MASTERARBEIT

PROZESSMODELL ZUR AUFWANDSSCHÄTZUNG AGILER SOFTWAREPROJEKTE

ausgeführt am

Studiengang
Informationstechnologien und Wirtschaftsinformatik

Von: Florian Stoppacher, BSc MA
Personenkennzeichen: 1610320009

Graz, am 12. Juli 2019 .....

Unterschrift

EHRENWÖRTLICHE ERKLÄRUNG

Ich erkläre ehrenwörtlich, dass ich die vorliegende Arbeit selbstständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen nicht benützt und die benutzten Quellen wörtlich zitiert sowie inhaltlich entnommene Stellen als solche kenntlich gemacht habe.

.....

Unterschrift

DANKSAGUNG

An dieser Stelle möchte ich mich bei allen bedanken, die mich durch ihre fachliche und persönliche Unterstützung beim Schreiben dieser Masterarbeit unterstützt haben.

Mein Dank gilt meinem Betreuer Dr. Michael Amann-Langeder, der mich bei der Themenfindung unterstützt und in weiterer Folge fachlich betreut hat. Des Weiteren möchte ich mich bei Pankl Racing Systems AG und insbesondere bei DI (FH) Birgit Thek für die Mitwirkung in Interviews bedanken.

Besonders möchte ich bei meiner Familie bedanken, ohne ihre Unterstützung und ihr Verständnis wäre es nicht möglich gewesen, die vorliegende Masterarbeit zu schreiben.

KURZFASSUNG

Agile Softwareprojekte unterliegen den Fixpunkten Kosten und Termin sowie der Variablen des Umfanges der Software. In dieser Arbeit wird beleuchtet wie der zeitliche Aufwand des Umfanges geschätzt werden kann. Die Forschungsfrage dient der Ergründung von Einflussfaktoren und der Erstellung eines standardisierten Prozessmodells. In diesem Zusammenhang wird einerseits erläutert, welche Methoden der agilen Softwareentwicklung existieren und welche Charakteristiken sie haben. Die Methoden der Aufwandsschätzung unterliegen drei verschiedenen Paradigmen. Die Paradigmen sind die Aufwandsschätzungen basierend auf Experten, basierend auf Daten und das hybride Paradigma. Das hybride Paradigma vereint die Vorteile von Aufwandsschätzungen basierend auf Expertenwissen und historischer Daten. Zur Erstellung eines standardisierten Prozessmodelles werden anhand einer Literaturrecherche Einflussfaktoren auf die Aufwandsschätzung ermittelt und anschließend in das Prozessmodell überführt. Es werden unter anderem humane Faktoren, Einsatz neuer Technologien, die Einheit der Aufwandsschätzung und der Testumfang in der Arbeit diskutiert. Diese Einflussfaktoren werden in das Prozessmodell als Eingangsgrößen übernommen, wobei zunächst immer geprüft wird, ob eine ähnliche Softwareanforderung bereits implementiert wurde und damit die Möglichkeit zur Wiederverwendung bestehender Softwaremodule besteht. Das Prozessmodell wird im Unternehmen Pankl Racing Systems AG experimentell eingesetzt und durch Experteninterviews evaluiert. Die Erkenntnisse des Experteninterviews werden in Form eines Fragebogens, der bei der Anwendung des Prozessmodells unterstützt, angewendet. Eine weitere Erkenntnis des Experteninterviews stellt dar, dass die Aufwandsschätzung durch den Einsatz eines standardisierten Prozessmodells verbessert ist. Der erstellte Fragebogen besitzt einen modularen Aufbau, wodurch er von anderen Unternehmen als dem Beispielunternehmen verwendet und adaptiert werden kann.

ABSTRACT

The characteristic of agile software projects is that they have fixed costs and deadline but can vary in scope. This thesis helps assess the estimated time to implement the project. The research question investigates the factors influencing this assessment and the creation of a standardized process model. Within this context, agile methods for software development and their characteristics are described. The methods to estimate software projects are categorized into three classes: expert-driven, data-driven, and hybrid. The hybrid class unites the advantages of expert knowledge and historical data. The standardized process model builds on literature research to determine the key influence factors. Human factors, the usage of new technologies, estimation metrics, and the test scope of the software project are discussed. The influence factors are transferred into the process model as input parameters. The first process step is to determine whether any similar software requests have already been solved. The process model is being trialed by Pankl Racing Systems AG and is evaluated in expert interviews. The expert interviews are built on a questionnaire to support the usage of the standardized process model and that the process model is able to improve the estimates in the company. The questionnaire is built in a manner that enables the disposition of the standardized process model for other companies or to adapt the process model to the needs of other companies.

INHALTSVERZEICHNIS

1 EINLEITUNG

Diese Masterarbeit enthält eine Auflistung verschiedener Methoden zur Aufwandsschätzung von agilen Softwareprojekten. Die Vorgehensweise zur Ermittlung der geeigneten Schätzmethode ist das Ergebnis der Masterarbeit. Die Vorgehensweise ist anhand eines Prozessmodells beschrieben. Das Prozessmodell verbessert die Aufwandsschätzung durch die Standardisierung des Prozesses, wobei dieser Umstand durch die Hypothese im empirischen Teil der Arbeit geprüft wird. Die Anwendung des Prozessmodells findet im empirischen Teil der Masterarbeit im Beispielunternehmen Pankl Racing Systems AG statt.

1.1 Methode

Zur Bearbeitung des Masterarbeitsthemas wird die Forschungsmethode Literaturanalyse angewendet. Im theoretischen Teil der Arbeit werden aus bestehender Literatur Faktoren in der agilen Softwareentwicklung identifiziert, die ausschlaggebend für die Aufwandsschätzung und das anzuwendende Schätzverfahren sind. Diese Faktoren werden aus der bestehenden Literatur identifiziert und beschrieben. Eine besondere Beachtung wird der Kombination verschiedener Schätzmethoden zugewiesen. Im nächsten Schritt findet die Überführung der Faktoren und Schätzmethoden in ein Prozessmodell statt. Im praktischen Teil der Masterarbeit wird das erarbeitete Prozessmodell im Beispielunternehmen Pankl Racing Systems AG für Softwareprojekte der Abteilung Business Applications angewendet. Die Softwareprojekte sind dabei direkt im ERP-System „abas ERP“ oder in einem System, dass mit dem ERP-System in Verbindung steht, umgesetzt. Im Rahmen der Anwendung des Prozessmodells werden Interviews mit Experten durchgeführt und die Anwendung des Prozessmodelles mit einem Fragebogen unterstützt.

1.2 Forschungsfrage und Hypothese

Forschungsfrage: Welche Faktoren wirken sich auf den Prozess der Aufwandsschätzung von Softwareentwicklungen aus und wie können diese Faktoren in ein standardisiertes Prozessmodell überführt werden?

H1: Durch die Anwendung eines Prozessmodells zur Aufwandsschätzung wird eine Verbesserung der Schätzergebnisse erreicht.

H0: Durch die Anwendung eines Prozessmodells zur Aufwandsschätzung wird keine Verbesserung der Schätzergebnisse erreicht.

1.3 Ablauf

Der Ablauf der Masterarbeit sieht nachfolgende Phasen vor:

- Literaturrecherche und -analyse

  In dieser Phase wird die Literaturrecherche und Erstellung des theoretischen Teiles der Masterarbeit stattfinden. In dieser Phase werden die Faktoren, die die Aufwandsschätzung beeinflussen beschrieben.

- Synthese und Erstellung des Prozessmodells

  In diesem Schritt wird das Ergebnis der Literaturanalyse beschrieben und auf Basis der gewonnenen Erkenntnisse das Prozessmodell erstellt.

- Anwendung des Prozessmodells

  In dieser Phase wird das Prozessmodell in der Abteilung Business Applications von Pankl Racing Systems AG praktisch angewendet.

- Erkenntnisse aus der Anwendung

  Die Ergebnisse der Anwendung des erstellten Prozessmodells werden in der Masterarbeit beschrieben. In diesem Schritt wird auch die Tauglichkeit für den praktischen Einsatz bewertet. Abschließend werden ein Ausblick auf zukünftige Forschungen und Weiterentwicklungen des Prozessmodells gegeben.

2 AGILE SOFTWAREENTWICKLUNG

Der agilen Softwareentwicklung liegt ein Manifest zugrunde, welches als „Agile Manifesto“ bezeichnet wird. Dieses Manifest wurde auf einer Webseite[1] veröffentlicht und es kann online von Personen unterschrieben werden. Das agile Manifest umfasst die folgenden zwölf Prinzipen (Beck et al., 2001):

> Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
>
> Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
>
> Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
>
> Business people and developers must work together daily throughout the project.
>
> Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
>
> The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
>
> Working software is the primary measure of progress.
>
> Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
>
> Continuous attention to technical excellence and good design enhances agility.
>
> Simplicity--the art of maximizing the amount of work not done--is essential.

[1] http://agilemanifesto.org

> The best architectures, requirements, and designs
> emerge from self-organizing teams.
>
> At regular intervals, the team reflects on how
> to become more effective, then tunes and adjusts
> its behavior accordingly.

Das agile Manifest hat für die vorliegende Arbeit eine große Bedeutung. Das Prozessmodell für Aufwandsschätzung, welches in der dieser Masterarbeit entwickelt wird, wird für Softwareprojekte entwickelt, die nach agiler Methodik durchgeführt werden. Diesem agilen Manifest sind zahlreiche agile Prozesse zur Entwicklung von Software zugeordnet. Cockburn (2006) beschreibt die Entstehung des Manifests als Treffen von Personen, die sich die Frage stellten, ob es Zufall ist, dass ihre Aussagen und verwendeten Wörter sehr ähnlich sind. Es waren untere anderem Repräsentanten von XP, Scrum, Crystal, Adaptive, Feature Driven Development und DSDM anwesend.

Cockburn (2006) ortet als großen Unterschied zwischen der agilen Softwareentwicklung und der zuvor gelebten Paradigma des Wasserfall- und V-Modells vor allem die Variabilität in den Phasen und der frühen Einbindung von Kundinnen und Kunden. Durch das frühe Feedback ist es möglich, auf geänderte Wünsche einzugehen, bevor ein Produkt als fertig ausgeliefert wird.

Um einzelne Elemente aus den Softwareentwicklungsprozessen für die Aufwandsschätzung extrahieren zu können, werden im folgenden Teil agile Softwareentwicklungsprozesse detailliert beschrieben.

2.1 XP

Nach Beck und Andres (2004) legt Extreme Programming seinen Fokus auf die Erhöhung der Produktivität von Entwicklerinnen und Entwickler. Es sollen alle Blockaden entfernt werden, die sich kontraproduktiv zu diesem Ziel verhalten. Des Weiteren werden Sicherheitsmechanismen außer Kraft gesetzt, wodurch ein Gefühl von Auslieferung aufkommen kann. Grundlegend handelt es sich bei XP um eine Form der Softwareentwicklung, die auf den Werten Kommunikation, Feedback, Einfachheit, Courage und Respekt basiert. Des Weiteren werden eine Reihe von verifizierten Praktiken angeboten, die die Softwareentwicklung verbessern. Ein weiterer wichtiger Inhalt sind verschieden Prinzipien, die es erlauben, die Werte von XP in die Praxis umzusetzen. Als einen entscheidenden Faktor wird auch die starke Gemeinschaft, die hinter diesen Wert steht, angesehen.

Um einen näheren Einblick in die Praktiken von XP zu erhalten, haben Beck und Andres (2004) diese zusammengefasst. Ein wichtiger Punkt ist das Zusammensein des Teams, daher wird vorausgesetzt, dass ein Team in einem Raum sitzt. Trotz dieser engen Verbindung muss auf Privatsphäre der Teammitglieder geachtet werden, dies kann durch einzelne abgetrennte Bereiche oder differierende Anwesenheitszeiten sichergestellt werden. Eine weitere wichtige Praktik ist das Teamgefühl, jeder und jede, die sich im Team befindet, muss gemeinsam mit den anderen Mitgliedern zusammenarbeiten und diese unterstützen. Sofern ein Teammitglied

aber seine Arbeit an einem Projekt vollendet hat, kann sie das Team verlassen und ein anderes Team unterstützen. Als Beispiel wird das Erstellen der Datenbankstruktur genannt, sofern dies von der Datenbankadministratorin oder -administrator durchgeführt wurde, können sie in anderen Projekten ihre Expertise einbringen. Als weitere Strategie zu Extreme Programming wird ein informativer Arbeitsplatz gesehen. Es sollen User-Stories, die bearbeitet werden und geplant sind, sowie die Wochen- und Release-Planung direkt sichtbar sein. Ein Erfolgsfaktor von XP ist auch der Umgang mit der eigenen Energie, denn es sollte nur solange gearbeitet werden, wie ausreichend Produktivität vorhanden ist. Pair-Programming ist eine Aktivität, die in XP allgegenwärtig ist. Dabei wird gemeinsam an User Stories gearbeitet, dies umfasst das Ausarbeiten von Ideen, Konzepten und die Hilfestellung bei Hürden in der Programmierung. Als weitere wichtige Bestandteile gelten die wöchentlichen und vierteljährlichen Pläne, wobei bei Letzteren zu berücksichtigen ist, dass unter dem Motto „Slack“ auch Aufgaben vorhanden sein sollen, die bei Engpässen nicht durchgeführt werden. Die wöchentlichen Meetings sollen ein Review, eine Auswahl an User Stories für die nächste Woche, eine Aufteilung dieser User Stories in Tasks und die Zuweisung von Tasks zu Teammitgliedern durchgeführt werden soll, beinhalten. Es sind noch Continuous Integration und die maximale Build-Dauer von zehn Minuten zu nennen, die der Sicherstellung der Qualität dienen. Eine Praktik beschreibt Test-First-Programming, welches sich zuerst mit dem Aufsetzen der Testfälle für ein Feature beschäftigt. Es wird erst nach Abschluss dieser Tätigkeit mit der Programmierung des Features begonnen. Abschließend ist die Praktik inkrementelles Design zu nennen, welche darauf aufmerksam macht, das stets am Design des Systems gearbeitet werden soll. Als wichtigsten Design-Punkt wird das Entfernen von Duplikaten gesehen.

Als Container für Usernutzen werden in XP User Stories verwendet, welches im wöchentlichen Rhythmus geplant werden. Übergeordnet findet eine Planung auf vierteljähriger Basis statt. Als Sicherheitsnetz werden dabei einige User Stories miteinbezogen, welche unter Umständen wieder entfernt werden können. Diese Vorgehensweise stellt auch in Hinblick auf das Prozessmodell eine erstrebenswerte Praxis dar, denn möglicherweise kann eine solche Kennzeichnung bereits direkt im Prozess der Aufwandsschätzung erfolgen.

2.2 Scrum

Schwaber (1995) stellte den Softwareentwicklungsprozess Scrum als agile Alternative zum Wasserfallmodell vor. Bei der Verwendung von Scrum werden folgende Kriterien für die Release-Planung herangezogen:

- Anforderungen der Kundinnen und Kunden
- Zeitdruck
- Konkurrenz
- Qualität
- Vision

- Ressourcen

Der Scrum-Prozess wird von Schwaber (1995) durch einige Charakteristiken beschrieben. Zu Beginn sind die Planungsphase und die Software Architektur als fixer Bestandteil vorhanden, danach erfolgt eine nicht definierte Anzahl an Sprint-Phasen und die Abschlussphase ist wiederum definiert. In der Sprint-Phase erfolgt die Entwicklung von Features, welche zum Abschluss ein Review-Meeting hat, in welchem wiederum ein Review und auch die das neue Sprintbacklog besprochen wird. Die Sprintphase ist dabei ein fixer Zeitraum, welcher grundlegend ein bis vier Wochen umfasst. Der Umgang mit Risiko erfolgt dabei kontinuierlich und wird im Rahmen der Sprints werden Strategien zur Bekämpfung der Risiken eingerichtet. Sobald die zu Beginn dieses Abschnittes angeführten Kriterien zur Release-Pllanung dem Management zu erkennen geben, dass ein neues Release zu veröffentlichen ist, werden die Sprint-Aktivitäten von der Abschlussphase abgelöst. In dieser Phase wird der Release vorbereitet, Integrations- und Systemtests durchgeführt sowie Dokumentationen und Trainings bereitgestellt. Die Aufwandsschätzung beim Anwenden von Scrum kann sowohl in Story Points als auch in Function Points durchgeführt werden.

Der Softwareentwicklungsprozess Scrum beinhaltet in Hinblick auf die Entwicklung des Aufwandsschätzungsmodells vor allem in der Planungsphase von Relevanz. In dieser Phase wird die Software Architektur festgelegt und somit spezifische Entscheidungen getroffen, die sich letztendlich auf den Aufwand der Implementierungsphase auswirken.

2.3 Crystal

Crystal Clear wurde von Cockburn (2005) in Zusammenarbeit mit kleinen Teams entwickelt. Crystal Clear als Mitglied der Crystal-Familie hat den Anspruch einfach, tolerant und erfolgreich zu sein. Dies wird durch ständige Releases, reflektierenden Verbesserungen und enger Kommunikation erreicht. Viele Bedingungen werden variabel und angepasst an die Situation justiert. Es ist zum Beispiel nicht vorgegeben, wie detailreich die Anforderungen definiert werden müssen, es ist abhängig davon, wie schwerwiegend ein Fehler in den Anforderungen sein würde. Crystal Clear besteht aus den sieben Eigenschaften ständige Releases, reflektierende Verbesserungen, osmotische Kommunikation, persönliche Sicherheit, Fokus, einfacher Zugang zu Expertenusern und einer technischen Umgebung mit automatisierten Tests, Konfigurationsmanagement sowie ständiger Integration. Des Weiteren werden verschiedene Techniken und Strategien eingesetzt. Die eingesetzten Strategien sind:

- Exploration 360°,
- früher Erfolg,
- gehendes Skelett,
- inkrementelle Umgestaltung der Architektur und
- Informationen sichtbar machen.

