivitäten Workshops mit dem Auftraggeber durchgeführt. Im Mittelpunkt stand die Abstimmung der fachlichen Zielstellung unter Berücksichtigung der technischen Möglichkeiten und der gesetzlichen Rahmenbedingungen. Als Techniken wurden Szenarien und UML-Aktivitätsdiagramme etabliert, auf deren Basis die Fachanwender ihre funktionalen Anforderungen formulieren und auch aus fachlicher Sicht validieren können. Daneben erfolgte eine Einführung in die Problematik der Software-Architekturen und in komponentenbasierte Software-Entwicklung.

Falls Auftragnehmer und Qualitätssicherer unterschiedliche Lösungskonzepte für Teilprobleme entwickelten, wurden deren Vor- und Nachteile in Form von Workshops vor dem Auftraggeber diskutiert, so dass dieser darauf aufbauend eine endgültige Entscheidung treffen konnte.

Wesentliche Grundlage für einen erfolgreichen Wissenstransfer ist eine vollständige Dokumentation, die auch durch den Auftraggeber weitergepflegt werden kann. So wurde auch in der Qualitätssicherung auf die Vollständigkeit und Korrektheit der Dokumentation zu den Lieferungen der realisierten Komponenten ein Schwerpunkt gesetzt. Außerdem wurden auch die Anforderungen und Checklisten der Qualitätssicherung dem Auftraggeber dokumentiert zur Verfügung gestellt.

Für den Betrieb wurden ITIL-konforme Prozesse wie *Incident*-, *Problem*-, *Change*- und *Release-Management* gemeinsam mit dem Auftraggeber entwickelt, abgestimmt und dokumentiert.

Als Voraussetzung für die Übernahme des Betriebes durch Mitarbeiter von SenBJS wurden gemeinsam mit dem Auftraggeber mehrere mehrtägige Schulungen zu Applikationsserver, Datenbank und Data Warehouse geplant, die dann vom Hersteller der Applikationsplattform durchgeführt wurden.

3 Software- und Systemarchitektur für ISBJ

Um die Anforderungen an die Unabhängigkeit von den Software-Lieferanten und an die langfristige Weiterentwicklungsfähigkeit der Anwendungen zu gewährleisten, wurde durch das Fraunhofer ISST eine komponentenbasierte und dienstezentrierte Referenzarchitektur vorgegeben: Die Zerlegung in Komponenten und Dienste ermöglicht eine gezielte Weiterentwicklung der in ihnen gekapselten Funktionalität. Komponenten können unabhängig voneinander entwickelt und dann in die Anwendung integriert werden, bei konstanten oder additiv erweiterten Schnittstellen können Komponenten für den Benutzer transparent ausgetauscht werden. Obwohl bisher weder Notwendigkeit noch die Gelegenheit dafür bestand, dass Komponenten auch durch andere als die sie erstellenden Firmen gepflegt und weiterentwickelt werden können, sollte das durch die strikten Programmiervorgaben, geforderten Realisierungsbeschreibungen und durch das Einhalten der J2EE-Standards möglich sein.

Gemeinsam mit dem Auftraggeber wurden auf der Ebene der Fachlogik die von den Anwendungen benötigten Dienste identifiziert. Dabei wurde zwischen fachspezifischen Anwendungen und domänenspezifischen bzw. Infrastruktur-Diensten unterschieden. Fachanwendungen stellen keine wieder verwendbaren Dienste bereit und werden deshalb als Ganzes von einem Auftragnehmer realisiert. Domänenspezifische Dienste besitzen das Potenzial zur Wiederverwendung in der Domäne der Senatsverwaltung für Bildung, Jugend und Sport bzw. im Land Berlin und müssen wieder verwendet werden.

Die folgenden domänenspezifischen Dienste wurden im Kontext des Projekts „ISBJ" realisiert bzw. befinden sich gerade in der Realisierungspase:

- Buchungskomponente
  Das Land Berlin verwendet als Kassensystem die Software „ProFiskal", die z. Zt. nur Dateischnittstellen zur Verfügung stellt. Die Buchungskomponente stellt im Rahmen der ISBJ-Plattform eine Schnittstelle zu „ProFiskal" bereit, die unabhängig von der konkreten Realisierung der ProFiskal-Import-Schnittstelle ist. Damit wurde die Entwicklung der ISBJ-Fachverfahren von der Weiterentwicklung des Systems „ProFiskal" entkoppelt.

- Einrichtung- und Dienste
  Die Komponente „Einrichtungs- und Dienste" verwaltet zentrale Leistungsdaten und stellt damit in einheitlicher Weise Konfigurationsinformationen für alle Fachverfahren bereit.

- Personenstammverwaltung
  Die Komponente „PersonenstammVerwaltung" verwaltet fachverfahrens-übergreifend Personeninformationen und ordnet diesen einen eindeutigen Identifikator zu. Damit ist gesichert, dass pro Person genau ein Datensatz existiert und dass jedes Fachverfahren mit identischen Personenstammdaten arbeitet. Im Zuge des Anlegens eines neuen Personendatensatzes wird dieser über einen Abgleich mit dem Informationssystem des Landesamtes für Bürger- und Ordnungsangelegenheiten (LABO) verifiziert.

- Regionales Bezugssystem
  Das regionale Bezugssystem liefert Katasterinformationen und sozialraumbezogene Informationen, die für fachliche Auswertungen relevant sind. Die Informationen werden vom Statistischen Landesamt bereitgestellt und offline verteilt. Die zentrale Bereitstellung dieser Daten reduziert den Wartungs- und Pflegeaufwand. Eine spätere Nutzung eines „RBS-Dienstes" des Statistischen Landesamtes kann für den Nutzer transparent erfolgen.

Als Infrastrukturkomponenten wurden eine zentrale Benutzerverwaltung und ein Output-Manangement identifiziert, dass den zentralen Druck und E-Mail-Versand von Dokumenten im Portable Document Format (PDF) ermöglicht.

Bei der Analyse der Anforderungen wurde weiterhin festgestellt, dass die geforderten Auswertungen in allen Fachverfahren ähnlich sind. Deshalb wurde die Auswertungsfunktionalität aus den Fachverfahren in ein zentrales Auswertungssystem ausgelagert, das in Form eines Data Warehouses realisiert wurde.

Da in der Senatsverwaltung für Bildung, Jugend und Sport noch nie eine Diensteplattform betrieben wurde, musste sie im Rahmen des Projektes aufgebaut werden. Aus ökonomischen Gründen, und um Erfahrungen im Betrieb zu sammeln, sollte zunächst eine minimale Diensteplattform realisiert werden, die später erweitert wird.

Nach der Festlegung der Oracle-J2EE-Plattform als Realisierungsplattform wurde das Komponentenmodell für die J2EE-Plattform entsprechend präzisiert.

Sowohl bei der Realisierung von Fachanwendungen als auch beim Zugriff auf die realisierten Service-Komponenten wurde bewusst auf nachrichtenorientierte Schnittstellen und asynchrone Kommunikation verzichtet, da diese jeweils lokal *deployed* werden, und eine Typprüfung zur Kompilationszeit während der Entwicklungsphase den Nachteil einer festen Kopplung ausgleicht. Der Zugriff auf Dienste externer Anbieter, z.B. auf Einwohnermeldedaten oder das Output-Management erfolgt dagegen schon jetzt asynchron über eine nachrichtenorientierte Schnittstelle.

Auf die Einrichtung eines zentralen Service-Repositories und auf die Realisierung weiterer SOA-Features wurde verzichtet, da sie momentan keinen Mehrwert für den Auftraggeber darstellen.

Abbildung 1 gibt einen Überblick über die Systemarchitektur der ISBJ-Plattform in der endgültigen Ausbaustufe, in der der besseren Übersichtlichkeit halber die Test- und Schulungsserver nicht dargestellt sind. Auf der rechten Seite ist das Intranet-System dargestellt, das aus einem Applikationsserver besteht, auf dem die Präsentations- und die Geschäftslogik-Schicht lokalisiert sind.

Auswertungen können einen extremen Ressourcenbedarf haben, der sich negativ auf den Durchsatz der Fachanwendungen auswirken kann. Deshalb wird ein separater Applikationsserver für die Auswertungslogik betrieben. Die Datenhaltung erfolgt in einem Oracle-RAC[15]-System, dass über drei Rechner realisiert wird, die auf ein gemeinsames Speichersubsystem zugreifen.

---

[15] RAC: Real Application Cluster

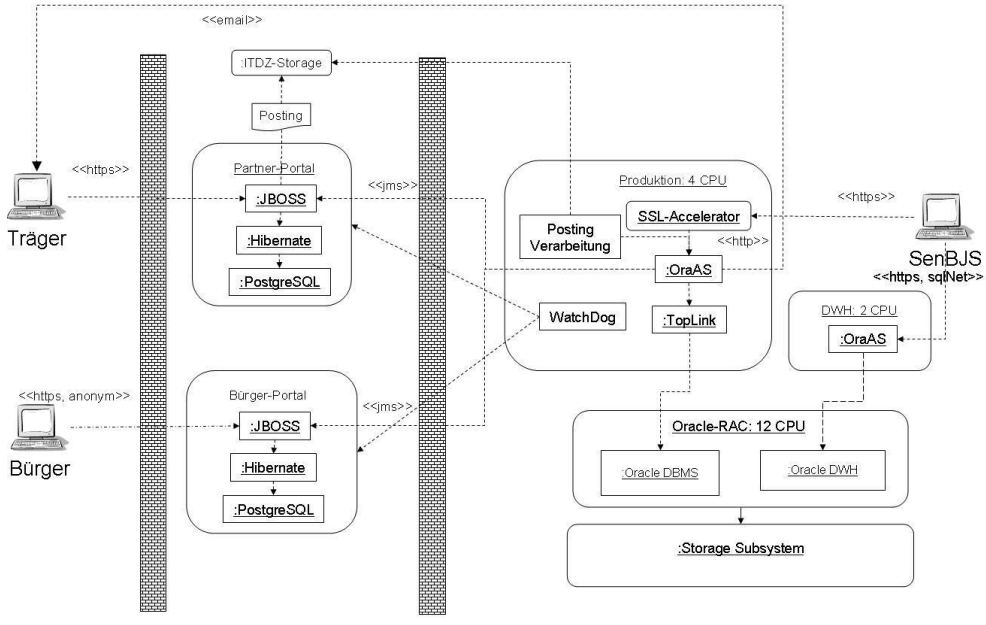


Abbildung 1: Überblick über die Systemarchitektur der ISBJ-Plattform

Für die Interaktion mit externen Partnern, z. B. Trägern von Kindertagesstätten wurde als IT-Sicherheitsarchitektur ein asynchroner Mechanismus entwickelt, der die Vorgaben des BSI für E-Government-Verfahren berücksichtigt, die einen einseitigen aus dem Intranet initiierten Verbindungsaufbau vorschreiben. Dieser Mechanismus kann später auch für die Übermittlung von Anträgen der Bürger verwendet werden. Die Träger-Arbeitsplätze werden zusätzlich durch SSL-Client-Zertifikate als geschlossene Benutzergruppe authentisiert.

Träger greifen aus dem Internet auf dedizierte Rechner in der DMZ des Rechenzentrums zu. Zur Erteilung von Auskünften besitzen diese Rechner eine separate Datenhaltung, die aus dem Intranet aktualisiert und deren Konsistenz mittels der Komponente „WatchDog“ überwacht wird.

Anträge werden in Form so genannter Postings asynchron auf einen sicheren Speicher im Intranet transferiert. Vom Produktionsrechner aus werden die Postings zyklisch abgefragt und auf ihre Konsistenz und Glaubwürdigkeit geprüft. Akzeptierte Postings werden in den Bearbeitungsprozess eingespeist. Kommt es im Zuge des Bearbeitungsprozesses zu einer Änderung des Datenbestandes, so wird diese asynchron propagiert, so dass auch der Datenbestand der Internet-Rechner aktualisiert werden kann. Um die Administration der Firewalls zu vereinfachen, wird das SMTP-Protokoll zur Übermittlung der Postings genutzt, die Aktualisierung der Rechner in der DMZ erfolgt über JMS-Nachrichten.

Die Rückmeldung durchgeführter Aktionen an die Träger erfolgt über E-Mails.

4 Erfahrungen aus dem Projekt

Nach einer Projektlaufzeit von 3 Jahren und der erfolgreichen Überführung von Anwendungen in den Probeecht- bzw. Echtbetrieb können jetzt erste Aussagen über den Erfolg des Projektes getroffen werden.

Gerade wenn Ziele wie die Unabhängigkeit von Software-Herstellern erreicht werden sollen, ist eine rein analytische Qualitätssicherung nicht ausreichend. Die Qualitätssicherung muss konstruktiv sowohl bei der Gestaltung der Software- und Systemarchitektur als auch bei der Planung des Betriebes eingreifen, um die Abhängigkeit von den Software-Produzenten zu minimieren. Hier sind herstellerunabhängige Organisationen wie Forschungseinrichtungen im Vorteil, da sie auf Grund ihrer Forschungstätigkeit auch konsistente Alternativen zu Standardlösungen aufzeigen können. Außerdem sind sie nicht genötigt, eigene Produkte zu vermarkten.

Die intensive Diskussion zwischen Auftraggeber, konstruktiver Qualitätssicherung und Lösungsanbietern half, den Realisierungsaufwand in Grenzen zu halten. Eine gemeinsame Abschätzung der Häufigkeit bestimmter Geschäftsprozesse führte zu Einsparungen, da diese nicht softwaretechnisch unterstützt sondern in Form von Organisationsanweisungen außerhalb des Systems umgesetzt wurden.

Zu Beginn der Arbeiten existierten bei allen Anbietern Defizite im Umgang mit den Kontrakten, die die J2EE-Plattform definiert. Diese konnten jedoch durch eine intensive analytische Prüfung von Quell- und Bytecode identifiziert und behoben werden.

Wesentlich für den Erfolg des Projektes ist auch die Frage, inwieweit das gewählte Architektur-Paradigma ausreicht, um einen kontinuierlichen Betrieb von Anwendungen und Ziele wie Wartbarkeit, Erweiterbarkeit und Unabhängigkeit vom Auftragnehmer über einen längeren Zeitraum zu gewährleisten und ob beim Betrieb der Anwendungen zusätzlicher organisatorischer Aufwand entsteht.

4.1 Einfluss der gewählten Architektur auf die Lösung

Durch die Verwendung von Konzepten wie „Komponente“ oder „Dienst“ in Verbindung mit strikten Programmiermodellen konnte man sich den gesetzten Zielen annähren. Die gute Wartbarkeit zeigt sich u. a. darin, dass geänderte gesetzliche Rahmenbedingungen problemlos in eine Fachanwendung überführt werden konnten.

Betrachtet man die Architektur jedoch nicht aus dem Blickwinkel der Software-Architektur, sondern in Hinblick auf einen sicheren Betrieb, so triff man auch auf Fragen, für die das gewählte Architektur-Konzept keine befriedigende Antwort liefert.

4.1.1 Wechselbeziehung zwischen Software- und Systemarchitektur

Prinzipien der modellbasierten Softwareentwicklung und das Architekturmuster SOA helfen, Software zu strukturieren und Dienste-Komponenten effizient zu entwickeln. Es sind jedoch außerhalb von Forschungseinrichtungen keine Techniken bekannt, die helfen, die wechselseitige Beeinflussung von Software- und Systemarchitektur zu beherrschen, die ggf. zu starken Einschränkungen bei den Möglichkeiten des Deployments von Diensten führen kann.

Wird bei der Realisierung z.B. ein Persistenzframework verwendet, das davon ausgeht, die alleinige Datenhoheit zu besitzen, dann können Service-Komponenten nicht mehr einfach zur Lastverteilung auf mehrere Server verteilt werden. Es muss dann für eine Synchronisierung der lokalen Caches gesorgt werden, was die Komplexität der Anwendungen und damit den Aufwand bei der Betriebsführung erhöhen kann: Gegebenenfalls müssen zusätzliche Kommunikationskanäle für die Synchronisation der Caches geöffnet werden, was negative Auswirkungen auf das Sicherheitskonzept des Verfahrens bedeuten kann.

Die Entwicklung einer SOA muss deshalb immer mit der Entwicklung der Systemarchitektur koordiniert werden. Nach einer Geschäftsprozessanalyse folgt die Zieldefinition, die im Wesentlichen eine Zerlegung der Funktionalität in Dienste beinhaltet. Daran schließt sich eine Definition einer Rahmenarchitektur an, die auf einer Auswahl geeigneter Architekturstile und -muster aufbaut. Die Auswahl dieser Architekturstile und -muster berücksichtigt die nicht-funktionalen Anforderungen der Geschäftsprozesse und Dienste. Die Definition der Referenzarchitektur hat wiederum Rückwirkung auf die Definition der Dienste.

