Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Agiles Projektmanagement – Scrum

von Heidrun Schienke

Heutzutage gibt es viele verschiedene Arten sein Team zu koordinieren, wenn ein Softwareprojekt umgesetzt werden soll. Hier soll Scrum näher erläutert werden. Dafür werden zunächst Probleme der klassischen Softwareentwicklung erörtert und dann Scrum erläutert. Außerdem werden eine Fallstudie beschrieben sowie eine Möglichkeit zum Informationsaustausch der Entwickler untereinander.

Probleme der klassische Softwareentwicklung

In der klassischen Softwareentwicklung wird die Vorgehensweise während des Projektes durch verschiedene Modelle beschrieben, wie zum Beispiel das Wasserfallmodell oder das V-Modell. Bei diesen Modellen wird der gesamte Entwicklungsprozess in verschiedene Stufen unterteilt, welche dann der Reihe nach abgearbeitet werden. Dieses Verfahren schränkt die Agilität der Bearbeitung deutlich ein. Das ganze Projekt muss vor der Anwendung auf eines der Modelle genau analysiert werden, um die Dauer und die Kosten bestimmen zu können. Dies ist in der Realität aber nicht möglich. Daher ergeben sich Probleme sowohl für den Kunden als auch für den Anbieter. Bevor der Kunde ein Projekt in Auftrag gibt, möchte dieser natürlich wissen was es kostet und wie lange die Fertigstellung dauern wird. Ohne eine langwierige Analyse kann darüber jedoch keine Auskunft gegeben werden. Diese Analyse würde den Kunden allerdings schon Geld kosten und dabei kann es dann passieren, dass das eigentliche Projekt trotzdem nicht durchgeführt werden kann. Für den Anbieter kann es zu dem Problem kommen, dass er etwas zu einem viel zu geringen Preis anbietet, da die Abschätzungen fehlerhaft waren. So würde er lediglich Gewinn machen, wenn er auf einen Folgeauftrag hoffen kann und somit bei dem Kunden nun den „Fuß in der Tür hat“ und diesem somit

bekannt ist. Ein weiteres Problem ergibt sich dadurch, dass die Anforderungen vor Beginn des eigentlichen Projektes definiert werden. Dadurch sind diese fix und können im Nachhinein nur schwer wieder geändert werden. Es kann dadurch geschehen, dass eine Änderung einen großen Mehraufwand bedeutet aufgrund von Abhängigkeiten. Um diese Nachteile auszugleichen kann zum Beispiel Scrum angewandt werden.

Scrum

Scrum ist eine Methode aus der agilen Softwareentwicklung. Dabei wird das Entwicklungsteam in verschieden Rollen eingeteilt, welche dann unterschiedliche Aufgaben übernehmen. Diese zeichnen sich dadurch aus, dass sie sich selbst organisieren, das heißt, dass sie zum Beispiel selber entscheiden in welcher Reihenfolge sie neue Features für ein Programm entwickeln. Eine weitere Besonderheit sind die verschiedenen Meetings, die regelmäßig stattfinden und sich an genau festgeschriebene Regeln halten. Außerdem werden die Anforderungen in den sogenannten Artefakten immer wieder aufgearbeitet und angepasst. Der eigentliche Entwicklungsprozess läuft in Sprints ab. Ein Sprint hat immer eine festgelegte Anzahl von Tagen in denen eine bestimmte Funktionalität umgesetzt wird. Nach maximal dreißig Tagen sollte ein Sprint beendet sein. Innerhalb dieses Zeitraumes wird eine bestimmte Funktionalität umgesetzt, welche vorher festgelegt wurde. Es soll bei jedem Sprint ein Produkt entwickelt werden, welches nach Beendigung auch produktiv einsetzbar ist. Dieses wird nach dem Sprint durch den Auftraggeber abgenommen. Je nachdem wie der Sprint verlaufen ist, werden danach weitere Schritte eingeleitet. Bei Erfolg kann beispielsweise eine neue Funktionalität umgesetzt werden oder aber es kann eine weitere Verbesserung der entwickelten Funktionalität erfolgen. Nach jedem Sprint gibt es dann ein Feedback für das Team. Die einzelnen Teile der Scrum Methode werden im Weiteren genauer erläutert.