Als Techniken werden von Cockburn (2015):

- Methoden formen,
- Reflexions-Workshops,
- Blitz-Planung,
- Delphi Schätzung unter Nutzung von Expertenwissen,
- tägliche Stand-up-Meetings,
- erforderliches Interaktions-Design,
- Prozessminiaturen,
- Seite-an-Seite-Programmierung und
- Burn Charts angeführt.

Für die vorliegende Arbeit von besonderer Bedeutung ist die Technik Delphi-Schätzung, die die Nutzung von Expertenwissen vorsieht. Es kann festgehalten werden, dass Crystal Clear die Methode zur Aufwandsschätzung bereits im Softwareentwicklungsprozess definiert. Eine genauere Beschreibung zur Delphi-Schätzung erfolgt in Kapitel 3.3.4 Wideband Delphi.

2.4 Adaptive Software Development

Der Adaptive Software Development Lifecycle ist im Gegensatz zu den traditionellen Softwareentwicklungsprozessen weniger auf Optimierung, sondern auf Adaption ausgelegt (Highsmith III, 1999). Ein weiteres Charakteristikum des Adaptive Software Development ist ihre im Gegensatz zur auferlegten die aufstrebende Kultur. Dieser Softwareentwicklungsprozess steht auch für eine komponentenbasierende und nicht für eine aufgabenbasierende Ausrichtung, dies zeigt, das ASD seinen Fokus auf Resultate und deren Schranken (z.B. Qualität) richtet. In ASD eingegliedert sind das Adaptive Conceptual Model, das Adaptive Development Model und das Adaptive Management Model. Das Adaptive Conceptual Model beschäftigt sich mit komplexen Systemen aus Entwicklungs- und Managementsicht. Im Adaptive Development Model werden die iterativen Phasen der Softwareentwicklung und die Kollaboration zur Erhöhung von Geschwindigkeit und Flexibilität beschrieben. Das Adaptive Management Model ist auf die Themen wie eine adaptive Kultur geschaffen werden kann und adaptive Praktiken, die Kollaboration vorantreiben, ausgerichtet.

In Hinblick auf die Aufwandsschätzung wird in ASD das Product Specification Outline verwendet. Das PSO liefert für die Stakeholder und die Projektkernteammitglieder eine Übersicht über den Umfang und die Grenzen der zu leistenden Softwareentwicklung. Es ist von besonderer Bedeutung zu definieren, was im Produkt inkludiert ist, allerdings ist es unter Umständen noch wichtiger zu definieren, was aus dem Projekt exkludiert ist. Das PSO ist auch der Grundstein für die initiale Aufwandsschätzung. Eine Schätzung ist als Herunterbrechen der Arbeitslast und Schätzung eines jeden Features möglich oder es können Werkzeuge wie Function-Point-Schätzungen eingesetzt werden.

In Hinblick auf das Prozessmodell zur Aufwandsschätzung ist PSO ein interessantes Dokument und dessen weitere Verarbeitung in ASD wird bei der Entwicklung des Prozessmodells Berücksichtigung finden.

2.5 Feature Driven Development

Wie der Name dieses Softwareentwicklungsprozesses schon vermuten lässt, dreht sich bei dem von Jeff De Luca (o. J.) beschriebenen Prozess alles um Features. Die fünf durchzuführenden Hauptaktivitäten sind:

- erstellen eines Gesamtmodells,
- erstellen einer Feature-Liste,
- Planung je Feature,
- Entwurf je Feature und
- Implementierung je Feature.

Den FDD-Prozessen sind Tasks zugeordnet, die entweder optional oder verpflichtend durchzuführen sind. Zur Bearbeitung der Tasks definiert FDD die Rollen (Jeff De Luca, o. J.):

- Domain Manager,
- Release Manager,
- Meister der Programmiersprache,
- Build-Verantwortlichen,
- Werkzeug-Verantwortlichen,
- Systemadministrator,
- Tester,
- Deployer und
- technischer Redakteur.

Rock (2007) ordnet FDD im agilen Umfeld als starrer als zum Beispiel Scrum oder XP ein. Dies begründet er mit dem Umstand, dass die einzelnen Prozesse von FDD in sequentieller Abfolge durchgeführt werden. Die Agilität wird durch die kurzen, definierten Zeiträume der Prozesse erreicht, wodurch nach diesen Zeiträumen ein Reagieren auf geänderte Anforderungen möglich ist. Die Entwurfs- und Implementierungsphase wird demnach mit einer gemeinsamen maximalen Dauer von zwei Wochen beschrieben.

Jeff De Luca (o. J.) beschreibt Planung und Schätzung im zweiten Prozess, in dem es gilt, eine Feature-Liste zu erstellen. Die hunderten Features, die auf der Liste sind, müssen gegeneinander in sehr genauem Maße abgewogen werden. Es ist nicht ausreichend sie in komplex, normal und einfach einzuteilen.

2.6 DSDM

Die Dynamic Systems Development Method wird von Stapleton (1997) als die Variante für Softwareentwicklungsprozesse beschrieben, die ein Ergebnis liefert, wie es eigentlich vom Wasserfallmodell gewünscht wäre. Dies wird mit dem effizienten Einsatz von Zeit, wodurch weniger Zeit benötigt wird und dem Umstand, dass nur jene Features entwickelt werden, die wirklich benötigt werden, begründet. Eine weitere Stärke des Systems ist die Erweiter- und Wartbarkeit über einen langen Zeitraum. DSDM erreicht diese Umstände durch die Anwendung der acht Prinzipen, die als (Agile Business Consortium, 2014):

- Fokus auf die Business-Anforderungen,
- Lieferung zum vereinbarten Termin,
- Kollaboration,
- Verbot zur Minderung der Qualität,
- inkrementelle Builds auf festen Grundlagen,
- iterativen Entwicklung,
- klarer und kontinuierlicher Kommunikation und
- Beweis der Kontrolle

bezeichnet werden.

Das Agile Business Consortium (2014) beschreibt die unterliegenden Prozesse flexibel und immer auf Business-Ziele ausgerichtet. Es wird übermittelt, das DSDM mit PRINCE2, PMI und XP integriert ist. In der Aufteilung der Teams wird zwischen Projektteam und Lösungsentwicklungsteam unterschieden. Den beiden Teams steht als Verbindung eine Business Analystin oder Business Analyst zur Verfügung.

In Bezug auf die Aufwandsschätzung wird vom Agile Business Consortium (2014) zu Beginn des Projektes eine Top-Down-Schätzung durch vergleichsbasierende Schätzung empfohlen. Die gleiche Person, von der die erste Schätzung abgegeben wurde, soll durch die Anwendung einer zweiten Methode ein genaueres Ergebnis erzielen. Die höchstpriorisierten Anforderungen, welche zu Beginn geliefert werden, sollen in zerkleinert und auf Aufgaben heruntergebrochen werden, welche wiederum individuell geschätzt werden. Wenn sich die beiden Schätzmethoden in ihrem Ergebnis stark voneinander unterscheiden, soll mit einer weiteren Methode die Genauigkeit gesteigert werden. Es soll sich eine gruppenbasierende Schätzung dafür eignen.

3 AUFWANDSSCHÄTZUNG

In diesem Kapitel werden mit der Aufwandsschätzung in Beziehung stehende Begriffe erläutert und Methoden zur Aufwandsschätzung erklärt.

3.1 Begriffe der Aufwandsschätzung

Die Begriffe Aufwand und Kosten sind in der Softwareentwicklung allgegenwärtig und werden teilweise auch als Synonyme verwendet. Trendowicz und Jeffery (2014) definieren den Begriff Kosten in der Domäne der Softwareentwicklung immer als monetären Wert eines Produktes oder einer Dienstleistung. Der Aufwand kann als aufgewendete Zeit zur Erzeugung des Produktes oder der Dienstleistung verstanden werden. Daraus wird die Schlussfolgerung gezogen, dass der Aufwand immer in den Kosten eines Softwareprojektes enthalten ist, aber keinesfalls dadurch limitiert wird.

Im Zusammenhang mit der Aufwandsschätzung von Softwareprojekten sind die Begriffe Schätzung („Estimation“), Vorhersage („Prediction“) und Planung („Planning“) verbunden. Die Abgrenzung dieser Begriffe ist vor der genaueren Betrachtung des Themas Aufwandsschätzung notwendig. Einen Einblick in diese Begriffe bieten ebenfalls Trendowicz und Jeffery (2014), durch einen Vergleich von Schätzung zu Vorhersage sowie Vorhersage zu Planung. Der Begriff Schätzung ist auf die Beurteilung des Ist-Zustandes zu beziehen, wobei der Begriff Vorhersage als Vorabschätzung definiert werden kann. Es sollte für Softwareprojekte der Begriff Aufwandsvorhersage („effort prediction“) verwendet werden. In der angeführten Quelle werden die Begriffe Schätzung und Vorhersage als Synonyme verwendet. Im Gegensatz zu diesen beiden Begriffen kann Planung nicht als Synonym verwendet werden. Planung beschreibt die Tätigkeit ein Projekt darauf auszurichten, dass die Projektziele und ein definiertes Projektende erreicht werden.

3.2 Paradigmen der Aufwandsschätzung

Im nachfolgenden Teil der Arbeit werden verschiedene Paradigmen zur Aufwandsschätzung sowie deren charakteristische Merkmale erläutert. In Abbildung 1 werden die verschiedenen Paradigmen, in welche die Methoden eingeteilt sind, dargestellt.

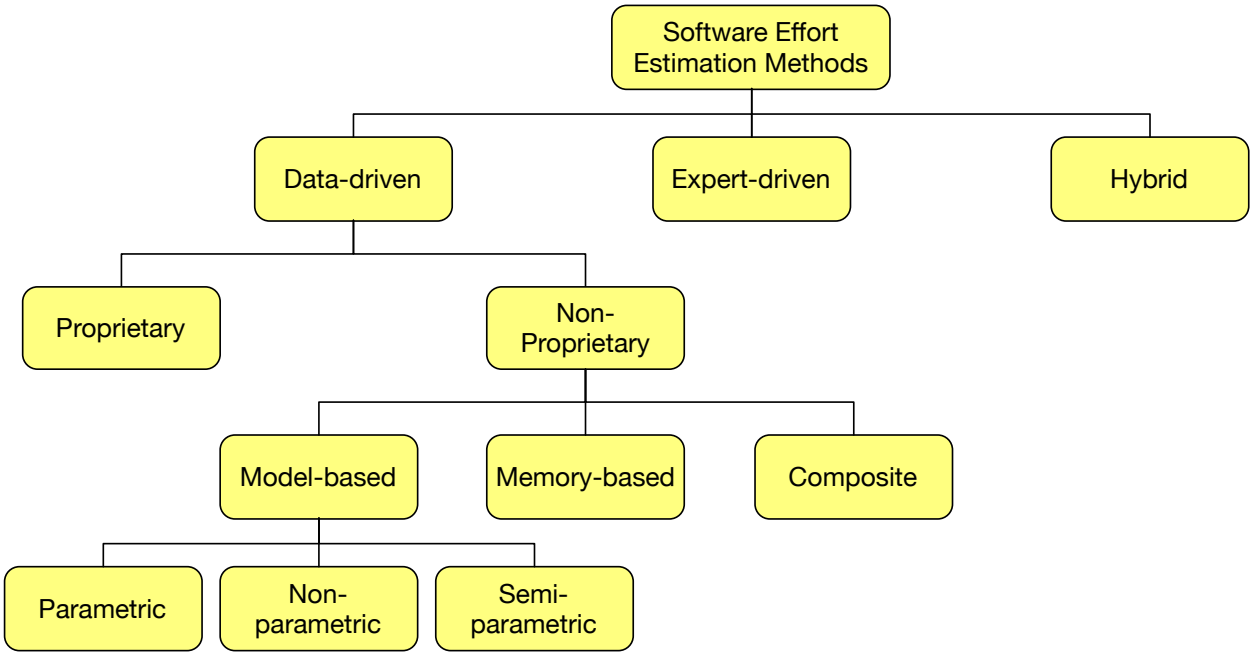

Abbildung 1 Klassifizierung von Methoden zur Aufwandsschätzung. Nachgezeichnet von Software Project Effort Estimation (S. 156), von A. Trendowicz und R. Jeffery, 2014, Kaiserslautern, Sydney: Springer

Jorgensen (2014) stellte fest, dass Modelle und Methoden von ihrem Anwendungsgebiet abhängig sind. Aus diesem Grund wurde dargelegt, dass es kein bestes Modell und keine beste Methode gibt. Es ist bei statistischen Herangehensweisen sogar das Gegenteil der Fall, denn statistisch überlegene Modelle neigen zur Überanpassung an historische Daten. Dies führt in zu berechnenden Fällen zu einer geringeren Genauigkeit. Es wird außerdem hervorgehoben, dass die Schätzung das Produkt beeinflusst, denn eine zu geringe Aufwandsschätzung kann zu weniger Qualität, mehr Aufwand in der Nacharbeit sowie einer höheren Gefährdung des Projektes führen. Für eine zu hohe Aufwandsschätzung kann zu einer verminderten Produktivität bei der Herstellung des Produktes führen. Nachfolgend werden die drei Klassifizierungen „basierend auf Daten“ („Data-Driven“), „Expertenschätzungen („Expert-driven“) und „Hybrid“ näher beschrieben (Trendowicz & Jeffery, 2014).

3.2.1 Schätzungen basierend auf Daten

Die Vorgehensweisen, welche unter dem Zweig „Data-Driven“ zu finden sind, haben einerseits proprietäre und nicht- proprietäre Ausprägungen. Die Unterscheidung zwischen diesen beiden Kategorien ist an der öffentlichen Verfügbarkeit ihrer Dokumentation gebunden. Demnach sind proprietäre Vorgehensweise nicht vollständig öffentlich dokumentiert (Trendowicz & Jeffery, 2014).

Der grundsätzliche Aufbau dieser Schätzverfahren ist allerdings identisch, denn es wird immer basierend auf historischen Projekten eine Schätzung für ein aktuelles Projekt durchgeführt. Dabei ist jedenfalls zu beachten, dass die aktuellen Projekte nicht mit historischen Daten der historischen Projekte geschätzt werden. Um eine Unter- bzw. Überschätzung aus historischen Projekten nicht in aktuelle Projekte zu übertragen, ist es ratsam aktuelle Daten zu den historischen Projekten zu verwenden (Trendowicz & Jeffery, 2014).

McConnell (2006) unterteilt die datengetriebene Aufwandsschätzung anhand der verwendeten Datenbasis, welche sich je nach Projektstadium verändert. Zu Beginn bis zur Mitte des Projektes werden historischen Daten aus Industrie-Datenbanken und unternehmensinternen Datenbanken verwendet. Ab der Mitte des Projektes werden die historischen Daten aus dem vorliegenden Projekt verwendet, um eine Aufwandsschätzung für die noch zu erfüllenden Aufgaben durchführen zu können. In diesem Zusammenhang wird erwähnt, dass bei der Verwendung von historischen Daten aus Industrie-Datenbanken nicht dieselbe Qualität wie bei unternehmensinternen Daten vorliegt. Dies wird mit der unterschiedlichen Produktivitätsrate der Unternehmen in einem Industriesektor begründet. Die Produktivitätsraten differieren dabei bis zum Faktor zehn. Hill (2010) stellt ebenfalls fest, dass unternehmensinterne Daten allgemeinen Datenbanken vorzuziehen sind. Die unternehmensinternen Daten berücksichtigen dabei Eigenschaften wie die Arbeitsumgebung, die Arbeitsweise, die Motivation der Mitarbeitenden, die Arbeitsaufteilung (Support zu Projektierung) sowie Rahmenbedingungen wie das Management und die Organisationsstruktur.

Eine Quelle für Daten aus der Industrie wird von Hill (2010) beschrieben und trägt den Namen ISBSG. Bei der Verwendung von Daten aus generalisierten Datenbanken ist Wissen über die Stärke, die Limitierungen und die Positionierung der Daten erforderlich. Einer der wichtigsten zu beachtendem Punkt ist, ob die vorliegenden Projektdaten mit dem zu schätzenden Projekt vergleichbar und tauglich sind eine zufriedenstellende Schätzung zu liefern. Die ISBSG gibt folgende Kriterien als die wichtigsten zu beachtenden an:

- Projektgröße
- Projekttyp (neues Projekt, Erweiterungsprojekt, Erneuerungsprojekt)
- Programmiersprache und Typ der Programmiersprache
- Entwicklungsplattform

Trendowicz und Jeffery (2014) unterscheiden bei den datengetriebenen Methoden in weiterer Folge zwischen modellbasierenden („model-based“) und vergleichsbasierenden („memory-based“ oder „analogy-based“) Verfahren. In beiden Verfahren ist die Größe des Softwareprojektes die wichtigste Charakteristik zur Durchführung der Aufwandsschätzung. Weitere Charakteristiken stellen die Entwicklungsprodukte, Prozesse und Ressourcen dar. Modellbasierende Methoden sind zum Beispiel die statistische Regression, wobei verschiedene Modelle mit verschiedenen Modellen erlernt werden und daraus ein Schätzungsmodell entwickelt wird. Ein Modell, das aus mehreren Teilmodellen besteht, ist CoBRA.

Die vergleichsbasierenden Schätzungen werden von Trendowicz und Jeffery (2014) dadurch charakterisiert, als dass sie kein Modell erstellen, sondern auf Basis von verschiedenen Projekteigenschaften nach dem ähnlichsten historischen Projekt suchen. Sofern dieses ähnlichste Projekt identifiziert wurde, wird es für Schätzungsmethoden verwendet, die Aufgrund von Ähnlichkeit eine Aussage über den zu erwartenden Aufwand treffen. Eine Schätzungsmethode, die in dieser Kategorie zu finden ist, trägt den Namen CBR.

Die Stärken und Schwächen von modellbasierenden und vergleichsbasierenden Modellen vereinen Kombinationsmethoden („composite models“). Die Umsetzung eines solchen

zweistufigen Modells kann in der Ausprägung zunächst auf Basis eines vergleichsbasierenden Modells ähnliche Projekte identifizieren und anschließend modellbasierend ein Aufwandsschätzungsmodell auf Basis der gefundenen Daten die Aufwandsschätzung vornehmen (Trendowicz & Jeffery, 2014).

