INSTITUT FÜR INFORMATIK
DER LUDWIG–MAXIMILIANS–UNIVERSITÄT MÜNCHEN

Ausarbeitung

Juristisches IT-Projektmanagement im Wintersemester 2017/2018

Vergütungsaspekte bei agilen IT-Projekten

Robert Keiselt

09. Januar 2018

Erklärung

Hiermit versichere ich, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

München, den 09.01.2018

R. Keiselt

_____

(Unterschrift des Kandidaten)

Inhaltsverzeichnis

Einleitung

Gerade für Wirtschaftsinformatiker spielt ein gutes Requirements Engineering eine entscheidende Rolle für den Projekterfolg. Da sich allerdings Anforderungen an IT Projekte meist nicht im Vorfeld abschließend bestimmen lassen, haben sich daher in den letzten Jahren agile Methoden wie u.a. Scrum ausgebildet und stark an Bedeutung gewonnen. Mit der Etablierung verschiedener agiler Softwareentwicklungsmethoden haben sich Besonderheiten für die Ausgestaltung der Verträge und das IT-Projektmanagement ergeben, insbesondere deswegen weil traditionelle Vertragsmodelle nicht ohne Anpassung auf agil geführte Projekte angewendet werden können. Im Rahmen dieser Ausarbeitung werden ausgewählte Entwicklungsmethoden, Vertragsarten bzw. Vergütungsmodelle vorgestellt und veranschaulicht. Der Fokus liegt dabei auf den Vergütungsaspekten in agilen IT-Projekten. Die flexibelste Lösung muss hierbei nicht immer die beste aller Alternativen darstellen.

Entwicklungsmethoden

Um die Softwareentwicklung zu optimieren finden verschiedenste Projektmanagementstandards Anwendung. Zu berücksichtigen wären die ICB der IPMA/GPM, Prince2 der OGC und das PMBOK des PMI. Auch das sog. Critical Chain Projektmanagement bietet weiteres Optimierungspotenzial [1]. Die grundsätzliche Unterscheidung, die eher auf Vorgehensmodelle abstellt, zielt allerdings auf klassische (wie z.B. das Wasserfall-Modell oder das V-Modell XT) und agilere Vorgehensweisen (wie z.B. Scrum, Extreme Programming, Feature Driven Development oder Kanban SD) der Softwareentwicklung ab.

Klassische Methoden

Bei klassischen Modellen wird die Konzeptionsphase klar von der Implementierungsphase getrennt. Diese zunächst logische Auftrennung funktioniert bei größeren Projekten traditionell nicht und löst Diskussionen aus, ob sog. Change Requests nicht schon im vereinbarten Leistungsumfang enthalten wären [2]. Strukturierte Vorgehensmethoden dienen vor allem dem Risikomanagement und der Qualitätssicherung von Entwicklungsprojekten. Die Phasen eines solchen Projekts gemäss dem Wasserfallmodell von Royce [3] wären Anforderungsanalyse, Grobplanung, Entwicklung des Systemdesigns gefolgt von der eigentlichen Implementierung. Nach einer (meist vernachlässigten) Testphase würde die Software in Betrieb genommen, gewartet und gegebenenfalls weiterentwickelt.

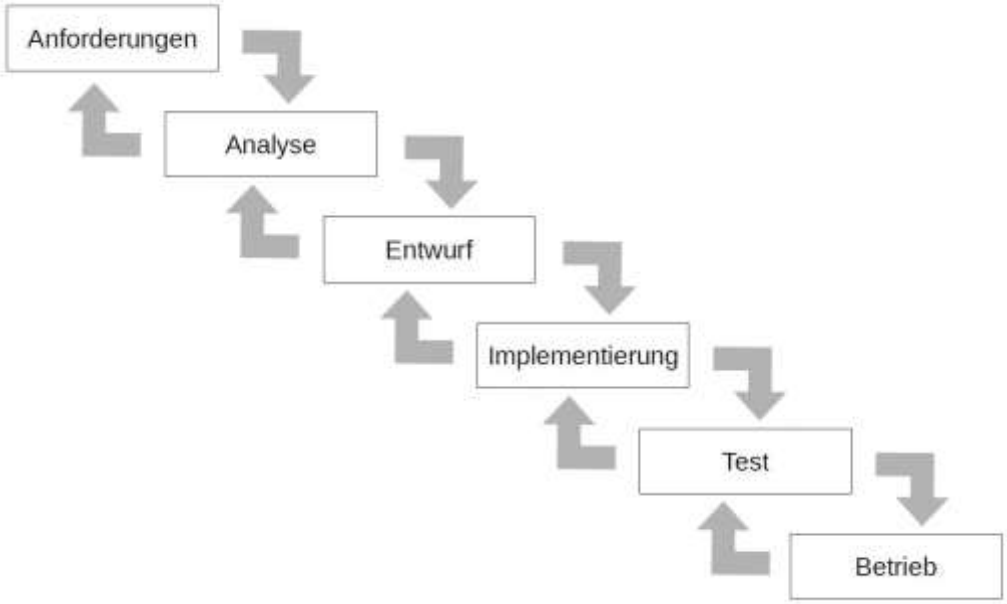

Abbildung 1: Wasserfallmodell nach W. Royce

Weiterentwicklungen wie das Spiralmodell [4] sehen Iterationen vor, d.h. die Software bzw. das IT-System wird hier in mehreren, nacheinander zu durchlaufenden Zyklen geplant, entwickelt, getestet und verbessert. Im V-Modell wird zusätzlich eine Korrelation von den Entwurfs- und Umsetzungs- zu den Test- und Abnahmephasen hergestellt. Dieses ursprüngliche von Boehm vorgeschlagene Modell wurde inzwischen ebenfalls weiterentwickelt. Daraus ist unter anderem das V-Modell XT entstanden, welches u.a. von den deutschen Bundesbehörden als Entwicklungsstandard für Systementwicklungsprojekte verwendet wird [5]. Die Ergebnisse der Realisierung, Systemintegration, Ablieferung sowie Abnahme werden gegenüber den Anforderungsdefinitionen und der Systemspezifikation validiert.