4.1.2 Garantierte Performance und Verfügbarkeit von Diensten

Sollen Dienste zentral angeboten werden, so stellt der Nutzer eines Dienstes bestimmte Anforderungen an Performance und Verfügbarkeit. Durch eine lose Kopplung auf der Basis einer asynchronen nachrichtenbasierten Kommunikation zwischen Diensten können Verfügbarkeitsschwankungen nivelliert werden, es gibt jedoch kein allgemeingültiges Verfahren zur Garantie einer bestimmten Performance. Eine horizontale Skalierung, d. h. das Clustering von Laufzeitumgebungen für Service-Komponenten, kann die Verfügbarkeit und auch die Performance eines Dienstes verbessern. Eine bestimmte Dienstqualität kann heute jedoch nur garantiert werden, wenn der Dienst über definierte Ressourcen verfügt. Die Möglichkeit, einen Dienst singulär auf einem dedizierten Server zu betreiben, scheitert meist an ökonomischen Gründen, so dass nur noch Virtualisierungsmethoden als Lösung in Frage kommen. Virtualisierungsmethoden sind heute in einigen Serverbetriebssystemen integriert und erlauben eine exakte Ressourcenzuordnung zu Applikationen, sie sind aber heute noch nicht feingranular genug, um eine Ressourcenzuordnung auf Dienste-Ebene zu ermöglichen. Die Lösung, eine Service-Komponente in einer Virtualisierungsumgebung mit definierten Ressourcenzuordnungen wie Prozessor-Anzahl oder I/O-Kapazität zu betreiben, scheitert an einem unverhältnismäßigen Management-Overhead und bei kommerziellen Applikationsservern auch an Lizenzkosten. Deshalb werden heute Applikationsserver meist überdimensioniert. Eine Validierung der abgeschätzten Leistungsgrenzen im Produktionsbetrieb muss auf experimentellem Weg mittels Lasttests erfolgen.

4.2 Zusätzliche organisatorische Anforderungen

Um verteilt entwickelte diensteorientierte Architekturen erfolgreich betreiben zu können, muss ein zusätzlicher organisatorischer Aufwand zur Koordination der Entwicklung betrieben werden.

4.2.1 Release- und Testmanagement

Besteht eine Anwendungslandschaft weitgehend aus isolierten „monolithischen“ Verfahren, so ist auch deren Releasemanagement weitestgehend entkoppelt. Werden Dienste jedoch auf einer gemeinsamen Plattform betrieben oder benutzen Service-Komponenten gemeinsame Ressourcen, so sind die Aktivitäten zum Releasemanagement der einzelnen Entitäten nicht mehr entkoppelt und müssen in ein übergeordnetes plattformweites Releasemanagement integriert werden. Eine analoge Aussage gilt auch für das Testmanagement.

Gegenüber dem Testen isolierter Einzelanwendungen entsteht so ein erhöhter Organisations- und Koordinationsaufwand.

4.2.2 Koordination verteilter Entwicklung und Verfahrensverantwortung

Der Aufbau komplexer dienstorientierter Architekturen ist häufig mit der parallelen Entwicklung von Diensten verbunden. Das betrifft gleichermaßen neue Dienste wie auch die Kapselung und Adaptierung von Altanwendungen. Gerade im öffentlichen Bereich, wo diese Arbeiten in der Regel ausgeschrieben werden müssen, werden diese Entwicklungen häufig von mehreren Dienstleistern durchgeführt. Das Ziel einer funktionsfähigen, den Ansprüchen genügenden Gesamtarchitektur ist nur durch eine sorgfältige Koordination der Entwicklungsarbeiten zu gewährleisten. Grundvoraussetzung für diese Koordination ist eine IT-Gesamt-Referenzarchitektur, in der die Regeln und Standards für die konkrete IT-Gesamtarchitektur vorgegeben sind.

In der öffentlichen Verwaltung tragen sogenannte Verfahrensverantwortliche die Gesamtverantwortung für Fachverfahren. Im Rahmen der Etablierung von SOAs sind entsprechende Verantwortlichkeiten für Dienste festzulegen, die z. B. aus den Dateneigentümern abzuleiten sind. Das kann im Vergleich zu isolierten Systemen einen erhöhten Organisationsbedarf zur Folge haben, vor allem dann, wenn die Dienste verteilt betrieben werden.

4.3 Fazit

Zusammenfassend kann man feststellen, dass die Einführung einer SOA in einer Organisation Potentiale einer erhöhten Evolutionsfähigkeit und Reaktionsfähigkeit gegenüber sich ändernden Anforderungen birgt. Sie ist eine große Verbesserung, die notwendig aber nicht hinreichend für evolutionssichere Unternehmens-IT-Systeme ist. Die Bewältigung der vielfältigen Organisations- und Koordiniationsaufgaben ist der wesentliche Erfolgsfaktor dafür, ob die mit SOA gesteckten Ziele, eine evolutionsfähige IT-Infrastruktur und die Reduzierung von Kosten, erreicht werden kann.

Literaturverzeichnis

[EGHB] Bundesamt für Sicherheit in der Informationstechnik (BSI): E-Government-Handbuch, Modul „Sicherer Internet-Auftritt im E-Government“, Stand 13.06.2005.

[BSIS1] BSI-Standard 100-1: Managementsysteme für Informationssicherheit (ISMS), Version 1.0

[BSIS2] BSI-Standard 100-2: IT-Grundschutz-Vorgehensweise, Version 1.0

[BSIS3] BSI-Standard 100-3: Risikoanalyse auf der Basis von IT-Grundschutz, Version 2.0

[GSK] BSI: IT-Grundschutz-Kataloge, Stand Dezember 2005

[ITIL] IT Infrastructure Library - Service Support (ccta), Central Computer & Telecommunications Agency, The Stationery Office Books, 2000.

[KNS04] E. U. Kriegel, L. Nentwig, E. Siegeris:“IT-Projekte in der öffentlichen Verwaltung – Analogien zum Outsourcing. Ein fallbeispiel aus der Berliner Verwaltung“, in J. Borchers, R. Kneuper (Eds): “Software Management 2004: Outsourcing und Integration”, Locture Notes in Informatics, 2004.

Systemfamilienentwicklung bei arvato direct services

Jan Leßner

DCO
arvato direct services
An der Autobahn
33310 Gütersloh
jan.lessner@bertelsmann.de

Abstract: Für die Erbringung von CRM-Dienstleistungen wird bei arvato direct services seit einigen Jahren eine Familie von Softwaresystemen entwickelt. Bei der Betrachtung der konkreten Vorgehensweise lassen sich dabei Analogien zu natürlichen Vorgängen und zur gezielten gentechnischen Steuerung derselben ziehen. Über eine solche Analogiebetrachtung lässt sich Aufschluss über erhaltenswerte Erfolgsfaktoren und über Risiken und Grenzen der angewandten Entwicklungsprozesse gewinnen.

1 Software und Familie

Die arvato direct services GmbH ist einer der größten Direkt-Marketing-Dienstleister weltweit und betreibt zur Erbringung ihrer Dienstleistungen für unterschiedliche Kunden eine Reihe maßgeschneiderter Software-Systeme. Dabei darf einerseits nicht jedes Mal das Rad neu erfunden werden, andererseits muss genügend Raum für die individuellen Bedürfnisse der Geschäfte bleiben. Im Folgenden wird dargestellt, welche Anforderungslage bei arvato dabei zur Bildung einer ganzen Familie verwandter Systeme geführt hat, und welche Konzepte sich auf lange Sicht als entscheidende Erfolgsfaktoren in der produktiven Praxis heraus gestellt haben. Verwendet man hierbei den Begriff der „Familie“ im Zusammenhang mit Software-Entwicklung, dann liegt es nahe, zum Verständnis von Zusammenhängen einen Blick auf natürliche Vorgänge der Familienbildung zu werfen, sowie auf die Möglichkeiten gezielter Einflussnahme durch menschliche Eingriffe in diese Prozesse. Und tatsächlich tun sich bei einer solchen Analogiebetrachtung interessante Parallelen auf, aus denen sich Empfehlungen, Chancen und Risiken für die Software-Entwicklung ableiten lassen. Ein Großteil der hier geschilderten Erkenntnisse ist dabei unabhängig von einem konkreten Geschäftsfeld.

Zum Kerngeschäft von arvato direct services gehört die gesamtheitliche Abbildung von sogenannten Kundenbindungsprogrammen, in denen Endverbraucher durch das Gewähren von Bonusleistungen zum regelmäßigen Einkauf bei den Programmbetreibern animiert werden. Die Betreiber solcher Programme – und damit die Kunden von arvato – kommen aus sehr unterschiedlichen Bereichen, z.B. Krankenversicherungen, Baumärkte, Lebensmittelketten. Mit der Entwicklung und auch der langfristigen Wartung der benötigten Software-Systeme wird üblicherweise eine interne IT-Abteilung beauftragt. Das Volumen der Entwicklungsprojekte liegt überwiegend im Bereich von 100 bis 1000 Personentagen bei einer Projektlaufzeit von 2 – 6 Monaten. Die Entwicklung erfolgt oft unter hohem Kosten- und Zeitdruck und setzt daher einen effizienten Entwicklungsprozess voraus. Die IT-Abteilung umfasst einen Stamm von ca. 15 festen und zurzeit noch einmal 15 externen Mitarbeitern und ist in dieser Form seit ca. 4 Jahren tätig. Dementsprechend wurde bereits eine ganze Reihe von Systemen entwickelt und befindet sich aktuell auch in Wartung. Die Entwicklung basiert vollständig auf Java und den Standards der Java 2 Enterprise Edition (J2EE).

2 Anforderungsprofile

Durch den thematischen Schwerpunkt ergeben sich im Anforderungsprofil für die beauftragten Systeme immer wieder erkennbare Parallelen. So besteht ein immer wiederkehrendes Verständnis dafür, wie bestimmte Geschäftsobjekte IT-technisch abzubilden sind, z.B. Kunde, Kundenkarte, Punktekonto oder Bestellung. Es gilt auch häufig, immer wieder die gleichen externen IT-Systeme zu integrieren, wie z.B. die Adressprüfung einer anderen Konzerntochter, ein Debitorenmanagement oder die Druckstraßen des Hauses Bertelsmann für hochvolumigen Postversand. Bzgl. der abzubildenden Geschäftprozesse lassen sich ebenfalls wiederkehrende Anforderungen finden, z.B. die Registrierung von Neukunden, der Verfall von Bonuspunkten, die Fakturierung von Bestellungen oder der Kündigungsvorgang. Und schließlich gibt es auch bestimmte Schnittstellen, die jedes IT-System nach außen anbieten muss – im Kern zumeist eine effiziente graphische Oberfläche für die Arbeit der Servicecenter und Webservices für den Anschluss von Web-Oberflächen an das Backend.

Der hohe Grad an wiederkehrenden Aufgaben spricht auf den ersten Blick für die Verwendung eines Standardprodukts für CRM-Lösungen als Basis der Entwicklung. Es bestehen aber andererseits mindestens ebenso viele projekt-individuelle Anforderungen, die diesen Ansatz schwer machen. Dies liegt unter anderem daran, dass Kundenbindung zwar der Schwerpunkt aber nicht das einzige Standbein ist. So liegt beispielsweise auch die Entwicklung einer Plattform zur Verwaltung von Riesterrentenanträgen in der Verantwortung des IT-Bereichs, sowie die Abwicklung von Bestellungen für SIM-Karten eines Mobilfunkbetreibers. Aber selbst die Kundenbindungsprogramme haben teilweise sehr spezifische Profile, die eine Individualentwicklung notwendig machen.

Das oben beschriebene Spannungsfeld aus wiederkehrenden und individuellen Anforderungen prägt das Bild der Software-Entwicklung bei arvato direct services und hat zum Prinzip einer Systemfamilienentwicklung geführt. Bewährte Bausteine aus früheren Projekten müssen in bestmöglicher Weise in neuen Projekten wieder eingesetzt werden, um auf diese Weise Aufwände zu minimieren. Andererseits muss die Verwendung solcher Bausteine in einer Weise erfolgen, die keine Barriere für die spezifische Individualentwicklung darstellt. Wie macht man also in diesem Umfeld das Beste aus seinem Erbe?

Bei der Suche nach einer geeigneten Erläuterung der verfolgten Konzepte ließ sich eine aufschlussreiche Analogie zu natürlichen Vorbildern aufstellen, aus denen sich einige einleuchtende Grundprinzipien für die Systemfamilienentwicklung ableiten lassen.

3 Eine Analogie zur Natur

Spricht man in der Software-Entwicklung von „Familien“, dann lohnt sich ein Blick darauf, wie sich bei natürlichen Lebensformen eine Fortentwicklung und Familienbildung ergibt und welche Einflussmöglichkeiten der Mensch dabei hat. Die Evolution – der natürliche Vorgang aus Mutation und Selektion – ist dabei das Fundament. Sie erzielt zwar allerbeste Ergebnisse von perfekt angepassten Lebensformen, ist aber auch extrem langwierig und produziert wegen des zufälligen Faktors darin eine große Menge von Varianten. Trotz des guten Rufs also kein unmittelbares Vorbild, obwohl der Begriff der evolutionären Entwicklung in der IT-Welt durchaus auftaucht (siehe z.B. [DAHME]).

Deutlich schneller kommt man zu neuen Ergebnissen durch menschlichen Eingriff in den Selektionsprozess – die Züchtung. Dieses Verfahren beschränkt sich üblicherweise auf eine einzige Familie von Pflanzen oder Tieren (was uns ja ausreichen würde) und genießt auch noch einen relativ guten Ruf. Trotzdem braucht es immer noch mehrere Generationen und oft ein paar Jahrzehnte Zeit, bis sich ein gewünschtes Ergebnis einstellt. Will man bereits innerhalb einer einzigen Generation eine nennenswerte Fortentwicklung bewirken, muss auch in den Prozess der Mutation eingegriffen werden, womit man in den Bereich der Genmanipulation kommt. Grundlage für die Mutation ist eine deutlich gezieltere Form der Selektion – und zwar die Extraktion und Isolation besonders gewinnträchtigen Erbguts. Hier ergeben sich die erstaunlichsten Möglichkeiten, wobei der Ruf dieser Technik zu Recht zweifelhaft ist, weil sie schwer in den Griff zu bekommen und vor allem ethisch bedenklich ist [FRAGNER]. Es besteht außerdem die erhebliche Gefahr, dass sich auch genetisch defekte und unerwünschte Lebensformen unkontrolliert selbst fortpflanzen.

In der Genmanipulation liegt also sowohl das größte Potential als auch die Grenze des menschlich Beherrschbaren bei der Fortentwicklung einer Familie von Lebensformen. Was lässt sich nun aus der allgemeinen Anschauung dieses Bereichs für die Softwareentwicklung lernen, wenn wir die Begrifflichkeiten und Methoden auf den Bereich der Systemfamilienentwicklung übertragen?

Für die Arbeitsweise im IT-Bereich von arvato lassen sich folgende Analogien bilden:

- Eine Generation von Lebewesen entspricht einem System der Systemfamilie
- Das „Ausbrüten“ einer solchen Generation entspricht der Entwicklung dieses Systems
- Das Erbgut lässt sich gleichsetzen mit dem arvato-eigenen Anwendungsframework
- Der Vorgang der Selektion (Erbgutextraktion) entspricht der Erweiterung dieses Frameworks
- Die Mutation kann mit der Anwendung des Frameworks in der Entwicklung eines neuen Systems gleichgesetzt werden.

Das Problem der ethischen Bedenklichkeit entfällt an dieser Stelle, da hier keine lebenden Wesen mehr manipuliert werden. Auch die Gefahr der Selbstfortpflanzung destruktiver Gene besteht hier glücklicherweise nicht - das wären sonst Computerwürmer, die bei arvato selbstverständlich *nicht* hergestellt werden. Auf die aber durchaus bestehende Frage der Beherrschbarkeit wird weiter unten noch genau eingegangen.

4 Analogien im Vorgehen

Betrachten wir uns das Vorgehen bei der Systemfamilienentwicklung in Analogie zur Gentechnik, so lässt sich feststellen, dass es sich hierbei um zwei mit einander verschränkte Prozesse handelt, was durch die folgende Graphik verdeutlicht werden soll:

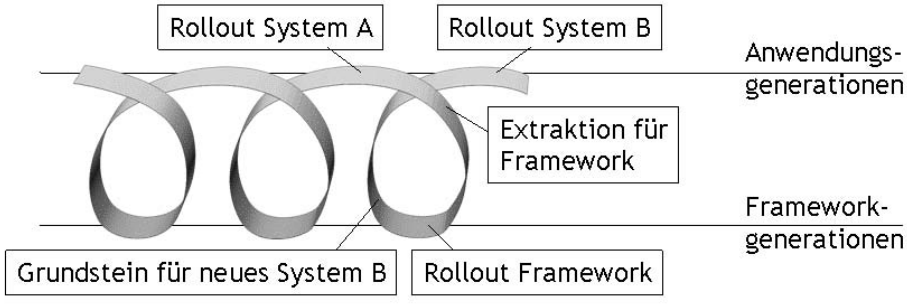


Abbildung 2: Prozess der Systemfamilienentwicklung

Ist ein System der Familie fertig gestellt worden, schließt sich ein Prozess der Extraktion wieder verwendbarer Bausteine an. Die Integration dieser gesäuberten und isolierten Bausteine ins Framework (den „Genpool“) führt zu einer neuen Version desselben. Diese neue Version dient nun wiederum als Grundlage für die Entwicklung der nächsten Generation von Systemen, indem die dafür verwendbaren Bausteine des Frameworks „injiziert“ werden. Dieser Vorgang wiederholt sich bei der nächsten Generation erneut und bildet auf diese Weise einen permanenten Anreicherungs- und Verbesserungsvorgang. Anwendungs- und Frameworkentwicklung sind dabei nicht nebenläufig und unabhängig von einander, sondern bedingen sich gegenseitig. Das Framework stellt dabei nicht nur eine technologische Basis dar, sondern umfasst auch konzeptionelle und organisatorische Aspekte wie z.B. den Aufbau des Software-Enwicklungsprozesses an sich.