3.2.2 Expertenschätzungen

Expertenschätzungen stellen nach wie vor die beliebteste Methode von Aufwandsschätzungen bei Softwareprojekten dar. Es handelt sich dabei um rund 70 bis 80% aller geschätzten Softwareprojekte. Zu den Methoden, die unter die Expertenschätzungen fallen, zählen sowohl Methoden, die nur eine Expertin oder einen Experten involvieren als auch jene die mehrere Expertinnen und Experten in die Schätzung miteinbeziehen. Wobei auch die Schätzung von nur einer Expertin und einem Experten durch eine strukturierte Methode durchgeführt werden kann und nicht auf einer „Daumenregel“ beruht. Die Unterscheidung zwischen strukturierten und unstrukturierten Methoden wird ebenfalls bei Methoden durchgeführt, die von mehreren Expertinnen und Experten ausgeübt werden. Ein Faktor ist bei all diesen Methoden gleich relevant, es handelt sich um den menschlichen Faktor. Der Erfolg ist davon abhängig, wie sehr sich die Menschen von ihrer Voreingenommenheit beeinflussen lassen. Aus diesem Grund sind die Gruppenschätzungen zu bevorzugen, da hier ein natürlicher Rückgang der Beeinflussung durch verschiedene Meinungen in Diskussionen erreicht wird (Trendowicz & Jeffery, 2014). Diese Schlussfolgerung wurde durch Jorgensen (2014) bestätigt und mit der Empfehlung versehen, dass sämtliche misslich beeinflussende Textelemente aus Anforderungsdefinitionen vor der Aufwandsschätzung entfernt werden sollen. Des Weiteren hat die Erwartungshaltung der Kundinnen und Kunden Einfluss auf die Aufwandsschätzung, denn erwarten sich die Kundinnen und Kunden einen günstigen Preis und eine schnelle Fertigstellung, tendieren die Expertenschätzungen dazu den Aufwand zu unterschätzen.

3.2.3 Hybride Schätzungen

Die bisher diskutierten Paradigmen der datenbasierenden Aufwandsschätzung und Expertenschätzung besitzen sowohl Stärken als auch Schwächen. Zu den Stärken der datenbasierenden Methoden zählen laut Trendowicz und Jeffery (2014):

- die minimale Einbeziehung von Menschen,
- die hohe Flexibilität,
- umfangreiche Dokumentationen und vorhandene Werkzeuge,
- alle Projekttätigkeiten sind abgebildet,
- alle Softwareentwicklungsstufen sind abgebildet,
- empirische Bestätigung ist vorhanden und
- relativ geringe Anwendungskosten.

Diesen Stärken stehen als Schwächen:

- hohe Ansprüche an die Daten,
- Schätzungsmethoden sind komplex,
- geringe Wiederverwendbarkeit der Ergebnisse,
- inkonsistente Robustheit gegen schlechte Daten,
- geringe Widerstandsfähigkeit gegen ungewisse Informationen,
- geringe Informationen für die Projektleitung,
- und inkonsistente Vorhersagekraft gegenüber.

Stärken von Expertenschätzungen sind:

- geringe Anforderungen an quantitative Daten,
- hohe Flexibilität der Schätzungsmethoden,
- und geringe Komplexität der Schätzungsmethoden.

Dem gegenüber sind ihre Schwächen:

- hohe Beteiligung des menschlichen Faktors,
- geringe Robustheit gegen negative Beeinflussung,
- geringe Unterstützung durch Dokumentation und Hilfe,
- geringe Wiederverwendbarkeit der Ergebnisse,
- inkonsistente Vorhersagekraft,
- geringe Informationen für die Projektleitung,
- und geringe Widerstandsfähigkeit gegen ungewisse Informationen.

Jorgensen (2014) stellte fest, dass es keine beste Methode zur Aufwandsschätzung gibt, demzufolge hat sie bereits zuvor dargelegt, dass es auch kein bestes Paradigma zur Aufwandsschätzung gibt (Trendowicz & Jeffery, 2014). Aus diesem Grund und der zuvor dargelegten Stärken und Schwächen der beiden Ausgangsparadigmen wurde ein hybrides Paradigma geschaffen, welches die Stärken aus beiden Paradigmen vereinen und die Schwächen kompensieren soll.

Zu den Methoden des hybriden Paradigmas zählen die Theorie von Bayes (Bayes Statistik) und das CoBRA-Modell. Durch die Einführung des hybriden Paradigmas wurden das datenbasierende Schätzen und Expertenschätzungen nicht abgelöst. Es gilt zwischen diesen drei Paradigmen abzuwiegen, welches sich am besten für den Einsatzzweck eignet (Trendowicz & Jeffery, 2014). Um die Paradigmen detaillierter betrachten zu können, werden im nächsten Abschnitt Methoden zur Aufwandsschätzung näher erklärt.

3.3 Methoden der Aufwandsschätzung

Nachfolgend werden Methoden zur Aufwandsschätzung näher erklärt und ihre Vor- bzw. Nachteile deutlich gemacht. Diese Methoden werden im zu entwickelnden Prozessmodell berücksichtigt. Die Methoden, die im nachfolgenden betrachtet werden, werden von McConnell (2006), Cohn (2005) oder Trendocwicz und Jeffery (2014) erwähnt, daher wurden sie in diese Arbeit aufgenommen.

3.3.1 CoBRA

Es handelt sich um ein hybrides Aufwandsschätzungsmodell. Es wurde speziell für Software Engineering erstellt und beinhaltet die wichtigsten Grenzen und Fähigkeiten in Hinblick auf Software Engineering. Ein Teil von CoBRA ist der nominale Aufwand und der andere Teil der Overhead-Aufwand. Beim nominalen Aufwand wird von jenem Aufwand gesprochen, der unter den bestmöglichen Umständen ausreichend ist, um das Softwareprojekt umzusetzen. Die bestmöglichen Umstände müssen dabei im Kontext des Projektes betrachtet werden und von realistischer Ausprägung sein. Dieser Aufwand beinhaltet den Programmier- und Managementaufwand der bei einem Softwareprojekt anfällt. Unter dem Overhead-Aufwand werden alle jene Aufwände verstanden, die sich aus der nicht perfekten Projektumgebung, den mangelnden Fähigkeiten von Projektmitarbeitenden und in Summe alle Aufwände, die nicht produktiv sind. CoBRA beschreibt den Overhead-Aufwand als Prozentsatz des nominalen Aufwandes. Ein Overhead-Aufwand von 50% entspricht damit nominalen Aufwand von 150%. Das Overhead-Aufwand-Modell und dessen Zusammenhang mit dem Produktivitätsmodell werden in der vorliegenden Arbeit nicht näher erklärt, da an dieser Stelle nur die Basis von CoBRA erläutert wird und als Beispiel für eine konkrete Anwendung der datenbasierenden Aufwandsschätzung dient. Bei den modellbasierenden Schätzungen wird zwischen parametrierbaren Modellen und nicht-parametrierbaren Modellen unterschieden. Bei parametrierbaren Modellen muss ein A-Priori angegeben werden. Eine Mischform aus diesen beiden stellen die semi-parametrierbaren Schätzung dar, welche sowohl parametrierbare als auch nicht-parametrierbare Bestandteile vorweisen.

Ein großer Vorteil von CoBRA ist, dass die Methode grundsätzlich für jede Art von Projekt und hier auf jeder Ebene und in jeder Phase angewendet werden kann. Es ist nicht von vielen menschlichen Personen abhängig, denn es sind rund drei Personen mit ausreichendem Domänenwissen erforderlich. Es sind keine historischen Daten notwendig und das wichtigste Attribut zur Kalkulation des Aufwandes ist der Umfang des Softwareprojektes. CoBRA ist sehr gut dokumentiert und existieren zahlreiche Publikationen, die die Anwendung der Methode beschreiben.

3.3.2 COCOMO

McConnell (2015) beschreibt Barry Boehms Methode COCOMO als Verfahren dem 17 Multiplikatoren für den Aufwand und fünf Skalierungsfaktoren zu Grunde liegen. Für die

Schätzung ist es erforderlich diese 22 Faktoren zu parametrieren. Die 22 Faktoren sind grundlegend dazu gedacht Attribute des Softwareprojektes abzubilden, sie werden von McConnell (2006), als 22 Möglichkeiten eine fehlerhafte Schätzung zu erzeugen, erläutert.

Trendowicz und Jeffery (2014) beschreiben COCOMO als ein Verfahren, das auf Basis von Attributen und Werten Vorhersagen für den Aufwand von Softwareprojekten liefert. Durch seine starre Grundausrichtung auf Multiorganisationsprojekte nicht in jeden Kontext passt. Das Verfahren ist zwar anpassbar, dafür ist allerdings Expertenwissen erforderlich, wodurch sich aber die Vorhersagekraft steigern lässt. Sofern diese Methode angewandt wird, muss allerdings berücksichtigt werden, dass die Daten, die in das Modell eingebracht werden, von jenen Daten differieren, mit denen das Modell erstellt wurde. Dadurch können die Schätzungen falsche Ergebnisse liefern. Als Beispiel für dieses Verhalten werden die Anzahl an Zeilen im Code und die Angabe von personellen Ressourcen verwendet. COCOMO verwendet für die Angabe des personellen Aufwands Personenmonate, wobei ein Personenmonat 152 Personenstunden entspricht. Dieser Wert ist wiederum abhängig von der jeweiligen Organisation oder dem Staat, in dem die Leistung erbracht wird. Diese doch vorhandene Starre und die sehr spezifische Anpassbarkeit haben zur Folge, dass das Modell keine hohe Wiederverwendbarkeit für verschiedene Kontexte vorweist. Als positiv wird erwähnt, dass COCOMO sehr gut dokumentiert ist und zahlreiche Publikationen existieren, darüber hinaus ist die Weiterentwicklung COCOMO II verfügbar.

3.3.3 CBR

Trendowicz und Jeffery (2014) zu Folge ist CBR eine Ausprägung aus dem Paradigma der analogiebasierten Aufwandsschätzungen. Der grundlegende Ablauf bei der Vorgehensweise nach CBR sieht vor, dass das ähnlichste Projekt zum zu schätzenden Projekt identifiziert wird und daraus die Aufwandsschätzung abgeleitet wird. Die Philosophie, die hinter dieser Vorgehensweise verborgen ist, ist, dass es für jedes zu schätzende Problem ein ähnliches Problem gibt, dass bereits gelöst wurde und aufgrund der Ähnlichkeit des Problems ist Aufwand zur Lösung ebenfalls ähnlich.

CBR ist bereits sehr lange in der Aufwandsschätzung verbreitet und basiert laut Trendowicz und Jeffery (2014) auf vier grundlegenden Schritten (zitiert nach Aamodt & Plaza, 1994):

1. Wiederauffinden („Retrieve“)
2. Wiederverwenden („Reuse“)
3. Bearbeiten („Revise“)
4. Aufbewahren („Retain“)

Die Schritte bedeuten im Wesentlichen, dass zunächst nach einem ähnlichen Problem gesucht wird, dessen Lösung wiederverwendet wird, durch Bearbeitung abgeändert und an die Situation angepasst wird sowie im letzten Schritt für ein zukünftiges Problem wieder aufbewahrt wird, wenn die Lösung erfolgreich getestet wurde. Damit ein ähnliches Problem aufgefunden werden kann, müssen Attribute definiert und gewichtet werden. Zur Verwendung des oder der

gefundenen Analogien existieren verschiedene Herangehensweisen, wobei die einfachste Methode jene darstellt, die das Projekt für die Aufwandsschätzung heranzieht, das als ähnlichstes klassifiziert wurde. Es eigenen sich auch Methoden, die den statistischen Mittelwert oder Median aus ähnlich klassifizierten Projekten bilden.

Der größte Vorteil an CBR ist, dass keinerlei Expertise bei den Anwendern in Hinblick auf das Projekt vorliegen muss. Des Weiteren ist CBR einfach anzuwenden und bietet aufgrund seiner langen Geschichte bereits eine Vielzahl an Dokumentationen und Publikationen, die bei der Anwendung dieser Methode unterstützen. Der offensichtlichste Nachteil dieser Methode ist das Vorhandensein von Daten aus der Vergangenheit, sind diese nicht vorhanden, ist die Anwendung dieser Methode nicht möglich.

3.3.4 Wideband Delphi

Trendowicz und Jeffery (2014) beschreiben Wideband Delphi als eine strukturierte Methode zur Aufwandsschätzung, die in einzelne Phasen gegliedert ist (siehe Abbildung 2).

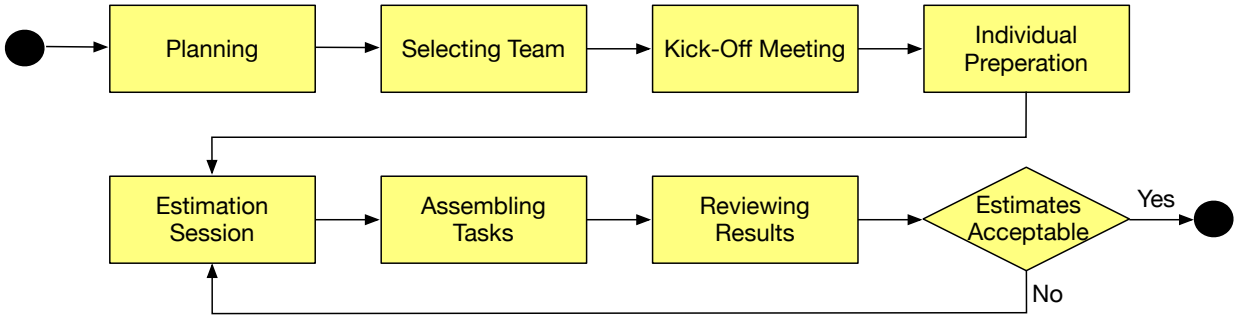

Abbildung 2 Wideband Delphi Methode. Nachgezeichnet von Software Project Effort Estimation (S. 316), von A. Trendowicz und R. Jeffery, 2014, Kaiserslautern, Sydney: Springer

Im ersten Schritt „Planning“ wird das Softwareprojekt grundlegend geplant. Dazu gehören Aktivitäten wie das Zuweisen von Ressourcen und das Erstellen des Schätzungsplanes. Es werden jegliche Informationen, die für das Schätzen des Aufwandes und zur Erstellung des Planes wertvoll sein können, gesammelt. Des Weiteren werden aus Datenbanken relevante historischen Daten, die die Schätzung unterstützen können, identifiziert.

Darauffolgend muss ein Team ausgewählt werden, das die Aufgaben der Aufwandsschätzung durchführt. In diesem Team müssen verschiedene Rollen besetzt werden. Die Koordinatorin oder der Koordinator ist dafür verantwortlich die Rahmenbedingungen von Wideband Delphi zu berücksichtigen und sollte deshalb bereits Erfahrung mit dieser Methode gesammelt haben. Diese Rolle muss außerdem Fähigkeiten im Moderieren von Diskussionen, Behandeln von Konflikten und dem Vorantreiben des Teams vorweisen können. Weitere wichtige Eigenschaften dieser Rolle sind die Objektivität im Umgang mit Schätzungen und dem Projekt selbst. Die zweite Rolle sind die Schätzerinnen und Schätzer. Sie schätzen die Aufgaben, die im Rahmen des Projektes bearbeitet werden müssen. Von besonderer Bedeutung ist die Diversität der Schätzerinnen und Schätzer. Es sollen möglichst alle Beteiligten an einem

Softwareprojekt in dieser Rolle ihre Vertretung haben. In weiterer Folge müssen die ausgewählten Personen über grundlegendes Wissen in der Softwareentwicklung sowie über spezifisches Wissen zum vorliegenden Projekt besitzen. Die dritte Rolle bilden die Beobachterinnen und Beobachter. Es handelt sich bei ihnen um Stakeholder, die einen Blick auf das Projekt und die Aufwandschätzung aus externer Perspektive haben. Sie haben auch die Verantwortung spezifische Fragen zu beantworten und notwendige Informationen an die Schätzerinnen und Schätz weiterzugeben. Eine Eigenschaft, die alle Rollen teilen, ist die Übereinkunft darüber, dass nicht ein bestimmtes Ergebnis, sondern ein gutes Ergebnis im Schätzungsprozess erzielt werden soll.

Im Kickoff Meeting werden alle relevanten Themen rund um die Wideband Delphi Methode durch die koordinierende Person erläutert. In dieser Phase werden auch die Projekt- und Schätzziele diskutiert. Einen weiteren Punkt stellt das Besprechen der wesentlichen Aktivitäten, die im Rahmen des Projektes durchzuführen sind, dar.

In der nächsten Phase bereiten sich alle teilnehmenden Personen auf individuelle Art und Weise auf die Aufwandsschätzung vor. Die identifizierten Aktivitäten werden persönlich geschätzt und Annahmen über das Projekt werden schriftlich festgehalten. Es ist auch möglich, dass die angeführten Aktivitäten in kleinere Aufgaben aufgeteilt und diese geschätzt werden, da die Verkleinerung des Aufgabenumfanges eine Erleichterung der Aufwandsschätzung darstellt. Der Schätzungsprozess selbst sieht vor, dass drei Schätzungen zu jeder Aufgabe abgegeben werden. Dies sind Zahlen für den minimalen, den realistischen und den maximalen Aufwand.

