Inhaltsverzeichnis

1 Einleitung

Wir betrachten sechs Projektmanagement Methoden mit dem Hauptaugenmerk auf Ablauf und Charakteristik. Diese werden am Ende auf der Basis der von uns gewählten Kriterien verglichen und bewertet.

Folgende Kriterien werden von uns verwendet:[1] [2]

Projektphase: Phasen-spezifisch oder ebenfalls für das komplette Projekt geeignet?

Die Methode sollte sich für alle Phasen der Entwicklung anwenden lassen.

Projektgröße: Für welche Teamgröße und Projektdauer ist es geeignet?

Unwichtig für uns da wir ein kleines Team mit einem kleinen Projekt sind.

Grad der Planung: Darf das Datum bzw. Budget fest oder offen sein?

Wir haben kein Budget. Eine zu feste Strukturierung mit feste Daten sowie eine vollständig fehlende Ordnung ist nicht gewünscht, weshalb wir nach einem Mittelmaß suchen.

Änderungen während Iterationen: Dürfen Änderungen während einer Iteration durchgeführt werden oder nur zwischen diesen? (z.B. Aufgaben-Zuteilung; Aufgaben-Aufwandseinschätzung, Neue Aufgaben in die Iteration aufnehmen, Priorität von Aufgaben)

Wir wollen flexibel in der Ausführung der Iterataionen sein.

Flexibilität: Wie gut/flexibel lässt sich die Methode auf die eigenen Umstände anpassen? Lässt sich die Methodik schrittweise oder in Teilen einführen? (z.B. Länge der Iteration, Dauer der Meetings anpassen)

Es soll alles so flexibel wie möglich sein.

Grad der Eigenverantwortung: Wie hoch ist sie und wer trägt sie?

Alle Mitglieder sollen gleichermaßen verantwortlich sein.

Einarbeitungsaufwand: Wie schwer/aufwendig ist es, die Methodik einzuführen?

Da wir für dieses Projekt eine Managementmethode sofort anwenden, sollte diese so wenig Hürden haben wie möglich.

Einprojekt- oder Mehrprojektumgebungen: Ermöglicht es das Managen mehrerer Projekte parallel? (z.B. eine Abteilung → zwei Projekte, Teammitglieder bearbeiten Aufgaben aus beiden Projekten)

Für uns unwichtig, da wir nur an einem Projekt arbeiten.

[1] http://www.gpm-infocenter.de/PMMethoden/EinfuehrungMethodenauswahl
[2] http://www.heise.de/developer/artikel/Kriterien-fuer-eine-Entscheidung-fuer-Scrum-oder-Kanban-1071172.html

2 Beschreibung der Projektmanagementmethodiken

2.1 Kanban

Der Begriff Kanban kommt aus dem Japanischem und bedeutet in etwa „Signalkarte“. Dieses Projektmanagement-Framework beinhaltet das „Pull-Prinzip“, d.h. ein Arbeiter erteilt Anfragen an andere (z.B. Nachschub anfordern) statt einer Person, die über dem ganzen steht und das Projekt überwacht. Dies soll dem Produktionsfluss dienen.
Kanban legt einem wenige Vorlagen auf, weshalb es flexibel und anpassbar ist, da der Schwerpunkt auf einem optimalen Fluss (Flow) liegen soll.

2.1.1 Kanban Ablauf

Aufgaben, Features, User-Stories und andere Formen von Anforderungen werden als Tickets aufgefasst. Jedes Ticket (Work-Item) durchläuft verschiedene Prozessschritte bis diese fertig ist; dabei wird die benötigte Zeit gemessen. Pro Schritt dürfen sich nur eine maximale Anzahl an Tickets gleichzeitig befinden. „Staut“ es sich, so zeigt es, dass der Flow nicht optimal ist. Es müssen dann für den entsprechenden Abschnitt mehr Ressourcen hinzugefügt oder die Prozessschritte überarbeitet werden.

All das soll als selbstregulierendes System den Produktionsfluss optimal halten.

2.1.2 Kanban-Charakteristik

Kaizen:

Kaizen ist aus dem Japanischen (Kai = Veränderung, Zen = zum Besseren) und repräsentiert den KVP-Anteil (Kontinuierlicher Verbesserungsprozess) von Kanban.[3] Dieser beinhaltet u.a. auch die Treffen und Operations Reviews.