Rollen

Innerhalb eines Scrum Teams gibt es drei verschiedene Rollen. Die erste Rolle ist der Product Owner. Dieser vertritt fachlich gesehen den Auftraggeber. Er steht mit diesem in Kontakt und pflegt die Anforderungen (siehe Product Backlog). Er hat den Überblick über die Anforderungen und legt fest,

welche davon im nächsten Sprint umgesetzt werden sollen. Dabei muss er diese dementsprechend auch in eine logische Reihenfolge bringen und alle Abhängigkeiten miteinbeziehen.
Das Entwicklerteam entwickelt das Produkt. Es sollte idealerweise aus sieben Personen bestehen. Allerdings soll es sich aus Personen unterschiedlichen Typs zusammensetzen. So gibt es dort nicht nur Softwareentwickler, sondern auch Softwarearchitekten etc. In diesem Team spielt die Selbstorganisation eine sehr große Rolle. Die Aufgaben werden selbstständig vom Team aufgeteilt und zugewiesen.
Die letzte Rolle ist der Scrum Master. Dieser dient als Moderator innerhalb der verschiedenen Meetings und kümmert sich um die Kommunikation und Vermittlung innerhalb des gesamten Teams. Obwohl die Aufgaben des Scrum Masters einen übergeordneten Eindruck vermitteln, ist der Scrum Master dabei nicht der Projektleiter. Dasselbe gibt auch für den Proodut Owner. Alle Rollen sind gleichgestellt und es sollte, um Interessenskonflikte zu vermeiden, eine Person nicht mehrere dieser Rollen innehaben.

Meetings

Für Scrum gibt ein wichtiges Prinzip, das Time Boxing. Es beinhaltet, dass man immer sehr nahe Abgabezeitpunkte hat. Meistens arbeitet man um einiges schneller und effektiver, je näher ein Abgabezeitpunkt rückt. Damit man immer „unter Druck arbeitet“, werden die Abgabezeitpunkte möglichst kurz gewählt, um diesen Zustand praktisch künstlich hervorzurufen. Dies soll auch auf die Meetings übertragen werden. Damit diese nicht ausufern und ewig diskutiert wird, wird festgelegt, dass diese immer pünktlich beginnen und nach einer festgelegten Zeit beendet sein sollen.
Das erste Meeting ist das Sprint Planning Meeting. Dabei erarbeitet der Product Owner vorher ein Ziel für den nächsten Sprint. Das Entwicklerteam muss diesem dann entweder zustimmen oder widersprechen, falls dieses nicht umsetzbar sein sollte. Innerhalb des Meetings werden nun in Absprache mit den Entwicklern die Arbeitspakete für das Ziel des nächsten Sprints festgelegt. Diese werden dann innerhalb des Teams später selbstständig in Selbstorganisation umgesetzt. Der Scrum Master greift nicht in dieses Meeting ein, außer um Kommunikationsprobleme zu lösen. Für das Sprint Planing Meeting sind acht Stunden Zeit angedacht, an die sich aufgrund von Time Boxing gehalten werden soll.
Ein weiteres Meeting innerhalb von Scrum Teams ist das Daily Scrum Meeting. Dieses soll, wie der Name schon sagt, täglich durchgeführt werden und nicht länger als 15 Minuten dauern. Das Meeting dient dazu, dass das Entwicklerteam sich über den derzeitigen Stand und Fortschritt

austauschen kann. Damit das Zeitfenster eingehalten werden kann, ist dieses Meeting ein Stand-Up-Meeting. Der Product Owner und der Scrum Master sind ebenfalls anwesend. Jedoch hat dabei der Product Owner nur eine passive Rolle und greift nicht ins Meeting ein. Er beantwortet nur gegebenenfalls Fragen, die aufkommen in Bezug auf die Anforderungen etc. Der Scrum Master hat hier wieder seine Kommunikationsrolle, da das Meeting von ihm moderiert wird. Auch er beteiligt sich jedoch nicht am Austausch des Teams.