Auch das ITIL Release Management und das (mehr) im schweizerischen Bundesumfeld verwendete HERMES Modell beruhen letztlich auf dem Wasserfallmodell. HERMES unterscheidet grundsätzlich die Phasen Initialisierung, Konzeption, Realisierung und Einführung [6] und das sehr stark verbreitete ITIL Release Management gliedert sich in die Phasen Planung/Realisierung/Test/Produktivsetzung [7].

Wie bereits einleitend erwähnt ist die Anwendung von Phasenmodellen im Allgemeinen nur dann sinnvoll, wenn sich Anforderungen, Leistungen und Abläufe in der Planungsphase bereits relativ präzise beschreiben lassen. Bei komplexeren Vorhaben lassen sich aber Anforderungen kaum im Vorfeld definieren und gerade in länger dauernden Projekten entstehen aufgrund neuer technischer, organisatorischer oder auch (zunehmend) rechtlicher Entwicklungen neue Anforderungen während der Implementierungsphase. Hierfür sind modernere Vorgehensweisen besser geeignet und führen auch zu größeren Erfolgsaussichten, wie folgende Grafik nochmals deutlich veranschaulichen soll.

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

Tabelle 1: Chaos Report Agil versus Wasserfall

Agile Vorgehensweisen

Nicht nur aus o.g. Gründen haben in den letzten Jahren agile Methoden wie Scrum, Extreme Programming (XP), Feature Driven Development (FDD), Dynamic Systems Development Method (DSDM), Crystal Methods, Kanban und Lean Software Development stark an Bedeutung gewonnen. Diese in der Praxis tws. völlig unterschiedlichen Vorgehensmethoden resultieren aus einer fehlenden verbindlichen Definition von Agilität. Während der Fokus von XP auf der Vorgehensweise bei der Programmierung liegt, stellt Scrum ein Management Framework dar, das Rollen, Abläufe und Verantwortlichkeiten für Projekte festlegt. Agilität kann aber auch als eine Verbindung von Team Leadership Philosophie, Projekt Management Prozess und Engineering Best Practices charakterisiert werden [8]. Grundsätzlich wollen Agile Methoden den Softwareentwicklungsprozess (inklusive des Projekt- und Produktmanagements) unmittelbarer, effizienter, flexibler sowie schlanker gestalten und orientieren sich inhaltlich an Werten, die im sog. „Agilen Manifest“ [9] festgehalten wurden.

Im Zentrum dieses Manifests stehen vier Leitsätze:

1. **Individuen und Interaktionen sind wichtiger als Prozesse und Werkzeuge,**
2. **Funktionierende Software hat Vorrang vor umfassender Dokumentation,**
3. **Zusammenarbeit mit den Kunden ist wichtiger als Vertragsverhandlungen sowie**
4. **Reagieren auf Veränderung geht vor Befolgen eines Plans.**

Vorgeschlagen werden zudem insgesamt zwölf Prinzipien zur Operationalisierung dieser vier zentralen Werte für die agile Entwicklung. Die daraus abgeleiteten Praktiken machen konkrete Vorgaben für bestimmte Aspekte und betreffen beispielsweise die Aufwandschätzung, die Durchführung von Retrospektiven sowie die Testvorgehensweise im Sinne von Best Practices. Dazu gehören u.a. eine frühe und kontinuierliche Auslieferung von Software, die wirklich enge Zusammenarbeit von Fachbereichen und Softwareentwicklern, motivationsfördernde Maßnahmen und das Augenmerk auf technische Exzellenz sowie sich selbstorganisierende Teams (siehe Tabelle2).

1	Höchste Priorität ist es, den Kunden durch frühe und kontinuierliche Auslieferung wertvoller Software zufrieden zu stellen.
2	Anforderungsänderungen sind selbst spät in der Entwicklung willkommen. Agile Prozesse nutzen Veränderungen zum Wettbewerbsvorteil des Kunden.
3	Funktionierende Software soll regelmäßig innerhalb weniger Wochen oder Monate unter Bevorzugung der kürzeren Zeitspanne geliefert werden.
4	Fachexperten und Entwickler müssen während des Projektes täglich zusammenarbeiten.
5	Errichte Projekte rund um motivierte Individuen. Gib ihnen das Umfeld und die Unterstützung, die sie benötigen und vertraue darauf, dass sie die Aufgabe erledigen.
6	Die effizienteste und effektivste Methode, Informationen an und innerhalb eines Entwicklungsteams zu übermitteln, ist im Gespräch von Angesicht zu Angesicht.
7	Funktionierende Software ist das wichtigste Fortschrittsmaß.
8	Agile Prozesse fördern nachhaltige Entwicklung. Die Auftraggeber, Entwickler und Benutzer sollten ein gleichmäßiges Tempo auf unbegrenzte Zeit einhalten können.
9	Ständiges Augenmerk auf technische Exzellenz und gutes Design fördert Agilität.
10	Einfachheit – die Kunst, die Menge nicht getaner Arbeit zu maximieren – ist essenziell.
11	Selbstorganisierte Teams erstellen die besten Anforderungen, Architekturen und Entwürfe.
12	Das Team reflektiert regelmäßig wie es effektiver werden kann und passt sein Verhalten an.