Treffen:

Diese finden täglich und in der Regel morgens statt. Dabei wird mittels des Kanban-Boards vorgetragen, wie weit die einzelnen Aufgaben seit dem letzten Meeting vorangeschritten sind und wo Probleme liegen, sofern vorhanden. Meetings sollten nicht länger als 15 Minuten dauern; bei größeren Diskussionen müssen diese zu einem anderen Zeitpunkt verschoben werden.

Operations Reviews:

Eine Form des Meetings, die im Gegensatz zu den täglichen Treffen unregelmäßig stattfinden. Sie dient der stetigen Verbesserung indem alte Daten betrachtet werden (z.B. Fehlerate, Durchlaufzeiten, Anzahl geblockter Tasks) und daraus entschieden wird, was zu Verbessern ist. Bei diesem Meeting treffen sich alle Mitglieder der Organisation. Operations Reviews sind nicht mit Retrospektiven aus anderen agilen Methoden zu verwechseln, die subjektiv auf die letzte Iteration(en) zurückblicken. Bei Kanban geht es um Zahlen und Fakten, die als Indikator gelten statt der Eindrücke der Mitarbeiter.

Kanban-Board:

Ein Kanban-Board ist eine simple Tafel, auf der die Prozessschritte als Spalten dargestellt werden. Die Aufgaben (Tasks) werden als Magnete, Post-its oder andere geeigneten Gegenstände spaltenweise verschoben. Dabei ist zu beachten, dass nur die erlaubte Anzahl an Tasks pro Spalte

[3] https://de.wikipedia.org/wiki/Kaizen

enthalten sein dürfen. Die Grenze wird üblicherweise oberhalb der Spalte eingetragen, so dass man insgesamt auf einem Blick den Status der Prozesse sieht, Bottlenecks erkennt und wie gut der Produktionsfluss "fließt".

2.1.3 Zusammenfassung

Projektphase:

Kanban lässt sich für alle Arten von Aufgaben anwenden.

Projektgröße:

Kanban kann für nur ein einzelnes Team verwendet werden bis hin das es das ganze Portfolio (mehrere Wertschöpfungsketten) eines Unternehmens abdeckt. Dementsprechend ist es auch für langfristigen Einsatz geeignet, es kann jedoch auch für ein einzelnes Projekt angewendet werden.

Grad der Planung:

Kanban erlaubt die Prioritätensetzung von Tickets. Es gibt Ticketklassen, Tickets die

Änderungen während Iterationen:

Neue Anforderungen können zu jedem Zeitpunkt an das Team gegeben werden, falls Kapazitäten frei sind.

Flexibilität:

Iterationen haben keine feste Länge. Das ist gut für Einsatzgebiete, in denen man mit vielen Unterbrechungen bzw. Notfällen rechnen muss (z.B. Wartung). Tägliches Treffen zur gleichen Uhrzeit mit fixer Länge. Andere Besprechungen sind flexibel.

Grad der Eigenverantwortung:

Hohe Eigenverantwortung wegen dem Pull-Prinzip, Mitglieder und Teams holen sich ihre Aufgaben.

Einarbeitungsaufwand:

Kanban setzt weder viel Vorwissen voraus, noch braucht man spezielles Equipment. Ein Whiteboard und Kärtchen reichen bereits aus. Jedoch kann man auch fertige Software dafür einsetzten was mit finanziellem Aufwand und Einarbeitung ins Programm verbunden ist.

Einprojekt- oder Mehrprojektumgebungen:

Es ist möglich an mehreren Projekten gleichzeitig zu arbeiten. Die Tickets können mit verschiedenen Farben hervorgehoben werden, um diese dann entsprechend zuzuordnen bzw. man kann pro Projekt eigene Zeilen/Zeilengruppen auf dem Board bilden.

2.2 Scrum

Scrum ist eine Projektmanagementmethodik zur agilen Softwareentwicklung. Es zeichnet sich besonders dadurch aus, dass es nur wenige Regeln kennt und daher einfach zu erlernen ist.

2.2.1 Projektmitglieder

Personen die direkt am Prozess beteiligt sind können in drei Rollen eingeteilt werden:

- Product Owner: Stellt die fachlichen Anforderungen und priorisiert diese
- Scrum: Managt den Prozess und beseitigt gegebenenfalls Hindernisse
- Team: Entwickelt das Produkt

Als Beobachter und Berater gibt es zudem noch die sogenannten Stakeholder.

