Projekthoheit durch iterative Softwareentwicklung

Jan Bruns
up to data professional services GmbH
Wörrstadt

Schlüsselworte
Softwareentwicklung, Probleme/Fehler, Projektmanagement, Entwicklungsmodelle, Iterationen, V-Modell, Risikomanagement, Anforderungen

Einleitung

In den letzten Jahren ist die Zahl der scheiternden IT-Projekte stetig angewachsen. Dem „CHAOS Summary Report der Standish Group“ aus dem Jahr 2009 zu Folge sind nur 32% aller IT-Projekte überhaupt erfolgreich. Ein erfolgreiches Projekt ist im CHAOS Summary Report nicht Sache des Betrachters sondern ist klar mit „on time, on budget, with required features and functions“ definiert.
Die Ursachen für das Scheitern der Projekte sind vielfältig und werden später näher betrachtet. Was jedoch gerade im stark dynamischen Umfeld der Softwareentwicklung auffällt ist, dass die traditionellen sequenziellen Entwicklungsmodelle wie das V-Modell den Projekterfolg nur bedingt unterstützen. Die strikte Teilung in einzelne Projektphasen, die sequenziell nacheinander abgearbeitet werden, kann dazu führen, dass ein in der Integrationsphase auftretender Fehler zu einem kompletten Neuentwurf der gesamten Software führt. Spätestens dann kann das Projekt nicht mehr erfolgreich gestaltet werden. Die komplette Funktionalität kann durch eine notwendige Systemänderung noch erreicht, aber das Projekt wird nicht mehr „on time“ und „on budget“ abgeschlossen werden können.
Das Scheitern des Projektes ist bedingt durch das gewählte Entwicklungsmodell und dem damit verbundenen zu späten Erkennen des Designfehlers in der Integrationsphase. An diesem Manko setzten die iterativen und agilen Entwicklungsmodelle an. Im Rahmen dieses Vortrages soll nur der iterative Ansatz und nicht die agilen Methoden betrachtet werden - „Projekthoheit durch iterative Softwareentwicklung“.

Erfolgreiche Projekte und Ihr Umfeld

Die Hauptgründe für das Scheitern von IT Projekten sind gemäß dem CHAOS Summary Report eine unvollständige Anforderungsanalyse, das Nichteinbeziehen der Business User sowie unrealistische Annahmen und sich ständig ändernde Anforderungen. Diese Ursachen werden durch eine Studie von Gartner aus dem Jahre von 2004 unterstützt (siehe Abb.1).
Laut der Gartner Studie wird fast die Hälfte der ausgelieferten Software mit ungeeignet bewertet (46%) und 38% der Endanwender lehnen die Software ab.

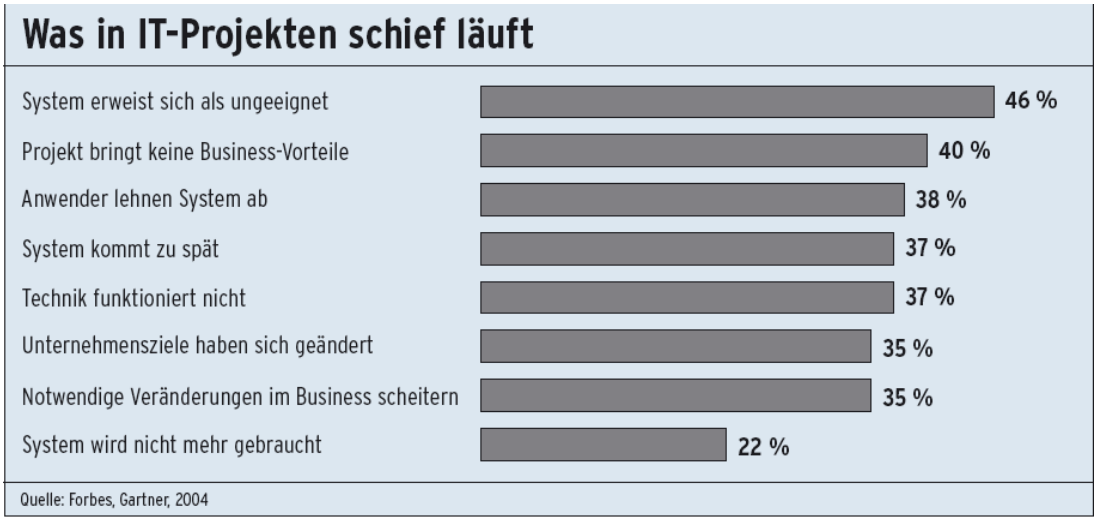

Abb. 1: Ursachen für das Scheitern von IT Projekten (Quelle Forbes, Gartner 2004)

