

Mit Unsicherheiten richtig umgehen



Peter Gabriel,
Partner AWK Group

Das Ignorieren von Unsicherheiten ist einer der häufigsten Gründe, weshalb ein Projekt länger dauert und mehr kostet als ursprünglich geplant. Unsicherheit kann heissen: unklare Vorstellungen vom Endprodukt, neuartige Technologieplattform, unzuverlässige Aufwandschätzung usw. Dies trifft vor allem auf Projekte mit grossem Neuheitscharakter bzw. Innovationsanteil zu, wie zum Beispiel Entwicklungsprojekte.

Solche Projekte sind vergleichbar mit Expeditionen: einzigartig und einmalig, nicht berechenbar, das Gegenteil von Routine-Aktivitäten. Im Zentrum steht die Notwendigkeit, für jene Ereignisse vorzusorgen, die den Erfolg gefährden können. Das bedeutet gedankliches Durchspielen aller vorstellbaren Szenarien (Risikomanagement), sorgfältige Vorbereitung und Einplanung von Reserven (Projektplanung) und richtiges Handeln im entscheidenden Moment (Projektführung).

Daneben ist aber auch die „Routenwahl“ bzw. das Vorgehensmodell entscheidend. In Projektphasen mit hoher Unsicherheit sind „agile“ Vorgehensweisen dem starren Wasserfall-Modell überlegen. Mit kurzen, überschaubaren Iterationen, mit Reviews der Resultate am Etappenende und mit Plankorrekturen vor jeder neuen Etappe behält der Projektleiter das Projekt jederzeit im Griff, und Änderungen können kontrolliert ins Projekt einfließen.

Herzlich Ihr

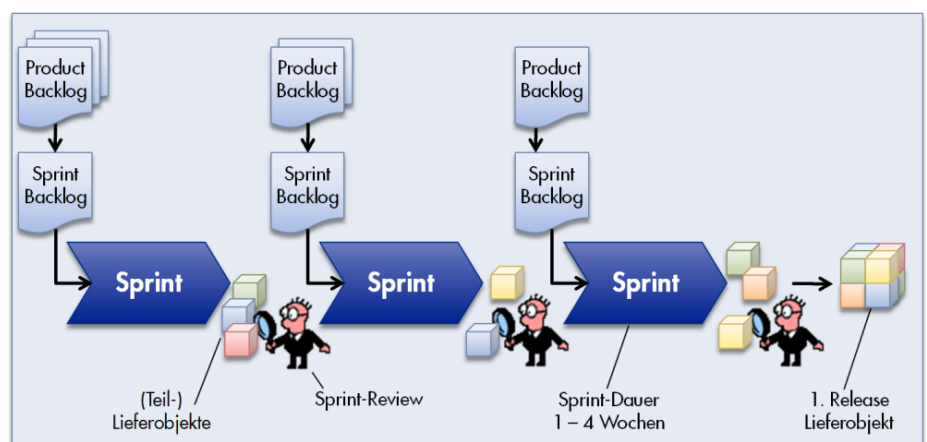
P. Gabriel

Agilere Projekte dank Methoden aus der Software-Entwicklung

Trotz immer raffinierteren Projektmanagement-Methoden, Prozessen und Werkzeugen bleibt die Zahl der Projekte, die unter Leistungsmängeln, Terminverzögerungen und Budgetüberschreitungen leiden, erschreckend hoch. Laut „Chaos Report“ der Standish Group^[1] werden 18% aller IT-Projekte abgebrochen und 53% überschreiten Kosten oder Termine. Vorgehensmodelle aus der Software-Entwicklung – angewendet auf allgemeines Projektmanagement – könnten hier Besserung bringen.

Manuel Thiemann, Dr. Richard Forster, Peter Gabriel

Projekte, die mit vielen Unsicherheiten behaftet sind, kämpfen erfahrungsgemäss oft mit Terminverzögerungen und Budgetüberschreitungen. Dies gilt besonders für Software-Entwicklungsprojekte. In diesem Bereich wurde deshalb schon früh nach Methoden gesucht, wie Projekte trotz unvollständig spezifizierbarem Endprodukt, häufigen Anpassungen und hohem Innovationsgrad zum Erfolg geführt werden können. Einer der erfolgreichsten Vertreter der als agil bezeichneten Vorgehensmodelle ist *Scrum*. Wie bei allen agilen Modellen liegt der Fokus auf einer inkrementellen Annäherung an die Lieferobjekte und zwar nach dem Grundsatz „Das Wichtige zuerst“. So erhält man schon früh erste Resultate sowie Feedback vom Auftraggeber und den Benutzern. Diese frühe Rückkopplung und die Möglichkeit, die nächsten Schritte an die veränderte Situation anzupassen, sind Kernelemente agiler Modelle. Bei Scrum heissen diese inkrementellen Schritte *Sprints* und die in einem Sprint zu realisierenden Anforderungen *Sprint Backlog*.



