

Agile Softwareentwicklung: Wie IT-Projekte im Facility Management von neuen Erkenntnissen der IT-Welt profitieren könnten

Die IT-Welt begegnet den seit langem bekannten Schwächen beim traditionellen Vorgehen in der Softwareentwicklung mit neuen Methoden. "Agile Softwareentwicklung" hat insbesondere im Umgang mit sich rasch ändernden Anforderungen und kurzen Entwicklungszyklen an Bedeutung gewonnen.

Klassische Einführung eines CAFM-Systems

Bei der Einführung eines CAFM-Systems handelt es sich gemäss [1] um einen "Prozess, der als komplexes Projekt vom Konzept bis zur Implementierung zu organisieren und konsequent zu führen ist". Es wird selbstverständlich davon ausgegangen, dass ein auf dem Markt erhältliches Softwareprodukt ausgewählt, beschafft und implementiert wird.

Für die erfolgreiche Einführung steht in diesem Vorgehensmodell die systematische Projektabwicklung im Vordergrund. Dem Erstellen des Pflichtenhefts – ausgehend von der Analyse – wird grosse Bedeutung beigemessen. Das klassische Auswahlverfahren einer CAFM-Software sieht zudem eine Ausschreibung vor. Die Bewertung der Angebote, die Auswahl der Anbieter, Präsentationen und Testläufe sind weitere Teilschritte der Produktauswahl vor der Implementierungsphase. Das Projekt soll man durch einen neutralen Berater organisieren und konsequent führen lassen, "denn der direkte Weg zum Softwarehersteller kann den Erfolg des gesamten Projektes gefährden" [2].

Darüber, wie man die "richtige" Software für das Facility Management "richtig" auswählt und "richtig" einführt, gibt es genügend Dienstleistungsangebote.

In Anbetracht der Behauptung, dass nur 20-30% der gekauften CAFM-Systeme wirklich produktiv genutzt werden, stellt sich die Frage, ob nicht ein grundsätzliches Überdenken des Vorgehensmodells angebracht wäre.

Softwareprojekt als Betriebsstörung

Die Ausgangs- und Motivationslage der verschiedenen Betroffenen eines Softwareprojekts sind unterschiedlich. Der Auftraggeber, der eine neue Informatiklösung beauftragt, sieht die Dinge anders als die künftigen Anwender.

Die Interessen der Anwender sind unter anderen: reibungslose Arbeitsabläufe, klare Arbeitsverhältnisse, Sicherheit, Stabilität und Komfort. Sie sind normalerweise auf Dauer in ihrer Funktion tätig und haben gelernt, mit einer bestehenden Lösung zu "leben". Ein Softwareprojekt stört ihren gewohnten Arbeitsalltag, ähnlich wie Umbauarbeiten in einem Haus, in dem man wohnt. Die Befürchtungen, dass durch ein Softwareprojekt Mehrbelastungen entstehen und Einführungsprobleme auftreten, sind nicht unbegründet.

Das primäre Interesse der Projektmitarbeiter ist, das Resultat des Projekts unter den gegebenen Rahmenbedingungen zu erreichen. Der Einsatz der Softwareentwickler in einem Projekt ist zeitlich befristet. In der Endphase eines Projekts werden die Entwickler hektisch, schlafen wenig und die Anspannung nimmt zu. Wenn der Fertigstellungstermin näher rückt, helfen die Projektmitarbeiter einander

Projektmitarbeiter einander vermehrt; es gilt das Ziel zu erreichen und das Projekt abzuschliessen.

Die Softwareentwickler sehen ihr Tun – Entwerfen, Schreiben, Testen, Dokumentieren und Debuggen von Software – als kreativen Akt, während aus der Haltung der Anwender klar wird, dass der Betrieb schon vor der "Betriebsstörung", Softwareprojekt, funktioniert hat und ihr Interesse dem reibungslosen Betrieb nach dem Projekt gilt.

Weit mehr als die Hälfte aller IT-Projekte verlaufen nicht reibungslos und viele davon scheitern ganz [3]. Das Scheitern von Softwareprojekten wird thematisiert, aber Publikationen darüber sind kaum zu finden. Kein Wunder, wer möchte schon über Entwicklungsprojekte berichten, die abgebrochen werden mussten, oder über Projekte deren Nutzen in keinem Verhältnis zu den Kosten standen, und/oder deren Terminüberschreitungen ins Unermessliche stiegen?

In einer Untersuchung über die wichtigsten Ursachen für Misserfolge in Softwareprojekten [4], werden erschreckende Zahlen genannt. Allein für das Jahr 1995 haben Staat und Wirtschaft der USA 81 Milliarden US\$ für abgebrochene Softwareprojekte und zusätzliche 59 Milliarden US\$ für überzogene Budgets in Softwareprojekten bezahlt. Die Verluste durch verunglückte Softwareprojekte in Europa dürften eine vergleichbare Grössenordnung erreichen, da die angewandten Verfahren, Methoden und Tools bei der Softwareentwicklung dieselben sind.