Beide Studien zeigen, dass die Endanwender die maßgebliche Komponente für den Erfolg des Projektes und die Akzeptanz des Endproduktes – der Software sind. Nur durch ein frühes Einbeziehen der Benutzer können die wirklichen Anforderungen definiert und eine Fehlentwicklung vermieden werden. Im CHAOS Summary Report ist diese Erfolgsregel in ersten von „10 Laws of CHAOS“ festgehalten „Users(?) are your best and worst enemy“.

Des Weiteren wird durch die Zahlen deutlich, dass in vielen Fällen zu Beginn des Projektes nur das Ziel der Lösung definierbar ist. Die damit verbundenen Anforderungen an die Software, die den späteren Einsatz der Lösung in den Fachabteilungen beschreiben sind nicht klar definierbar.
Aus diesen Misserfolgsfaktoren können die Erfolgsfaktoren für Projekte direkt abgeleitet werden. Diese sind neben der hohen Anwenderbereitschaft die Unterstützung des Managements, sowie eine realistische und anforderungsgetriebene Planung.
Die Erfolgsfaktoren müssen durch entsprechende Rahmenbedingungen wie eine dedizierte Projektleitung, ein fortlaufendes Projectcontrolling und durch kompetente Projektmitarbeiter unterstützt werden.
Doch wie können diese Erfolgsfaktoren weiter ausgebaut und so die Wahrscheinlichkeit des Projekterfolges noch weiter erhöht werden?
Neben offener Kommunikation innerhalb des Projektteams lautet die Antwort durch die Wahl eines modernen iterativen Entwicklungsmodelles.

Traditionelle Entwicklungsmodelle
Traditionelle, sequenzielle Entwicklungsmodelle wie das V-Modell (siehe Abb. 2) sind starr. Die einzelnen Projektphasen/Arbeitsschritte laufen einmalig nacheinander ab. Die Software wird spezifiziert, das Design erstellt, implementiert und die Anwendung getestet.

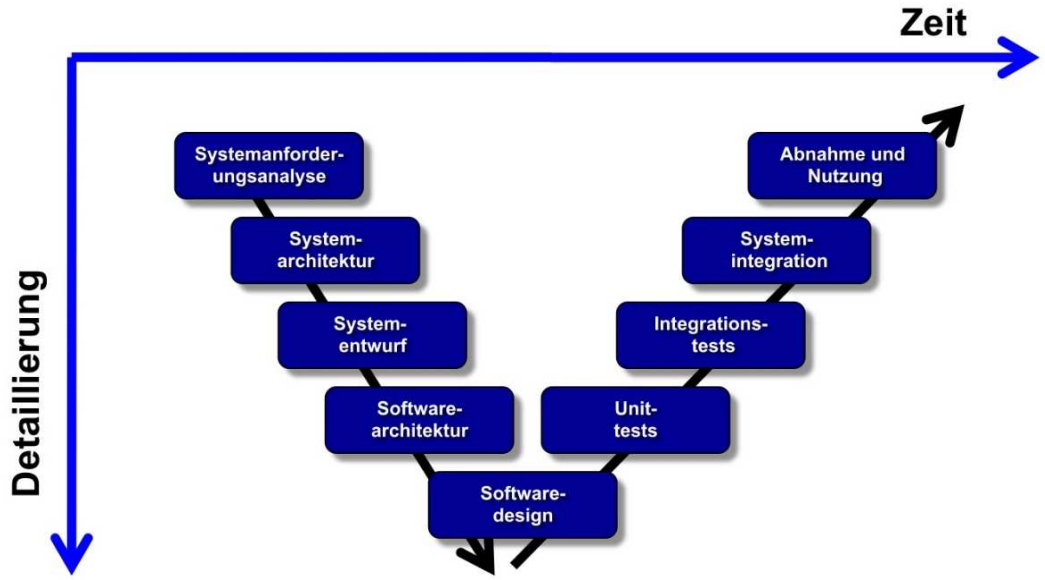

Abb. 2: V-Modell

Dieser starre Ablauf ist jedoch nur für stabil definierbare Anforderungen/Aufgabenstellungen anwendbar und geeignet. Was passiert wenn erst in den Integrationstests oder bei der Übergabe an den Endanwender Fehler identifiziert werden?
Ein solches Problem kann leicht zu einem komplett neuen Softwaredesign führen, verbunden mit erheblichen Kosten und Verzögerungen.