2.2.2 Ablauf

- Anforderungen werden in den Product Backlog eingepflegt, priorisiert und ggf. erweitert
- Monatlich wird ein Arbeitspaket aus dem Product Backlog entnommen und umgesetzt (inkl. Test und Dokumentation)
- Das Arbeitspaket (Increment) wird während einer Iteration (Sprint) nicht modifiziert
- Das Arbeitspaket wird in kleinere Arbeitspakete (Tasks) eingeteilt, einem Bearbeiter zugeteilt, bekommt einen Restaufwand der täglich aktualisiert wird und wird in den Sprint Backlog geschoben
- Während des Sprints darf der Bearbeiter keinen Störungen ausgesetzt sein
- Täglich muss ein 15 Minütiges Daily Meeting abgehalten, um herauszufinden was die anderen Teammitglieder bearbeiten, was man selbst als nächstes bearbeiten muss und wo es Probleme gibt
- Am Ende eines Sprints wird die voll funktionsfähige Funktionalität dem Product Owner und den Stakeholdern live am System gezeigt
- Entsprechende Feedbacks fließen in den nächsten Sprint und das Sprint Planing Meeting mit ein
- Eine neue Iteration beginnt

2.3 Feature-Driven-Development (FDD)

FDD ist eine in der Praxis entstandene Methodik, die versucht, agile Prinzipien in klassischen Unternehmenskulturen zu verwirklichen. Sie ist insbesondere für größere Teams geeignet.

2.3.1 Prozess

Der FDD-Prozess läst sich in 2 Phasen aufteilen: Die erste Phase beinhaltet die Erstellung eines Modells, einer Feature-Liste und eines Plans (geplante Fertigstellung der Features).
Die zweite Phase ist die iterative, agile Phase, in der in einem maximal 2-wöchigem Cyclus Features designt und implementiert werden. Am Ende einer Iteration muss etwas stehen, was die Hände des veranwortlichen Teams verlassen kann.

2.3.2 Teamorganisation

Die Rolle des klassischen Team-Managers wird in FDD beibehalten. Der Manager weist jedem Feature Teammitglieder zu und bildet so sogenannte "Feature-Teams". Solch ein Team besteht nur temporär solange, bis das Feature fertiggestellt wurde. Ein Teammitglied kann mehreren Feature-Teams zur gleichen Zeit angehören.

2.3.3 Zusammenfassung

Projektphase:

FDD beinhaltet keine Prozessschritte für die allgemeine grobe Spezifikation.

Projektgröße:

FDD skaliert problemlos auch mit einem sehr großen Team inklusive zusätzlich vorhandenen Spezialisten oder Spezial-Teams.

Grad der Planung:

In der Planungsphase werden die Fertigstellungstermine der Features festgelegt, es sind nur geringe Änderungen am Plan vorgesehen. Verspätungen weißsen auf Probleme hin.

Änderungen während Iteration:

Änderungen sollten im Rahmen des jeweils dafür vorgesehenen Prozess-Schrittes ausgeführt werden, nicht außerhalb.

Flexibilität:

Der FDD-Prozess ist relativ spezifisch. Für jeden Prozessschritt gibt es eine Empfehlung, wie viel Zeit für diesen Schritt eingeplant werden sollte.

Grad der Eigenverantwortung:

Die Eigenverantwortung der Enwtickler ist abhängig vom Managment-Stil des Team-Managers. Dies kann je nach Stil zu geringer Eigenveranwortung führen, wenn der Manager auf einer direkten Leitung besteht. In jedem Fall ist der Manager für sein Team verantwortlich.

Einarbeitungsaufwand:

FDD ist sehr einfach auf klassische Unternehmensstrukturen abbildbar und damit in einem Unternehmen sehr einfach und konfliktfrei einführbar. In einer Struktur, die einem Unternehmen nicht ähnelt, ist jedoch viel Aufwand erforderlich.

Einprojekt- oder Mehrprojektumgebungen:

FDD ist besonders geeignet für hierarchisch organisierten Projektstrukturen, in denen mehrere Unterprojekte auf das Erfüllen einer gemeinssamen Aufgabe hinarbeiten. Diese hierarchische Strukturen können relativ frei gestaltet werden.

Die jeweiligen Projektleiter können sich dabei einigen, wie die vorhandenen Mitarbeiter für Features eingeteilt werden sollen.