In der Estimation Session wird durch die koordinierende Person eingeleitet und befasst sich zunächst mit den Ergebnissen aus dem Kickoff Meeting und der individuellen Vorbereitung. Nachfolgend werden die persönlich getroffenen Aufwandsschätzungen in eine Vorlage eingetragen. Diese Schriftstücke dienen als Diskussionsgrundlage und leiten einen iterativen Prozess ein, in dem die Aufwandsschätzungen in jedem Zyklus angepasst und erneut diskutiert werden. Eine weitere Möglichkeit besteht darin statistische Mittelwerte, der drei abgegebenen Werte zu berechnen und diese in die Diskussion einzubinden. Die Estimation Session findet ein Ende, wenn die persönlichen Schätzungen in einem angemessenen Rahmen angenähert wurden, die Übereinkunft getroffen wurde, dass keine angemessene Annäherung der Schätzungen erfolgen kann, dass die definierte Anzahl an Schätzungsrunden oder ein Zeitlimit erreicht wurden.

Im darauffolgenden Schritt wird eine finale Liste mit den Aufgaben des Projektes und den zugehörigen Aufwandsschätzungen erstellt. In dieser Phase sollen Ausreißer bei den Schätzungen behandelt und gegebenenfalls gestrichen oder näher ergründet werden. Die Aufwandsschätzung für das Projekt wird durch Aggregation der Werte erreicht.

In der letzten Phase wird die Aufwandsschätzung auf ihre Plausibilität geprüft und ob sie in einem vertretbaren Rahmen sind. Darüber hinaus wird die Vollständigkeit der Aufgabenliste geprüft. In dieser Phase werden Aktivitäten, die in ihren drei Schätzungen sehr weit auseinanderliegen, erneut geprüft und in Betracht gezogen, dass durch Verkleinerung des Aufgabenumfanges und detailreichere Formulierung der Aufgabe eine bessere Schätzung möglich sein könnte. Sofern die Aufwandsschätzung akzeptabel ist, kann der Prozess beendet

werden, ansonsten muss die Estimation Session und die darauffolgenden Phasen erneut durchgeführt werden.

Wideband Delphis Stärke ist zugleich auch seine Schwäche – der menschliche Faktor. Es lässt sich sehr große Flexibilität und Anwendbarkeit in jeder Phase der Softwareentwicklung erzielen. Allerdings ist das Ergebnis von der Expertise der beteiligten Personen abhängig.

McConnell (2006) empfiehlt die Verwendung von Wideband Delphi vor allem in den frühen Phasen eines Softwareprojektes und wenn bisher noch keine Projekte in der spezifischen Domäne durchgeführt wurden oder neue Technologien zum Einsatz kommen. Dies begründet er mit dem hohen Einsatz an personellen Ressourcen, welcher durch die notwendigen Besprechungen mit vielen Personen entsteht. Allerdings spricht er Wideband Delphi auch ein gewisses Maß an Erfolg zu, er konnte feststellen, dass durch den Einsatz dieser Methode die Fehler der Aufwandsschätzung im Durchschnitt von 290% auf 170% gesenkt werden konnten.

3.3.5 Planning Poker

Bei der Methode Planning Poker wird das Schätzen spielerisch im Team durchgeführt. Ein Team umfasst nicht nur Programmierer, sondern wie von Cohn (2005) angeführt Tester, Datenbanktechnischer, Analysten, GUI-Designer und alle anderen am Entwicklungsprozess beteiligten Personen. Die Methode sieht vor, dass alle Personen ein Deck von Karten erhalten. Die Karten sind mit Zahlen entsprechend der Fibonacci-Folge beschrieben (0, 1, 2, 3, 5, 8, 13, …). Die Vorgehensweise sieht vor, dass die zu schätzenden User-Stories vom Product Owner oder einem Analysten vorgelesen werden. Nachdem die User-Story verlesen wurde, können von den anwesenden Personen Fragen an den Product Owner gestellt werden. Im Anschluss daran wird von jeder Person eine Karte ausgewählt, bleibt aber verdeckt für die anderen Personen liegen. Danach werden alle Karten gleichzeitig umgedreht und die Person mit der niedrigsten und höchsten Schätzung erklären ihre Beweggründe. Die teilnehmenden Personen können zu diesem Zeitpunkt für ein paar Minuten über die User-Story diskutieren und führen dann eine erneute Schätzung nach dem zuvor beschriebenen Verfahren durch. Dieser Vorgang wird solange wiederholt bis alle Personen mit der gleichen Zahl einverstanden sind. Es ist nicht notwendig, die gleiche Zahl auszuwählen. Sollte nur noch eine Person mit einer anderen Zahl als alle anderen Personen verblieben sein, ist es durchaus üblich, dass durch die moderierende Person das Einverständnis über die Zahl der Mehrheit eingeholt wird.

Trendowciz und Jeffery (2014) sehen durch die Anwendung von Planning Poker vor allem Vorteile darin, dass die Methode nicht komplex und ihrer Anwendung intuitiv ist. Des Weiteren sind keine historischen Daten notwendig und ist aufgrund nicht notwendiger Parameter (abgesehen von der Software Spezifikation) flexibel zu verwenden. Der größte Nachteil an dieser Methode ist der menschliche Faktor, denn sie ist von Expertenwissen abhängig. Die Methode eignet sich dadurch nicht zur Wiederverwendbarkeit und ist aufgrund der notwendigen humanen Ressourcen kostenintensiv.

3.4 Einheiten zur Aufwandsschätzung

In diesem Teil der Arbeit wird erläutert, mit welchen Einheiten die Aufwandsschätzung durchgeführt werden kann. Da für das Prozessmodell ebenfalls entschieden werden muss, welche Einheit zur Aufwandsschätzung, in welcher Phase zu bevorzugen, sind, ist die Theorie zu den einzelnen Einheiten von großer Bedeutung für die Arbeit.

3.4.1 Ideale Tage

Cohn (2005) beschreibt die Definition von idealen Tagen anhand eines Footballspiels. Es wird die Frage gestellt, wie lange ein Footballspiel dauert. Darauf gibt es einerseits die Antwort die vier 15-minütigen Viertel zu 60 Minuten bzw. einer Stunde aufzusummieren. Eine weitere korrekte Antwort wäre aber auch die Angabe von drei Stunden. Dies ist er Unterschied zwischen idealer Dauer und Durchlaufzeit. Bei der idealen Dauer sind sämtliche peripheren Aktivitäten nicht vorhanden. Es wird festgestellt, dass die Aussage in idealer Dauer immer einfacher einzuschätzen ist. Wenn das Beispiel Footballspiel wiederum betrachtet wird, kann die Frage nach der idealen Dauer sofort mit 60 Minuten beantwortet werden, wobei die Frage nach der Durchlaufdauer für ein bestimmtes Spiel an viele Faktoren geknüpft ist und eine Schätzung möglicherweise nur nach erfolgter Recherche möglich ist (z.B. Anzahl Overtime-Spiele der vorliegenden Mannschaften in den letzten Spielen).

Im Bereich der Softwareentwicklung gibt es zahlreiche Faktoren, warum die ideale Zeit nicht der Durchlaufzeit entspricht, diese sind auszugsweise (Cohen, 2005):

- Unterstützung beim aktuellen Release,
- Krankenstand,
- Meetings,
- Telefonate,
- Schulungen,
- Spezialprojekte,
- E-Mails und
- Reviews.

Die Kriterien für eine Schätzung auf Basis idealer Arbeitstage sind (Cohen, 2005):

- die vorliegende User Story ist die einzige, die bearbeitet wird,
- alles Notwendige, um die Arbeit durchzuführen ist vorhanden und
- es gibt keine Unterbrechungen.

Cohen (2005) sieht den Vorteil von idealen Tagen als Schätzmaß darin, dass die Schätzung unabhängig von der Umgebung durchgeführt werden kann. Es ist unerheblich, ob die Schätzung in einem Startup oder einem multinationalen Konzern durchgeführt wird. Die Zeit, die

für die Programmierung in idealer Zeit veranschlagt wurde, ist immer von gleicher Dauer. Von Bedeutung ist außerdem, dass die Schätzung nicht auf Gruppen oder Individuen stattfinden soll, sondern eine Dauer für eine User Story geschätzt werden soll. Dies dient vor allem dem verringerten Aufwand bei der Schätzung, dem verbleibendem Aufwand beim Fortschreiten der User Story und der Verfolgung der Velocity, da diese ebenfalls getrennt betrachtet werden müssten.

3.4.2 Story Points

Eine weitere Möglichkeit zum Messen von Aufwand stellt die Schätzung in Story Points da. Es handelt sich dabei laut Cohen (2005) um ein relatives Maß, welches keinen Bezug zu ISO-Einheiten hat. Durch Story Points erfolgt die Abschätzung in Form von Punkten, wobei eine User Story mit zwei Punkten den doppelten Aufwand einer User Story von einem Punkt beträgt. In diesem Zusammenhang stellt sich die Frage, wie mit dem Schätzen von Story Points begonnen werden kann. Dazu werden zwei Methoden vorgeschlagen. Die erste Methode sieht vor, dass eine der kleinsten User Storys mit denen jemals gearbeitet wird, gesucht wird und diese mit einem Story Point geschätzt wird. Die andere Methode sieht vor, dass mit einer mittelgroßen User Story begonnen wird und diese etwa mit dem mittleren Wert auf der Skala an angedachten Story Points verwendet wird. Als Empfehlung wird eine Skala von eins bis zehn angegeben.

Cohen (2005) beschreibt, warum die Schätzung mit einem Maß, welches keinen Bezug zu einer ISO-Einheit hat, funktionieren kann. Dahinter verbirgt sich ein Konzept mit dem Namen „Velocity“. Unter Velocity wird die Rate an Story Points verstanden, die ein Team in einem bestimmten Zeitraum fertigstellen kann. Wenn dieses Szenario weiterverfolgt wird, dann kann berechnet werden, wie viele Story Points (welche Velocity) ein Team pro Iterationen umsetzen kann. Daraus kann berechnet werden, wie viele Iterationen noch benötigt werden, um die Fertigstellung der Software erreicht zu haben. Fehler bei der Aufwandsschätzung sind allgegenwärtig und werden ebenfalls durch das Konzept „Velocity“ korrigiert. Als Beispiel wird angeführt, dass ein Team annimmt 25 Story Points pro Iteration umzusetzen und auf Basis dessen davon ausgegangen wird, dass das Projekt nach acht Iterationen abgeschlossen ist. Während der Umsetzung wird erkannt, dass nur 20 Story Points pro Iteration geschafft werden, daher korrigiert sich die Dauer des Projektes automatisch auf zehn Iterationen ohne einzelne Schätzungen korrigieren zu müssen. Ein wichtiges Detail ist, dass nicht mehr die Dauer eines Projektes, sondern der Umfang der Arbeit geschätzt wird. Die Projektdauer wird durch Addition aller Aufwände und anschließender Division durch die Velocity berechnet.

3.4.3 Function Points

McConnell (2006) beschreibt Function Points als logisches Maß zur Berechnung den Umfang eines Softwareprojektes. Die Vorgehensweise sieht vor, dass verschiedene Eigenschaften mit Multiplikatoren je nach Komplexität berechnet werden. Die Summe darauf ergibt die

unangepassten Function Points, welche mit einem Multiplikator noch angepasst werden können. Die Eigenschaften sind:

- externe Eingaben,
- externe Ausgaben,
- externe Abfragen,
- interne logische Dateien und
- externe Interface-Dateien.

McConnell (2006) hat die oben genannten Eigenschaften als Adaption zur ursprünglichen Ausprägung entwickelt. Die Rückschlüsse von Function Points zu Projektdauer und Projektdurchlaufzeiten ist in ähnlicher Ausprägung wie mit Story Points möglich.

4 EINFLUSSFAKTOREN AUF DEN AUFWAND

Die Bestimmung der Aufwandsschätzung muss zahlreichen Einflussfaktoren Sorge tragen. Es gilt wahrscheinliche und unwahrscheinliche Szenarien zu berücksichtigen. Die Ergründung von Einflussfaktoren in der vorliegenden Arbeit dient dem Entwurf eines stabilen Prozessmodells im nächsten Kapitel.

4.1 Anforderungsanalyse

McConnell beschreibt nicht nur die Ergebnisse der Anforderungsanalyse als Einflussfaktor auf die Aufwandsschätzung von Software Projekten, sondern sieht auch die Anforderungsanalyse selbst als wesentlichen Einflussfaktor. Der geschätzte Aufwand für ein Softwareprojekt verteilt sich auf die verschiedenen Aktivitäten, die im Rahmen der Softwareentwicklung durchzuführen sind. Eine grobe Einteilung kann in die Kategorien Architektur, Konstruktion und Systemtest erfolgen. Die Anforderungsanalyse wird nicht als fixer Bestandteil dieser Kategorien angeführt, da sich kein Standard in der Industrie darüber gebildet hat, ob die Anforderungsanalyse als Teil des Softwareentwicklungsprozesses gesehen wird. Beim Vergleich beziehungsweise beim Anwenden von Aufwandsschätzungen ist jedenfalls zu hinterfragen, ob die historischen Vergleichsdaten die Anforderanalyse enthalten oder nicht. Als Begründung für die nur teilweise Berücksichtigung wird die große Variabilität des Aufwandes in der Anforderungsanalyse verstanden. Es ist einerseits möglich große Anforderungsdokumente sehr vage oder sehr detailreich zu beschreiben. Nachfolgende Tabelle bezieht sich auf 1000e Zeilen Code und den daraus abgeleiteten Aufwand für die Anforderungsanalyse.

Größe [KLOC]	Aufwand in der Anforderungsanalyse [%]
1	5
25	5
125	8
500	10

Tabelle 1 Aufwand der Anforderungsanalyse pro KLOC. Entnommen von Software Estimation (S. 235), von S. McConnell, 2006, Redmond, Washington: Microsoft Press

Cohn sieht als ein Problem in der Anforderungsanalyse und Aufwandsabschätzung die Ignoranz der Unsicherheit an. Unsicherheit über die Reichweite und Auswirkungen von Projekten sind vor allem zu Beginn am höchsten. Dies muss daher auch entsprechend berücksichtigt und gehandhabt werden. In diesem Zusammenhang ist auch die Mehrdeutigkeit von Begriffen im Software-Bereich ein oftmaliges Problem.

Trendowicz und Münch (2009) beschrieben die Mehrdeutigkeit von Begriffen im Kontext von Anforderungen an Software als Unsicherheitsfaktor. Dieses Problem ist nicht nur auf der Kunden-Lieferanten-Ebene vorhanden, sondern auch unter Experten weit verbreitet. Als plakatives Beispiel wird der Begriff Software-Zuverlässigkeit angeführt, der je nach Definition (möge dies auch nur taktischer Natur sein) auch Safety- und Security-Themen beinhaltet.

Aus den Ausführungen der beiden Autoren kann entnommen werden, dass eine gemeinsame Sprache sowie ein gemeinsames Problemverständnis die Anforderungsdefinition auf eine stabile Basis stellen. Diese Basis ist dringend notwendig, um eine möglichst genaue Aufwandsschätzung durchführen zu können.

Trendowicz und Jeffrey führen die Integration der Kundinnen und Kunden in den gesamten Entwicklungsprozess als positiven Aspekt an. Ein Teil dieser Zusammenarbeit wird als Review der Anforderungsdokumente verstanden.

Dieser Review-Gedanke sollte bereits bei der Definition von gemeinsamen Begriffen im Projekt eingeführt werden, um Irritationen im späteren Verlauf verhindern zu können und damit durchgeführte Abschätzungen des Aufwands verbessern zu können.

Ein weiterer Aspekt, der berücksichtigt werden muss, ist das Change-Management. Es sind verschiedene Strategien verfügbar, wie mit Änderungen in den Anforderungen umgegangen werden kann. Trendowicz und Jeffrey führen an, dass es die Möglichkeit Änderungen zu verhindern, zu transferieren, abzuschwächen oder zu akzeptieren. Im Kontext des agilen Paradigmas ist es unerlässlich mit Anforderungsänderungen umzugehen, daher scheint die Strategie des Verhinderns nicht zielführend zu sein. Als Faktoren, die direkt in einer Änderung in den Anforderungen münden, können Technologien, Projektanforderungen, Personal und die externe Umgebung genannt werden. Als besonders wichtig wird angesehen, jede Änderungsanfrage im Detail zu analysieren. Eine Änderung, die als geringer Aufwand erscheint und daher nicht eine Änderung in der Aufwandsschätzung bedingt, könnte Folgeänderungen nach sich ziehen, die einen wesentlich höheren Aufwand als die ursprüngliche Änderung bedeuten. Aus diesem Grund kann auch das Verhindern einer Änderung in agilen Softwareprojekten eine notwendige Maßnahme sein, sollte durch die Änderung der gesamte Release-Plan gefährdet werden.

4.2 Irrelevante und irreführende Informationen

Jorgensen und Grimstad (2006) führten ein Experiment durch, das seinen Fokus darauf hatte, in wie fern irrelevante und irreführende Informationen zu einer anderen Abschätzung des Aufwandes führen, als wenn diese Informationen nicht vorhanden wären. Es wurden folgende Manipulationen an den Informationen vorgenommen:

1. Die Länge der Informationen wurde reduziert, allerdings ohne den Inhalt zu verändern.
2. Es wurde die Information ergänzt, dass für das alte (zu ersetzende) System sehr geringer Aufwand notwendig war.

3. Es wurde die Information ergänzt, dass die Auftraggeberinnen und Auftraggeber unrealistisch niedrige Erwartungen an die Kosten haben.
4. Es wurde die Restriktion ergänzt, dass die Entwicklungszeit kurz ist und diese in wenigen Monaten startet.

Alle Manipulationen führten zu einer Reduktion der Aufwandsschätzung, wobei nur Manipulation Nummer vier einen statisch signifikanten Effekt zeigte. Die Verwertung von irrelevanten und irreführenden Informationen wird als Anchoring bezeichnet und wurden bereits 2005 von Aranda beschrieben. Aranda stellte grundlegend fest, dass es Anchoring auch beim Schätzen von Softwareprojekten gibt und konnte vor allem bei der Anwendung von Expertenschätzungen und Experten-basierenden Methoden nachgewiesen werden.