Dieses Vorgehen ist auch Kerngedanke des sog. *Software Product Line Engineerings* das einen zunehmend verwendeten Begriff für ganzheitliche Wiederverwendung bildet (siehe z.B. [POHL]). Die darin verankerten Prinzipien sind zur Zeit Ausgangspunkt für Überlegungen zur Formalisierung des Requirements- und Test-Managements bei arvato direct services, wobei die praktische Anwendbarkeit im Vordergrund steht. Aktuelle öffentliche Aktivitäten in diesem Bereich haben oft noch Forschungscharakter, wie ein Blick auf den aktiven Teilnehmerkreis der jüngsten Software Product Line Conference[16] zeigt. Die industrielle Nutzbarkeit wird nur selten belegt, was u.U. auch strategische Gründe in den jeweiligen Unternehmen haben mag [FRAUN].

Funktionieren kann das oben beschriebene Verfahren nur, wenn in den Anwendungsprojekten klar ist, dass die Entwicklung der Software einerseits durch die Nutzung des Frameworks profitiert, dass sie andererseits aber auch ihren Beitrag zum Fortschritt des Frameworks beitragen muss. Dies umfasst sowohl simple Verbesserungen (Bugfixes) als auch das Beisteuern von Innovationen. Die Graphik macht deutlich, dass mit dem Rollout eines Systems die Entwicklung im Grunde noch gar nicht abgeschlossen ist, sondern sich der Extraktionsvorgang anschließt, der (mindestens zu großen Teilen) innerhalb des Projektbudgets geleistet werden muss. Diese Extraktion ist – analog zur Erbgutisolation der Gentechnik – eine anspruchsvolle und durchaus kostspielige Angelegenheit, da es sich hier um weit mehr als bloße Copy-Paste-Arbeit handelt. Hier gilt es Überzeugungsarbeit zu leisten und den Return-on-Invest nachzuweisen, was im IT-Bereich von arvato in Form einer entsprechenden ROI-Liste der Framework-Bausteine vorgenommen wird.

Voraussetzung für die Extraktion brauchbarer Bausteine ist die konsequente Auswertung von „Lessons Learned“, die bei arvato im Rahmen der Anwendungsentwicklung vom gesamten Team arbeitsbegleitend erfasst und nach Projektabschluss priorisiert und besprochen werden. Dabei handelt es sich um Erfahrungswerte sowohl im Umgang mit bestehenden als auch mit neu entwickelten Bausteinen, die helfen sollen, das Verbesserungs- und Wiederverwendungspotential aus praktischer Sicht einzuschätzen. Da nur „gesundes Erbgut“ extrahiert und ins Framework aufgenommen werden soll, sind die Beobachtungen der unmittelbar Beteiligten von besonderer Bedeutung.

---

[16] Siehe http://www.sei.cmu.edu/splc2006/

5 Das KISS-Prinzip

Wie oben bereits erwähnt, ist ein wesentliches Problem der Gentechnik ein Mangel an Beherrschbarkeit. Hauptgrund dafür ist (neben der Komplexität der Technik im Allgemeinen) die ungeheure Menge von Erbinformationen, mit denen es die Techniker zu tun haben. Selbst ehemals überzeugte Pioniere der Gentechnik wie Erwin Chargaff kommen auf Dauer zu einer kritischen Betrachtung [CHARGAFF]. Obwohl z.B. das menschliche Genom seit einigen Jahren als vollständig entschlüsselt gilt, ist nur ein Bruchteil dieser Informationen soweit verstanden, dass eine gezielte Manipulation mit einem vorhersagbaren Ergebnis überhaupt möglich ist. Ein Großteil der Erbmasse bleibt ein Rätsel, und stellt auch bei erfolgreichen Eingriffen an anderer Stelle eine schwer einzuschätzende Gefahr dar, weil die Querabhängigkeiten der einzelnen Informationen nur ungenügend einzuschätzen sind. Auch wenn die Erfolge der Gentechnik nicht von der Hand zu weisen sind, ist es an dieser Stelle fraglich, ob wir hier von einer Vorbildfunktion sprechen können. Übertragen wir dieses Problem in die Software-Entwicklung, dann findet es sich mit der bisher getroffenen Analogie tatsächlich wieder: übersteigt das für die Entwicklung verwendete Framework eine gewisse Komplexität, dann nimmt die Beherrschbarkeit ab. Selbst grundsätzlich als vorteilhaft zu wertende Eigenschaften können sich ins Gegenteil verkehren und eine vordergründige Einsparung durch erhöhte Instabilität wieder zunichte machen. Im Gegensatz zu biologischen Erbinformationen sind zumindest öffentlich entwickelte Software-Frameworks i.R. beständiger Anreicherung und Veränderung unterworfen. Es liegt daher in der Natur der Sache, dass sich keine dauerhafte Wissensstabilisierung einstellen kann.

Nun hat die Software-Entwicklung gegenüber der Gentechnik den Vorteil, in gewissen Grenzen selbst über die Komplexität des verwendeten Erbguts zu entscheiden. Es besteht also die Möglichkeit, das Anwendungs-Framework so klein zu halten, dass es auf der einen Seite ausreichende Hilfestellung bietet und auf der anderen Seite auch ausreichende Beherrschbarkeit. In der IT von arvato direct services wird aus diesem Grund konsequent das KISS-Prinzip (Keep It Small & Simple) bei der Pflege des eigenen Frameworks angewendet. Die Bausteine des Frameworks werden gar nicht erst darauf ausgelegt, eine 100%'ige Lösung darzustellen, sind aber dafür leicht durchschaubar und flexibel und so klein, dass jeder Entwickler die Chance hat, zur Not durch Studium des Codes die inneren Zusammenhänge innerhalb kurzer Zeit zu begreifen. Dieses Prinzip ist gerade für die Systemfamilienentwicklung von zentraler Bedeutung, da die benötigten Flexibilitätsdimensionen des Frameworks schwer vorhergesagt werden können. Um bei einem einfachen Bild zu bleiben: es reicht nicht aus, nur das Farb-Gen zu beherrschen, weil zuletzt immer nur rote, grüne und blaue Züchtungen gefordert wurden. Schon die nächste Anforderung kann lauten, dass sie 1, 2 oder 10 Meter groß sein müssen. Und diese Anforderungen in einer ganz anderen Dimension müssen umsetzbar sein, ohne dass zunächst jahrelange Forschung vorweg geht.

Die folgende Graphik zeigt einen ungefähren Größenvergleich zwischen dem bei arvato eingesetzten Framework und einem aus Mainstream-Komponenten aufgebauten Baukasten.

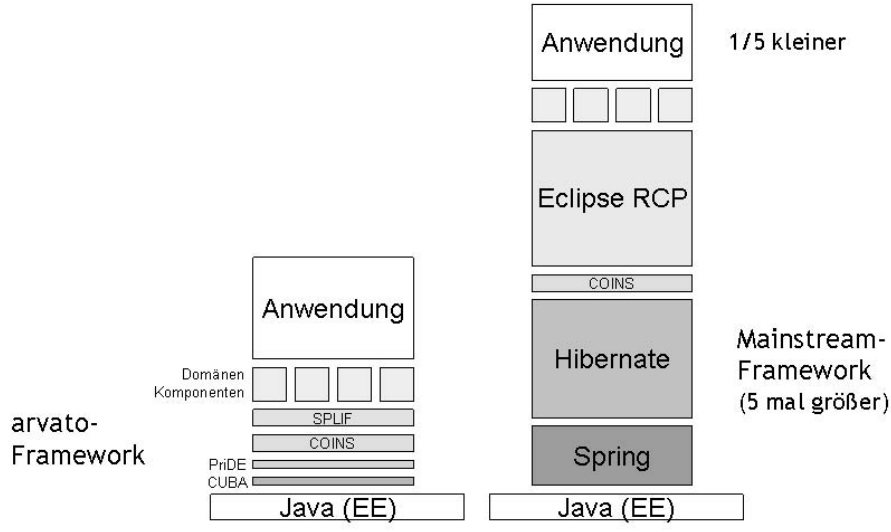


Abbildung 3: Frameworks im Größenvergleich

Als Basis für die Entwicklung der Geschäftslogik wird das OpenSource-Framework CUBA eingesetzt (dazu später noch mehr), und auch der Datenbankzugriff wird über ein OpenSource-Framework, den O/R-Mapper PriDE[17], unterstützt. Beide Frameworks haben Bibliotheken von unter 200 KByte Größe (inkl. zugehöriger Generatorwerkzeuge). Die Hilfsbibliothek COINS, sowie die Domänenkomponenten (Kundenverwaltung, Kontoführung, Adressvalidierung u.v.m) sind arvato-spezifisch und auch in einem anders gearteten Framework alternativlos. Der graphische Rahmen für Servicecenter-Applikationen wird von dem ebenfalls arvato-eigenen Framework SPLIF gestellt. In Summe ergibt sich dadurch ein Framework von weniger als 2 MByte Größe. Würde man die einzelnen Bestandteile durch Komponenten des aktuellen Mainstreams ersetzen (Hibernate[18] für den DB-Zugriff, Spring[19] als Basis für die Geschäftslogik und die Eclipse Rich Client Platform[20] als graphischen Rahmen), dann entsteht ein Framework von mindestens 5-facher Größe bei einer geschätzten Einsparung von ca. 20% in der Größe der Anwendung. Diese 20% sind jedoch größtenteils simpler, kanonischer Code, so dass die damit verbundene Aufwandsersparnis schätzungsweise bei 5-10% läge. Und auch dieser vordergründige Gewinn würde noch einmal durch Aufwände geschmälert, die durch das schwerer zu beherrschende Framework entstehen. Betrachtet man die Kosten für Erstellung und Wartung einer ganzen Herde von Systemen derselben Familie, stellt sich das KISS-Prinzip schließlich als der günstigere Weg heraus.

[17] Siehe http://pride.sourceforge.net
[18] Siehe http://www.hibernate.org/
[19] Siehe http://www.springframework.org/
[20] Siehe http://www.eclipse.org/

Insbesondere bei Themen der allgemeinen Infrastruktur (z.B. DB-Zugriff) sind häufig OpenSource-Lösungen oder Fremdprodukte verfügbar. Es macht durchaus Sinn, solche Komponenten als Bestandteile des eigenen Frameworks aufzunehmen, um hier die eigenen Entwicklungsressourcen zu schonen. Aber ein wachsames Auge ist trotzdem von Nöten: was hier zählt ist nicht ein Maximum an Features und Modernität, sondern die beste Eignung der Komponente für die Domäne und für die Einbettung in den Entwicklungsprozess und den Gesamtzusammenhang des „Erbguts". Auch fremd entwickelte Software muss vollständig verstanden und passend sein! Ein überschaubares Framework ist in diesem Zusammenhang auch deswegen von Interesse, weil es gefährliche Kopfmonopole vermeidet. Das Wissen um die Funktionsweise der Bausteine muss leicht transferierbar sein und sollte insbesondere nicht nur in den Händen externer Mitarbeiter liegen.

Im Gegensatz zu Infrastruktur-Bausteinen ist für die Komponenten der eigenen Kerndomäne üblicherweise nichts Vorgefertigtes auf dem freien Markt zu finden. Zur Bildung eines leistungsstarken Frameworks gehört also auch der Mut zur Extraktion eigener Bausteine. Frameworkbildung ist daher für die Systemfamilienentwicklung eine Kernkompetenz des Entwicklungsbereichs, wenngleich diese Aufgabe des hohen Anspruchs wegen auf wenige, konzeptionell starke Köpfe beschränkt ist. Trotzdem ist wichtig: auch diese Leute arbeiten im Normalfall in den Anwendungsprojekten und stellen kein separates Team im Elfenbeinturm dar. Die Extraktionsarbeit ist ein Vorgang, der in die Anwendungsentwicklung als nachgelagerte Phase einbezogen wird (s.o. Graphik zum Vorgehen).

6 Schneller Brüter: Die Systementwicklung

Betrachtet man die bisher getroffene Analogie, dann entspricht die Systementwicklung einer Art „Brutvorgang". Durch das Framework folgt dieser Vorgang zu gewissen Teilen bestimmten, wiederkehrenden Mustern. Die Entsprechung für auszubrütende Eier - genauer gesagt: die *Eiform* - sind die Modelle, denen eine Software-Entwicklung folgt. Solche Modelle (Denk-, Prozess-, Programmier-, Komponentenmodelle) müssen vom Framework mitgeliefert und selbst eingehalten werden, und sie müssen einfach und stabil sein. Gelingt es dabei, die Modelle über verschiedene Projekte und Anwendungsgenerationen gleich zu halten, ist damit eine wesentliche Basis für zielgerichtete und effiziente Entwicklung geschaffen. Der Aufbau nach einheitlichen und einfachen Modellen ist daher eine grundlegende Eigenschaft des bei arvato eingesetzten Frameworks. Die folgende Graphik macht dies am Beispiel der Verwendung des OpenSource-Toolkits CUBA[21] für die Entwicklung von Geschäftslogik deutlich.

[21] Siehe http://cuba.sourceforge.net

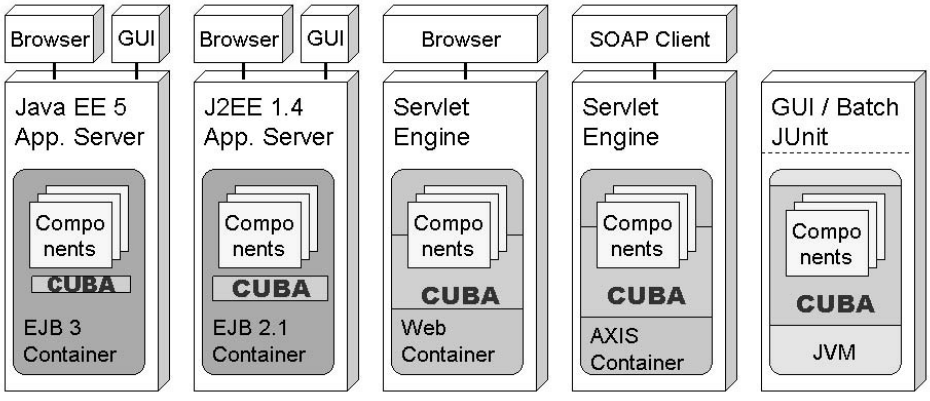


Abbildung 4: Runtime-Umgebungen für CUBA-Komponenten

Serverseitige (besser gesagt: containerseitige) Geschäftslogik wird nach den Mustern des EJB-3-Standards entwickelt, kann aber mit Hilfe des CUBA-Frameworks sowohl in EJB-3- und EJB-2.1-Containern betrieben werden als auch als AXIS Webservices oder in Stand-Alone-Applikationen. Eine dieser Stand-Alone-Umgebungen sind z.B. auch Unittests, die jederzeit einen Test der Komponenten ohne Server-Deployment möglich machen. Da also auf Basis desselben Modells jegliche aktuell benötigte Architekturen aufgebaut werden können, erfolgt die Entwicklung der Geschäftslogik grundsätzlich einheitlich nach den entsprechenden Mustern, unabhängig von der späteren Runtime-Umgebung. Diese Unabhängigkeit der Bausteine erhöht natürlich deutlich ihr Potential für Wiederverwendung in ganz anderen Kontexten. Die eventuell leichte Überdimensionierung des Modells in sehr kleinen Projekten wird bewusst in Kauf genommen, um die Einheitlichkeit zu gewährleisten.

Mir dem arvato-eigenen und java-swing-basierten GUI-Framework SPLIF für die Erstellung von Servicecenter-Applikationen wird das EJB-Modell teilweise auch auf die Client-Seite übertragen. Während die APIs hier natürlich ganz anders aussehen, wird trotzdem grundsätzlich das Konzept unabhängiger Module (sog. Teilapplikationen) verfolgt. Es gibt eine Unterscheidung zwischen „Stateful“ und „Stateless“ bei den Client-Bausteinen und dementsprechend den Vorgang der Aktivierung und Passivierung.

Was sich in der IT-Abteilung bei arvato direct services zur Zeit außerdem in der „Genforschung“ befindet ist die Schaffung eines einheitlichen Programmiermodells für Swing- und Web-Oberflächen. Dabei basiert die Web-Variante des SPLIF-Frameworks auf dem OpenSource-Framework Echo[22], das nach AJAX[23]-Verfahren arbeitet und Swing-Controls für das Web nachbildet. Die Programmierung erfolgt nach Observer-Pattern durch Anhängen von Listenern an die graphischen Controls, wie Entwickler dies bereits von Swing gewohnt sind. Die Eignung für die Aufnahme in die bereits bestehende Framework- und Modell-Landschaft steht hier im Vordergrund und weniger die Betrachtung des Mainstreams, der mit JSF[24] und Struts[25] im Web-Bereich zurzeit weitgehend einen anderen Weg verfolgt. Ein gewisser Mut zum Verzicht auf den maximalen Featureset ist auch hier wieder notwendig, denn Echo 2 bildet zwar eine gute Grundlage für die bei arvato selbst zu entwickelnden Web-Oberflächen, eine Auszeichnung für aller modernstes Webdesign ließe sich aber sicher nicht gewinnen. Auch hier spielt vor allem die Eignung für die Domäne der Systemfamilie eine Rolle und die Möglichkeit, nach gleichartigen Modellen zu arbeiten.

