

Fach-Beitrag zum Thema

Software-Projektmanagement oder Wieso scheitern Software-Projekte?

Immer noch sind erschreckende Statistiken im Bereich Software-Projektentwicklung ein Faktum. Die Standish Group spricht gar davon, dass 75% aller Softwareprojekte scheitern bzw. nicht termin- oder kostengerecht fertiggestellt werden. Könnten andere Engineering-Branchen wie die Fahrzeugindustrie, der Anlagenbau, die Elektronikindustrie, ... mit diesen oder ähnlichen Tatsachen überleben? Warum ist dies in der Software-Branche anders? Muss dies so sein?

Software-Projekte ...

Ein Software-Projekt ist im Grunde nichts ungewöhnliches im Vergleich zu anderen Projekten. Es ist ebenfalls gekennzeichnet durch die klassischen Projekteigenschaften:

- **inhaltliche Zielsetzung und Neuartigkeit**
(„Ich möchte ein neues Warenwirtschaftssystem“)
- **zeitliche Zielsetzung**
(„Echtbetrieb des neuen Systems ab 1.9.2004“)
- **beschränkte Ressourcen**
(„Kosten darf es max. €500.000,--“)
- **Komplexität**
(„Es gibt verschiedenste Schnittstellen zu anderen Systemen, die Altdaten sind zu übernehmen, ...“)

Daraus resultiert natürlich ein entsprechend erhöhtes Risiko (technisch, zeitlich, organisatorisch, personell, finanziell, ...).

Um diese Risiken einigermaßen beherrschbar zu machen, gibt es eine Vielzahl an Methoden und Vorgehensweisen, die in den klassischen Engineering-Bereichen auch bereits meist mit Erfolg angewandt werden:

- **Detaillierte und klare Spezifikationen**

- **Ausführliche Projektplanung**
- **Risikoanalyse des Projekts (vor Projektstart)**
- **begleitendes Qualitätsmanagement**
- **begleitendes Controlling (laufend)**
- **Ausführliche Tests während der Entwicklung und vor der Auslieferung**
- ...

Für IT-Projekte gibt es diese Methoden und Vorgehensweisen natürlich ebenfalls. Die Wissenschaft hat fast seit Beginn der ersten Software-Projekte entsprechende Methoden bereitgestellt und weiterentwickelt.

... sind anders!

Doch warum werden diese Methoden in den IT-Projekten oft nicht verwendet oder angenommen (Ausnahmen bilden sicherheitskritische Bereiche)?

Die Gründe dafür sind sehr vielschichtig. Aus Sicht der IT stellen sich die Problembereiche in den vorher genannten Methoden und Vorgehensweisen wie folgt dar:

- **Detaillierte Spezifikation**

Ein gutes Projekt beginnt mit einer Spezifikation und wird maßgeblich durch die Qualität der Spezifikation beeinflusst.

Es gilt das Motto „Sage mir wie dein Projekt beginnt, und ich sage Dir, wie es endet.“

Da eine gute Spezifikation auch aufwändig zu erstellen ist, ist einer der Hauptfehler, das hier aus Zeit- und Ressourcen-Gründen versucht wird, zu sparen.

In vielen Fällen wird versucht, die ‚Verantwortung‘ auf die Lieferanten abzuschieben. Aus unzähligen Präsentationen und Vorschlägen der Anbieter wird versucht, Ideen zu sammeln und schnell zu einer groben Leistungsbeschreibung zu kommen, die dann meist als Pflichtenheft bezeichnet wird. Dies ist jedoch oft sehr dürftig. Viele Auftraggeber geben sich dann auch noch der Illusion hin, sich mit einer Präsentation und eventuell noch mit einer Testinstallation von einigen Tagen auf ein System festlegen zu können.

Wenn dies im Zusammenhang mit einer ‚Schmalspur‘-Spezifikation / Ausschreibung praktiziert wird, handelt der Auftraggeber nach dem beim Autokauf bekannten Prinzip „Wie besichtigt und probegefahren“. Der Lieferant hat dann ein leichtes Spiel, wenn im Nachhinein vom

Auftraggeber Leistungen und Funktionen erwartet werden, die in der präsentierten Software nicht enthalten waren.

