MASTERARBEIT

EIN PROZESSMODELL ZUR EINFÜHRUNG UND STEUERUNG EINER VERTEILTEN SOFTWAREENTWICKLUNG FÜR DIE PROJEKTABWICKLUNG AM BEISPIEL EINES ÖSTERREICHISCHEN INDIVIDUALSOFTWAREDIENSTLEISTERS

ausgeführt am

Studiengang

Informationstechnologien und Wirtschaftsinformatik

Von: Haris Bećirević

Personenkennzeichen: 2010320017

Graz, am 25. November 2021 .....

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

Ich danke meiner Familie sowie meinem Freundeskreis für die Unterstützung und Ermutigung während des Schreibens dieser Arbeit sowie der gesamten Studienzeit. Während dieser Jahre standen sie stets motivierend an meiner Seite und auch in kritischen Zeiten konnte ich mich immer auf sie verlassen. Ein weiterer Dank gilt auch meinen StudienkollegInnen, welche bei Fragen immer ein offenes Ohr für mich hatten.

Zudem möchte ich mich bei der Geschäftsführung sowie den Gesellschaftern der Guid.New GmbH mit Sitz in Graz bedanken. Ohne die von ihnen bereitgestellten Informationen sowie die von ihnen aufgewendete Zeit wäre ein erfolgreiches Abschließen dieser Arbeit wohl kaum möglich gewesen.

Ein besonderer Dank gilt ebenso Herrn Mag. Dr. Michael Amann-Langeder für die Begleitung und Unterstützung in allen Phasen dieser Arbeit. Mit seiner Erfahrung und seinem Know-how unterstützte er mich umfangreich und konnte somit einen erheblichen Beitrag zur Professionalität meiner Arbeit liefern.

KURZFASSUNG

„Verteilte Zusammenarbeit so gestalten, dass sie an das Gefühl der Zusammenarbeit an einem Ort herankommt“, sagte Dr. Vincent Tietz, Senior Consultant bei der Saxonia Systems AG, im Jahr 2016. In der heutigen Zeit, in der hohe Internetgeschwindigkeiten und hohe Bandbreiten als gang und gäbe betrachtet werden, entstehen neue Möglichkeiten der Arbeitsformen. In Zeiten der Coronakrise wurde dieser Trend noch mehr verstärkt und viele Unternehmen weltweit setzten auf den Denkansatz „Homeoffice“. Doch welche Herausforderungen birgt diese neue Form des Arbeitens und bietet sie nur Vorteile? Wie können wir speziell in der IT-Branche verteilte Softwareentwicklungsteams optimal steuern oder eine derartige Konstellation generell einführen? Sind Projektabwicklungen hinsichtlich solcher Umstände erfolgreich realisierbar und welche Vorrausetzungen sind hierfür unverzichtbar?

Das Ziel dieser Masterarbeit war das Ermitteln und Modellieren von Prozessen, die einen wesentlichen Beitrag zur Steuerung sowie Einführung einer verteilten Softwareentwicklung leisten. Das Prozessmodell umfasst dabei nicht nur die Projektabwicklung selbst, sondern auch die vorgelagerten und nachgelagerten Arbeitsabläufe. Als praktische Grundlage dient hierbei die Guid.New GmbH, ein österreichischer Individualsoftwaredienstleister mit Sitz in Graz, welcher schon vor der Pandemie auf die „Remote“-Methode setzte. Samt der wissenschaftlichen Literaturrecherche und der praktischen Validierung des entstandenen Prozessmodells durch ExpertInnen, dient das Resultat als Vorlage für Unternehmen in der Individualsoftwarebranche. Als ExpertInnen wurden die Geschäftsführung, Gesellschafter sowie MitarbeiterInnen der Guid.New GmbH herangezogen – infolgedessen enthält das Ergebnis Praxisbezogenheit, Wiederverwendbarkeit und schafft neuen Mehrwert.

ABSTRACT

"Design distributed collaboration in such a way that you get the feeling of working together in one place," Dr Vincent Tietz, Senior Consultant at Saxonia Systems AG, 2016. High-speed internet and high bandwidths are now commonplace, shaping new working environments. The coronavirus pandemic accelerated this trend; many companies relied on working from home. What challenges and advantages do new working paradigms pose? How can we, especially IT, best manage distributed software development teams or introduce such a constellation? Can project management be successfully implemented in such circumstances, and what are the essential requirements for this?

This thesis determines and models processes that assist the management and introduction of distributed software development. The process model includes both the project management itself and the upstream and downstream workflows. As a practical basis, the thesis looks at Guid.New GmbH, an Austrian individual software service provider headquartered in Graz, used the remote method before the pandemic. The literature research, in combination with the practical validation of the developed process model by experts, provides a template for companies in the individual software industry. The director, stakeholders, and Guid.New GmbH staff are consulted as experts. The result is practical, reusable, and creates new added value.

INHALTSVERZEICHNIS

1 EINLEITUNG

Verteile Softwareentwicklungsprojekte existierten bereits vor der COVID-19-Pandemie (Schirmeyer, 2020). Das Ziel ist die Entwicklung von digitalen Lösungen geografisch zu verteilen und somit innerhalb eines Projekts globales Know-how einzuholen (Oshri, Kotlarsky, & Willcocks, 2008). Durch diese Arbeitsform kann das Entwicklungsteam ortsunabhängig Implementierungen sowie Meetings durchführen – daraus ergeben sich organisatorische Herausforderungen im Projekt (Holden, 2002). Zudem werden die vor- und nachgelagerten Prozesse analysiert, sodass Aktivitäten außerhalb des verteilten Softwareprojekts ebenso mitberücksichtigt werden.

Schlussfolgernd sind die Motivation und Problemstellung dieser Arbeit das Ermitteln von Prozessen zur Einführung und Steuerung einer verteilten Softwareentwicklung. Das daraus entstandene Prozessmodell kann als Vorlage für Unternehmen in der Individualsoftwarebranche, die eine Einführung verteilter Softwareentwicklung anstreben, dienen.

1.1 Forschungsfrage

Diese Arbeit soll eine Antwort auf folgende Forschungsfrage liefern:

„Welche Prozesse ermöglichen die Einführung und Steuerung einer verteilten Softwareentwicklung für die Projektabwicklung und wie sind diese in ein Prozessmodell überführbar?“

1.2 Angewandte Methoden

Die theoretische Grundlage zu den in der Forschungsfrage verwendeten Begriffen wird mittels Literaturrecherche aufgebaut. Mithilfe dieser Grundlage werden im darauffolgenden Schritt ein theoretisches Prozessmodell entwickelt und mehrere Hypothesen aufgestellt, welche wiederum in Form von ExpertInneninterviews untersucht werden.

1.3 Aufbau der Arbeit

Die Arbeit ist in fünf Hauptkapitel unterteilt. Der Schwerpunkt der einzelnen Hauptkapitel wird im Folgenden zusammengefasst beschrieben.

Das erste Kapitel (*1 Einleitung*) dient als Einleitung und liefert den LeserInnen einen Überblick über diese Arbeit. Hier werden die Problemstellung und Motivation dieser Arbeit dargelegt sowie angewandte Methoden beschrieben. Zudem wird aus dem Arbeitsthema eine Forschungsfrage abgeleitet und definiert.

Im zweiten Kapitel (*2 Theoretischer Ansatz des Prozessmodells*) werden die theoretischen Grundlagen dieser Arbeit ausgearbeitet. Dazu werden theoretische Antworten zur Forschungsfrage geliefert und diese darauffolgend beschrieben.

Im dritten Kapitel (*3 Entwicklung des Prozessmodells*) wird anhand der theoretischen Erkenntnisse ein Prozessmodell entwickelt. Das Modell enthält die analysierten Prozesse und dient als Basis für die Hypothesenbildung im darauffolgenden Kapitel.

Im vierten Kapitel (*4 Validierung des Prozessmodells*) wird das entwickelte Prozessmodell im Rahmen von ExpertInneninterviews validiert. Anhand des Modells werden mehrere Hypothesen aufgestellt, auf Basis derer wiederum ein Fragebogen für die ExpertInneninterviews abgeleitet wird. Die in den ExpertInneninterviews ermittelten Daten werden analysiert, um schließlich eine Antwort auf die Forschungsfrage zu finden. Basierend auf den gewonnenen Erkenntnissen wird die Forschungsfrage am Ende dieses Kapitels beantwortet.

Das abschließende Kapitel (*5 Conclusio*) bietet eine Übersicht über die in dieser Arbeit gewonnenen Erkenntnisse und gibt einen Ausblick auf weitere Forschungsmöglichkeiten.

2 THEORETISCHER ANSATZ DES PROZESSMODELLS

Dieses Kapitel liefert theoretische Antworten auf die Forschungsfrage und dient als Basis für das zu entwickelnde Prozessmodell.

2.1 Verteilte Softwareentwicklung

Software ist zu einem zentralen Bestandteil von Unternehmen geworden. Der Erfolg vieler Unternehmen hängt oftmals vom effizienten Nutzen von Software ab. Viele SoftwaredienstleisterInnen haben begonnen, Software „verteilt" zu entwickeln. Durch global verteilte Softwareentwicklung können Unternehmen einerseits Kostenvorteile realisieren und andererseits Zugang zu qualifizierten Ressourcen schaffen. Darüber hinaus haben große Investitionen in die Digitalisierung sowie IT-Infrastrukturen eine Transformation von mehreren lokalen Märkten zu einem globalen Markt ermöglicht – durch diesen Prozess sind neue Formen des Wettbewerbs, aber auch der Kooperationen entstanden. (Prikladnicki, Audy, & Evaristo, A Reference Model for Global Software Development: Findings from a Case Study, 2006)

Folgende Faktoren befürworten die verteilte Softwareentwicklung (Herbsleb & Moitra, 2001):

- Stetig wachsender Druck zur Optimierung von Markteinführungszeiten durch Nutzung von Zeitzonenunterschieden sowie „Rund um die Uhr"-Entwicklungen.
- Ein Pool an global verteilten, qualifizierten SoftwareentwicklerInnen, um dahingehend Kosten innerhalb der Softwareentwicklung zu sparen.
- Die Verteilung der Softwareentwicklung minimiert das Risiko bei Naturkatastrophen und anderen Ereignissen.

Verteilte Softwareentwicklung besitzt folgende Eigenschaften: global verteilt, multikulturell sowie ortsunabhängig. Eine wachsende Anzahl an Organisationen lagert ihre Softwareentwicklungsprozesse weltweit aus, mit dem Ziel, höhere Profite zu erwirtschaften, Produktivität und Qualität zu steigern sowie Kosten zu minimieren. Diese Veränderung hat tiefgreifende Auswirkungen auf das Marketing, den Vertrieb oder auch die Art und Weise, wie Produkte und Dienstleistungen konzipiert, entworfen, entwickelt, getestet sowie an KundInnen geliefert werden. (Prikladnicki & Yamaguti, Risk Management in Global Software Development: A Position Paper, 2004)

Es gibt vier verschiedene Möglichkeiten die Softwareentwicklung zu verteilen. Die Definition erfolgt auf Basis der geografischen Verteilung sowie der Verantwortung oder Steuerungsstruktur des Projektes selbst. Die Steuerungsstruktur wird in zwei Dimensionen unterteilt: „Outsourcing" und „Insourcing". „Outsourcing" bedeutet, dass die Softwareentwicklung von externen IT-DienstleisterInnen übernommen wird. „Insourcing" bedeutet hingegen, dass das Softwareprojekt

über interne Prozesse sowie Ressourcen durchgeführt wird. In Bezug auf den geografischen Standort werden hingegen die folgenden beiden Arten unterschieden: „Onshoring“ und „Offshoring“. „Onshoring“ bedeutet, dass sich die Softwareentwicklung im selben Staat, wie auch das Projektmanagement und die Geschäftsführung, befindet. Im Fall von „Off-shoring“ hingegen, befindet sich ein Teil der SoftwareentwicklerInnen außerhalb der Staatsgrenzen. (Lehtonen, 2009)

Oft wird „Onshoring“-Entwicklung als „verteilte Softwareentwicklung“ bezeichnet und „Offshoring“ als „globale Softwareentwicklung“ – im Rahmen dieser Arbeit gehen wir jedoch immer von „Offshoring“ aus.

Im „Offshoring“-Konzept entstehen folgende Organisationen und Rollen (Nagura & Iida, 2007):

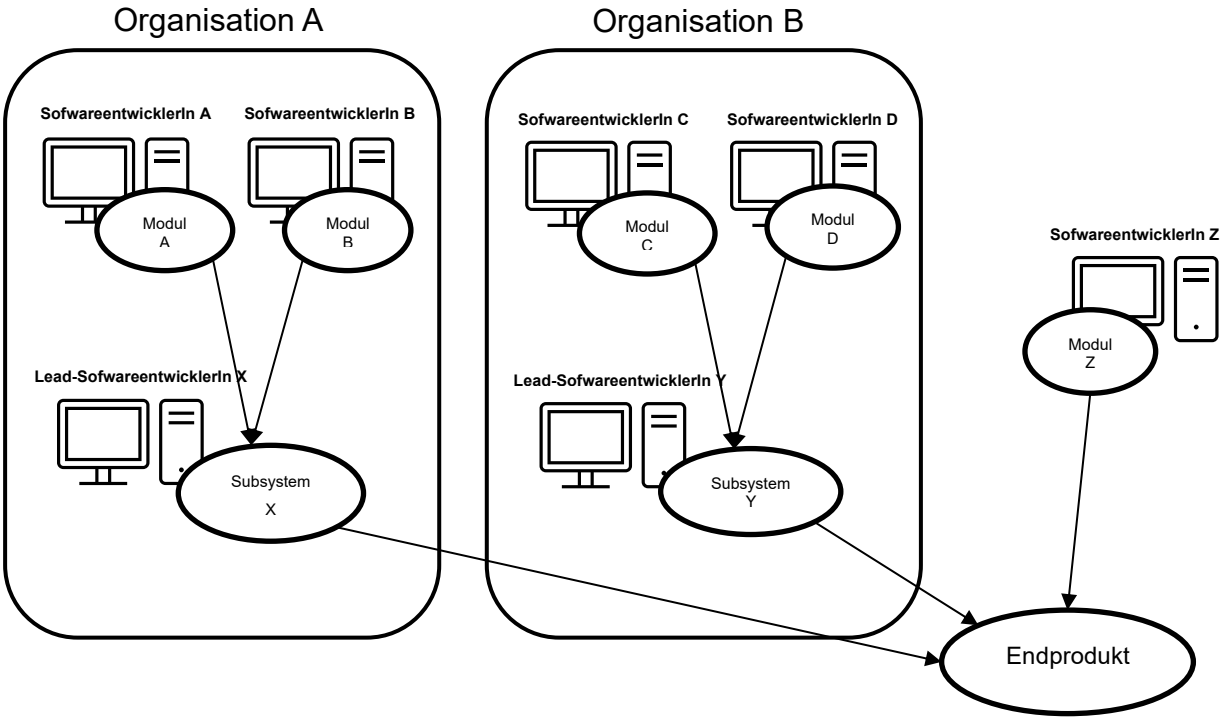

Abbildung 1 - Distributed software development (in Anlehnung an (Nagura & Iida, 2007))

Die *Abbildung 1* besteht aus mehreren Bestandteilen (Nagura & Iida, 2007):

- Organisation: Ein Unternehmen, das Softwaredienstleistungen anbietet. Ein Unternehmen beschäftigt ein oder mehrere SoftwareentwicklerInnen.
- SoftwareentwicklerInnen: Entwickeln Softwarekomponenten, die letztendlich in ein Endprodukt integriert werden. Die geografische Lage der SoftwareentwicklerInnen variiert auch innerhalb einer Organisation. Ebenso können SoftwareentwicklerInnen außerhalb von Organisationen am Gesamtprojekt teilnehmen und ihre Implementierungen in das Endprodukt integrieren.
- Lead-SoftwareentwicklerInnen: Innerhalb von Organisationen entsteht Managementaufwand, da mehrere SoftwareentwicklerInnen ihre Entwicklungen in das jeweilige Subsystem integrieren. Durch die Softwareintegration besteht Koordinierungsbedarf in Richtung Subsystem, als auch Endprodukt. Die

Lead-SoftwareentwicklerInnen tragen nicht nur die Verantwortung für die Koordination des Softwareprozesses, sondern auch für das Subsystem.

- Modul: Eine einzelne Softwarekomponente, für die die/der jeweilige/r Softwareentwickler/In verantwortlich ist. Diese Softwarekomponenten werden in ein übergeordnetes System integriert – Subsystem oder Endprodukt.
- Subsystem: Je Organisation gibt es ein Subsystem, welches iterativ in das Endprodukt integriert wird. Intern besteht das Subsystem aus mehreren Modulen und die Verantwortlichkeit obliegt der/m Lead-Softwareentwickler/In.
- Endprodukt: Die Gesamtsoftware, das an die EndkundInnen geliefert wird und somit alle Module der beteiligten SotftwareentwicklerInnen beinhaltet.

Diese Rollen werden in einer verteilten Softwareentwicklung benötigt und bei der Entwicklung des Prozessmodells berücksichtigt.

