Agile IT-Projekte zum Festpreis - ein Widerspruch in sich?

Alexandra Kaiser

Juristisches IT-Projektmanagement
Wintersemester 2016/2017

Lehrstuhl für Programmierung und Softwaretechnik
Institut für Informatik
Ludwig-Maximilians-Universität München

15. Januar 2017

Erklärung

Ich erkläre, dass ich die vorliegende Arbeit selbstständig, unter Angabe aller Zitate und nur unter Verwendung der angegebenen Literatur und Hilfsmittel angefertigt habe.

München, den 15. Januar 2017

Zusammenfassung

Große, unüberschaubare und komplexe IT-Projekte sind in der heutigen Zeit keine Seltenheit mehr. Umso wichtiger sind daher Werkzeuge und Methoden, die das IT-Projektmanagement dabei unterstützen, solche Projekte zu strukturieren. Zu diesem Zweck wurden verschiedene Vorgehensmodelle entwickelt. Klassische Verfahren, wie das Wasserfallmodell, konkurrieren mit neueren, agilen Methodiken. Das Wasserfallmodell zeichnet sich durch detaillierte Anforderungsspezifikation aus, agile Methoden durch ihre Anpassungsfähigkeit. Agile Vorgehensweisen erfreuen sich in der Softwareentwicklung immer größerer Beliebheit. Juristische Rahmenbedingungen werden allerdings meist über herkömmliche Festpreisverträge geregelt, die auf dem Wasserfallmodell basieren. Unternehmen sind der Meinung, dass ein Festpreis Sicherheit bietet. Agilität und Festpreis widersprechen sich jedoch auf den ersten Blick. Um nicht auf die Vorteile agiler Softwareentwicklung verzichten zu müssen, aber trotzdem auf Grundlage eines Festpreises zu agieren, wurde der agile Festpreisvertrag entwickelt. Agilität und Festpreis passen nämlich sehr wohl zusammen.

Inhaltsverzeichnis

1 Einleitung

„Planung ist alles, Planung ist nichts,
und kein Plan übersteht den Kontakt mit der Realität “

Das Zitat von Helmuth Karl Bernhard Graf von Moltke beschreibt auf eine poetische Art und Weise, womit sich in dieser Ausarbeitung auseinander gesetzt wird: In der Softwareentwicklung stehen sich zwei konträre Gruppen gegenüber, die Befürworter des traditionellen Wasserfallmodells und die Anhänger der agilen Werte. Wo die, die am Wasserfallmodell festhalten, den Schwerpunkt auf eine detaillierte Planung legen, wird den Vertretern des Agilen Manifest nachgesagt, nicht viel von Plänen zu halten. Die Realität zeigt oftmals, dass Pläne flexibel gestaltet werden müssen, um im Einsatz zu bestehen.
Diese Ausarbeitung beschäftigt sich mit genau diesem Sachverhalt und gibt eine Antwort auf die Fragestellung, ob agile IT-Projekte mit einem Festpreis kombinierbar sind. Dazu werden in Kapitel 2 agile Methoden dem klassischen Wasserfallmodell gegenüber gestellt und in Kapitel 3 die Auswirkungen dieser Unterschiede auf IT-Projektverträge untersucht. In diesem Zusammenhang wird der agile Festpreisvertrag vorgestellt, der eine Adaption des herkömmlichen Festpreisvertrags und einen juristischen Rahmen zu einer möglichen Umsetzung agiler IT-Projekte zu einem Festpreis darstellt.

2 Vorgehensmodelle im IT-Projektmanagement

Laut dem Chaos Report der Standish Group im Jahr 2015 ist es in den vergangenen Jahren bei nur rund 30% (vgl. Tabelle 1) der 50.000 ermittelten IT-Projekte zu einem erfolgreichen Abschluss gekommen. Der Erfolg wird hierbei gemessen an der Einhaltung von vereinbarter Zeit und Kosten mit zufriedenstellendem Ergebnis. Rund 50% der Projekte haben mindestens eines der Kriterien nicht vollständig erfüllt, grobe 20% sind gänzlich fehlgeschlagen. [HW15]

	2011	2012	2013	2014	2015
Erfolgreich	29%	27%	31%	28%	29%
Teilweise erfolgreich	49%	56%	50%	55%	52%
Gescheitert	22%	17%	19%	17%	19%

Tabelle 1: Standish Group 2015 Chaos Report

(Quelle: https://www.infoq.com/articles/standish-chaos-2015 (09.Januar 2017))

Um dem Scheitern von IT-Projekten vorzubeugen, werden sogenannte Vorgehensmodelle eingesetzt. Vorgehensmodelle sind abstrakte Beschreibungen von Aufbau und Ablauf der Aktivitäten innerhalb einer Projektdurchführung. Sie bilden damit einen Rahmen, der den gesamten Projektprozess strukturiert und verbessert. [Lin14] Meist enthalten Vorgehensmodelle Angaben zu den einzelnen „Projektphasen und Meilensteinen, den Zwischenergebnissen, den Arbeitsschritten und deren Zuständigkeiten sowie den anzuwendenden Standards, Richtlinien, Methoden und Werkzeugen in einem Projekt “[Sei06] und helfen

somit die Komplexität eines Projekts zu verringern und die Transparenz zu erhöhen. Man unterscheidet hierbei zwischen traditionellen, prozess- oder planorientierten Ansätzen und agilen Vorgehensweisen.