Erstaunlich ist die Erkenntnis, dass die konkrete Ausgestaltung des Entwicklungsprozesses eher zweitrangig ist. Aufgabe des Prozesses ist es, stabile „Brütbedingungen“ zu schaffen. Der Prozess muss bekannt sein und konsequent gelebt werden und muss (wie eingangs schon festgestellt) die Entwicklung des Frameworks mittragen. Ob aber ein agiler oder klassisch wasserfallartiger Vorgang besser geeignet ist, lässt sich aus der Analogie zur Gentechnik nicht ablesen und hat sich auch im IT-Bereich von arvato nicht als Erfolgsfaktor heraus gebildet. Der Prozess muss lediglich neben den oben genannten, eher nicht-funktionalen Aspekten, zur Organisationsstruktur des Unternehmens passen. Bei arvato ist dies ein usecase-orientierter Ansatz und ein iterativ-inkrementelles Vorgehen auf dieser Basis, um ausreichend Schnittpunkte für eine stufenweise Produktivstellung zu bieten. Das Verfahren hat sich bewährt, ist aber im Grunde wenig spektakulär und sicher nur bedingt übertragbar auf andere Unternehmen.

Interessanter Weise verfolgen die inkrementelle und agile Software-Entwicklung und das bis hierhin beschriebenen Vorgehen des Systemfamilienentwicklung bei arvato ein analoges Prinzip der schrittweisen Verbesserung (siehe z.B. [LARMAN]). Dabei bezieht sich die agile Software-Entwicklung jedoch auf Iterationen *ein und desselben* Systems, während die Systemfamilienentwicklung einen Prozess über *mehrere Systeme* hinweg betrachtet. Die Verfahren lassen sich also völlig unabhängig voneinander betrachten.

7 Chancen und Risiken

Die Gestaltung der Systemfamilienentwicklung nach den oben beschriebenen Prinzipien birgt eine ganze Reihe von Vorteilen, die für den IT-Bereich von arvato strategisch interessant sind.

---

[22] Siehe http://www.nextapp.com/platform/echo2/echo/
[23] Asynchronous JavaScript And XML, ein modernes Kommunikationsverfahren für Web-Clients und -Server
[24] Siehe http://java.sun.com/javaee/javaserverfaces/
[25] Siehe http://struts.apache.org/

- Da ein nennenswerter Teil der Entwicklung nach wiederkehrenden Mustern erfolgt, gibt es einen definierten, projektunabhängigen Einarbeitungspfad, der es erlaubt, neue Kräfte schnell einzuweisen. Dazu existieren eine Reihe von Workshops mit praktischen Übungen, die den Einstieg erleichtern, ohne dass produktive Kräfte zu stark durch die Einarbeitung neuer Kollegen gebunden werden.

- Durch die relativ kurze Einarbeitungszeit kann das Team recht kurzfristig (innerhalb von 1-2 Wochen) im Bedarfsfall verstärkt werden. Die Mannschaft wird dadurch gut skalierbar, was im Projektgeschäft ein großer Vorteil ist.

- Durch die Einheitlichkeit der Infrastruktur können Entwickler auch mit vergleichsweise geringem Aufwand zwischen Projekten wechseln und ausgetauscht werden. Insbesondere für das schwer voraussagbare Aufkommen von Änderungswünschen ist es vorteilhaft, wenn die Ressourcenverteilung möglichst wenig an bestimmte Personen geknüpft ist.

- Durch die Stringenz im Vorgehen wird eine gute Vorhersagbarkeit von Aufwänden und Kosten bei Projektkalkulationen erreicht.

- Die Einheitlichkeit der Komponenten- und Programmiermodelle ist eine gute Basis für generative Ansätze (Model-driven Software-Development). Letztere bieten momentan zwar wegen des leicht anwendbaren Frameworks wenig Geschwindigkeitsvorteile in der Entwicklung, wohl aber eine deutliche Qualitätssteigerung.

Die Risiken bei dem geschilderten Vorgehen sind eine schleichende Überalterung des Frameworks und ein Verlust an Arbeitsqualität, da das Konzept der Vereinheitlichung von den Software-Entwicklern als massive Einschränkung ihrer Kreativität empfunden werden könnte. Beiden Punkten wird durch eine Reihe explizit etablierter Maßnahmen bei arvato entgegen gewirkt:

7.1 Gezielte Innovation

Jedes Anwendungsprojekt ist gehalten, eine Innovation auszuprobieren. Dabei ist es wichtig, dass dieser experimentelle Raum tatsächlich auf *eine* Innovation beschränkt ist, um sich nicht der Gefahr der Instabilität auszusetzen. In den jüngsten Projekten wurde beispielsweise die Umstellung des Build-Managements von Make auf Ant vollzogen, ein Versuch der Entwicklung von Webservices nach WSDL-First-Ansatz und die Anbindung von Rich-Clients an das Backend über eine generierte SOAP[26]-Kommunikationsschicht. Die Innovationen werden üblicherweise in einer regelmäßig stattfindenden Software-Architektenrunde abgesprochen, um sicher zu gehen, dass ein von allen Seiten als sinnvoll erachteter Ansatz verfolgt wird.

[26] Simple Object Access Protocol, siehe http://www.w3.org/TR/soap/

7.2 Mitwirkung am Framework

Da die Arbeit am Framework einen höheren Anspruch als die Anwendungsentwicklung hat, ist die Mitwirkung daran auch nicht jedem Entwickler gleichermaßen möglich. Andererseits hat es auch die Wirkung eines „Ritterschlags“, wenn eine bis dato rein als Anwendungsentwickler tätige Person eingebunden wird. Auch hier wirkt sich wieder positiv aus, wenn das Framework möglichst klein ist und sich auf diese Weise möglichst vielen Leuten erschließt. Ein indirektes Mittel der Mitwirkung ist ein für alle zugänglicher Bugtracker für die Sammlung von Feedback zum Framework. Besonders intensiv und konstruktiv berichtende Entwickler können sich hier gleichzeitig einen Namen machen und die Fortentwicklung des Frameworks mitbestimmen.

7.3 „Spiel“-Räume

Neben den projektbezogenen Maßnahmen hat es sich bewährt, eine unabhängige Vortragsreihe ohne konkreten Projektbezug ins Leben zu rufen, in der über den Tellerrand hinaus geschaut werden kann. Wegen des hohen Anteils externer Mitarbeiter, die in dieser Reihe ebenfalls aktiv tätig sind, werden auch bereichernde Ansätze aus ganz anderen „Denkfabriken“ in die Breite getragen. Die Reihe wird von allen für alle gestaltet, trägt sich also ausschließlich über das Interesse und Engagement der Mitarbeiter. Es gilt die Regel, dass im Rahmen der Veranstaltung selbst keine Entscheidungen getroffen werden, was schon wegen der überwiegend projektfinanzierten Organisation schwierig wäre. Wohl aber dienen die Informationen als Anregung für das weiter oben beschriebene Innovationswesen.

8 Zu guter Letzt…

Die Analogiebetrachtung hat im IT-Bereich von arvato direct services geholfen, das Wesen der eigenen Systemfamilienentwicklung besser zu verstehen und einzuordnen. Interessant ist zu erwähnen, dass die Betrachtung erst im Nachhinein beim „Pattern Mining“ festgestellt wurde, als es darum ging zu ergründen, welche spezifischen Dinge im Vorgehen man als Erfolgsfaktoren verstehen und erhalten muss. Für den aktuell vielerorts hoch gehandelten Begriff der *service-orientierten Architektur* (SOA) lässt sich übrigens keine passende Analogie finden – dementsprechend kritisch und eher abwartend werden die Konzepte zur Zeit bei arvato beobachtet. Hierbei ergeben sich Querabhängigkeiten zwischen Systemen unterschiedlicher Generationen, die eine nicht tolerierbare Fortschrittseinbuße bedeuten könnten. Davon abgesehen lässt sich die Analogie aber vermutlich zukünftig auch auf andere Bereiche ausdehnen. So gibt es im Hause eine Initiative, um über die feature-orientierte Modellierung (siehe [CZARNECKI]) den Gedanken der Systemfamilie auch bis in die Geschäftsmodellierung in Schwesterabteilungen zu tragen. Auch für die Verfolgung von Test-First-Strategien lassen sich hieraus Erkenntnisse ableiten.

Literaturverzeichnis

[DAHME] Ch. Dahme, W. Hesse: Editorial - Evolutionäre und kooperative Software-Entwicklung; Informatik-Spektrum 20.1, S. 3-4 (1997)

[FRAGNER] Fragner, J./Greiner, U./Vorauer, M. (Hg.): Menschenbilder. Zur Auslöschung der anthropologischen Differenz. Schriften der Pädagogischen Akademie des Bundes in Oberösterreich Bd. 15. Linz (2003)

[CHARGAFF] Erwin Chargaff: Das Feuer des Heraklit, Skizzen aus einem Leben vor der Natur, Klett-Cotta (2002)

[POHL] Klaus Pohl, u. a: Software Product Line Engineering, Springer Verlag (2005)

[LARMAN] Craig Larmann: Agile and Iterative Development, Addison-Wesley Professional (2003)

[CZARNECKI] Krysztof Czarnecki, Ulrich W. Eisenecker: Generative Programming, Addison-Wesley Professional (2000)

[FRAUN] Fraunhofer IESE, MARKET MAKER Software AG: Pressemitteilung, http://www.iese.fhg.de/Press_Media/2003/pm_pulse-cebit2003/ (2003)

Wiederverwendung von Entwicklungsprozessen durch Prozessfamilien

Maik Thränert[27]

Institut für Informatik, Universität Leipzig
Augustusplatz 10–11, 04109 Leipzig, Germany
E-Mail: thraenert@informatik.uni-leipzig.de

Andrej Werner[28]

Institut für Wirtschaftsinformatik, Universität Leipzig
Marschnerstr. 31, 04109 Leipzig, Germany
E-Mail: a.werner@wifa.uni-leipzig.de

Abstract: Ein Großteil der Entwicklungsprozesse wird oft für jedes Projekt aufwendig neu definiert. Eine Wiederverwendung des Prozesswissens über Projekte hinweg findet dabei bislang nicht in ausreichendem Maße statt. In diesem Papier werden deshalb die Grundlagen eines Konzepts vorgestellt, das unter Nutzung des Prozessfamilienansatzes eine projektübergreifende Wiederverwendung von Prozesswissen und ebenso die projektindividuelle Anpassung von Prozessen im Sinne eines Mass Customization erlaubt. Die grundsätzliche Annahme, damit das Konzept sinnvoll eingesetzt werden kann, ist dabei, dass genügend relativ ähnliche Projekte vorliegen und diese stark strukturiert ablaufen müssen.

Stichworte: Entwicklungsprozesse, Mass Customization, Process Factory, Vorgehen

1 Einleitung und Problemstellung

In vielen Unternehmen werden Entwicklungsprojekte durchgeführt, innerhalb derer Entwicklungsprozesse durchlaufen werden. Diese Entwicklungsprozesse orientieren sich an bekannten und dokumentierten Vorgehensweisen, unterscheiden sich jedoch von Projekt zu Projekt, bedingt durch den Charakter der relativen Einmaligkeit desselben, teilweise erheblich.

[27] Arbeitet im BMBF geförderten Projekt IE (Integration Engineering) mit.
[28] Arbeitet im BMBF geförderten Projekt PESOA (Process Family Engineering in Service-Oriented Applications) mit.

Diese Entwicklungsprozesse lassen sich in Kernprozesse, unterstützende Prozesse und Führungsprozesse unterscheiden. Innerhalb der Kernprozesse wird die eigentliche Wertschöpfung erzeugt. Diese wird dabei unternehmens-, projekt-, kunden- oder produktabhängig gestaltet. Unterstützende Prozesse – wie z. B. Abrechnung und Berichtswesen – hingegen sind über die Projekte hinweg relativ standardisiert. Führungsprozesse dienen der Steuerung und Entscheidung, z. B. im Rahmen des Projekt- und Multiprojektmanagements, und sind projektübergreifend organisiert.

Die eigentlichen Kernprozesse – wie z. B. Softwareentwicklung, Serviceentwicklung oder auch Software-Service-Co-Design [KTB05] – bieten ein hohes Potenzial zur Effizienzverbesserung. Diese Verbesserung lässt sich sowohl während der Projektvorbereitung – also dem Design des Entwicklungsprozesses – als auch während der Projektdurchführung – z. B. durch Standardisierung von Teilprozessen – erreichen. Analog zu Produktionsprozessen können so genannte Leitstandssysteme[29] für Entwicklungsprozesse in Form vom Prozessgesteuerten Software-Entwicklungsumgebungen (PSEU) [Neu02] eingesetzt werden, die eine prozessbasierte Projektsteuerung und Überwachung erlauben und darüber hinaus eine Durchgängigkeit zwischen den verschiedenen Aktivitäten – z. B. durch Transformation der entstehenden bzw. benötigten Artefakte untereinander – sicherstellen.

Die Synthese aus Massenfertigung und der Herstellung kundenindividueller Endprodukte ist in der Fertigungsindustrie unter dem Begriff Mass Customization bereits etabliert. Dieser Begriff verbindet dabei die an sich gegensätzlichen Begriffe „Mass Production“ und „Customization“. Durch die Anwendung von bestimmten Technologien und innovativen Organisationsstrukturen soll die Produktion variantenreicher und oft sogar kundenindividueller Produkte und Leistungen zu Kostensätzen der Massenfertigung bei uneingeschränkt hoher Produktqualität und kurzen Lieferzeiten möglich sein [Pin94]. Nach Pine stellt die Konzentration aus absatzbezogener Sicht auf den Kunden, der in seinen Augen ein individuelles Produkt erhält – sei dieses jetzt einzelgefertigt, modulgefertigt oder nur eine konfigurierbare Variation eines Standardproduktes – die weitgefassteste Begriffsinterpretation dar. Daraus folgt als Ziel der Mass Customization die Produktion von Gütern und Leistungen mit so viel Variabilität zu planen, dass fast jeder Kunde genau die von ihm gewünschte Lösung finden kann, wobei gleichzeitig die Produkte finanziell interessant sind.

Während Wiederverwendung und Individualisierung bezogen auf Funktionalität z. B. im Software-Bereich seit längerem eine wirtschaftliche Rolle spielen (Komponentenentwicklung, Produktlinienansätze[30], Service-orientierte Ansätze etc.), erfolgt die Wiederverwendung wie auch die projektspezifische Anpassung von Prozesswissen bislang nicht in ausreichendem Maße. Daraus resultiert ein hoher Anpassungsaufwand beim Design der Prozesse neuer Projekte. Der Entwicklungsprozess wird auf Basis des Vorgehensmodells teilweise komplett neu entwickelt, obwohl das zu realisierende Projekt viele gemeinsame Merkmale mit anderen Projekten teilt.

---

[29] Als Beispiel sei die Prozessleitstand-Software SimProQ unter http://www.simproq.de genannt, der in einem aktuellen Drittmittelprojekt um den Prozessfamiliengedanken erweitert wird.

[30] PuLSE unter http://www.software-kompetenz.de/?2197 und Software Product Lines unter http://www.sei.cmu.edu/productlines/index.html können hierfür als Beispiele aufgeführt werden.

In diesem Artikel werden deshalb die Grundlagen eines Konzepts zur Adaption von Entwicklungsprozessen aufgrund individueller Projektmerkmale vorgestellt. Gesamtziel ist dabei ein „Mass Customization“ im Sinne einer individuellen Bereitstellung von Entwicklungsprozessen mit einem hohen Anteil an Wiederverwendung von dokumentiertem Wissen über Entwicklungsprozesse. Erforderlich ist dafür das effiziente Management der Entwicklungsprozesskomponenten. Darüber hinaus ist für den sinnvollen Einsatz eines solchen Konzepts eine große Anzahl relativ ähnlicher Projekte erforderlich, die auch sehr stark strukturiert abgewickelt werden können. Die Prozessqualität bestimmt sich damit wesentlich aus der Strukturierung der Prozesse und nicht – wie z. B. in agilen Ansätzen – primär aus der Qualität der Mitarbeiter, die die Prozessqualität zur Laufzeit des Projekts liefern.

Bevor die Grundideen des Konzepts dargestellt werden, werden zunächst grundlegende Begrifflichkeiten im Umfeld von Vorgehens- und Prozessmodellen erläutert. Anschließend wird dargestellt, wie Prozesse auf Basis von Vorgehensmodellen definiert werden können. Die sich anschließende Darlegung der grundlegenden Ideen des Systemfamilienansatzes schafft die Basis für das Konzept der familienbasierten Adaption der Vorgehensmodelle. Ein Fallbeispiel und ein Ausblick schließen dieses Papier ab.

2 Grundlagen

2.1 Vorgehens- und Prozessmodelle

Die Erläuterung der folgenden Begriffe orientiert sich wesentlich am Verständnis der Fachgruppe WI-VM der Gesellschaft für Informatik, das z. B. im [KMO98] dokumentiert ist. Abbildung 1 zeigt einen Überblick über die hier verwendeten Begriffe und die zwischen ihnen bestehenden Beziehungen.