2.1.1 Herausforderungen der verteilten Softwareentwicklung

Die verteilte Softwareentwicklung stellt auch Herausforderungen dar. Organisatorische Komplexität, Planung, Aufgabenzuweisung, Kostenschätzung, Projekt- sowie Prozessmanagement werden im Zuge verteilter Umgebungen noch problematischer. Dieser Effekt wird durch flexible Anforderungen, kulturelle Vielfalt und mangelnde Kommunikation verstärkt (Casey & Richardson, 2006). Daraus folgend müssen die ProjektleiterInnen den gesamten Entwicklungsprozess kontrollieren, ihn während der Umsetzung verbessern und alle Faktoren minimieren, die die Produktivität innerhalb des Projektes verringern könnten. Hinzu kommt, dass die ProjektleiterInnen Rücksicht auf verschiedene Kulturen nehmen, miteinander verbundene Aufgaben identifizieren und die Abhängigkeiten zwischen verteilten Gruppen minimieren müssen (Jimenez, Piattini, & Vizcaino, 2009).

Verteilte Softwareentwicklung erfordert eine enge Zusammenarbeit von Personen mit unterschiedlichem kulturellen Hintergrund. Kultur unterscheidet sich in vielen kritischen Dimensionen wie beruflicher, ethischer, organisatorischer, technischer sowie Teamkultur. Kulturelle Unterschiede verschärfen häufig Kommunikationsprobleme. (Marquardt & Horvath, 2001)

Eine weitere Herausforderung stellt das Wissensmanagement dar. Die Erfahrungen, Methoden, Entscheidungen und Fähigkeiten der Teammitglieder müssen während des Entwicklungsprozesses durch einen effektiven Informationsaustausch gesammelt werden, damit Teammitglieder die Erfahrungen ihrer VorgängerInnen nutzen können. Dies führt folglich zur Reduktion von Kosten sowie überflüssiger Arbeit. Ohne effektive Mechanismen zum Informations- und Wissensaustausch können ManagerInnen die Vorteile der verteilten Softwareentwicklung nicht nutzen. Der Wissensaustausch wird erleichtert, indem ein Produkt- oder Prozess-Repository verwaltet wird – dieses beinhaltet verschiedene Quellen, wie z. B. E-Mail, Online-Diskussionen etc. (Jimenez, Piattini, & Vizcaino, 2009)

Unzureichende Kommunikation erzeugt eine weitere Schwierigkeit, da Softwareentwicklung viel Kommunikation über definierte Kommunikationskanäle erfordert. Die komplexe Infrastruktur für die verteilte Softwareentwicklung und die hohe Anzahl an Personal, die sich im Laufe der Zeit ändert, führen zu einer Verringerung der Kommunikationsfrequenz und -qualität, was sich direkt auf die Produktivität auswirkt. Zeitzonenunterschiede des verteilten Projektteams kommen zusätzlich zum Kommunikationsproblem hinzu. (Marquardt & Horvath, 2001)

Bei standortübergreifend arbeitenden Projektteams kann fehlende Synchronisierung besonders kritisch sein. Es müssen allgemein definierte Meilensteine und klare Vor- sowie Nachbedingungen für alle Aufgaben sichergestellt werden. Da Netzwerke, die sich über global verteilte Standorte erstrecken, häufig langsam und unzuverlässig sind, müssen Aufgaben, wie das Konfigurationsmanagement für Datenübertragung, eingeplant werden. (Casey & Richardson, 2006)

Abschließend sind in der verteilten Softwareentwicklung ebenso strategische Analysen ein Thema. Der Entschluss, ob ein Softwareprojekt verteilt oder von gemeinsam lokalisierten Teams entwickelt werden soll, ist eine strategische Entscheidung. In dieser strategischen Phase müssen einige Risiko- und Nutzenanalysen durchgeführt werden. Darüber hinaus werden mögliche Projektkonstellationen durch die an verschiedenen Standorten verfügbaren Ressourcen, wie beispielsweise das bestehende Fachwissen, die vorherrschende Infrastruktur etc., eingeschränkt. Mehrere Lösungsmodelle sind unter verschiedenen Umständen möglich und angemessen. (Herbsleb & Moitra, 2001)

2.2 Prozessmodell

Dieses Kapitel dient zur Veranschaulichung der allgemeinen Prozessgestaltung innerhalb von Unternehmen sowie in Hinblick auf Softwareentwicklungsprojekte. Zusätzlich wird der Unterschied zwischen den Unternehmensprozessen und Softwareentwicklungsprozessen theoretisch erläutert.

2.2.1 Allgemein

Prozessmodelle sind vereinfachte Abbildungen von Prozessen innerhalb von Organisationen. Das Modell repräsentiert die chronologische und logische Sequenz von Aktivitäten oder Funktionen. Je nach Anwendungsbedarf können Prozesse in unterschiedlichem Detaillierungsgrad oder Umfang modelliert werden. Aufgrund der Komplexität und Vielfalt von Abläufen können nicht alle Prozesse einer Organisation in einem einzigen Modell dokumentiert werden. Deshalb werden Prozessmodelle über verschiedene hierarchische sowie beschreibende Ebenen definiert. Zuerst beginnt man mit der Übersicht aller Hauptaufgaben, welche dann Schritt für Schritt verfeinert werden und somit ein detailliertes Modell mit Teilaufgaben entsteht. Die Anzahl der Beschreibungsebenen variiert je nach benötigtem Detaillierungsgrad des Modells. Basierend auf den jeweiligen Anforderungen können je Beschreibungsgrad unterschiedliche Prozessmodelltypen verwendet werden. Daraus resultierend unterstützen Modelle bestehende

Prozesse verständlich zu dokumentieren und werden für Projekte sowie im Arbeitsalltag einer Organisation genutzt. (BMI, 2018)

2.2.2 Geschäftsprozesse in Unternehmen

Ein Geschäftsprozess ist eine Reihe von sequenziell oder parallel laufenden Aktivitäten zur Durchführung betrieblicher Aufgaben. Das Betriebsergebnis wird durch die Bereitstellung von Informationen und/oder materielle Transformation erreicht (Allweyer, Geschäftsprozessmanagement, 2005). Geschäftsprozesse bestehen aus Aktivitäten, Verantwortlichkeiten, beteiligten Personen sowie Organisationseinheiten. Durch diese Bestandteile ermöglichen es Geschäftsprozesse den Unternehmen, ihre Aktivitäten auf Erfüllung von KundInnenanforderungen sowie Erreichung der Unternehmensziele auszurichten (Schmelzer & Sesselmann, 2008). Bei entsprechender Pflege der Geschäftsprozesse innerhalb von Organisationen arbeitet ein Unternehmen kundenorientiert und beinhaltet weniger interne Schnittstellen – daraufhin reduziert sich der Koordinierungsaufwand. Wie bereits vorhin erwähnt, besteht ein Geschäftsprozess aus mehreren Komponenten: Input, KundInnenanforderungen, Wertschöpfung, Resultaten, Prozessverantwortlichen sowie Mess- und Zielgrößen um Geschäftsprozesse entsprechend steuern zu können (Schmelzer & Sesselmann, 2008). Geschäftsprozesse werden daher in zwei Kategorien unterteilt:

- Primäre Geschäftsprozesse: Hier findet die Wertschöpfung statt und der Mehrwert für KundInnen wird erzeugt.
- Sekundäre Geschäftsprozesse: Unterstützen die primären Geschäftsprozesse sowie Wertschöpfung mit Leistung.

Primäre Geschäftsprozesse werden auch als Kernprozesse charakterisiert, die sekundären wiederum als Support- oder Unterstützungsprozesse. (Schmelzer & Sesselmann, 2008)

Zusätzlich zu den Kern- und Supportprozessen existieren die Managementprozesse. Der Managementprozess beschäftigt sich mit der Steuerung der Kernprozesse mit Fokus auf die Strukturierung der Verantwortlichkeiten, Rollen sowie deren Aufgaben. Die Entwicklung von Strategie, Definition der Ziele, Projektmanagement, Finanzmanagement werden als Managementprozesse kategorisiert. Als Kernprozesse gelten zum Beispiel die Produktion oder der Vertrieb. Das Buchhaltungswesen oder Personalmanagement zählen zu den Supportprozessen. (Fischermanns, 2013)

Das Zusammenspiel zwischen den drei Prozesskategorien wird in folgender Abbildung grob dargestellt:

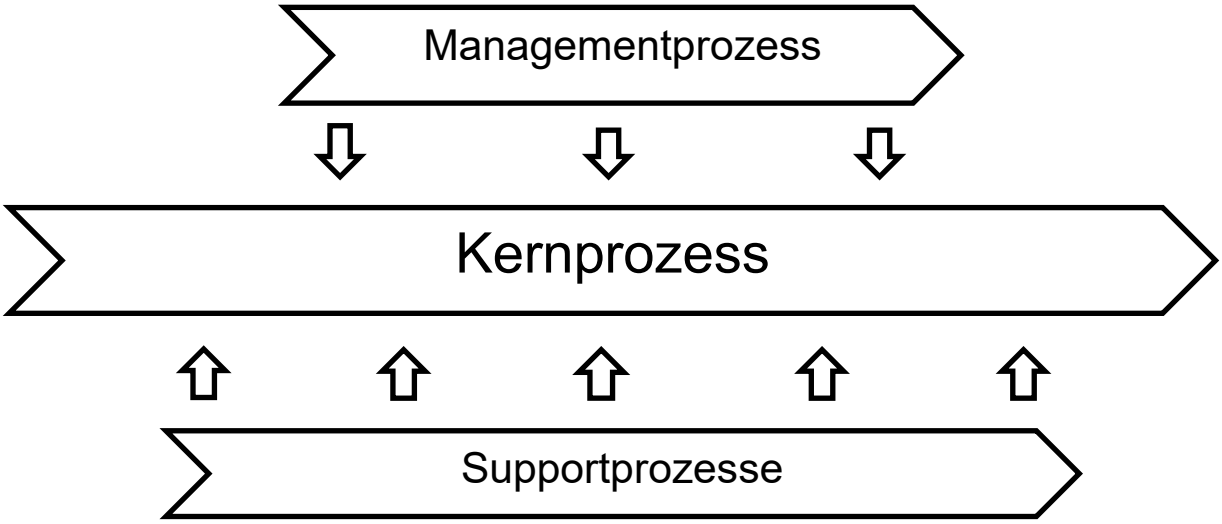

Abbildung 2 - Prozessarten (in Anlehnung an (Fischermanns, 2013))

Die nächste Abbildung fokussiert sich auf die Kern- sowie Supportprozessen:

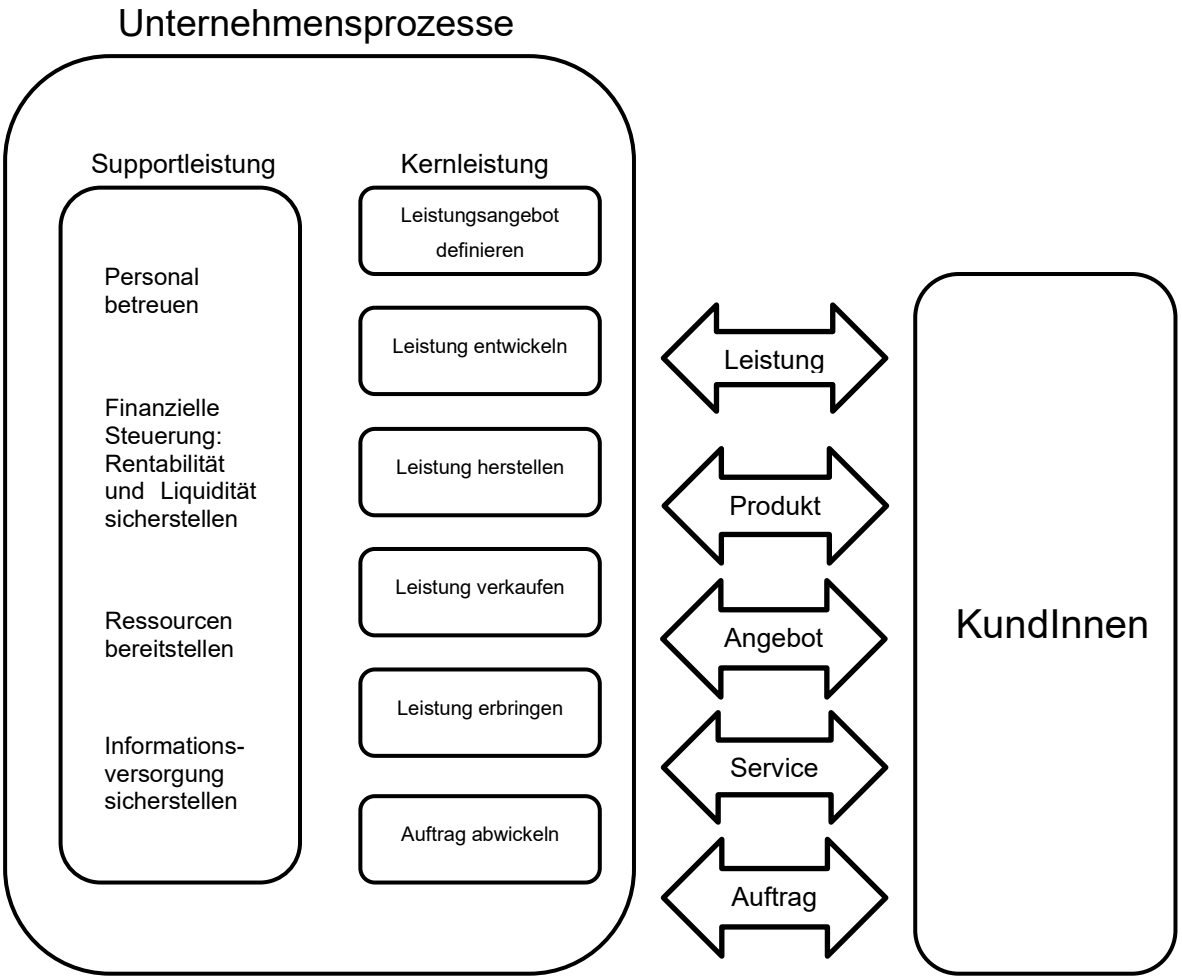

Abbildung 3 - Kern- und Supportprozesse (in Anlehnung an (Pongratz, Tramm, & Wilbers, 2009))

Das Verständnis der internen Leistungserstellung ist eine Voraussetzung für den betrieblichen Erfolg des Unternehmens. Die Geschäftsprozessmodellierung abstrahiert die Geschäftsprozesse auf einer hohen Ebene und visualisiert sie in einem Modell. Die Modellierung dient als Werkzeug des Prozessmanagements und wird somit für die Konzeption sowie Wartung der internen Geschäftsabläufe genutzt. Das Ziel der Geschäftsprozessmodellierung ist die Dokumentation der operativen Prozesse und auf diese Weise Verständnis dafür zu schaffen, wie die internen Geschäftsabläufe funktionieren oder miteinander harmonieren. Hierfür wird eine Ist-Analyse der aktuellen betrieblichen Aktivitäten durchgeführt. Durch die Analyse können Optimierungspotenziale entdeckt sowie Schwachstellen identifiziert und beseitigt werden.

Es gibt verschiedene Spezifikationen der Geschäftsprozessmodellierung – eine davon ist die „Business Process Model and Notation“, abgekürzt „BPMN“. Der Fokus in BPMN liegt in der Visualisierung von Geschäftsprozessen, wobei BMPN 2.0 zusätzlich auf die Ausführung abzielt. Die erstellten Diagramme des „BPMN“ werden als „Business Process Diagram“, abgekürzt „BPD“, bezeichnet. (Allweyer, BPMN 2.0 Business Process Model and Notation. Einführung in den Standard für die Geschäftsprozessmodellierung, 2015)

Die Notation definiert folgende grafische Elemente (Allweyer, BPMN 2.0 Business Process Model and Notation. Einführung in den Standard für die Geschäftsprozessmodellierung, 2015):

- Flow Objects: beschreiben die Knoten, wie zum Beispiel „Activity“, „Event“ oder „Gateway“
- Connecting Objects: Verbindungen zwischen den Objekten
- Pools und Swimlanes: Akteure sowie Systeme werden in Bereiche dargestellt
- Artifacts: Elemente zur weiteren Dokumentation, wie zum Beispiel „Annotations“ sowie „Data Objects“

Basierend auf diesen grafischen Elementen wird im Rahmen dieser Arbeit ein Prozessmodell visualisiert.

2.2.3 Unterschied zum Vorgehensmodell

Die Autoren Schatten, Biffl, Demolsky, Gostischa-Franta, Östreicher und Winkler erläutern Vorgehensmodelle wie folgt:

„Allgemein definieren Vorgehensmodelle eine konkrete Abfolge von Schritten im Entwicklungsprozess, die dem Entwickler helfen, sich im Projekt zu orientieren, und regeln, welche Produkte in welchem Fertigstellungsgrad und mit welcher Qualität vorliegen sollen. Sie helfen also, im Rahmen des Projektmanagements das Software-Projekt zu planen und zu begleiten. Software-Prozesse oder Vorgehensmodelle legen eine Projektstrategie fest, die sowohl die zeitliche Reihenfolge der zu erstellenden Produkte als auch das notwendige Qualitätsniveau zu einem bestimmten Zeitpunkt (etwa bei einem Meilenstein) definiert. Sie regeln also, wann welche Produkte von wem in welchem Fertigstellungsgrad und auf welchem Qualitätsniveau vorliegen müssen. Vorgehensmodelle bilden also den organisatorischen Rahmen eines Software-Projekts. In der industriellen Praxis existieren zahlreiche Vorgehensmodelle, die auf unterschiedliche Projektbedürfnisse und Projektcharakteristika abgestimmt

sind. Wesentliche Unterscheidungskriterien sind beispielsweise Projektdauer und Projektgröße, Anwendungsdomäne und Komplexität.“ (Schatten, et al., 2010)

Unterschiedliche Vorgehensmodelle wurden bereits für die Softwareentwicklung entwickelt und sind schon im praktischen Einsatz. Folgende Modelle sind hiermit gemeint (Kuhrmann & Linssen, 2013):

- Scrum
- V-Modell
- Rational Unified Process (RUP)
- Wasserfallmodell
- Xtreme Programming (XP)
- Kanban

Die vor- und nachgelagerten Geschäftsprozesse zum Vorgehensmodell erweitern das Spektrum und liefern ein gesamtheitliches Prozessmodell eines Unternehmens.