2.1 Klassische Verfahren

Klassische Verfahren umfassen alle prozess- und planorientierten Vorgehensmodelle. Wie der Name schon sagt, liegt der Schwerpunkt bei solchen Vorgehensweisen auf dem Prozess selbst. Dieser wird in einzelne Phasen aufgeteilt. Somit ergibt sich ein klar vorgegebener Ablaufplan. Einer ausführlichen Planungsphase und detaillierter Dokumentation werden hohe Bedeutung beigemessen. [WT04] Bekannte Vertreter der klassischen Verfahren sind das Wasserfallmodell [Roy70] und das V-Modell [AHZ11].

2.1.1 Wasserfallmodell

Das 1970 von Dr. Winston W. Royce beschriebene Wasserfallmodell ist das erste veröffentlichte Vorgehensmodell und zählt zu den bekanntesten und viel verwendeten Modellen in der Softwareentwicklung.

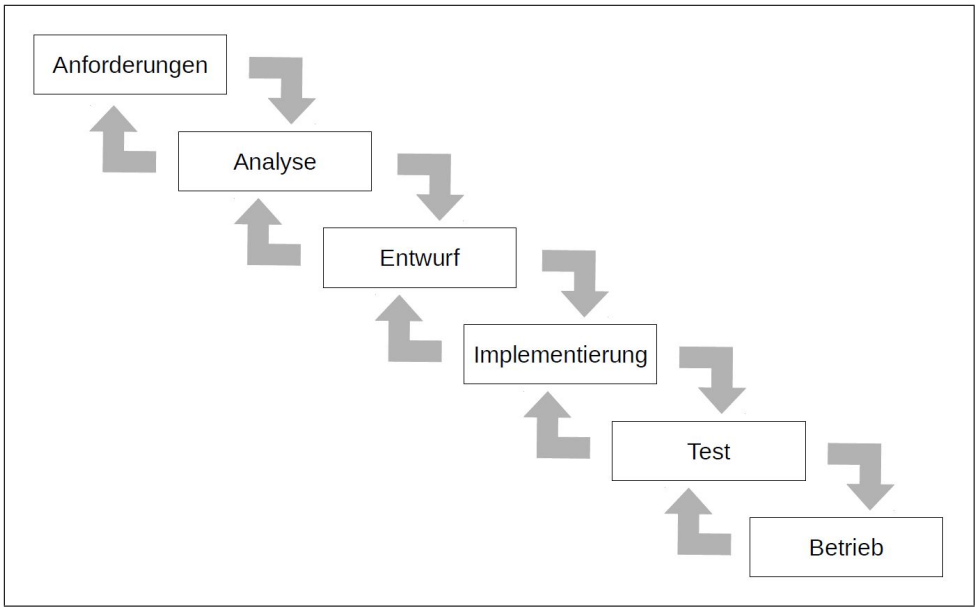

Abbildung 1: Wasserfallmodell nach W. Royce

Beim Wasserfallmodell sind die fundamentalen Aktivitäten in der Entwicklung (Spezifi-

kation, Design, Entwicklung und Test) in jeweils einer eigenen Phase realisiert. [WT04] An ein spezifisches Anwendungsszenario angepasst besteht es meist aus fünf oder sechs Schritten. Wie in Abbildung 1 beispielhaft zu sehen, wird immer mit der Anforderungsanalyse und -spezifikation begonnen. Das Lastenheft wird geschrieben. In der darauffolgenden Systemanalyse entsteht u.a. das Pflichtenheft. Dieser Phase folgt dann die „Design "-Phase. In dieser Phase werden auf der Grundlage der vorherigen Phase UML-Diagramme entworfen. Die Software wird designed. Nachdem der Entwurf steht, wird die Software implementiert; anschließend getestet und schließlich in Betrieb genommen und gewartet. Allen Modifizierungen des Wasserfallmodelles ist jedoch gemein, dass jede ihrer Phasen in sich abgeschlossen ist und Dokumente produziert, die als Eingabe für die nächste Phase dienen. Eine nachfolgende Phase wird erst begonnen, wenn die Phase davor vollständig beendet und geprüft wurde. Durch die kaskadenartige Darstellung der einzelnen Phasen erhielt das Wasserfallmodell seinen Namen. [Som10]

Für kleine, klar absehbare Projekte, d.h. für Projekte, bei denen zu Beginn bereits alle Anforderungen eindeutig definiert werden können und sich diese über die gesamte Dauer des Projekts auch nicht schwerwiegend verändern, ist das Wasserfallmodell ein äußerst effizientes Modell. [OGPM14]