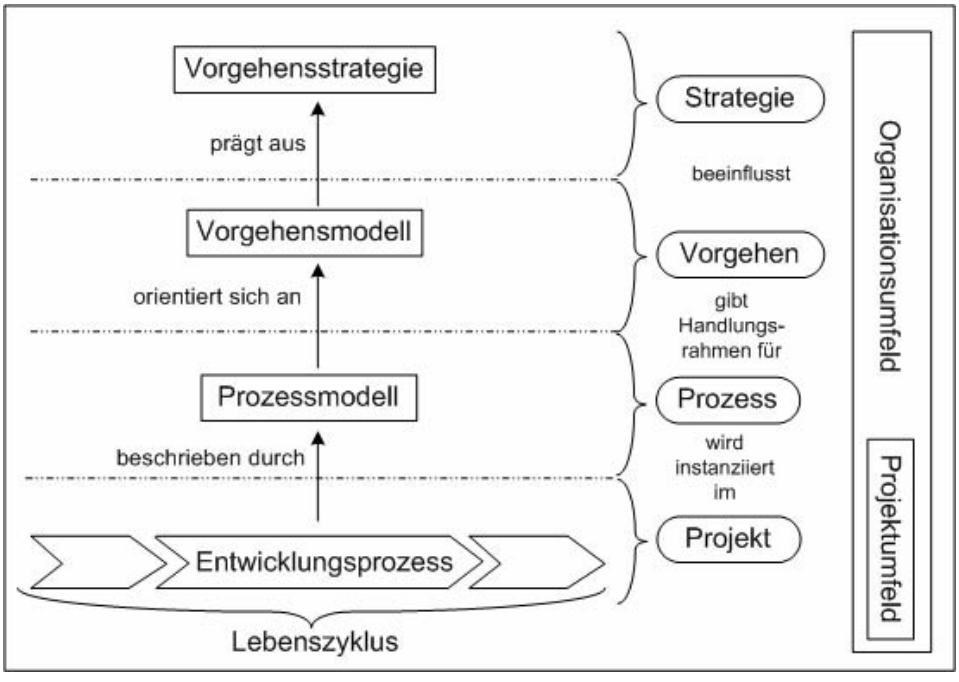


Abbildung 1: Grundlegende Begriffe

Die Entwicklungsprozesse – die eigentlichen Handlungsabläufe – werden innerhalb der Entwicklungsprojekte durchlaufen. Sie sind damit zur Laufzeit des Projekts von Bedeutung. Die Entwicklungsprozesse – und damit auch das Entwicklungsprojekt – sind Teil des Lebenszyklus des Gesamtproduktes, also z. B. Teil des Software-Lebenszyklus.

Die Entwicklungsprozesse können in einem Prozessmodell dokumentiert werden. Dieses Prozessmodell ist das Ergebnis des Prozessdesigns für das spezielle Entwicklungsprojekt und damit direkt abhängig von dessen Merkmalen und den Merkmalen des Projektumfeldes. Das Prozessmodell ist aufgrund der Spezifika des Projekts nur direkt in diesem Projekt verwendbar und ist – zumindest in Teilen – auch Teil des Projektplans. Es kann auch vorkommen, dass ein Prozessmodell im Vorfeld nicht genau dokumentierbar ist, da der genaue Prozessablauf sich erst zur Laufzeit des Projekts ergibt. Insbesondere bei agilen Prozessen kann dies der Fall sein.

Ein Vorgehensmodell ist das Referenzmodell der Entwicklungsprozesse bzw. der Prozessmodelle – es hat damit den Charakter einer Empfehlung [Bro03, S. 31ff]. Dazu beschreibt das Vorgehensmodell auf abstrakte Weise, in welchen Stadien sich das zu erstellende System befinden kann. Anders ausgedrückt: Das Prozessmodell wird aufgrund spezifischer Projektmerkmale auf Basis des Vorgehensmodells geformt – sei es vor oder während der eigentlichen Projektdurchführung.

Das Vorgehensmodell wird wiederum durch die grundlegenden Vorgehensstrategien beeinflusst. Das Vorgehensmodell ist somit eine Ausprägung der Vorgehensstrategie. Eine Vorgehensstrategie ist relativ abstrakt und beschreibt eine „Vision bzw. eine Vorstellung über die Anwendungsentwicklung“ [Bre98, S. 33]. Versionsgetriebener Ansatz oder risikogetriebener Ansatz sind Beispiele für Vorgehensstrategien.

2.2 Möglichkeiten zur Ableitung von Prozessmodellen

Um aus Vorgehensmodellen Prozessmodelle abzuleiten sind verschiedene Wege denkbar. Diese werden im Folgenden beginnend mit der einfachsten Lösung dargestellt. In der Realität werden sich aber auch Mischformen zwischen den vorgestellten drei Varianten ergeben.

Variante 1: Ein generisches Vorgehensmodell aus der Literatur, das dem Projektumfeld passend erscheint, wird an die Rahmenbedingungen der Organisation und der konkreten Projektanforderungen angepasst. Somit entsteht ein spezifisches Vorgehensmodell für die Organisation und den Projekttyp. Dieses spezifische Vorgehensmodell wird anschließend zu einem projektspezifischen Prozessmodell verfeinert. Diese Vorgehensweise wiederholt sich entsprechend bei jedem Projekt. Problematisch hieran ist jedoch, dass die Wiederverwendung nur auf dem hohen Abstraktionsniveau des generischen Vorgehensmodells erfolgt. Damit werden gleiche oder ähnliche Anpassungen evtl. mehrmals durchgeführt, vgl. die Darstellung in Abbildung 2.

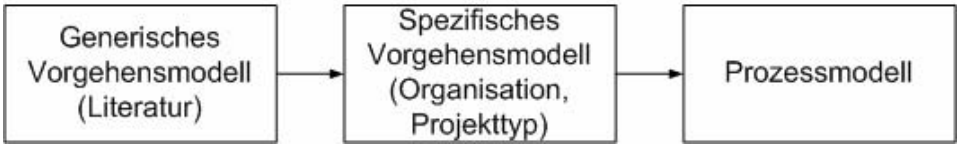


Abbildung 2: Variante 1 zur Ableitung von Prozessmodellen

Variante 2: Es wird ein für den Organisations- und Projekttyp geeignetes Referenzvorgehensmodell, das auf einem generischen Vorgehensmodell basiert, ausgewählt. Ein Referenzmodell hat den Anspruch der Allgemeingültigkeit und des Empfehlungscharakters für eine Klasse von Anwendungsfällen. Eine ausführliche Auseinandersetzung mit dem Begriff und den Merkmalen von Referenzmodellen ist in [Tho06, S. 12ff] zu finden. Dieses Referenzmodell wird an organisations- und projektspezifische Gegebenheiten angepasst. Gehört zum Referenzmodell auch ein Referenzprozess, dann kann dieser für das konkrete Projekt angepasst werden (Customization). Falls nicht, wird auf Basis des angepassten Referenzmodells ein Prozessmodell erstellt.

Das Referenzmodell (Vorgehens- und/oder Prozessmodell) kann als Ergebnis der aktuellen Projektarbeit durch den Referenzmodellkonstrukteur aufgrund der Erfahrungen des Referenzmodellnutzers um organisations- oder projektspezifische Eigenschaften erweitert werden und somit als Vorlage für weitere Projekte dienen. Auf diese Weise wächst die Qualität des Referenzmodells ständig mit der Durchführung weiterer Projekte.

Die Wiederverwendung in dieser Variante ist gegenüber der ersten Variante höher, da spezifische Eigenschaften bereits in Form von Referenzmodellen dokumentiert sind, vgl. die Darstellungen in Abbildung 3. Diese Variante ist dann geeignet, wenn ein Unternehmen bereits über umfangreiches Know-how in der Gestaltung der Entwicklungsprozesse im Problembereich verfügt und dieses Know-how in Form des Referenzmodells dokumentiert ist bzw. dokumentiert werden soll.

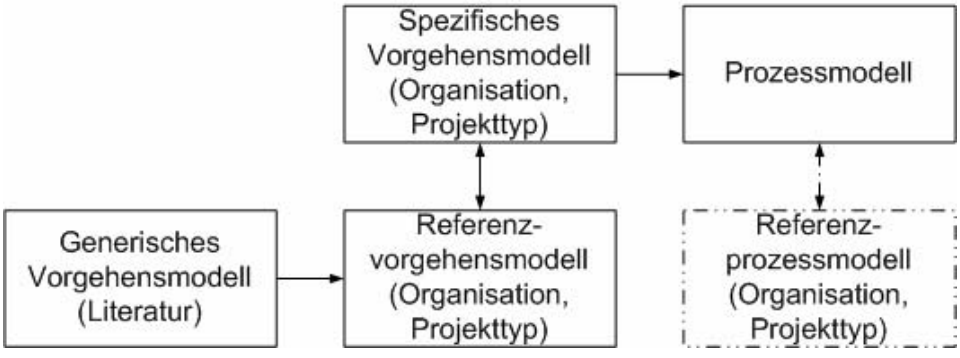


Abbildung 3: Variante 2 zur Ableitung von Prozessmodellen

Variante 3: Hier wird beginnend von einem für den Organisations- und Projekttyp geeigneten Referenzvorgehensmodell – das wiederum auf einem generischen Vorgehensmodell basiert – die Anpassung an organisations- und projektspezifische Gegebenheiten begonnen. Steht ein Referenzprozess zur Verfügung, dann wird dieser entsprechend angepasst (Customization). Falls nicht, muss das Referenzvorgehensmodell als Ausgangsbasis der Anpassungen eingesetzt werden. Bislang ist der Ablauf gleich zu Variante 2.

Im nächsten Schritt werden jedoch die Projektmerkmale identifiziert, die auf die Anpassung des Prozessmodells für das Projekt Einfluss haben. Die gemeinsamen und unterschiedlichen Merkmale werden in einem Merkmalmodell dokumentiert. Durch diese Variantenplanung entsteht eine Prozessmodellfamilie. Mit jedem Projekt erhöht sich somit die Wissensbasis. Die Wahrscheinlichkeit steigt, dass bestimmte Entwicklungsprozesselemente (Prozessbausteine), die sich vom Referenzmodell unterscheiden, in einem weiteren Entwicklungsprozess Anwendung finden können. Voraussetzung ist, dass Merkmale eines Prozessbausteines mit Merkmalen eines Projektes in Beziehung stehen. Diese Variante ist in Abbildung 4 dargestellt.

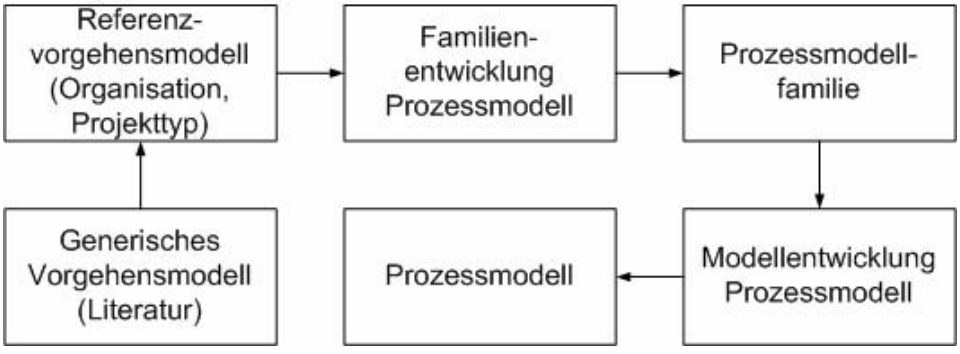


Abbildung 4: Variante 3 zur Ableitung von Prozessmodellen

Mit diesem Ansatz wird der Entwicklungsprozess – ausgehend von einem generischen oder Referenzmodell – nicht immer wieder neu beschrieben, sondern gemeinsame und variierende Prozessbausteine werden wieder verwendet. Man hat es so mit einer Konfiguration des Prozessmodells zu tun. Das Ergebnis ist ein Entwicklungsprozess, der aufgrund von bereits existierenden Prozessbausteinen konfiguriert wurde. Bislang nicht passende Prozessbausteine werden erstellt und der Wissensbasis hinzugefügt, um diese zukünftig wieder verwenden zu können. Der Grad der Wiederverwendung ist höher als bei Variante 2, da sowohl Prozessbausteine als auch Konfigurationswissen wieder verwendet werden. Dadurch wird das Wissen des Modellkonstrukteurs expliziert und der Ablauf ist insgesamt systematischer. Die Kosten für die Individualisierung eines Entwicklungsprozesses sind dabei am geringsten. Durch die Wiederverwendung des Prozesswissens erhöht sich auch dessen Qualität, die Reife steigt. Dieses aggregierte Wissen ist auch handel- bzw. austauschbar.

Im Sinne eines Reifegradmodells bauen die drei Varianten der Wiederverwendung aufeinander auf und zeigen den Reifegrad der Prozesse. Es ist nicht möglich, direkt mit der dritten Variante zu beginnen, ohne vorher ein Referenzmodell erstellt zu haben – sei es explizit oder nur mental durch den Modellkonstrukteur. Die Wiederverwendung des Prozesswissens und der Dokumentationsgrad steigen von Variante zu Variante an.

Die dritte Variante wird ausführlich in Kapitel 3 vorgestellt. Im Nachfolgenden werden die notwendigen Grundlagen dieses Familiengedankens erläutert.

2.3 Systemfamilien

Die Domänenentwicklung (Domain Engineering) wurde von dem Gedanken der effektiven Wiederverwendung von Software, insbesondere von Systembestandteilen, in einer Domäne beflügelt [Pri90, WPD92, Ara94, CE00, Cle00, CN02 et al.]. Ausgehend von den Zielen der Interessenbeteiligten beschreibt die Anforderungsanalyse die gemeinsamen und unterschiedlichen Systembestandteile der untersuchenden Domäne. Diese stellen die Grundlage für eine Systemfamilie dar und beeinflussen insbesondere die Verständlichkeit, Wartbarkeit, Nutzbarkeit und Wiederverwendbarkeit eines Systems bzw. einer Systemfamilie [For96].

Eine Domäne ist der Wissens- oder Anwendungsbereich, welcher von den Interessenbeteiligten (Stakeholders) definiert wird. [CE00] charakterisiert sie durch nachfolgende Punkte:

- die Eingrenzung (Scoping) des Betrachtungsbereiches auf die bestmögliche Überlappung der Anforderungen der Interessenbeteiligten,
- eine Domäne beherbergt eine Menge an Konzepten und Begriffen, welche von Praktikern auf diesem Gebiet verstanden werden,
- sowie das Wissen, wie man Systeme bzw. Komponenten davon erstellt.

Eine Systemfamilie wird nach [CE00] charakterisiert als eine Menge von Systemen, die aufbauend auf einer Menge gemeinsamer Komponenten viele gemeinsame Merkmale besitzen. Das Ziel der Domänenentwicklung ist, durch den hohen Grad an Wiederverwendung von wichtigen Systembestandteilen (Assets) neue Systeme oder neue Systemvarianten schnell, kostengünstig und mit hoher Qualität entwickeln zu können. Auf der Domänenentwicklung aufbauend beginnt die Anwendungsentwicklung, wo eine konkrete Variante der Systemfamilie betrachtet und entwickelt wird. Die gewonnenen Erfahrungen im Verlauf einer iterativen Entwicklung fließen in die Weiterentwicklung der Systemfamilie ein.

Die Unterschiede zwischen Systemen einer Systemfamilie entstehen durch verschiedene Anforderungen an einzelne Systemausprägungen. Die Anforderungen an die Systemeigenschaften können auf einer hohen Abstraktionsebene als Merkmale der Systeme interpretiert werden. Dabei bilden die Merkmale, die alle Systeme der Systemfamilie gemeinsam haben, die Gemeinsamkeiten. Die Merkmale, die von System zu System variieren, bilden die Variabilität der Systemfamilie. Die Merkmalmodellierung stellt diese Aspekte in einem Modell dar. Das Konzept der Merkmalmodellierung wurde erstmals im Rahmen von FODA [KCH+90] entwickelt und durch weitere Arbeiten, z. B. von [CE00, S. 82 ff.], erweitert. Ein Merkmal stellt nach FODA eine für den Benutzer deutlich sichtbare Eigenschaft oder Charakteristika eines Systems dar [KCH+90, S. 3]. Einer der grundlegenden Kompromisse, die ein Systemarchitekt zu lösen hat, ist die Entscheidung, wann ein Merkmal „eingebunden“, d. h., zu welchem Zeitpunkt das Merkmal ausgewählt wird, oder ein Merkmal „geprägt“ wird, d. h. einen Wert erhält. In FODA wurde dazu der so genannte Bindungszeitpunkt (binding time) für ein Merkmal eingeführt und drei Zeitpunkte – Compile-time, Load-time oder Run-time – beschrieben [KCH+90].

Merkmale werden nach [GBS00, S. 3] in vier verschiedene Kategorien klassifiziert. Externe Merkmale sind keine direkten Bestandteile des Systems sondern der Zielplattform. Die notwendigen Merkmale stellen die Gemeinsamkeiten der Systeme dar und müssen in jedem Fall eingebunden werden. Optionale Merkmale können, müssen aber nicht, eingebunden werden und erhöhen bei ihrer Einbindung die Funktionalität eines Systems. Als letztes dienen die alternativen Merkmale zur Auswahl einer Merkmalsausprägung aus einer Gruppe von Merkmalen. Das Auswahlkriterium, das die Wahl zwischen den alternativen Merkmalen ermöglicht, wird als Variationspunkt bezeichnet.