Iterative Softwareentwicklung
Im Bereich der Softwareentwicklung unterliegen Projektanforderungen einer großen Dynamik.
Nur selten können zu Projektbeginn Anforderungen vollständig definiert werden. Dies liegt unter anderem daran, das nicht die IT Abteilung der Endanwender der Software ist sondern die Fachabteilung. Die Fachabteilung kann jedoch zu dem frühen Zeitpunkt, an dem in traditionellen Prozessmodellen die Anforderungen definiert werden, diese nicht vollständig und abschließend beschreiben. Die Lösung existiert nur „auf dem Papier“ und die Auswirkungen der Anforderungen können nicht erkannt werden.
Diesen dynamischen Anforderungen und dem damit verbundenen Kosten- und Zeitdruck des Projektes begegnen die iterativen Entwicklungsmodelle.

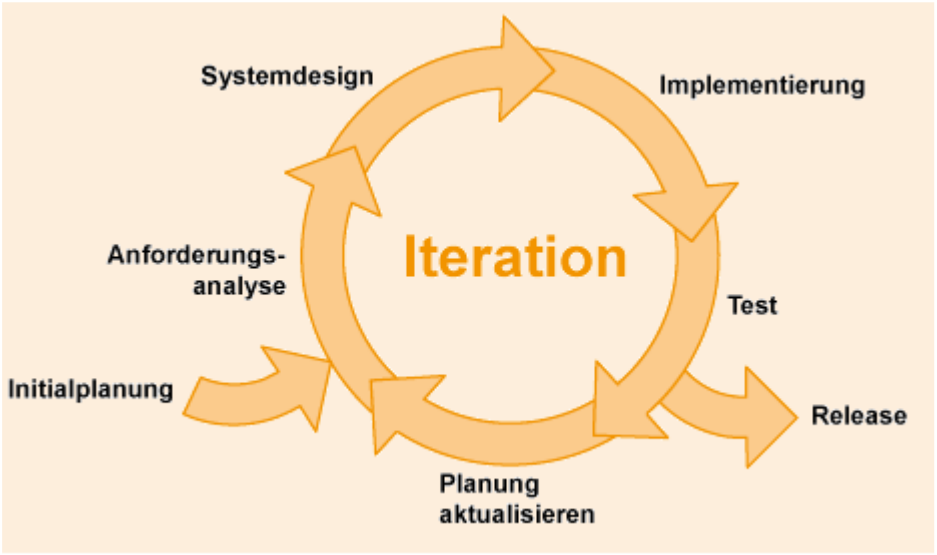

Abb. 3: Iterationszyklus

In der iterativen Softwareentwicklung wird die Anwendung in kleinere lauffähige Releases unterteilt. Diese Release werden frühzeitig durch den Endanwender getestet und dessen Feedback, Änderungswünsche und Findings fließen direkt in die Entwicklung mit ein. Meist können die Punkte bereits im nächsten Release behoben werden. Das rechtzeitige Einbinden der Endanwender minimiert das Risiko, das falsche Produkt zu entwickeln oder ein nicht den „wirklichen" Anforderungen entsprechendes Produkt auszuliefern.
Jede Iteration hat eine feste Dauer in der Regel liegt diese bei vier bis sechs Wochen. Die Anzahl der benötigten Iterationen und die Dauer der einzelnen Iterationen hängen von der Größe des Projektes und Komplexität der zu implementierenden Funktionen ab. Unmittelbar vor jeder Iteration wird deren Inhalt und die umzusetzenden Features/Funktionen definiert. Jede Iteration besteht aus einem zyklischen Prozess. Der Iterationszyklus umfasst die Planung, Anforderungsanalyse, das Design, die Implementierung, die Tests der implementierten Funktionen und die Bewertung der Ergebnisse. Nach jeder Iteration wird die Gesamtplanung aktualisiert.
Jeder Testphase einer Iteration geht ein „Kickoff-Meeting" voran. Hier wird das zu testende Release den Endanwendern vorgestellt und die neuen Features werden durch das Projektteam demonstriert. In der Testphase der Iteration wird das Release von den Endanwendern überprüft und verifiziert. Findings und Verbesserungsvorschläge werden in „Issuelisten" festgehalten. Diese Issuelisten werden in einem abschließenden „Review-Meeting" für jede Iteration vom Projektteam kategorisiert und bewertet. So ist eine effiziente Weiterverfolgung der Punkte bis hin zu einer notwendigen Aktualisierung der Anforderungsdefinition möglich.