2.2 Agile Softwareentwicklung

Allerdings trifft dies auf die wenigsten Projekte in der Softwareentwicklung wirklich zu. Viele IT-Projekte laufen über mehrere Jahre. Dass alle Anforderungen zu Beginn des Projekts bereits bis ins Detail ausformuliert werden können, ist eine Fehlannahme. Genauso, dass sich der Vertragsgegenstand während der Projektdurchführung kaum ändert. Sogar im Gegenteil: Der Verfall von Wissen in einer solchen Zeitspanne ist extrem hoch und gravierende Änderungswünsche von Seiten des Kunden keine Seltenheit. Doch genau diese Fehleinschätzungen führen zu dem Trugschluss, dass ein Vorgehen nach dem Wasserfallmodell Budgetsicherheit bietet. [OGPM14]

Den klassischen Verfahren gegenüber stehen daher agile Methoden. Diese entstanden in den 90er Jahren als Reaktion auf die prozessbezogenen, dokumentenlastigen Vorgehensmodelle und umschiffen genau diese Problematik der allumfassenden Anforderungsspezifikation. Statt auf Prozess und Dokumentation wird bei agilen Methoden der Fokus auf die Zusammenarbeit innerhalb des Teams und mit dem Kunden und auf schnelle Realisierung der Software gelegt. [WT04]

Methode	Erfolgreich	Teilweise erfolgreich	Gescheitert
Agil	39%	52%	9%
Wasserfall	11%	60%	29%

Tabelle 2: Chaos Report: Agil versus Wasserfall

(Quelle: https://www.infoq.com/articles/standish-chaos-2015 (09.Januar 2017))

Wie Tabelle 2 zeigt, werden erheblich mehr Projekte erfolgreich abgeschlossen, wenn sie

agil durchgeführt werden anstatt nach dem Wasserfallmodell. Die Beliebtheit agiler Softwareentwicklung steigt demnach stetig. [OGPM14]

Allen agilen Methoden liegt das Agile Manifesto zugrunde. [BBvB+01]

Manifest für Agile Softwareentwicklung

Wir erschließen bessere Wege, Software zu entwickeln, indem wir es selbst tun und anderen dabei helfen.
Durch diese Tätigkeit haben wir diese Werte zu schätzen gelernt:

Individuen und Interaktionen mehr als Prozesse und Werkzeuge
Funktionierende Software mehr als umfassende Dokumentation
Zusammenarbeit mit dem Kunden mehr als Vertragsverhandlung
Reagieren auf Veränderung mehr als das Befolgen eines Plans

Das heißt, obwohl wir die Werte auf der rechten Seite wichtig finden, schätzen wir die Werte auf der linken Seite höher ein.

Dieses beschreibt die Werte der agilen Softwareentwicklung: Kommunikation ist dabei einer der zentralen Grundsätze. Nicht nur die Softwareentwickler untereinander müssen regelmäßig miteinander reden, sondern auch der Kunde sollte sich kontinuierlich mit in das Projekt einbringen, nicht nur zu Beginn. Ihm wird eine eigene Rolle im Team zuteil und er trägt somit maßgeblich zum Projekterfolg bei. Lieferant und Kunde verfolgen dasselbe Ziel. So entsteht ein respektvoller und partnerschaftlicher Umgang miteinander, niemand arbeitet nur zu seinen eigenen Gunsten. Prozesse und Werkzeuge sollen unterstützend, nicht einschränkend wirken. [OGPM14] Ein weiterer Gegensatz zum Wasserfallmodell ist, dass bei agilen Methoden nicht erst am Ende des Projekts eine fertige Software entstanden ist, sondern in kurzen Abständen (meist innerhalb von zwei bis vier Wochen) immer wieder ein funktionsfähiges Produkt geliefert wird. [WT04] Dies gelingt, indem kurze Iterationen und Release-Zyklen eingeführt werden. So wird das Produkt inkrementell entwickelt, Planungs- und Entwicklungsphasen wechseln sich ab.[AHZ11] Aufgrund dieser Vorgehensweise ist es dem Kunden möglich, direktes und vor allem frühes (in Bezug auf die gesamte Projektdauer) Feedback zur bereits vorhandenen Software zu geben, der Lieferant hingegen kann flexibel und zeitnah auf Änderungswünsche des Kunden eingehen. [OGPM14]

Auch in den zwölf **Prinzipien der agilen Softwareentwicklung** [BBvB+] sind diese Werte verankert:

- „Unsere höchste Priorität ist es, den Kunden durch frühe und kontinuierliche Auslieferung wertvoller Software zufrieden zu stellen."
- „Heiße Anforderungsänderungen selbst spät in der Entwicklung willkommen. Agile Prozesse nutzen Veränderungen zum Wettbewerbsvorteil des Kunden."