Die Merkmalmodelle bestehen aus einem oder mehreren Merkmaldiagrammen und ergänzenden Informationen. Die Merkmaldiagramme dienen zur grafischen Darstellung der Merkmale in Form von Strukturbäumen. Textuelle Kompositionsregeln (Composition Rules) dienen der Beschreibung von Querbeziehungen, die nicht in einem Merkmaldiagramm darstellbar sind.

Im Rahmen der RSEB-Methode führt Jacobson et al. in [JGJ97, S. 98] den Begriff des Variationspunktes ein. Er dient zur Darstellung der Stellen, an denen Unterschiede in einer Gruppe ähnlicher Systeme auftreten können. In der Merkmalmodellierung dient ein Variationspunkt zur Darstellung der Auswahlmöglichkeit alternativer oder optionaler Merkmale. Während die Merkmalmodelle zur implementierungsunabhängigen Darstellung von Systemeigenschaften dienen, ermöglichen Variationspunkte die Berücksichtigung von Variabilität in jeder Phase des Entwicklungsprozesses (von der Anforderungsanalyse über das Design bis zur Implementierung). Bühne et al. identifizierten in [BHL+04, S. 14–16] das Auftreten von Variabilität in Merkmalen, in Abläufen, im Datenumfang, im Datenformat, im Systemzugang, in Benutzerschnittstellen, in Systemschnittstellen und in der Qualität. Variabilität in Abläufen bedeutet, dass eine Funktionalität zweier Produktausprägungen durch eine voneinander abweichende Reihenfolge von Tätigkeiten realisiert werden kann.

Wie erläutert, stellt das Konzept der Systemfamilien eine Möglichkeit der Wiederverwendung von Software-Assets dar. Diese Assets bestehen primär aus Software-Artefakten (z. B. Komponenten, Sourcecode usw.) oder dem Wissen zur Herstellung von Software-Artefakten (z. B. Dokumentation, Anforderungen usw.). Der Entwicklungsprozess selber stellt ebenfalls einen signifikanten Wettbewerbsfaktor dar. Sutton et al. stellten in [SO97] fest, dass Vorgehen Einfluss auf die Software-Produktentwicklung haben und aus ihnen konkrete Entwicklungsprozesse abgeleitet werden. Variiert innerhalb dieses Vorgehens der Entwicklungsprozess, so variiert auch das Ergebnis der Entwicklung und umgekehrt. Weiterhin wurde in [SO97] beschrieben, dass Vorgehen keinen starren Entwicklungsprozess für Softwareprodukte vorschreiben, sondern eher einen Handlungsrahmen darstellen, in dem der individuelle Entwicklungsprozess abläuft. Diese individuellen Entwicklungsprozesse variieren in Abhängigkeit verschiedener Kriterien (Projekttyp, Produkt, Personal, Ressourcen usw.).

Daher verstehen Sutton et al. einen konkreten Entwicklungsprozess als Mitglied einer Prozessfamilie, die in Bezug auf die genannten Kriterien eine Vielzahl von Optionen beherbergt. Aufbauend auf diesen Überlegungen und den Arbeiten zur Process Factory [Wer05], wo die organisatorische Flexibilität, also das Variantenmanagement von individuellen Geschäftsprozessmodellen aus Geschäftsprozessfamilien Betrachtung findet, wird im folgenden Kapitel ein Konzept zur Individualisierung von Entwicklungsprozessen aus Prozessfamilien vorgestellt.

3 Familienansatz für Prozessmodelle

Um den vorgestellten Familienansatz für Prozessmodelle verwenden zu können, sind mehrere Aktivitäten erforderlich, vgl. Abbildung 5. Zunächst sind Prozessmodellfamilien zu entwickeln, was dem Konzept des Domain Engineering entspricht. Als Ergebnis werden die Merkmale von Prozessfamilien erfasst und in Form von Merkmalmodellen und Konfigurationsmodellen im Prozessmodellfamilien-Repository abgelegt. Ausgangsbasis für dieses Domain Engineering sind entweder eine Analyse vergangener Projekte oder das Know-how des „Domain Engineers".

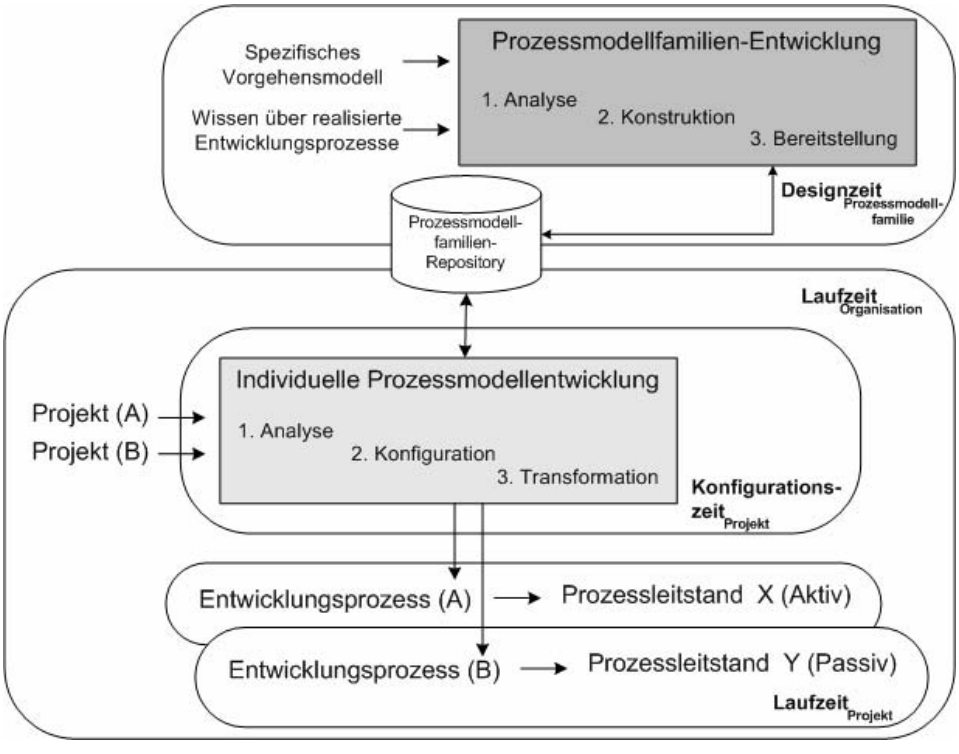


Abbildung 5: Wiederverwendung von Entwicklungswissen durch Prozessfamilien

Auf Basis des Prozessmodellfamilien-Repositories erfolgt die projektindividuelle Prozessmodellentwicklung. Hierfür sind zunächst Projektmerkmale zu analysieren (Analyse) und die Prozessfamilie anschließend geeignet zu konfigurieren (Konfiguration) bevor schließlich ein geeignetes Prozessmodell erzeugt werden kann (Transformation). Diese Aktivitäten zählen zur Konfigurationszeit des Projekts.

Während der **Analyse** wird das Projektumfeld untersucht und aus dem zur Verfügung stehenden Prozessmodellfamilien-Repository eine geeignete Prozessmodellfamilie in Bezug auf Vorgehen und Strategie (vgl. Abbildung 5) ausgewählt. Diesen Vorgang bezeichnet man auch als Scoping. Anschließend erfolgen eine Analyse des Projekts aufgrund definierter Projektmerkmale und die Dokumentation in Form eines projektspezifischen Merkmalmodells. Als Ergebnisse entstehen die Auswahl einer geeigneten Prozessmodellfamilie und die Charakterisierung der Projektanforderungen durch das Merkmalmodell.

Während der **Konfiguration** wird der Abgleich des Projektmerkmalmodells mit den Merkmalen der Prozessmodellfamilie durchgeführt. Es werden Prozessbausteine identifiziert, die in dem konkreten Entwicklungsprozess verwendet werden. Weiterhin können Prozessbausteine mit Hilfe von Merkmalausprägungen parametrisiert werden. Bislang nicht vorhandene und im Prozess benötigte Prozessbausteine sind zu erstellen. Das Ergebnis der Konfiguration ist ein konkreter individueller Entwicklungsprozess, ausgedrückt in Form eines Prozessmodells.

Die anschließende **Transformation** ermöglicht die Erstellung eines konkreten, individuellen Entwicklungsprozessmodells in der Sprache des Prozessleitstands, so dass eine Instanziierung des Prozessmodells vorgenommen werden kann, also das Projekt ausgeführt wird. Als Input werden hierfür das während der Konfiguration entstandene Entwicklungsprozessmodell, Beschreibungen der Zielsprache sowie Transformationsvorschriften und zusätzliche, zur Instanziierung erforderliche Angaben zu Projektspezifika (Ressourcen, Namen, Adressen etc.) benötigt. Bei einer aktiven Unterstützung durch den Prozessleitstand wird der Entwicklungsprozess geeignet deployed und anschließend ausgeführt (instanziiert). Der Prozess wird abgearbeitet, d. h. der Prozessleitstand steuert aktiv das Projekt mit Ressourcen, Mitarbeitern, Dokumenten usw. Die Zustände der Aktivitäten werden durch den Prozessleitstand selbst erfasst. Alternativ wird bei einem passiven Prozessleitstand der Entwicklungsprozess lediglich zur Dokumentation des Projekts verwendet. Die Steuerung des Projekts liegt dann beim Projektleiter. Zustände der Aktivitäten im Entwicklungsprozess werden durch die Mitarbeiter erfasst und dokumentiert.

4 Fallbeispiel

Als Fallbeispiel für das vorgestellte Konzept soll ein CRM-Dienstleister dienen. Dessen Kerngeschäft ist die Einführung und Anpassung von CRM-Anwendungssoftware für Kunden aus dem Maschinenbau. Diese Fokussierung auf eine Kundenbranche hat den Vorteil, dass sich Projekte dieser Art aus Sicht des Dienstleisters relativ ähnlich zueinander verhalten und ein sehr strukturiertes, im Vorfeld planbares, Vorgehen möglich ist. Dieses sehr strukturierte Vorgehen sichert die Qualität durch den Prozess selbst. Durch die nicht geringe Menge an solchen Projekten ist es aus Sicht des Dienstleisters damit sinnvoll, eine projektspezifische Anpassung des Prozesses umzusetzen.

Deshalb wurde zunächst – analog zu dem in Kapitel 3 beschriebenen Vorgehen – eine Prozessmodellfamilie entwickelt, vgl. [WS05, S. 28]. Dazu gehört auch eine Sammlung von Merkmalen und Variationspunkten der Prozessmodellfamilie und deren Dokumentation in Form eines Merkmalmodells. Ein solches Merkmalmodell ist – exemplarisch für zwei Merkmale – in Abbildung 6 dargestellt. Die Variantenplanung hat bislang allerdings nur die Eigenschaften des Projekts betrachtet. Sinnvollerweise müsste die Erfassung der prozessrelevanten Merkmale auch auf die Bereiche Personal, Ressourcen etc. ausgedehnt werden.

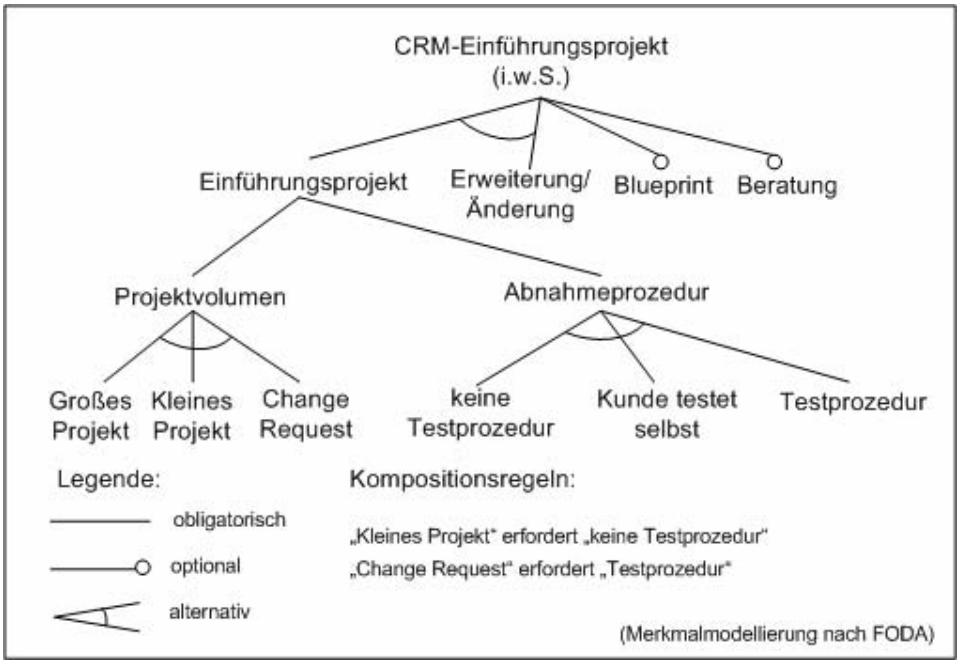


Abbildung 6: Exemplarisches Merkmalmodell für das Fallbeispiel

Zunächst ist zu unterscheiden, ob es sich bei dem betrachteten Projekt um ein Einführungsprojekt oder ein Erweiterungs- bzw. Änderungsprojekt handelt. Blueprint- und Beratungsprojekte können zusätzlich zum Einführungs- oder Erweiterungsprojekt durchgeführt werden. Um das Beispiel kompakt zu halten, wird im Folgenden nur auf Einführungsprojekte eingegangen.

In einem solchen Einführungsprojekt spielen Projektvolumen und die Abnahmeprozedur der Ergebnisse am Ende des Projekts eine wichtige Rolle für den Ablauf des Projekts. Handelt es sich um ein „kleines Projekt" fallen Projektvorbereitung und Abnahme schlanker aus als bei einem „großen Projekt". Bei „kleinen Projekten" und „Change-Requests" fällt die interne Freigabe vollständig weg. Abhängig von der konkreten Ausprägung dieses Merkmals sind also mehr oder weniger Ressourcen für die Projektvorbereitung und die Abnahme einzuplanen. Eventuell kann die komplette Aktivität „interne Freigabe" auch vollständig entfallen.

Die Eigenschaft „Abnahmeprozedur“ legt fest, wie die Art der Tests beim Kunden am Ende der Implementierung erfolgen soll. Möglichkeiten sind hier, dass 1.) keine Abnahme durch den Dienstleister selbst erfolgt; 2.) der Kunde selbst testet und eine Liste von Restpunkten an den Dienstleister gibt oder 3.) ein gemeinsamer Test von Kunden und Dienstleister erfolgt. Abhängig von der Ausprägung dieses Merkmals ändert sich die Gestaltung der Phase „Abnahme“ im Vorgehen. Hier sind die Rollen (Dienstleister, Kunde) entsprechend zuzuordnen bzw. die Phase insgesamt wegzulassen.

Um nun aus dem oben erstellten Prozessfamilienmodell ein konkretes Prozessmodell für das Projekt erstellen zu können, sind zunächst die Eigenschaften des konkreten Projekts zu analysieren und anhand des Merkmalmodells zu dokumentieren. Diese Eigenschaften beeinflussen dann – an den definierten Variationspunkten – die Ausprägung des konkreten Projektmodells in der oben beschriebenen Art.

Der beschriebene Ansatz kann in dem vorgestellten Bereich sinnvoll eingesetzt werden. Dies liegt nicht zuletzt an der Charakteristik des Problembereichs: Man hat es hier mit relativ ähnlich gearteten Projekten zu tun, sodass ein strukturiertes Vorgehen in den Projekten selbst möglich ist. Darüber hinaus sind die konkreten, das Vorgehen bestimmenden Merkmale, klar definiert und im Vorfeld der Projektdurchführung bekannt.

5 Kritik und Ausblick

Das vorliegende Papier konnte lediglich die Grundideen des Konzepts vorstellen. Insgesamt sind noch folgende Arbeiten zu leisten:

- Ausbau von Merkmalmodellen und Konfigurationsmodellen: Es ist sehr schwierig, diese Modelle aufzubauen, da hierfür ein tiefes Fachwissen über Prozesse in Projekten und deren Abhängigkeit von Projektmerkmalen und Merkmalen des Projektumfelds erforderlich ist. Darüber hinaus stellt sich hier das Problem der „Willigkeit“ der entsprechenden Mitarbeiter, dieses hochwertige Wissen zu explizieren.

- Granularität der Wiederverwendung: Es ist zu prüfen, wie granular Prozessbausteine zu definieren sind, damit eine sinnvolle Wiederverwendung erfolgen kann. Bei zu groben Elementen sind die entstehenden Prozessmodelle nicht projektadäquat produzierbar; sind die Elemente zu fein, steigt der Aufwand bei der Merkmalmodellierung und der Definition der Abhängigkeiten stark an.