Die Endanwender spielen in der Testphase des Iterationszyklus die Schlüsselrolle. Ohne deren Bereitschaft und der Freistellung von anderen Aufgaben während der Testphase wird keine effektive Iteration möglich sein. Um dies zu erreichen ist die Unterstützung des Managements notwendig.

In der iterativen Softwareentwicklung entsteht die vollständige Softwareware durch die einzelnen Iterationen. Das System wächst mit jedem Release. Wie kann dabei vermieden werden, dass das aktuelle Release Wechselwirkungen mit den vorhergehenden Releases erzeugt?
Die Änderung der Funktionalität erfolgt immer als Erweiterung. Eine neue Alternative wird immer an einem vorhandenen Variationspunkt hinzugefügt. Wenn kein Variationspunkt vorhanden ist, muss dieser durch das Einführen eines neuen Interfaces geschaffen werden. Erst danach kann die neue Funktionalität aufgenommen werden. Um die Codequalität sicherzustellen und zu vermeiden, dass

Funktionen doppelt implementiert werden, ist ein Refactoring zu Beginn einer neuen Iteration notwendig. Die Modell- und Codestruktur wird bereinigt, Methoden und Funktionen werden gekapselt und Redundanzen beseitigt. Diese Optimierung muss so erfolgen, dass die Software in seiner Gesamtfunktionalität unverändert bleibt.
Jede Codeoptimierung birgt ein Risiko, dass Fehler in das System eingebracht werden, daher ist es zwingend erforderlich Unit-Testfälle zu definieren und nach Möglichkeit Hilfswerkzeuge zu nutzen oder automatisierte Testfälle durchzuführen.
In den sequenziellen Entwicklungsmodellen wird der Anteil der Qualitätssicherung mit etwa 10% angesetzt, dieser Wert ist für die iterative Softwareentwicklung nicht ausreichend und sollte bei etwa 15% liegen.
Im Projektbudget, müssen diese Optimierungsschritte vorab einkalkuliert werden, da diese notwendigen Schritte später nur schwer gegenüber dem Auftraggeber durchsetzbar sind.

Durch die iterative Softwareentwicklung ergeben sich klare Vorteile für den Auftraggeber und die Softwareentwickler. Die leichtgewichtigen Prozesse ermöglichen, dass die Anwendung mit jeder Iteration wächst und flexibler angepasst werden kann. Eine lückenhafte oder noch nicht vollständig vorhandene Anforderungsdefinition fällt geringer in das Gewicht, da diese iterationsbegleitend vervollständigt und korrigiert werden kann. Ein Projekterfolg durch die frühere Einbeziehung derer, die mit der Software arbeiten müssen wird wahrscheinlicher und damit auch der finanzielle Erfolg. Die Software wird nutzbar sein und wird sich selbst bei einem nicht kostendeckenden Projekt durch den Einsatz zügig amortisieren (ROI).