2.3 Projekte in der Softwareentwicklung

In diesem Kapitel wird auf die Besonderheiten von Projekten in der Softwarebranche eingegangen. Zusätzlich wird der Begriff „Individualsoftware“ erläutert und es werden mehrere Abrechnungsmodelle beschrieben.

2.3.1 Allgemein

Sowohl für Projekte als auch Prozesse werden Ziele definiert. Für die Zielerreichung wird in beiden Situationen Input in Output transformiert und eine Zusammenarbeit verschiedener Organisationen gefordert. (Gareis & Stummer, 2007)

Ein Projekt ist ein zielgerichtetes und einmaliges Vorhaben, welches aus mehreren koordinierten und gesteuerten Aktivitäten besteht. Bei der Durchführung von Projekten müssen im Zuge der Zielerreichung folgende Aspekte berücksichtigt werden: Zeit, Ressourcen und Qualität. Unter Ressourcen versteht man die Finanzierungskosten, Personal sowie Infrastruktur. Zur Erfüllung von Projektzielen werden Projektteams gebildet, welche durch Projektmanagement gesteuert werden müssen. Die Effizienz des Projektmanagements wird durch Universitätslehrgänge, Beratungsfirmen sowie Softwareprogramme optimiert. (Möller & Dörrenberg, 2010)

Mehrere Methoden des Projektmanagements empfehlen, die Ziele sowie Aufgaben nach dem „SMART“-Schema zu formulieren (Drucker, 1977):

- Spezifisch: Ziele müssen so konkret und spezifisch wie möglich formuliert werden.
- Messbar: Qualitative und quantitative Messgrößen bestimmen.

- Akzeptiert: Das Ziel muss von allen Beteiligten akzeptiert und angenommen werden. Die Zielerreichung wird somit attraktiv.
- Realistisch: Machbarkeit der Aufgabe innerhalb der Zeit und mit den Mitteln.
- Terminiert: Die Ziele müssen zeitlich bindend eingeplant werden.

Das Vorgehen wird durch die Projektdefinition konkretisiert – daraufhin werden die Prozesse und Aktivitäten strukturiert. (Möller & Dörrenberg, 2010)

Professionelle Softwareentwicklungen werden als Projekte organisiert, welche sich untereinander stark unterscheiden können. Diese Projekte weisen unterschiedliche Merkmale auf. Basierend auf diesen Charakteristika werden je Projekt, der am besten geeigneten Ansatz sowie die erforderliche Methodik oder Technologie bestimmt. (Züllighoven, 2004)

Folgende Eigenheiten unterscheidet ein Softwareentwicklungsprojekte von anderen Projekten (Züllighoven, 2004):

- Projekt: Das Projekt selbst kann eine neue Applikation, eine verbesserte Variante oder eine Umstrukturierung einer bestehenden Software sein.
- Projektziel: Eine Software muss aus wiederverwendbaren und zusammenspielenden Komponenten entwickelt sein. Auf diese Weise wird die Erweiterbarkeit für Folgeprojekte gewährleistet.
- Fachdomäne: Software wird von ein oder mehreren Fachbereichen, auch Fachdomänen genannt, genutzt. Diese Domänen besitzen einen hohen Anteil an Komplexität, welches durch die entwickelte Software abgedeckt sein muss. Hinzukommend soll die Software über einen längeren Zeitraum (mehrere Jahre) von verschiedenen BenutzerInnen genutzt werden – diese BenutzerInnen unterschieden sich zusätzlich durch verschiedene Profile und Qualifikationen.
- Organisatorischer Aspekt: Das Entwicklungsteam ist innerhalb der Unternehmensorganisation integriert oder arbeitet temporär auf Basis eines Dienstleistungsvertrags zwischen AuftragnehmerInnen und KundInnen. Ein kleines Softwareentwicklungsteam besteht aus mindestens drei bis fünf Mitgliedern. Die Projektgröße kann auf bis zu sieben parallel arbeitende Teams mit je 16 Mitgliedern skaliert werden.
- Technischer Aspekt: Software wird im sozialen sowie auch technologischen Kontext in eine Umgebung integriert. Die technologische Integration ist eine Kombination aus Soft- und Hardwareintegration – das Ergebnis ist eine Landschaft an heterogenen Softwareprodukten.

Aufgrund dieser Unterschiede werden Softwareprojekte spezifisch behandelt sowie abgewickelt.

2.3.2 Individualsoftware

Als Individualsoftware wird eine individuell geschaffene Software bezeichnet. Im Zuge der Entwicklung wird ein neues Softwareprodukt gefertigt, welches auf den individuellen Anforderungen basiert. Unternehmen lagern die Entwicklung der benötigten Software in den meisten Fällen aus, um sich in weiterer Folge auf ihr Kerngeschäft zu konzentrieren. Dabei übernehmen Individualsoftware-DienstleisterInnen die Beauftragung und konstruieren in Zusammenarbeit mit ihren KundInnen die benötigte Softwarelösung. Unter anderem existiert das Prinzip „Buy, Customize and Integrate“. Darunter wird die Anschaffung einer Standardsoftware, die in weiteren Schritten verändert oder um neue individuelle Module erweitert wird, verstanden. (Schiele, 2009)

Individualsoftware benötigt initial ein höheres finanzielles Budget als Standardsoftware, da ersteres für das Zielunternehmen maßgeschneidert entwickelt wird. Dadurch werden bestehende Aufbaustrukturen sowie Geschäftsprozesse als Sollkonzepte übernommen und schon funktionierende Prozesse müssen nicht unmittelbar verändert werden. Aufgrund dieser hohen Anpassung der Individualsoftware kann eine fließende Integration beim Ersetzen alter Software gewährleistet werden – vorausgesetzt es entstehen keine unvorhergesehenen Probleme. Beim Einsatz eines eigenen Entwicklungsteams, wird die Abhängigkeit eines Unternehmens in Bezug auf Standardsoftware-LieferantInnen erheblich gesenkt und das Unternehmen ist bezüglich Erweiterungen kosteneffektiver. Ein Nachteil beim Beschäftigen eines eigenen Softwareentwicklungsteams ist die konstant benötigte Arbeitsauslastung, welche sich jederzeit aufgrund schwankender Anzahl an Softwareanforderungen ändern kann. Ein weiterer Nachteil ist, dass Individuallösungen nicht detailliert dokumentiert werden im Vergleich zu mehrfach verwendeten Standardlösungen. (Schiele, 2009)

Individuelle Softwarelösungen sind initial teurer als standardisierte Lösungen. Dennoch gibt es bedeutsame kostensparende Faktoren (Vaher, 2003):

- Standardsoftware ist oft überdimensioniert und daher für kleine Betriebe nicht geeignet oder überteuert. Die Supportkosten sind somit bei Individualsoftware dementsprechend realistisch angepasst.
- Individuelle Softwarelösungen können effizienter und kostengünstiger erweitert werden. Angepasste Fehlermeldungen für die BenutzerInnenzielgruppe verringern das Maß an Unverständlichkeit der Software und der interne Unternehmensbetrieb wird somit am Laufen gehalten.
- Bei professioneller und vertrauensbasierter Zusammenarbeit zwischen AuftragnehmerInnen und AuftraggeberInnen können bei der Verrechnung nach Aufwand ebenso Kosten gespart werden.
- Die Anwendbarkeit der Individuallösung muss nur in fix vorgegebenen Systemumgebungen getestet werden und auch die Qualität der Software muss nur in den definierten Systemumgebungen gegeben sein. Somit werden auch hier Kosten gespart.

Basierend auf diesen Erkenntnissen sind Individualsoftwarelösungen initial teurer, jedoch langfristig betrachtet, durch ihre maßgeschneiderte Eigenschaft, kostengünstiger.

2.3.3 Vertrags- und Abrechnungsmodelle

Softwareprojekte besitzen spezifische Eigenschaften, die eigene Vertrags- und Abrechnungsmodelle benötigen. Insbesondere bei längeren oder komplexeren Projekten wird es um ein Vielfaches schwieriger, faire Kostenabschätzungen anzubieten. In den meisten Fällen wird somit ein ohnehin schon komplexes Softwareprojekt noch komplizierter aufgeplant oder konzipiert. (Opelt, Gloger, Pfarl, & Mittermayr, 2012)

Für Softwareprojekte existieren daher folgende Vertrags- und Abrechnungsmodelle (Opelt, Gloger, Pfarl, & Mittermayr, 2012):

- Festpreis
- T&M – Time & Materials
- T&M mit Abbruch
- „T&M on Steroids“
- Agil mit Festpreis
- „Money For Nothing, Changes For Free”

Ein Festpreisvertrag zwischen DienstleisterInnen und KundInnen definiert den Umfang, Features, Projektzeitplan sowie Preis eines Softwareprojekts. Grundsätzlich werden all diese Eigenschaften fixiert – also nicht nur der Preis. KundInnen präferieren Festpreisprojekte aus folgenden Gründen (Van Cauwenberghe, 2003):

- KundInnen möchten das Preis-Leistungs-Verhältnis zwischen mehreren AnbieterInnen vergleichen können.
- KundInnen tragen ein geringes Risiko aus rechtlicher Perspektive: Wenn SoftwaredienstleisterInnen die Softwarefunktionalitäten nicht rechtzeitig oder entsprechend der definierten Anforderungen liefern, können KundInnen Entschädigungen fordern.
- KundInnen möchten ein geringes finanzielles Risiko eingehen: Aufgrund des im Vorhinein angebotenen Fixpreises fühlen sich KundInnen sicher. Erstaunlicherweise werden festpreisbasierte Softwareprojekte im Laufe des Projektes teurer, da KundInnen ihre Anforderungen ändern.
- Der vorausgeplante Projektzeitplan gibt den KundInnen ein besseres Gefühl bezüglich Projektsteuerung. Das Resultat solcher Projekte sind oft Schwierigkeiten im Endspurt des Projekts.

Festpreisbasierte Softwareprojekte werden grundsätzlich von SoftwaredienstleisterInnen gemieden, da diese hohe finanzielle und funktionale Risiken bergen. Diese Vertragsart schützt die KundInnen auf Kosten der DienstleisterInnen. (Van Cauwenberghe, 2003)

Das zweite Vertrags- und Abrechnungsmodell ist „Time & Materials“. Es setzt voraus, dass KundInnen den tatsächlichen Arbeitsumfang auf der Grundlage der Stundensätze abrechnen. KundInnen werden die aufgewendeten Arbeitsstunden in Bezug auf das Softwareprojekt verrechnet. Der Hauptvorteil des „T&M“-Modells ist die Flexibilität und die Möglichkeit, Anforderungen anzupassen, Entwicklungsrichtungen zu ändern, Funktionen zu ersetzen und BenutzerInnen mehr in den Entwicklungsprozess miteinzubeziehen. Auf diese Weise erhalten die KundInnen ein mehrwertschaffendes Produkt. (Korotia, 2017)

Die Vorteile von Time & Material in Softwareprojekten sind (Korotia, 2017):

- Flexibilität: Unternehmen können das Arbeitsvolumen ändern, Designs und Konzepte überarbeiten, den Fokus verschieben oder Funktionalitäten nach der Projektimplementierung ändern.
- Dynamischer Arbeitsumfang: Anfangs wird ein generelles Ziel definiert. Der Weg zum Ziel wird im Laufe des Softwareprojektes bestimmt – ist also nicht im Vorhinein klar definiert. Für Startups und kleinere Unternehmen ist diese Arbeitsweise oft die bessere Variante, da Entscheidungen, Strategieentwicklung und Entwicklung von Individualsoftware parallel zueinander ablaufen – man spricht hier von agilen Methoden.
- Optimiertes Timing: Das Vermeiden von Festpreisvergleichen hilft den KundInnen kostbare Zeit zu sparen und unmittelbar mit der Entwicklung zu starten. Zusätzlich wird je Projektmitglied eine genauere Zeiterfassung garantiert und somit je Softwarefeature eine genauer Kostenbericht verfasst.

Die beiden Abrechnungsmodelle beinhalten unterschiedliche Vor- und Nachteile – der Auswahlgrund kann schlussfolgernd für jedes Softwareprojekt variieren.

2.4 Prozesse in der Projektabwicklung

In diesem Kapitel werden Prozesse und Phasen der Projektabwicklung beschrieben, welche ebenso im Zuge der verteilten Softwareentwicklung durchgeführt werden.

2.4.1 Projektmanagement

Die ISO 21500 „Leitfaden zum Projektmanagement“ ist ein internationaler Standard, der 2007 von der International Organization for Standardization (ISO) entwickelt und 2012 veröffentlicht wurde. Es soll allgemeine Anleitungen geben und Kernprinzipien sowie bewährte Verfahren im Projektmanagement erläutern (International Standards Organization (ISO), 2013). ISO plante, dass der „21500“-Standard der erste in einer Familie von Projektmanagement-Standards wird. Aus diesem Grund konzipierte ISO diese Norm auch so, dass sie sich an anderen verwandten Normen, wie ISO 10005 „Qualitätsmanagementsysteme − Richtlinien für Qualitätspläne“, ISO 10006 „Qualitätsmanagementsysteme − Richtlinien für das Qualitätsmanagement in Projekten“, ISO 10007 „Qualitätsmanagementsysteme − Richtlinien für das Konfigurationsmanagement“,

ISO 31000 „Risikomanagement – Prinzipien und Richtlinien“ ausrichten kann (International Standard Organization (ISO), 2012).

ISO 21500 sieht 39 Projektmanagementprozesse vor, die in fünf Prozessgruppen zusammengefasst sind. Es handelt sich um eine Sammlung von Projektmanagementprozessen, die auf jede Organisation angewendet werden können, aber gleichermaßen auf jede Art von Projekt anwendbar sind. ProjektleiterInnen und ausgewählte StakeholderInnen entscheiden gemeinsam, welche Prozesse für ihre jeweiligen Projekte sinnvoll sind und in welchem Umfang sie angewendet werden sollen. Dabei muss immer das Kosten-Nutzen-Verhältnis des Projektmanagements im Auge behalten werden. Die in ISO 21500 aufgeführten Prozessmodelle können aus zwei Perspektiven betrachtet werden (Wagner & Grau, 2014):

- aus Sicht der Prozessgruppen (für das Projektmanagement, z. B. Initiierung oder Controlling) und
- aus der Sicht von Themengruppen (zur Zuordnung von Prozessen zu Themen, z. B. StakeholderInnen oder Ressourcen).