Seit dem die so genannte Softwarekrise an der berühmten Garmisch-Konferenz 1968 diagnostiziert wurde, wird weltweit daran gearbeitet, die Vorgehensweise in der Softwareentwicklung zu standardisieren. Software soll – wie ein anderes technisches Produkt – ingenieurmässig entwickelt werden. Dieses Ziel wird verfolgt durch den Einsatz von Methoden und Werkzeugen, durch Projektorganisation und Projektmanagement, durch Ausbildung und permanente Schulung. Trotz enormer Fortschritte im Software Engineering ist die Softwareentwicklung nicht mit industriellen Produktionsmethoden vergleichbar.

Agile Softwareentwicklung

Exponenten im Software Engineering hinterfragen seit einigen Jahren, ob die Schwerpunktsetzung auf immer bessere Methoden und detailliertere Vorgehensmodelle zur Softwareherstellung richtig ist. Eine eigentliche "(Rück-) Besinnung auf Grundwerte in der Softwareentwicklung" [5] ist im Gange.

Im Februar 2001 traf sich in Utah eine Gruppe von Personen, die alle aus dem Umfeld der inkrementellen Softwareentwicklung stammen und bereits jahrelange Erfahrung damit hatten. Während den Diskussionsrunden konnte man sich auf verschiedene grundlegende

Punkte des Softwareentwicklungsprozesses einigen, die für alle ihre Gültigkeit hatten. Diese Punkte wurden im so genannten "Manifesto for Agile Software Development" [6] zusammengefasst.

Wir entdecken bessere Wege zur Entwicklung von Software, indem wir Software entwickeln und anderen bei der Entwicklung helfen. Durch diese Tätigkeiten haben wir gelernt, dass uns:

<i>Menschen und Zusammenarbeit</i>	mehr als Prozesse und Werkzeuge bedeuten,
<i>lauffähige Software</i>	mehr als umfangreiche Dokumentation,
<i>Zusammenarbeit mit Auftraggebern</i>	mehr als Vertragsverhandlungen und
<i>Reagieren auf Änderungen</i>	mehr als das sture Befolgen eines Plans.

Natürlich sind auch die Dinge rechts wichtig, aber im Zweifelsfall schätzen wir die linken höher ein.

An weiteren solchen Treffen wurde der Teilnehmerkreis erweitert und die agile Softwareentwicklung in alle Welt proklamiert. Dazu wurde die "AgileAlliance" [7] ins Leben gerufen.

Eine Gruppe von deutschen Befürwortern agiler Methoden [8] hat die vier Wertaussagen des agilen Manifests in "6+1 Maximen" konkretisiert.

<i>offen für Änderungen</i>	statt Festhalten an starren Vorgaben
<i>mehr ergebnis-orientiert</i>	als prozess-orientiert
<i>eher miteinander reden</i>	als gegeneinander schreiben
<i>mehr Vertrauen</i>	als Kontrolle
<i>eher bottom-up "Best Practices" austauschen und etablieren</i>	als top-down den State-of-the-art diktieren
<i>eher Angemessenheit</i>	als Extremismus

Die letzte Maxime [5] beschreibt die Grundeinstellung in allen anderen Bereichen und relativiert einige Aussagen des populärsten Vertreters agiler Methoden, dem "eXtreme Programming" [9].

Bei allen agilen Verfahren wird dem Erreichen der Ergebnisse mehr Bedeutung beigemessen als dem Weg dorthin. Dokumente werden nur geschrieben, wenn keine bessere Möglichkeit zur direkten Kommunikation und Kollaboration gefunden wird. Wo Dokumentation notwendig ist, um den sichern Betrieb zu garantieren, wird sie häufig erst nachträglich erstellt, wenn sich das System bereits stabilisiert hat. Die Idee vom einheitlichen, wiederholbaren Entwicklungsprozess wird ebenso aufgegeben, wie die Vorstellung, man könne Projekte detailliert durchplanen, wenn man sich nur ausreichend Mühe gibt. Wenn der Entwicklungsprozess weniger exakt durchgeplant ist, dann wird die Hoffnung, man könne ein Projekt durch Kontrolle und Weisungen steuern, zur Illusion.

Agilität in CAFM-Projekten

Das klassische Vorgehensmodell zur Einführung eines CAFM-Systems steht im krassen Gegensatz zum Wertesystem der agilen Entwicklung. Es geht an dieser Stelle nicht darum, die heissen Diskussionen der IT-Welt pro oder kontra Agilität im Facility Management zu wiederholen, sondern um die Frage:

Wie können IT-Projekte im Facility Management von den neuen Erkenntnissen der IT-Welt profitieren?

Durch Zusammenarbeit mit dem Auftraggeber:

Die vollständige Spezifikation eines CAFM-Systems ist nicht nur extrem schwierig zu erstellen, sondern auch sehr teuer. Im laufenden Projekt ändern die Anforderungen fast immer. Statt also Zeit und Geld in den Spezifikationsprozess zu verschwenden, könnten die wichtigsten Requirements mit minimalem Aufwand in direkter Zusammenarbeit mit dem Auftraggeber erarbeitet werden.