Tabelle2: Prinzipen des Agilen Manifests

Die Grundmaximen und Prinzipien dürfen allerdings nicht überinterpretiert werden. Sie bedeuten keineswegs, dass Verträge, Spezifikationen oder auch Dokumentationen bei agilen Entwicklungen überflüssig sind. Tatsächlich müssen in Verträgen für agile Projekte viele Themen sogar eingehender geregelt werden. Folgende Aspekte müssen in einem agilen Vertrag unbedingt geregelt werden:
In der Vorgehensbeschreibung muss dargestellt werden welches Vorgehensmodell verwendet wird. Zusätzlich soll der Projektablauf skizziert und die Dauer von Sprints festgelegt werden. Im Bereich Vertragsgegenstand sollte u.a. definiert werden, wie die Zielgruppe, die Kundenbedürfnisse, die Produktattribute und die Wettbewerbsposition aussehen wird, damit für die Vertragsparteien die Chance besteht, kalkulierbare Anforderungen abzuleiten. Gerade in agilen Projekten ist die starke Mitwirkung des Auftraggebers emminent wichtig. Daher sollten die Begutachtung der User Stories, Teilnahme an Abnahmetreffen und Besprechungen sowie die Zusammenstellung von Mängellisten als Hauptpflichten definiert werden. Agile Projektmethoden verzichten grundsätzlich auf formale Dokumentation und Pflichtenhefte. In agilen Projekten findet die Auslieferung und die Prüfung von Zwischenergebnissen (z.B. im Rahmen von Sprint Reviews) idealerweise regelmäßig statt, daher sollte der Vertrag im Rahmen der Abnahme und Gewährleistung klarstellen, was nur eine reine Funktionsprüfung sein und was als Teilabnahme gelten soll, um nicht zuletzt die darauf aufbauende Frage der Gewährleistungsfrist zu klären. Da mit jedem erfolgreich abgeschlossenen Sprint ein Stück nutzbare Software entsteht ist insbesondere zu klären, welche Nutzungsrechte wem und zu welchem Zeitpunt bei vorzeitiger Beendigung des Projekts zustehen [10]. Nicht zuletzt müssen sehr eindeutige Regelungen zur Vergütung getroffen werden, dies setzt allerdings zunächst eine klare Einordung der zugrunde liegenden Vertragsmodalität voraus. Als adäquate Vertragstypen (insbesondere) für agile Softwareentwicklung werden vor allem der Kaufvertrag, Werkvertrag und Dienstvertrag diskutiert, wobei es zu beachten gilt, dass in Zeiten hoher Technisierung, schnelllebiger Unternehmensprozesse und ansteigender Komplexität Projekte meist anders als ursprünglich geplant enden, nicht kalkulierte Kostenüberschreitungen nicht unbedingt ungewöhnlich und sogar Komplettabbrüche nicht mehr auszuschließen sind. Agile Methoden versuchen zwar bezüglich unklarer Vertragsmodalitäten und unbekannter Abnahmekriterien Abhilfe zu schaffen, werfen allerdings Fragen bzgl. der schlüssigen Vertragsgestaltung insbesondere bei agiler Softwareentwicklung auf. Nachfolgend wird zunächst einmal auf die grundsätzlichen Möglichkeiten der Vertragsgestaltung und tws. damit verbundener Probleme eingegangen, um im Anschluss zum Thema Vergütung in agilen IT-Projekten überzuleiten.

Vertragsarten

Ausgangspunkt der agilen Entwicklung ist eine Vision des zukünftigen IT-Systems in Form einer priorisierten Liste von Funktionen, die jedoch noch nicht in der Tiefe detailliert sind. Diese werden im Fall von Scrum in Form von Userstories (einzelne Anwendungsfälle) formuliert und in Abstimmung mit dem Auftraggeber in einem Product Backlog (Anforderungsliste) nach ihrer Priorität sortiert. Der Auftraggeber kann die Liste der noch nicht umgesetzten Funktionalität jederzeit verändern und neu priorisieren. So können im Projektverlauf Veränderungen der Anforderungen und Erfahrungen der Anwender flexibel in die Entwicklung einfließen. Die Phase der initialen Analyse ist in agilen Projekten daher deutlich kürzer. Die Entwicklung kann schneller beginnen und der Auftraggeber erhält früher als bei klassischem Vorgehen eine lauffähige Version der Software. Nochmals zum Verständnis:

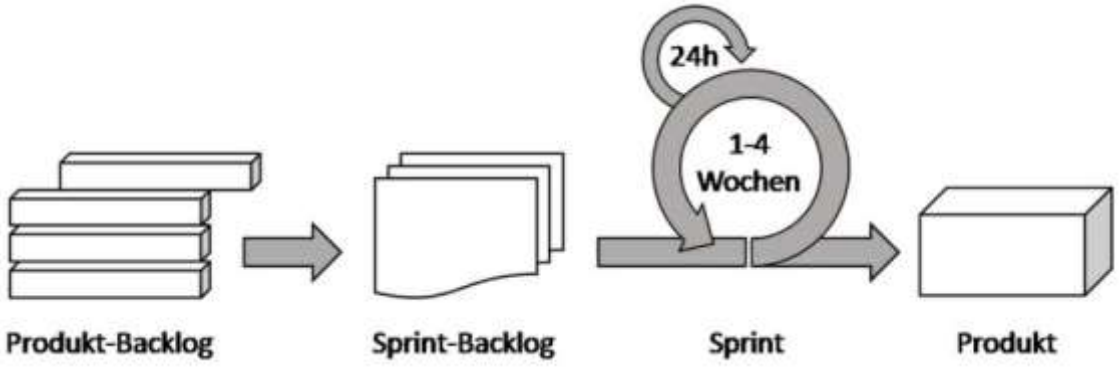

Abbildung 2: Der Scrum-Prozess

Vor Beginn eines Sprints werden vom Produktverantwortlichen (Product Owner) aus einem Produkt Backlog jene Anforderungen (Backlogitems) ausgewählt, die im nächsten Schritt umgesetzt werden sollen und in Abstimmung mit dem Entwicklerteam in das Sprintbacklog aufgenommen. In einem Sprint mit einer Dauer von wenigen Tagen bis zu einem Monat erfolgt dann die Umsetzung. Jeden Tag trifft sich das Entwicklerteam zum „Daily Scrum“ zwecks Synchronisation seiner Teammitglieder. Am Ende jedes Sprints steht das Produktinkrement, welches um die Funktionen der im erfolgten Sprint umgesetzten Backlogitems reicher ist [11]. Durch den in den letzten Jahren vermehrten Einsatz agiler Projektmethoden wie zum Beispiel Scrum kommt unter anderem die Frage auf, wie diese Art von Softwareprojekten in IT-Verträgen festgehalten werden soll. Zudem bestimmt der ausgewählte Vertragstyp, welche Gewährleistungsregelungen im Falle eines Mangels angewendet werden [12]. In den folgenden Abschnitten soll untersucht werden, welche Vertragstypen insbesondere für agile Softwareentwicklungsprojekte prinzipiell in Frage kommen ohne dabei die zuvor beschriebenen Eigenschaften agiler Projekte einzuschränken. Die Gestaltung von Verträgen für solche Projekte wurden in der Vergangenheit wenig bis gar nicht thematsiert und eine valide Rechtsprechung fehlt zumeist. In Frage kommen allerdings grundsätzlich Kauf-, Werk-, Dienst- ggfs. auch Rahmen- bzw. Projektvertrag. Weitere Gestaltungsoptionen können hier aber nur am Rande betrachtet werden.