Sowohl Jorgensen und Grimstad (2006) stellten fest, dass die Personen, die für die Aufwandsschätzungen zuständig sind, möglichst nicht in Kontakt mit Ankern kommen sollen. Daher sollten Anker bereits vor der Durchführung von Aufwandsschätzungen aus den zugehörigen Dokumenten entfernt werden. Es ist nicht ausreichend die Personen vor der Aufwandsschätzung auf die Anker hinzuweisen oder diese nachträglich zu entfernen. Sobald das Anchoring stattgefunden hat, ist es irreparabel.

4.3 Maß und Einheit der Schätzung

Die Maße, die für die Schätzung von agilen Softwareprojekten herangezogen werden können, wurden bereits im vorigen Kapitel erläutert (3.4 Einheiten zur Aufwandsschätzung). In diesem Abschnitt wird ausgehend von den erläuterten Einheiten der Einfluss auf die Aufwandsschätzung beschrieben.

Jorgensen (2015) stellte fest, dass die Höhe der Aufwandsschätzung auch vom verwendeten Maß und sogar von der dem Maß zu Grunde liegenden Einheit abhängig ist. Es konnte festgestellt werden, dass Schätzungen in Minuten höher waren als vergleichsweise in Sekunden sowie ergaben Schätzungen in Arbeitstagen höhere Ergebnisse als jene in Arbeitsstunden. Es wurden zwei Mechanismen identifiziert, die dafür verantwortlich sind. Es handelt sich um den „numerosity effect“ und den „unity effect“. In Domänen wie der Softwareentwicklung sollen Schätzungen grundsätzlich niedrig sein, wodurch durch die Verwendung von Arbeitstagen gegenüber von Arbeitsstunden der Optimismus der Schätzung reduziert werden kann. Es ist zu beachten, dass je nach Projektgröße oder Arbeitspaketgröße die Einheit der Schätzung angepasst werden sollte, da dies realistische Schätzungen fördert.

Halkjelsvik und Jorgensen (2018) untersuchten die genannten Effekte weiter und konnten nachfolgende Erkenntnisse festhalten. Als prägnanter Vergleich wird angeführt, das wenn sich ein Jahr kürzer als zwölf Monate oder 365 Tage anfühlen würden, dann hat dies auch Auswirkungen auf Aufwandsschätzungen, denn die schätzende Person würde annehmen, dass sich in 365 Tagen mehr Aufgaben als in zwölf Monaten durchführen lassen. In Hinblick auf die Einheit könnte bereits durch die Frage, ob eine Aufgabe in Stunden, Tagen oder Wochen durchführbar ist, eine Beeinflussung stattfinden. Die Frage impliziert durch die Wahl der Einheit,

ob es sich um eine kürzere oder längere Aufgabe handelt. Um diesen Effekt zu untersuchen, wurden Softwareexpertinnen und -experten eingeladen und gebeten eine kleinere und eine größere Aufgabe zu schätzen. Die eingeladenen Personen wurden in zwei Gruppen aufgeteilt, wobei eine Gruppe in Arbeitsstunden und die andere in Arbeitstagen schätzen musste. Für beide Aufgaben wurde der Aufwand von den Personen höher eingeschätzt, die in Arbeitstagen schätzen mussten. Für die kleinere Aufgabe wurde ein Faktor von zwei nahezu erreicht.

4.4 Software Tests

Turpitka (2016) stellt fest, dass die notwendige Qualität in Softwareprojekten erreicht werden kann, sofern der Testprozess professionell gehandhabt wird. Die Qualität wird erreicht ohne Zeit- und Kostenrestriktionen zu überschreiten. Aus statistischer Sicht kann der Aufwand für Softwaretests von 20% bis zu 50% der Gesamtentwicklungszeit betragen. Die Softwaretests unterliegen verschiedenen Teststrategien und können aus den folgenden Teilen bestehen:

- Nicht funktional
    - Konfigurations-Test
    - Performance-Test
    - Load-Test
    - Stress-Test
    - Usability-Test
- Change-Management
    - Smoke-Test
    - Regressions-Test
- Funktional
    - Installations- und Lizenzierungs-Test
    - Funktions-Test
    - Datenintegritäts-Test
    - Sicherheits-Test
    - GUI-Test

Anhand der angeführten Zahlen und den verschiedenen Ausprägungen von Softwaretests wird klar, dass es sich nicht um einen trivialen Einflussfaktor auf die Aufwandsschätzung handelt, sondern das Thema Softwaretest sehr genau behandelt werden muss.

4.5 Menschliche Faktoren und Teamstruktur

Jorgensen (2014) stellt fest, dass Aufwandsschätzungen nicht nur positive Faktoren haben, sondern auch negative Ausprägungen auf die Produktivität haben können. Zu niedrige Aufwandsschätzungen haben die Folge, dass die Qualität der durchgeführten Arbeit nicht ausreichend ist. Bei zu hohen Aufwandsschätzungen leidet die Produktivität, da sich die Tätigkeiten gemäß der veranschlagten Aufwandsschätzung in die Länge zieht. Es ist nach derzeitigem Stand der Forschung noch nicht bekannt, in wie fern sich der Umfang eines Softwareprojektes auf die Produktivität des Teams auswirkt. Die meisten Studien stellen eine Steigerung der Produktivität bei größeren Softwareprojekten in Aussicht, wohingegen die meisten Experten dies gegenteilig bewerten. Cohn und Jorgensen beschreiben beide, dass die Involvierung des gesamten Teams in die Aufwandsschätzung sich als positiver Faktor erweist. Einerseits wird die Aufwandsschätzung nicht von einer Einzelperson erstellt, damit die Unabhängigkeit der Aufwandsschätzung gewahrt bleibt. Ein weiterer wichtiger Aspekt ist, dass die Aufwandsschätzung nicht von einer Einzelperson getragen wird, sondern ein ganzes Team hinter der getroffenen Entscheidung steht. Trendowicz illustriert in diesem Zusammenhang die Begriffe Bias, Präzision und Genauigkeit in einer Grafik (siehe Abbildung 3). Als hohe Präzision werden verschiedene Aufwandsschätzungen verstanden, die sehr nahe aneinander liegen. Als Bias wird der Fehler verstanden, der zwischen den durchschnittlichen Aufwandsschätzungen und dem richtigen Wert liegt. Bei einer hohen Genauigkeit ist der angesprochene Fehler – der Bias – sehr niedrig und die Präzision sehr hoch.

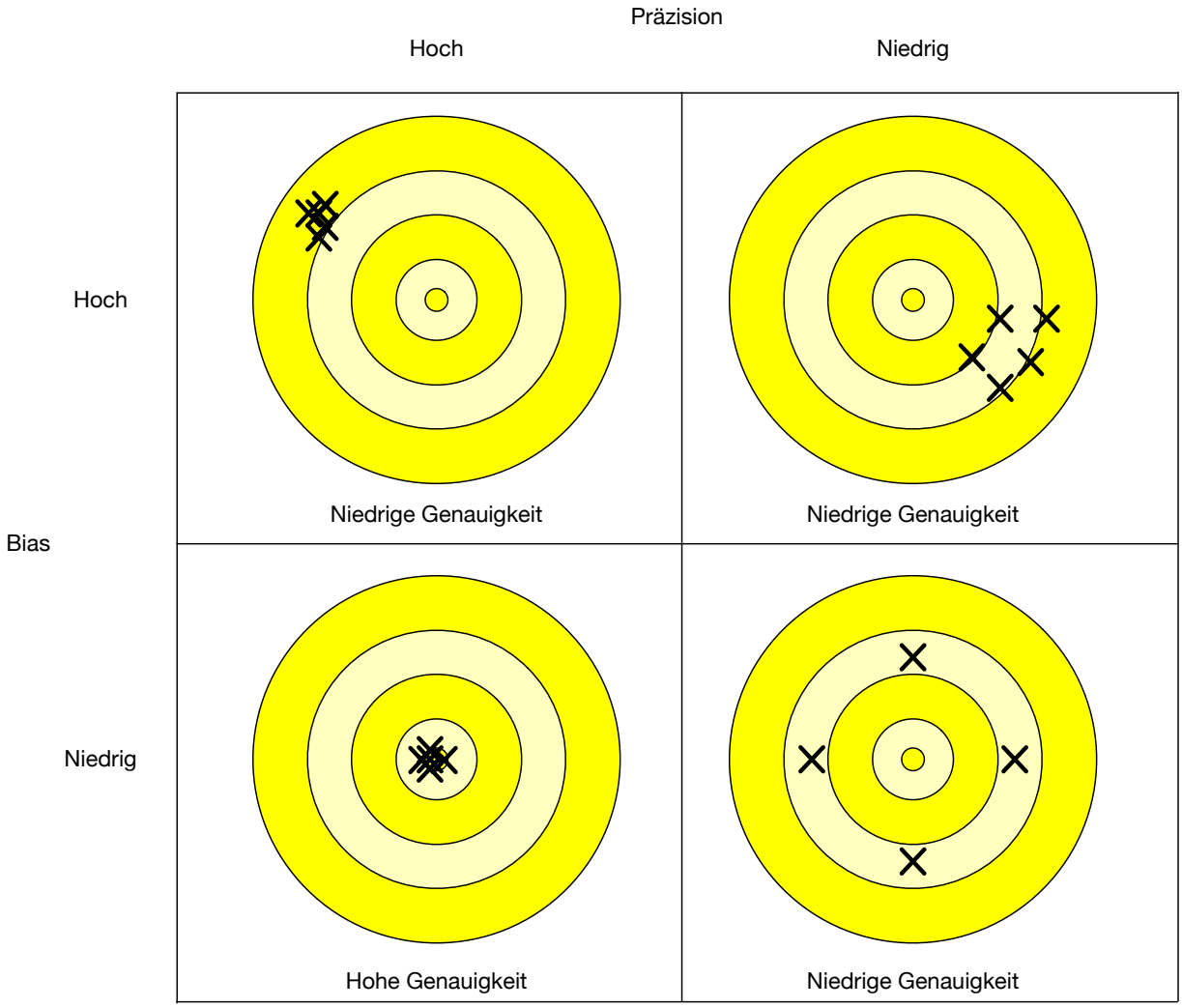

Abbildung 3 Estimation accuracy, precision, and bias. Nachgezeichnet von Software Project Effort Estimation (S. 90), von A. Trendowicz und R. Jeffery, 2014, Kaiserslautern, Sydney: Springer

Trendowicz beschreibt die Interaktion in einem Team als wesentlichen Einflussfaktor auf die Produktivität eines Teams. Je größer der Umfang eines Softwareprojektes wird, desto weniger sind es die individuellen Charakteristiken der Teammitglieder, die für hohe Produktivität sorgen. Es sind dann vielmehr die Interaktionen im Team, die bei umfangreichen Softwareprojekten die notwendige Kommunikation und Koordination sicherstellen. Eine entsprechende Kommunikationsstruktur in einem Softwareprojekt ist unerlässlich, da die Gründe für fehlgeschlagene Projekte oftmals fehlende Kommunikation und gestörter Informationsfluss sind. Die Kommunikationsstruktur kann vertikal oder horizontal umgesetzt werden. Bei einer vertikalen Kommunikationsstruktur gibt es Projektmanager, die für die Kommunikation mit den Benutzerinnen und Benutzern beauftragt sind, wobei in einer horizontalen Kommunikationsstruktur direkt mit dem Team kollaboriert wird. Alex Puscasu unterstreicht diesen Umstand und nennt ebenfalls das Teamgefüge als Einflussfaktor für Aufwandsschätzungen. Einerseits sieht er Kontinuität in der Teamzusammensetzung als wichtigen Faktor für effizientes Arbeiten an und andererseits spricht er auch das Thema an, dass Projektmitglieder meist nur sechs produktive Arbeitsstunden pro Tag zur Verfügung

haben. Bezogen auf einen Achtstundentag bedeutet das, dass die restliche Zeit für indirekte Arbeit wie Meetings und der Beantwortung von E-Mails aufgewendet wird.

4.6 Technologie

McConnell verweist darauf, dass die verwendete Programmiersprache eine Auswirkung auf die Aufwandsschätzung in zumindest vier Wegen zeigt. Sofern die Programmiersprache ausgewählt werden kann, ist dieser Faktor nicht zu unterschätzen. Es müssen die nachfolgenden Faktoren Berücksichtigung finden:

- Erfahrung im Team
- Effizienz der Programmiersprache
- Support der Werkzeuge
- Interpretierbare oder kompilierbare Programmiersprache

Für die Erfahrung in einer Programmiersprache und den eingesetzten Werkzeugen spricht eine Einflussnahme von bis zu 40% auf den Aufwand. Die Effizienz der Programmiersprache sagt aus, dass Sprachen wie Java und C# produktiver sind als C und Cobol. Durch die Wahl einer bestimmten Programmiersprache stehen bestimmte Werkzeuge und Frameworks zur Verfügung, sofern diese keinen ausreichenden Support genießen, kann der Aufwand um bis zu 50% steigen. Interpretierbare Programmiersprachen werden oftmals als bis zu zweimal so produktiv als ihre kompilierbaren Verwandten gesehen. Trendowicz und Münch streichen ebenfalls die Wichtigkeit der Programmiersprache hervor und konnte einen signifikanten Einfluss auf die Produktivität feststellen.

4.7 Wiederverwendbarkeit

Trendowicz und Münch sehen in der Wiederverwendbarkeit von Software eine maßgebliche Errungenschaft zur Erhöhung der Produktivität in der Softwareentwicklung. Die gesteigerte Produktivität wirkt sich dabei nicht nur auf die Herstellung, sondern auch auf die Wartbarkeit der Software aus. Die weiteren positiven Effekte sind verringerter Aufwand beim Erstellen und Testen fundamentaler Funktionen für ein neues Softwareprojekt. Als Nachteil wird gesehen, wenn es noch keine Investition in die Wiederverwendbarkeit von Software gesetzt wurde, dann ist der initiale Aufwand für die Herstellung universeller Module ein nicht zu unterschätzender Aufwand. Bei kleinen Repositorien ist die Effizienzsteigerung durch die Verwendung von wiederverwendbarer Software noch gering, wird aber bei zunehmender Größe der Repositorien entscheidend höher. Um die Wiederverwendbarkeit erfolgreich durchführen zu können, sind noch weitere Faktoren relevant. Diese Faktoren sind unter anderen die eingesetzten Mitarbeiterinnen und Mitarbeiter, die Verwendung von objekt-orientierten Programmiersprachen und das Level der Wiederverwendbarkeit. Es ist möglich neben internen Bibliotheken für die

Wiederverwendbarkeit auch außerhalb der eigenen Organisation erstellte Bibliotheken zu verwenden.

5 PROZESSMODELL ZUR AUFWANDSSCHÄTZUNG

Die im vorigen Kapitel Einflussfaktoren auf die Aufwandsschätzung werden in diesem Kapitel durch ein Prozessmodell operationalisiert und wirken sich auf das Ergebnis des Prozessmodells aus.

5.1 Aufbau eines Prozessmodells

Trendowicz und Jeffrey (2014) stellen klar das Irren menschlich ist, aber aus Fehlern gelernt werden soll. Der Prozess der Aufwandsschätzung wird von den beiden Autoren in einen Kontext gesetzt (siehe Abbildung 4). Als Eingabe für den Prozess der Aufwandsschätzung werden Objekte und quantitative Daten von historischen Projekten gesehen. Die Ressourcen für die Aufwandsschätzung werden durch Personen, die die Aufwandsschätzung durchführen, die Zeit, die dafür benötigt wird, und die verwendeten Werkzeuge angegeben. Der Begriff Aufwandsschätzung steht für den Prozess selbst und die angewendeten Methoden und Prozeduren. Dies alles befindet sich in einem Kontext, die Aufwandsschätzung positiv als auch negativ beeinflussen kann. Die Ausgabe des Prozesses stellt die Aufwandsschätzung oder ein Schätzmodell dar. Im Anschluss an die Aufwandsschätzung und des Publizierens der Ausgabe kann diese von Personen verwendet werden. Wenn mit verschiedenen Methoden und Prozeduren gearbeitet wurde, können Projektentscheidungen (Best-Case, Worst-Case, Ressourcenplanung) getroffen werden.

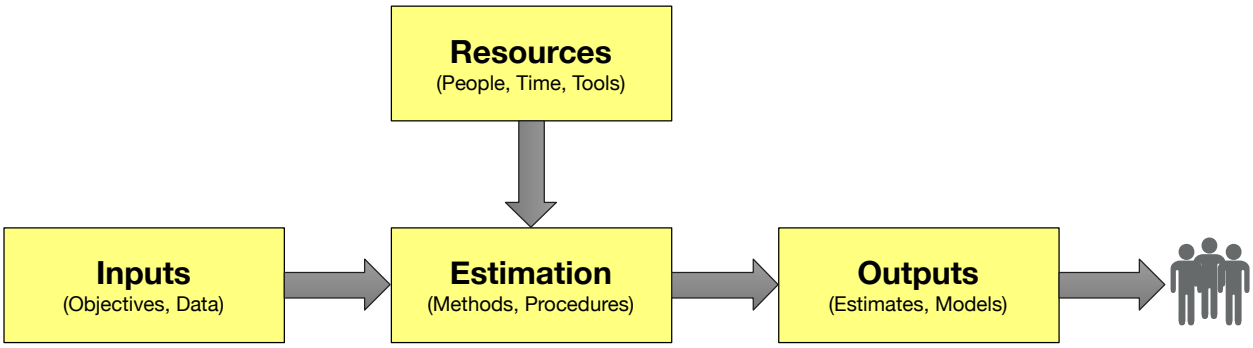

Abbildung 4 Estimation context. Nachgezeichnet von Software Project Effort Estimation (S. 402), von A. Trendowicz und R. Jeffery, 2014, Kaiserslautern, Sydney: Springer

Die vorliegende Beschreibung des Ablaufes der Aufwandsschätzung soll im nachfolgenden Teil dieser Arbeit als Ausgangspunkt für das Prozessmodell dienen. Das Prozessmodell soll im Bereich der Eingaben und Ressourcen verschiedene Ansätze unterstützen und die Einflussfaktoren aus den Kapiteln zuvor berücksichtigen. Für den Prozess des Aufwandsschätzung selbst soll ein mehrstufiger Ablauf entwickelt werden, der sowohl agile Softwareprojekte mit geringem Aufwand als auch großem Aufwand unterstützt. Die dafür

benötigen Rahmenbedingungen werden im empirischen Teil dieser Arbeit erarbeitet und fließen dann in einer neuen Iteration in das Prozessmodell ein. Bevor das Prozessmodell erstellt wird, wird das vorliegende Softwareentwicklungsmodell beleuchtet, sowie die Einheit der Aufwandsschätzung festgelegt.