2.4 Extreme Programming (XP)

Ein bekannter agiler Softwareprozess ist Extrem Progamming. Dabei werden die sich während der Entwicklung ändernden Kundenwünsche Sets berücksichtigt. Der Softwareprozess wird immer wieder in kurzen Zyklen durchlaufen. Nur die im Iterationsschritt festgelegten Anforderungen werden implementiert. Diese Methode beruht auf der Erfahrung, das der Kunde alle vollständigen Anforderungen seiner Software zu Projektbeginn noch nicht genau kennt.

2.4.1 Projektphase

Der Kunde hat den gleichen Projeckstand wie der Entwickler. Jede Komponente besitzt einen Modultest, um Fehler frühstmöglichst zu erkennen und zu beseitigen. Auf einen ehrlichen Umgang mit dem Kunden wird Wert gelegt, um Fehler mit Mitarbeitern und Kunden zu minimeren.

Die Projektphase gliedert sich in folgende Einheiten, die aufeinander aufbauen:

- Sekundentakt: Durch Pair-Programming wird die gegenseitige Kontrolle erhöht, und Fehler vermieden.
- Minutentakt: UNIT-Tests werden vor dem Code geschrieben, so das der Code ständig überprüft wird
- Stundentakt: Entwickelte Komponenten werden sofort im Gesamtsystem integriert, und somit Fehler schneller gefunden.
- Tagestakt: tägliches Treffen des Entwicklungsteams um den Projektfortschritt zu reflektieren, und Fehlentwicklungen zu vermeiden.
- Wochentakt: Wöchentlicher Kundenkontakt, um die aktuell auffähige Version auszuliefern, und um neue Änderungen zu besprechen.
- Monatstakt: dem Kunden wird regelmäßig eine neue Release ausgeliefert.

Die kleineren Einheiten sind immer mehrfach in der größeren enthalten. Solange eine Iteration fehlerhafte Tests liefert wird diese in meherere kleiner Iterationshasen aufgeteilt und in die nächste Iteration verschoben. Auch während der iteration können Änderungen vom Kunden durchgeführt werden.

Um erfolgreiche Software entwickeln zu können müssend die folgenden Werte: Kommunikation, Einfachheit, Rückmelsdung, Mut und Respeckt eingehalten werden.

Das Vorgehen wird im Team ständig hinterfargt, um Prinzipien wie z.B. Selbstsicherheit, Qualität zu gerwährleisten. Dasurch ist lauffähige Software zu jedem Zeitpunkt gewährleistet, bei dieser jedoch anfängliche Fehlschläge mit einkaluliert werden mussen.

Zum Einstz kommen Praktiken wie z.B. Pair-Programming, bei den durch den Vier-Augen-Prinzip Fehler gleich gefunden werden, oder Refectoring, bei dem ständig Architektur-, Design- und Code-Verbesserungen vorgenommen werden.

2.4.2 Vorteile

- schnelle Entwicklung der Software (eine funktionierende Software)
- nur eingesetzte Funktionen werden entwickelt
- Softwareentwicklung lässt sich mit den Kunden vereinbaren, und bietet schnelle bewertbare Ergebnisse
- mehr Kommunikation im Projekt durch die kürzeren Zyklen
- Kunde wird stärker beim Entwicklungsprozess mit eingebunden, und hat dadurch ein besseres Verständniss
- Kosten bleiben bei großen Software-Projekten verhältnismäßig gering
- Höhere Qualität, Erweiterbarkeit und Nachhaltigkeit bei der Entwicklung
- XP hat eine positive Auswirkung auf dem Teamgeist und der Motivation der Entwickler so wie ein Erfolgserlebnis bei der Auslieferung einer neuen Version

2.4.3 Nachteile

- schlecht bei Kleinen Softwareprojekten
- schwierig das nur bei Beginn definierte Anforderungen mit dem geringsten Aufwand umgesetzt werden können
- hohe Anforderung an die Entwickler, da die Software im jeden Zustand erweiterbar sein sollte
- Kunde hat nicht ausreichend Zeit, um sich regelmäßig mit den Entwicklern zu treffen
- die Test-Entwicklung kann schnell zum Endprodukt werden bzw. ausgelassen werden
- der Kunde muss Vertrauen in den Programmierer haben

2.4.4 Zusammenfassung

Projektphase:

Das Projekt wird in kleine Teilschritte zerlegt, die einzelne Funktionsblöcke enthalten.

Projektgröße:

Gut geeignet für kleine bis mittelgroße Teams (<10-12 Mitglieder).

Grad der Planung:

Durch das Mitwachsen der Anforderungen verläuft die Kostenkurve gleichmäßig linear.

