

Innovative Lösungsansätze aus unserer aktuellen Projektarbeit

Agile Softwareentwicklung

Wie man schnell und effektiv Systeme baut, die die wirklichen Anforderungen der Anwender erfüllen

beck et al.
projects

Agile Softwareentwicklung

Wie man schnell und effektiv Systeme baut, die die wirklichen Anforderungen der Anwender erfüllen

Problemstellung Die Fragestellung ist so alt wie die Softwareentwicklung selbst: Wie konzipiert und baut man Software effektiv, schnell, effizient und qualitativ hochwertig?

- **Effektivität** steht an oberster Stelle und bedeutet, das Richtige zu bauen: Applikationen, die die tatsächlichen Anforderungen der Anwender erfüllen, auch wenn diese sie selbst eher ahnen als wirklich kennen. Applikationen, die einen echten Wertbeitrag für das Geschäft leisten und dafür angemessen dimensioniert sind.

- **Effizienz und Schnelligkeit** bedeutet die Entwicklungszeiten kurz und die Kosten niedrig zu halten, der sinkenden Lebensdauer der unterstützten Geschäftsprozesse entsprechend.

- Und schließlich hohe **Qualität**. Sie drückt sich aus in Selbstverständlichem wie Stabilität und kurzen Antwortzeiten. Entscheidender sind jedoch Fehlerfreiheit und Wartbarkeit. Hohe Softwarequalität zahlt sich aus in niedrigen Lebenszykluskosten (»Total-Cost-of-Ownership«).

Viele Vorgehensmodelle gaben seit Jahrzehnten unterschiedliche Antworten auf die Frage, wie diese Ziele sicher erreicht werden, vom klassischen, »schweren« Wasserfallmodell bis hin zum »ultraleichten« Rapid Application Development (RAD). Als Praktiker interessieren uns – mehr als die Theorie – die tatsächlichen Ergebnisse, die wir für unsere Kunden mit der Leitidee »agile Auftragsentwicklung« erreichen.

Lösungsansatz Agilität heißt Beweglichkeit. Sie setzt voraus, dass unnötiger Ballast über Bord geworfen wird. Für uns: ein iteratives, inkrementelles und leichtgewichtiges Vorgehen, um in kurzer Zeit robuste Systeme zu entwickeln, die schnell und effizient umgebaut werden können und exakt das tun, was der Anwender von ihnen erwartet. Konkret: Keine umfangreiche Detailspezifikation. Der Anwender ist im Kernteam und immer dabei. Wöchentliche Releases. Offener Umgang mit Risiken. Ständige Neu-Priorisierung der Anforderungen. Testen, Testen, Testen. Überarbeiten und Verbessern statt genialen ersten Würfeln. Einsatz von Design Patterns und Frameworks. »Extremes Programmieren«.

In konkreten Projekten haben wir überwiegend gute Erfahrungen mit agilem Vorgehen gemacht, aber auch seine Grenzen kennen gelernt. Vor allem wissen wir heute: effektive und hochwertige Software baut man effizient und schnell mit »soviel Agilität wie möglich und so viel Gewicht wie nötig«.

Zielgruppe Jeder Pragmatiker, der wissen will wie seine Organisation schnell und effizient zu den richtigen Informationssystemen kommt, also CIO's, IT-Projektleiter, Anwendungsentwickler und alle, die meinen, dass Individualsoftwareentwicklung zwangsläufig teuer ist und lange dauert.

Ziele und Grundsätze agiler Softwareentwicklung

Die Erwartungen unserer Kunden ...

Agile Softwareentwicklung ist ein zeitgemäßer methodischer Ansatz, um den Anforderungen gerecht zu werden, die heute mehr denn je an individuelle Softwaresysteme gestellt werden. Informationssysteme müssen

- einen schnellen und nachhaltigen **Return-on-Investment** erwirtschaften
- niedrige **Lebenszyklus-Kosten** (Total-Cost-of-Ownership) verursachen
- **flexibel und anpassungsfähig** an sich wandelnde Anforderungen sein
- (sehr) **kurzfristig verfügbar** sein.

... definieren für uns das Ziel

Um den Erwartungen unserer Kunden gerecht zu werden, ist unser Arbeiten konsequent auf vier Ziele ausgerichtet.

