Dynaxity

Patrick Hamilton

Dynaxity

Management von Dynamik und Komplexität im Softwarebau

Mit 28 Abbildungen

Springer

Patrick Hamilton
Am Grundweg 22
64342 Seeheim-Jugenheim
patrick@drhamilton.de

Bibliografische Information der Deutschen Bibliothek
Die Deutsche Bibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über http://dnb.ddb.de abrufbar.

ISBN-10 3-540-31743-0 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-31743-2 Springer Berlin Heidelberg New York

Springer ist ein Unternehmen von Springer Science+Business Media
springer.de

Satz und Herstellung: LE-TEX, Jelonek, Schmidt & Vöckler GbR, Leipzig
Umschlaggestaltung: KünkelLopka Werbeagentur, Heidelberg
Gedruckt auf säurefreiem Papier 33/3100 YL – 5 4 3 2 1 0

Ich danke meinen Lufthansa-Kollegen, die mich in unseren gemeinsamen Projekten immer wieder dazu angeregt haben, Dinge neu zu hinterfragen und Zusammenhänge aus anderen Perspektiven zu beleuchten.

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Prolog

Wer sich nicht verändern will, wird auch verlieren,
was er bewahren möchte.
(Gustav Heinemann)

Folgt man aktuellen Statistiken über Softwarebau, so belegen diese, dass zwischen 60% und 80% aller Projekte im strengen Sinne gesehen nicht erfolgreich sind, d. h., dass die drei Kriterien in-time, in-quality und in-budget nicht planungsgemäß erfüllt wurden. Nicht berücksichtigt sind hierbei die unnötigen Folgekosten, die Firmen dadurch entstehen, dass sie durch Unachtsamkeit und Mängel entstandene Fehler während der Planungs- und Produktionsphasen nachbessern müssen bzw. durch konstruktive Gründe dauerhafte Mehraufwände bei der Pflege oder Weiterentwicklung ihrer Software haben.

1.1 Sichten müssen sich ändern

Sollten Sie sich die Frage stellen, ob, was und wie Dinge zum Besseren verändert werden können, so werden Sie in diesem Buch Antworten finden. Dabei wird nicht in erster Linie die Rede von Bits & Bytes, Objekten & Klassen oder irgendwelchen spezifischen Entwicklungsmethoden der Software sein. Hierzu gibt es vielfältige andere Fachliteratur. Zudem werden vermutlich Sie und Ihre Mitarbeiter ohnehin über viel Sachkenntnisse in diesem Bereich verfügen. Im Rahmen dieses Werkes möchte ich mich mit dem „nichtbinären“ Umfeld der Softwareentwicklung befassen. Der Tatsache, dass den größten Projektrisiken, nämlich Komplexität und Dynamik, nicht ausreichend oder rechtzeitig begegnet wird, soll besonderes Augenmerk gewidmet werden. Die Rede wird sein von all denjenigen Aspekten, die sich nicht mit Programmen und Prozessen abbilden lassen, sondern diesen zugrunde liegen müssen. Es geht um Menschen, die hierin aktiv handeln, persönliche Erfahrungen einbringen oder aufgrund persönlicher Unkenntnis Dinge verbergen oder verhindern. Dieses Buch versteht sich als ein hinterfragendes Werk und nicht als ein Text mit fertigen Rezepten. Es wird Ihnen viele Hintergrundinformationen anbieten und möchte Sie vor allem zum Nachdenken und Hinterfragen Ihrer aktuellen Vorgehensweisen einladen.

1.1.1 Lohnt sich dieses Buch für Sie?

Soweit zum Einstieg – jetzt sind Sie gefragt. Da ich Ihre Zeit nicht unnötig strapazieren möchte, schlage ich Ihnen drei Kategorien vor, die Ihnen dabei helfen sollen, für sich zu klären, ob dieses Buch es wert ist, dass Sie es zum jetzigen Zeitpunkt lesen:

- Wenn Sie mit Ihren Resultaten als IT-Professional bzw. denen Ihrer IT-Mannschaft völlig einverstanden sind und Sie auf durchweg positive Resultate blicken können, so sollten Sie dieses Buch beiseite legen und sich wichtigeren Themen zuwenden.
- Für den Fall, dass es bei Ihnen gewisse Anfragen oder Zweifel bzgl. der Effizienz Ihrer Softwareentwicklung geben sollte, so dürfte dieses Buch Sie an der einen oder anderen Stelle durchaus zum Nachdenken anregen.
- Stellen Sie sich den entsprechenden Fragestellungen, bevor Sie dieses Buch wieder weglegen!

Einen weiteren Punkt möchte ich Ihnen an dieser Stelle nicht vorenthalten. Dieses Buch hat die erklärte Absicht, nicht nur als eine Quelle für Sachinformationen zu fungieren. Vielmehr möchte ich Sie diskret dazu herausfordern einige Ihrer eigenen Handlungsparadigmen in diesem Kontext zu hinterfragen. Wenn Sie grundsätzlich bereit sind, selbst erste Schritte zu gehen, dann investieren Sie sinnvoll in diese Lektüre, andernfalls sind Sie gerade im Begriff, eine Fehlinvestition an Zeit und Geld zu leisten.