Es existieren fünf Prozessgruppen (Wagner & Grau, 2014):

- Initiierung
- Planung
- Umsetzung
- Controlling
- Abschluss

Das Projekt beginnt mit dem Initiierungsprozess und endet mit dem Projektabschluss. Alle Prozessgruppen interagieren miteinander, sind voneinander abhängig und sind selten jeweils unabhängig. Für jede Phase des Projekts kann der Prozess gleichermaßen wiederholt werden. Je nach Komplexität des Projekts bzw. Projektfortschritt können die Prozesse in der Prozessgruppe parallel ablaufen, sich überschneiden oder, falls nicht erforderlich, gar nicht genutzt werden. (Wagner & Grau, 2014)

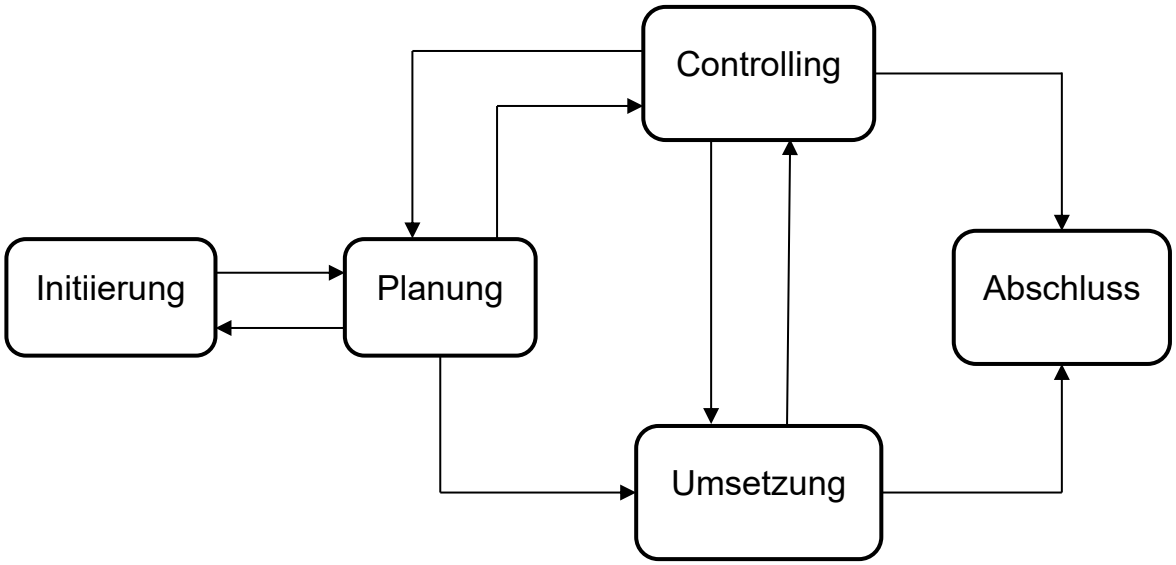

Abbildung 4 - Prozessgruppen des Projektmanagements (in Anlehnung an (DIN e.V., 2013))

Initiierung

Ein Projekt muss durch einen Projektauftrag formal genehmigt werden. Dieser Vorgang kann auf Basis eines vorab formulierten Business Cases oder eines bestehenden Realisierungsvertrages übernommen werden. Die Inhalte dieser beiden Dokumente sind jedenfalls in der Beschreibung des Projektauftrags zu berücksichtigen (Wagner & Grau, 2014). Zusätzlich ist es ratsam, neben diesen zwei Dokumenten ebenso eine Leistungsbeschreibung zu dokumentieren. Weitere Bestandteile des Initiierungsprozesses sind die Auswahl von ProjektleiterInnen sowie die klare Beschreibung dessen, wie das Projekt mit der Unternehmensstrategie verknüpft ist. (Wagner & Grau, 2014)

Nach der Initiierung erfolgt die Zusammensetzung des Projektteams. Diese Zusammensetzung kann sich abhängig von der Projektumgebung sowie dem Projektverlauf ändern. Innerhalb der Projektzusammensetzung ist es zudem essenziell zu definieren, wie Projektteammitglieder rekrutiert werden und wie das Projektteam bei Bedarfsnachlass reduziert werden kann. Mit der Beteiligung der Projektleitung werden die ProjektmitarbeiterInnen ausgewählt. In der Auswahl müssen individuelle Fähigkeiten sowie Know-how mitberücksichtigt werden. Zudem müssen die Rollen, Verantwortlichkeiten und Kommunikationswege klar definiert werden. Das Resultat der Projektzusammensetzung sind ausgeschriebene Stellenanzeigen sowie entsprechende Dienstverträge. (Wagner & Grau, 2014)

Planung

In der Planung werden Detailpläne erarbeitet, welche als Grundlage für die Projektumsetzung und das Projektcontrolling dienen. (Wagner & Grau, 2014)

Im ersten Schritt muss der Leistungsumfang des Projektes beschrieben und abgegrenzt werden. Der Leistungsumfang definiert sich über Projektziele, Lieferobjekte, Relation zur Unternehmensstrategie etc. und dient als Grundlage für Entscheidungen im Zuge des Projekts. Wenn die Projektziele und Lieferobjekte im Detail vorliegen, werden in einem darauffolgenden Planungsvorgang die Arbeitspakete erzeugt. Diese Arbeitspakete werden in einen Projektstrukturplan integriert, wodurch das Gesamtprojekt in plan- und kontrollierbare Bestandteile strukturiert wird. (Wagner & Grau, 2014)

Die Durchführung der Arbeitspakete muss entsprechend zeitlich geplant werden. Hierfür stehen drei Planungsprozesse zur Verfügung (Wagner & Grau, 2014):

- Definition der logischen Abfolge zur Durchführung von Arbeitspaketen und Aktivitäten sowie Erkennung der Abhängigkeiten zwischen den Paketen
- Abschätzung der Dauer von Arbeitspaketen und Aktivitäten
- Erzeugung eines Terminplans

Auf Grundlage der Leistungs- und Terminplanung kann die Ressourcenplanung erfolgen. Beim Schätzen des Umfangs der notwendigen Ressourcen wird für jedes Arbeitspaket der jeweilige Bedarf an Ressourcen (Personen, Infrastruktur, etc.) analysiert. Zu diesem Zweck ist es hilfreich, Schätzungen aus bereits spezifizierten Arbeitspaketen heranzuziehen und diese für die Ressourcenplanung zu nutzen. Im nächsten Schritt wird die Projektorganisation definiert und dahingehend die nötigen Bestätigungen für die Ressourcenfreigaben eingebracht. (Wagner & Grau, 2014)

Die Resultate aus dem Planungsprozess werden für die Erstellung des Projektmanagementplans sowie Projektplans verwendet. Der Projektmanagementplan beschreibt, wie die Umsetzung und das Controlling eines Projekts realisiert werden. Der Projektplan beinhaltet die Basispläne (Inhalte, Termine, Schätzungen etc.) für die Umsetzung. Diese Basispläne werden im Zuge der Planungsprozesse erstellt. (Wagner & Grau, 2014)

Umsetzung

Auf Basis der Planung werden Arbeitspakete und Aktivitäten entsprechend umgesetzt. Konkret bedeutet dies, dass ProjekleiterInnen in Kooperation mit KundInnen sowie Projektmitgliedern (Wagner & Grau, 2014):

- alle Tätigkeiten untereinander abstimmen,
- alle relevanten ProjektmitarbeiterInnen organisieren und
- die Umsetzung aller Arbeitspakete kontrollieren

um in weiterer Folge die definierten Projektziele zu erreichen. Die Beibehaltung der Motivation sowie Kooperation innerhalb des Projektkernteams ist ein essenzieller Bestandteil während der Projektumsetzung und muss gegebenenfalls verstärkt werden. (Wagner & Grau, 2014)

Die ständige Datenkonsolidierung hinsichtlich des Projektfortschrittes sowie die Erfassung von Lessons Learned, zur späteren Analyse ungeklärter Themen, erfolgen ebenso während der

Projektumsetzung. Diese offenen Punkte werden in einem sogenannten Themenspeicher festgehalten. Die resultierenden Informationen zum Projektstatus müssen allen StakeholderInnen transparent kommuniziert werden und jederzeit abrufbar sein. (Wagner & Grau, 2014)

Controlling

Die Projektarbeit muss regelmäßig kontrolliert werden. Im Wesentlichen geht es beim Controlling darum, die Umsetzung von Projekten zu überwachen, zu messen und zu kontrollieren. Bei Notwendigkeit werden vorbeugende und korrigierende Maßnahmen eingeleitet sowie Änderungen beauftragt. In der ISO 21500 werden Prozesse zu mehreren „Controlling"-Bereiche beschrieben (Wagner & Grau, 2014):

- Leistungscontrolling,
- Ressourcencontrolling und
- Termincontrolling

Die Bereiche werden dem Controlling unterzogen, damit (Wagner & Grau, 2014):

- der derzeitige Projektstatus erkannt wird,
- eventuelle Abweichungen vorzeitig identifiziert werden sowie
- passende Maßnahmen eingeleitet werden.

Prognosen helfen, den weiteren Projektverlauf zu steuern. Ziel ist es, die positiven Auswirkungen auf das Projekt zu erhöhen und negative Abweichungen, die auch zu Änderungswünschen führen können, zu reduzieren. Alle Resultate werden in den beiden Prozessen „Controlling der Projekttätigkeiten" und „Controlling von Änderungen" eingepflegt. Ersterer wird kontinuierlich verwendet, um alle Projektaktivitäten abzuschließen. Das Ergebnis dieses Prozesses kann daher eine Änderungsanforderung, ein Fortschrittsbericht und/oder ein Abschlussbericht sein. Die Resultate des Controllings aus den einzelnen Bereichen werden konsolidiert. (Wagner & Grau, 2014)

In der ISO 21500 sticht das „Projektteammanagement" im Kontrollprozess hervor. Feedback und Kommunikation zielen darauf ab, die Gesamtleistung des Teams zu verbessern, auftretende Probleme offen zu lösen und zu positiven MitarbeiterInnenbewertungen in der Organisation beizutragen. (Wagner & Grau, 2014)

Abschluss

Die ISO 21500 bietet zum Ende eines Projekts zwei Prozesse (Wagner & Grau, 2014):

- Projektphase oder Projekt abschließen
- Lernerfahrungen sammeln

Am Ende eines Projekts überprüfen ProjektleiterInnen, ob tatsächlich alle Leistungen erbracht, die Projektmanagementprozesse ordnungsgemäß verwendet und die Resultate entsprechend

festgehalten wurden. Die ProjektleiterInnen sorgen dafür, dass alle ProjektmitarbeiterInnen sowie Ressourcen im Projekt wieder freigegeben werden. Das Fazit wird im Abschlussbericht festgehalten. Im Projektverlauf kann es auch Situationen geben, in denen das Projekt vorzeitig beendet werden muss. Ist dies der Fall, muss der Abschlussprozess eingeleitet und das Projekt offiziell abgeschlossen werden. (Wagner & Grau, 2014)

Der letzte Schritt besteht darin, das Projekt zu evaluieren, alle Erfahrungen zu sammeln und aufzuzeichnen. Der Zweck dieses finalen Schrittes ist, aus diesem Projekt und anderen Projekten für die Linearorganisation zu lernen. Die Erfassung von Lernerfahrungen kann in allen Projektmanagementprozessen erfolgen, jedoch ist es wichtig, alle Erfahrungen spätestens zum Projektabschluss zu erfassen und entsprechend bereitzustellen. (Wagner & Grau, 2014)

2.4.2 Requirements Engineering

Requirements Engineering ist ein fester Bestandteil der Softwareentwicklung. Es verfügt über verschiedene Methoden, Verfahren und Techniken zur Ermittlung, Analyse, Spezifikation und Überprüfung von Anforderungen. Requirements, die auch als Anforderungen bezeichnet werden, werden von AnforderungsingenieurInnen entwickelt. Diese Anforderungen müssen SoftwareentwicklerInnen verstehen, um die gewünschte Software entwickeln zu können. Die AnforderungsingenieurInnen müssen sich über die verschiedenen Funktionalitäten und Einschränkungen des zu entwickelnden Systems im Klaren sein und analytisch Zusammenhänge erkennen können. Das Requirements Engineering ist eine der kritischen Phasen im Lebenszyklus der Softwareentwicklung. Von dieser initialen Anforderungsanalyse hängen Erfolg oder Misserfolg eines Softwaresystems ab. Es handelt sich um den Prozess des Sammelns, Verstehens, Formulierens, Dokumentierens sowie der Verwaltung von Anforderungen – für die Verwaltung der Softwareanforderungen ist ein systematischer und disziplinierter Ansatz erforderlich. (Tripathi & Goyal, 2014)

AnforderungsingenieurInnen arbeiten mit folgenden Zielen (Tripathi & Goyal, 2014):

- AnforderungsIngenieurInnen konzentrieren sich auf die Wünsche der KundInnen und StakeholderInnen. Sie erzeugen und verwalten die Anforderungen – diese verringern das Risiko eines Ausfalls der zukünftigen Software und Erreichen die vollständige Erfüllung der KundInnenanforderungen.
- AnforderungsingenieurInnen erkennen die relevanten Informationen, beseitigen Konflikte zwischen den StakeholderInnen und unterstützen somit bei der Konsensfindung, die als Basis neuer Anforderungen dient. Diese müssen mit vorgegebenen Standards eindeutig dokumentiert und systematisch verwaltet werden.

Die Qualität eines Softwareprojekts wird durch den Anforderungsentwicklungsprozess bestimmt. Zur Verwaltung der Anforderungen muss das Projektteam mehrere Praktiken und Prozesse befolgen. Diese Prozesse spielen eine wichtige Rolle für den Erfolg des Projekts. Heutzutage arbeiten viele Softwareprojekte in agilen Umgebungen – auf diese Weise wird die schnelle Lieferung hochwertiger Software gewährleistet. Ein Projektteam, das an der Einführung agiler Prozesse und Praktiken beteiligt ist, muss offen und flexibel genug sein, um sich an ändernde

Anforderungen und ein dynamisches Umfeld anzupassen. Bei Projekten mit agilen Methoden werden häufig Anforderungsänderungen durchgeführt. Die Anforderungen ändern sich folglich im Laufe des Projektes, wobei die Änderungen dokumentiert sowie versioniert werden. (Tripathi & Goyal, 2014)

In der agilen Methodik wird das Verstehen von Softwareanforderungen durch die kontinuierliche Kommunikation mit den KundInnen erreicht. Die KundInnen liefern die Anforderungen in Form von User Stories, welche aus KundInnenanforderungen sowie Akzeptanzkriterien bestehen. Die/der Product Owner übernimmt die Priorisierung all dieser Anforderungen und beschreibt diese den EntwicklerInnen zu Beginn jeder Iteration. Ebenso beteiligen sich Product Owner an der Produktentwicklung sowie Demonstration des Produkts vor KundInnen. Product Owner können ein oder mehrere Business AnalystInnen haben, die ihr/ihm bei der Dokumentation formaler Anforderungen helfen können. Des Weiteren schlägt das Requirements Engineering die Verwendung systematischer und wiederholbarer Techniken vor, um die Vollständigkeit, Konsistent und Relevanz der Projektanforderungen sicherzustellen. (Tripathi & Goyal, 2014)

Das agile Requirements Engineering ist ein vergleichsweise neues und aufstrebendes Fachgebiet, das den Prozess der Anforderungsanalyse flexibler und effizienter gestalten soll. Es fokussiert sich nicht darauf, die Anforderungen zu Beginn des Projekts zu erfüllen, sondern bietet einen iterativen, inkrementellen Ansatz, der agile Teams dazu anleitet, sich auf die schnelle Lieferung von Produkten innerhalb einer kurzen Zeitspanne, das heißt maximal einem Monat, zu konzentrieren. Ebenso steht beim agilen Requirements Engineering die ständige Kommunikation zwischen KundInnen und agilen TeamentwicklerInnen im Fokus. Infolgedessen liefern EntwicklerInnen Softwaresysteme, welche die Erwartungen der KundInnen voll erfüllen und somit den Geschäftswert steigern. (Tripathi & Goyal, 2014)

Der kontinuierliche Prozess des agilen Requirements Engineering wird in folgender Abbildung dargestellt:

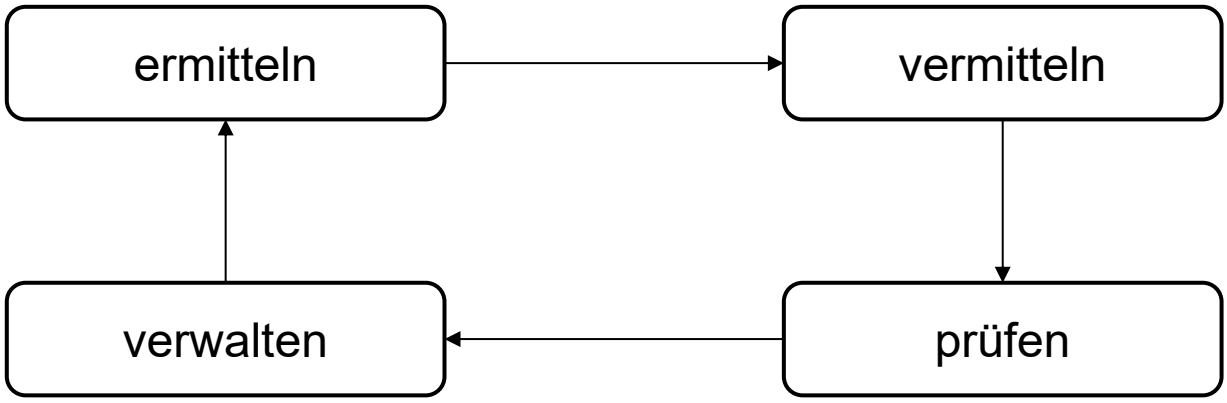

Abbildung 5 - Agiles Requirements Engineering (in Anlehung an (Lindner, 2016))

2.4.3 Scrum und agile Softwareentwicklung

Die agile Softwareentwicklung ist eine spezifische Methode innerhalb der Softwareprojektbranche. Unter dem Begriff „agil“ versteht man Schnelligkeit, Leichtigkeit, Flexibilität sowie Aufmerksamkeit. Zudem beschreibt „agil“ ein Prozessmodellkonzept, welches sich von bestehenden Konzepten oder Vorgangsweisen unterscheidet (Martin, 2002). Ursprünglich prägten Kent Beck und seine 16 ArbeitskollegInnen die Idee der agilen Softwareentwicklung. Die erstmalige Definition besagte, dass Software entwickelt wird, indem sie gleichzeitig aufgebaut sowie die beteiligten Personen unterstützt werden (Dingsøyr, Dybå, & Moe, 2010). Ein wesentlicher Unterschied zu anderen Vorgangsweisen in Softwareprojekten ist, dass Interaktionen und das Personal wichtiger sind als der Prozess oder die Werkzeuge selbst. Eine funktionierende Software ist wichtiger als eine vollständige Dokumentation, die Beziehungen mit KundInnen ist wichtiger als Vertragsgestaltungen und auf Änderungen zu reagieren ist wichtiger als einem Projektplan zu folgen. Jedenfalls besitzt die agile Softwareentwicklung, wie auch alle anderen Prozessmodelle, ihre spezifischen Vorteile. Folglich ist diese Art von Softwareprojekten nicht für jedes Projekt, Produkte, Personen oder Situationen geeignet. Des Weiteren wird ein Prozessmodell ermöglicht, das Änderungen in den Anforderungen toleriert und somit schneller und effizienter auf KundInnenwünsche reagiert werden kann. (Permana, 2015)

