Agil, aber Stabil

Michael Mahlberg
Unabhängiger Berater
Mitglied der „Consulting Guild“
mm@consulting-guild.de

Neusser Strasse 41
50670 Köln

Agile Prozesse scheinen sich entsprechend der aktuellen Diskussion hauptsächlich für in sich geschlossene Softwareprojekte zu eignen. Bei umfangreicheren, interdisziplinären Projekten scheinen die harten Termine der Zulieferer, das komplexe Projektumfeld und die langwierigen Abstimmungsprozesse gegen agile Methoden zu sprechen. Die wichtigsten Eckpunkte des „Agilen Manifests“ [Be01] gelten aber auch für solche Projekte. Dieser Artikel beschreibt die Einbettung agiler Teilprozesse in ein Gesamtprojekt mit starren Strukturen und fixen Rahmenterminen (Unternehmensweite Migration zwischen zwei Konfigurationsmanagementsystemen) und die Auswirkungen auf das Gesamtprojekt.

Das Projektumfeld

Die hier betrachteten Prozesse stammen aus einem Projekt mit der Zielsetzung das Konfigurationsmanagementsystem zweier Entwicklungshäuser zu harmonisieren, eingebettet in die Harmonisierung der gesamten Softwarelandschaft. Wie bei fast allen Projekten in der realen Welt war es in diesem Umfeld nicht möglich, auf der grünen Wiese anzufangen. Vielmehr galt es, die konkreten Anforderungen der unterschiedlichen bereits produktiven Softwareentwicklungsprojekte (SE-Projekte) zu berücksichtigen. Aus Sicht der Gesamtprojektleitung waren zudem die unterschiedlichsten Teilprojekte zu koordinieren. Die Aktivitäten im Bereich Konfigurationsmanagement hatten sich dem allgemeinen Zeitplan unterzuordnen. Da alle bisher im Konzern eingesetzten KM-Systeme ihre individuellen Eigenheiten haben und die Entwicklungsprozesse (derzeit noch) stark damit verzahnt sind, war bereits zu einem frühen Zeitpunkt klar, dass es notwendig werden würde, eigene Erweiterungen zu erstellen, um zu einem konzernweit einheitlichen KM-System zu kommen. Innerhalb des Subprojektes „Einführung des ausgewählten KM-Systems“ gab es demzufolge folgende Teilbereiche zu bearbeiten:

- Aufnahme der Anforderungen der betroffenen SE-Projekte
- Ausarbeitung neuer Konzepte
- Erstellung der Erweiterungen
- Umstellung der betroffenen SE-Projekte

Um den unterschiedlichen SE-Projekttypen (von Smalltalk über Java bis hin zu COBOL sowie Mischungen aus diesen) gerecht zu werden, wurden die beiden SE-Projekte mit dem größten Risikopotential und der größten Sichtbarkeit als Prototypen zur Aufnahme in das zukünftige KM-System ausgewählt. Deren Migration wurde exemplarisch umgesetzt. Hierbei wurden für die beiden Pilotsysteme unterschiedliche Vorgehensweisen gewählt, die im weiteren geschildert werden und deren Auswirkungen im Hinblick auf die Agilität signifikant sind.

Initiale Planungsansätze

Zu Beginn des KM-Projektes wurde versucht, die Aktivitäten der vier Teilbereiche (Anforderungen, Konzepte, Erweiterungen erstellen und Umstellung der SE-Projekte) streng linear über die Gesamtlaufzeit verteilt umzusetzen, also nach klassischem Wasserfallmodell vorzugehen. Für die beiden Pilotprojekte bedeutete dies, dass im Rahmen der prototypischen Migration die Anforderungen an das Zielsystem ermittelt werden sollten. Dabei wurde für das eine der beiden SE-Projekte (nennen wir es V) eine enge Zusammenarbeit mit den Entwicklern des Projektes gewählt, während bei dem zweiten SE-Projekt (Nennen wir es T) die Entwicklungsabteilung durch einen Stellvertreter aus einem anderen Bereich als Mittelsmann repräsentiert wurde.