Scrum – ein agiles Vorgehensmodell für die Software-Entwicklung

Wie lassen sich nun agile Ansätze auf das allgemeine Projektmanagement anwenden? Welche Voraussetzungen müssen erfüllt sein? Eine Untersuchung der wichtigsten Prinzipien von Scrum hilft, diese Fragen zu beantworten.

Woher kommt Scrum?

Scrum (engl. für „Gedränge“) ist ein Begriff aus dem Rugby-Sport und beschreibt die enge Formation einer Mannschaft.

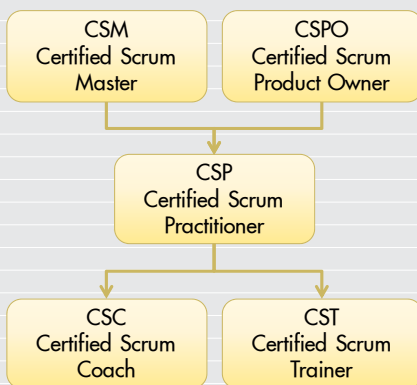
Die Grundlagen von Scrum als Modell kommen aus der Produkte-Entwicklung. Scrum wurde erstmals 1995 von Jeff Sutherland und Ken Schwaber beschrieben.



Scrum – als Team mit vereinten Kräften gemeinsam zum Ziel

Scrum-Zertifizierungen

2003 initiierte Ken Schwaber das Scrum-Zertifizierungsprogramm. Mittlerweile wird es von der Scrum Alliance geführt.



Scrum-Zertifizierungsprogramm

CSM und CSPO sind zweitägige Kurse, die die theoretischen Konzepte von Scrum vermitteln. Bisher haben sich etwa 22'000 Personen als CSM zertifizieren lassen. Nach einem Jahr praktischer Mitarbeit in Scrum-Projekten kann man sich dann als CSP zertifizieren lassen.

Für das CSC-Zertifikat ist der Nachweis umfangreicher Beratungserfahrung in mehreren Scrum-Projekten erforderlich. Der CST schliesslich ist berechtigt, selbst CSM- und CSPO-Kurse durchzuführen.

Scrum – Agilität in der Software-Entwicklung

Software-Entwicklung ist komplex und anspruchsvoll. Sie ist geprägt von einem dynamischen Marktumfeld, unscharfen Anforderungen, hohem Innovationstempo und vielen neuen Technologien. Mit Scrum hat sich in den letzten Jahren ein neues Vorgehensmodell der agilen Software-Entwicklung etabliert, das diesen Herausforderungen Rechnung trägt und zu greifbaren Resultaten führt. Die Ecksteine dieses Vorgehensmodells sind:

- **Sprints.** Das Projekt besteht aus einer Abfolge von kurzen Etappen, sogenannten Sprints, an deren Ende je ein in sich abgeschlossenes und funktionsfähiges Produkt-Inkrement vorliegt. Aus jedem Sprint entsteht für den Auftraggeber ein konkreter und eventuell auch sofort einsetzbarer Nutzen, zum Beispiel die Umsetzung einer neuen Funktion, die anschliessend an Kunden ausgeliefert werden kann. Je nach Charakter des Projekts dauern diese Sprints ein bis vier Wochen.
- **Product Backlog.** Die priorisierten Anforderungen werden in einer Liste (= Product Backlog) verwaltet. Zu Beginn eines Sprints wählt das Projektteam daraus die Elemente, die im Sprint zu realisieren sind. Dies führt zu hoher Identifikation mit den selbst gesteckten Zielen und zu kollektiver Verantwortung für die Ergebnisse.
- **Sprint Backlog.** Das Team organisiert sich und seine Arbeit selbständig, indem es die für den Sprint notwendigen Arbeitspakete und Tätigkeiten (= Sprint Backlog) definiert und diese untereinander aufteilt.
- **Daily Scrum Meeting.** In einer kurzen täglichen Zusammenkunft bringt sich das Projektteam auf den aktuellen Stand. Die Moderation dieses Meetings obliegt dem *Scrum Master*. Er begleitet das Team während des Sprints, blockt Störungen von aussen ab und schafft optimale Arbeitsbedingungen.
- **Sprint Review Meeting.** Am Ende eines Sprints werden die erreichten Ergebnisse im Sprint Review Meeting durch den Auftraggeber sowie interessierte Personen (Stakeholder) geprüft und abgenommen bzw. zurückgewiesen.