- „Liefere funktionierende Software regelmäßig innerhalb weniger Wochen oder Monate und bevorzuge dabei die kürzere Zeitspanne.“
- „Fachexperten und Entwickler müssen während des Projekts täglich zusammenarbeiten.“
- „Errichte Projekte rund um motivierte Individuen. Gib ihnen das Umfeld und die Unterstützung, die sie benötigen und vertraue darauf, dass sie die Aufgabe erledigen.“
- „Die effizienteste und effektivste Methode, Informationen an und innerhalb eines Entwicklungsteams zu übermitteln, ist im Gespräch von Angesicht zu Angesicht.“
- „Funktionierende Software ist das wichtigste Fortschrittsmaß.“
- „Agile Prozesse fördern nachhaltige Entwicklung. Die Auftraggeber, Entwickler und Benutzer sollten ein gleichmäßiges Tempo auf unbegrenzte Zeit halten können.“
- „Ständiges Augenmerk auf technische Exzellenz und gutes Design fördert Agilität.“
- „Einfachheit – die Kunst, die Menge nicht getaner Arbeit zu maximieren – ist essenziell.“
- „Die besten Architekturen, Anforderungen und Entwürfe entstehen durch selbstorganisierte Teams.“
- „In regelmäßigen Abständen reflektiert das Team, wie es effektiver werden kann und passt sein Verhalten entsprechend an.“

2.2.1 Scrum

Eine der beliebtesten agilen Softwareentwicklungsmethoden ist Scrum [SS16] und wird daher nun exemplarisch als agile Methode vorgestellt.
In Scrum gibt es klar verteilte Rollen, feste Meetings und Abläufe und bestimmte Scrum-spezifische Artefakte. Der Projektlauf ist in mehrere, kurze (üblich sind zwei bis vier Wochen) Iterationen, sogenannte Sprints, unterteilt. Am Ende jedes Sprints sollte ein potentiell auslieferbares Produkt-Inkrement (Potentially shippable product increment) stehen, d.h. ein Teilprodukt, das voll funktionsfähig und getestet allen vereinbarten Qualitätsmerkmalen entspricht und dem Kunden zum Einsatz übergeben werden kann. Ein Scrum-Team besteht aus dem Product Owner (Kunde), einem Scrum Master und fünf bis neun (idealerweise sieben) Softwareentwicklern. Der Scrum Master sorgt für reibungslose Einhaltung des Scrumprozesses. Am Anfang der Entwicklung steht eine Projektvision (grundlegende Idee des Produkts), die der Product Owner in Form von User Stories (kurze anwendungsbezogene Textbeschreibung der Anforderungen) in einen priorisierten Product Backlog einträgt. Der Produkt Owner kann jederzeit Änderungen (z.B.: Entfernen noch nicht implementierter Anforderungen oder Einfügen neuer Anforderungen) am Product Backlog vornehmen. In einem Sprint-Planungsmeeting wird vom gesamten

Team das Sprint-Ziel festgelegt: User Stories, die in diesem Sprint bearbeitet werden, werden in einen Sprint Backlog geschrieben, dann verfeinert und (nach Aufwand) geschätzt. Während des Sprints wird das Teilprodukt, welches durch die User Stories beschrieben ist, entworfen, vollständig implementiert und getestet. Jeden Tag findet ein Daily Scrum Meeting statt, bei dem der aktuelle Stand der Entwicklung offen gelegt wird. Dies sorgt ebenfalls für erwünschte Transparenz. Zum Abschluss des jeweiligen Sprints wird das Teilprojekt in der Sprint Review vorgestellt und vom Product Owner abgenommen. In einer anschließenden Retrospektive wird Feedback diskutiert. [Pre12]

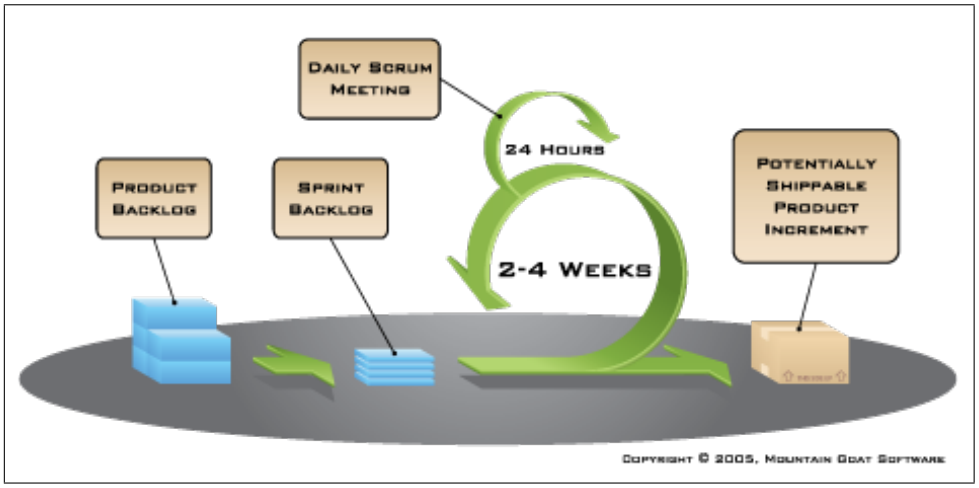

Abbildung 2: Scrum

(Quelle: https://www.mountaingoatsoftware.com/uploads/blog/ScrumMediumLabelled.png) (13.Januar 2017)