Vergleichen wir diese Situation mit den Forderungen von Kent Beck und anderen [Be00] so haben wir im Projekt V tatsächlich einen Kunden vor Ort (On Site Customer), während wir im Projekt T nur einen (unbeteiligten) Stellvertreter des Kunden finden.

Bereits in der ersten Phase des Migrationsprojektes wurde deutlich, dass zumindest für die Anforderungen des Projektes T die Gefahr der Analyse-Lähmung des KM-Teams (vgl. Analysis Paralysis, [Br98]) bestand. Obwohl das KM-Team nicht nur „im eigenen Saft schmorte“, sondern durchaus auch einen Kundenvertreter zur Seite hatte, der die Interessen des Kunden repräsentieren sollte, wurde viel Zeit für hypothetische Betrachtungen der Kundenwünsche verwendet. Es fiel dem KM-Team schwer, die tatsächlich notwendigen Anforderungen zu isolieren.

Im Projekt V, in dem ein direkter Kontakt zum Kunden bestand, war zwar die Identifikation der Anforderungen einfacher, jedoch stellte sich hier ein anderer nachteiliger Effekt ein: Je mehr Verständnis die Entwickler für die Anforderungen und Möglichkeiten des KM-Teams gewannen, desto häufiger änderten sich die Anforderungen des Projektes V. Da das Projekt KM-Migration – wie geschildert – nicht autark, sondern in einem komplexen Projektumfeld mit festen extern bestimmten Terminen agiert, gefährdete diese „moving target“ Situation auf Dauer den Projekterfolg.

Schritte in die Agilität

Die wichtigsten Maßnahmen, um aus dieser Situation heraus zu einem besser plan- und steuerbaren Prozess zu kommen, scheinen auf den ersten Blick sehr überschaubar. Sie dienen vordringlich dazu, den Prozess der Planung als Bestandteil des Entwicklungsprozesses zu etablieren und das Entwicklungsteam stärker einzubinden.

Im Umfeld agiler Prozesse wird häufig das Eisenhower-Zitat „In der Vorbereitung auf den Kampf habe ich festgestellt, dass Pläne nahezu nutzlos sind, Planung aber absolut unerlässlich ist“ erwähnt (vgl. [BF01]). In diesem Sinne ist es Ziel der Maßnahmen, das Projektteam in die Lage zu versetzen, die Auswirkungen von Planänderungen frühzeitig selbst beurteilen zu können.

Im Projekt T wurde der Mittelsmann durch einen direkten Kontakt zum Entwicklungsteam ersetzt. Anfängliche Befürchtungen, hierdurch mehr Koordinationsaufwand zu generieren, bewahrheiteten sich glücklicherweise nicht. Statt dessen führte dieser Schritt zu einer Übernahme von mehr Verantwortung und eigener Entwicklungstätigkeit durch das Team des Projektes T, was zusammen mit den bereits im Projekt V gesammelten Erfahrungen zu einem drastischen Anstieg der Produktivität führte.

Da jetzt für beide SE-Projekte eine hohe Interaktion mit den Entwicklungsteams bestand, musste verstärkt darauf geachtet werden, das Problem des beweglichen Ziels – wie im Projekt V – unter Kontrolle zu halten. Hier waren deutlich umfangreichere Maßnahmen notwendig, um den gewünschten Effekt zu erzielen. Die Entwicklungsarbeiten innerhalb des KM-Projektes mussten von einem „Entwurf im Voraus“ (vgl. Up-Front Design, [Be00], [Br98]) umschwenken auf einen iterativen Entwicklungsprozess, bei dem sich die konkreten Anforderungen erst im Laufe der Projektarbeit ergeben würden.

Anders als vielfach von den Verfechtern der agilen Prozesse postuliert, wurde diese Änderung des Prozesses von den Entwicklern keineswegs bereitwillig angenommen. Es zeigt sich, dass Projektsteuerungsaufwand nur langsam auf Entwickler verlagert werden kann. Somit wäre es, zumindest in diesem konkreten Projekt, ohne umfangreiches Projektmanagement mit besonderer Betonung des Zieles „Flexibilität“ nicht möglich gewesen, agile Vorgehensweisen zu etablieren.