1.1.2 Nicht Bits und Bytes, sondern Menschen

Nach diesen Anmerkungen sind noch einige Punkte klarzustellen und deutlich herauszuheben. Sollten Sie an dieser Stelle ein Buch erwarten, welches Ihnen sagt, wie man z. B. UML[1], Java-Klassen oder Softwarearchitekturen erstellt, so wird Sie dieses Buch enttäuschen. – Natürlich wird in der einen oder anderen Form die Technik zu Wort kommen, erklärtes Ziel dieses Werkes ist es jedoch, die Menschen hinter diesen Technologien zu betrachten: die Entwickler, IT-Architekten, Fachspezialisten sowie die entsprechenden Manager. Der Schwerpunkt wird dabei auf dem Umgang mit der innewohnenden Komplexität bei der Erstellung von Software liegen und den Möglichkeiten, diese zu beherrschen. Es geht sowohl um Vorgehensweisen bei der Produktion wie auch deren Risiken, die durch diese Prozessbeteiligten hervorgerufen werden, und die Beleuchtung möglicher Fehlerquellen. Wir mögen inzwischen auf faszinierende Softwarewerkzeuge und

[1] UML – „Unified Modelling Language“ – ist eine derzeit häufig verwendete Softwareentwicklungs-Standardsprache zur Spezifizierung, Visualisierung und Konstruktion von Softwarekomponenten.

Entwicklungsmethoden blicken. Diese bleiben jedoch ineffizient, wenn die Menschen, die sie nutzen, zum Flaschenhals werden und nicht ausreichend in diese Vorgehensweisen integriert wurden. Da diese Techniken in enger Interaktion mit ihrem Umfeld, den eigenen Entwicklungsteams, dem Management und vor allem auch den Kunden, stehen, möchte ich mit Ihnen integrative Ansätze aufspüren und diskutieren.

Ja, Sie lesen richtig: Dieses Buch stellt sich immer wieder der Frage, ob nicht gerade Sie und ich – oder aber Ihre und meine IT-Kollegen – zum zentralen Faktor des Erfolgs oder Misserfolgs eines Softwareprojektes werden! Haben wir eine genügend breite Sichtweise auf die wirklichen Probleme, die sich geschickt hinter dem Tagesgeschäft verstecken, oder haben wir diese eigentlichen Verursacher noch nicht in ausreichendem Maße als die eigentlichen Feinde erkannt und ihnen den Krieg erklärt? Diese Probleme werden nur dann gelöst werden können, wenn Sie und ich fundierte, aber pragmatische Lösungen in die Wege leiten.

1.2 Die drei INs

Was erwartet Sie also in dieser Lektüre? Überall strebt man in der IT-Branche nach Software, die **IN**^{3} gerecht wird, nämlich „in-time“, „in-quality“ und „in-budget“. Die Software soll robust, am besten fehlerfrei und vor allem für die Nutzungsdauer leicht wartbar und erweiterbar sein. Aufgrund des hohen Kostendrucks soll die zu bauende Software dazu beitragen, die Unternehmensgewinne stabil zu halten. Viele andere Kostenfaktoren werden absehbar steigen, hier wurde jedoch mittels Unterstützung durch ein Softwareprodukt eine Nische erspäht, von der man sich bei sinkenden Stückpreisen immer noch steigende oder wenigstens gleichbleibende Erlöse erhofft. Ein solches Unterfangen ist grundsätzlich richtig, es verlangt jedoch einen hohen Reifegrad der jeweiligen Entwicklungsabteilungen. Diese müssen geprägt sein durch ganzheitliches, vorausschauendes und unternehmerisches Denken. Ist Ihre Organisation bereits so mündig?

Der erfolgreiche Bau solcher Software ist nicht nur geprägt von den technischen Randbedingungen und dem Einsatz geeigneter Technologien, sondern die Resultate werden stark durch die darin mitwirkenden Menschen beeinflusst. So betrachtet gehen diese für die Zeit der Softwareentwicklung eine Art Symbiose mit der Technik ein, um auf diese Art und Weise ihr „Baby“ hervorzubringen.

Auf diesen Seiten geht es somit um diejenigen Randbedingungen und Einflussfaktoren, die **IN**^{3} stören. In diesem Werk wird die Rede davon sein, welches die anfänglichen Einflussfaktoren sind, die Ihnen die Chance

geben, auch noch gegen Ende eines Software-Lebenszyklus Kosten und Qualität zusammenzuhalten, bzw. für welche frühen Fehler Sie den Rest des Software-Lebenszyklus büßen müssen. Hierbei möchte ich Sie mit wirkungsvollen Ansätzen bekannt machen, die man im wahrsten Sinne als **ganzheitlich Psycho-Logische** Aspekte beim Bau von Software bezeichnen könnte.

Jeder dieser drei Teilbegriffe steht hierbei für eine entscheidende Grundaussage:

Unter **Ganzheitlichkeit** wird es in diesem Kontext um das Gesamtsystem Softwareentwicklung gehen. Die Rede wird von denjenigen Aspekten sein, die häufig unterschlagen werden, weil sie nicht unbedingt in ein ingenieurmäßiges Weltbild passen. Daneben wird immer wieder die Frage beleuchtet werden, was Software komplex macht und wie man dieses entschärfen kann.

Der zweite Begriff, nämlich **„Psycho“**, steht für die Skills, Teamstrukturen und Eigenarten der Entwicklungsmannschaft.

Als Letztes bezieht sich die **„Logik“** auf all das, was wir ohnehin schon in der einen oder anderen Weise innerhalb der IT betreiben: auf Prozesse, Verfahren, Methoden und ingenieurmäßiges Vorgehen.