Kauf-/Werkvertrag

Nach § 651 BGB unterliegt jeder Vertrag, der die Herstellung und Lieferung einer Sache vorsieht, dem Kaufrecht und den damit verbundenen Bestimmungen. Beispiele, in denen u.a. Kaufverträge Anwendung finden, sind Beschaffung von Hardware oder Kauf von Standardsoftware (zu den kaufrechtlichen Vereinbarungsmöglichkeiten siehe u.a. EVB-IT Kaufvertrag [13] etc.). Durch den Kaufvertrag (§433) wird der Verkäufer einer Sache verpflichtet, dem Käufer die Sache zu übergeben

und das Eigentum an der Sache zu verschaffen. Der Verkäufer hat dem Käufer die Sache frei von Sach- und Rechtsmängeln zu übergeben. Der Käufer ist verpflichtet, dem Verkäufer den vereinbarten Kaufpreis zu zahlen und die gekaufte Sache abzunehmen. Softwareprogrammen wird durch das OLG München die Sacheigenschaft aufgrund beweglicher Datenträger abgesprochen [14] und mit der Schuldrechtsmodernisierung (2002) wurden vermehrt kaufrechtliche Vorschriften auf Werkverträge angewendet (auf die Problematik von Werklieferungsverträgen wird hier nicht weiter eingegangen). Werkverträge finden hauptsächlich Anwendung bei der Neuerstellung von Software sowie bei der Anpassung von Softwaresystemen und auch bei der Wartung von Hardware oder Gutachten [15]. Verpflichtet sich der Unternehmer nur zur Beschaffung von Zutaten oder sonstigen Nebensachen, so finden ausschließlich die Vorschriften über den Werkvertrag Anwendung. Beim Werkvertrag schuldet der Dienstleister die Erstellung eines mangelfreien Werks, gemessen auf Grundlage einer konkreten Spezifikation, die vor Arbeitsbeginn einmalig vereinbart wurde. Die Vergütung erfolgt (meist) über einen Festpreis und Änderungen an oder der Austausch von Features erfolgt im Rahmen von Change Requests mit Preissteigerungen auf der Basis von Time and Materials. Der klassische Werkvertrag sieht somit eine gewisse Kostensicherheit bei gleichbleibender Spezifikation, die Verantwortung des Auftragnehmers bei Mängeln und dadurch eine Vorhersehbarkeit für die Vertragsparteien vor. Dies bedeutet zugleich, dass Änderungen stets Nachverhandlungen erfordern, immer mit zunächst nicht einkalkulierten Kosten verbunden sind und es bedingt Anreize für den Auftragnehmer gibt, über egal welche Verbesserungen im Sinne des Auftraggebers nachzudenken. Der klassische Werkvertrag ist in der Gesamtbetrachtung primär für einfache Projekte in unveränderlicher Umgebung geeignet [16]. Zu beachten gilt, dass der Auftragnehmer die Erfolgsverantwortung, das gesamte unternehmerische Risiko trägt und auch die Projektleitung übernimmt. Wird allerdings durch den Auftraggeber die Erfolgsverantwortung übernommen, der Auftragnehmer in seinen Freiheiten zu sehr eingeschränkt oder holt sich der Auftraggeber gar die Projektleitung (nachträglich) zurück, so handelt es sich nicht mehr um einen Werkvertrag, egal wie der Vertragstitel ursprünglich vereinbart wurde. Des Weiteren muss beachtet werden, dass der Gefahrübergang vom Auftragnehmer zum Auftraggeber erst nach der erfolgreichen Abnahme des fertiggestellten Werks durch den Auftraggeber stattfindet und erst danach die Beweislast auf den Auftraggeber übergeht. Das heißt, er hat dann zu beweisen, dass ein Mangel vorliegt, der Mängelansprüche nach sich ziehen kann. Liegt, wie in der Regel bei (fast allen) agilen Projekten der Fall, kein Pflichtenheft vor, wird bei einem Werkvertrag laut Koch ein mittlerer Ausführungsstandard geschuldet [17]. Auch wenn es anscheinend noch keine wirklich verwertbare Rechtsprechung zu agilen Softwareprojekten zu geben scheint, wird hier vom Autor durchaus die Auffassung vertreten, dass ein einzelner Sprint (wie in Abbildung 2 bereits veranschaulicht) als Mini-Werk angesehen werden kann. Zusammengefasst soll das heißen, dass auch die Vergütung also erst nach erfolgreicher Abnahme fällig wird und (meist) ein Festpreis ist (dazu allerdings erst später).

ANÜb-/Dienstvertrag

Der Dienstvertrag scheint eher auf die Bedürfnisse der agilen Projektleitung zugeschnitten zu sein. Hier schuldet der Dienstleister kein definiertes Ergebnis (Werk), sondern nur seine Arbeitsleistung in Form der Tätigkeit. Dadurch verschiebt sich das Erfolgsrisiko zu Lasten des Auftraggebers. Innerhalb des Projekts würde die Software in sog. Sprints entwickelt, bei denen jeweils einzelne Userstories entwickelt und implementiert werden. Die Vergütung erfolgt nach Aufwand (Time and Materials). Dadurch bietet der Dienstvertrag eine völlige Flexibilität im Hinblick auf den Vertragsgegenstand und eine gewisse wirtschaftliche Sicherheit für den Auftragnehmer. Die Projektkosten können für den Auftraggeber jedoch nicht abschließend bestimmt werden, und so bleiben Zeit sowie Kosten größtenteils unvorhersehbar. Die Mängelbeseitigung durch den Auftragnehmer führt unweigerlich zu Zusatzkosten für den Auftraggeber. Im Ergebnis ist der Dienstvertrag eher für kleinere Projekte

wie für die Entwicklung von Einzelkomponenten geeignet, die im Rahmen einer bereits eingespielten Zusammenarbeit verwirklicht werden soll. Neuverträge dürften eher schwer verhandelbar sein, da die Erfolgsverantwortung beim Auftraggeber liegt und daher weniger Bereitschaft zum Abschluss solcher Verträge liegen dürfte. Hinzu kommt die (gerade für Berater) relevante Verschärfung der Scheinselbständigkeitsregelungen zum 01.04.2017, die sogar strafrechtlichen Charakter im Bereich Sozialabgaben haben kann. Des Weiteren heilt eine vorhandene Lizenz zur Arbeitnehmerüberlassung nicht mehr wie bisher (versehentlich) falsch vereinbarte Verträge und wird ebenfalls uninteressanter.