3 IT-Projekt-Verträge

Um sowohl IT-Projekten, die nach dem klassischen Wasserfallmodell entwickelt werden, als auch agilen Projekten einen juristischen Rahmen zu geben, schließen Auftraggeber (Kunde) und Auftragnehmer (Softwarelieferant) Verträge miteinander ab. Über Jahre durchgesetzt und erhalten hat sich hierbei der klassische Festpreisvertrag. Doch diese traditionelle Vertragsform hat gerade in Zusammenhang mit Projekten, die agil umgesetzt werden, einige Schwachstellen. Um diese zu umgehen, wurde mit dem agilen Festpreisvetrag eine mögliche Alternative geschaffen. [OGPM14]

3.1 Herkömmliche Festpreisverträge

Unter dem herkömmlichen Festpreisvertrag versteht man einen Werkvertrag mit Festpreisregelung, bei dem auf Basis einer vollständigen und endgültigen Leistungsbeschreibung wichtige (juristische) Rahmenbedingungen für das Projekt wie Vergütung, Abnahme und Termine definiert werden. Er basiert auf dem klassischen Wasserfallmodell, bei dem davon ausgegangen wird, dass zu Beginn des Projekts bereits der gesamte Vertragsgegenstand festgeschrieben werden kann. [OGPM14]

3.1.1 Nachteil herkömmlicher Festpreisverträge

Gerade in Hinblick auf agile IT-Projekte weist der herkömmliche Festpreisvertrag allerdings einige ungeklärte Sachverhalte auf, die zu möglichen Problemen und letztendlich auch zum Scheitern des Projekts führen können, aber nicht zwangsläufig müssen.
Eines der größten Probleme hierbei ist wohl die Variabilität des zu Liefernden. Der Vertragsgegenstand ist eben nicht, wie fälschlicherweise angenommen, zu Beginn des Projekts schon klar bzw. ändert sich mit hoher Wahrscheinlichkeit noch gravierend während des Projektverlaufs. Das hat teure Change Requests zur Folge. Bei agiler Entwicklung kann dieser Fall (des sich ändernden Scopes) nicht nur eintreten, sondern ist sogar explizit erwünscht bzw. beabsichtigt. Außerdem ist eine detaillierte Beschreibung des vollen Leistungsumfangs zu Beginn des Projekts nicht vorgesehen. [OGPM14]
Ein weiterer Punkt ist, dass der Auftraggeber in dieser Vertragsform lediglich eine Mitwirkungspflicht hat. Für eine agile Projektdurchführung ist es jedoch essentiell und von großer Bedeutung, dass der Kunde Teilverantwortungen übernimmt und aktiv an der Projektdurchführung beteiligt ist.
Ebenfalls sieht der herkömmliche Festpreisvertrag meist nur eine große Endabnahme vor, nicht, wie bei agilen Projekten üblich, Zwischenabnahmen. Das wiederum wirft die Frage nach einer passenden Vergütungsregelung auf, da hierbei, im Gegensatz zum Wasserfallmodell, schon vor Ende des Projekts benutzbare Software an den Kunden übergeben wird. Beim klassischen Festpreisvertrag ist die Vergütung meist an die Endabnahme durch den Auftraggeber gebunden. [Sar16]

3.2 Agiler Festpreisvertrag

In Anbetracht der schlechten Vereinbarkeit von agilen Projekten mit dem klassischen Festpreisvertrag wurde der agile Festpreisvertrag entwickelt. Er kombiniert laut [OGPM14] „den Bedarf an einem fixen Kostenrahmen mit den Grundlagen agiler Entwicklungsmethoden“, indem er eine Preisobergrenze (auch Maximalpreis genannt) vorgibt ohne jedoch eine finale Leistungsbeschreibung vorauszusetzen. Abbildung 3 zeigt eine Einordnung beider Vertragsarten.

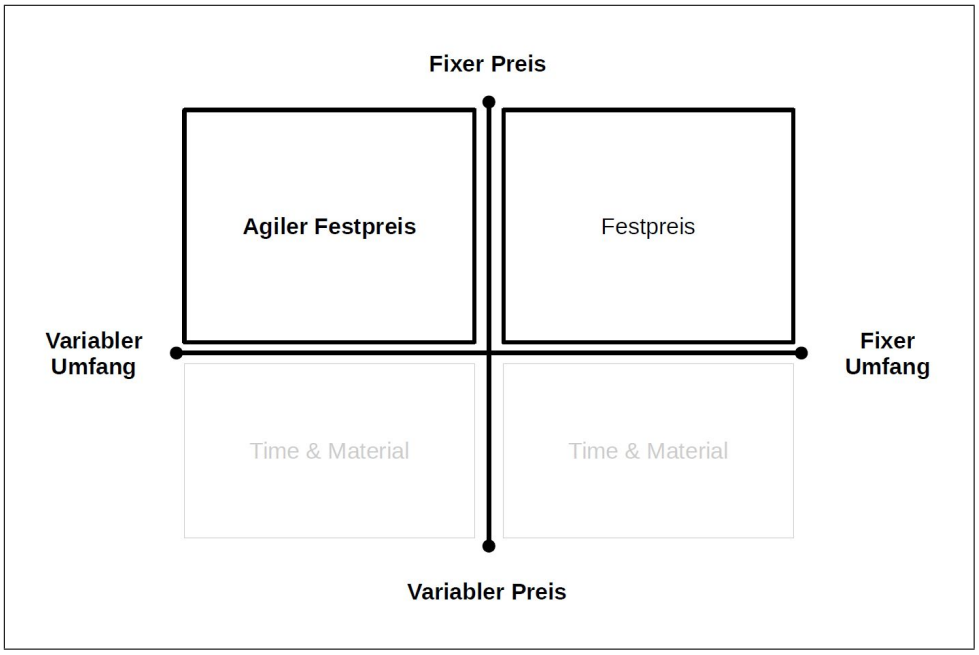

Abbildung 3: Einordnung der Vertragstypen [OGPM14]