Da der Ganzheitlichkeit in diesem Buch große Bedeutung zugeordnet wird, mag es Themen geben, die sie ansprechen, andere wiederum könnten Ihre eigenen Handlungsparadigmen betreffen und zuletzt wird es solche Ansätze geben, die Sie völlig anders sehen bzw. ablehnen. Jede dieser Positionen ist legitim! Im Fall der letzten Sichtweise hilft mir persönlich eine Einstellung des protestantischen Reformators Martin Luther weiter: Bei Aussagen, die er nicht verstand oder akzeptierte, so berichtet man über ihn, zog er innerlich seinen Hut, grüßte freundlich, ließ das Thema stehen und zog weiter. Diesen Ansatz halte ich persönlich für differenzierter und adäquater als solche Vorgehensweisen, bei denen man wegen divergenter Sichtweisen zu bestimmten Teilthemen den gesamten Ansatz verwirft.

1.3 Bilanz ist gefragt

Sollte ich Sie jetzt neugierig gemacht haben, so möchte ich Sie zu einer Bestandsaufnahme in dem für Sie zutreffenden Kontext einladen. Im klassisch medizinisch-therapeutischen Sinne möchte ich hierbei im Wesentlichen folgende Schwerpunkte setzen:

Anamnese. Zunächst heißt es Inventur machen! Wie sieht es tatsächlich in Ihrer Organisation aus? Kennen Sie die kleinen und großen „Sünden", die sich regelmäßig wiederholen? Das Augenmerk wird hier auf dem gesamten Lebenszyklus einer Software liegen. Ebenso geht es darum, einen Blick auf die Prozessbeteiligten zu werfen und die guten wie schlechten Einflüsse zu ergründen, welche diese auf Ihre Produkte haben.

Analyse. Zunächst wird es darum gehen, die eigentlichen „Feinde" im Softwarebau zu ermitteln und herauszufinden, durch welche Mechanismen diese wirksam werden. Hieran anschließend werde ich mit Ihnen verschiedene „Best Practices", also bewährte Verfahren – auch aus anderen Industriezweigen –, betrachten und deren Verwertbarkeit für effizienten Softwarebau beleuchten. Es wird die Rede von Prozessmodellen sowie von gut untersuchten Be- und Entschleunigungsfaktoren aus dem Praxisalltag sein.

Therapie. Mit diesem so ermittelten gemeinsamen Begriffshintergrund ist der letzte Teil dieses Werkes Praxisaspekten gewidmet. Anhand von Schwerpunktthemen aus dem Bereich Projekt- und Prozessmanagement werde ich solche Aspekte zusammenfügen, die m. E. zusammengehören. Es werden geeignete Methoden und „Werkzeuge" vorgestellt und es wird darüber gesprochen werden, wie sich die beteiligten Menschen so integrieren lassen, dass hierdurch die Qualität eines Softwareproduktes und die Effizienz der Umsetzung deutlich gesteigert werden können. Zum Abschluss werden diese Elemente, Puzzleteilen ähnlich, zu einem praxistauglichen Gesamtmodell verbunden.

Die hier zusammengestellten Vorgehensweisen verstehen sich als holistischer, also ganzheitlicher Ansatz, wobei diese Ganzheitlichkeit sowohl ingenieurmäßige wie auch personenorientierte Aspekte umfasst. Was dies bedeutet, lässt sich mittels eines kleinen Gedankenexperiments anhand von Abb. 1.1 beantworten: In dieser Abbildung sehen Sie unterschiedlich weite Röhren, die in verschiedener Weise kombiniert und übereinander gestapelt wurden. Nach Abschluss des Aufbaus wird nun z. B. Sand durch die jeweiligen Konstrukte hindurchgeleitet. (Ich unterstelle dabei, dass kein Sand zwischen den Rohrversatzstücken verloren geht!) Um eine solche „Sandmaschine" zu bauen, müssen Sie Geld ausgeben: Dünne Rohre kosten wenig, mittlere Rohre mehr, dicke Rohre viel Geld!

Die Gretchenfrage lautet nun: Mit welchem System machen Sie die größten Gewinne? – Nun, simpel wie das Beispiel aufgebaut ist, ergibt sich nachfolgende Lösung (vgl. Abb. 1.2, Experiment, Auswertung & Regeln). Sie lautet: Fall (C) hat das Rennen klar gewonnen, obwohl es NICHT die billigste Variante in der Erstellung war. Punktuelle Verbesserungen von abhängigen Prozessen mögen werbewirksam sein, sie verringern jedoch nicht per se die Stückkosten. – Um möglichst rasch eine