Rahmen-/Projektvertrag

Einen Kompromiss versucht der Rahmenvertrag mit einzelnen, untergeordneten Werkverträgen. Hierbei wird die Beziehung von Auftragnehmer und Auftraggeber einmalig geregelt und im Anschluß für jeden Sprint ein präziser Scope mit umzusetzenden Userstories (meist) zu einem festen Preis nach werkvertraglichem Charakter vereinbart. Die Mängelbeseitigung erfolgt grundsätzlich nach jeder Teilabnahme im Rahmen der gesetzlichen Gewährleistung. Dieses Modell erreicht eine sehr hohe Flexibilität ohne die Kostensicherheit bei gleichbleibender Spezifikation zu gefährden und bietet durch die Teilabnahmen und die gesetzliche Gewährleistung einen hervorragenden Ausgleich in der Qualitätssicherung. Jedoch ist jeder Sprint mit einem Verhandlungsaufwand verbunden und die eigentlich beabsichtigte Agilität (also Änderungen im Scope) führen zu Kostenunsicherheiten. Der Rahmenvertrag mit einzelnen Werkverträgen ist demnach besonders gut für komplizierte Projekte geeignet, an denen ohne Zeitdruck gearbeitet werden kann. Beim IT-Projektvertrag handelt es sich im Grunde nicht um einen tatsächlich gesetzlich geregelten Vertragstyp. Er findet oft bei komplexen Langzeitprojekten Anwendung, wobei im Pflichtenheft lediglich eine grobe Spezifikation vorliegt und der Auftragnehmer typischerweise das Festpreisrisiko trägt. Da ein IT-Projekt aber oft aus mehreren unterschiedlichen Teilleistungen, wie beispielsweise der Softwareerstellung und der Schulung von Mitarbeitern besteht, stellt sich die Frage, welches gesetzlich geregelte Vertragsrecht hierbei Geltung findet. Ist der Gesamtvertrag nicht wirklich in mehrere gleichwertige Leistungen auftrennbar, so wird die Absorptionstheorie auf den Gesamtvertrag angewendet (das Recht der Hauptleistung gilt) [18]. Da die wenigsten Projekte (mit Ausnahme der Forschung) eine gemeinsame Vermarktung vorsehen, wird hier von gesellschaftlichen Konstruktionen (und von Lizenzvertragsmodalitäten) abstrahiert.

Die vorgestellten Möglichkeiten unterscheiden sich in vielerlei Hinsicht, aber auch in Bezug auf die Fälligkeit der Vergütung. So ist die Vergütung bei Kaufverträgen mit der Übereignung der Ware fällig, während diese bei einem Werkvertrag mit der Abnahme der Ware und bei einem Dienstvertrag mit Ableistung des Dienstes fällig wird. Des Weiteren können beim Werkvertragsrecht auch Abschlags-zahlungen vereinbart werden. Siehe also hierzu nachfolgende Auswahl von Vergütungsmodellen.

Vergütungsmodelle

Nach der Vorstellung von verschiedenen Entwicklungsmethoden und von möglichen Vertragsarten geht es nun um die Untersuchung von unterschiedlichen relevanten Vergütungsmodelle. So besteht beispielsweise die Möglichkeit die Entwicklung einer Software nach Aufwand, analog wie zu einem Dienstvertrag oder Werkvertrag variabel, oder aber analog zu einem Kaufvertrag oder Werkvertrag zu einem Festpreis zu vergüten. Da besonders bei der agilen Softwareentwicklung auf Grund des zu Beginn nicht vollständig bekannten Umfangs, die Kosten nicht abschätzbar sein können, bieten sich auch verschiedene Mischformen an. Hierbei entsteht insbesondere die Frage, was geschieht, wenn das ursprünglich vereinbarte Budget aufgebraucht ist. Zu diesen und ähnlichen Poblemen später.

Budgetlimit/Festpreis

Eine mögliche Art der Vergütung für agile Softwareentwicklungsprojekte ist die Vereinbarung eines festen Budgets, da bei agilen Softwareentwicklungsprojekten nach jedem Entwicklungszyklus eine Art lauffähiges Produkt verfügbar sein sollte. Hierfür eignet sich z.B. ein Werkvertrag mit Fixpreis. Wie im Agilen Manifest gefordert, wird entwickelt, bis die Budgetgrenze erreicht wird. Nicht mehr umsetzbare Aufgaben aus dem Backlog werden bei diesem Auftragsmodell nicht in die Software mit aufgenommen, also nicht implementiert, sondern werden in einer weiteren Beauftragung umgesetzt.

Fixed Price, Fixed Scope

Ähnlich zum vorher beschriebenen limitierten Budget sind bei dem sog. Fixed Price bzw. Fixed Scope der Preis bzw. die Ziele festgeschrieben. Als Vertragsform eignen sich hierbei sowohl der Kaufvertrag als auch der Werkvertrag mit Fixpreis (wie bereits in vorigen Kapiteln angedeutet). Hervorzuheben ist hierbei, dass bereits zu Beginn ein gewisses Anforderungspaket mit einer Art tatsächlichem Festpreis vereinbart wird. Allerdings liegt durch diese Vorabvereinbarung des Preises das volle Projektrisiko beim Auftragnehmer. Als weitere Probleme sind die bei agilen Projekten häufigen Änderungen an den Projektanforderungen zu nennen. So kann es bei unzureichender Definition der Anforderungen oder einem zu niedrigen Angebot schnell zu (unlösbaren) Schwierigkeiten zwischen Auftraggeber und Auftragnehmer kommen und damit auch zu einer in Zukunft vergifteten Kundenbeziehung führen.

Vergütung nach Aufwand

Als Alternative zur Vergütung mit einem fixen Budget ist es möglich nach angefallenem Aufwand zu vergüten. Hierbei könnte man sowohl einen Werkvertrag, als auch einen Dienstvertrag nutzen [19]. Das entscheidende Problem ist dabei (wie bereits angedeutet), dass der Kunde während des Projekts kaum oder gar keinen Überblick über den Endumfang des Produktes bzw. die anfallenden Kosten hat.