Derartige Situationen zeigen sich dann meist in bekannten Standardaussagen der Auftraggeber: „Das haben wir uns aber anders vorgestellt!“, „In einer Standard-Lösung muss das doch enthalten sein!“, „So können unsere Anwender das unmöglich bedienen!“, ...

Ein weiteres großes Problem im Bereich der Spezifikation ist, dass - wenn überhaupt - sehr oft nur die *funktionalen* Bereiche der Software („Das System soll Artikel wie folgt erfassen ...“, „Wir benötigen einen Report X“, ...) im Detail spezifiziert werden. Die *nicht funktionalen* Bereiche (Qualität, Ergonomie, Sicherheit, ...) werden zumeist gar nicht oder oft nur sehr ungenau spezifiziert („Das System soll ausreichend performant sein.“, „Die Software soll bei einem Absturz keine Daten verlieren“, ...). Die Gründe liegen vor allem darin, dass die nicht funktionalen Bereiche einerseits teilweise sehr subjektive Eindrücke des Auftraggebers widerspiegeln und andererseits diese Bereiche sehr schwer ‚greifbar‘ sind und damit wiederum einen großen Aufwand für die Spezifikation bedeuten.

Leider führen in Software-Projekten jedoch genau diese nicht funktionalen Bereiche oft zu Konflikten und nachträglich höheren Aufwänden.

Ein exzellenter Projektleiter und ein ebensolches Projektteam können natürlich Fehler in der Spezifikation zudecken und dem Projekt trotzdem noch zu einem Erfolg verhelfen. In den meisten Fällen jedoch werden Fehler in der Spezifikation den Projektverlauf massiv beeinflussen.

- **Ausführliche Projektplanung und Risikoanalyse des Projekts**

Dies ist natürlich nur dann sinnvoll möglich, wenn vorher auch die Anforderungen klar spezifiziert wurden.

Da dies - wie vorher dargestellt - vielfach nicht oder nur unzureichend passiert, wird die Projektplanung und Risikoanalyse großteils nur auf der Basis von Vermutungen, Annahmen und vagen Schätzungen durchgeführt, was natürlich ab einem gewissen Komplexitätsgrad zwangsläufig zum Scheitern verurteilt ist.

- **begleitendes Qualitätsmanagement und ausführliche Tests**

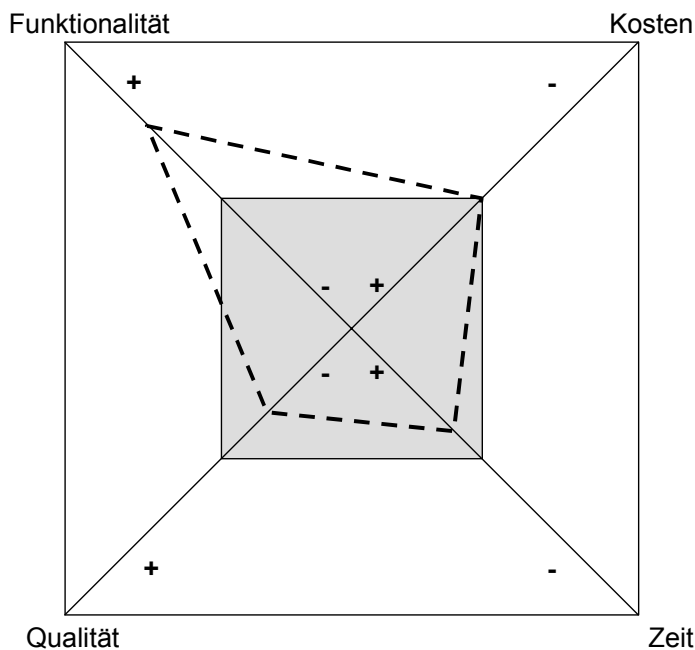
Qualität kostet Geld! Und die meisten Auftraggeber beauftragen natürlich entsprechend sparsam. Jedoch wird hier leider nur dem Anschein nach und sehr kurzfristig gespart.

