Klassisch und doch agil - Scrum Erfahrungen

Stephan La Rocca
TEAM GmbH
Paderborn

Schlüsselworte
Projektmethoden, Scrum, Agile Software-Entwicklung, Oracle Forms

Einleitung
Der Vortrag begleitet ein Projekt zur umfassenden Erweiterung einer Individual-Software auf Basis klassischer Entwicklungswerkzeuge in einer agilen Methodenlandschaft. Von der Schwierigkeit der Angebotserstellung ohne "festes Pflichtenheft" über die Definition der User-Stories bis hin zur Pflege von Backlog-Dokumenten und den Erkenntnissen aus den Sprint-Retroperspektiven, sollen die Elemente der Scrum-Vorgehensweise in der "Praxistauglichkeit" bewertet werden.
Neben der Einbindung des Kunden in diese Vorgehensweise beleuchtet dieser Vortrag auch die Bedeutung der Rollen des Product Owners sowie des Scrum Masters.
Dass das Projektleben dann nicht immer der reinen Lehre entspricht und welche Hürden zu nehmen und Fettnäpfe zu umgehen sind, stellen das Fazit des Vortrages dar. Resümierend wird dieser budgetgetriebenen, agilen Vorgehensweise plakativ das Risikomanagement eines im Festpreis realisierten Pflichtenheftes gegenübergestellt.

Die Ausgangslage
Eine Software, die fast zwei Jahrzehnte Oracle Produktentwicklung erlebt hat, ein Kunde, der seine Systeme konsolidieren möchte, eine Architektur im Umbruch von Client/Server auf eine Web- und Service-orientierte Systemlandschaft, neue Anforderungen aus einem immer weitere Kreise umfassenden Anwender Umfeld. Eine gelungene Mischung für eine komplexe Individualentwicklung, Integration und Migration.
Die Basissoftware basierend auf Oracle Forms mit über 50 Integrationen auf Daten- und Applikationsebene.

Das Angebot
Wie würden Sie an diese Aufgabenstellung gehen? Die Erstellung eines Pflichtenhefts mit der akribischen Anforderung jeden erdenklichen Wunsch des Kunden abschätzen und anbieten zu wollen? Was ist bereits zu diesem Zeitpunkt in Stein gemeißelt und wie reagieren Sie auf Änderungswünsche oder „Interpretationen“ des Pflichtenheftes, die im Verlaufe des Projektes entstehen?
Darüber hinaus ist die Erstellung des Pflichtenhefts mit Kosten verbunden, die der Kunde in der Regel übernehmen muss. Sicherlich gibt es Modelle, in denen diese Kosten bei Auftragserteilung mit den Implementierungskosten verrechnet werden, aber es bleibt die Tatsache, dass die Erstellung des Pflichtenheftes sichtbaren Aufwand erzeugt.
Zusätzlich ist dieses Dokument in der Regel abnahmerelevant, und das in doppelter Hinsicht.
Zum Einen dient es beim Kunden zur Bewilligung des Budgets und zum Anderen dient es bei Projektabschluss zur Abnahme der Software.

Agile Softwareentwicklungsmethoden haben das Ziel, während des Entwicklungszyklus auf Änderungswünsche des Kunden entspannt reagieren zu können. Getreu dem Dire Straits Motto „Money for nothing chicks for free“ postulierte Jeff Sutherland (einer der Scrum Urväter) im Jahre

2008 den Kern eines Agilen Vertrages: „Money for nothing and change for free“.
Diese beiden Anforderungen für agile Verträge sollen festhalten, dass zum Einen (Money for nothing) der Kunde zu jedem Zeitpunkt des Projektes (sinnvollerweise nach Ende eines jeden Sprints) aussteigen kann und für eventuell bereits abgeschätzte aber nicht realisierte Funktionen nichts bezahlen muss. Zum Anderen kann er im Verlauf des Projektes (change for free) eine nicht mehr relevante durch eine andere Anforderung ohne zusätzliche Kosten ersetzen.

Mit diesem Wissen über die Vertragsgestaltung galt es nun ein Angebot zu erstellen, mit dem auf der einen Seite für den Kunden ein Budget sichtbar gemacht werden kann und auf der anderen Seite die Anforderungen des Kunden in sogenannten User-Stories benannt werden, ohne deren Umsetzung bis ins kleinste Detail zu bestimmen. User Stories werden im sogenannten Produkt-Backlog zusammengefasst.

In dem vorliegenden Fall wurden die UserStories sinnvoll gebündelt und durch verschiedene Abschätzungsmodelle beziffert. Auf die Abschätzungsmodelle wird auf Grund der Komplexität in diesem Artikel nicht weiter eingegangen.