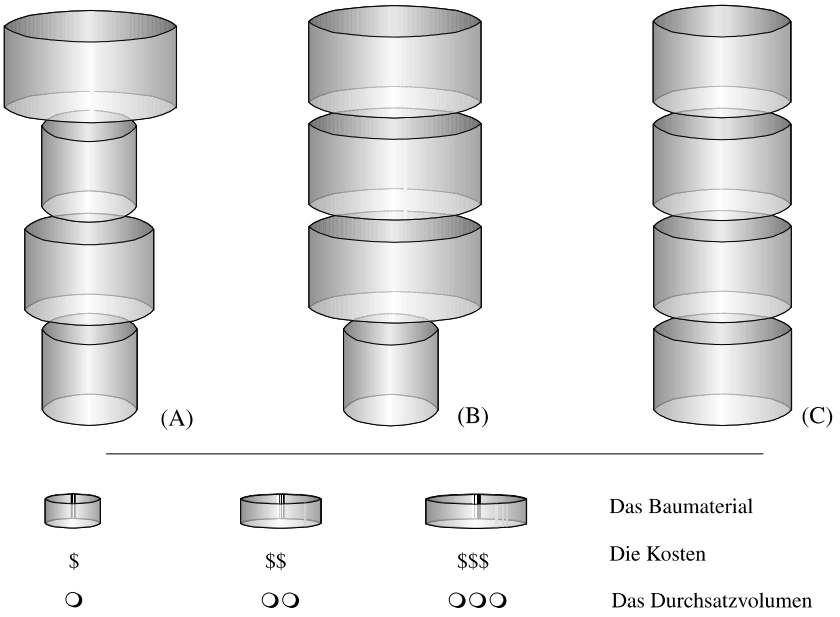

Abb. 1.1. Ein Gedankenexperiment

messbare Steigerung des Durchsatzes bei möglichst geringen Gesamtkosten zu erreichen, müssen alle betroffenen Segmente zeitgleich, dafür aber in kleinen Schritten auf ein gemeinsames Maß gebracht werden.

Um auf das Kernthema dieses Buches zu kommen, werde ich nun dieses kleine Röhrenexperiment auf IT-Entwicklungsabteilungen, wie sie vielfach zu finden sind, übertragen (vgl. Abb. 1.2, Übertragung): Die IT-Mitarbeiter kennen die neuesten Softwaretechnologien und brennen meistens darauf, diese einsetzen zu können. Unternehmen investieren stetig mehr in Testen und Qualität, um so zu qualitativ besseren Produkten zu gelangen. Unternehmensführungen investieren z. T. erhebliche Summen in Softwarewerkzeuge und bilden die Mitarbeiter hierin auch entsprechend aus, um hierdurch die Effizienz im Softwarebau zu steigern. Die Menschen mit ihren Grenzen und den hierdurch verursachten Störungen, die massiv auf die Produktion einwirken können, bleiben vielfach unberücksichtigt. Ihnen wird häufig unterstellt, dass sie als Professionals hierfür bereits alle notwendigen Skills besäßen. Vergessen werden dabei jedoch zwei Aspekte:

- Der erste Aspekt besteht darin, dass die allerwenigsten IT-Werker diese Faktoren in ihren Ausbildungen vermittelt bekommen haben.
- Zum Zweiten haben sehr viele IT-Entwicklungsmannschaften ein eher jugendliches Durchschnittsalter, weshalb sie neueste Technologien gut kennen und auch sehr schnell in deren Ausimplementierung sind, jedoch

fehlt es oftmals an der notwendigen Erfahrung in den anderen Bereichen. Hierbei ist nicht die Erfahrung zu kodieren gemeint: Dafür sind viele Softwaremitarbeiter schon seit ihrer Jugend mit diesem Medium vertraut. Die Erfahrung, von der ich hier spreche, hat mit ganzheitlichen Lösungen zu tun, denen erschwerte Randbedingungen in einer komplexen Gesamtsituation beigegeben sind.

Das vorliegende Buch kann und will die aufgeworfenen Einzelthemen nicht in aller Tiefe ergründen. Es schließt sich nicht der Fastfood-Mentalität an, die oftmals einem „Kochrezept“ gleich, lautet: „How to get all your complexity related IT problems solved within 10 minutes.“ Untersuchungen bei Unternehmen, die solche oder ähnliche Veränderungsprozesse erfolgreich hinter sich gebracht haben und nun die Früchte ihrer Anstrengungen einsammeln, belegen, dass mit 3–5 Jahren harter Arbeit zu rechnen ist. Primäres Ziel dieses Buches ist es daher, Ihnen zu vermitteln, wie weit hier die wechselseitigen Abhängigkeiten reichen. Ich möchte Sie für diese Thematik sensibilisieren, um Ihnen danach

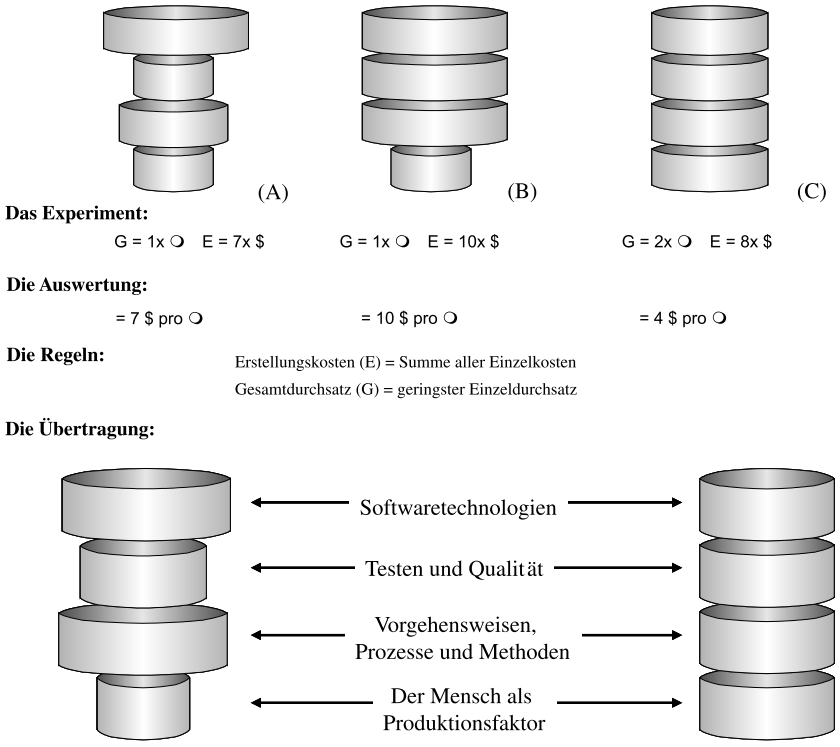

Abb. 1.2. Auflösung des Gedankenexperiments und Übertragung auf die Softwareproduktion