Scrum wurde 1993 von Jeff Sutherland entwickelt und ist ein responsives Framework für die Entwicklung von Softwareprojekten, Produktmanagement oder auch einzelnen Applikationen (Pham & Pham, 2011). Der Fokus hierbei liegt auf der Strategie, eine flexible sowie ganzheitliche Produktentwicklung durchzuführen, in welcher das Projektteam als eine Einheit arbeitet, um auf diese Weise die gemeinsamen Ziele zu erreichen. Das agile Framework beinhaltet einen Prozess, in dem viele Faktoren das Endergebnis beeinflussen. (Permana, 2015)

Folgende Rollen werden in Scrum vergeben (Woodward & Surdek, 2010):

- Product Owner: ProjektmanagerInnen und Business AnalystInnen
- Scrum Master: SystemanalystInnen
- Entwicklungsteam: ProgrammiererInnen und SoftwaretesterInnen

Die/der Product Owner ist die Person, die für die Festlegung der Anforderungsspezifikationen des zu entwickelnden Softwaresystems verantwortlich ist. Sie/er registriert alle initialen Anforderungen, die vom Entwicklungsteam zu erfüllen sind. Diese Ansammlung von Anforderungen wird Product Backlog genannt. Die Verantwortlichkeit des Projekts obliegt dem gesamten Scrum-Team, also sind beispielweise SoftwaretesterInnen und Business AnalystInnen gleichermaßen für die Qualität der Software verantwortlich. Zudem ist das Scrum-Team gemeinsam für das Abschließen des Product Backlogs, welches von Product Ownern gemeinsam mit KundInnen konzipiert wurde, verantwortlich. Das gesamte Team muss also anhand des Backlogs zu jedem Zeitpunkt wissen, welcher individuelle Schritt als Nächstes getan werden muss. Die/der Scrum Master definiert und optimiert die Scrum-Prozesse innerhalb des Projektes. Des Weiteren erklären die Scrum Master den Teams, wie das agile Framework funktioniert und was dieses für die Prozesse innerhalb des Projektes bedeuten. Auf diese Weise wird

gewährleistet, dass alle Projektmitglieder nach den Scrum-Methoden arbeiten und die Prozesse optimal eingeführt werden. (Permana, 2015)

Ein auf der Scrum-Vorangehensweise basierendes Projekt beginnt mit einer Beschreibung des zu liefernden Softwaresystems. Daraufhin leitet die/der Product Owner die Geschäftsprozesse des gewünschten Systems ab und transformiert diese in das Product Backlog (Pham & Pham, 2011). Das Product Backlog wird allgemein als eine Aufgabenliste betrachtet, welches vom Projektteam abgearbeitet wird. Ein weiterer bekannter Begriff im agilen Framework ist der Sprint. Dieser ist ein Zeitraum mit einem vordefinierten zeitlichen Anfang sowie Ende und beinhaltet Ziele. Diese Sprintziele definieren, welche neuen Funktionalitäten im Softwareprodukt am Ende des Sprints an die KundInnen geliefert werden. Die Dauer eines Sprints beläuft sich auf zwei oder vier Wochen. Jeder Sprint beginnt mit einem Sprint Meeting Planning, in welchem beschlossen wird, welche Backlogs im nächsten Sprint entwickelt werden. Zusätzlich dazu finden sich die Projektmitglieder jeden Tag in Daily Scrum Meetings zusammen und diskutieren untereinander: „Was wurde seit dem letzten Daily Meeting erledigt?“ oder „Welche Probleme sind bei der Arbeit aufgetaucht?“ (Falls, 2004). Die Besprechungen werden von Scrum Mastern moderiert und am Ende des Sprints findet ein Sprint Review statt, in dem die entwickelten Funktionalitäten den AuftraggeberInnen präsentiert werden. (Permana, 2015)

Der beschriebene Sprintprozess wird in folgender Abbildung visuell dargestellt:

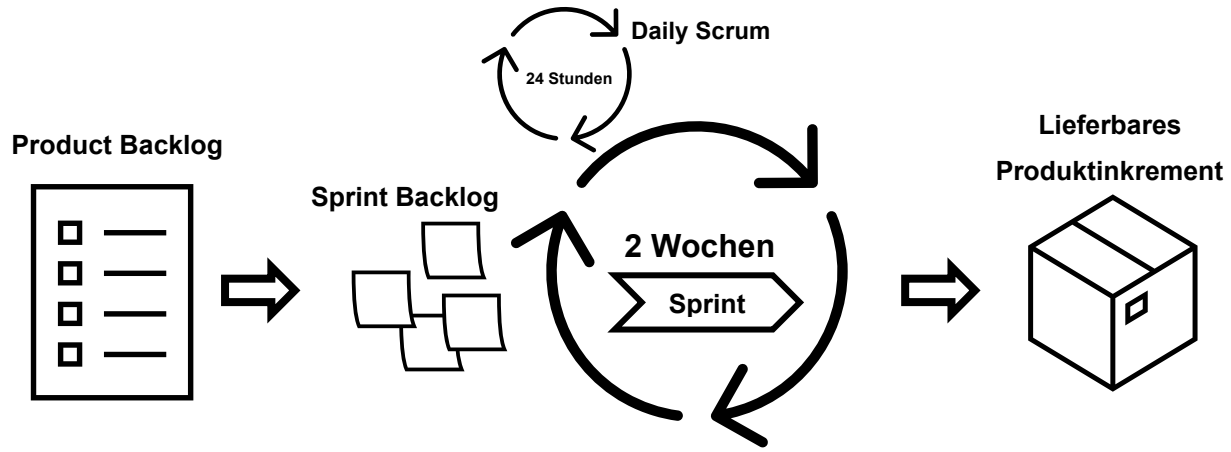

Abbildung 6 - Sprint in Scrum (in Anlehnung an (Permana, 2015))

Das Sprint Review und die Sprint Retrospective werden im Folgenden noch genauer beschrieben, um schlussendlich das gesamtheitliche Zusammenspiel im agilen Framework zu veranschaulichen.

Die Demonstration der neu entwickelten Softwarefunktionalitäten für die KundInnen wird vom gesamten Projektteam durchgeführt. Dies erfolgt im Zuge des Sprint Reviews. Diese Präsentation besteht nur aus Softwaredemos und auch der Einsatz von Präsentationtools entfällt. Der Fokus liegt zur Gänze auf dem Produkt. Auf diese Weise wird den KundInnen der aktuelle Fortschritt

ihrer Software offenbart und sie erhalten ein besseres Bild von ihrem Produkt. Während der Demonstration können KundInnen ihre Meinung darüber mitteilen, welche zusätzliche Änderungen sie in der Funktionalität bestellen möchten. Diese können dann bis zum Ende des nächsten Sprints implementiert werden. (Permana, 2015)

Die Sprint Retrospective ist ein internes Meeting des Scrum-Teams und wird von den Scrum Mastern moderiert. Das Hauptthema sind hierbei die Probleme des letzten Sprints und wie diese zukünftig verhindert werden können. Durch diese Offenheit erhalten Scrum Master wertvollen Input und können den agilen Prozess stets weiter optimieren. (Permana, 2015)

Die folgende Abbildung visualisiert den beschriebenen iterativen Scrum-Prozess:

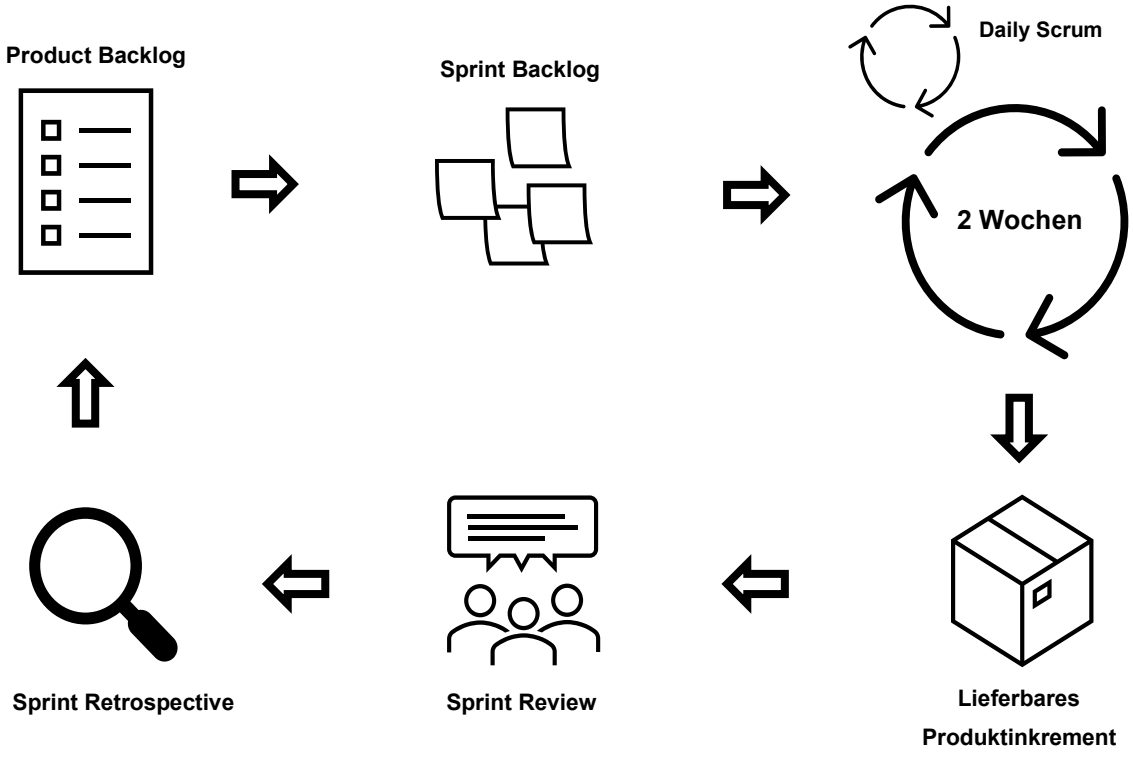

Abbildung 7 - Iterativer Scrum Prozess (in Anlehnung an (Permana, 2015))

2.5 Personalwesen

Das Personal für verteilte Softwareprojektteams soll vor Projektstart beschafft sowie in bestehende Strukturen eingeführt werden. Schlussfolgernd ist das Ziel dieses Kapitels das Personalwesen theoretisch zu erläutern, wobei ein spezieller Fokus auf Personalbeschaffung sowie Personaleinführung gerichtet wird.

2.5.1 Beschaffung von Personal

Das Werk „*Personalbeschaffung im Web 2.0*“ von Roy Engel (2014) wird für dieses Subkapitel als theoretische Grundlage herangezogen. Die Personalbeschaffung konzentriert sich auf die Eingliederung neuer MitarbeiterInnen in das eigene Unternehmen. Zudem ist es wichtig, dass die MitarbeiterInnen den Bedürfnissen des Unternehmens quantitativ und qualitativ genügen. Die Schwierigkeit an dieser Stelle ist, einen kostenoptimalen Rekrutierungsaufwand je MitarbeiterIn zu halten. Es gibt mehrere Möglichkeiten, neues Personal zu beschaffen. Je nach benötigter Qualifikation oder Funktion eignet sich eine andere Beschaffungsart. (Engel, 2014)

In folgender Abbildung werden die Möglichkeiten der Personalbeschaffung übersichtlich dargestellt:

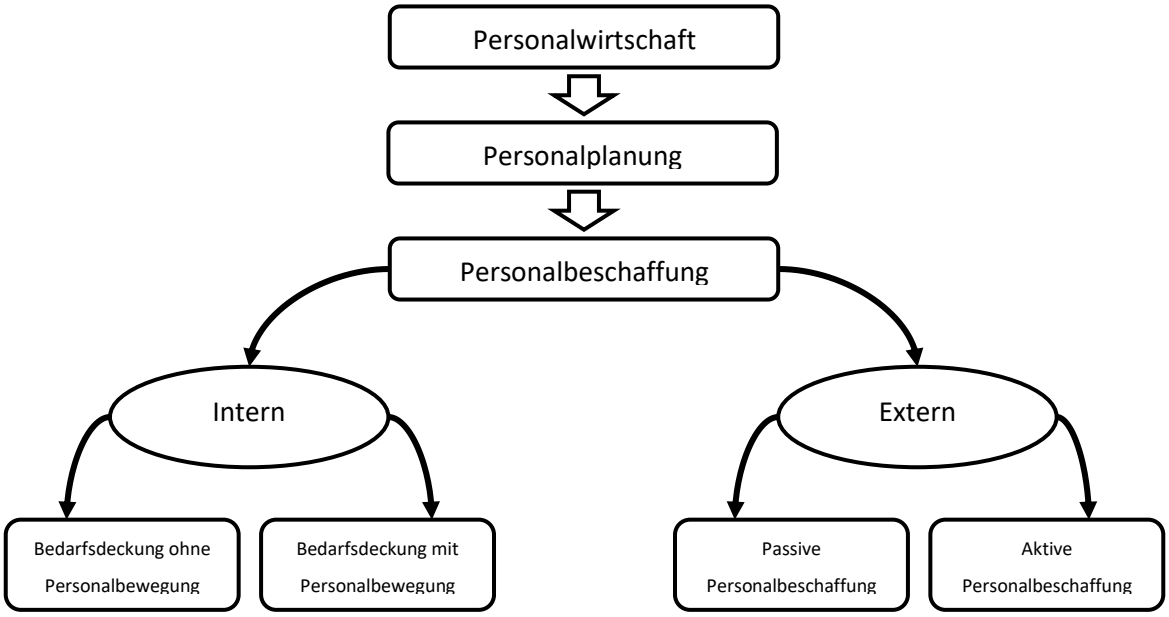

Abbildung 8 - Möglichkeiten der Personalbeschaffung (in Anlehnung an (Engel, 2014))

Bei der internen Personalbeschaffung werden schon bestehenden MitarbeiterInnen neue Funktionen angeboten oder zugewiesen. Das Personal kann dies als positiven Schritt in seiner Karriere wahrnehmen, was zu einer Steigerung der Motivation führt. Möglichkeiten zur internen Personalbeschaffung sind (Olfert, 2012):

- Personalentwicklung
- Interne Ausschreibung

- Überstunden

Ist der Personalbedarf mit internen Möglichkeiten nicht zu decken, muss neues externes Personal herangezogen werden. (Lindner-Lohmann, Lohmann, & Schirmer, 2016)

Die externe Personalbeschaffung bezieht potenzielle MitarbeiterInnen außerhalb des Unternehmens. Sie wird angewendet, wenn die interne Variante nicht sinnvoll erscheint. Folgende Beispiele zur externen Beschaffung existieren (Engel, 2014):

- Internetportale
- Personalberatungen, Headhunter
- Staatliche Arbeitsvermittlungen
- Printmedien
- Abwerbung

Beide Varianten der Personalbeschaffung bergen ihre Vor- und Nachteile. Personalverantwortliche müssen situationsbedingt abwägen können, welche Art der Beschaffung für sie am sinnvollsten erscheint. (Olfert, 2012)

2.5.2 Einführung von Personal

Als theoretische Grundlage wird hier das Werk „*Erfolgsfaktor Inplacement: Neue Mitarbeiter systematisch und zielgerichtet integrieren*“ von Nicole Blum (2010) herangezogen. Nach der Personalbeschaffung müssen die Projektmitglieder ordnungsgemäß in den Projektalltag eingeführt und integriert werden. Der Prozess der Integration wird auch als „organisationale Sozialisation“ beschrieben (Blum, 2010). Die Autoren Friedemann Nerdinger, Gerhard Blickle sowie Niclas Schaper erläutern den Begriff wie folgt:

> *„Mit dem Begriff organisationale Sozialisation wird der Prozess der Vermittlung und des Erwerbs von Kenntnissen, Fertigkeiten, Fähigkeiten, Überzeugungen, Werthaltungen und Normen beschrieben, der eine Person dazu befähigt, die von der Organisation an sie gestellten Handlungsanforderungen zu erfüllen.“* (Nerdinger, Blickle, & Schaper, 2011)

Grundsätzlich gibt es drei verschiedene Ebenen der Integration, die neue MitarbeiterInnen durchleben (Blum, 2010):

1. Fachliche Einarbeitung
2. Soziale Integration
3. Wertorientierte Integration

Zuerst muss neues Personal im Zuge der fachlichen Einarbeitung ihr Verantwortungsgebiet sowie die dazugehörigen Aufgaben tiefgreifend kennenlernen (Holtbrügge, 2017). Dabei müssen die MitarbeiterInnen allgemeines Wissen über die Produkte und Strukturen des Unternehmens sowie spezifisches Wissen über ihr Aufgabengebiet aufbauen. Zusätzlich werden die neuen

MitarbeiterInnen in die Arbeitsabläufe, Regelungen im Betrieb sowie Prozesse eingeschult. (Blum, 2010)

Neben der fachlichen Perspektive ist die soziale Integration eine ebenso relevante Ebene zur Einführung neuer MitarbeiterInnen. Diese knüpfen soziale Kontakte mit ihren neuen Vorgesetzten, KollegInnen sowie KundInnen. Auf diese Weise erlangt neues Personal Akzeptanz ihn seinem Arbeitsumfeld und das individuelle Ansehen innerhalb der Organisationsstruktur steigt. Die soziale Integration ist weitaus kritischer als die fachliche Einarbeitung, da mehr Reibungspunkte entstehen können. Hinzukommend müssen sich neue MitarbeiterInnen selbst als Teil der Arbeitsgemeinschaft fühlen, erst dann ist die soziale Integrationsphase abgeschlossen. (Brenner, 2014)

Die letzte Ebene ist die wertorientierte Integration. Sie beschreibt, dass jedes Unternehmen über eigene Vorstellungen in Bezug auf Werte und Ziele besitzt. Diese Grundsätze werden oft als Unternehmensleitlinien formuliert. Der Inhalt dieser Grundsätze definiert das Verhalten der MitarbeiterInnen gegenüber Vorgesetzten, KollegInnen sowie KundInnen (Hubert & Pionczyk, 2006). Damit das Personal diese Erwartungen erfüllen kann, muss es zunächst die Werte und Zielvorstellungen vermittelt bekommen. Im Unterschied zur fachlichen Einarbeitung oder sozialen Integration ist die wertorientierte Integration ein mittel- bis langfristiger Prozess, welcher täglich von den MitarbeiterInnen erlebt wird. (Brenner, 2014)

2.6 Verrechnung

Softwareprojekte mit verteilten Teams sollen KundInnen korrekt verrechnet werden. Aus diesem Grund ist die strukturierte Erfassung der Personalkosten für die verteilten EntwicklerInnen essenziell. In diesem Kapitel werden daher die Verrechnung der Projekte sowie die Erfassung Personalkosten erläutert.