Konkrete Maßnahmen

Als erstes musste ein Medium geschaffen werden, welches es dem Team ermöglichte in kurzen Iterationen zu planen, ohne dabei das Gesamtziel und die mit den anderen Projekten abzustimmenden Termine aus den Augen zu verlieren. Nachdem erste Versuche auf Basis von Karteikarten (wie von XP-Verfechtern empfohlen) fehlschlugen, wurde als primäres Medium eine ToDo Liste auf Wochenbasis mit einem Fokus von drei Monaten eingeführt. Mit diesem Mittel wurden bereits deutliche Planungsverbesserungen gegenüber den vorherigen Planungen im reinen 3-Monats-Raster erreicht. Erwartungsgemäß wurde eine Neuplanung in ungefähr dreiwöchigem Rhythmus notwendig, um den sich schnell ändernden Anforderungen aus den Entwicklungsprojekten gerecht zu werden.

Im weiteren Projektverlauf wurden verschiedene Tools (wie z.B. Excel) zur Projektplanung eingesetzt, ohne dass eine Verbesserung gegenüber den ToDo-Listen erreicht wurde. Aufgrund der zunehmenden Notwendigkeit der Koordination mit Fremdterminen sowie durch den zunehmenden Umfang der ToDos wurde die Projektplanung letztendlich auf MS-Project umgestellt. Dennoch blieben die ToDo-Listen das primäre Kommunikationsmittel zwischen Planung und Entwicklung sowie die Grundlage der Tagesplanung der Entwickler.

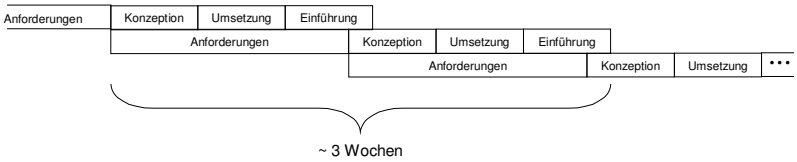
Architekturfragen

Auch bei einer Anwendung, die – wie die hier beschriebene – „nur“ aus Aufsätzen auf ein existierendes System besteht, und deren Implementierung sich auf viele unabhängige Subsysteme – die noch dazu in den unterschiedlichsten Sprachen geschrieben sind – verteilt, gibt es Architekturen auf den unterschiedlichsten Ebenen.

Während hierbei die Architektur der gesamten Einbettung des KM-Systems im Rahmen der Entwicklung organisch wuchs (vgl. [Co04]), waren grundlegende Architekturfragen im Bereich der Mikroarchitektur zu klären. Diese wurden stark von der Forderung nach Agilität beeinflusst. Nach ca. 2/3 der bisherigen Laufzeit wurde ein weiteres Projektmitglied integriert, dessen Kenntnisstand über das Konfigurationsmanagementsystem noch gering war. Seine Hauptaufgabe innerhalb des Teams lag anfänglich in der Erstellung von Elementen, die nicht direkt mit dem Kern des Systems interagieren müssen. Während es verhältnismäßig einfach war, die Gesamtarchitektur der Erweiterungen zu vermitteln, wurde im Bereich der unabhängigeren Erweiterungen besonders deutlich, dass vorschnelle Architekturentscheidungen zu einer massiven Einschränkung der Reaktionsfähigkeit führen. Insbesondere ausgefeilte Optimierungen, die auf den aktuellen technischen Rahmenbedingungen basierten, mussten häufig aufgrund von Änderungen der Umgebung angepasst werden. Die Eigenentwicklungen, die das KM-System erweitern, wurden erst nach mehreren Iterationen so flexibel, dass sie von konzeptionellen Änderungen des Gesamtsystems nur noch wenig beeinflusst werden. Die konkrete Mikroarchitektur stellt inzwischen die Änderbarkeit in den Vordergrund, während Aspekte wie Vollständigkeit und Performanceoptimierung auf diejenigen Iterationen verlagert werden, in denen sie benötigt werden.