Das Prinzip „Scrum“

Der Ablauf des Entwicklungsvorgehens orientiert sich an Projektzyklen, sogenannten Sprints. Für

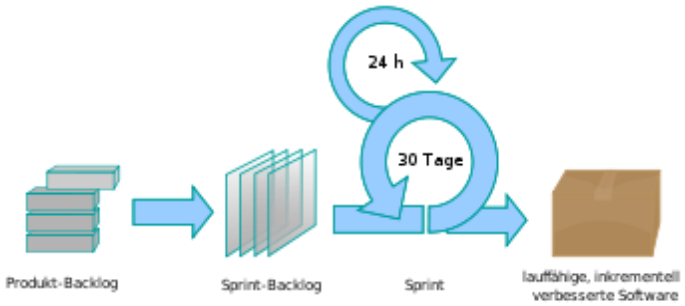

Abbildung 1: Scrum Entwicklungsprozess

jeden Sprint werden eine Reihe von User Stories, d.h. Kundenanforderungen, aus dem Produkt-Backlog gewählt. Diese werden dann im Sprint-Backlog zusammengefasst und im Verlauf des Sprints entwickelt. Am Ende entsteht daraus das Stück „fertige Software“, das die User Stories des zugehörigen Sprint-Backlogs umsetzt, real ablauffähig ist und vom Kunden nicht nur betrachtet, sondern auch getestet werden kann. Im Rahmen unseres Projektes haben wir uns für 3-wöchige Sprints entschieden. Dies definiert einen Zeitrahmen, nach dem dem Kunden deutlich sichtbare Fortschritte in den Projektergebnissen präsentiert werden konnten.

Das Team

In der Lehre sollen in einem Scrum-Projekt folgende Rollen besetzt werden:

User: Der Anwender, der das System später nutzen wird.

Product Owner: Der Kunde, der die Software Software-Erstellung in Auftrag gegeben hat.

Scrum Master: Eine Person, die den gesamten Prozess und die Kommunikation überwachen soll

Entwickler: Das Team von Entwicklern, die für die Software-Erstellung verantwortlich sind.

Der Product Owner ist in diesem Prozess sehr eng und nah mit dem Entwicklerteam verbunden. Das ist eine Rolle, die in dieser Ausprägung nur schwer beim Kunden gesehen wird. Wie wir später beleuchten, hat der Product Owner die Aufgabe am Daily Standup teilzunehmen.

Um den Product Owner in diesen Besprechungsrythmus integrieren zu können, haben wir in diesem Projekt die Rolle des Product Owner auf zwei Personen aufgeteilt. Auf der einen Seite steht der Projektleiter auf der Seite des Kunden, aufgestellt wie in jedem klassischen Modell mit Budgetverantwortung und dem Verständnis davon, was die Software leisten soll.
Auf der anderen Seite haben wir eine Person aus unserer Firma mit der Rolle des Product Owners betraut, insbesondere um die Teilnahme an den Daily Standups zu sichern und um Kleinstentscheidungen im Rahmen des Entwicklerteams nah und schnell beantworten zu können. Somit war die Kommunikation des Entwicklerteams zum Product Owner kundenseitig durch den TEAMProduct Owner gebündelt.
Die Rolle des Scrum Masters wurde ebenfalls durch eine Person der Firma TEAM besetzt. Neben den rollenbezogenen Aufgaben als „Kümmerer" des Teams zeigte der Projektverlauf, dass hier auch die Aufgaben der klassischen Projektleitung (Erstellung der Projektberichte, Fakturierung, Klassifizierung von Coding-Standards, etc.) zusammenfielen.

Die Meetings

Das Warmup:
Eigentlich nicht nur ein Meeting, sondern schon eine ganze Projektwoche als PreSprint haben wir investiert, um das Product Backlog zu erstellen und grob die Meilensteine der Realisierung zu skizzieren. Da die Reihenfolge der Umsetzung der User Stories nicht nur allein von der Priorität im Backlog abhängig ist, sondern auch durch andere Randbedingungen (Personalverfügbarkeit, Abhängigkeiten zu vorausgelagerten Projekten, sinnvolle Reihenfolgen durch Abhängigkeiten in den Softwaremodulen) beeinflusst wird, wurden in diesem Warmup-Sprint auch schon die Themen der nachfolgenden Sprints festgelegt.