Schlechte Qualität wirkt sich meist langfristig aus. Die Wartungs- und Betreuungskosten steigen dadurch an. Oft können durch schlechte Qualität verursachte Kosten auch nicht leicht

erfasst werden und verschwinden dann meist in den Gemeinkosten (z.B. wenn aufgrund eines Designfehlers die Anwender für einen Vorgang 5 Minuten benötigen, der bei optimalem Design in 2 Minuten abzuwickeln wäre).

- **begleitendes Controlling (laufend)**

Ein Software-Projekt befindet sich immer in einem Spannungsfeld aus den Parametern Funktionalität, Zeit, Kosten und Qualität.



Wenn man diese Parameter nicht permanent kontrolliert und entsprechende Maßnahmen zur Gegensteuerung durchführt, verändern sich diese Parameter fast unvermeidbar von selbst und führen am Projektende oft zu unliebsamen Überraschungen.

Meist ist es so, dass sich die Funktionalität/Leistung erhöht. Da die Kosten oft pauschaliert sind und Terminüberschreitungen auch nur in geringem Maß toleriert werden, sinkt meist die Qualität der Ergebnisse entsprechend.

Weitere Gründe für die Schwierigkeiten in Softwareprojekten liegen nicht in den funktionalen oder technischen Bereichen sondern sind oft nicht technische wie z.B. die Person des Projektleiters, die Kommunikation zwischen den Projektpartnern,

Durch den permanenten Druck der IT-Industrie mit ihren extrem kurzen Innovationszyklen und der dadurch millionenfach verkauften fehlerbehafteten Software haben wir zumindest auch im privaten

Bereich gelernt, mit Software-Fehlern zu leben. Dies wirkt sich daher leider auch auf den professionellen Software-Einsatz aus.

Das muss nicht so sein!

Software-Projekte sind objektiv betrachtet eigentlich nicht anders als klassische Engineering-Projekte!

Da leider in der Vergangenheit viele Personen, Unternehmen und Organisationen schlechte Erfahrungen in Software-Projekten und -Produkten gemacht haben, leiden diese jedoch von vornherein unter einem schlechten Image:

- Software-Projekte kosten immer mehr, als man vereinbart.
- Software ist fehlerbehaftet, man muss mit den Fehlern leben.

Zusätzlich kommt erschwerend hinzu, dass Software allgegenwärtig ist, jeder damit in Berührung kommt und von den Software-Anbietern suggeriert wird, dass der Computer und die Software kinderleicht zu bedienen sind (wie meine Kaffeemaschine).

Durch neue Technologien wie Web-Services oder Rapid-Application-Development werden derartige Fehleinschätzungen noch weiter verstärkt, da die Hersteller dieser Tools und Technologien glaubhaft machen, dass auch komplexe Software-Projekte nur noch wenig Aufwand bedeuten (Werbe-Aussagen wie „Web-Shop-Erstellung in wenigen Minuten“).

Da das alles so einfach und unkompliziert präsentiert wird, müssen dann scheinbar auch die Software-Projekte, die man im eigenen Unternehmen startet oder beauftragt, entsprechend einfach abzuwickeln sein.

Anscheinend ist dies sehr weit im Unterbewusstsein der Entscheidungsträger und Projektverantwortlichen verbreitet, denn sonst würde nicht an so viele Software-Projekte mit einer fast naiven Unbefangenheit und Leichtigkeit herangegangen werden.

Natürlich spielt auch die Technologie eine wichtige Rolle. Jedoch ist diese Rolle einerseits bei weitem nicht so gewichtig, wie manche potentiellen Auftragnehmer suggerieren, und andererseits auch nicht an erster Stelle (bevor man eigentlich noch genau weiß, was man eigentlich benötigt, weiß man jedenfalls schon, dass die neue Lösung web-service-fähig sein soll!).

Um die oft vorherrschende miserable Situation bei den Software-Projekten zukünftig zu verbessern, ist eine Bewusstseinsbildung bei den Entscheidungsträgern und Projektverantwortlichen notwendig.