Fixed Profit als Alternative

Im Sinne der im Agilen Manifest geforderten Kundennähe bietet es sich an, wenn das Budget, wie oben beschrieben, erschöpft ist, eine Vergütung nach dem Fixed Profit Modell anzusetzen. Hierbei werden nach Erreichen des Budgets nur noch kostendeckende Stundensätze abgerechnet, so dass der Auftragnehmer mit dem Projekt einen festen Gewinn gemacht und der Kunde absehbare Kosten hat [20]. Hierfür eignet sich ein Werkvertrag mit Fixpreis (wie bereits oben beschrieben) besonders. Das Risiko wird bei diesem Vergütungsmodell zwischen Auftragnehmer und Auftraggeber geteilt. Wird das Projekt vorzeitig fertiggestellt, sinken die Kosten für den Auftraggeber, wird das Projekt überzogen, zahlt der Kunde zwar mehr, der Auftragnehmer macht jedoch keinen weiteren Gewinn. Man möchte fast von einer Art Win Win Situation sprechen, muss aber nun auch nicht übertreiben.

Phasenbasierte Beauftragung

Besonders für agile Softwareentwicklungsprojekte eignet sich auch die phasenbasierte Beauftragung, da zum Ende jeder Phase ein lauffähiges Produkt existieren sollte. Für den Kunden bietet diese Art der Vergütung den Vorteil, dass das Risiko durch die Aufteilung in einzelne Phasen reduziert wird. Obwohl der Kunde bei dieser Vertragsart einen höheren Einfluss auf die Entwicklung hat, ist es auch möglich hier einen Werkvertrag zu nutzen. Die Form des Werkvertrags empfiehlt sich ferner auch, da

so eine Abnahme des Phasenprodukts durch den Auftraggeber erfolgt und so für jede neue Beauftragung die Anforderungen angepasst werden können. Phasenbeauftragung wird oft bei Inhouseentwicklung mit Gesamtbudget und stundengenauer Abrechnung verwendet. Das nötige Fachpersonal kann dann extern zugebucht, allerdings aus Budgetgründen auch jederzeit wieder ausgestellt werden. Zusätzlich zur Flexibilität behält der Auftraggeber die Kontrolle über das Projekt.

Probleme und Empfehlung

Es gibt eine Vielzahl von Problemen, die bereits bei der Erstellung eines Lastenhefts durch die Seite der Auftraggeber beginnt, denn dies unterstellt eine größere Kompetenz, die Fachlinienmitarbeiter nicht unbedingt leisten können. Weitere immer wieder auftretende Probleme sind z.B. Budget- bzw. Terminüberschreitungen, wobei es immer verschiedene Lösungsmöglichkeiten gibt. Zum einen kann der Kunde eine Erhöhung des Budgets genehmigen, so dass doch noch eine lauffähige Software geliefert werden kann, allerdings müßte er dann die Mehrkosten tragen. Will der Kunde keinesfalls die Mehrkosten übernehmen, muss diese der Auftragnehmer als Risikoträger eines Werkvertrags tragen. Erfolgt keine Lieferung, so ist er ggfs. gegenüber dem Auftraggeber schadensersatzpflichtig. Auftretende Verzögerungen, z.B.Terminüberschreitung bei agilen Entwicklungsmethoden, sind in Scrum relativ einfach erkennbar und so ist es grundsätzlich möglich frühzeitig geeignete Strategien als Gegenmaßnahmen zu entwickeln. Da nach jedem Entwicklungszyklus eine lauffähige Software existiert, kann dem Kunden somit bereits vor dem eigentlichen Liefertermin ein Prototyp übergeben werden, so dass Mitarbeiter geschult werden können und Zwischenabnahmen möglich bleiben. Zum Liefertermin wird dem Kunden das zu diesem Zeitpunkt aktuelle Release zugestellt und im weiteren Verlauf können dem Kunden alle Zwischenstände bis zum endgültigen Release übergeben werden, so dass durch den Verzug ein möglichst geringer Schaden entsteht. Vertragsrechtlich wäre vermutlich der Auftragnehmer trotzdem schadenersatzpflichtig, zusätzlich bleiben Konventionalstrafen möglich. Bei Mangelware kommt es darauf an, ob der Auftraggeber die Ware bereits abgenommen hat oder ob die Abnahme noch aussteht. Wenn die Abnahme noch nicht stattfand, kann der Auftraggeber die Mängel feststellen und die Abnahme verweigern. In diesem Fall muss der Lieferant, um Anspruch auf die Vergütung zu haben, nachbessern und eine neue Abnahme vereinbaren. Für den Fall, dass die Abnahme bereits erfolgte und der monierte Mangel nicht ersichtlich oder bekannt war, so kann vom Abnahmedatum an (aber innerhalb der Verjährungsfrist gemäss § 634a II BGB) der Mangel beim Auftragnehmer gemeldet und deren Behebung gefordert werden. War der Mangel bei Abnahme bekannt und nicht von der Abnahme ausgeschlossen, so kann er nicht mehr nachträglich bemängelt, sondern, die Behebung muss gesondert beauftragt werden. Beim Kaufvertrag kann es zu Problemen kommen, wenn der Lieferant den Vertrag nicht ordnungsgemäss einhalten kann. In diesem Fall hat der Auftraggeber das Recht auf Rücktritt oder Schadensersatz. Ist die übereignete Ware mangelhaft, so hat der Auftraggeber das Recht auf Nachbesserung oder Neulieferung. Sollte dies fehlschlagen, so besteht nach mehrmaliger Nachbesserung das Recht vom Vertrag zurückzutreten. Für Projekte, in denen eine Änderung der Anforderungen nicht erfolgt, bietet sich die Nutzung von sog. Straf- und Bonusklauseln an. Hierbei erhält der Lieferant einen Bonus, wenn er vor der Deadline fertigstellt, und zahlt Strafe, wenn er verspätet abliefert. Das Problem und die Gefahr bei Nutzung derartiger Klauseln ist jedoch, dass Änderungswünsche durch den Auftragnehmer nur sehr schwer angenommen werden können und dass die Anforderungen bereits zu Beginn möglichst detailgetreu definiert sein müssen. Die bisher ausgeführten Problemstellungen ergeben sich (wie aufgezeigt) aus der Vertragsgestaltung. Aus Vergütungssicht stellt sich eher das Problem, inwiefern Vergütungsmodelle mit den vorliegenden Vertragsmodalitäten verknüpft werden können, ohne die tatsächlichen Anforderungen zu kennen und den damit eigentlich verbundenen Aufwand einschätzen zu können. Alternative Modelle wollen das Risiko auf die Parteien ausgewogen verteilen und bedienen sich entweder der Einschätzung