5.2 Der Softwareanforderungsprozess

Um das Prozessmodell erstellen zu können, wird nachfolgend zunächst der Softwareanforderungsprozess bei Pankl Racing Systems AG als Prozessflussmodell dargestellt (siehe Abbildung 5) und beschrieben.

Der Prozess einer Softwareanforderung kann verschiedene Auslöser haben, in den meisten Fällen handelt es sich um:

- Ideen,
- Rechtliche Anforderungen,
- Auditvorgaben,
- oder Verbesserungsvorschlägen aus dem KVP.

Aus den oben genannten Auslösern wird eine Softwareanforderung in Form eines Lastenheftes erstellt und über den innerbetrieblichen Business Applications Service Desk zur weiteren Verarbeitung als Change Request angelegt. Die Anforderung wird im Anschluss durch den zuständigen Product Owner auf Vollständigkeit geprüft und entsprechend nochmals zur Überarbeitung an die anfordernde Einheit übermittelt oder in weiterer Folge kategorisiert. Die Überprüfung auf Vollständigkeit schließt folgende Kriterien ein:

- Betroffene Geschäftsprozesse beschrieben
- Änderungen an Geschäftsprozessen dokumentiert
- Anforderungen an die Software beschrieben
- Abnahmekriterien definiert
- Betroffene Key-User informiert

Die Kategorisierung der Anforderung erfolgt nach vier Kriterien, die aufgrund des beschriebenen Vollständigkeitsgrades bestimmt werden können:

- Prozessänderung
- Komplexität
- Technische Herausforderung
- Organisation

Das Kriterium Prozessänderung beinhaltet die aktive Änderung von bestehenden Geschäftsprozessen durch die Softwareanforderung. Da in diesen Fällen umfangreiche

Prüfungen durchgeführt, um keine in Verbindung stehenden Geschäftsprozesse zu beeinträchtigen. Die Komplexität bezieht sich unter anderem auf die Integration der Softwareanforderung in die Produktivumgebung. Als Beispiel für dieses Kriterium kann die Integration einer Softwareimplementierung genannt werden, die durch eine Schnittstelle zu einem anderen System/zu einer Anlage als komplex zu bezeichnen ist. Das Kriterium der technischen Herausforderung zielt darauf ab, die Neuanforderung hingehend der eingesetzten Technologien und den damit verbunden Herausforderungen zu bewerten. Wenn für die Umsetzung eine neue Programmiersprache, ein neues Framework oder eine neue Schnittstelle zu einem noch nicht angebundenen System geschaffen wird, kann dies eine technische Herausforderung darstellen. Die Organisation beschreibt die Notwendig im Laufe des Softwareprojektes organisatorische Aufgaben in Form eines abgeschwächten Projektmanagements zu betreiben oder. durch einen Projektmanager unterstützt zu werden.

Aus den angeführten Kriterien wird abgeleitet, ob die Neuanforderung ein minimaler Eingriff in das bestehende System ist und damit als „Task“ ohne weitere Prozesse in der Anforderungsphase durch den Service Desk bereitgestellt wird. Im anderen Fall erfolgt die Kategorisierung als Umsetzung im Projektteam und wird einer Aufwandsschätzung unterzogen. Es wird derzeit auf Basis von Expertenwissen der Aufwand einer Neuanforderung abgeschätzt. Die Aufwandsschätzung soll einem standardisierten Prozess folgen. Um dies zu realisieren, wird in den nächsten Abschnitten ein Prozessmodell entwickelt und evaluiert.

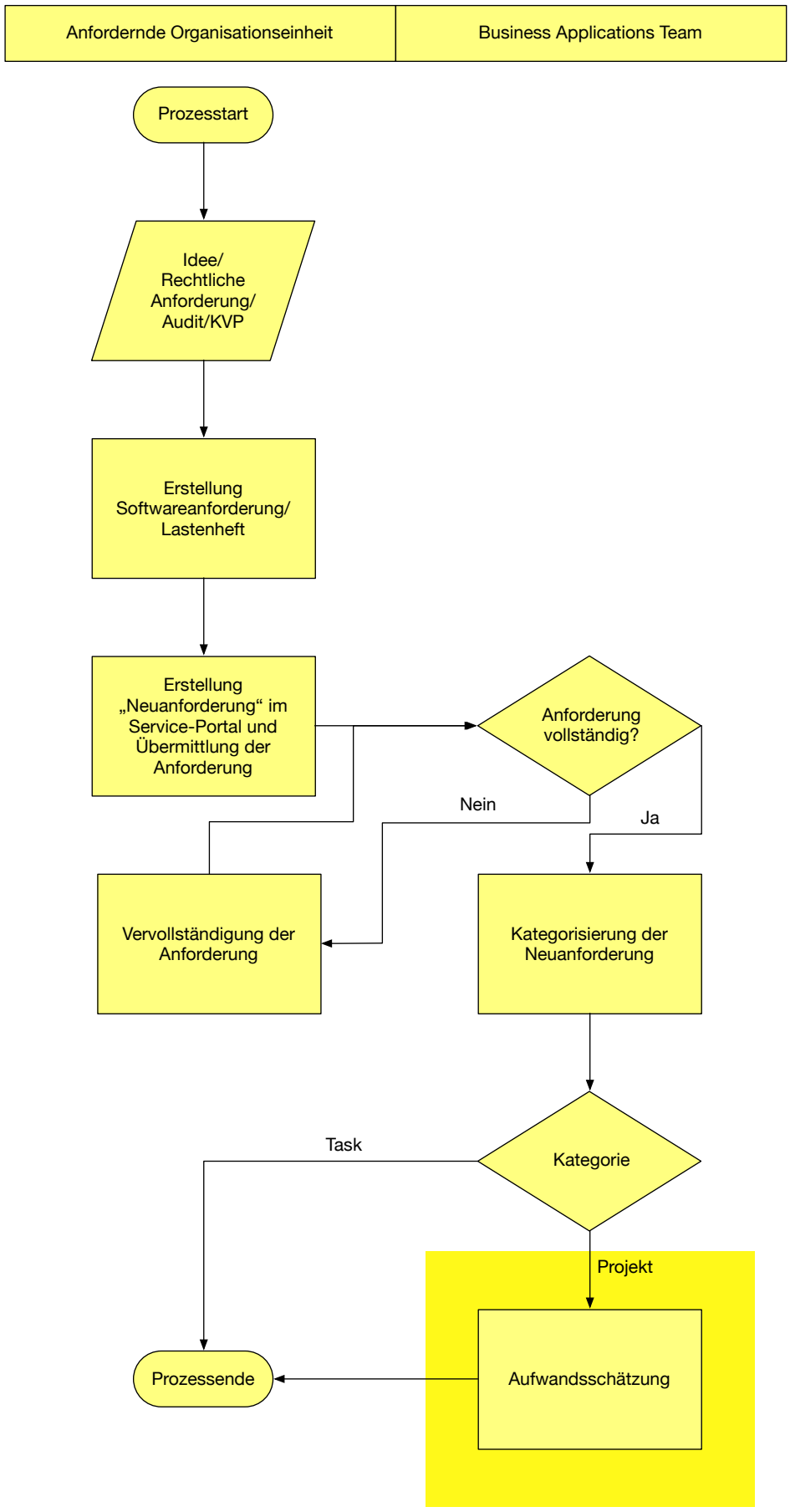

Abbildung 5 Softwareanforderungsprozess Pankl Racing Systems AG (eigene Darstellung)

5.3 Das Prozessmodell

Im nachfolgenden Teil der Arbeit wird auf Basis der Erkenntnisse aus den vorherigen Kapiteln ein Prozessmodell (siehe Abbildung 6) erstellt.

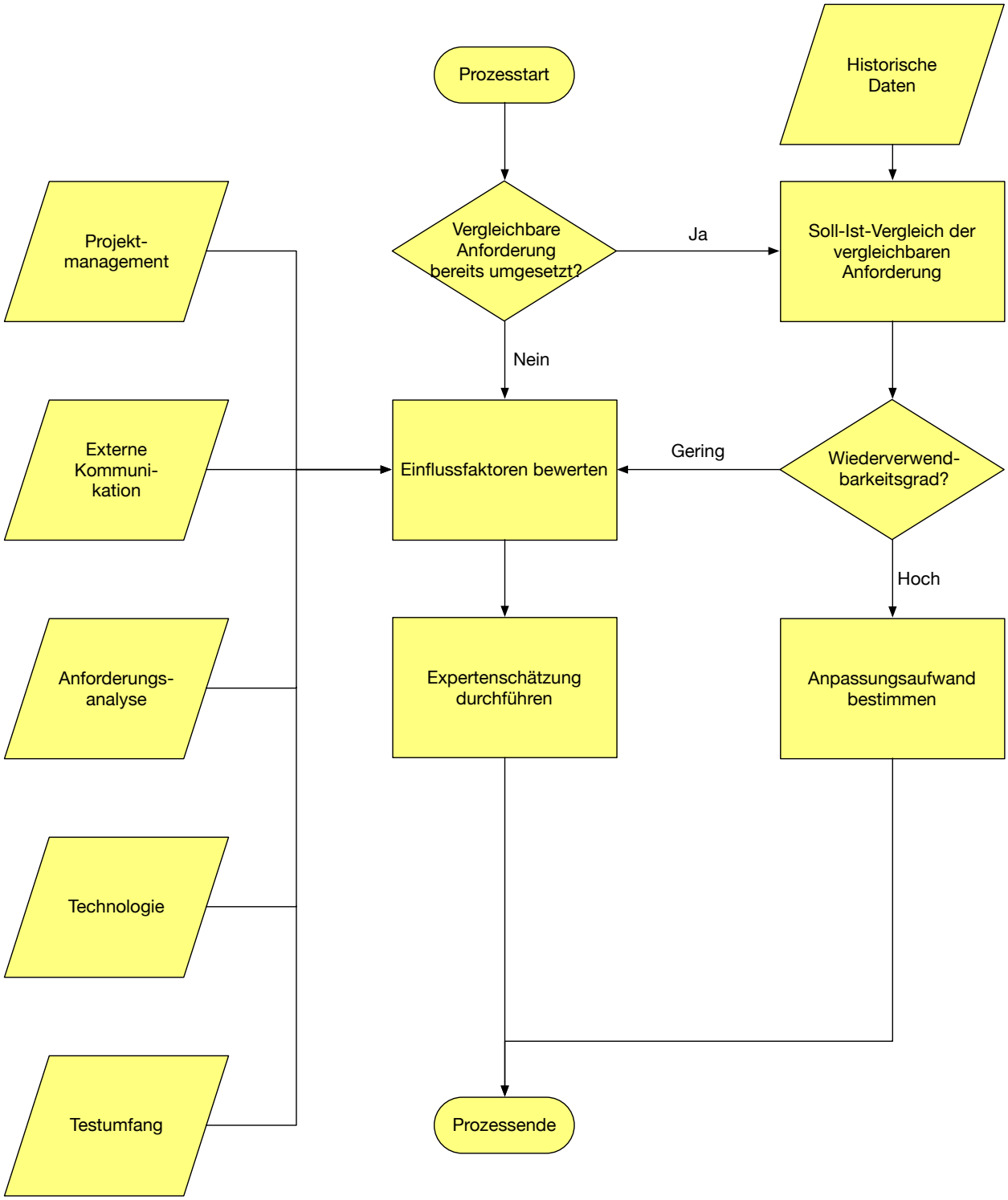

Abbildung 6 Prozessmodell zur Aufwandsschätzung von agilen Softwareprojekten (eigene Darstellung)

5.3.1 Eingangsgrößen

Als Eingangsgröße des Prozesses zur Aufwandsschätzung eines agilen Softwareprojektes dienen alle Informationen, die verfügbar sind. Ein Teil der Informationen wird dem Lastenheft bzw. der Anforderungsbeschreibung entnommen, als weiteren Teil kann das Domänenwissen des Product Owners angesehen werden. Im Gegensatz zur bisherigen Herangehensweise wird durch Anwendung des Prozessmodells nicht mit den zuvor genannten Informationen alleine die Aufwandsschätzung betrieben. Es werden die im nachfolgenden Teil angeführten Ressourcen durch Anwendung einer Checkliste eingeschätzt und ermöglichen damit eine strukturierte Verarbeitung der Eingangsgrößen.

5.3.2 Ressourcen

In Hinblick auf die Ressourcen soll das Prozessmodell den besonderen Nutzen verfolgen, dass es Ressourcen als Unterstützung für die Expertenschätzung zur Verfügung stellt. Die Ressourcen sollen einerseits historische Daten darstellen und andererseits eine strukturierte Bewertung der Kriterien, welche als Erkenntnisse der Einflussfaktoren auf die Aufwandsschätzung verstanden werden können, ermöglichen. Sofern noch keine historischen Daten im betroffenen Unternehmen gesammelt werden, ist eine Erfassung dieser Daten für zukünftige Aufwandsschätzungen anzustreben. Es eignet sich eine systemische Erfassung der Daten besonders gut, da die Möglichkeit besteht, durch Verwendung von Schlagwörtern eine eingegrenzte Auswahl an potenziellen Referenzprojekten zu erhalten. Neben dem geschätzten Aufwand ist auch das Erfassen des tatsächlichen Aufwandes anzuraten, da dies einen Soll-Ist-Vergleich zwischen Schätzung und tatsächlichem Aufwand ermöglicht. In weiterer Folge kann durch den Soll-Ist-Vergleich auch gezielt nach unter-/überschätzten Projekten gesucht werden und diese näher analysiert werden. Dies kann wiederum in eine Guideline oder eine Schulung zum Thema Aufwandsschätzung einfließen.

Die Erfassung von Softwareprojekten und damit auch die Möglichkeit zur Auffindung von historischen Daten wird bei Pankl Racing Systems AG mit der Software Jira[2] von Atlassian durchgeführt. In der Software werden für jeden Vorgang, der im Zuge des Softwareprojektes umgesetzt werden muss, eine ursprüngliche Aufwandsschätzung erfasst und in weiterer Folge bei der Abarbeitung auch die Zeit verfolgt, die für die Durchführung des Vorganges aufzuwenden war. Auf Basis dieser Daten wird ein Soll-Ist-Vergleich auf Vorgangs- und Projektebene dargestellt. Dadurch ist beim Auswerten und Auffinden von Auffälligkeiten zwischen Schätzung und tatsächlichem Aufwand ein Drill-Down möglich.

[2] https://de.atlassian.com/software/jira

5.3.3 Expertenschätzung

In diesem Abschnitt wird die Verwertung der Ressourcen für die Expertenschätzung beschrieben. Das Ziel dieses Prozessschrittes ist es eine Aufwandsschätzung durchzuführen, die als Ergebnis des gesamten Prozesses an nachgelagerte Prozesse übergeben werden kann.

Die Expertenschätzungen, die derzeit bei Pankl Racing Systems AG durchgeführt werden, verwenden als Einheit ideale Stunden bzw. Tage. Es existiert keine Vorgabe, ob die Angabe des Aufwandes in Stunden oder Tage gemacht werden muss. Jedenfalls ist festgelegt, dass ein Personentag acht Stunden entsprechen. Die Wahl auf diese Einheit ist damit zu begründen, dass sie direkte Transparenz für die Stakeholder bietet und ohne weitere Kalkulation die Aufwandsschätzung kommuniziert werden kann. Da die Verwendung von Story Points zukünftig nicht ausgeschlossen ist, ist die Erstellung des Prozessmodells unter der Prämisse durchgeführt worden, dass es einheitenunabhängig eingesetzt werden kann.

Die Schätzung wurde bisher immer als Expertenschätzung durchgeführt und es erfolgte keine strukturierte Verwendung von Ressourcen. Es wurde eine Bewertung des Lastenheftes vorgenommen und auf Basis dessen ein Aufwandsschätzung durchgeführt. Es erfolgt eine Schätzung einer Expertin oder eines Experten, diese konnten sich innerhalb der Abteilung durch Erfahrungsaustausch unterstützen.

Im entwickelten Prozessmodell wird der Prozess weiterhin von einer Person durchgeführt, allerdings wird die Aufwandsschätzung durch das Prozessmodel einer strukturierten Arbeitsgrundlage unterworfen. Die erste Entscheidung, die durch die Person getroffen werden muss, ist zu bestimmen, ob eine ähnliche Anforderung bereits umgesetzt wurde. Dieser Prozessschritt wird durch die Nutzung einer Datenbank (im Fall von Pankl Racing Systems durch Jira) unterstützt. Die Datenbank bietet die Möglichkeit nach Begriffen oder nach Schlagworten zu suchen. Zu den gefundenen Treffern wird aus den historischen Daten der Titel, die Beschreibung, die damalige Aufwandsschätzung, die tatsächlich benötigte Arbeitszeit sowie daraus berechnet ein Soll-Ist-Verhältnis angezeigt. In einer Detailansicht besteht die Möglichkeit den gesamten Projektverlauf durch Kommentare und Verknüpfungen zwischen einzelnen Vorgängen zu analysieren. Sofern eines oder mehrere Softwareprojekte einen hohe Wiederverwendbarkeitsgrad mit dem zu schätzenden Softwareprojekt aufweisen, wird der Anpassungsaufwand eruiert. Zu den Tätigkeiten bei der Anpassung eines Softwareprojektes ist zunächst zu bestimmen, ob eine vollständige Wiederverwendbarkeit der Version möglich ist, wenn dies der Fall ist, dann ist der Anpassungsaufwand gering. Sofern Anpassungen an der Version notwendig sind, muss analysiert werden, ob die Anpassung am bereits bestehenden Modul durchgeführt wird und somit eine gemeinsame Version des Moduls/der Programmierung realisierbar ist. Wenn dies nicht der Fall ist, dann müssen die beiden Versionen getrennt werden, wodurch in weiterer Folge der Wartungsarbeit (z.B. bei der Implementierung von Fehlerbehebungen oder gemeinsamen Features) erheblich steigt. Wenn nach der Bewertung des Wiederverwendbarkeitsgrades erkannt wird, dass dieser gering ist, dann erfolgt die Aufwandsschätzung in gleicher Art und Weise einer Aufwandschätzung, die durchgeführt wird, wenn keine ähnliche Anforderung vorhanden ist. Die Schätzung für den Fall, dass keine vergleichbare historische Anforderung in der Datenbank gefunden werden kann, verwendet die

im vorherigen Abschnitt diskutierten Ressourcen und bringen diese als Faktoren in die Aufwandsschätzung ein.