Sprint Planning I:
Start eines jeden Sprints, in dem die Product Owner gemeinsam abstimmen, welche User Stories in dem kommenden Sprint umgesetzt werden. Ein durchaus schwieriger Prozess, da der Product Owner nicht freiwillig einräumt, dass eine Funktionalität *nicht* wichtig ist. Ein Lernprozess für den Kunden, dass es nicht „Alles oder Nichts" heißt, sondern, dass er mit einem begrenzten Budget *einkaufen* geht. Teilnehmer des Meetings sind darüber hinaus der Scrum-Master sowie das gesamt Team. Für bestimmte Sprints ist es sinnvoll, dass auch repräsentative Anwender an der Besprechung teilnehmen. Ziel der Besprechung ist ein (neu)priorisiertes Product Backlog und eine Auswahl der User Stories, die im kommenden Sprint umgesetzt werden sollen.

Sprint Planning II:
In dieser Besprechung steht die Abschätzung und die Diskussion über die Umsetzung der priorisierten und mit Business Value versehenen User Stories. Die Entwickler haben das Wort, diskutieren Ansätze und schätzen die Komplexität der einzelnen Punkte ab.
Besonders positive Erfahrungen haben wir mit der Abschätzung durch das sogenannte Planning Poker gemacht. Die Konsequenz, dass die Ausreißer in der Abschätzung nach oben und unten ein kurzes Statement ob ihrer Abschätzung geben, war sehr hilfreich um eine sichere Abschätzung abzugeben. Neben der Abschätzung der User Stories ist in diesem Meeting auch die Abstimmung darüber zu erzielen, welche Punkte das Team im nächsten Sprint realisieren wird.

Der Scrum-Master und der TEAMseitige Product Owner sind bei dieser Besprechung anwesend, haben aber kein Mitspracherecht.

Daily Standup:
Sicherlich das Meeting, welches den nachhaltigsten Effekt für die geänderte Vorgehensweise mit sich brachte. Nur eine Viertelstunde eines jeden Morgens, jeder mit dem kurzen Statement, welchen Task er gestern bearbeitet hat, was heute vor ihm liegt und ob es Punkte gibt, die ihn behindern.
Der tägliche Austausch förderte enorm den Wissenstransfer innerhalb des Teams und Hemnisse wurden direkt im Impediment Log festgehalten, welches die Aufgabenliste des Scrum Masters war.
Teilnehmer waren neben dem Team der Scrum Master und der TEAMseitige Product Owner.
Da wir in verteilten Teams arbeiteten und nicht über den gesamten Projektzeitraum alle Entwickler im Team waren, war doch, wenn immer es möglich war, die Teilnahme am Daily Stand up sicherzustellen. So blieb der Entwickler halbwegs im Thema und der Wiedereinstieg fiel nicht so schwer.

Sprint Abnahme:
Ausgesprochenes Ziel war es, nach jedem Sprint eine lauffähige, inkrementell erstellte Software zur Verfügung zu stellen. Auf dieser Version wurde eine Sprint Abnahme mit dem Product Owner und repräsentativen Anwendern durchgeführt. Die User Stories des Sprints wurden von dem Entwickler, der sie erstellt hat, präsentiert und im besten Fall abgenommen.
Entspricht die Implementierung nicht der Anforderung, wird sie als fehlerhaft eingestuft und verbleibt für den nächsten Sprint auf dem Sprint-Backlog. Generiert der Product Owner aus der Präsentation neue Ideen, Veränderungen oder Erweiterungen, so wurden daraus neue User Stories, die priorisiert im Product Backlog landeten.

Sprint Retroperspektive:
Im Anschluss an einen jeden Sprint haben sich Entwickler, Scrum Master und TEAMseitiger Product Owner zusammengesetzt, um die Highlights und Lowlights des letzten Sprints zusammenzutragen. Die Highlights sollten die Stärke für den nächsten Sprint visualisieren, die Lowlights waren Ansätze für Veränderungen.

Die Software-Entwicklung

In diesem Kapitel möchte ich nur exemplarisch auf zwei Besonderheiten eingehen.

Inkonsistente Teams in und über Sprints.
Agile Software-Entwicklung erfordern auch agile Teams. Ein Blick in unsere Ressourcenplanung zeigt, dass die Entwickler, die für dieses Projekt vorgesehen waren, im ersten Halbjahr 2011 insgesamt in über 100 Projekten unterwegs waren, im Schnitt pro Entwickler 11 Projekte pro Monat, davon in der Regel 5 gleichzeitige Projekte pro Woche.
Diese Tatsache führte in den bisherigen Projekten häufig zu der Gefahr, das mit der Abwanderung eines Entwickler aus dem Team auch das Spezialwissen ob seiner Aufgaben aus dem Team verschwanden. Dieser Gefahr wollten wir mit folgenden Ansätzen entgegentreten:
- keiner ist ausschließlich für eine Aufgabe verantwortlich
- Pair-Programming sollte Erfahrungsaustausch sicherstellen
- Teaminterne Querschnittspräsentationen machten jeden Entwickler zum wöchentlichen Präsentator seiner Arbeiten.