Praxisbeispiel
An Hand der Implementierung einer Reportinglösung für Zulassungsberichte für die Firma Amgen durch up to data soll die iterative Softwareentwicklung veranschaulicht werden.
Das Projekt wurde durch die IT Abteilung von Amgen in Denver, USA initiiert und mit up to data in Wörrstadt durchgeführt.
Da die Flexibilität der Berichte und die Qualität der Ausgangsdaten, sowie die gelebten Workflows innerhalb der Fachabteilung nur begrenzt bekannt waren, wurde ein iterativer Projektansatz gewählt. Statt an der Erstellung der übergeordneten Anforderungsdefinition zu arbeiten wurde durch up to data ein Prototyp der Reportinglösung entwickelt und im Rahmen eines initialen Kickoff Workshops den Endanwendern vorgestellt. An Hand dieses Prototyps wurde gemeinsam die erste Entwurfsversion der Anforderungsdefinition erarbeitet. Mit diesem Entwurf startete die erste von drei Iterationen. Die Iterationen hatten eine Dauer von jeweils sechs Wochen, wobei die letzten beiden Wochen der Iteration die Testphase der jeweiligen Releases durch die Endanwender bildeten. Jeweils eine Woche zu Beginn der Iteration wurde für die Qualitätssicherung des Quellcode und das Refactoring aufgewendet. Da es sich um ein virtuelles Projektteam handelte, wurden die Iterationen durch eine initiale Webkonferenz eingeleitet. Während der Iterationen wurden Findings und Anmerkungen in Issuelisten festgehalten. Eine Review-Webkonferenz schloss jede Iteration ab.
Das Kernprojektteam erstellte parallel zu den ersten beiden Iterationen die endgültige Anforderungsdefinition. Diese basierte auf diesen Issuelisten und dem geäußerten Feedback der Anwender in den Webkonferenzen. Mit der Freigabe bildete diese Anforderungsdefinition die Grundlage für die dritte Iteration.
Die dritte Iteration war somit der formale Abschluss der Entwicklung. Der Fokus der dritten Iteration lag auf der technischen Dokumentation und dem Durchführen der Validierungstests gegen die nun freigegebene Anforderungsdefinition.
Trotz der Komplexität und des virtuellen Projektteams konnte das Projekt innerhalb von nur fünf Monaten erfolgreich angeschlossen werden.
In den abschließenden Lessons Learned Meetings wurde schnell klar, dass die Fachabteilung positiv überrascht war, dass die Lösung alle benötigten Workflows und die wirklich benötigten Features

beinhaltete. Dies war in den IT Projekten zuvor nicht immer der Fall gewesen. Die Projekte hatten formal alle Features implementiert, diese entsprachen aber nicht den eigentlichen Anforderungen. Auch betriebswirtschaftlich war das Projekt ein Erfolg. Der Projektzeitplan wurde eingehalten und durch das gemeinsame Evaluieren der wirklichen Anforderungen mit den Anwendern konnte die benötigten Features innerhalb des geplanten Budgets implementiert werden.

Fazit

Wie jedes andere Entwicklungsmodell hat auch die iterative Softwareentwicklung neben seinen Stärken auch seine Schwächen. Um iterative Softwareentwicklung effizient zu betreiben, müssen immer die Unterstützung des Managements und die Verfügbarkeit von den Endanwendern während der Iterationen sichergestellt sein.
Wenn dies der Fall ist, sollte im Interesse des Projekterfolges ein iterativer Ansatz gewählt werden. Auf dem oben beschrieben iterativen Ansatz setzten viele der heute bekannte Entwicklungsmodelle wie die agilen Methoden oder auch RUP (Rational Unified Process) auf.
Aus diesen verschiedenen Modellen muss für jedes Projekt und für jedes Projektteam des am besten geeinigten Entwicklungsprozess gewählt werden.
Das Entwicklungsmodell wird immer direkt von der Firmenkultur, den zur Verfügung stehenden Mitarbeitern und deren Bereitschaft sich einzubringen abhängen. Eine aktive und offene Kommunikation innerhalb des Projektes und mit den Endanwendern ist immer notwendig um das Projekt unabhängig vom gewählten Entwicklungsmodell erfolgreich zu gestalten.

Im Interesse des Projekterfolgs, sollten Sie iterative Entwicklung nur für Projekte einsetzen, von denen Sie wollen, dass sie erfolgreich verlaufen (Martin Fowler, UML Distilled).

Kontaktadresse:

Jan Bruns
Up to data professional services GmbH
Schornsheimer Chaussee 7
D-55286 Wörrstadt

Telefon: +49 (0) 6732 - 949017
Fax: +49 (0) 6732 - 949041
E-Mail jan.bruns@uptodata.de
Internet: www.uptodata.de