Ein besonders gelagerter Fall stellt eine Softwareneuanforderung dar, die weder mit einer historisch vergleichbaren Anforderung verbunden noch im Zuge des alternativen Weges einer Aufwandsschätzung zugeführt werden kann. Für diesen Fall tritt ein Schätzungskomitee aus mindestens drei Personen, die innerhalb des Business Applications Teams eine Produktverantwortung tragen, zusammen und die Aufwandsschätzung wird in diesem Komitee durchgeführt. Die Anwendung dieses Komitees soll vor allem Neuanforderungen abdecken, die einen vermutlich hohen zeitlichen sowie auch finanziellen Aufwand für das Unternehmen bedeuten können.

Aus der angeführten Vorgehensweise ist ersichtlich, dass das Prozessmodell der hybriden Methodik zuzuschreiben ist und durch die daten-basierenden Elemente versucht, historische Daten möglichst gut zu verwerten, um die Qualität der Aufwandsschätzung zu erhöhen. Sofern keine historischen Daten verfügbar sind, werden die erhobenen Einflussfaktoren bewertet.

5.3.4 Ergebnis

Als Ergebnis des durchgeführten Prozessmodells ist eine Aufwandsschätzung über die Tätigkeiten, die für die Durchführung der beschriebenen Softwareneuanforderung notwendig sind, in transparenter, strukturierter Form entstanden. Zum Zeitpunkt der Verfassung dieser Arbeit wird die Aufwandsschätzung in idealen Tagen oder Stunden angegeben und kann dadurch direkt an die Stelle kommuniziert werden, die den Auftrag zur Softwareneuanforderung erteilt hat.

In Hinblick auf die agile Arbeitsweise bei Pankl Racing Systems AG und dem Thema Story Points erfolgt nachfolgend ein Muster wie auch auf Basis von Story Points eine Aufwandsschätzung an abteilungsfremde Stellen kommuniziert werden kann. Zugrunde einer solchen Berechnung liegt immer die Velocity (siehe 3.4.2 Story Points). Als Durchrechnungszeitraum dient in der Softwarevorgehensmethode Scrum ein Sprint, wobei die Länge eines Sprints wählbar ist, aber definiert sein muss. Da bei Pankl Racing Systems AG im Bereich Business Applications ohne Sprints gearbeitet wird, sonstige Elemente aus Scrum und Kanban eine agile Methodik ergeben, würde die Berechnung der Velocity durch „virtuelle“ Sprints möglich sein. Als virtuellen Sprint dient ein Durchrechnungszeitraum von einem Monat und es wird betrachtet, wie viele Story Points in diesem Monat durchgeführt werden können. Dies wird in Form einer tabellarischen Auflistung durchgeführt.

Issue ID	Summary	Story Points
1	Ein User möchte…	11
2	Ein Key-User führt…	13
3	Als Admin ist…	5

Tabelle 2 Exemplarische Velocity (eigene Tabelle)

Im angeführten Beispiel (siehe Tabelle 2) sind die angeführten Storys fertiggestellt worden und damit wurde eine Velocity von 29 erreicht. Als Beispiel kann dadurch abgeleitet werden, dass eine zukünftige Softwareneuanforderung mit 25 Story Points innerhalb des Durchrechnungszeitraums von einem Monat implementiert werden könnte.

6 EMPIRISCHE UNTERSUCHUNG

In der empirischen Untersuchung des Prozessmodels wird es Expertinnen und Experten im Bereich Softwareentwicklung bei Pankl Racing Systems AG vorgestellt und diskutiert. Es wird ein halbstrukturiertes Interview mit offenen Fragen durchgeführt und die Möglichkeit beleuchtet das Prozessmodel durch Entwicklung eines Fragebogens zu unterstützen. Der erste Teil dieses Kapitels steht daher im Fokus des Experteninterviews und die Beschreibung der daraus gewonnenen Erkenntnisse. Im zweiten Teil wird das Thema Fragebogen diskutiert und vertiefend einen generischen Fragebogen entwickelt.

6.1 Experteninterview

Die Entscheidung ein qualitatives Interview zu führen und keine quantitative Befragung durchzuführen, ist damit zu begründen, dass die offenen Fragen lediglich als Leitfaden dienen sollen und die Expertinnen und Experten somit die Möglichkeit haben das Interview mitzugestalten. Sofern durch eine Frage ein Themenfeld eröffnet wird und dieses vertiefend diskutiert wird, soll dies nicht durch geschlossene Fragen eingeschränkt sein. Eine qualitative Befragung hätte auch durch ein unstrukturiertes Interview durchgeführt werden können. Im Rahmen dieser Arbeit wurde diese Möglichkeit nicht in Betracht gezogen, da die Strukturierung über das Werkzeug eines Leitfaden unverzichtbar war, damit sichergestellt werden konnte, dass im Rahmen des Interviews alle wesentlichen Themen, die nach der Literaturrecherche im ersten Teil der Arbeit in das Prozessmodell übernommen wurden.

In der nachfolgenden Tabelle (siehe Tabelle 3) werden die durchgeführten Interviews aufgelistet.

Nr.	Name Interviewpartner	Datum	Unternehmen	Position
1	DI (FH) Birgit Thek	15.05.2019	Pankl Racing Systems AG	Head of Business Applications
2	Dipl.-Ing. Christian Thalmann	11.06.2019	Pankl Racing Systems AG	Product Owner Systems

Tabelle 3 Auflistung durchgeführter Interviews (eigene Tabelle)

6.1.1 Leitfaden für das Interview

Der Leitfaden für das Interview (siehe Tabelle 4) wurde wie bereits im vorigen Abschnitt erläutert, aus den Erkenntnissen der Literaturrecherche abgeleitet.

Abschnitt	Beschreibung
Briefing	Vorstellung der Masterarbeit Aufnahmeerlaubnis Erklärung zum Ablauf des Interviews
Einführung in das Thema	Vorstellung des Prozessmodells • Einordnung der Aufwandsschätzung in den Prozess „Softwareneuanforderung“ • Prozessmodell zur Aufwandsschätzung
Methoden der Aufwandsschätzung	Diskussion der Ergebnisse aus der Literaturrecherche: • Daten • Experten • Hybrid
Einflussfaktoren auf die Aufwandsschätzung	Diskussion der Ergebnisse aus der Literaturrecherche: • Anforderungsanalyse • Irrelevante und irreführende Informationen • Maß und Einheit der Schätzung • Softwaretests • Menschliche Faktoren und Teamstruktur • Technologie • Wiederverwendbarkeit
Fragebogen für das Prozessmodell	• Wie kann eine standardisierte Aufwandsschätzung mit dem Prozessmodell auf Basis eines Fragebogens durchgeführt werden? • Welche Kriterien zur Aufwandsschätzung sollen enthalten sein? • Wie könnte eine Gewichtung der Kriterien dargestellt werden? • Wie kann das Prozessmodell etabliert werden?

Tabelle 4 Leitfaden für das Interview (eigene Tabelle)

Im ersten Abschnitt des Interviews erfolgt die Vorstellung des Masterarbeitsthemas und die organisatorischen Themen der Aufnahmeerlaubnis sowie die Beschreibung des groben Interviewablaufes.

Im Abschnitt „Einführung in das Thema“ wird die Eingliederung des Prozessmodells in den Softwareneuanforderungsprozess erläutert und das Prozessmodell (siehe 5.3 Das Prozessmodell) vorgestellt. In diesem Abschnitt des Interviews erfolgt eine detailreiche Erklärung des Prozessmodells, um in den weiteren Abschnitten Querverweise zum entwickelten Prozessmodell herstellen zu können.

Im darauffolgenden Teil des Interviews werden die Unterschiede zwischen Daten- und Experten-getriebenen Schätzungen erläutert. Es wird auf die Möglichkeit hingewiesen, dass die Verwendung von historischen Daten nur möglich ist, wenn eine mit entsprechender Sorgfalt gewartete Datenbasis vorliegt. Die hybriden Methoden werden als Vereinigung aus den besten Eigenschaften der beiden zuvor genannten Methoden beschrieben. Es erfolgt ein Querverweis auf das entwickelte Prozessmodell, welches sich auch aus beiden Methoden bedient und somit eine hybride Vorgehensweise darstellt.

Die Einflussfaktoren im nächsten Teil des Interviews beziehen sich auf die gefundenen Faktoren in der Literaturrecherche (siehe 4 Einflussfaktoren auf den Aufwand). Es wird jeder Faktoren mit den Expertinnen und Experten diskutiert, um die Erkenntnis zu erlangen, in wie Fern dieses Thema aus ihrer allgemeinen Erfahrung in der Praxis relevant ist, als auch im speziellen den Grad der Beeinflussung durch die Faktoren bei Pankl Racing Systems AG zu bestimmen.

Im letzten Abschnitt des Interviews ist das zentrale ein Fragebogen zur Unterstützung des Prozessmodells. Die Fragen, die es zu beantworten gilt, sind grob eingeteilt, wie können die Einflussfaktoren als Kriterien überführt werden, ist eine Gewichtung möglich und wie kann ein solcher Fragebogen in den Prozess integriert werden. Bei diesen Fragen ist einerseits die Betrachtung auf Pankl Racing Systems AG wichtig, um einen Fragebogen entwickeln zu können, der die Bedürfnisse des Unternehmens abdeckt, andererseits muss der Fokus auch auf die Verwendbarkeit der Fragen in anderen Unternehmen gerichtet sein, um die Erkenntnisse auch für andere Unternehmen verwerten zu können.

6.1.2 Erkenntnisse aus den Interviews

In diesem Teil der Arbeit werden die Erkenntnisse aus den durchgeführten Interviews beschrieben und im darauffolgenden Teil bei der Erarbeitung des Fragebogens angewendet.

Das Interview mit DI (FH) Birgit Thek, Leiterin der Abteilung Business Applications bei Pankl Racing Systems AG, beleuchtet das Thema dieser Masterarbeit mit Erfahrungen aus der Softwareentwicklungspraxis (persönliche Kommunikation, 15. Mai, 2019). Sie sieht als größten Einflussfaktor die Anforderungsanalyse, denn nur klar formulierte und nicht irreführende Anforderungsdokumente sind eine stabile Basis für die Aufwandsschätzung. Großteils sind die Lastenhefte bei Pankl Racing Systems AG in einem sehr guten Detailierungsgrad geschrieben und beinhalten alle notwendigen Angaben zur Umsetzung des Softwareprojektes. Ein Punkt,

der noch deutlicher formuliert und dezidiert angeführt werden sollte, sind Abnahmekriterien. Aus ihrer Sicht könnten dadurch auch Testfälle für die Softwaretests abgeleitet werden. Die formulierten Eingangsgrößen für das Prozessmodell sollen nicht um weitere Einflussfaktoren ergänzt werden und decken die Anforderungen bei Pankl Racing Systems AG ab. Für unsichere Projekte, die als Vorstufen von konkreten Projekten mit Lastenheft durchgeführt werden, ist die Aufwandsschätzung schwierig durchführbar. Es handelt sich hierbei oftmals auch um einen Proof of Concept, um danach ein konkretes Projekt starten zu können. Diese Projekte sollen nicht durch ein standardisiertes Prozessmodell abgeschätzt werden. Als Erweiterung des grundsätzlichen Softwareentwicklungsprozesses bei Pankl Racing Systems AG kann die Einführung einer nachgelagerten Aufwandsschätzung verstanden werden. Diese würde wie bisher nach Eingang eines vollständigen Lastenheftes und Klassifizierung als Projekt eine Aufwandsschätzung vorsehen, sowie nachgelagert nach der Konzptphase eine erneute Aufwandsschätzung vorsehen, um bei großen Differenzen die Auswirkungen in der Projektplanung darzustellen. Die korrekte Abschätzung des Aufwandes für die Durchführung der Softwaretests erscheint oftmals aufgrund der fehlenden Abnahmekriterien als schwierig und wird oftmals unterschätzt. Die Einheit der Aufwandsschätzung mit idealen Personentagen ist vor allem für die Kommunikation mit Stakeholdern komfortabel, allerdings sollen zukünftig Versuche unternehmen werden, um Story Points für Pankl Racing Systems AG zu evaluieren. Als wichtiges Erfolgskriterium für die Einführung des Fragebogens wird die Integration in die vorhandenen Softwareprojektmanagement-Tools (Anmerkung: Jira bei Pankl Racings Systems AG) angesehen. Ein positiver Einfluss eines standardisierten Prozessmodells auf Aufwandsschätzungen wird bestätigt.

In einem weiteren Interview wurden das Prozessmodell und die Entwicklung des Fragebogens mit Dipl.-Ing. Christian Thalmann besprochen. Er ist Product Owner in der Abteilung Business Applications und die Aufwandsschätzung agiler Softwareprojekte ist eines seiner Aufgabengebiete (persönliche Kommunikation, 11. Juni, 2019). Er sieht in der Anforderungsanalyse ebenfalls hochwertige Lastenhefte, die eine Aufwandsschätzung ermöglichen. Als Gefahrenpotenzial sieht er bei fehlenden Informationen das Domänenwissen der Product Owner, welches möglicherweise zu Interpretationen führt, die nicht korrekt sind. Anstelle der Interpretation sollte eine Nachbesserung des Lastenheftes angefragt werden. Ein zentrales Thema ist für ihn die Aufwandsschätzung im Bereich des Einsatzes neuer Technologien, weil diese oftmals aus mangelnder Erfahrung noch nicht abschätzbar sind und bevor ein produktives Arbeiten mit der neuen Technologie möglich ist, bereits erheblicher Aufwand anfällt. In diesem Zusammenhang wird auch die Notwendigkeit angeführt, dass Technologien und Funktionen, die zukünftig nicht mehr gewartet werden, nicht mehr verwendet werden sollen, um den Wartungsaufwand nach Projektabschluss zu verringern. Dies ist besonders im Zusammenhang mit der Wiederverwendbarkeit von Software zu berücksichtigen. Es wird neben der Wiederverwendbarkeit von Software auch für die Aufwandsschätzung auf Erfahrungen aus historischen Projekten zurückgegriffen, allerdings erfolgt dies nur unter der Maßgabe des Product Owners und unterliegt keinem standardisierten Prozess. In Hinblick auf den Fragebogen soll der Detailierungsgrad nicht zu hoch sein, sondern als Gedankenstütze bei der Beurteilung der Einflussfaktoren auf die Aufwandsschätzung eine Unterstützung anbieten.

Die Integration in ein bestehendes Tool, um eine einfache Bedienung und keinen Systembruch zu erfahren, wird als besonders wichtig angesehen. Dass ein standardisiertes Prozessmodell zu einer Verbesserung der Aufwandsschätzung führt, wird bestätigt.

Die Erkenntnisse aus den Interviews werden im nächsten Abschnitt in die Fragebogenentwicklung einfließen, damit dieser im Unternehmen eingesetzt werden kann. Die Erkenntnisse in Hinblick auf das Prozessmodell werden in den Schlussfolgerungen berücksichtigt.

6.2 Einsatz im Unternehmen

Im Unternehmenseinsatz wurde daran gearbeitet, einen Fragebogen aufzubauen, der die Einflussfaktoren aus der Literaturrecherche mit den Erkenntnissen aus den Experteninterviews in Verbindung setzt.

Projektmanagement	
	Projektmanagement innerhalb der Abteilung?
Externe Kommunikation	
	Werden externe Ressourcen benötigt?
Anforderungsanalyse	
	Abnahmekriterien der Software definiert?
	Mehrdeutigkeiten in der Anforderung vorhanden?
Technologie	
	Sind Einschränkungen bei der Verwendung von bekannten Technologien zu erwarten?
	Werden unbekannte Technologien für die Umsetzung benötigt?
Testumfang	
	Sind neben den Standardtests erweiterte interne Tests notwendig?
	Sind Tests durch/mit externen Teilnehmern notwendig?

Tabelle 5 Fragebogen zur Aufwandsschätzung (eigene Tabelle)

Für jedes Kriterium wird angegeben als wie stark der Einfluss auf das Softwareprojekt zu erwarten ist. Dies wird in den Abstufungen „gering“, „mittel“ und „hoch“ durchgeführt. Um eine operationalisierte Aussage in Hinblick auf die Aufwandsschätzung zu erhalten, wird die Schätzung für den reinen Softwareaufwand angegeben und dieser wird auf Basis der Abstufungen prozentuell verändert. Für einen geringen Einfluss wird kein Aufschlag angeführt. Für einen mittleren und hohen Einfluss erfolgt ein Aufschlag nach Maßgabe der Expertenschätzung. In weiterer Folge ist es denkbar, dass dieser Aufschlag einem spezifizierten Wert zu Grunde gelegt wird und nicht mehr individuell angeführt wird.