- Praktische Relevanz: Das Konzept ist anhand geeigneter Anwendungsfälle weiter praktisch zu erproben und zu detaillieren. Daraus kann abgeleitet werden, welche Kriterien geeignete Anwendungsfälle erfüllen müssen, damit das Konzept sinnvoll eingesetzt werden kann. Voraussetzung ist dabei, Anwender zu finden, die genügend ähnliche Projekte bearbeiten, damit eine sinnvolle Modellierung und ein sinnvoller Einsatz des Konzepts erreicht werden kann. In dem in Kapitel 4 vorgestellten Fallbeispiel ist dies so gegeben. Jedoch müssen hier weitere praktische Erfahrungen gesammelt werden, um die Nützlichkeit des Konzepts zu überprüfen.

Das hier vorgestellte Konzept der Wiederverwendung von Prozesswissen versucht, eine Wiederverwendung bei der Umsetzung des Prozessmodells auf den Entwicklungsprozess (vgl. Abbildung 5) zu erreichen, indem die Merkmale von Prozessmodellen untersucht werden. Es ist jedoch ebenso möglich, eine Wiederverwendung bei der Umsetzung des Vorgehensmodells auf das Prozessmodell, also eine Ebene höher, zu erreichen. Es wären dann die Merkmale von Vorgehensmodellfamilien zu analysieren und deren Komponenten geeignet wieder zu verwenden. Ebenso ist es möglich, das Konzept auf beiden Ebenen gleichzeitig einzusetzen.

6 Literaturverzeichnis

[Ara94] Arango, G.: Domain Analysis Methods. In (Schäfer, R.; Prieto-Díaz, M; Ellis, M. Hrsg.): Software Reusability, Ellis Horwood, 1994.

[BHL+04] Bühne, S.; Halmans, G.; Lauenroth, K.; Pohl, K.: Variabilität in Software-Produktlinien. In (Böckle, G.; Knauber, P.; Pohl, K.; Schmid, K., Hrsg.): Software-Produktlinien – Methoden, Einführung und Praxis. dpunkt-Verlag, Heidelberg, 2004.

[Bre98] Bremer, G.: Genealogie von Entwicklungsschemata. In: (Kneuper R. et. al. Hrsg.): Vorgehensmodelle für die betriebliche Anwendungsentwicklung. Teubner-Verlag, 1998.

[Bro03] Brocke, J. v.: Referenzmodellierung – Gestaltung und Verteilung von Konstruktionsprozessen. Logos-Verlag, Berlin, 2003.

[CE00] Czarnecki, K.; Eisenecker, U.: Generative Programming: Methods, Tools and Applications. Addison-Wesley, Boston, 2000.

[Cle00] Cleaveland, C.: Program Generators using Java and XML. Prentice Hall, New Jersey, 2000.

[CN02] Clements, P.; Northrop, L.: Software Product Lines: Practices and Patterns. Addison-Wesley, Boston, 2002.

[FBM98] Fischer T.; Biskup H.; Müller-Luschnat G.: Begriffliche Grundlagen für Vorgehensmodelle. In: (Kneuper, R. et. al. Hrsg.): Vorgehensmodelle für die betriebliche Anwendungsentwicklung. Teubner-Verlag, 1998.

[For96] Foreman, J.: Product Line Based Software Development – Significant Results, Future Challenges. Software Technology Conference, Salt Lake City, 1996.

[GBS00] Gurp, J.; Bosch, J.; Svahnberg, M.: Managing Variability in Software Product Lines. 2000, http://www.jillesvangurp.com/publications/managingvariabilityinSPLs.pdf, abgerufen am 10.10.2004.

[JGJ97] Jacobson, I.; Griss, M.; Jonsson, P.: Software Reuse – Architecture, Process and Organization for Business Success. Addison-Wesley, 1997.

[KCH+90] Kang, K.; Cohen, S.; Hess, J.; Novak, W.; Peterson, S.: Feature-Oriented Domain Analysis (FODA) Feasibility Study. CMU/SEI-90-TR-21. Software Engineering Institute, Carnegie Mellon University, Pittsburgh 1990.

[KMO98] Kneuper, R.; Müller-Luschnat, G.; Oberweis, A. (Hrsg.): Vorgehensmodelle für die betriebliche Anwendungsentwicklung, . B. G. Teubner, Stuttgart/Leipzig, 1998.

[KTB05] Kühne, S.; Thränert, M.; Böttcher, M.: A tool approach for supporting software service co-design. In (Backlund, P.; Carlsson, S.; Söderström, E., Hrsg.): BIR 2005 – Proceedings of the 4th International Conference on Business Informatics Research (BIR) 2005.

[Neu02] Neumann, O.: Integration von Werkzeugen in heterogene, prozessgesteuerte Software-Entwicklungsumgebungen – Tool Integration Concept. Dissertation, Universität-Gesamthochschule Paderborn, 2002.

[Pin94] Pine, B. J.: Maßgeschneiderte Massenfertigung: Neue Dimensionen im Wettbewerb. Wirtschaftsverlag Ueberreuter, Wien, 1994.

[Pri90] Prieto-Díaz, R.: Domain Analysis: An Introduction. Software Engineering Notes, 15(2), 1990, S. . 47–54.

[SO97] Sutton, S. M.; Osterweil, L. J.: Product Families and Process Families. Proceedings of the 10th International Software Process Workshop (ISPW '96), IEEE (2)97, S. 109–111.

[Tho06] Thomas, O.: Das Referenzmodellverständnis in der Wirtschaftsinformatik: Historie, Literaturanalyse und Begriffsexplikation. Veröffentlichungen des Instituts für Wirtschaftsinformatik Saarbrücken, Heft 187, Januar 2006.

[Wer05] Werner, A.: Fertigung kundenindividueller Geschäftsprozesse mittels einer Process Factory. Lectures Notes in Informatics (LNI) - Proceedings GI-Informatik 2005, Volume P-67, 2005, S. . 64–68.

[WPD92] Wartik, S.; Prieto-Diaz, R.: Criteria for Comparing Reuse-Oriented Domain Analysis Approaches. International Journal of Software Engineering and Knowledge Engineering, 2(3), 1992, S. 403–431.

[WS05] Wetzel, P.; Soutchilin, A.: Integration Engineering im Maschinen- und Anlagenbau. In: (Fähnrich, K. P. et. al., Hrsg.): Integration Engineering – Umsetzung von kooperativen Geschäftsprozessen auf eine internetbasierte IT-Struktur. Universität Leipzig, 2005.

Projektaudit: Eine Methode zur kurzfristigen Standortbestimmung

Randolph Kappes

Geschäftsbereich IT-Beratung
sd&m AG
software design & management
Mülheimer Straße 9a
53840 Troisdorf
kappes@sdm.de

Abstract: Die Anforderungen an die Anwendungsentwicklung – insbesondere aufgrund neuer Trends in der Software-Industrie – wachsen weiter an. Gerade innovative Projekte geraten leicht in schwierige Situationen: Sie überschreiten die veranschlagten Zeit- und Kostenrahmen oder liefern nicht die gewünschte Funktionalität. Eine Beratung in Form eines Projektaudits ermöglicht die Bewertung einer solchen Projektsituation. Jedes Softwareentwicklungsprojekt folgt einem methodischen, strukturierten Vorgehen. Ein methodisch strukturiertes Vorgehen auch im Projektaudit ist Basis für eine erfolgreiche Untersuchung. Das Tutorial vertieft die Methodik und stellt die Erfolgsfaktoren eines Projektaudits anhand eines Projektbeispiels vor.

Inhalt des Tutorials

Wenn ein Projekt eine Verzögerung meldet, ist es ohne interne Kenntnisse der Projektsituation schwierig, die richtigen Handlungsempfehlungen auszusprechen. Auch die in das Projekt involvierten Manager sind meist schon „betriebsblind“ und können die notwendigen Handlungsempfehlungen nicht erkennen.

Die Verantwortlichen in der Anwendungsentwicklung oder die Geschäftsführung beauftragen oft in einer solchen Situation IT-Experten, die nicht in das Projekt involviert sind, um das Projekt zu bewerten und um eine unvoreingenommene Meinung zu notwendigen Handlungsempfehlungen zu bekommen.

Jedes Softwareentwicklungsprojekt folgt einem methodischen, strukturierten Vorgehen. Ein methodisch strukturiertes Vorgehen auch im Projektaudit ist neben der fundierten Erfahrung der projektexternen Experten die Basis für eine erfolgreiche Untersuchung. Nur so können die notwendigen Maßnahmen und Handlungsempfehlungen zur Sanierung in der zur Verfügung stehenden Zeit geliefert werden.

Das Tutorial vertieft die Methodik und stellt die Erfolgsfaktoren eines Projektaudits anhand eines Projektbeispiels vor.

Neben dem erweiterten Vortrag werden in der Gruppe zwei Themen erarbeitet:

- Anhand eines Projektbeispiels wird ein konkreter Untersuchungsgegenstand abgeleitet und Normen zur Objektivierung der Bewertung diskutiert.

- In einem zweiten Schritt werden Hypothesen bzgl. des Ergebnisses gestellt und entsprechende Schlüsselfragen gemeinsam entwickelt.

Die Übung erfolgt an einem realen Projektbeispiel, entweder von einem Teilnehmer der Gruppe oder dem Tutorialleiter vorgestellt.

Über den Tutorialleiter

Randolph Kappes, studierte Informatik an der Universität in Saarbrücken. Seine Berufslaufbahn begann er 1991 als Softwareentwickler, Anwendungsarchitekt und Projektleiter. Seit 9 Jahren ist er bei sd&m tätig als Projektmanager in Softwareentwicklungsprojekten. In den letzten Jahren lag sein Schwerpunkt in der IT-Beratung im Umfeld Projektmanagement, Anwendungsarchitekturen und Wirtschaftlichkeit von IT-Vorhaben.

WirtschaftsMediation im Softwaremanagement: Konflikte produktiv lösen

Axel R. Raulinat

TENOS Aktiengesellschaft
Konfliktberatung Konfliktlösung
Konfliktprävention
Neuer Wall 35
D-20354 Hamburg
Axel.Raulinat@Tenos.de

Hans Brandt-Pook

Fachbereich Produktion und Wirtschaft
FH Lippe und Höxter

Liebigstr. 87
D-32657 Lemgo
Hans.Brandt-Pook@FH-LuH.de

Beschreibung des Tutorials

Bekanntermaßen scheitern viele IT-Projekte aus unterschiedlichen Gründen – nicht selten an mangelnder Kommunikation. Als Konsequenz des Scheiterns, und auch schon während der Projektdurchführung, sind Konflikte zwischen den Projektparteien an der Tagesordnung, bspw.:

- Wer trägt die Verantwortung für den Misserfolg?
- Wie kann aus der Situation das Beste gemacht werden?
- Wie soll die zukünftige Zusammenarbeit der Projektparteien aussehen?

Ein moderner Ansatz zur Lösung solcher Konflikte im Geschäftsleben ist die WirtschaftsMediation. Mediation ist im Wesentlichen eine Verhandlung zwischen Geschäftspartnern in einer konfliktbeladenen Situation mit einigen Besonderheiten und Vorteilen. Das Besondere an Mediationsverhandlungen besteht darin, dass sie von einem speziell ausgebildeten neutralen Experten – dem Mediator – nach einem strukturierten Verfahren moderiert werden. Die Vorteile einer Mediation sind u.a.:

- Das Verfahren ist im Vergleich zu einer gerichtlichen Auseinandersetzung sehr schnell
- Die Verhandlungen sind freiwillig und vertraulich – nichts dringt ohne Zustimmung der Parteien an die Öffentlichkeit
- Es gibt eine hohe Erfolgsquote

- Ein Mediationsverfahren ist kostengünstig im Vergleich zu einer gerichtlichen Auseinandersetzung.

Durch eine WirtschaftsMediation werden die Ursachen eines Konfliktes benannt und mithilfe des Mediators klare, rechtsverbindliche Lösungen gefunden. Insbesondere im Vergleich zu einer juristischen Auseinandersetzung ist ein Mediationsverfahren schnell, wirtschaftlich und zukunftsorientiert. Die Erfolgsquote liegt bei 70 bis 90 %.

In dem Tutorial soll dieser Problemkreis aufgearbeitet werden. Es wird aufgezeigt, wie die WirtschaftsMediation im Softwaremanagement erfolgreich angewendet werden kann. Dabei wird auch auf die besondere Rolle des Beraters/der Beraterin in einem IT-Projekt eingegangen. Im Zentrum stehen insbesondere folgende Fragen:

- Was sind die Ursachen für Konflikte in IT-Projekten?
- Wie kann Konflikten vorgebeugt werden?
- Was ist WirtschaftsMediation – wie funktioniert sie?
- Welches Vorgehen und welche Methoden haben sich in konfliktbehafteten Verhandlungssituationen bewährt?
- Wie kann Konflikten innerhalb des IT-Projektteams begegnet werden?
- Welche Rolle kann ein externer Berater oder eine externe Beraterin in einer Konfliktsituation spielen?

Das Tutorial bietet Platz für verschiedene Sichten auf das Thema und richtet sich daher bspw. an:

- Mitglieder und Führungskräfte von IT-Teams
- Mitglieder und Führungskräfte von Fachabteilungen, die in IT-Projekte involviert sind
- Menschen, die aus der externen Perspektive in IT-Projekten beratend tätig sind

Auf Wunsch der Teilnehmenden kann auf spezielle Fragestellungen aus dem angesprochenen Problemkreis eingegangen werden.

Referenten

Axel R. Raulinat ist Vorstandsmitglied der TENOS AG für Konfliktberatung, Konfliktlösung, Konfliktprävention und Partner einer Anwaltssozietät (Hamburg und München), die sich auf das Außergerichtliche Konfliktmanagement spezialisiert hat und ausschließlich auf diesem Gebiet tätig ist. Herr Raulinat ist WirtschaftsMediator und bildet im Rahmen der TENOS Akademie neben Richtern und Rechtsanwälten auch Führungskräfte der Wirtschaft zu WirtschaftsMediatoren aus. Er ist Lehrbeauftragter der Universitäten Hamburg und Bayreuth.

Hans Brandt-Pook ist Professor für Informatik/Wirtschaftsinformatik im Fachbereich Produktion und Wirtschaft an der Fachhochschule Lippe und Höxter. Seine Erfahrungen in der IT-Beratung führten ihn zum Thema Verhandlungs- und Konfliktmanagement. Als WirtschaftsMediator spezialisiert sich Herr Brandt-Pook u.a. auf Konflikte im Softwaremanagement.

Einführende Literatur

[Di05] Diez, H: Werkstattbuch Mediation. Centrale für Mediation, Köln, 2005.
[FUP04] Fisher, R. ; Uri, W; Patton, B.: Das Havard-Konzept. Campus, Frankfurt, 2004.
[Ha00] Haft, F.: Verhandlung und Mediation. C.H. Beck, München, 2000.

Sozialkompetenz in Software-Projekten

Uwe Vigenschow

Training und Coaching
oose. Innovative Informatik GmbH
Straßenbahnring 7
20251 Hamburg
uwe.vigenschow@oose.de

Abstract: Software-Projekte werden von Menschen mit Menschen für Menschen durchgeführt. Die Gestaltung der direkten und indirekten Zusammenarbeit aller Beteiligten ist damit einer der wesentlichen Erfolgsfaktoren. In diesem Tutorial werden Kommunikationsmodelle und -techniken vorgestellt und in Kleingruppen geübt. Diese Modelle und Techniken unterstützen Sie in der Kommunikation sowohl innerhalb des Entwicklungsteams als auch in der Zusammenarbeit mit anderen Abteilungen. Inhaltlich werden Kommunikations- und Frage-Grundtechniken behandelt, eine einfache und in der IT sehr praxistaugliche Typologie für eine empfängergerechte Kommunikation vorgestellt und Feedback als Lösung für zwischenmenschliche Probleme dargestellt.

1 Kommunikationsbasis: Über Wertschätzung Beziehungen aufbauen

Kommunikation wirkt schnell wie das Spiel „stille Post“ und kann zu zum Teil zu erheblichen Kommunikationsverlusten führen. Mit dem Mittel des *aktiven Zuhörens* kann diesen Verlusten entgegen gewirkt werden [Vi06c]. Kommunikation ist dabei stets wesentlich mehr als der reine inhaltliche Transfer objektiver Informationen. Die wesentlichen Faktoren erfolgreicher Kommunikation liegen in den Ebenen darunter. Das *Eisberg-Modell* liefert eine Basis, konstruktiv mit tieferen Ebenen umzugehen. Es gilt dabei, zum Gesprächspartner eine Beziehung aufzubauen [Zo06]. Die Basis einer solchen konstruktiven Beziehung ist die gegenseitige *Wertschätzung*. Gerade in schwierigen Situationen bildet sie das Gerüst, um sich auf die wichtigen Themen konzentrieren zu können und nicht von anderen Aspekten abgelenkt zu werden [Vi06a].

2 Kommunikation strukturieren

2.1 Projektumfeldanalyse und Projektpolitik

Wer sind unsere Ansprechpartner und wie gehen wir mit dem teils hoch politischem Projektumfeld um? Das Mittel der Projektumfeldanalyse und dabei im Besonderen die *Stakeholder-Elicitation* bzw. die *Stakeholder-Map* gibt uns ein Mittel, angemessen agieren zu können [Vi06c].

2.2 Fragetechniken

Aus Basis der natürlich-sprachlichen Analyse kann uns eine fünf-stufige Fragetechnik bei analytischen Aufgaben helfen. Sie kann sowohl im Gespräch wie auf beim Dokumenten-Review eingesetzt werden [Vi06b, Vi06c].