Es gibt einige einfache Grundsätze, die jeder am Projekt Beteiligte wissen und anwenden sollte, um Fehleinschätzungen und Fehlschläge zu vermeiden:

- Software-Projekte sind komplex.
- Software-Projekte sind wie ein klassisches Engineering-Projekt zu sehen und daher auch so abzuwickeln.
- Software, die an die eigenen Bedürfnisse angepasst wird (auch wenn es sich um günstige Standard-Software handelt), kostet entsprechend viel (wenn man ein günstiges Auto kauft und es gerne etwas tiefer gelegt, um 30 cm verlängert und mit gestreifter Lackierung haben will, kostet dies ja auch unverhältnismäßig mehr als die Standard-Ausführung).
- Qualitativ hochwertige Ingenieur-Leistungen benötigen entsprechende Zeit für Planung und Abwicklung (ein Haus wird ja meistens auch nicht nach der ersten schnellen Handskizze gebaut, sondern bedarf detaillierter Entwürfe und Planungen, um die Bedürfnisse optimal abzudecken).

10 Goldene Regeln (Erfolgsfaktoren) des Software-Projektmanagements:

(aus Sicht des Auftraggebers)

- 1.) **Genau und möglichst vollständig spezifizieren!**
Die meisten Probleme resultieren aus einer unklaren oder unvollständigen Spezifikation. Wer hier spart, spart am falschen Platz. Wenn intern keine Zeit, zu wenig Know-How oder ev. auch nur Schwierigkeiten bei der richtigen Formulierung vorhanden sind, sollte hier auf jeden Fall Hilfe von Außen hinzugezogen werden. Jedoch sollte man nicht den Fehler machen, einen potentiellen Software-Anbieter mit dieser Aufgabe zu betrauen. Dann erhält man mit ziemlicher Sicherheit ein System, das der Anbieter liefern kann (möchte) und nicht das, das man eigentlich benötigen würde.
- 2.) **Einen realistischen Kosten- und Zeitrahmen kalkulieren!**
Erfahrungsgemäß ändern sich Anforderungen im Laufe des Projekts
→ addiere bei ungenauer Spezifikation 50-100% und bei genauer Spezifikation ca. 20% zu den Preisen und Projektzeiten der Anbieter).
Nicht auf die internen Kosten vergessen!
- 3.) **Auch die nicht funktionalen Anforderungen berücksichtigen!**
Anforderungen hinsichtlich Qualität, Ergonomie, Sicherheit, Dokumentation, Testdurchführung, ... werden zumeist gar nicht oder oft nur sehr ungenau spezifiziert. Dies führt dann oft am Projektende oder am Beginn des Echtbetriebs zu Konflikten zwischen Auftraggeber und Lieferant.
- 4.) **Das System nicht nur aus der Sicht von heute planen, sondern die Anforderungen von morgen spezifizieren!**
Es besteht die Gefahr, dass man ‚alten Wein in neuen Schläuchen‘ erhält. Innovative Wege werden dadurch oft nicht erkannt und eventuell sogar auf Jahre blockiert.
- 5.) **Auf die Projekt-Verantwortung nicht vergessen – auch der Auftraggeber benötigt einen Projektleiter!**
Es kommt häufig vor, dass der Lieferant nach dem Projektstart bis zum vereinbarten Realisierungstermin großteils allein gelassen wird (z.B. aus Zeitmangel oder fehlendem Projektleiter-Wissen beim Auftraggeber). Dadurch verliert der Auftraggeber die Kontrolle über das Projekt. Es ist unbedingt erforderlich, dass es auf Auftraggeberseite auch einen (1) Verantwortlichen für das Projekt gibt, der sich auch über den gesamten Projektverlauf entsprechend darum kümmert (Projektmanagement, Projektcontrolling, Qualitätssicherung, ...). Wenn der Auftraggeber selbst dazu nicht in der Lage ist, kann diese Aufgabe eventuell auch ein externer Projektmanager übernehmen.
- 6.) **Nicht zu sehr auf die Technologie konzentrieren.**
Manche Projekt-Mitarbeiter (und teilweise auch Verantwortliche) sind oft technologie-verliebt und konzentrieren sich daher zu stark auf technologische Feinheiten, die eigentlich für den Projektverlauf irrelevant sind. Dadurch geht viel wertvolle Zeit verloren. Die meisten Projekte scheitern jedoch nicht an der Technologie, sondern an anderen Problemfeldern wie der Methodik, der Kommunikation, den Personen, ...