2.6.1 Projektverrechnung

Die lückenlose und rasche Rechnungsstellung ist unter kaufmännischen Gesichtspunkten eine maßgebliche Unternehmensaufgabe. Im Geschäftsalltag wird sie trotzdem des Öfteren vernachlässigt. Hauptgründe hierfür, dass gebrauchte Leistungsdaten nicht zu jedem Zeitpunkt vorbereitet sind und aus unterschiedlichen Quellen zusammengetragen werden müssen. Hinzu kommt, dass Rechnungen meist mit Standardsoftware, wie Word oder Excel erstellt werden. Erreicht die Projektanzahl im Unternehmen eine gewisse Größe oder sind etliche ArbeitnehmerInnen, wie BeraterInnen und IngenieurInnen, an den Projekten beteiligt, verursacht die Projektverrechnung einen großen Aufwand und birgt überdies eine Vielzahl an Fehlerquellen. Detaillierte Leistungsnachweise gegenüber den AuftraggeberInnen gestalten sich zunehmend schwierig. Aufgrund dessen soll die Verrechnung im Rahmen des Projektmanagements keine separate Aktivität sein, sondern ein fixer Bestandteil des vollständigen Dienstleistungsprozesses. Relevante Verrechnungsdaten sollen möglichst automatisch aus den vorangegangenen Abläufen aufbereitet werden. (Marxer, 2016)

Anstelle von Office-Anwendungen verwenden einige DienstleisterInnen spezifische Abrechnungs- oder Buchhhaltungsoftware. Doch auch diese Systeme sind nicht in der Lage, sämtliche für die Projektverrechnung relevanten Daten automatisch bereitzustellen. Die Folge ist zumeist eine manuelle Übertragung von Wissen, die einen Medienbruch mit all seinen negativen Folgen darstellt. Werden Informationen im Vorsystem im Nachgang geändert, steigern sich Aufwand und Fehlerquote stetig. Abhilfe für eben diese Problematik leistet schlussendlich allein eine kaufmännische Projektsoftware, die sämtliche Projektphasen samt der Rechnungsstellung abbildet. (Marxer, 2016)

2.6.2 Personalaufwandserfassung

Personalkosten tragen einen großen Anteil an den Gesamtkosten eines Projekts, deshalb ist eine genaue Erfassung der Personalaufwendungen in Projekten essenziell. Ziel der Erfassung ist die Transparenz der Kosten. Die Schaffung von Transparenz ist jedoch nicht nur ein Anspruch der Personalkostenerfassung, sondern ein allgemeines Ziel der Leistungsverrechnung. Bei den Personalaufwendungen ist insbesondere ein Soll-/Ist-Vergleich, der Aussagen darüber liefert, wie effektiv die MitarbeiterInnen wirken oder inwieweit, die Soll-Vorgaben revidiert werden sollten, von Bedeutung. Die Arbeitsstunden werden von den einzelnen MitarbeiterInnen manuell erfasst. Diese können jedoch Zeitbuchungen verfälschen, falls keine konsequente Überwachung stattfindet. Das ganze Verrechnungssystem verliert an Aussagekraft, wenn das Datenmaterial nicht vollständig und aktuell ist. Folglich muss man die ArbeitnehmerInnen dahingehend motivieren, Zeitbuchungen wahrheitsgemäß und zeitnah wie möglich einzugeben. (Humberg, 2002)

MitarbeiterInnen buchen ihre Arbeitsstunden für getätigte Arbeitsschritte. Solche Arbeitsschritte können Projektteilaufgaben oder allgemeine Tätigkeiten sein. Diese müssen innerhalb der Leistungsverrechnung identifiziert und dargestellt werden. Zum einen müssen Projektteilaufgaben sowie sämtliche auftretende Aufgaben in der Managementsoftware modelliert werden, zum anderen müssen ebenso Prozesse gegeben sein, die es ermöglichen, die verschiedenen allgemeinen Arbeitsabläufe, wie Arbeitsvorbereitung, Weiterbildung, Einarbeitung oder Organisatorisches, zu buchen. (Humberg, 2002)

Zusätzliche Aussagefähigkeit auf Basis der gewonnenen Informationen ergibt sich, wenn die Teilaufgaben den verschiedenen Phasen des Software-Lebenszyklus zugeordnet werden können. Eine derartige Betrachtung könnte insbesondere für SoftwaredienstleisterInnen relevant sein, die hieraus empirische Vorhersagen in Bezug auf die Relation zwischen Entwicklungsaufwand sowie Wartung der Software ableiten. Die Erfassung der Personalkosten ist eine große Herausforderung. Nicht bloß für das Projektmanagement selbst, stattdessen auch für ein unternehmensweites Informationsmanagement kann die genaue Personalkostenerfassung von großer Relevanz sein. (Humberg, 2002)

3 ENTWICKLUNG DES PROZESSMODELLS

Auf Basis der theoretischen Erkenntnisse aus Kapitel *2 Theoretischer Ansatz des Prozessmodells* werden folgende Problemstellungen abgeleitet und in weiterer Folge ihre Beziehung zueinander analysiert:

- Wie kann ein Softwareprojekt mit verteilten SoftwareentwicklerInnen initiiert werden?
- Wie wird ein verteiltes Softwareentwicklungsteam rekrutiert und wie läuft der Bewerbungsprozess ab?
- Wie läuft die Ressourcenplanung bei verteilten SoftwareentwicklerInnen ab und wie sieht die Aufbau- sowie Ablauforganisation bei verteilten Softwareprojekten aus?
- Wie werden die Umsetzung und das Controlling bei Projekten mit verteilten SoftwareentwicklerInnen durchgeführt?
- Wie können verteilte SoftwareentwicklerInnen ein besseres Verständnis für die KundInnen- und Softwareanforderungen im Projekt erlangen?
- Wie kann man bei Abschluss eines verteilten Softwareentwicklungsteams die Lernerfahrungen mit verteilten SoftwareentwicklerInnen identifizieren und verwalten?

Im aktuellen Kapitel wird je definierter Fragestellung ein eigenes Subkapitel erstellt und die Begründung der einzelnen Fragen beschrieben. Des Weiteren werden Verweise auf die theoretischen Erkenntnisse aus Kapitel *2 Theoretischer Ansatz des Prozessmodells* eingefügt. Auf diese Weise wird eine Rückverfolgbarkeit zur Literaturrecherche gewährleistet und die LeserInnen erhalten ein besseres Verständnis darüber, weshalb die definierten Problemstellungen für die Forschungsfrage relevant sind. Je Subkapitel wird nach der Problembegründung ein Prozessmodell inklusive Beschreibungen dargestellt. Diese Prozessmodelle sollen mögliche Antworten auf die definierten Problemstellungen bereitstellen und schlussendlich ein Gesamtprozessmodell ergeben.

Die dargestellten Prozessmodelle sind ebenso in der beigelegten „Compact Disc“ (CD) dieser Arbeit archiviert – der Verzeichnisname lautet „Ergebnisse“. Auf diese Weise wird den LeserInnen Zugriff auf die Detailansichten der Abbildungen geboten.

3.1 Initiierung eines verteilten Softwareentwicklungsprojektes

Dieses Kapitel beschreibt die Fragestellung *„Wie kann ein Softwareprojekt mit verteilten SoftwareentwicklerInnen initiiert werden?“* und liefert dahingehend ein Prozessmodell als mögliche Antwort. Zusätzlich wird das Prozessmodell anhand der darin inkludierten Aktivitäten beschrieben und eine Verbindung zur vorhergehenden theoretischen Recherche hergestellt.

3.1.1 Beschreibung des Problems

Softwareprojekte müssen intern einen Initiierungsprozess durchlaufen, um in weiterer Folge geplant werden zu können. Im Initiierungsprozess muss das verteilte Softwareprojekt durch einen internen Projektauftrag formal genehmigt werden. Als Basis-Input dient hier ein Realisierungsvertrag mit den KundInnen. Zudem müssen eine interne Leistungsbeschreibung dokumentiert sowie ein/e ProjektleiterIn vor Ort für das verteilte Softwareprojekt bestimmt werden. (siehe Kapitel *2.4.1 Projektmanagement*).

Die grundsätzliche Herausforderung bei verteilten Softwareprojekten liegt in der Zusammensetzung des verteilten Projektteams während der Initiierung (siehe Kapitel *2.4.1 Projektmanagement*). Dennoch muss die Initiierung im Rahmen dieser Arbeit berücksichtigt und modelliert werden, da basierend der beinhaltenden internen Leistungsbeschreibung geeignetes Personal für das verteilte Softwareprojekt rekrutiert werden muss (siehe Kapitel *2.5.1 Beschaffung von Personal*).

Bei der Auswahl der Projektmitglieder können auch ProjektleiterInnen beteiligt sein, um die geeignetsten KandidatInnen zu finden. Hierbei werden die individuellen Fähigkeiten sowie das Know-how der SoftwareentwicklerInnen bewertet. Anschließend werden die Rollen und Verantwortlichkeiten der potenziellen Projektmitglieder definiert sowie die vereinbarten Kommunikationswege zwischen den Personen vereinbart. (siehe Kapitel *2.4.1 Projektmanagement*)

3.1.2 Prozessmodell

Angesichts der theoretischen Erkenntnisse wird folgendes Subprozessmodell „*Initiierung eines verteilten Softwareprojektes*“ entwickelt:

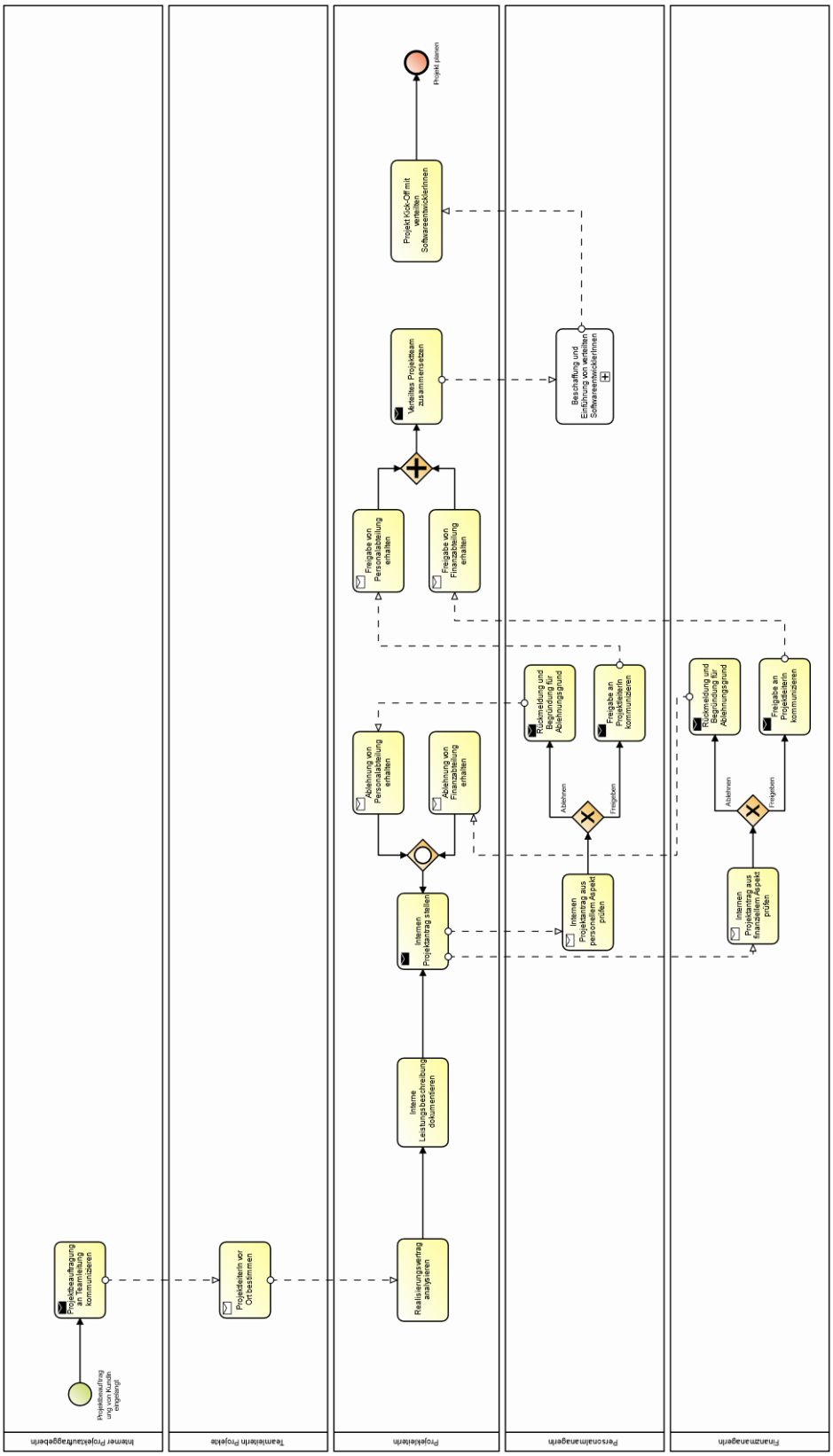

Abbildung 9 - P1: Initiierung eines verteilten Softwareprojektes

3.1.3 Beschreibung des Prozessmodells

Das entwickelte Subprozessmodell *„Initiierung eines verteilten Softwareprojektes“* wird im Folgenden durch die darin inkludierten Aktivitäten beschrieben.

Projektbeauftragung an Teamleitung kommunizieren

Der Vertrieb kommuniziert, als interner Projektauftraggeber, die Projektbeauftragung der KundInnen an die Teamleitung der ProjektleiterInnen. Der vereinbarte Realisierungsvertrag wird ebenso an die Teamleitung übermittelt.

ProjektleiterIn vor Ort bestimmen

Die/der TeamleiterIn der ProjektleiterInnen bestimmt die/den ProjektleiterIn für das verteilte Softwareprojekt. Die Teamleitung sowie die verfügbaren ProjektleiterInnen sind vor Ort beschäftigt und nicht standortspezifisch verteilt. Der vereinbarte Realisierungsvertrag wird ebenso an die Projektleitung übermittelt.

Realisierungsvertrag analysieren

Der/die ProjektleiterIn analysiert den unterzeichneten Realisierungsvertrag. Innerhalb dieser Analyse müssen ProjektleiterInnen die groben KundInnenanforderungen bestimmen, den groben Terminplan ableiten sowie grob die benötigten internen Ressourcen ermitteln.

Interne Leistungsbeschreibung dokumentieren

Der/die ProjektleiterIn muss die ermittelten Daten aus der Aktivität *„Realisierungsvertrag analysieren“* in der internen Leistungsbeschreibung dokumentieren. Die interne Leistungsbeschreibung ist ein unternehmensstandardisiertes Dokument mit vorgegebenen Eingaben (siehe Kapitel *2.4.1 Projektmanagement*). Zusätzlich müssen neben den groben KundInnenanforderungen, dem Terminplan und den notwendigen Ressourcen alle nötigen projektspezifischen Aktivitäten auf KundInnenseite, die Verrechnungsart sowie die Zahlungsmeilensteine beschrieben werden.

Internen Projektantrag stellen

Der/die ProjektleiterIn muss nach Fertigstellung der Leistungsbeschreibung einen internen Projektantrag stellen. Der interne Projektantrag basiert auf der dokumentierten Leistungsbeschreibung und wird an die Personal- sowie Finanzabteilung gestellt – dort findet die weitere Prüfung hinsichtlich der realisierbaren Projektdurchführung statt.