Das Sprint Review Meeting dient zur Präsentation der Ergebnisse. Nach jedem Sprint wird dieses Meeting abgehalten. Dabei werden die während des Sprints erbrachten Ergebnisse präsentiert und zwar am entwickelten System, um die Fortschritte und Neuerungen besser erkennen zu können. Alle Anwesenden, darunter sollte auch der Auftraggeber sein, geben nun Feedback zu den erbrachten Leistungen. Nun kann entschieden werden, wie es mit dem Produkt und der entwickelten Funktionalität weitergehen soll. Dies geschieht durch den Product Owner. Er entscheidet nun, ob die Funktion in den produktiven Einsatz gehen kann oder ob noch eine Weiterentwicklung benötigt wird. Ist die Funktionalität produktiv einsetzbar, kann eine Weitere entwickelt werden. Dieses Meeting soll vier Stunden dauern.

Das letzte Meeting ist das Sprint Retrospective Meeting. Hierbei geht es um die Verbesserung der Arbeit des Teams und der Zusammenarbeit. Das Team stellt zusammen, was während des Sprints gut gelaufen ist und was schlecht. Dann wird über die Ergebnisse diskutiert und es sollen Möglichkeiten zur Verbesserung für die schlechten Punkte gefunden werden. Auch dieses Meeting sollte ebenfalls nach jeden Sprint durchgeführt werden und dafür drei Stunden eingeplant werden.

Artefakte

Das erste Artefakt ist der Product Backlog. Dies sind die Anforderungen an das zu entwickelnde Produkt. Im Gegensatz zu den Anforderungen aus der klassischen Softwareentwicklung befindt sich im Product Backlog jedoch nur ein grob geschätzter Umfang der Projektes, sowie eine ungefähre Komplexität und Priorisierung der einzelnen Anforderungen. Diese werden über den gesamten Projektverlauf vom Product Owner kontinuierlich gepflegt und angepasst, wo dies nötig ist. So können auch Änderungen schnell eingearbeitet werden.

Was der Product Backlog für das gesamte Projekt ist, ist der Sprint Backlog für den Sprint. Darin werden die Anforderungen für den jeweils aktuellen Sprint festgehalten. Diese werden allerdings nicht durch den Product Owner, sondern durch das Team gepflegt. Sie sollten jeden Tag überprüft

und gepflegt werden. Dadurch kann das Team die Übersicht über den jeweiligen Sprint behalten und es ist zu jedem Zeitpunkt klar, welche Anforderungen schon fertiggestellt wurden und welche noch nicht. Der Scrum Master kann, falls er eine Gefahr für das Ziel der Sprints sieht, in den Sprint Backlog eingreifen. Dies ist allerdings wieder nur eine steuernde Rolle und kein aktives Eingreifen. Er kann so das Team wieder in die richtige Richtung lenken, um das Ziel doch noch zu erreichen.
Das letzte Artefakt ist das Product Increment. Dies ist ein Zwischenprodukt des Produktes, das entwickelt werden soll. Es wird im Sprint Review Meeting vorgestellt.