Nightly Build und automatisierte Test
Begriffe, die jedem aus der javabasierten Entwicklungsumgebung sicherlich bekannt sind, sollten auch in diesem, mit den klassischen Entwicklungsumgebungen ausgestatteten Tools Forms und PL/SQL-Programmierung umgesetzt werden.

Als Versionsverwaltung kam Subversion zum Einsatz, jeder Entwickler hatte alle Sourcen lokal in seiner Sandbox, nur die Datenbank-Objekte waren zentral auf der Entwicklungsdatenbank konsolidiert.
Leider konnten die Konfigurationsdateien des Forms-Servers nicht ungesehen über Subversion verwaltet werden, da diese Dateien in der Regel Bezüge auf die lokalen Verzeichnisse hatte, die nicht bei allen Entwicklern identisch waren und erst recht nicht mit den Umgebungsinformationen des Integrationsservers übereinstimmten.
Die Tests für die Datenbank-Sourcen wurden mit der Unit-Test-Komponente des Oracle SQL Developers erstellt und konnten so durch den Build-Server Hudson nächtlich automatisiert werden.
Der nächtliche Job sah also so aus, dass
1.) Alle eingecheckten Sourcen aus Subversion auf dem Datenbank-Server und dem Applikation Server ausggecheckt
2.) Alle Datenbank-Sourcen in die Datenbank geskriptet
3.) Alle Unit-Tests aufgerufen und die Ergebnisse protokolliert
4.) Alle Forms-Komponenten übersetzt und die Protokolldateien nach Fehlern geparst wurden.

Leider war es bislang noch nicht möglich, auch die Forms Module automatisiert zu testen. Eruiert wurden eine Reihe von Werkzeugen, zu denen aber zum Zeitpunkt der Erstellung dieses Artikels abschließend keine Beurteilung gegeben werden konnte.

Das Management

Natürlich ist es angebracht, um die Methode nicht durch die eingesetzten Werkzeuge, sondern durch die Philosophie der Vorgehensweise zu erfahren, auf Tools im ersten Scrum-Projekt zu verzichten. Jedoch ist es am Ende des Tages notwendig, in Zeit&Aufwandsprojekten dem Kunden einen Nachweis über die geleisteten Stunden in Bezug auf die geforderten User Stories zur Verfügung zu stellen.
Zum Einsatz kommt die Software Polarion mit Templates und Erweiterungen für Scrum Projekte. Diese ALM-Software unterstützt die Zuordnung von Tasks zu den User Stories, die diese implementieren sollen.
Da das Entwickler Team in seinen Debatten die Umsetzung der User Stories definiert, sind sie auch angehalten die notwendigen Schritte in Form der Tasks in Polarion zu erfassen. Ansatz für die Aufteilung der Tasks ist die Richtlinie, dass ein Task den Umfang einer Tagesaufgabe nicht überschreiten sollte. So ergibt sich recht einfach der abgeschätzte Aufwand und der Fortschritt ist in den täglichen Besprechungen leicht nachzuvollziehen.
Aus all diesen Daten (Zeiterfassung pro Projekt, Ressourcenplanung des Entwicklungsteams, und die Anzahl der offenen und abgeschlossenen Tasks) konnte täglich ein aussagekräftiges Sprintboard erstellt werden.

Das Fazit

Die enge Integration des Kunden über die gesamte Projektlaufzeit, das von Anfang an verfügbare und inkrementell wachsende System, sowie der gesteuerte Erfahrungsaustausch im Entwicklerteam stehen auf der Habenseite unserer Erfahrungen.
Die geänderte Rollenverteilung hat auf der anderen Seite zu gewissen Reibungsverlusten geführt. Aus Bereichsleitern wurden Scrum-Master, aus Projektleitern wurden Product Owner und Entwickler wurden aufgefordert, eigenständig und im Team Konzepte zu erarbeiten und Tasks zu generieren.

Ebenfalls noch nicht geformt haben sich alle Aufgaben, die flankierend zum zentralen Entwicklungsprozess stehen. Dazu gehören die Definition von Programmierrichtlinien und Prüfung deren Einhaltung, die Installation und Konfiguration der Entwicklungs-, Integrations- und

Testumgebung und schlussendlich auch Aufgaben zur Inbetriebnahme, Schulung und Wartung. Diese Aufgaben lassen sich schlecht über das Verfahren „Money for nothing, changes for free“ betrachten.

Kontaktadresse:
Stephan La Rocca
TEAM GmbH
Hermann-Löns-Str. 88
D-33104 Paderborn

Telefon: +49 (0) 5254-800865
Fax: +49 (0) 5254-800819
E-Mail sr@team-pb.de
Internet: www. team-pb.de