- ♣ **Schnelligkeit**
- ♥ **Effektivität**
- ♠ **Effizienz**
- ♦ **Hohe Qualität**

Diese Ziele erreichen wir mit agilen Methoden.

Leitideen des agilen Vorgehens

In agilen Softwareprojekten entsteht ein Informationssystem

- **inkrementell** – Schritt für Schritt
- **iterativ** – in mehreren sich wiederholenden Schleifen
- **leichtgewichtig** – sehr zielorientiert und ohne Ballast

und damit **schnell, flexibel, beweglich** = agil

Das agile Manifest

Im Februar 2001 formulierte die *Agile Alliance*, eine Gruppe von Befürwortern leichtgewichtiger Entwicklungsprozesse, die wichtigsten Prinzipien agiler Softwareentwicklung:

1. **Individuen und Interaktionen** sind wichtiger als Prozesse und Werkzeuge
2. **Funktionierende Software** ist wichtiger als umfassende Dokumentation
3. **Kundenzusammenarbeit** ist wichtiger als Vertragsverhandlungen
4. **Auf Änderungen reagieren** ist wichtiger als einem Plan zu folgen

Wir schätzen die Punkte auf der rechten Seite, aber wir bewerten die Punkte auf der linken Seite höher.



Das Weglaßprinzip ist Grundvoraussetzung für den Erfolg. Reinhold Messner hat mit agilen Ideen das Bergsteigen verändert.