Überblick über Scrum

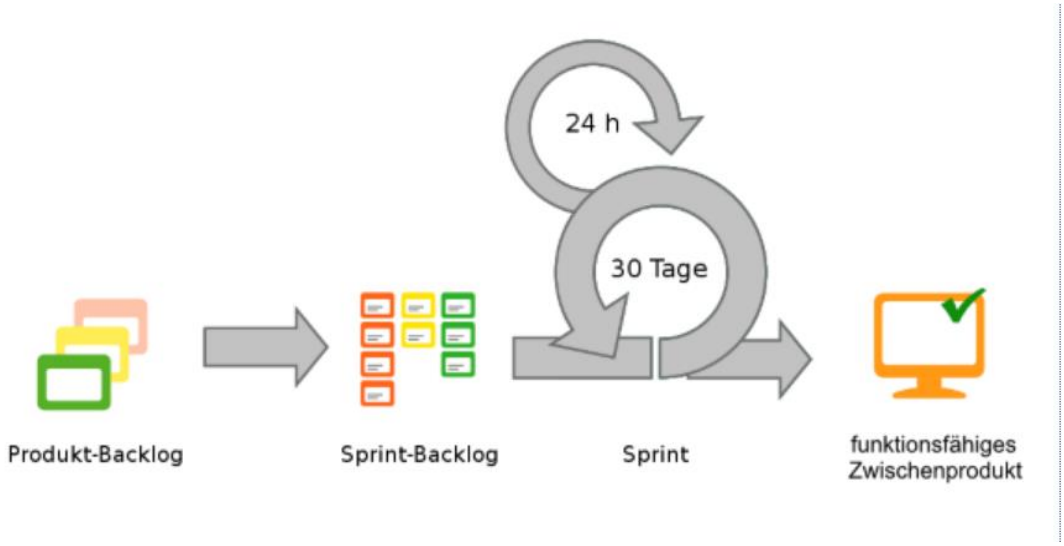

Abb. 1 Überblick über Scrum

In Abbildung 1 sieht man noch einmal übersichtlich die Zusammenhänge von Scrum dargestellt. Es startet links mit dem Product Backlog. Aus diesem ergibt sich nun der entsprechende Sprint Backlog für den nächsten Sprint. Nun geht es in eine Schleife von Sprints, wobei alle 30 Tage ein neuer Sprint ausgeführt wird. Innerhalb dieses Sprints sind hier die Daily Scrum Meetings dargestellt, welche jeden Tag oder wie hier alle 24 Stunden stattfinden. Ist der Sprint beendet, gibt es ein funktionsfähiges Zwischenprodukt, das Product Increment.

Fallstudie

In dem Paper „Software Evolution in Agile Development: A Case Study“ aus dem Jahre 2010 wurde bei IBM in Indien eine Fallstudie durchgeführt. Dabei wurde eine von IBM entwickelte Plattform eingesetzt, das IBM Rational Team Concert. Diese Plattform sollte den Mitarbeitern einen Überblick über das gesamte Projekt liefern, stellt die Sprints dar, bietet die Möglichkeit jederzeit die Anforderungen und User Stories einzusehen usw.

Die Studie wurde mit 60 Mitarbeitern über einen Zeitraum von 15 Monaten durchgeführt. Dabei gab es insgesamt 25 Sprints. Während der ersten elf Sprints wurde da bei ein Prototyp entwickelt, welcher dann zwischen den Sprints zwölf bis fünfzehn in eine Beta Release Phase gingen. Im Laufe der Studie wurden verschiedene Parameter aufgezeichnet, wie zum Beispiel die Änderung an allen Dateien des Projektes.

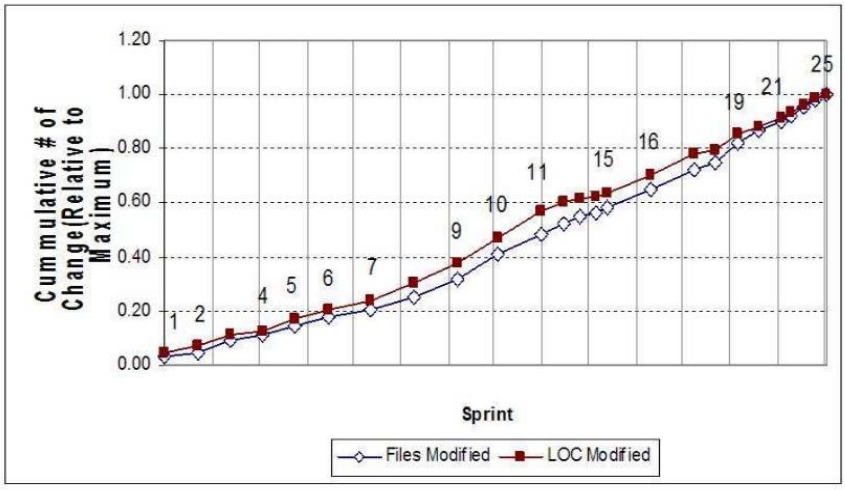

Figure 3. Law #1 - Changes in Files Relative to Maximum

Abb. 2 Änderungen in Dateien

In der obigen Abbildung (Abb. 2) sieht man wie viel Prozent der Dateien während der einzelnen Sprints geändert wurden. Dabei erkennt man deutlich den Zeitpunkt des Releases. Etwa ab Sprint sieben steigt die Anzahl der Dateien stärker an bis zu Sprint elf. Kurz vor dem Release wurden also noch viele Dateien verändert. Während des Beta Releases wurden dann kaum noch Änderungen vorgenommen. Dies stieg jedoch danach wieder sehr stark an, welches auf Verbesserungen und Bugs, die durch den Nutzer gefunden wurden, zu erklären ist.