Im Vergleich zum herkömmlichen Festpreis, basierend auf dem Wasserfallmodell, gibt es erst einmal keinen detaillierten Scope, sondern nur einen geschätzten Umfang (vgl. auch Abbildung 4). Dies ermöglicht eine spätere Änderung der Anforderungen (ohne teure Change Requests). Dies bedeutet allerdings nicht, dass der Umfang beliebig in Größe schrumpfen oder wachsen kann, sondern nur, dass bereits festgelegte Features gegen andere, gleich große Anforderungen ausgetauscht werden können. Diese Möglichkeit ist unter der „**(Ex)Change for Free**“- Klausel bekannt. [RW15] Können neue Anforderungen nicht durch dieses Prinzip in den Projektscope eingebracht werden, müssen auch beim agilen Festpreisvertrag Change Requests gestellt werden, die dann zu Mehrkosten führen, was jedoch selten eintritt. [OGPM14]
Die Leistungsbeschreibung wird erst während des Projekts verfeinert. Somit spart man sich unnötige (Stichpunkt Variabilität) Detailspezifikation am Projektstart, beugt Wissensverfall vor und kann ohne großen Aufwand Änderungen am Scope vornehmen. Genau das, was agile Softwareentwicklung ausmacht. [OGPM14]
Beim agilen Festpreisvertrag handelt es sich um einen Kooperationsvertrag, d.h. die Zusammenarbeit von Kunde und Lieferant ist unumgänglich und fest im Vertrag verankert. Beide Parteien arbeiten Hand in Hand und fällen wichtige Entscheidungen immer zusammen. Methoden zur gemeinsamen Aufwandsschätzung und eine genaue Vorgehensbeschreibung für die Scope-Governance (Steuerung des Projektinhalts) sind feste Bestandteile des Vertrags. [OGPM14]

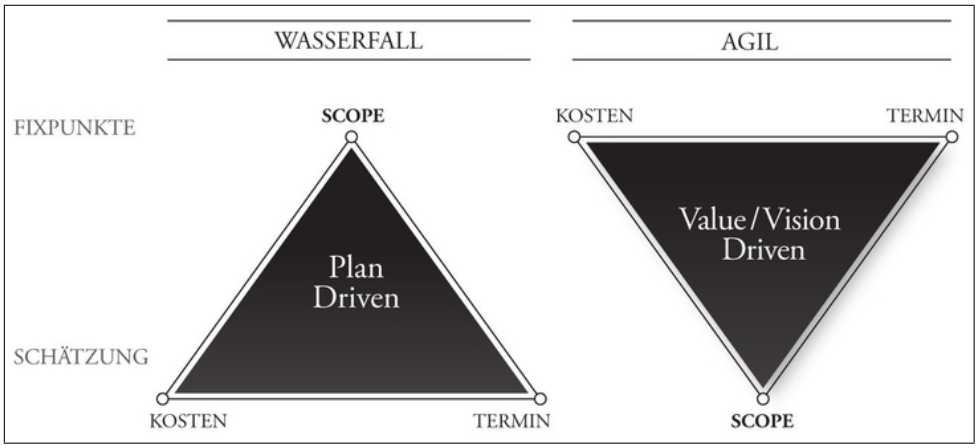

Abbildung 4: Das Projektmanagement-Dreieck [Glo16]

3.2.1 Kernelemente des agilen Festpreisvertrags

Um einen agilen Festpreisvertrag auszuarbeiten, sind folgende zentrale Schritte zu befolgen:

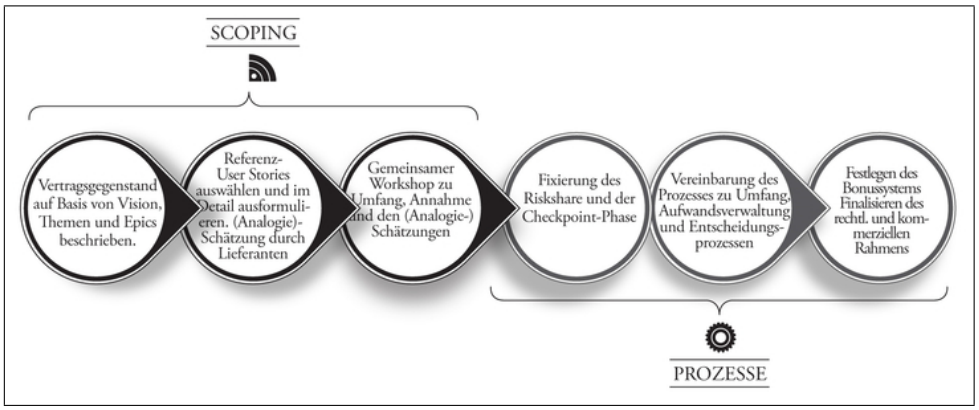

Abbildung 5: Scoping und Prozessdefinition [Glo16]