- 7.) **Darauf achten, dass alle künftigen ‚Stakeholder‘ berücksichtigt und in das Projekt einbezogen werden!**
Wenn die späteren Nutznießer und auch Benutzer einbezogen werden, verläuft das Projekt meist zielgerichteter und die Akzeptanz wird deutlich verbessert.
- 8.) **Die Anbieter-Auswahl sorgfältig vornehmen!**
Oft reicht ein Studium der Funktionsliste oder eine Folien-Präsentation nicht aus, um den wahren Gehalt eines Software-Systems beurteilen zu können. Die Zeit für eine Probeinstallation, in der man sich dann natürlich auch entsprechend intensiv mit der Software auseinandersetzen sollte, ist sicherlich gut investiert. Um den Blick für das Wesentliche nicht zu verlieren und sich nicht von einigen netten ‚Gimmicks‘ blenden zu lassen, sollte außerdem eine detaillierte Nutzwertanalyse zur Bewertung durchgeführt werden. Damit ist die Wichtigkeit der einzelnen Bereiche für beide Partner transparent und auch die Abnahme scheitert dann nicht an einer Kleinigkeit.
- 9.) **Die Balance im Spannungsfeld Funktionalität, Zeit, Kosten und Qualität halten und ‚ein faires Spiel spielen‘!**
Oft wächst die Komponente ‚Funktionalität‘ im Projektverlauf unscheinbar aber stetig aufgrund von Anforderungsänderungen des Auftraggebers. Es ist oft so, dass der Auftraggeber dann meint, bei einem Fixtermin und pauschalierten Kosten dem Lieferanten auch noch verschiedenste zusätzliche Funktionalitäten im Projektverlauf ‚unterjubeln‘ zu können. Oft ist der Druck entsprechend groß, sodass dies vom Lieferanten hingenommen wird. Der einzige Parameter, der dann noch für den Lieferanten veränderbar ist, ist die Qualität. Er wird daher dann in diesem Bereich einen Ausgleich schaffen, da dies vom Auftraggeber meist nicht spezifiziert wurde und daher kaum kontrollierbar ist. Und zumeist schlägt sich schlechte Qualität dann langfristig in der Wartung nieder. Der Auftraggeber wird daher oft für kurzfristige, scheinbare Verhandlungserfolge langfristig zur Kasse gebeten.
- 10.) **Genügend Zeit für den Test und die Abnahme der Software einplanen!**
Gerade hier wird oft der Fehler gemacht, dass man nicht so genau prüft, auf einen Probetrieb verzichtet und sich auf die Gewährleistungsfrist verlässt. Sofern eine Gewährleistungsfrist nicht ausgeschlossen wurde, muss der Lieferant natürlich nachbessern. Jedoch ist das System dann eventuell schon im Echtbetrieb und daher jede Störung um ein Vielfaches teurer als vor der Inbetriebnahme.

Über den Autor



Herr Dipl.-Ing. Johannes Bergsmann ist gerichtlich beideter Sachverständiger für Informatik und war Gründer und 6 Jahre Geschäftsführer eines Software-Engineering-Unternehmens im Großraum Linz und anschließend 2 ½ Jahre in einem Systemhaus in Linz als Gesellschafter und technischer Leiter tätig.

Anfang 2003 gründete Herr Bergsmann das Unternehmen ‚Software Quality Lab‘. Herr Bergsmann gibt sein Wissen auch als Lehrbeauftragter an der HTL-Leonding im Bereich Software-Projekt- und Qualitätsmanagement weiter und ist Mitglied in der STEV-Österreich (österreichische Vereinigung für Software-Qualitätsmanagement), der OCG (Österreichische Computer Gesellschaft) und der ADV (Arbeitsgemeinschaft für Datenverarbeitung).

Privat: verheiratet, 2 Kinder, Hobbies: Singen, Gitarre, Fotografieren

Rückfragen bitte an:

SOFTWARE QUALITY LAB
Dipl.-Ing. Johannes Bergsmann

Fliederstraße 8
4222 Langenstein

Tel.: +43-664-1620220 oder +43-7237-64370

Fax: +43-7237-4941

e-mail: johannes.bergsmann@software-quality-lab.at

Web: www.software-quality-lab.at