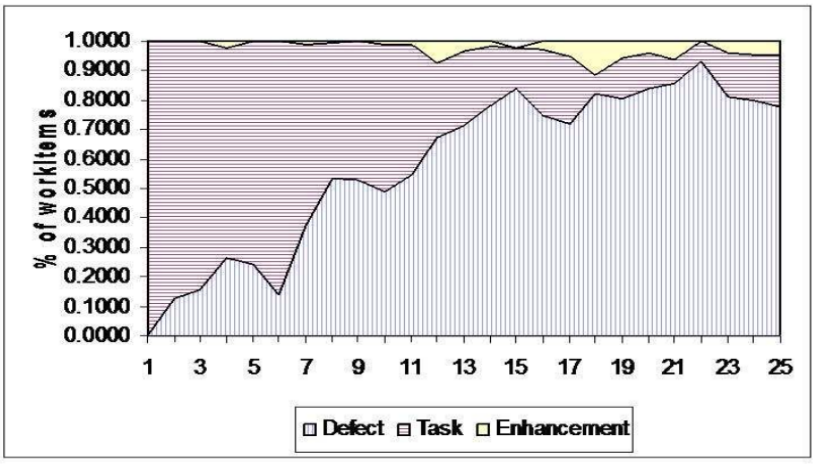

Figure 11. Law #7 - Distribution of Work Item Types across Sprints

Abb. 3 Klassifizierung der Arbeitspakete

Abbildung 3 zeigt die Aufteilung der Arbeitspakete während des Projektes. Hier sieht man, dass zu Beginn natürlich weder Bugs noch Erweiterungen existieren. Kurz vor dem Release steigt der Anteil der Bugs jedoch rapide an von etwa 15 Prozent auf 50. Dieser Trend setzt sich während des Releases weiter fort. Nun steigt auch die Zahl der Erweiterungen an. Gegen Ende haben die Erweiterungen einen Anteil von 5 Prozent, die Aufgaben haben etwa 20 und die Bugs 75. Allgemein ist zu erkennen, dass die Bugs den größten Teil der Arbeitspakete ausmachen. Diese steigen natürlich um den Release Zeitpunkt herum an. Es ist außerdem zu sehen, dass der Anteil der Bugs bei einem Anstieg der Erweiterungen ebenfalls ansteigt. Er sinkt jedoch, wenn sich die Erweiterungen auf einem Level einpendeln.