Schritt 1 - Definition des Vertragsgegenstandes
Auch, wenn zum Zeitpunkt des Vertragsabschlusses noch keine *detaillierte* Leistungsbeschreibung vorliegen muss, ist es notwendig eine *vollständige* Beschreibung der Gesamtanforderungen zu definieren, d.h. im 1. Schritt auf dem Weg zum agilen Festpreisvertrag wird der Vertragsgegenstand grob, aber dennoch vollständig auf der Ebene von Produkt- oder Projektvision, Themen und Epics (inhaltlich zusammenhängende Gruppe von User Stories) beschrieben.

Schritt 2 - Detailspezifikation einer exemplarischen Menge an Referenz-User Stories
In Schritt 2 wird ein repräsentatives Epic ausgewählt und alle zugehörigen User Stories formuliert. Diese User Stories werden daraufhin geschätzt und dienen als Referenz-User Stories für den nächsten Schritt.

Schritt 3 - Workshop zum Gesamtscope
In einem gemeinsamen Workshop schätzen Kunde und Lieferant auf der Grundlage der Referenz-User Stories die übrigen Epics. So wird ein vorläufiger, indikativer Maximalpreis für das Projekt ermittelt.

Schritt 4 - Riskshare, Checkpoint-Phase und Ausstiegspunkte
Im 4.Schritt werden Vereinbarungen zu Riskshare, Checkpoint-Phase und Ausstiegspunkten getroffen.
Checkpoint: Der Checkpoint ist ein Termin nach meistens zwei bis fünf Sprints (vergleichbar mit einer Probezeit), zu dem der Vertragsgegenstand nochmals modifiziert werden kann, der indikative Maximalpreis in einen endgültigen, echten Festpreis umgewandelt wird und bei unüberbrückbaren Differenzen jeder der zwei Vertragspartner die Möglichkeit hätte, zu diesem Zeitpunkt erstmals aus dem Projekt auszusteigen.
Ausstiegspunkte: Die Vertragspartner haben nach der Checkpoint-Phase jederzeit die Möglichkeit nach einer vereinbarten Frist (beispielsweise zwei Sprints) die Zusammenarbeit für beendet zu erklären. Das Modell „**Money for Nothing**“ kann hierbei zusätzlich mitaufgenommen werden: Es besagt, dass bei frühzeitiger Beendigung des Projekts durch den Auftraggeber der Auftragnehmer 20% des restlichen Vertragsbestandteils an Kosten als Entschädigung und zur Durchführung des verfrühten Projektabschlusses erhält. [Dub11]
Riskshare: Wird der Maximalpreis wegen eines Fehlers auf beiden Seiten trotz getroffener Maßnahmen zur Vermeidung eben dieses Szenarios überschritten, kommt der Riskshare zum Tragen, d.h. beide Parteien kommen für den Mehraufwand auf. Der Wert des Riskshares ist dabei ein festgelegter Prozentsatz der vereinbarten Teamkosten (meist ein Wert zwischen 30% und 70%), der dem Kunden bei Überschreitung des Maximalpreises erlassen wird. (Ein Riskshare von 0% bedeutet also, dass der Kunde alle Kosten für den Mehraufwand alleine trägt.) Bei Change Requests gilt der Riskshare nicht. [OGPM14]

Schritt 5 - Vereinbarung zur Scope-Governance
Aufgabe von Scope-Governance in agilen Projekten ist die iterative Navigation eines High-Level- zu einem Detail-Scope. Um dies möglichst reibungslos durchzuführen, werden Gremien gebildet, die sich zu bestimmten Zeiten zusammenfinden. Wichtige Aktivitäten und Prozesse hierbei sind die Ausformulierung und Schätzung der restlichen User Stories zu Sprintbeginn, der Scope-Governance-Prozess und der Scope-Eskalationsprozess. Der Scope-Governance-Prozess ist dafür zuständig, dass der Budgetrahmen nicht überschritten wird. Er sorgt dafür, dass Pläne erarbeitet werden, um zu komplexe User-Stories zu vereinfachen oder gänzlich zu eliminieren. Tritt während des Scope-Goverance-Prozess keine Einigung ein, wird die nächsthöhere (und höchste) Eskalationsstufe eingeleitet, der Scope-Eskalationsprozess. Es wird erneut versucht, einen Kompromiss zu finden, ggf. unter Zu-

hilfenahme eines vorher von beiden Seiten festgelegten IT-Sachverständigen.

Schritt 6 - Festlegen eines Bonussystems
In Schritt 6 werden Regelungen für eventuelle Boni getroffen, beispielweise der oben genannte „Money for Nothing“- Ausgleich.