mittels Storypoints oder verwenden z.B. die Function Point Analyse, worauf abschließend (allerdings ohne Praxisbeispiel) einzugehen ist. In Scrum gibt es zur Schätzung des Aufwandes z.B. das Planning Poker, wobei die einzelnen Teammitglieder rasch und völlig unabhängig voneinander Schätzungen vornehmen und sich in einem interaktiven Prozess auf gemeinsame Bewertungen verständigen [21]. Der Ausgangspunkt ist die Bewertung von Grösse, Aufwand und Entwicklungsrisiken der einzelnen Produktfunktionen mit Storypoints (ggfs. verbunden mit einem Stückpreis und Bezahlung eines Fixbetrags pro umgesetzten Storypoint). Für das Gesamtprojekt ergibt sich unter anderem aus der Anzahl der Stoypoints und dem Preis pro Storypoint ein Zeit- und Kostenrahmen, welcher vertraglich relativ gut im Vorfeld verbindlich fixiert werden kann. Die andere oben angesprochene Möglichkeit zu vereinbarende Preise besser einschätzen zu können ist die Function Point Analyse, mittels der versucht wird über den Umfang des Sourcecodes (Menge und Qualität sowie Produktivität, also das Verhältnis zwischen Ergebnis und Aufwand bei der Softwareentwicklung) das Finanzbudget besser regeln zu können. Durch die Ermittlung der Größe und Schwere von Anwendungen mittels Function Points (elementarer Prozess, Funktion oder UML Systemanwendungsfall) lassen sich auch klassische Fehleinschätzungen vermeiden. Im Februar 2013 veröffentlichten die Object Management Group (OMG) und das Consortium for IT Automated Function Points (AFP), um die vorher gängige manuelle Zählung von Function Points zu ersetzen und Softwareentwicklern sowie IT-Abteilungen (z.B. über die Application Intelligence Platform von CAST Software) ein leistungsstarkes Werkzeug an die Hand zu geben, mit dem sie die Qualität von Anwendungen und die Produktivität von Softwareentwicklungs-projekten messen/optimieren, sogar ineffiziente (Änderungs-)Prozesse identifizieren können [22].

Nach diesen Ausführungen zu Entwicklungsmethoden, Vertragsarten und Vergütungsmodellen soll abschließend noch eine Empfehlung für die Vertragsgestaltung abgegeben werden, die allerdings aufgrund der Schwierigkeit der ausgeführten Themen kaum als eine allgemeingültige Empfehlung angesehen werden kann. Für (gerade agile) Projekte wäre es (wie oben ausgeführt) immens wichtig eine passende Vertragsmodalität zu wählen und nahezu alle kostentechnischen Auswirkungen bzw. alle möglichen Risiken im Rahmen der Vergütung theoretisch bereits bei der Vertragsgestaltung mit einzukalkulieren. Bedauerlicherweise schafft auch die Einordnung des Projekts als reiner Werkvertrag Probleme. Bei Vertragsschluss stehen die Eigenschaften der zu erstellenden Software noch nicht hinreichend konkret fest, so kann im Prinzip weder eine Vereinbarung zur Beschaffenheit noch eine Vergütungsvereinbarung getroffen werden. Teilweise wird in der Literatur deshalb vorgeschlagen, einen Rahmenvertrag über die Grundlagen der Zusammenarbeit zu schließen. Dieser Vertrag sieht dann sowohl dienstvertraglich ausgestaltete Reglungen über die Zusammenarbeit der Parteien vor, als auch den gesonderten Abschluss von Teilprojektverträgen, die dann zur Realisierung einzelner Abschnitte als gesondertes Teilprojekt mit werkvertraglichen Elementen ausgestaltet sind. In diesen Bestimmungen über die Zusammenarbeit im Allgemeinen wird das von beiden Parteien gewünschte Vorgehensmodell so genau wie möglich vereinbart. Entscheidend sind sogenannte Sollbruchstellen und die Vereinbarung einer eindeutigen Exitstrategie, um vorhersehbaren Problemen schon im Vorfeld zu begegnen [23]. Dies hat durchaus Charme, geendet wird allerdings mit einer Empfehlung für agile Projekte, die grundsätzlich auf einen Dienstvertrag mit eindeutig definierten Prozessen und sog. Sollbruchstellen hinauslaufen soll. Die Risikoverschiebung des konventionellen Dienstvertrags zu Lasten des Auftraggebers wird durch dieses Konzept mit den entsprechend vertraglichen Regelungen aufgefangen, die bessere Kontroll- und Steuerungsmöglichkeiten bieten und die auch eine frühzeitige Erkennung bzw. eine effektive Lösung von Problemen ermöglichen. Dies verspricht allerdings nicht nur die vollständige Ausnutzung von Verbesserungs- und Einsparpotentialen, sondern hält im Zweifel Sicherheitsmaßnahmen bereit, um durch Sollbruchstellen und ein geordnetes Exit Management eine einvernehmliche Trennung in beiderseitigem Interesse zu ermöglichen, was alle Parteien begrüßen.