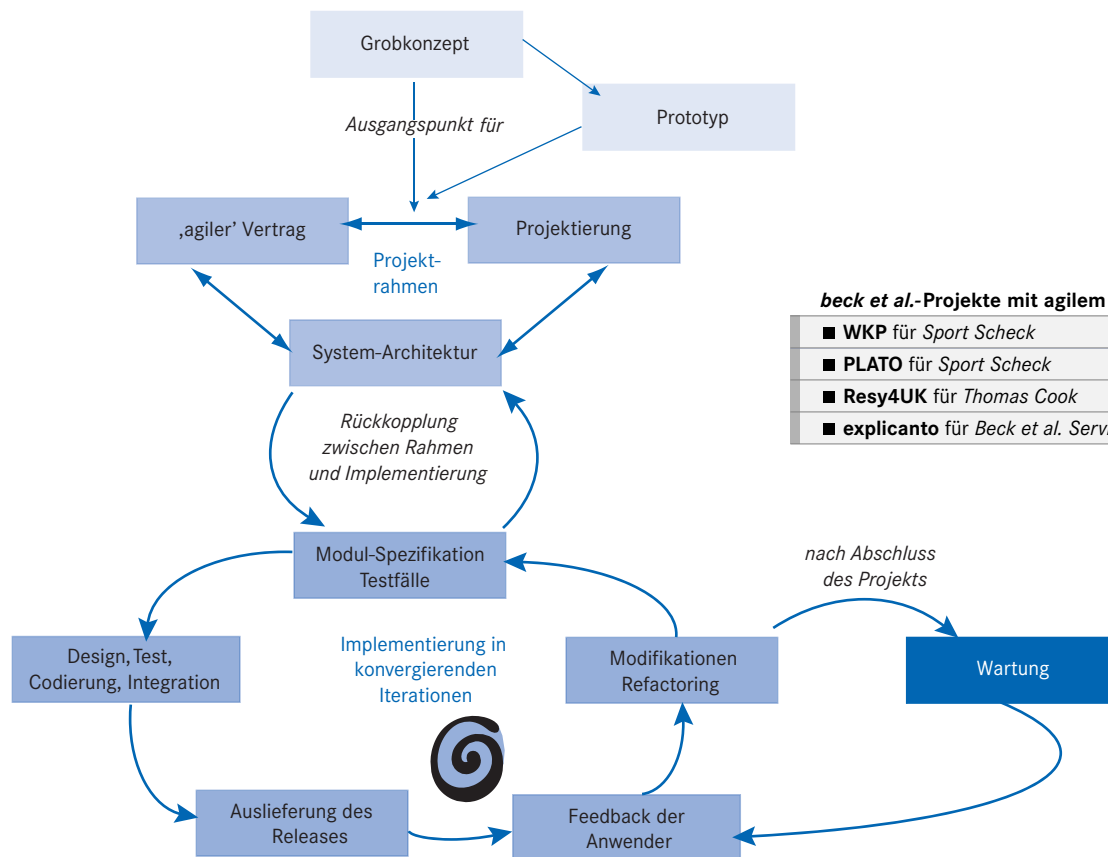
Agiles Verhalten in Softwareprojekten

1. **Frühe und zahlreiche Lieferungen hochwertiger Software** haben höchste Priorität.
2. **Funktionierende Software** muss **innerhalb weniger Wochen** oder Monate geliefert werden.
3. **Veränderte Anforderungen** sollten **immer positiv** aufgenommen werden, auch spät in der Entwicklung.
4. **Anwender und Entwickler** arbeiten **täglich gemeinsam** im Projekt.
5. Die effizienteste und effektivste Methode zur Informationsübermittlung ist die **direkte Kommunikation**.
6. Die besten Architekturen und Designs entwickeln sich in **echten Teams**, die sich selbst organisieren.
7. **Ständige Aufmerksamkeit gegenüber technisch hervorragender Qualität** verbessert die Agilität.
8. **Schlichtheit** – die Kunst, die Menge der nicht geleisteten Arbeit zu maximieren – ist essenziell.
9. Das Team muß in regelmäßigen Abständen **beraten, wie es noch effektiver arbeiten kann**.

In enger Anlehnung an die zwölf unterstützenden Aussagen der Agile Alliance, nach Alistair Cockburn

Vorgehensweise und Vorteile

Vorgehensweise



beck et al.-Projekte mit agilem Vorgehen

- WKP für Sport Scheck
- PLATO für Sport Scheck
- Resy4UK für Thomas Cook
- explicanto für Beck et al. Services

Vorteile

Effektivität

Ein agiles Vorgehensmodell ist optimal geeignet, um Geschäftsabläufe zu unterstützen, die erst parallel zur Softwareentwicklung neu gestaltet werden. Das früh verfügbare System hilft bei der Präzisierung der echten Anforderungen. Änderungen werden sofort eingearbeitet. Auf eine Veränderung der Prioritäten kann sofort reagiert werden. So entsteht ein effektives System, das auch innovative Geschäftsprozesse bestmöglich unterstützt.

Effizienz und Schnelligkeit

Ein agiles Vorgehensmodell lässt alles beiseite, das nicht direkt zum Erreichen der Projektziele beiträgt. Es orientiert sich am unmittelbaren Nutzen für den Anwender und ist daher effizient, schnell und kostengünstig.

Hohe Qualität

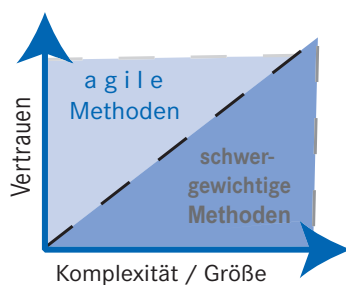
Ein agiles Vorgehensmodell erhebt die Flexibilität der Software zum Prinzip und strebt schon während der Entstehung ein permanentes Überarbeiten und Verbessern des Codes bewußt an. Wegen der häufigen Releases sind automatisierte Regressionstests unverzichtbar. So entsteht eine hohe technische Qualität, die für geringe Wartungskosten sorgt.

unsere Erfahrungen

Vertrauen und Partnerschaft

Je größer das Vertrauen zwischen Kunde und Dienstleister ist, desto leichter fällt die Umsetzung agiler Prinzipien.

Nur dann sind eine für beide Seiten attraktive Vertragsvereinbarung, ein reibungsloses tägliches Zusammenarbeiten, der Verzicht auf übermäßige Dokumentation und ein offener Umgang mit Risiken möglich. Allerdings setzt auch die Komplexität der Agilität Grenzen.



Requirements Engineering

Die funktionalen und nicht-funktionalen Anforderungen werden zunächst in einem **Grobkonzept** formuliert und die technischen und gestalterischen Ansätze durch einen Prototypen erfahrbar gemacht. Es wird eine Feature List erarbeitet und entsprechend der Prioritäten der einzelnen Funktionen sortiert.