4 Fazit

Die Frage „Agile IT-Projekte zum Festpreis - ein Widerspruch in sich? “ kann also ganz klar mit Nein beantwortet werden. Festpreis und Agilität widersprechen sich nicht grundsätzlich, sondern können sehr gut miteinander harmonieren solange man bestimmte Vorkehrungen (z.B. „(Ex)Change for Free“- Klausel im Vertrag, variabler Scope) trifft. [Bec13] Der Festpreis hat sich aus Sicht des Kunden etabliert, weil er eine bestimmte Budgetsicherheit bietet. Agilität sieht der Auftraggeber jedoch oft noch als Unsicherheitsfaktor an, weshalb ein Festpreis auch bei agilen Projekten durchaus Sinn macht, um einen Kunden von agiler Projektdurchführung überzeugen zu können. Aufgrund von klassischen Vergabeprozessen in Unternehmen werden auch agile Projekte in einen unpassenden herkömmlichen Festpreisvertrag gezwängt. Das kann gut gehen, diese Vertragsform ist allerdings keineswegs auf agile Methoden zugeschnitten und behindert agile Vorgehensweisen damit sogar eher. Im Rahmen dieser Ausarbeitung wurde daher der agile Festpreisvertrag vorgestellt, eine Alternative zum klassischen Festpreisvertrag. Er bietet den nötigen Kundeneinsatz, die Tranparenz und die Flexibilität, die ein agiles IT-Projekt braucht, um auch mit Festpreis erfolgreich umgesetzt werden zu können. [OGPM14]

Abbildungsverzeichnis

Tabellenverzeichnis

Literatur

[AHZ11] Muhammet Altindal, Frank Hinkel und Robert Zyla. Abgrenzung der agilen Softwareentwicklung von klassischen Verfahren. WinfWiki, 2011.

[BBvB+] Kent Beck, Mike Beedle, Arie van Bennekum, Alistair Cockburn, Ward Cunningham, Martin Fowler, James Grenning, Jim Highsmith, Andrew Hunt, Ron Jeffries, Jon Kern, Brian Marick, Robert C. Martin, Steve Mellor, Ken Schwaber, Jeff Sutherland und Dave Thomas. Prinzipien hinter dem Agilen Manifest. http://agilemanifesto.org/iso/de/principles.html (14:08, 09. Jan. 2017 (UTC+1)).

[BBvB+01] Kent Beck, Mike Beedle, Arie van Bennekum, Alistair Cockburn, Ward Cunningham, Martin Fowler, James Grenning, Jim Highsmith, Andrew Hunt, Ron Jeffries, Jon Kern, Brian Marick, Robert C. Martin, Steve Mellor, Ken Schwaber, Jeff Sutherland und Dave Thomas. Manifest für Agile Softwareentwicklung. http://agilemanifesto.org/ (14:00, 09. Jan. 2017 (UTC+1)), 2001.

[Bec13] Peter Beck. Wie passen Agilität und Festpreis zusammen? *Das Scrum Team*, 2013.

[Dub11] Daniel Dubbel. Money for nothing, change for free. Inspect&Adapt, August 2011.

[Glo16] Boris Gloger. Der agile Festpreis: Gemeinsamer Nenner für Dienstleister und Unternehmen. *Informatik Aktuell*, 2016.

[HW15] Shane Hastie und Stéphane Wojewoda. Standish Group 2015 Chaos Report - Q&A with Jennifer Lynch. *InfoQ*, Oktober 2015.

[Lin14] Oliver Linssen. Vorgehensmodelle für die betriebliche Anwendungsentwicklung. Gesellschaft für Informatik e.V., Juni 2014.

[OGPM14] Andreas Opelt, Boris Gloger, Wolfgang Pfarl und Ralf Mittermayr. *Der agile Festpreis - Leitfaden für wirklich erfolgreiche IT-Projekt-Verträge*. Carl Hanser Verlag München, 2014.

[Pre12] Christian Prehofer. B.2 Agile Methoden / Scrum. Vorlesung Methoden des Software Engineering an der Ludwig-Maximilians-Universität, 2012.

[Roy70] Winston W. Royce. Managing the Development of Large Software Systems. Proceedings, IEEE WESCON, August 1970.

[RW15] Stefan Roock und Henning Wolf. *Scrum - verstehen und erfolgreich einsetzen*. dpunkt.verlag, 2015.

[Sar16] Dr. Frank Sarre. Vorgehensmodelle. Vorlesung Juristisches IT-Projektmanagement an der Ludwig-Maximilians-Universität, 2016.

[Sei06] Siegfried Seibert. Das aktuelle Stichwort: V-Modell XT. *GPM-Magazin PMaktuell*, 2006.

[Som10] Ian Sommerville. *Software Engineering*. Pearson, 2010.

[SS16] Ken Schwaber und Jeff Sutherland. *The Scrum Guide - The Definitive Guide to Scrum: The Rules of the Game*, Juli 2016.

[WT04] Jonathan Weiss und Emel Tan. Klassische vs. agile Methoden der Softwareentwicklung. SWT Berlin, Ausarbeitung im Rahmen der Vorlesung Methoden und Werkzeuge zur Softwareproduktion, November 2004.