schrittweise in ausgewählten Bereichen Verbesserungen vorzuschlagen. – Wie schon am Eingangsbeispiel ersichtlich, wird dies mit zusätzlichen Aufwänden verbunden sein, die sich nicht nur im kaufmännischen Sinne definieren lassen, sondern darin bestehen, dass persönliche Investitionen im Sinne von Engagement und Durchsetzungswille eingebracht werden müssen. Nur dann, wenn Sie bereit sind, großzügig alte Paradigmen und Klischeevorstellungen hinter sich zu lassen und neue Wege zu beschreiten, wird eine Umstellung gelingen.

Genug der Vorrede. Ich lade Sie nun in die Wirklichkeit der Softwareproduktion ein, wie sie immer noch in vielen IT-Häusern anzutreffen ist.

1.4 Tote Pferde reiten

Bei den Dakota-Indianern in Amerika gab es früher eine Stammesweisheit, die besagte: „Wenn Du feststellst, dass Du auf einem toten Pferd reitest, so besteht deine beste Strategie darin abzusteigen." Gerade in der IT-Branche gehen wir leider recht häufig anders mit dieser Thematik um, weil Projektdruck und Anforderungen an Management und Mitarbeiter ein Überleben auf Tagesbasis notwendig werden lassen. Daher betreiben wir oftmals mit Akribie Symptombehandlung, vergessen dabei jedoch sehr häufig, die eigentlichen Probleme wahrzunehmen und zu lösen. Nicht selten geschieht es dann, dass die Weisheit der Dakotas in eine oder mehrere eigene Weisheiten umgemünzt wird:

- Wir besorgen uns eine stärkere Peitsche, um das tote Pferd anzutreiben. (Etwa indem Service-Level-Verträge mit Zulieferern abgeschlossen und diese vertraglich zur Besserung der Gesamtsituation gezwungen werden.)
- Wir wechseln den Reiter, um mit frischem Elan voranzukommen. (Erfahrene Mitarbeiter werden durch weniger erfahrene Kräfte ausgetauscht. Die neuen Kräfte lassen sich – so hofft man wohl – besser durch das Linienmanagement steuern und leisten weniger Widerspruch?!)
- Wir sagen: „So haben wir das Pferd immer geritten" und bleiben bei Altbewährtem. (Man verschließt sich gegen echte Innovation. Es wird damit argumentiert, dass man ja nur „ein wenig" neue Software haben wolle und keine Weltrevolution.)
- Wir gründen einen Arbeitskreis, um das Pferd gründlich zu analysieren und all seine Leistungen der Vergangenheit akribisch zu dokumentieren. (Leitungskreise werden eingesetzt, die wiederum Mitarbeiterstäbe einsetzen, die wiederum akribisch jedes einzelne alte Bauteil bibliographisch erfassen.)

- Wir besuchen andere Orte, um zu sehen, wie man dort tote Pferde reitet, um zu neuen Erkenntnissen zu gelangen. (Man reist zu Schwester-, Nachbar- oder Referenzfirmen, um zufrieden festzustellen, dass man dort genauso arbeitet wie man selbst. Vergessen wurde bei der Auswahl der Firmen leider nur, dass alle besuchten Firmen aus dem Verzeichnis „Club der Halter toter Pferde“ stammen.)
- Wir erhöhen die Qualitätsstandards für den Beritt toter Pferde und führen hierfür sehr strenge Audits ein. (Qualität kommt ins Spiel und bewertet die Güte ineffizienter Vorgehensweisen – was in letzter Konsequenz oftmals nichts anderes heißt: Beim Audit ist herausgekommen, dass die (ineffizienten) Prozesse hervorragend gelebt werden. – Die Frage nach der Werthaltigkeit der Prozesse wird häufig nicht gestellt.)
- Wir bilden eine Taskforce, um das tote Pferd wiederzubeleben. (Es wird viel Kraft und Kapital mobilisiert, um halbtote Software zu revitalisieren, ihr etwas Kosmetik zu verpassen oder dem Kunden gegenüber ein neues Produktimage aufzubauen. – Technisch gesehen bleibt die Software jedoch unverändert. Die Tour zum Kosmetiker bringt vielfach nur teilweise erfüllte Versprechungen für die Kunden und garantierte Zusatzaufwände für das eigene Haus mit sich!)
- Wir schieben eine Trainingseinheit ein, um effizienter tote Pferde zu reiten. (Mitarbeiter werden in Prozessen oder auch Softwareproduktionen geschult – dumm dabei ist nur, dass das Ganze vielfach in den Köpfen der Mitarbeitet endet. Bis ans Licht der Wirklichkeit dringt es nicht vor, weil z. B. die hierfür notwendige Infrastruktur noch fehlt – und somit alles Theorie und unnützer Aufwand bleibt.)
- Wir stellen Vergleiche zwischen unterschiedlich toten Pferden an, um die Wettbewerbschancen zu erhöhen. (Es werden Gremien gebildet, die sich Firmen mit ähnlicher Arbeitsweise anschauen, um herauszufinden, mit Hilfe welchen Mysteriums bestimmte marginale Verbesserungen erreicht werden konnten.)