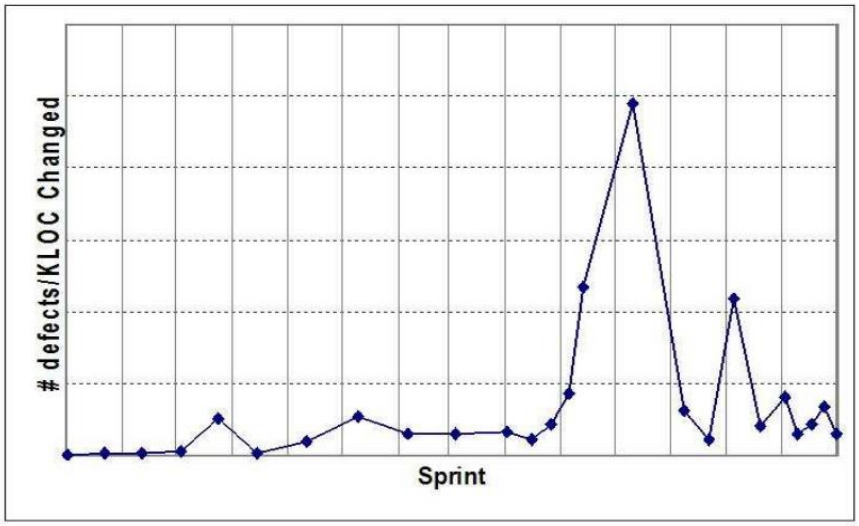

Figure 12. Law #7 - Defect Density across Sprints

Abb. 4 Dichte der Bugs

In dieser Abbildung (Abb. 4) sieht man die Meldungen der Bugs über das Projekt hinweg. Dabei ist der größte Anstieg etwa zum Release Zeitpunkt und kurz danach zu verzeichnen. Dies erklärt sich dadurch, dass der Prototyp dann durch die „echten“ Nutzer des Programms getestet wird und diese somit noch einmal sehr viele Bugs aufdecken.

Ambient Surface

Das Paper „Ambient Surfaces: Interactive Displays in the Informative Workspace of Co-located Scrum Teams“ aus 2016 beschreibt einen Versuch, der in einem Unternehmen mit 76 Mitarbeitern durchgeführt wurde. Dabei wurden in einem Raum im Unternehmen, den die Mitarbeiter häufig nutzen zum Beispiel in der Mittagspause, zwei Touchscreen Monitore aufgestellt (Abb. 5). Auf diesen Monitoren war JIRA zu sehen, welches alle Aufgaben des aktuellen Sprints anzeigt sowie ein Wiki, in welchem Infos der Softwarearchitekten zusammengefasst waren. Außerdem wurden dort die Build Informationen der Jenkins Server dargestellt und alle aktuellen Errors.

Figure 1. Left: The installation setup of both Ambient Surfaces in a common room on the ground floor; printers, whiteboards and the stairway to the upper level are in this area as well. Right: A custom visualization showing a build summary for a specific Jenkins job.

Abb. 5 Touchscreen Monitore

Es wurde nun untersucht, wie viele Touch-Events, also Interaktionen mit den Bildschirmen durchgeführt wurden. Dies waren durchschnittlich 203 pro Woche, wobei diese am häufigsten in der Mittagszeit aufgezeichnet wurden. Dies lag zum einen an den Daily Scrum Meetings, die vor den Bildschirmen durchgeführt wurden und diese mit einbezogen haben, zum anderen daran, dass die Mitarbeiter anfingen miteinander zu diskutieren über die sichtbaren Informationen auf den Bildschirmen. Die Mitarbeiter wurden dann zu dieser Methode befragt und von 28 Mitarbeitern gaben 13 eine tägliche Benutzung an, 14 eine wöchentliche und nur ein Mitarbeiter eine weniger häufige Nutzung. Außerdem fanden 90 Prozent aller Diskussionen vor den Bildschirmen mit Bezug auf die Informationen statt und fast 77 Prozent gaben an, dass die Sichtbarkeit von Informationen nun deutlich erhöht ist. Im Allgemeinen wurden die Bildschirme benutzt um sich einen Überblick zu verschaffen, Neuigkeiten zu erfahren und Informationen über die Arbeit der Kollegen einzuholen.

Zusammenfassung

Zusammenfassend ist zu sagen, dass Scrum den Entwicklungsprozess von Software positiv beeinflussen kann. Durch die enge Zusammenarbeit des Teams und der gemeinsamen Einteilung von Arbeitspaketen kann ein Produkt effizient entwickelt werden. Orte zum Austausch von Informationen wie Plattformen oder das Ambient Surface unterstützen dies. Es ist jedoch zu sagen, dass in der Praxis Scrum meist nicht komplett umgesetzt wird. So kann je nach Größe des Projekts und Sprint die Länge der Meetings stark variieren und von der Norm abweichen. Auch wird die Rollenverteilung nicht so streng eingehalten werden. Allerdings kann im Vergleich zur klassischen Softwareentwicklung mit Scrum ein engerer Kontakt zwischen den Mitarbeitern und dem Kunden erreicht werden, was ein wichtiger Grundstein für die erfolgreiche Umsetzung eines Softwareprojektes ist.

Quellen

http://www.scrumexpert.com/scrum-conferences/

http://projektmanagement-definitionen.de/glossar/scrum/

http://scrum-master.de/