Zusammenfassung

Die Schwierigkeit Projekte durch bessere Vorgehensweisen erfolgreich mit klaren Vertragsstrukturen und damit verbundenen Regeln zur Vergütung zu verknüpfen, wurde versucht in dieser Ausarbeitung darzustellen. Agile Projekte bestimmen zunehmend den Alltag der Softwareentwicklung auch wenn die juristischen Gestaltungen der notwendigen Projekt- bzw. Vergütungsverträge noch nicht den wirklich perfekten Standard bereithalten. Der Dienstvertrag bei agilen Projekten ist nicht zwingend, werkvertragliche Regelungen, die den Wünschen der Auftraggeber eher entsprechen, bleiben n.w.v. möglich. Auf die oben ausgeführten Aspekte sollte insbesondere aus der Sicht von Auftragnehmern Rücksicht genommen werden. Die angesprochene Problematik kleiner Auftragnehmer erscheint auch durch eine vermeintlich richtige Vertragsgestaltung und flexiblere Vergütungsmodelle weiterhin ungelöst. Wie in dieser Ausarbeitung angedeutet wird es trotzdem immer wichtiger, sich aktiv mit der Wahl des geeigneten Vertrags zu beschäftigen. Es gibt, wie in dieser Arbeit vorgestellt, gemäß BGB überschaubare Arten von Verträgen, jedoch lassen sich diese in Ihren eigenen Beschaffenheiten aneinander annähern, oder durch Rahmenverträge miteinander kombinieren. IT-Projekte sind in ihrer Art, Beschaffenheit und Anforderungen derart heterogen, dass jedes IT-Projekt individuell zu betrachten ist und es daher keine Universallösung für die perfekte Projektgestaltung oder für den perfekten Vertrag gibt. Entscheidend für ein gutes Projektergebnis ist das Miteinander zwischen Auftragnehmer und Auftraggeber. So müssen beide Parteien eine gute gemeinsame Basis und ein Bewusstsein für Form und Umfang des Endprodukts bilden. Zielführend hierbei ist die Kollaboration beider Parteien in Form von beidseitigen Feedback und Vertrauen. Alle wesentlichen Probleme sollten idealerweise durch geschickte vertragliche Regelungen (gerade was Kostenauswirkungen durch wiederkehrende Projektrisiken angeht) quasi vorhergesehen werden. Auch wenn dies nicht in allen Fällen möglich erscheint, gibt diese Ausarbeitung einige Hinweise für die Vertragsgestaltung, die gängigen Vergütungsmodelle und weist auf Problemstellungen hin, die es u.a.U. zu beachten gilt.

Abbildungsverzeichnis

Literaturverzeichnis

[1] http://www.amendos.de/fachartikel-whitepaper- download/klassisches versus agiles it projektmanagement die wahl der richtigen vorgehensweise.html abgerufen am 20.11.2017

[2] Siehe dazu auch den Überblick bei Reichert Jörg, Vertragsfreiheit und agile Softwareentwicklung: vertragstypologische Einordnung agiler Softwareentwicklungsverträge, Saarbrücken 2013, S. 15ff;

[3] Royce Winston W., Managing the Development of Large Software Systems, in: Proceedings of IEEE Western Electronic Show and Convention, Los Angeles 1970, S. 1–9.

[4] Boehm Barry, Guidelines for Verifying and Validation Software Requirements and Design Specifications, in: Samet Jonathan (Hrsg.), Proceedings of Euro IFIP 79, Amsterdam 1979, S. 711–719, zit. Boehm, Guidelines for Verifying and Validation, S. 711ff.

[5] Siehe zur Einordnung von Phasenmodellen in den Beschaffungsablauf auch Schreiber Josef, Beschaffung von Informatikmitteln: Submissionsverfahren, Pflichtenheft, Evaluation, 5. A., Bern 2015, S. 20ff.

[6] Informatiksteuerungsorgan des Bundes ISB (Hrsg.), HERMES 5.1 Referenzhandbuch, 2.A., Bern 2015, http://www.isb.admin.ch/themen/methoden/01661/01662/index.html?lang=de, S. 5ff.

[7] Hoppen Peter/Victor Frank, ITIL – Die IT Infrastructure Library – Möglichkeiten, Nutzen und Anwendungsfälle in IT-Verträgen, CR 2008, S. 199–204, S. 201ff.

[8] https://planit.legal/blog/de/agile-projekte-agile-vertraege/ abgerufen am 20.11.2017

[9] Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., et al. Manifesto for agile software development, 2001. URL: http://agilemanifesto.org abgerufen am 08.01.2017

[10] Pichler, R., Scrum: agiles Projektmanagement erfolgreich einsetzen, dpunkt. Verlag, 2013.

[11] Highsmith, Jim, Agile Software Development Ecosystems. Addison-Wesley, 2002.

[12] Thewalt, S. Der Softwareerstellungsvertrag nach der Schuldrechtsreform: Rechtsnatur,Leistungsbestimmung und Mängelhaftung. Rhombos-Verlag, 2004.

[13] http://www.evb-it.de/pages/frame_a.html abgerufen am 05.01.2018

[14] Witte, A. Agiles Programmieren und § 651 BGB. ITRB 2010, 44.

[15] Sarre, F., Juristisches IT-Projektmanagement. München, Ludwig-Maximilians-Universität, 2014.

[16] Wirtschaftslexikon, Gabler: Werkvertrag, 2016: http://wirtschaftslexikon.gabler.de/Archiv/3791/werkvertrag-v7.html abgerufen am 23.12.2017

[17] Koch, F., Agile Softwareentwicklung - Dokumentation, Qualitätssicherung und Kundenmitwirkung. ITRB 2010, 114.

[18] Sprau: Palandt - Bürgerliches Gesetzbuch, 2009.

[19] Söbbing, T., Agile Projekte in der IT-rechtlichen Praxis. Der IT-Rechtsberater,2014, Verlag Dr. Otto Schmidt, 214-219.

[20] Opelt, A., Gloger, B., Pfarl, W., und Mittermayr, R. Der agile Festpreis: Leitfaden für wirklich erfolgreiche IT-Projekt-Verträge. Carl Hanser Verlag GmbH Co KG, 2014.

[21] Schwaber, K., Sutherland, J., The Scrum Guide, Version 1.2, 2013, online verfügbar unter http://www.scrumguides.org/docs/scrumguide/v1/Scrum-Guide-US.pdf#zoom=100 (deutsche Übersetzung: http://www.scrumguides.org/docs/scrumguide/v1/Scrum-Guide-DE.pdf), S. 3ff;

[22] https://www.heise.de/developer/artikel/Standard-fuer-automatisierte-Function-Point-Analyse-2067044.html abgerufen am 22.11.2017

[23] http://blog-it-recht.de/2016/08/11/vertragsgestaltung-bei-agiler-softwareerstellung-teil/ abgerufen am 21.11.217

Tabellenverzeichnis