Langer Rede kurzer Sinn: Es werden viele Nebenaspekte betrachtet und diskutiert, die oftmals jedoch eher den Gesetzen blinden Aktionismus folgen, statt Probleme von den Wurzeln her grundsätzlich zu lösen. In diesem Sinne möchte ich Sie an dieser Stelle zu einer kurzen Selbstreflektion einladen. Nachfolgend habe ich Ihnen noch einige Aspekte zu toten Pferden offen gelassen. Die Frage an Sie lautet nun: Wie sehen Ihre Pferdekadaver aus?

- Wir ändern die Kriterien, die besagen, ob ein Pferd tot ist.
- Wir kaufen Leute von außerhalb ein, um das tote Pferd zu reiten.
- Wir schirren mehrere tote Pferde zusammen an, damit diese schneller werden.

- Wir machen zusätzliche Mittel locker, um die Leistung des toten Pferdes zu erhöhen.
- Wir geben Studien in Auftrag, um zu sehen, ob es billigere Berater für tote Pferde gibt.
- Wir kaufen etwas hinzu, das tote Pferde schneller laufen lässt.
- Wir erklären, dass das Pferd „besser, schneller und billiger“ tot ist.
- Wir bilden einen Qualitätszirkel, um eine Verwendung für tote Pferde zu finden.
- Wir überarbeiten die Leistungsbedingungen für Pferde.
- Wir richten eine unabhängige Kostenstelle für tote Pferde ein.

2 Basteln Sie oder produzieren Sie schon?

Wenn Sie zentrale Probleme Ihres Marktes dauerhaft sichtbar besser lösen als andere und Ihre Ziele konsequent verfolgen, dann können Sie Ihren Erfolg nicht verhindern.
(Rolf-Bodo Brombacher, Ludwig-Erhard-Preisträger 2000)

Folgt man Auswertungen z. B. der Gartner-Group über die Realisierung begonnener Softwareprojekte, so belegen deren Zahlen, dass weit über 60% all dieser Projekte als nicht erfolgreich zu betrachten sind. Erfolglos insofern, als sie entweder abgebrochen wurden, nicht die geforderte Qualität bzw. den Leistungsumfang erbrachten oder deutlich teurer ausfielen, als veranschlagt worden war. Anders herum gesprochen: Statistisch betrachtet verbrennen Sie mehr als die Hälfte Ihres Geldes, das Sie in die IT stecken! – Nun, das ist sehr plakativ und bedarf der Präzisierung. In diesem Abschnitt werde ich mit Ihnen einen Streifzug durch die Wirklichkeit der Softwareherstellung unternehmen und eine Bestandsaufnahme wagen.

2.1 Biographie eines Softwarelebens

Lösungen auf der Basis von Vermeidung sind die Probleme von morgen.
(Heinrich Fallner)

Wie entsteht Software und wann vergeht sie wieder? – Welche Schritte finden bei Ihnen statt und welche werden unterschlagen? Welche frühen „Sünden“ wurden begangen und wo bringen gerade diese Entscheidungen Sie später in Schwierigkeiten? Um aus alten Fahrwassern aussteigen zu können, ist es notwendig die eigene Situation selbstkritisch zu betrachten. Niemand zwingt Sie dazu, Ihre persönlichen Erkenntnisse später öffentlich zu präsentieren!

Ich möchte Sie an dieser Stelle einladen, mich auf einem Streifzug durch den gesamten Lebenszyklus einer Software – wenn Sie so wollen vom ersten bis zum letzten Atemzug – zu begleiten. Auf diesem „Lebensweg“ Ihrer Software werde ich an markanten Stellen immer wieder Stopps einlegen, um auf gewisse Dinge etwas genauer zu schauen, gegebenenfalls auch bereits

die Konsequenzen getroffener Entscheidungen zu betrachten. Dabei ist es klar, dass jede Software ihre Spezifika hat und somit nicht alles auf Sie exakt so zutreffen wird. Die wiedergegebenen Stationen sind aus meinen Beobachtungen heraus sehr regelmäßig anzutreffen und stets mit vergleichbaren Begleiterscheinungen gepaart. Wichtig ist mir in diesem Zusammenhang weder die Reihenfolge noch die Vollständigkeit der einzelnen Stationen dieses exemplarischen Lebenszyklus. Worauf es mir besonders ankommen wird, sind die vielen kleinen Fehler, die vom ersten Tag an begangen werden. Im Sinne des Fehlerfortpflanzungsgesetzes sind sie es, die sich konstant, aber zunächst unsichtbar kummulieren, bis sie in Summe zur Bedrohung für das Projekt werden, jedoch nicht mehr bereinigt werden können.