Scrum-Rolle	Aufgaben und Verantwortungen
Product Owner (Auftraggeber)	<ul style="list-style-type: none"> ▪ Verwaltet das Product Backlog mit Verantwortung für die Definition und Priorisierung der Anforderungen ▪ Nimmt die Lieferobjekte ab ▪ Verantwortet das Budget
Scrum Master (Projektleiter)	<ul style="list-style-type: none"> ▪ Schafft optimale Arbeitsbedingungen für das Team ▪ Ist verantwortlich für den Scrum-Prozess
Team	<ul style="list-style-type: none"> ▪ Organisiert sich selbst ▪ Wählt das Ziel eines Sprints selbst und ist kollektiv verantwortlich für die Lieferung der Lieferobjekte ▪ Arbeitet möglichst in einem gemeinsamen Büro
Stakeholder	<ul style="list-style-type: none"> ▪ Bringen sich bei der Anforderungsdefinition und in den Sprint Review Meetings ein

Rollen und Aufgaben bei der Software-Entwicklung mit Scrum

Trotz schlankem Vorgehen zeichnet sich Scrum durch einen hohen Formalisierungsgrad aus. Scrum konzentriert sich auf wenige Elemente, dafür werden die Regeln, Rollen und Meetings ebenso wie die Aufgabenpakete sehr detailliert definiert. So ergeben sich nicht nur Erleichterungen bei der täglichen Arbeitsorganisation, sondern auch ein aktuelles Reporting sowie eine saubere Verwaltung von Anforderungen, Änderungen und Releases. Zusammen mit einer fixen Struktur und einem fixen Zeitrahmen für Meetings resultiert eine optimale Ausgangslage für effiziente Entwicklungsarbeit.

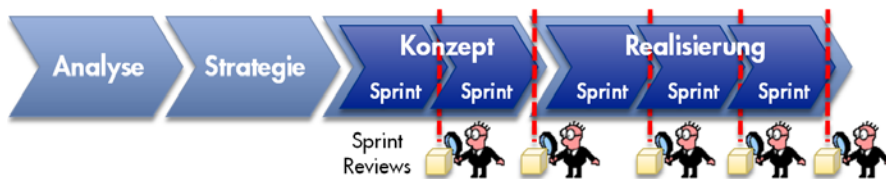
Lehren für den Projektalltag

Die Erfolge von Scrum und weiteren Techniken unter dem Label „Agile Software-Entwicklung“ sind beachtenswert. Doch sind die Rezepte aus der Software-Entwicklung auf allgemeines Projektmanagement übertragbar? Aus unserer Erfahrung lassen sich einige Scrum-Prinzipien durchaus verallgemeinern und in anderen Projektarten anwenden.

▪ Aufteilung von Projektphasen in kurze Etappen (Sprints)

Dank kurzen Sprints von ein bis vier Wochen lassen sich schon früh erste brauchbare Ergebnisse erzielen, die z.B. durch Experten geprüft oder ausgewählte Benutzer in einem Pilot getestet werden können. Die frühzeitige Durchführung von Reviews und Abnahmen von (Teil-)Lieferobjekten ermöglicht es auch, Probleme rechtzeitig zu entdecken und die Schwerpunkte für den nächsten Sprint richtig zu setzen.

Voraussetzungen: Sprints eignen sich nur für längere Projektphasen mit einem hohen Anteil an konzeptionellen, innovativen und gestalterischen Tätigkeiten (z.B. Erstellung von Konzepten oder Spezifikationen, Programmierung von neuen Funktionen oder Benutzeroberflächen).