Das Kriterium Projektmanagement soll vor allem den organisatorischen Aufwand in einem Projekt widerspiegeln. Sofern das Softwareprojekt nahezu keinen Aufwand im klassischen Projektmanagement verursacht, wird die Aufwandsschätzung nicht beeinflusst. Im Gegenteil dazu kann der Fall eintreten, dass im Zuge der Softwareentwicklung auch klassische

Projektmanagementaufgaben übernommen werden und diese den Aufwand bzw. dadurch auch den Fertigungsstellungstermin beeinflussen. Dieser Aufwand kann im Fall von Standardprojekten mit „mittel“ eingestuft werden, muss aber jedenfalls beachtet werden. Im Fall von „hoch“ soll das Projektmanagement durch das Projektmanagement Office durchgeführt werden. Da durch diese Maßnahme aber zahlreiche Projektmanagementaufgaben (z.B. regelmäßiges Projektcontrolling) durchgeführt werden müssen, ist auch im Bereich des Softwareprojektes der Aufwand mit einzukalkulieren.

Die externe Kommunikation ist ein essentielles Thema bei der Aufwandsschätzung, da es zwei Subkategorien gibt, die zu beachten sind. Externe Kommunikation ist immer dann notwendig, wenn für die Umsetzung des Softwareprojektes die Notwendigkeit von externen Ressourcen besteht. Es kann sich um die teilweise oder gänzliche Umsetzung des Softwareprojektes handeln. Die teilweise Umsetzung ist meist notwendig, wenn Anlagen oder Systeme gekauft werden und diese an vorhandene Business Applications Software angebundene werden soll. Die gänzliche Umsetzung ist möglich, wenn stark gekapselte Themen ohne Prozesstiefe umgesetzt werden sollen. Um den internen Aufwand, der durch die Verwendung von externen Ressourcen entsteht, abschätzen zu können, ist einerseits die bisher gemachte Erfahrung mit dem Lieferanten sowie andererseits die Anforderung selbst maßgeblich. Bei einer Kategorisierung in der Stufe „gering“ ist der interne Aufwand sehr gering, die Anforderungen werden einmalig übergeben, es sind nahezu keine Abstimmungen bis zur Übergabe des fertigen Softwareprojektes notwendig. Die mittlere Stufe würde zusätzlich umfangreiche Abstimmungen während der Projektphase vorsehen. In der höchsten Stufe fallen Lieferanten, die eine komplexe Integration umsetzen müssen, noch kein Domänenwissen bei Pankl Racing Systems AG haben und dadurch sowohl organisatorische als auch fachliche Betreuung benötigen.

Das Kriterium der Anforderungsanalyse zielt vor allem auf definierte Abnahmekriterien und der Wahrscheinlichkeit auf Mehrdeutigkeit in der Anforderung selbst ab. Definierte Abnahmekriterien stellen für die Durchführung eines Softwareprojektes nicht nur ein Qualitätskriterium dar, sondern lassen auch Rückschlüsse auf den notwendigen Aufwand, der notwendig ist, um die Kriterien zu erreichen, zu. Becker (2011) beschreibt den Weg von der User Story zu Abnahmekriterien als folgende fünf Schritte:

1. Schlüsselwörter identifizieren
2. Schlüsselwörter in Fragenkatalog einsetzen
3. Antworten finden, prüfen und diskutieren
4. Akzeptanzkriterien definieren
5. Testfälle spezifizieren

Als Beispiel auf Basis der Vorgehensweise von Becker beschrieben kann das Substantiv „Artikel“ herangezogen werden und das Vollverb, welches die Userkation beschreibt, ist „anlegen“. Mit diesen Schlüsselwörtern werden zahlreiche „W“-Fragen (wer, was, wann, wie häufig, …) definiert und im Schritt 3 beantwortet sowie hinterfragt. Aus diesen „W“-Fragen können die Abnahmekriterien definiert werden. Ein Beispiel auf den Artikel bezogen könnten

lauten: Es müssen immer die Felder Artikelnummer, Artikelbeschreibung und Preis bei der Neuanlage als Pflichtfelder angegeben werden. Aus diesen Abnahmekriterien können in weiterer Folge auch Testfälle durch die Softwaretesterinnen und -tester erstellt werden. Die zuvor beschriebene Vorgehensweise ist auch in der Lage den zweiten Fall der Mehrdeutigkeit einzuschränken. Durch das gezielte Verwenden der Schlüsselwörter im Fragenkatalog und der Diskussion werden Mehrdeutigkeiten erkannt und können korrekt formuliert werden. In Bezugnahme auf den Fragebogen und die Kategorisierung von „gering“, „mittel“ und „hoch“ ist für diesen Abschnitt festzuhalten, dass die Exploration der User Storys grundsätzlich immer zu einem geringen Einfluss führen soll. Für die Ausnahmen von dieser Regel muss der Grad der Einflussnahme bestimmt werden und dann dieses Kriterium entsprechend bewertet werden. Wenn ein Softwareprojekt umgesetzt werden muss, aber zu Beginn der Ausgang völlig unklar ist, weil es sich um ein experimentelles Softwareprojekt handelt, dann muss die Kategorie „hoch“ vergeben werden. Für Unklarheiten in den Details, die sich zum Beispiel auf die genaue Visualisierung oder das Design der Benutzeroberfläche beziehen, ist die mittlere Kategorie zu verwenden.

Das Thema Technologie ist in Hinblick auf die Aufwandsschätzung ein sehr gewichtiges Thema, da der Einsatz bewährter sowie bekannter Technologien eine hohe Effizienz bei der Umsetzung von Softwareprojekten garantiert. Sofern eine Technologie nicht mehr oder nur noch für eine begrenzte Zeit (z.B. zukünftige Systemaktualisierung) zur Verfügung steht, dann kann nicht auf bewährte Technologien zurückgegriffen werden. Wenn bei der Analyse der Anforderungsbeschreibung erkannt wird, dass die Durchführung des Softwareprojektes mit etablierten Technologien nicht möglich ist, dann muss einerseits eine Analyse bezüglich alternativer Technologien, die in der Lage sind, die Anforderungen zu realisieren, durchgeführt werden. Die Analyse selbst verursacht dabei bereits Aufwand und in der Aufwandsschätzung des Softwareprojektes selbst entsteht die Ungewissheit, wie hoch der Aufwand sein wird die neue Technologie zu erlernen und für das Projekt produktiv einzusetzen. Die Kategorisierung erfolgt anhand mehrerer Gesichtspunkte. Wenn eine Technologie für das Softwareprojekt zum Einsatz kommen soll, die bereits etabliert ist, dann ist die Beeinflussung jedenfalls als „niedrig“ zu kategorisieren. Wenn noch nicht etablierte Technologien verwendet werden müssen, dann muss der notwendige Umfang der Lernphase, vorhandene Vorkenntnisse der Abteilung und die Komplexität der Technologie bestimmt werden. Anhand dieser Kriterien wird bestimmt, ob für die Kategorisierung die Stufe „hoch“ vergeben werden muss.

Die Durchführung von Softwaretests ist essentiell, um Software mit einem Qualitätsniveau auszuliefern, dass die Kundinnen und Kunden zufrieden stellen. Der Aufwand hinter dementsprechenden Softwaretests ist sehr stark von der Softwareanforderung selbst sowie auch dem Einsatzgebiet der Software abhängig. In diesem Zusammenhang gilt es wie in den Erkenntnissen des Interviews angeführt auch das Thema der Risikoabschätzung für das Softwareprojekt zu berücksichtigen. Wenn ein Softwareprojekt sowohl eine hohe Schadensauswirkung als auch eine hohe Schadenswahrscheinlichkeit aufweist, ist dies in der Aufwandsschätzung für die Softwaretests zu berücksichtigen. Es ist zusätzlich zu berücksichtigen, ob der Softwaretest durch die Entwicklerinnen und Entwickler durchgeführt werden kann oder ob die Expertise eines ausstehenden Teammitglieds herangezogen werden

muss. Die Formulierung der Softwaretests orientiert sich an den Abnahmekriterien aus dem Lastenheft und wird dadurch von der Phase der Anforderungsanalyse beeinflusst. Je konkreter die Abnahmekriterien angeführt und spezifiziert sind, desto besser können sie als direktes Kriterium für die Durchführung der Softwaretests herangezogen werden. Dies bezieht sich vorwiegend auf die Abnahmetests und nicht auf Unit- oder Integrations-Tests.

7 SCHLUSSFOLGERUNGEN

Die verschiedenen Vorgehensweisen zur Aufwandsschätzung werden in drei Kategorien eingeteilt. Es handelt sich die Kategorien basierend auf Daten, basierend auf Expertenwissen sowie Hybrid. Die Schätzungen basierend auf Daten bedienen sich aus allgemein verfügbaren Datenbanken oder intern verfügbare historische Daten. Bei Schätzungen basierend auf Expertenwissen werden eine oder mehrere Expertenmeinungen eingeholt, wodurch eine Aufwandsschätzung erzielt wird. Die hybriden Vorgehensweisen vereinen die Vorteile aus den Schätzungen basierend auf Daten und Experten in eine Vorgehensweise.

Es existieren verschiedene Methoden zur Durchführung von Aufwandsschätzungen wie Wideband Delphi, Planning Poker, CBR, COCOMO und CoBRA. Diese Methoden können den oben angeführten Kategorien zugeordnet werden.

Aufwandsschätzungen können in verschiedenen Einheiten angegeben werden, diese können einen realen oder unrealen Bezug zu Zeiteinheiten darstellen. Es ist möglich die Aufwandsschätzung in klassischen Personenstunden oder -tagen sowie in nicht mit realen Einheiten verbindbaren Story Points oder Function Points anzugeben. Die letzteren genannten Einheiten können über eine Velocity und ein Burn-Down-Chart in einen zeitlichen Bezug gesetzt werden. Es wird dabei berechnet, wie wie viele Story Points oder Function Points in einem gegebenen Durchrechnungszeitraum implementiert werden können.

Die Forschungsfrage der vorliegenden Arbeit sollte ergründen welche Einflussfaktoren auf die Aufwandsschätzung von agilen Softwareprojekten existieren und wie diese Einflussfaktoren in einem standardisierten Prozessmodell dargestellt werden können. Zur Beantwortung der Forschungsfrage wurde eine Literaturrecherche durchgeführt und es wurden zusammenfassend die Einflussfaktoren:

- Anforderungsanalyse,
- irreführende und irrelevante Information,
- Maß und Einheit der Schätzung,
- Software Tests,
- menschliche Faktoren und Teamstruktur,
- Technologie und
- Wiederverwendbarkeit

ermittelt.

Diese Einflussfaktoren wurden in ein Prozessmodell überführt und damit eine standardisierte Vorgehensweise zur Aufwandsschätzung von Softwareprojekten definiert. Das Prozessmodell

sieht eingangs vor, dass die Softwareanforderung in Form eines schriftlich dokumentierten Lastenheftes vorliegt und daraus der schätzenden Person die Anforderungen bzw. die Abnahmekriterien an das Softwareprojekt bewusst sind. Das Prozessmodell sieht als ersten Prozessschritt vor, dass geprüft wird ob bereits eine ähnliche Anforderung umgesetzt wurde. Wenn diese Überprüfung positiv verläuft, wird in der Datenbank nach der bereits umgesetzten Anforderung gesucht und der historische Soll-Ist-Vergleich der Aufwandsschätzung herangezogen. Im nächsten Schritt wird geprüft, ob das bereits implementierte Modul wiederverwendet werden kann bzw. wie hoch der Anpassungsaufwand ist. Wenn eine Anpassung nicht möglich ist und eine Neuimplementierung durchgeführt werden muss, dann wird in weiterer Folge wie bei einer Neuanforderung ohne Ähnlichkeit zu einem historischen Softwareprojekt mit der Ausnahme, dass der historische Soll-Ist-Vergleich einen Einflussfaktor darstellt, vorgegangen. Die Einflussfaktoren aus oben angeführter Aufzählung werden für die Expertenschätzung im Prozessmodell als die Eingangsgrößen:

- Projektmanagement,
- externe Kommunikation,
- Anforderungsanalyse,
- Testumfang und
- Technologie

dargestellt.

Das Prozessmodell wurde in zwei Experteninterviews im Beispielunternehmen Pankl Racing Systems AG evaluiert und daraus ergab sich für die Anwendung die Notwendigkeit eines unterstützenden Fragebogens, der für die einzelnen Eingangsgrößen als Hilfestellung zur Ermittlung der Höhe des Einflusses dient. Dieser Fragebogen wurde exemplarisch für Pankl Racing Systems AG erstellt und ist universell auch für andere Unternehmen anwendbar, da er entsprechend erweitert oder eingegrenzt werden kann. Als wichtiges Kriterium für die praktische Einsetzbarkeit des Fragebogens wurde die Integration in die genutzten Tools zum Projektmanagement und der Projektvorgänge genannt. Bei Pankl ist dieses Tool Jira von Atlassian und es wird angestrebt das Prozessmodell zukünftig direkt in den Jira-Workflow zu integrieren.

In Hinblick auf die Hypothese der vorliegenden Arbeit kann festgestellt werden, dass das Experteninterview die Erkenntnis gewonnen hat, dass die Einführung eines standardisierten Prozessmodells die Aufwandsschätzung verbessert, somit kann H0 verworfen werden. Eine statistische Auswertung in einer quantitativen Studie ist aus wissenschaftlicher Sicht für weitere Forschungen jedenfalls anzustreben. Diese ausgedehnte Studie sollte im Sinne einer möglichst breit gefächerten Probandengruppe mehrere Unternehmen unterschiedliche Größe und mit unterschiedlicher Unternehmensstruktur umfassen.

In die Zukunft blickend wird für Aufwandsschätzungen auch das Schlagwort „Machine Learning“ interessant, womit aus historischen Daten durch die Anwendung von Algorithmen Vorhersagen für zukünftige Softwareprojekte durch verschiedene Parameter erstellt werden können. Als

Parameter könnten sich für ein solches Vorhaben die Einstufung nach dem in dieser Arbeit angeführten Fragebogen eignen.

ABKÜRZUNGSVERZEICHNIS

ASD – Adaptive Software Development
CBR – Case-based reasoning
CoBRA – Cost estimation, benchmarking, and risk assessment
COCOMO – Constructive Cost Model
DSDM – Dynamic systems development method
GUI – Graphical User Interface
ISBSG – International Software Benchmarking Standards Group
KLOC – Kilo Lines of Code
KVP – Kontinuierlicher Verbesserungsprozess
XP – Extreme Programming
PMI – Project Management Institut
PSO – Product Specification Outline (Dokument in ASD)

ABBILDUNGSVERZEICHNIS

TABELLENVERZEICHNIS

LITERATURVERZEICHNIS

Agile Business Consortium (2014). *The DSDM Agile Project Framework (2014 Onwards). Abgerufen am 14.10.2017 von https://www.agilebusiness.org/resources/dsdm-handbooks/the-dsdm-agile-project-framework-2014-onwards*

Aranda, J. (2005). *Anchoring and Adjustment in Software Estimation*.

Beck, K. & Andres, C. (2004). *Extreme Programming Explained - Embrace Change*. Addison Wesley

Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M. … Thomas, D. (2001). *Agile Manifesto*. Abgerufen am 14.10.2017 von http://agilemanifesto.org/principles.html

Becker, A. (2011). *Online-Themenspecial Testing*. OBJEKTspektrum

Cockburn, A. (2005). *Crystal Clear - A Human-Powered Methodology for Small Teams*. Addison Wesley

Cockburn, A. (2006). *Agile Software Development - The Cooperative Game*. Addison Wesley

Cohn, M. (2005). *Agile Estimating and Planning*. Prentice Hall

Davis, J. & Daniels, K. (2016). *Effective DevOps - Building a Culture of Collaboration, Affinity, and Tooling at Scale*. O'Reilly

De Luca, J. (o. J.). *The Latest FDD Processes*. Abgerufen am 14.10.2017 von http://www.nebulon.com/articles/fdd/latestfdd.html

Grimstad, S. & Jorgensen M. (2006). *Software Estimation Error*. Abgerufen am 08.11.2018 von https://www.simula.no/file/simulase16pdf/download

Halkjelsvik T. & Jorgensen M. (2018). *Time Predictions*. Schweiz: Springer

Hill, P. R. (2011). *Practical Software Project Estimation - A Toolkit for Estimating Software DevelopmentEffort & Duration*. McGraw-Hill

Jorgensen, M. (2014). *What We Do and Don't Know about Software Development Effort Estimation*. IEEE. DOI: 10.1109/MS.2014.49

Jorgensen, M. (2015). *The Effect of the Time Unit on Software Development Effort Estimates*. IEEE. DOI: *10.1109/SKIMA.2015.7399992*

Leffingwell, D. (2011). *Agile Software Requirements - Lean Requirements Practices for Teams, Programs, and the Enterprise*. Addison Wesley

McConnel, S. (2006). *Software Estimation*. Redmont, Washington: Microsoft Press

Patton, J. (2014). *User Story Mapping - Discover the whole Story, build the right Product*. O'Reilly

Puscasu, A. (2019, 30. April). *Influencing time, resources and cost estimates quality*. Abgerufen am 30.06.2019 von http://apepm.co.uk/influencing-estimates/

Reddy, A. (2016). *The Scrumban [R]Evolution - Getting the Most Out of Agile, Scrum, and Lean Kanban*. Addison Wesley

Roock, S. (2007). *Feature Driven Development*. Abgerufen am 14.10.2017 von https://www.it-agile.de/fileadmin/docs/veroeffentlichungen/Artikel_OS_05.2007_FDD.pdf

Schwaber, K. (1997). *SCRUM Development Process*. London: Springer

Trendowicz, A. & Jeffery, R. (2014). *Software Project Effort Estimation.* Kaiserslautern, Sydney: Springer

Trendowicz, A. & Münch, J. (2009). *Chapter 6 Factors Influencing Software Development Productivity-State-of-the-Art and Industrial Experiences*. Advances in Computers

Turpitka, T. (2016, 2. August). *Time estimation for software testing*. Abgerufen am 29.09.2018 von https://jaxenter.com/time-estimation-for-software-testing-128078.html

Vigenschow, U. (2015). *APM - Agiles Projektmanagement - Anspruchsvolle Softwareprojekte erfolgreich steuern*. Heidelberg. dpunkt.verlag