2.1.1 Erste Ideen – vom Fischen in trüben Gewässern

Die Zukunft gehört denen, welche die Möglichkeiten erkennen, bevor sie offensichtlich werden. (Oscar Wilde)

Lassen Sie mich ganz vorne anfangen und der eigentlichen „Zeugung" eines neuen Softwareproduktes beiwohnen. Es ist der Zeitpunkt, an dem das erste Mal öffentlich darüber gesprochen wurde, bestimmte Abläufe oder Tätigkeiten mit einem Computer erledigen zu lassen bzw. ein vorhandenes Produkt durch einen viel genialeren Nachfolger zu ersetzen. Am Anfang steht dabei das Bedürfnis, bestimmte Probleme kostengünstig, flexibel und zukunftssicher per EDV zu lösen. Schnell fügen sich zu diesem ersten Gedanken weitere Ideen hinzu, was man mit dieser geplanten Software alles erreichen möchte und was alles zu integrieren ist. Je nach fachlichem Wissensstand der geistigen Väter entsteht so sehr schnell ein gedankliches Märchenschloss auf der Basis des eigenen Wissens, dessen Grenzen nur durch die Phantasie der Beteiligten festgelegt sind.

Bleiben wir noch einen Augenblick bei den Fremdkunden. Hier stellt sich die Situation oft wie folgt dar: Sie haben als IT-Anbieter mittels intensiver Akquise, mit Messeständen, Besuchen von Consultants, Hochglanzdemos oder Prospekten Ihre Zielkunden und somit potentiellen Auftraggeber beackert, wohlwissend, dass die eigene Verkaufsprämie einzufahren ist. Letztendlich gelang es Ihnen, den Interessenten davon zu überzeugen, dass dessen geäußerte Vorstellungen durch Ihre Organisation realisierbar sind! Jetzt ist er reif! Sie haben den Zugang zum Kunden hergestellt und vermitteln auf unterschiedlichste Weisen, dass gerade Ihre Firma in eben diesem Marktsegment wirklich (fast) alle seine Wünsche realisieren kann. Die angedachte Software wird extrem viel leisten können, auf jeden Fall deutlich mehr als mögliche Wettbewerbsprodukte. Die Firma hinter der Software ist hochflexibel und kann selbstverständlich, falls notwendig, die

Besserstellungsmerkmale des Wettbewerbs kurzfristig ebenfalls realisieren. Liefertreue, Qualität und Support sind dabei kein Thema, immerhin präsentiert man sich dem Kunden als Profi. – Kurz: „Wir bauen für Sie die ideale Software in einer idealen Welt, bei der alle Bereiche perfekt zu einem Optimum zusammengefügt sein werden."

Sollte der Kunde näheres Interesse zeigen, so wird ihm mit Hilfe von Geschäftsessen in gediegenem Ambiente oder Betriebsbesichtigungen mit Blick auf die vielen Monitore und fleißigen Mitarbeiter klar vermittelt: „Dies hier ist eine Softwarefabrik! Mit Routine und Perfektion werden wir Deine Probleme schnell und kostenkünstig lösen." Das Ganze – natürlich hypermodern – hat ein wenig den Charakter und auch den Charme eines mittelalterlichen Wochenmarktes. Auf der einen Seite sitzen die Marktfrauen, die marktschreierisch ihre Waren feilbieten. Auf der anderen Seite gehen die Hausfrauen, Diener etc. an den Marktständen auf und ab und suchen nach der für sie besten Ware. Dabei ist die Auswahl ziemlich groß und es sellt sich die Frage, warum die Äpfel gerade hier und nicht am qualitativ gleichwertigen Nachbarstand gekauft werden.

Was aber ist die beste Ware? Was sind die Entscheidungsfaktoren dafür? Und was gibt am Ende tatsächlich den Ausschlag für einen speziellen Lieferanten. Sobald für den Kunden nachvollziehbar ist, dass diese Ware alle seine Wünsche erfüllt, wird er sich andere Kriterien suchen, um zu einer Lieferantenentscheidung zu gelangen. Diese Kriterien sind keineswegs immer nur der Preis, der angebotene Leistungsumfang oder die Qualität. Gerade bei Neukunden ist es zu einem erheblichen Anteil das Gefühl, hier gut aufgehoben zu sein, professionell bedient zu werden und vertrauenswürdigen Verkaufsberatern gegenüberzusitzen.

2.1.2 Angebissen! – die Geburtsstunde einer neuen Software

Die meisten meiner Ideen gehörten ursprünglich anderen Leuten, die sich nicht die Mühe gemacht haben, sie weiterzuentwickeln.
(Thomas A. Edison)

Kommt es dann wirklich zur heißen Verkaufsphase, so spielen die Randthemen auch weiterhin keine geringe Rolle. Der umworbene Kunde möchte sich weiterhin gut bei Ihnen aufgehoben fühlen, Sie haben vieles getan, um ihm im Vorfeld genau dieses Gefühl zu vermitteln, und haben ihn regelrecht verwöhnt. Durch wiederholte Präsentation einer sehr positiven Wirklichkeit, durch Kontakte mit kompetenten und netten Menschen und durch verschiedene Formen von verkaufsfördernden Maßnahmen haben Sie eine Bindung zwischen der eigenen Firma und dem potentiellen Kunden aufgebaut.