Die Konzept- und die Realisierungsphasen von ICT-Projekten sind am besten geeignet, um in kurze Etappen (Sprints) aufgeteilt zu werden. Am Ende eines Sprints sollte immer ein Review der jeweiligen Lieferobjekte durchgeführt werden.

▪ Eigenverantwortung und Selbstorganisation des Teams

Indem sich das Team selber organisiert, intensiv zusammenarbeitet und gemeinsam die Ergebnisverantwortung trägt, ist das Engagement der Teammitglieder grösser, als wenn jeder nur für die „Abarbeitung“ seines Aufgabenpakets zuständig ist. Eine kurze tägliche Sitzung sorgt dafür, dass möglichst alle alles wissen. Sämtliche relevanten Informationen wie Teamregeln, Projektplanung, Etappenziele, Aufgabenfortschritt, Probleme usw. werden laufend nachgeführt und an alle kommuniziert.



Voraussetzungen: Die Teammitglieder arbeiten zu mindestens 50% über mehrere Monate im Projekt, idealerweise in einem gemeinsamen Büro. Ein besonderes Risiko sind dominante Teammitglieder, die den Prozess der Selbstorganisation stören.

▪ Intensiver Einbezug des Auftraggebers

Der intensive Einbezug des Auftraggebers (bzw. seines Vertreters) verbessert spürbar die Abstimmung des Projekts mit den Geschäftszielen (Business Alignment). Dadurch, dass der Auftraggeber vor jedem Sprint die zu realisierenden Anforderungen priorisiert, ist die Fokussierung der beschränkten Ressourcen auf die geschäftsrelevanten Anforderungen sichergestellt. Durch die Abnahme am Sprint-Ende geht die Verantwortung für die Lieferobjekte vom Team an den Auftraggeber über, und das Team kann sich dem nächsten Sprint widmen.



Agile und schlanke Methoden

Die Wurzeln der Methoden, die heute als „agil“, „schlank“ oder „leichtgewichtig“ bezeichnet werden, reichen bis in die 60er Jahre zurück und liegen in der Industrieproduktion: Das *Toyota Production System (TPS)* und darauf aufbauend der Ansatz des *Lean Management* beschäftigten sich bereits damals mit den grundlegenden Zielen Qualitätsverbesserung und Kostenkontrolle. Enge Feedback-Schleifen und damit rasche Adaption sowie hohe Transparenz des Projektstatus – Kernelemente agiler Methoden – stammen ebenfalls aus jener Zeit.

In den 90er Jahren wurden diese Konzepte auf die Software-Entwicklung angewandt, und es entstand eine ganze Reihe von neuen Methoden und Vorgehensmodellen. Die wichtigsten Vertreter sind: *Scrum*, *Extreme Programming (XP)*, *Crystal*, *Feature Driven Development (FDD)*, *Test Driven Development (TDD)* und *Adaptive Software Development (ASD)*. Ob *Rational Unified Process (RUP)* ebenfalls zu den agilen Methoden zählt, ist umstritten. Es kann jedenfalls kaum als „leichtgewichtig“ gelten. Erwähnenswert ist ausserdem die Kombination von Scrum und XP. Während Scrum hauptsächlich Fragen der Organisation und des Vorgehens adressiert, fasst XP grundlegende Best Practices und Techniken aus Programmierung und Software-Entwicklung zusammen. Bei Projekten mit einem hohen Anteil an Software-Entwicklung ist daher der gemeinsame Einsatz durchaus sinnvoll.

Die höchste Reputation und wohl auch die grösste Verbreitung unter all diesen Methoden genießt Scrum. Wesentlich zu der hohen Akzeptanz, die Scrum heute genießt, hat die 2003 gegründete Scrum Alliance beigetragen. Die gemeinnützige Organisation widmet sich der Verbreitung von Scrum und bietet ein Zertifizierungssystem an (siehe Spalte linke Seite).



www.scrumalliance.org

Voraussetzungen: Der Auftraggeber muss genügend Zeit für das Projekt aufbringen können und über Budgetverantwortung verfügen. Er muss die Bedürfnisse der Kunden gut kennen und diese bei Bedarf einbeziehen. Anforderungen und Lieferobjekte sollten immer auch von Kunden bzw. Benutzern selber verifiziert werden, da der Auftraggeber diese nur beschränkt repräsentieren kann.