Änderungen während Iterationen:

XP ist auf schnelle Anforderungen ausgelegt weswegen Änderungen immer möglich sein müssen.

Flexibilität:

XP muss in einem Zug eingeführt werden, die Dauer der Iteration hängt vom Kunden ab.

Grad der Eigenverantwortung:

Die Eigenverantwortung liegt von der Funktionalität beim Kunden, und die Zuverlässigkeit und Umsetzung beim Entwcker.

Einarbeitungsaufwand:

XP erfordert Softwareentwickler mit besonderen Fähigkeiten, da alle Teammitglieder Entwickler sind.

Einprojekt- oder Mehrprojektumgebungen:

Der Projektmanager kann an mehreren Projekten arbeiten.

3 Gewichtung

Im Folgenden werden die Projektmanagementmethodiken, die wir ausgewählt haben verglichen und gegeneinander aufgewogen.
Für die Kriterien Projektphase und Mehrprojektumgebung wird eine Range von 0 bis 1 gewählt, da es entweder geht oder nicht. Für das Kriterium Flexibilität wird eine Range von 1 bis 2 gewählt, da eine Methodik entweder flexibel sein kann oder nicht. Alle anderen Kriterien haben eine Range von 1 bis 3, jeweils ob diese das Kriterien gut, sehr gut oder ausgezeichnet erfüllen.

	Scrum	*Kanban*	*XP*	*FDD*	*Gewichtung*	*Range*
Projektphase	1	1	0	0	2	0 - 1
Projektgröße	1	3	1	3	1	1 - 3
Grad der Planung	3	2	2	1	2	1 - 3
Änderung während Iteration	0	1	1	0	2	0 - 1
Flexibilität	1	2	1	1	2	1 - 2
Grad Eigenverantwortung	2	3	3	1	3	1 - 3
Einarbeitungsaufwand	2	2	3	1	3	1 - 3
Mehrprojektumgebung	1	1	0	1	1	0 - 1
Gesamt	24	31	27	14		

Berechnungsgrundlage:
Gesamt = Projektphase * Gewichtung + Projektgröße * Gewichtung + …

Beste Projektmanagementmethodik:

Platz	Framework
1	Kanban
2	XP
3	Scrum
4	FDD

Die beste Projektmanagementmethodik nach dem gewichteten Vergleich ist Kanban gefolgt von Extreme Programming. Aufgrund der vom Kunden gestellten Anforderungen wird allerdings dennoch Scrum genommen.

4 Quellen

Kanban

http://www.heise.de/developer/artikel/Kriterien-fuer-eine-Entscheidung-fuer-Scrum-oder-Kanban-1071172.html

https://de.wikipedia.org/wiki/Kanban-Tafel

http://de.wikipedia.org/wiki/Kanban_%28Softwareentwicklung%29

https://www.it-agile.de/fileadmin/docs/veroeffentlichungen/Artikel_OS_02.10_Interview.pdf

https://www.it-agile.de/wissen/methoden/kanban/fragen-zu-kanban/

Scrum

http://de.wikipedia.org/wiki/Scrum

http://scrum-master.de/Was_ist_Scrum/Scrum_auf_einer_Seite_erklaert

https://www.crisp.se/file-uploads/Kanban-vs-Scrum.pdf

http://www.heise.de/developer/artikel/Kriterien-fuer-eine-Entscheidung-fuer-Scrum-oder-Kanban-1071172.html

https://www.scrumalliance.org/community/articles/2014/july/scrum-vs-kanban

FDD

Feature Driven Development (FDD): Scrum v.s. FDD:
https://www.youtube.com/playlist?list=PLS7DQ8YkIVTpXyiWb2zl5F5rtpHhhPfsJ

https://en.wikipedia.org/wiki/Feature-driven_development

https://dzone.com/articles/introduction-feature-driven

XP

https://blog.seibert-media.net/blog/2005/05/01/extreme-programming-vorgehensmodell-zur-software-entwicklung-bei-seibertmedia/

https://www.it-agile.de/wissen/methoden/extreme-programming/

http://www.computerwoche.de/a/extreme-programming,2352505,2

http://www.computerwoche.de/a/extreme-programming,2352505,3

https://de.wikipedia.org/wiki/Extreme_Programming#Flexibilit.C3.A4tsgrad_vs._Steifheit

https://www.st.cs.uni-saarland.de/edu/lehrer/xp.pdf