Eine **Feinspezifikation** der Module ist dennoch nötig und erfolgt **parallel zur Implementierung** durch die Vertreter der Anwender im Projektteam.

Die Dokumentation bleibt jedoch auf die wichtigsten und komplexesten Anforderungen beschränkt. Sie dient vor allem zur Unterstützung der Kommunikation.

Die **Veränderungen bestehender Anforderungen**, die Aufnahme **neuer Anforderungen** sind zusammen mit einer **Neu-Priorisierung jederzeit möglich**.

Zusammenarbeit und Kommunikation

Die Anwender des Kunden arbeiten als **on-site customer** in einem Projektteam mit den Entwicklern direkt zusammen und nehmen unmittelbaren Einfluss auf die Entwicklung. Die (wenigen) entstehenden Dokumente stehen in einem gemeinsamen **Projektforum** jedem zur Verfügung.

Tests

Das Testen der Software spielt eine zentrale Rolle. Im Idealfall werden in einem **Test-First**-Ansatz zuerst die Testprogramme geschrieben, die das erwünschte Systemverhalten überprüfen und erst dann der eigentliche Programmcode. **Automatisiert ablaufende Modultests** (Unit Tests) und **Integrationstests** (Akzeptanztests) stellen sicher, dass dem Anwender auch bei häufiger Überarbeitung des Systems stets voll funktionsfähige Software bereit steht.

Vertragsgestaltung

Die vertraglichen Vereinbarungen müssen Flexibilität erlauben. Wir erreichen dies durch eine **Differenzierung des Budgets in Basisleistungen und optionale Leistungen**.

Projektrahmen

Auch, oder gerade agile Projekte brauchen einen festen Rahmen: **Dauer, Aufwand und Budget** werden in der Projektierung **festgelegt**. Gesteuert wird durch die Priorisierung des **variablen Funktionsumfangs**.

Architektur & Design

Einfachheit, Schlichtheit und Klarheit sind die wichtigsten Architektur- und Designprinzipien in agilen Projekten.

Ziel ist lediglich, die jeweils anstehenden Anforderungen zu lösen. Ein **Refactoring**, die Überarbeitung bestehender Codes zu einem späteren Zeitpunkt, wird nicht nur in Kauf genommen, sondern bewußt angestrebt.

Risikomanagement

Die möglichen Risiken für das Erreichen der Projektziele werden zu Beginn des Projekts gemeinsam identifiziert und mit ihrer Eintrittswahrscheinlichkeit und Schadenshöhe gewichtet. Aus dem daraus entstehenden **Risikoportfolio** werden Maßnahmen abgeleitet, um den Projekterfolg zu gewährleisten. Die kurzen Versions-Zyklen (siehe Releasemanagement) tragen in Verbindung mit ausführlichen Funktionstests (siehe Tests) zu einer jederzeit realistischen Einschätzung der Risiken bei.

Releasemanagement

Die frühe Lieferung einer ersten funktionsfähigen Systemversion bereits nach 4-6 Wochen und ein **fester 1-2 wöchentlicher Rhythmus** von Updates erlauben den Anwendern frühes und kontinuierliches Feedback. Dabei sind die Releasetermine heilig, der Funktionsumfang jedoch variabel. Jede Version wird in einem **Releasemeeting** übergeben, das gleichzeitig Einweisung der Anwender und Feedback an die Entwickler umfasst.

unsere Tools

- Eclipse, CVS
- JUnit, Ant
- Lotus Quickplace
- Lotus Sametime
- Feature-List-Datenbank
- MS Excel, MS Word
- Gummibärchen

Dokumentation

Die ideale Art und der optimale Umfang der Dokumentation wird auch in agilen Projekten heiß diskutiert. Extremisten wollen auf jegliche technische Dokumentation verzichten, die über **Kommentare im SourceCode** hinaus geht. Pragmatiker halten eine **begleitende Dokumentation**, die zum Ende des Projekts erstellt wird für richtig. Und die Kreativen schlagen die **Video-Aufzeichnung** einer Präsentation des Chefdesigners vor ...