Durch iterative Entwicklung:

Das Customizing eines CAFM-Systems ist hervorragend geeignet für ein inkrementelles und interaktives Vorgehen. Die Anwender und der Auftraggeber sehen das Endergebnis konkret wachsen; die Entwickler arbeiten nicht mehr so lange "nur mit Papier" – sie erhalten schnell Feedback. Bei jeder Iteration wird die Möglichkeit eröffnet, auf neue oder geänderte Anforderungen einzugehen. Wer bei der Systementwicklung stur einen Plan verfolgt, erhält das CAFM-System, das er geplant hat, nicht das System, das er braucht!

Durch Menschen und Zusammenarbeit:

Ein hoch motiviertes, selbstständiges und kompetentes Projektteam kann ein System weitgehend ohne definierten Prozess und ohne Methoden entwickeln; es ist aber völlig unmöglich, ein CAFM-System ohne qualifiziertes Team einzuführen. Die Teammitglieder müssen das Vertrauen haben, dass sie aufgrund ihrer Erfahrungen situationsgerecht richtig entscheiden können, wie die Projektziele effizient erreicht werden. Für agile Verfahren braucht es aber nicht nur mutige, mitdenkende Mitarbeiter, sondern vor allem auch charismatische Führungspersönlichkeiten und eine entsprechende Unternehmenskultur.

Durch lauffähige Software:

Kein (Facility-) Manager wird je in Frage stellen, dass er schon immer "nur" ein lauffähiges System gefordert hätte. Das Rezept ist einfach, es heisst "Einfachheit" und es erfordert Mut von allen Beteiligten (Auftraggeber, Projektteam und Anwender). Die begleitende Frage heisst stets: "Welche ist die einfachste Lösung, die möglicherweise funktioniert?" Denn es ist besser, heute etwas Einfaches zu implementieren, als viel Aufwand in etwas zu investieren, das vielleicht niemals eingesetzt wird.

Unabhängig davon, ob man der Meinung ist, dass agile Verfahren weniger eine technische als vielmehr eine soziologische Errungenschaft sei, gilt es auch im Facility Management, situationsgerecht einen Mittelweg zwischen kreativer Freiheit und Regelungen in CAFM-Projekten zu finden.

Wer zudem der gemeinsamen Arbeit und dem gemeinsamen Erreichen von Zielen zum Erfolg und zur Freude aller Beteiligten einen hohen Wert beimisst, müsste

künftig die Zusammensetzung der Projektteams höher gewichten, als die Auswahl (resp. Ausschreibung und Kauf) der Software.

Autorenportrait

Andreas Duppenthaler ist Mitglied der Geschäftsleitung und Mitinhaber der Byron Informatik AG. Diese Firma in Basel wurde 1992 gegründet und ist in der Softwareentwicklung tätig. Sie hat die Standardsoftware Byron/BIS - eine objektorientierte Entwicklungsplattform zur Konfiguration von Facility Management Systemen - entwickelt.

A. Duppenthaler ist dipl. Mathematiker-Ingenieur ETH, Fachrichtung Informatik. Als Mediator befasst er sich mit dem Thema "Informatikprojekte und Mediation".

Am Paul Scherrer Institut war er wissenschaftlicher Mitarbeiter und leitete ein Informatikprojekt, das zwei Zielsetzungen verfolgte: die Entwicklung eines integrierten Softwaresystems als Planungswerkzeug für die Haustechnikbranche und den Know-how-Transfer vom Forschungsinstitut in die Privatwirtschaft.

Literatur

- [1] Deutscher Verband für Facility Management: Einführung eines CAFM-Systems, GEFMA Richtlinie 420
- [2] Schöne, L.B: Nicht bloss ein Kostenfaktor... – Consulting bei der Einführung von CAFM: Facility Management, Heft 02/00; Gütersloh: Bertelsmann Fachzeitschriften 2000.
- [3] Duppenthaler, A.: Informatikprojekte und Mediation. Diplomarbeit im Nachdiplomkurs Mediation in Wirtschaft, Umwelt und Verwaltung; Fachhochschule Aargau, Nov. 2001
- [4] May, Lorin J.: Major Causes of Software Project Failures, CROSSTALK The Journal of Defense Software Engineering; July 1998
- [5] Hruschka, P.: Agility – (Rück-) Besinnung auf Grundwerte in der Softwareentwicklung. Informatik Spektrum, Dez. 2003
- [6] Agiles Manifest: <http://www.agilemanifesto.org>
- [7] AgileAlliance: <http://www.agilealliance.com>
- [8] Deutsche Befürworter agiler Methoden: <http://www.b-agile.de>
- [9] Beck, K.: Extreme Programmierung – Das Manifest. Addison Wesley Longman Verlag 2000