Der aktuelle Prozess

Heute, nachdem die ersten SE-Projekte bereits integriert sind, lässt sich der Prozess der ursprünglich angedacht war, zwar noch wiedererkennen, aber er ist stark iterativ geworden. Die Schritte „Aufnahme der Anforderungen“ => „Konzeption“ => „Erstellung der Erweiterungen“ => „Umstellung der betroffenen SE-Projekte“ erfolgen jetzt in Zyklen von drei Wochen. Dabei findet der Schritt „Aufnahme der Anforderungen“ für Iteration n in der Iteration n-1 statt. Welche der Anforderungen in der aktuellen Iteration umgesetzt werden, wird im Rahmen der Konzeption entschieden, so dass sich heute folgendes Prozessbild ergibt:



Die dreiwöchige Iterationsdauer verträgt sich allerdings nur eingeschränkt mit Forderungen nach langfristigen Terminvorgaben für die beteiligten Projekte und Bereiche. Insbesondere Bereiche, die Hardware zur Verfügung stellen, können schwer in Zyklen von drei Wochen planen. Um diesen Widerspruch aufzulösen, werden die mit anderen Abteilungen abgestimmten Termine als Eingangsgröße für die Anforderungsphase betrachtet. So können übergreifende Termine mit deutlich höherem Vorlauf abgestimmt werden.

Fazit

Die Einführung agiler Prozesse im Kleinen ist keineswegs unmöglich, aber auch kein Selbstläufer. Insbesondere Methoden aus dem XP Umfeld sind nur mit einem Team einführbar, das über eine sehr hohe Selbstdisziplin und Eigenmotivation verfügt (vgl. [GFN02]). Während das Management die Argumentation für agile Vorgehensweisen sehr positiv aufnimmt und unterstützt, haben gerade Entwickler, die lange mit Wasserfallbasierten Prozessen leben, Schwierigkeiten, sich auf diese Vorgehensweise einzulassen. Damit fällt ein großer Anteil der Verantwortung für die Agilität auf die Projektleitung, die auch viel Überzeugungsarbeit bei den Entwicklern leisten muss. Auch in diesem Projekt wurde deutlich, dass es nicht ausreicht, iterativ zu entwickeln, um einen agilen Prozess zu etablieren, sondern Architektur und Entwicklerkultur mindestens ebenso große Rollen spielen (vgl. [Co04]). Die Qualität und Stabilität der einzelnen Produkte in diesem Projekt sind insgesamt deutlich gestiegen, da mögliche Fehler bereits sehr viel früher erkannt und beseitigt wurden. Dies zeigt sich in den jetzt stabilen Terminen und der Nutzungsdauer bereits ausgelieferter Komponenten. Ähnliche Beobachtungen finden sich in [La04]. Der Qualitätszuwachs, die gewonnene Flexibilität für neue Anforderungen sowie die deutlich gestiegene Akzeptanz bei den Kunden (in diesem Fall bei den Entwicklungsprojekten) sprechen zudem für agile Prozesse.

Literaturverzeichnis

[Be00] Beck, Kent; eXtreme Programming Explained; Addison Wesley; 2000

[Be01] Beck, Kent et. al.; URL www.agilemanifesto.org

[BF01] Beck, Kent; Fowler, Martin; Planning Extreme Programming; Addison Wesley; 2001

[Br98] Brown, William H. et. al.; AntiPatterns, John Wiley & Sons; 1998

[Co04] Coldewey, Jens; Kolumne: Agile Entwicklung: Die "Top Ten" der agilen Missverständnisse; in ObjektSpektrum 3/2004,

[GFN02] Grütter, Georg; Ferber, Stefan; Neumann, Michael; Der persönliche Softwareprozess: Konzepte, Erfahrungen und Synergien mit XP; in ObjektSpektrum 06/2002

[La04] Larman, Craig; Agile & Iterative Development; Addison Wesley; 2004