Internen Projektantrag aus personellem Aspekt prüfen

Ein/eine PersonalmanagerIn muss den internen Projektantrag aus der Aktivität „*Internen Projektantrag stellen*“ in personeller Hinsicht prüfen. Hierbei entscheiden PersonalmanagerInnen, welche Art der Personalbeschaffung für das verteilte Softwareprojekt Sinn macht und wie gegebenenfalls das neue Personal eingeführt werden soll (siehe Kapitel *2.5 Personalwesen*). Bei dieser Aktivität handelt es sich um eine Vorbereitung für den selbstständig geführten Prozess *„Beschaffung und Einführung von verteilten SoftwareentwicklerInnen“* (siehe Kapitel *3.2 Beschaffung und Einführung von verteilten SoftwareentwicklerInnen*).

Internen Projektantrag aus finanziellem Aspekt prüfen

Ein/eine FinanzmanagerIn muss den internen Projektantrag aus der Aktivität „*Internen Projektantrag stellen*“ in finanzieller Hinsicht prüfen. Hierbei setzen die FinanzmanagerInnen den Fokus auf die vereinbarte Verrechnungsart sowie die Zahlungsmeilensteine des Projektes – diese werden im weiteren Projektverlauf für Rechnungsstellungen relevant (siehe Kapitel *2.3.3 Vertrags- und Abrechnungsmodelle* und *2.6 Verrechnung*).

Rückmeldung und Begründung für Ablehnungsgrund

Personal- und FinanzmanagerInnen haben die Möglichkeit interne Projektanträge von ProjektleiterInnen abzulehnen. Unabhängig davon, welche Abteilung den Projektantrag ablehnt, muss der Projektantrag von ProjektleiterInnen neu gestellt werden. Bei einer Ablehnung müssen Personal- sowie FinanzmanagerInnen jedenfalls eine Begründung an die Projektleitung kommunizieren, damit diese die interne Leistungsbeschreibung anpassen und anschließend den Projektantrag erneut stellen kann.

Freigabe an ProjektleiterIn kommunizieren

Die Personalabteilung gibt die Freigabe für das verteilte Softwareprojekt nach der Prüfung hinsichtlich Personalbeschaffung sowie gegebenenfalls Personaleinführung (siehe Kapitel *2.5 Personalwesen*). Auf die Vorplanung im Rahmen der Prüfung kann im späteren Prozess *„Beschaffung und Einführung von verteilten SoftwareentwicklerInnen“* (siehe Kapitel *3.2 Beschaffung und Einführung von verteilten SoftwareentwicklerInnen*) zurückgegriffen werden.

Die Finanzabteilung gibt die Freigabe für das verteilte Softwareprojekt nach der Prüfung hinsichtlich Verrechnung und Zahlungsmeilensteine (siehe Kapitel *2.3.3 Vertrags- und Abrechnungsmodelle* und *2.6.1 Projektverrechnung*). Die FinanzmanagerInnen legen das verteilte Softwareprojekt im unternehmensweiten Zeiterfassungssystem an, um Leistungsdaten der verteilten SoftwareentwicklerInnen rückverfolgen zu können (siehe Kapitel *2.6.2 Personalaufwandserfassung*).

In beiden Fällen muss die Freigabe an die Projektleitung kommuniziert werden. Erst nachdem beide Freigaben erfolgt sind, wird die nächste Aktivität *„Verteiltes Projektteam zusammensetzen“* durch den/die ProjektleiterIn durchgeführt.

Verteiltes Projektteam zusammensetzen

Nach der Projektfreigabe durch Personal- sowie FinanzmanagerInnen erfolgt die Zusammensetzung des verteilten Projektteams. Die Projektleitung erhält Unterstützung von der Personalabteilung hinsichtlich der Personalbeschaffung (siehe Kapitel *2.5.1 Beschaffung von Personal*). Die ProjektleiterInnen können sich zudem im Personalprozess einbringen, da diese bereits anhand der dokumentierten internen Leistungsbeschreibung ein Vorwissen bezüglich des benötigten Know-hows sowie der individuellen Fähigkeiten mitbringen (siehe Kapitel *2.4.1 Projektmanagement*).

Beschaffung und Einführung von verteilten SoftwareentwicklerInnen

Dieser Prozess besteht aus selbständig geführten Aktivitäten und wird im Kapitel *3.2 Beschaffung und Einführung von verteilten SoftwareentwicklerInnen* genauer beschrieben.

Projekt Kick-Off mit verteilten SoftwareentwicklerInnen

Nach der Personalbeschaffung sowie gegebenenfalls Personaleinführung (siehe *3.2 Beschaffung und Einführung von verteilten SoftwareentwicklerInnen*) sind die verteilten SoftwareentwicklerInnen für das Softwareprojekt bestimmt. Der/die ProjektleiterIn initiiert einen internen Projekt-Kick-off mit den verteilten SoftwareentwicklerInnen, um auf diese Weise einen groben Überblick über das Softwareprojekt zu geben. Die dokumentierte, interne Leistungsbeschreibung kann hierbei als Unterstützung dienen, damit die verteilten SoftwareentwicklerInnen eine Übersicht hinsichtlich der Projektziele sowie des Projektzeitplans erhalten.

Nach dem Projekt-Kick-off wird das verteilte Softwareprojekt von ProjektleiterInnen und TeamleiterInnen detailliert aufgeplant (siehe Kapitel *3.3 Ressourcenplanung bei verteilten Softwareentwicklungsprojekten*).

3.2 Beschaffung und Einführung von verteilten SoftwareentwicklerInnen

Dieses Kapitel beschreibt die Fragestellung *„Wie wird ein verteiltes Softwareentwicklungsteam rekrutiert und wie läuft der Bewerbungsprozess ab?“* und liefert dahingehend ein Prozessmodell als mögliche Antwort. Zusätzlich wird das Prozessmodell anhand der darin inkludierten Aktivitäten beschrieben und eine Verbindung zur vorhergehenden theoretischen Recherche hergestellt.

3.2.1 Beschreibung des Problems

Die Personalabteilung evaluiert in der Projektfreigabephase, ob eine Personalbeschaffung oder gegebenenfalls Personaleinführung benötigt wird (siehe Kapitel *3.1 Initiierung eines verteilten Softwareentwicklungsprojektes*).

Die Personalbeschaffung basiert auf der unternehmensweiten Personalplanung (siehe Kapitel *2.5.1 Beschaffung* von Personal). Folglich muss die Geschäftsführung des Unternehmens in den Prozess involviert werden. Es existieren mehrere Arten der Personalbeschaffung, wobei diese grundsätzlich in zwei Gruppen unterteilt werden: interne und externe Personalbeschaffung (siehe Kapitel *2.5.1 Beschaffung* von Personal).

Zur internen Personalbeschaffung stehen folgende Möglichkeiten zur Verfügung (siehe Kapitel *2.5.1 Beschaffung* von Personal):

- Interne Personalbewegung zur Bedarfsdeckung – also Personalressourcen der verteilten Entwicklungsteams untereinander verschieben
- Interne Ausschreibung

Zur externen Personalbeschaffung stehen folgende Optionen zur Auswahl (siehe Kapitel *2.5.1 Beschaffung* von Personal):

- Passive Personalbeschaffung – einen Pool an vergangenen BewerberInnen nutzen
- Aktive Personalbeschaffung – aktiv für neue Stellen Personal rekrutieren

Innerhalb der aktiven Personalbeschaffung gibt es die Möglichkeit, externe PersonaldienstleisterInnen anzuheuern. Diese nutzen ihr spezielles Fachwissen, um die benötigten verteilten SoftwareentwicklerInnen zu finden. Die Unterstützung durch externe PersonaldienstleisterInnen wird als Supportprozess (siehe Kapitel *2.2.2 Geschäftsprozesse in Unternehmen*) wahrgenommen. Ebenso können in Internetportalen Stellenausschreibungen geschaltet werden – auf diese Weise können mehrere verteilte SoftwareentwicklerInnen auf die Position aufmerksam gemacht werden. (siehe Kapitel *2.5.1 Beschaffung* von Personal)

Die interne Personalbeschaffung muss mit den TeamleiterInnen der verteilten Entwicklungsteams abgestimmt werden. Die externe Personalbeschaffung muss wiederum mit den externen PersonaldienstleisterInnen koordiniert werden, da zusätzlich ein Bewerbungsgespräch mit der eigenen Personalabteilung sowie den TeamleiterInnen der verteilten Entwicklungsteams notwendig ist. Je Region wird mit einer/m andere/n PersonaldienstleisterIn kooperiert, da diese je geografischem Standort unterschiedlich mit verteilten SoftwareentwicklerInnen vernetzt sind. (siehe Kapitel *2.5.1 Beschaffung* von Personal)

In weiterer Folge müssen verteilte SoftwareentwicklerInnen langfristig in das Unternehmen eingeführt werden. Diese Einführung wird in folgende Punkte aufgeteilt (siehe Kapitel *2.5.2 Einführung* von Personal):

- Fachliche Einarbeitung in die verteilte Softwareprojektentwicklung
- Soziale Integration in das verteilte Softwareprojektteam
- Wertorientierte Integration in das Unternehmen

Die soziale sowie wertorientierte Integration sind langfristige Vorhaben und gehen somit über die verteilte Softwareprojektentwicklung hinaus (siehe Kapitel *2.5.2 Einführung* von Personal). Aus diesem Grund wird im Rahmen des zu entwickelnden Prozessmodells der Fokus auf die fachliche Einarbeitung gelegt.

3.2.2 Prozessmodell

Angesichts der theoretischen Erkenntnisse wird folgendes Subprozessmodell „*Beschaffung von verteilten SoftwareentwicklerInnen*“ entwickelt:

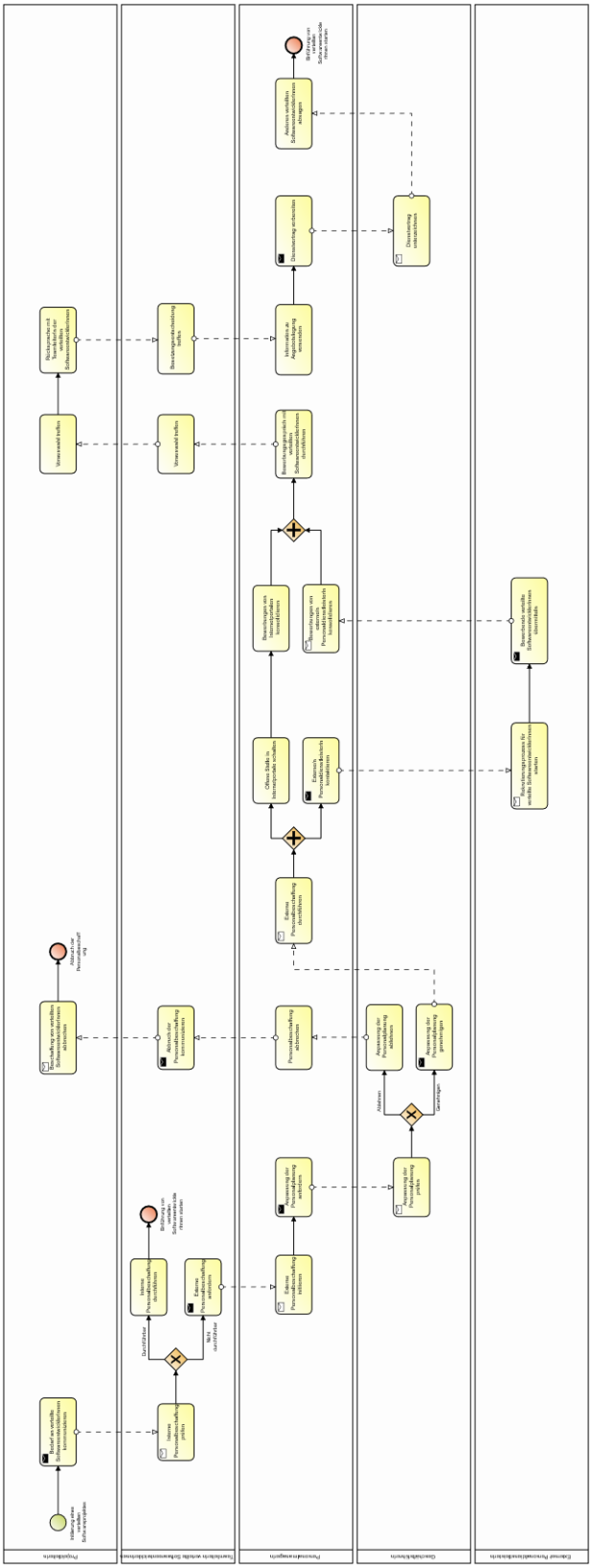

Abbildung 10 - P2: Beschaffung von verteilten SoftwareentwicklerInnen

Angesichts der theoretischen Erkenntnisse wird folgendes Subprozessmodell „*Einführung von verteilten SoftwareentwicklerInnen*“ entwickelt:

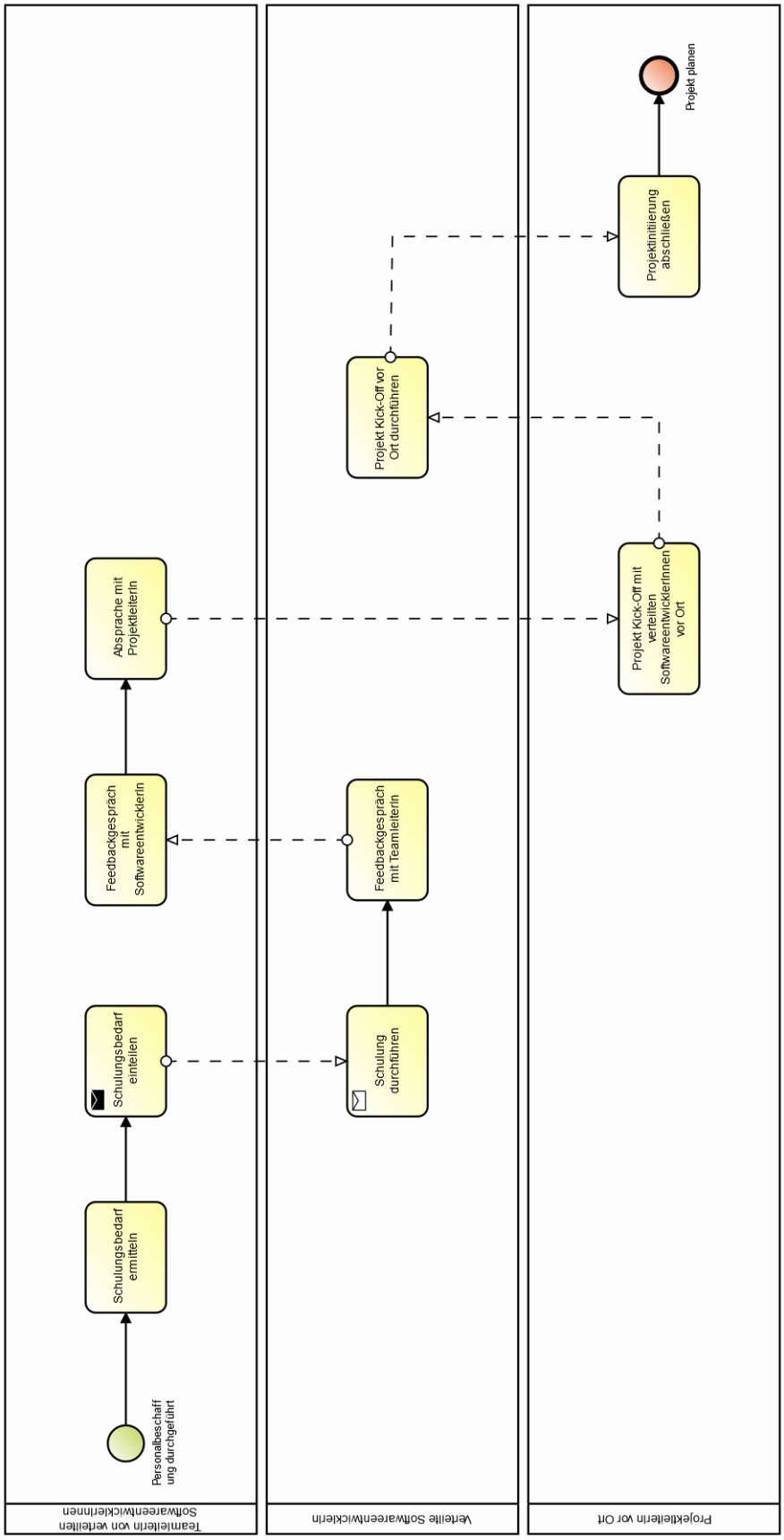

Abbildung 11 - P3: Einführung von verteilten SoftwareentwicklerInnen

3.2.3 Beschreibung des Prozessmodells

Das entwickelte Subprozessmodell *„Beschaffung von verteilten SoftwareentwicklerInnen“* wird im Folgenden durch die darin inkludierten Aktivitäten beschrieben.

Bedarf an verteilte SoftwareentwicklerInnen kommunizieren

ProjektleiterInnen kommunizieren bei der Initiierung von verteilten Softwareprojekten den Bedarf an Entwicklungsressourcen an die Teamleitung der verteilten SoftwareentwicklerInnen.

Interne Personalbeschaffung prüfen

Die Teamleitung der verteilten SoftwareentwicklerInnen prüft die Möglichkeit den Bedarf durch interne Personalbeschaffung zu decken. Dies kann einerseits durch interne Personalbewegungen, andererseits durch interne Ausschreibungen erfolgen (siehe Kapitel *2.5.1 Beschaffung von Personal*). Falls die Möglichkeit besteht, wird eine interne Personalbeschaffung durchgeführt und daraufhin die Personaleinführung in das verteilte Softwareprojekt umgesetzt.