2.3 Typologie

Eine einfache Typologie kann uns helfen, empfänger-orientiert zu kommunizieren. Dies gilt beispielsweise in einer Diskussion bei der Behandlung von Einwänden wie auch bei der Erstellung von Präsentationen oder Dokumenten [Le06, Vi06c].

3 Feedback geben

Wie gehe ich mit schwierigen Kollegen um? Wie spreche ich Themen an, die mich an meinem Chef stören? Reicht ein jährliches Personalgespräch als Rückmeldung? Abschließend kommen wir zum schwierigsten Punkt: Wie können Sie angemessen, konstruktiv und wirkungsvoll Feedback geben? Der Feedback-Ablauf und seine Regeln werden vorgestellt und exemplarisch geübt. Die wichtigen Grundvoraussetzungen wie Wertschätzung oder aktives Zuhören führen uns wieder zurück zum Anfang und schließen den Themenkreis des Tutorials [Vi06c].

Autor

Uwe Vigenschow ist Trainer, Coach und Berater bei der oose. Innovative Informatik GmbH in Hamburg. Aus den praktischen Erfahrungen als Führungskraft, Projektleiter und Software-Entwickler wuchs sein Interesse an arbeitspsychologischen Themen. Er baut für oose. die neue Trainingssparte Softskills zu einem vollwertigen IT-Trainingskonzept auf.

Literaturverzeichnis

[Le06] Lenhard, A.; Vigenschow, U. : Erfolgsfaktor Mensch (4) – Mit schwierigen Menschen umgehen, Javamagazin 9-2006

[Vi06a] Vigenschow, U.; Zockoll, G.: Erfolgsfaktor Mensch (1) – Wieso Entwickler nicht reden und Fachbereiche nicht programmieren können, Javamagazin, 6-2006

[Vi06b] Vigenschow, U; Zockoll, G.: Erfolgsfaktor Mensch (3) – Wieso, weshalb, warum – wer nicht fragt bleibt dumm, Javamagazin, 8-2006

[Vi06c] Vigenschow, U., Schneider, B.: Softskills für Software-Entwickler I, dpunkt, 2006 (in Vorbereitung).

[Zo06] Zockoll, G.; Vigenschow, U.: Erfolgsfaktor Mensch (2) – Der Eisberg mit vier Ohren, Javamagazin, 7-2006

GI-Edition Lecture Notes in Informatics

P-1 Gregor Engels, Andreas Oberweis, Albert Zündorf (Hrsg.): Modellierung 2001.

P-2 Mikhail Godlevsky, Heinrich C. Mayr (Hrsg.): Information Systems Technology and its Applications, ISTA'2001.

P-3 Ana M. Moreno, Reind P. van de Riet (Hrsg.): Applications of Natural Language to Information Systems, NLDB'2001.

P-4 H. Wörn, J. Mühling, C. Vahl, H.-P. Meinzer (Hrsg.): Rechner- und sensorgestützte Chirurgie; Workshop des SFB 414.

P-5 Andy Schürr (Hg.): OMER - Object-Oriented Modeling of Embedded Real-Time Systems.

P-6 Hans-Jürgen Appelrath, Rolf Beyer, Uwe Marquardt, Heinrich C. Mayr, Claudia Steinberger (Hrsg.): Unternehmen Hochschule, UH'2001.

P-7 Andy Evans, Robert France, Ana Moreira, Bernhard Rumpe (Hrsg.): Practical UML-Based Rigorous Development Methods - Countering or Integrating the extremists, pUML'2001.

P-8 Reinhard Keil-Slawik, Johannes Magenheim (Hrsg.): Informatikunterricht und Medienbildung, INFOS'2001.

P-9 Jan von Knop, Wilhelm Haverkamp (Hrsg.): Innovative Anwendungen in Kommunikationsnetzen, 15. DFN Arbeitstagung.

P-10 Mirjam Minor, Steffen Staab (Hrsg.): 1st German Workshop on Experience Management: Sharing Experiences about the Sharing Experience.

P-11 Michael Weber, Frank Kargl (Hrsg.): Mobile Ad-Hoc Netzwerke, WMAN 2002.

P-12 Martin Glinz, Günther Müller-Luschnat (Hrsg.): Modellierung 2002.

P-13 Jan von Knop, Peter Schirmbacher and Viljan Mahnič (Hrsg.): The Changing Universities – The Role of Technology.

P-14 Robert Tolksdorf, Rainer Eckstein (Hrsg.): XML-Technologien für das Semantic Web – XSW 2002.

P-15 Hans-Bernd Bludau, Andreas Koop (Hrsg.): Mobile Computing in Medicine.

P-16 J. Felix Hampe, Gerhard Schwabe (Hrsg.): Mobile and Collaborative Business 2002.

P-17 Jan von Knop, Wilhelm Haverkamp (Hrsg.): Zukunft der Netze –Die Verletzbarkeit meistern, 16. DFN Arbeitstagung.

P-18 Elmar J. Sinz, Markus Plaha (Hrsg.): Modellierung betrieblicher Informationssysteme – MobIS 2002.

P-19 Sigrid Schubert, Bernd Reusch, Norbert Jesse (Hrsg.): Informatik bewegt – Informatik 2002 – 32. Jahrestagung der Gesellschaft für Informatik e.V. (GI) 30.Sept.-3.Okt. 2002 in Dortmund.

P-20 Sigrid Schubert, Bernd Reusch, Norbert Jesse (Hrsg.): Informatik bewegt – Informatik 2002 – 32. Jahrestagung der Gesellschaft für Informatik e.V. (GI) 30.Sept.-3.Okt. 2002 in Dortmund (Ergänzungsband).

P-21 Jörg Desel, Mathias Weske (Hrsg.): Promise 2002: Prozessorientierte Methoden und Werkzeuge für die Entwicklung von Informationssystemen.

P-22 Sigrid Schubert, Johannes Magenheim, Peter Hubwieser, Torsten Brinda (Hrsg.): Forschungsbeiträge zur "Didaktik der Informatik" – Theorie, Praxis, Evaluation.

P-23 Thorsten Spitta, Jens Borchers, Harry M. Sneed (Hrsg.): Software Management 2002 - Fortschritt durch Beständigkeit

P-24 Rainer Eckstein, Robert Tolksdorf (Hrsg.): XMIDX 2003 – XML-Technologien für Middleware – Middleware für XML-Anwendungen

P-25 Key Pousttchi, Klaus Turowski (Hrsg.): Mobile Commerce – Anwendungen und Perspektiven – 3. Workshop Mobile Commerce, Universität Augsburg, 04.02.2003

P-26 Gerhard Weikum, Harald Schöning, Erhard Rahm (Hrsg.): BTW 2003: Datenbanksysteme für Business, Technologie und Web

P-27 Michael Kroll, Hans-Gerd Lipinski, Kay Melzer (Hrsg.): Mobiles Computing in der Medizin

P-28 Ulrich Reimer, Andreas Abecker, Steffen Staab, Gerd Stumme (Hrsg.): WM 2003: Professionelles Wissensmanagement - Erfahrungen und Visionen

P-29 Antje Düsterhöft, Bernhard Thalheim (Eds.): NLDB'2003: Natural Language Processing and Information Systems

P-30 Mikhail Godlevsky, Stephen Liddle, Heinrich C. Mayr (Eds.): Information Systems Technology and its Applications

P-31 Arslan Brömme, Christoph Busch (Eds.): BIOSIG 2003: Biometric and Electronic Signatures

P-32 Peter Hubwieser (Hrsg.): Informatische Fachkonzepte im Unterricht – INFOS 2003

P-33 Andreas Geyer-Schulz, Alfred Taudes (Hrsg.): Informationswirtschaft: Ein Sektor mit Zukunft

P-34 Klaus Dittrich, Wolfgang König, Andreas Oberweis, Kai Rannenberg, Wolfgang Wahlster (Hrsg.): Informatik 2003 – Innovative Informatikanwendungen (Band 1)

P-35 Klaus Dittrich, Wolfgang König, Andreas Oberweis, Kai Rannenberg, Wolfgang Wahlster (Hrsg.): Informatik 2003 – Innovative Informatikanwendungen (Band 2)

P-36 Rüdiger Grimm, Hubert B. Keller, Kai Rannenberg (Hrsg.): Informatik 2003 – Mit Sicherheit Informatik

P-37 Arndt Bode, Jörg Desel, Sabine Rathmayer, Martin Wessner (Hrsg.): DeLFI 2003: e-Learning Fachtagung Informatik

P-38 E.J. Sinz, M. Plaha, P. Neckel (Hrsg.): Modellierung betrieblicher Informationssysteme – MobIS 2003

P-39 Jens Nedon, Sandra Frings, Oliver Göbel (Hrsg.): IT-Incident Management & IT-Forensics – IMF 2003

P-40 Michael Rebstock (Hrsg.): Modellierung betrieblicher Informationssysteme – MobIS 2004

P-41 Uwe Brinkschulte, Jürgen Becker, Dietmar Fey, Karl-Erwin Großpietsch, Christian Hochberger, Erik Maehle, Thomas Runkler (Edts.): ARCS 2004 – Organic and Pervasive Computing

P-42 Key Pousttchi, Klaus Turowski (Hrsg.): Mobile Economy – Transaktionen und Prozesse, Anwendungen und Dienste

P-43 Birgitta König-Ries, Michael Klein, Philipp Obreiter (Hrsg.): Persistance, Scalability, Transactions – Database Mechanisms for Mobile Applications

P-44 Jan von Knop, Wilhelm Haverkamp, Eike Jessen (Hrsg.): Security, E-Learning. E-Services

P-45 Bernhard Rumpe, Wofgang Hesse (Hrsg.): Modellierung 2004

P-46 Ulrich Flegel, Michael Meier (Hrsg.): Detection of Intrusions of Malware & Vulnerability Assessment

P-47 Alexander Prosser, Robert Krimmer (Hrsg.): Electronic Voting in Europe – Technology, Law, Politics and Society

P-48 Anatoly Doroshenko, Terry Halpin, Stephen W. Liddle, Heinrich C. Mayr (Hrsg.): Information Systems Technology and its Applications

P-49 G. Schiefer, P. Wagner, M. Morgenstern, U. Rickert (Hrsg.): Integration und Datensicherheit – Anforderungen, Konflikte und Perspektiven

P-50 Peter Dadam, Manfred Reichert (Hrsg.): INFORMATIK 2004 – Informatik verbindet (Band 1) Beiträge der 34. Jahrestagung der Gesellschaft für Informatik e.V. (GI), 20.-24. September 2004 in Ulm

P-51 Peter Dadam, Manfred Reichert (Hrsg.): INFORMATIK 2004 – Informatik verbindet (Band 2) Beiträge der 34. Jahrestagung der Gesellschaft für Informatik e.V. (GI), 20.-24. September 2004 in Ulm

P-52 Gregor Engels, Silke Seehusen (Hrsg.): DELFI 2004 – Tagungsband der 2. e-Learning Fachtagung Informatik

P-53 Robert Giegerich, Jens Stoye (Hrsg.): German Conference on Bioinformatics – GCB 2004

P-54 Jens Borchers, Ralf Kneuper (Hrsg.): Softwaremanagement 2004 – Outsourcing und Integration

P-55 Jan von Knop, Wilhelm Haverkamp, Eike Jessen (Hrsg.): E-Science und Grid Adhoc-Netze Medienintegration

P-56 Fernand Feltz, Andreas Oberweis, Benoit Otjacques (Hrsg.): EMISA 2004 - Informationssysteme im E-Business und E-Government

P-57 Klaus Turowski (Hrsg.): Architekturen, Komponenten, Anwendungen

P-58 Sami Beydeda, Volker Gruhn, Johannes Mayer, Ralf Reussner, Franz Schweiggert (Hrsg.): Testing of Component-Based Systems and Software Quality

P-59 J. Felix Hampe, Franz Lehner, Key Pousttchi, Kai Ranneberg, Klaus Turowski (Hrsg.): Mobile Business – Processes, Platforms, Payments

P-60 Steffen Friedrich (Hrsg.): Unterrichtskonzepte für inforrmatische Bildung

P-61 Paul Müller, Reinhard Gotzhein, Jens B. Schmitt (Hrsg.): Kommunikation in verteilten Systemen

P-62 Federrath, Hannes (Hrsg.): „Sicherheit 2005“ – Sicherheit – Schutz und Zuverlässigkeit

P-63 Roland Kaschek, Heinrich C. Mayr, Stephen Liddle (Hrsg.): Information Systems – Technology and ist Applications

P-64 Peter Liggesmeyer, Klaus Pohl, Michael Goedicke (Hrsg.): Software Engineering 2005

P-65 Gottfried Vossen, Frank Leymann, Peter Lockemann, Wolffried Stucky (Hrsg.): Datenbanksysteme in Business, Technologie und Web

P-66 Jörg M. Haake, Ulrike Lucke, Djamshid Tavangarian (Hrsg.): DeLFI 2005: 3. deutsche e-Learning Fachtagung Informatik

P-67 Armin B. Cremers, Rainer Manthey, Peter Martini, Volker Steinhage (Hrsg.): INFORMATIK 2005 – Informatik LIVE (Band 1)

P-68 Armin B. Cremers, Rainer Manthey, Peter Martini, Volker Steinhage (Hrsg.): INFORMATIK 2005 – Informatik LIVE (Band 2)

P-69 Robert Hirschfeld, Ryszard Kowalcyk, Andreas Polze, Matthias Weske (Hrsg.): NODe 2005, GSEM 2005

P-70 Klaus Turowski, Johannes-Maria Zaha (Hrsg.): Component-oriented Enterprise Application (COAE 2005)

P-71 Andrew Torda, Stefan Kurz, Matthias Rarey (Hrsg.): German Conference on Bioinformatics 2005

P-72 Klaus P. Jantke, Klaus-Peter Fähnrich, Wolfgang S. Wittig (Hrsg.): Marktplatz Internet: Von e-Learning bis e-Payment

P-73 Jan von Knop, Wilhelm Haverkamp, Eike Jessen (Hrsg.): “Heute schon das Morgen sehen“

P-74 Christopher Wolf, Stefan Lucks, Po-Wah Yau (Hrsg.): WEWoRC 2005 – Western European Workshop on Research in Cryptology

P-75 Jörg Desel, Ulrich Frank (Hrsg.): Enterprise Modelling and Information Systems Architecture

P-76 Thomas Kirste, Birgitta König-Riess, Key Pousttchi, Klaus Turowski (Hrsg.): Mobile Informationssysteme – Potentiale, Hindernisse, Einsatz

P-77 Jana Dittmann (Hrsg.): SICHERHEIT 2006

P-78 K.-O. Wenkel, P. Wagner, M. Morgenstern, K. Luzi, P. Eisermann (Hrsg.): Land- und Ernährungswirtschaft im Wandel

P-79 Bettina Biel, Matthias Book, Volker Gruhn (Hrsg.): Softwareengineering 2006

P-80 Mareike Schoop, Christian Huemer, Michael Rebstock, Martin Bichler (Hrsg.): Service-Oriented Electronic Commerce

P-81 Wolfgang Karl, Jürgen Becker, Karl-Erwin Großpietsch, Christian Hochberger, Erik Maehle (Hrsg.): ARCS´06

P-82 Heinrich C. Mayr, Ruth Breu (Hrsg.): Modellierung 2006

P-83 Daniel Huson, Oliver Kohlbacher, Andrei Lupas, Kay Nieselt and Andreas Zell (eds.): German Conference on Bioinformatics

P-84 Dimitris Karagiannis, Heinrich C. Mayr, (Hrsg.): Information Systems Technology and its Applications

P-85 Witold Abramowicz, Heinrich C. Mayr, (Hrsg.): Business Information Systems

P-86 Robert Krimmer (Ed.): Electronic Voting 2006

P-87 Max Mühlhäuser, Guido Rößling, Ralf Steinmetz (Hrsg.): DELFI 2006: 4. e-Learning Fachtagung Informatik

P-88 Robert Hirschfeld, Andreas Polze, Ryszard Kowalczyk (Hrsg.): NODe 2006, GSEM 2006

P-90 Joachim Schelp, Robert Winter, Ulrich Frank, Bodo Rieger, Klaus Turowski (Hrsg.): Integration, Informationslogistik und Architektur

P-91 Henrik Stormer, Andreas Meier, Michael Schumacher (Eds.): European Conference on eHealth 2006

P-93 Christian Hochberger, Rüdiger Liskowsky (Eds.): INFORMATIK 2006 – Informatik für Menschen, Band 1

P-94 Christian Hochberger, Rüdiger Liskowsky (Eds.): INFORMATIK 2006 – Informatik für Menschen, Band 2

P-95 Matthias Weske, Markus Nüttgens (Eds.): EMISA 2005: Methoden, Konzepte und Technologien für die Entwicklung von dienstbasierten Informationssystemen

P-96 Saartje Brockmans, Jürgen Jung, York Sure (Eds.): Meta-Modelling and Ontologies

P-97 Oliver Göbel, Dirk Schadt, Sandra Frings, Hardo Hase, Detlef Günther, Jens Nedon (Eds.): IT-Incident Mangament & IT-Forensics – IMF 2006

The titles can be purchased at:

Köllen Druck + Verlag GmbH
Ernst-Robert-Curtius-Str. 14
53117 Bonn
Fax: +49 (0)228/9898222
E-Mail: druckverlag@koellen.de