Sie haben soeben Standards gesetzt, an denen Sie dauerhaft gemessen werden und auf die man in schwierigeren Zeiten im Zweifelsfalle missmutig verweisen wird. In diesem Zusammenhang haben Sie, falls Sie leichtfertig waren, dem möglichen Kunden bereits wichtige Versprechungen gegeben. Immer wieder ist dies die Phase, in der man dem Kunden zusagt, nahezu alles für ihn realisieren zu können, dabei nur unwesentlich mehr Aufwände für ihn zu erzeugen und die Ergebnisse in Rekordzeit zu erreichen. Mag sein, dass Sie an dieser Stelle einwerfen: Der Markt ist hart umkämpft, auch der Wettbewerb erzählt manches, was er hinterher so nicht einhalten kann – durch die kommenden Vertrags- und Anforderungsphasen werden diese „Großzügigkeiten" ohnehin wieder nivelliert werden. – Dies ist sicherlich korrekt – worum es mir an dieser Stelle geht, ist die geweckte Erwartungshaltung für Dinge, von denen man als Lieferant weiß, dass sie sich nicht darstellen lassen. Es sind oftmals die ersten Schritte in eine unnötige Kompliziertheit zukünftiger Softwareprodukte, weil beim Kunden zu bestimmten Aspekten konkrete Erwartungshaltungen geweckt wurden, die nur unter großem Zusatzaufwand zu realisieren sind. Für alle unbemerkt beginnt der Kunde bereits an dieser Stelle, sich ein „Rabattmarkenheftchen" anzulegen, in dem jedes Mal ein „Rabattmärkchen"[2] verklebt wird, wenn Absprachen später nicht eingehalten wurden oder vereinbarte Ziele nicht in gewünschter Form zustande kamen. Bis dieses mentale Heftchen voll ist, kann eine ganze Weile dauern, sollten Dinge aber später nicht wunschgemäß verlaufen, so fallen dem Kunden auf einmal all die frühen Sünden wieder ein, und zwar meistens dann, wenn Sie es am wenigsten gebrauchen können!

2.1.3 Zeit für Verträge

Nun ist es die Zeit der Verträge und Vereinbarungen und die ungeschönte Wirklichkeit hält Einzug. Es beginnt das Feilschen zwischen Anbieter und Kunden, in dem sich entscheidet, welches Leistungspaket zu welchem Preis an den Kunden geht. Gerade hier finden immer wieder Fehler im großen Stil statt. Die Rede ist hierbei nicht von den Verträgen an sich, sondern den Aspekten, die mit der späteren Software zu tun haben werden! Jede Software – insbesondere die speziell für Kunden entwickelte Individualsoftware – hat in gewisser Weise den Charakter einer Traum-Maschine. Dabei sieht das Ganze aus Sicht des Kunden etwa so aus: Er gibt eine Applikation in Auftrag, gibt vor, was er meint zu brauchen, und bekommt

[2] Das Phänomen des „Rabattmarkenhefts" stammt aus dem Bereich des Konfliktmanagements. Kommt es in Situationen dazu, dass „das Fass überläuft", wie der Volksmund sagt, so entspricht all das, was das Fass angefüllt hat, den beschriebenen „Rabattmärkchen".

(hoffentlich!) Entsprechendes geliefert. Indem er diese neue Anwendung in Betrieb nimmt, fällt ihm all das ein, was man hätte noch integrieren, effizienter gestalten oder besser umsetzen können. Je nach Charakter, Zeitdruck und Liquidität werden daraufhin weitere Zusatzkomponenten beauftragt, eine Dauerbaustelle beim Hersteller eingerichtet, das Ganze als kundenspezifische Änderungen weiter betrieben oder aus Bordmitteln nach dauerhaften Provisorien gesucht.

Wenn das so absehbar ist, wie der nächste Sonnenuntergang, so frage ich Sie, was können Sie als Hersteller in dieser Phase unternehmen, um rasch und für Sie kostengünstig für die zu erwartenden Zusatzwünsche gerüstet zu sein? Möglich, dass Sie mir jetzt antworten würden:

- Wir werden selbstverständlich nachweisliche Fehlprogrammierungen beseitigen.
- Auch sind wir grundsätzlich bereit, das Produkt weiterzuentwickeln.
- Gerne nehmen wir auch weitere Kundenwünsche auf und schaffen hierfür individuelle Lösungen.

Das klingt gut, aber hierauf wollte ich nicht hinaus! – Worauf es mir an dieser Stelle ankommt, ist der „wir können alles für Dich machen, denn Du, Kunde, sollst ja König sein“ – Aspekt. Verkaufstechnisch mag so eine Aussage legitim sein. Aus Sicht der Softwareproduktion hingegen sind solche Sätze katastrophal, versprechen Sie doch ein magisches Stück Software, beim dem alle zukünftigen Änderungswünsche dem Kunden bereits im Vorfeld von den Augen abgelesen wurden und hierfür eine technische Plattform bereitgestellt sein wird. Sicherlich kennen Sie solche Software, bei der man schnell dem Kunden noch ein kleines Zugeständnis gemacht hat im Sinne von: „Ja, das bauen wir Ihnen selbstverständlich zusätzlich ein – das ist für uns kein großer Aufwand!“. – Die Wirklichkeit überholt Sie in sehr vielen Fällen bereits nach wenigen Monaten, wenn dieses kleine „Zugeständnis“ zur Geldverbrennungsanlage wird und man den erhofften Gewinn in eben diesem Zusatz-Softwaremodell in deutlichem Maß schmälern muss!

2.1.4 Erste Konzepte

Dem Kunden Zugeständnisse zu machen ist selbstverständlich, allerdings werden diese zum Fallstrick, wenn sie, soweit es um technische Zugeständnisse geht, ohne Abstimmung mit den IT-Planern gemacht werden. Wenn Sie in der Phase der Traumhäuser Ihres Kunden nicht bereits interne strategische Überlegungen angestellt haben, ob, wie und wo technische Ergänzungen aus Ihrer Sicht gewollt sind, so grenzt dies an Fatalismus. Die Chance, dass Ihre Entwicklungsabteilung aus Unkenntnis dieser Fakten später