▪ **Änderungen sind normal und keine Sündenfälle („embrace change“)**

Während des Projektverlaufs sind Änderungen an Anforderungen, Rahmenbedingungen oder Konzepten zwar unerwünscht, aber eine Realität. Deshalb sollten Änderungen systematisch in das Projekt einbezogen werden, am besten vor jedem neuen Sprint. Dank der kurzen Sprints und der jeweiligen Neu-Priorisierung der Anforderungen durch den Auftraggeber können Änderungen zeitgerecht und kontrolliert in den nächsten Sprint einfließen. Auf diese Weise kann „agil“ auf Veränderungen reagiert werden, ohne den Projekterfolg zu gefährden.



Voraussetzungen: Während eines Sprints bleiben die Anforderungen „eingefroren“. Eventuelle Änderungen werden auf den nächsten Sprint verschoben. Bei Lieferantenbeziehungen mit starren Vertragsverhältnissen sind Änderungen am Lieferumfang in der Regel sehr mühsam.

Fazit

Die oben beschriebenen agilen Ansätze sind weder Wundermittel noch grundsätzlich neu. Der Einsatz agiler Verfahren kann sogar problematisch sein. Wenn im Projekt die Anforderungen klar definiert sind, wenn nur wenige Änderungen zu erwarten sind und wenn die Realisierung kaum Risiken aufweist, so ist ein klassisches Vorgehen mit starr definierten Phasen besser geeignet.

Agile Verfahren eignen sich hingegen gut bei Projekten, wo die Anforderungen erst im Projektverlauf konkretisiert werden können. Dies ist besonders für Projekte mit einem hohen Anteil an innovativen und gestalterischen Tätigkeiten (Design und Entwicklung) der Fall, wo viele Teammitglieder intensiv über mehrere Monate zusammen arbeiten müssen. Konsequenter und richtig angewendet, führen agile Ansätze dort rascher und sicherer zum Projekterfolg als ein starres Vorgehen mit wenig Handlungsspielraum. Denn im agilen Projekt gilt:^[2]

- Personen und Kommunikation sind wichtiger als Prozesse und Tools.
- Funktionierende Lieferobjekte sind wichtiger als ausführliche Dokumentationen.
- Stetige Zusammenarbeit mit dem Auftraggeber bzw. Kunden ist wichtiger als detaillierte Vertragsverhandlungen.
- Auf Änderungen zu reagieren ist wichtiger, als stur einem Plan zu folgen.

Wie im allgemeinen Projektmanagement gilt aber auch für agile Ansätze:

- Der Kunde muss von Beginn an in das Projekt einbezogen sein.
- Für das Projekt müssen genügend Ressourcen bereit gestellt werden (Zeit und Geld).
- Das Projekt muss gut strukturiert und geplant werden. Für jedes Aufgabenpaket muss der Aufwand geschätzt und der Verantwortliche bestimmt werden.
- Lieferobjekte müssen geprüft und vom Auftraggeber abgenommen werden.

Weiterführende Quellen:

[1] Standish Group International, Chaos Report v 3.0 2004

[2] Abgeleitet aus dem *Agilen Manifest* u.a. von Ken Schwaber und Jeff Sutherland, 2001, <http://agilemanifesto.org>

Über die Autoren



*Manuel
Thiemann*

Manuel Thiemann ist Dipl. Informatik-Ing. ETH und Senior Consultant bei AWK. Er verfügt über mehrjährige Erfahrung als Entwickler, Architekt und Projektleiter in den Bereichen Web-Portale, Content Management Systeme, Datenmanagement und Service Oriented Architecture. Manuel Thiemann ist Certified Scrum Master. manuel.thiemann@awk.ch



*Dr. Richard
Forster*

Dr. Richard Forster ist promovierter Wirtschaftsinformatiker und Consultant bei AWK. Er verfügt über mehrjährige Erfahrung als Entwickler, Consultant und Projektleiter in den Bereichen Information Retrieval, Computerlinguistik, Datenmanagement und Dokumentenmanagement. richard.forster@awk.ch

Über die AWK Group

AWK ist ein führendes, unabhängiges Schweizer Consulting- und Engineering-Unternehmen für Informatik, Telekommunikation und Leitetchnik. An den Standorten Zürich und Bern sind über 100 Mitarbeitende tätig.

AWK Group AG, Leutschenbachstrasse 45,
CH-8050 Zürich, www.awk.ch



Consulting and Engineering