Externe Personalbeschaffung anfordern

Falls eine interne Personalbeschaffung nicht möglich erscheint, können TeamleiterInnen der verteilten SoftwareentwicklerInnen eine externe Personalbeschaffung (siehe Kapitel *2.5.1 Beschaffung von Personal*) anfordern. Diese Anforderung wird von den TeamleiterInnen an die PersonalmanagerInnen gestellt.

Externe Personalbeschaffung initiieren und Anpassung der Personalplanung anfordern

PersonalmanagerInnen initiieren den externen Personalbeschaffungsprozess auf Anforderung der Teamleitung der verteilten SoftwareentwicklerInnen. Die Personalbeschaffung basiert auf der unternehmensweiten Personalplanung (siehe Kapitel *2.5.1 Beschaffung von Personal*) – daher muss die Anpassung der Personalplanung zuerst bei der Geschäftsführung angefordert werden.

Anpassung der Personalplanung prüfen

Die Geschäftsführung prüft die von der Personalabteilung durchgeführte Anpassung der Personalplanung. Bei Ablehnung durch die Geschäftsführung wird dies bis hin zur Projektleitung vor Ort kommuniziert. Schlussfolgernd wird die Personalbeschaffung abgebrochen. Bei Genehmigung der angepassten Personalplanung wird die externe Personalbeschaffung darauffolgend von PersonalmanagerInnen durchgeführt.

Externe Personalbeschaffung durchführen

Nach der Genehmigung durch die Geschäftsleitung führen PersonalmanagerInnen die externe Personalbeschaffung durch. Die externe Personalbeschaffung kann durch Inseratschaltungen in Internetportalen oder auch durch externe PersonaldienstleisterInnen abgewickelt werden (siehe Kapitel *2.5.1 Beschaffung von Personal*). In den weiterfolgenden Aktivitäten werden beide Arten der externen Personalbeschaffung parallel umgesetzt.

Offene Stelle in Internetportalen schalten

PersonalmanagerInnen inserieren die offene Stelle in Internetportalen. Die eingehenden Bewerbungen werden von der Personalabteilung konsolidiert und gemeinsam mit den Bewerbungen der externen PersonaldienstleisterInnen analysiert.

Externe/n PersonaldienstleisterIn kontaktieren

PersonalmanagerInnen kontaktieren den/die jeweilige/n PersonaldienstleisterIn hinsichtlich des Personalbedarfs. Je Region wird mit einer/m anderen PersonaldienstleisterIn kooperiert, da diese je geografischem Standort unterschiedlich mit verteilten SoftwareentwicklerInnen vernetzt sind (siehe Kapitel *2.5.1 Beschaffung* von Personal). Der/die extern/e PersonaldienstleisterIn initiiert den Rekrutierungsprozess und übermittelt potenzielle verteilte SoftwareentwicklerInnen an das Personalmanagement. Die Bewerbungen werden von der Personalabteilung konsolidiert und gemeinsam mit den Bewerbungen des Internetportals analysiert. Die Aktivitäten von externen PersonaldienstleisterInnen werden auch zusammenfassend als Supportprozess (siehe Kapitel *2.2.2 Geschäftsprozesse in Unternehmen*) wahrgenommen.

Bewerbungsgespräch durchführen und Besetzungsentscheidung treffen

PersonalmanagerInnen laden die potenziellen verteilten SoftwareneentwicklerInnen zu einem Bewerbungsgespräch ein. Die Teamleitung der verteilten SoftwareentwicklerIn sowie auch die Projektleitung des verteilten Softwareprojekts nehmen am Gespräch teil. TeamleiterInnen und ProjektleiterInnen treffen nach dem Bewerbungsgespräch gemeinsam eine Vorauswahl. Nach Rücksprache zwischen Teamleitung und Projektleitung wird eine Besetzungsentscheidung getroffen und an die PersonalmanagerInnen kommuniziert.

Information zu Angebotslegung versenden

Nach der Besetzungsentscheidung durch TeamleiterInnen und ProjektleiterInnen versenden PersonalmanagerInnen ein Angebot an den/die ausgewählte/n SoftwareentwicklerIn.

Dienstvertrag vorbereiten und unterzeichnen

PersonalmanagerInnen bereiten den Dienstvertrag für den/die neue/n MitarbeiterIn vor. Der Dienstvertrag wird im ersten Schritt der Geschäftsführung zur Unterzeichnung vorgelegt. In weiterer Folge wird der Dienstvertrag an den/die einzuführende/n SoftwareentwicklerIn versendet.

Anderen verteilten SoftwareentwicklerInnen absagen

Im letzten Schritt der Personalbeschaffung müssen PersonalmanagerInnen ordnungsgemäß den restlichen BewerberInnen eine Absage kommunizieren. Daraufhin wird von der Personalabteilung der nächste Prozess *„Einführung von verteilten SoftwareentwicklerInnen“* für den/die neue/n MitarbeiterIn angestoßen.

Das entwickelte Subprozessmodell *„Einführung von verteilten SoftwareentwicklerInnen“* wird im Folgenden durch die darin inkludierten Aktivitäten beschrieben.

Schulungsbedarf ermitteln und einteilen

TeamleiterInnen von verteilten SoftwareentwicklerInnen ermitteln nach der Personalbeschaffung den Schulungsbedarf von neuen MitarbeiterInnen. Diese Ermittlung dient zur fachlichen Einarbeitung (siehe Kapitel *2.5.2 Einführung von Personal*) von verteilten SoftwareentwicklerInnen. Infolge der Ermittlung des Schulungsbedarf teilen TeamleiterInnen die Schulungsblöcke ein. Die Schulung wird daraufhin von verteilten SoftwareentwicklerInnen durchgeführt.

Feedbackgespräch mit TeamleiterIn

Nach der fachlichen Einarbeitung (siehe Kapitel *2.5.2 Einführung von Personal*) der verteilten SoftwareentwicklerInnen wird mit einem Feedbackgespräch mit der Teamleitung fortgesetzt. Nach dieser Absprache erfolgt die Integration in das verteilte Softwareprojekt. Hierfür sprechen sich TeamleiterInnen mit den ProjektleiterInnen ab.

Projekt-Kick-off mit verteilten SoftwareentwicklerInnen vor Ort

Im nächsten Schritt wird vor Ort ein Projekt-Kick-off organisiert und durchgeführt (siehe Kapitel *3.1 Initiierung eines verteilten Softwareentwicklungsprojektes*). Zu diesem Zweck versammelt sich das gesamte verteilte Softwareprojektteam am gleichen geografischen Standort, um das Projekt initial zu besprechen. Ein derartiger „Vor Ort“-Termin fördert die soziale Integration (siehe Kapitel *2.5.2 Einführung von Personal*) in das Unternehmen langfristig.

Projektinitiierung abschließen

ProjektleiterInnen können die Projektinitiierung nach dem Projekt-Kick-off abschließen (siehe Kapitel *3.1 Initiierung eines verteilten Softwareentwicklungsprojektes*) und im nächsten Schritt die Planung des verteilten Softwareprojektes mit den TeamleiterInnen anstoßen (siehe Kapitel *3.3 Ressourcenplanung bei verteilten Softwareentwicklungsprojekten)*.

3.3 Ressourcenplanung bei verteilten Softwareentwicklungsprojekten

Dieses Kapitel beschreibt die Fragestellung *„Wie läuft die Ressourcenplanung bei verteilten SoftwareentwicklerInnen ab und wie sieht die Aufbau- sowie Ablauforganisation bei verteilten Softwareprojekten aus?“* und liefert dahingehend ein Prozessmodell als mögliche Antwort. Zusätzlich wird das Prozessmodell anhand der darin inkludierten Aktivitäten beschrieben und eine Verbindung zur vorhergehenden theoretischen Recherche hergestellt.

3.3.1 Beschreibung des Problems

Nach der Initiierung eines Projektes erfolgt die Projektplanung. Ein wesentlicher Bestandteil dieser Prozessgruppe ist die Ressourcenplanung. (siehe Kapitel *2.4.1 Projektmanagement*)

Vor der Ressourcenplanung muss die geplante Leistungsbeschreibung innerhalb des Projektes in kleinere Arbeitspakete strukturiert werden. Auf diese Weise können diese Arbeitspakete nach Aufwand geschätzt und somit ein Gesamtaufwand berechnet werden. Die Schätzungen je Arbeitspaket unterstützen ProjektmanagerInnen ebenso in Ressourcenplanungen – also der Zuteilung, wer in welchem Zeitrahmen, welches Arbeitspaket, durchführt. (siehe Kapitel *2.4.1 Projektmanagement*).

Neben der „ISO 21500"-Prozessgruppe hinsichtlich Planung (siehe Kapitel *2.4.1 Projektmanagement*) wird ergänzend folgende Literaturerkenntnis berücksichtigt:

> *„Im Mittelpunkt der prozessorientierten Unternehmensgestaltung steht die Ablauforganisation der Unternehmung. Während die Aufbauorganisation die Gliederung der Unternehmung in Teilsysteme (z. B. Abteilungen, Divisionen, Stellen) und die Zuordnung von Aufgaben zu diesen Teilsystemen beinhaltet, befasst sich die Ablauforganisation mit der Durchführung dieser Aufgaben sowie der Koordination der zeitlichen und räumlichen Aspekte der Aufgabendurchführung (wer macht was wann und womit). Elementare Bestandteile einer Aufgabe sind die Aktivitäten, welche die Grundbestandteile eines (Arbeits-)Prozesses bilden. Eine Aktivität bzw. eine Funktion ist ein Arbeitsschritt, der zur Erbringung einer Leistung durchgeführt werden muss."* (Becker, Kugeler, & Rosemann, 2005)

Die ergänzende Literaturrecherche bezüglich Aufbau- und Ablauforganisation kennzeichnet die notwendige Integration der ProjektmanagerInnen und verteilten SoftwareentwicklerInnen in Aufbauorganisationen. Wie schon erwähnt, findet in der Ressourcenplanung die Zuteilung statt, wer in welchem zeitlichen Rahmen, welches Arbeitspaket, übernimmt (siehe Kapitel *2.4.1 Projektmanagement*). Diese Zuteilung weist Parallelen zur Aufgabendurchführung der Ablauforganisation auf.

In der Aufbauorganisation werden ProjektmanagerInnen und verteilte SoftwareentwicklerInnen gleichermaßen in Organisationen einbezogen. Die Initiierungs-, Personalbeschaffungs- sowie Personaleinführungsprozesse (siehe Kapitel *3.1 Initiierung eines verteilten Softwareentwicklungsprojektes* und *3.2 Beschaffung und Einführung von verteilten SoftwareentwicklerInnen*) enthalten bereits unterschiedliche TeamleiterInnen, welche als Vorgesetzte von ProjektmanagerInnen und verteilten SoftwareentwicklerInnen dargestellt werden. Aufgrund dieser Tatsache wird in Bezug auf die Aufbauorganisation davon ausgegangen, dass ProjektleiterInnen und verteilte SoftwareentwicklerInnen in separate Organisationen gegliedert sind. Diese Organisationen besitzen jeweils eine/n andere/n TeamleiterIn und sind in langlebige Strukturen integriert.

Die Ablauforganisation wird hingegen durch die Aktivitäten eines Vorhabens, wie zum Beispiel eines verteilten Softwareprojektes, beschrieben und wiederholt ausgeführt. Die Zuteilung der Aktivitäten wird im Rahmen der Ressourcenplanung geplant. Die Aktivitäten werden anhand des Prozessmodells im nachfolgenden Subkapitel visualisiert.

3.3.2 Prozessmodell

Angesichts der theoretischen Erkenntnisse wird folgendes Subprozessmodell „*Ressourcenplanung bei verteilten Softwareentwicklungsprojekten*“ entwickelt:

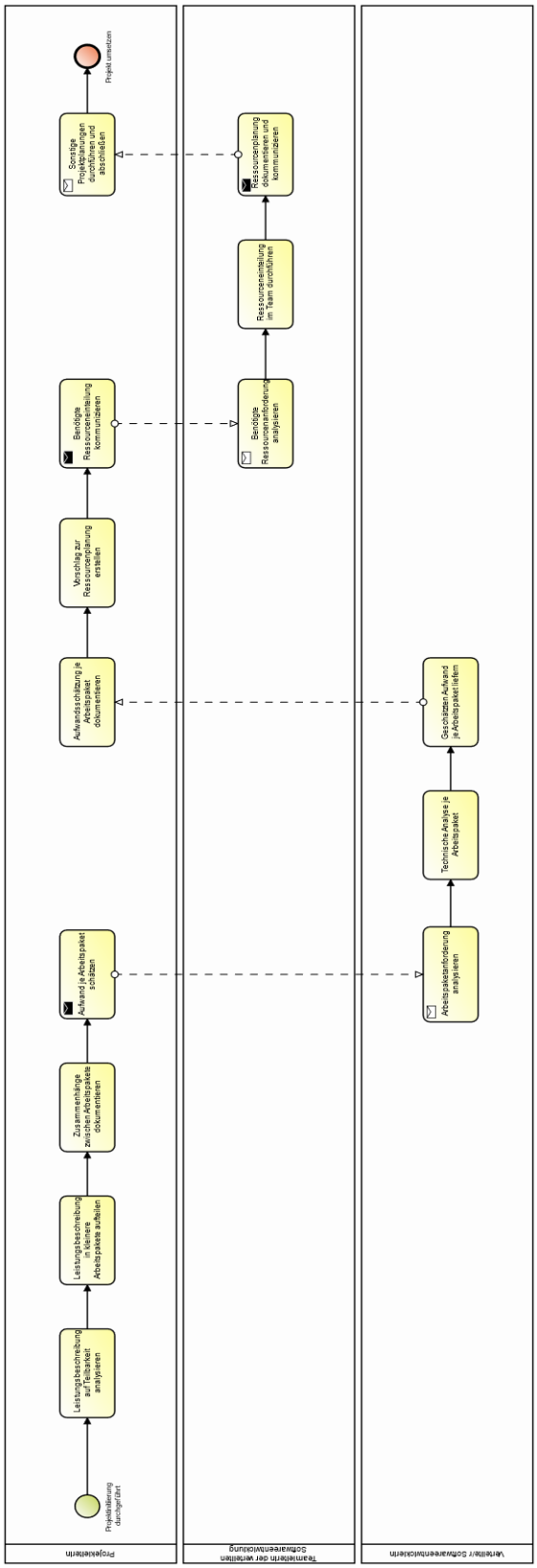

Abbildung 12 - P4: Ressourcenplanung bei verteilten Softwareentwicklungsprojekten

3.3.3 Beschreibung des Prozessmodells

Das entwickelte Subprozessmodell *„Ressourcenplanung bei verteilten Softwareentwicklungsprojekten“* wird im Folgenden durch die darin inkludierten Aktivitäten beschrieben.

Leistungsbeschreibung auf Teilbarkeit analysieren und in Arbeitspakete aufteilen

ProjektleiterInnen müssen nach der Projektinitiierung das Projekt planen (siehe Kapitel *2.4.1 Projektmanagement*). Als Basis hierfür verwenden ProjektleiterInnen die bereits dokumentierte Leistungsbeschreibung (siehe Kapitel *3.1 Initiierung eines verteilten Softwareentwicklungsprojektes*) und prüfen diese auf Teilbarkeit. Abhängig von der möglichen Teilbarkeit gliedern ProjektleiterInnen die Leistungsbeschreibung in mehrere Arbeitspakete für die verteilten SoftwareentwicklerInnen (siehe Kapitel *2.4.1 Projektmanagement*).

Zusammenhänge zwischen Arbeitspaketen dokumentieren

Nach der Definition der Arbeitspakete müssen ProjektleiterInnen die Zusammenhänge zwischen den Arbeitspaketen identifizieren (siehe Kapitel *2.4.1 Projektmanagement*). Auf diese Weise kann die sequenzielle Abarbeitung einzelner Pakete geplant sowie die Möglichkeiten zur parallelen Bearbeitung durch mehrere verteilte SoftwareentwicklerInnen berücksichtigt werden.

Aufwand je Arbeitspaket schätzen und technische Analyse

Für die Ressourcenplanung müssen die ProjektleiterInnen die einzelnen Arbeitspakete nach Aufwand schätzen (siehe Kapitel *2.4.1 Projektmanagement*). Die verteilten SoftwareentwicklerInnen unterstützen hierbei die Projektleitung mit ihrem technischen Wissen. Die Anforderungen innerhalb der Arbeitspakete werden von verteilten SoftwareentwicklerInnen auf Funktionalität geprüft sowie auch technisch analysiert.

Geschätzten Aufwand kommunizieren und dokumentieren

Verteilte SoftwareentwicklerInnen schätzen die Arbeitspakete in Projekttagen ab – ein Projekttag wird mit acht vollen Stunden kalkuliert. Die Schätzungen je Arbeitspaket werden dementsprechend an die Projektleitung kommuniziert und daraufhin durch die/den ProjektleiterIn dokumentiert.

Ressourcenplanung vorschlagen und benötigte Ressourceneinteilung kommunizieren

ProjektleiterInnen planen die Ressourcen auf Basis der dokumentierten Aufwände je Arbeitspaket (siehe Kapitel *2.4.1 Projektmanagement*). Die benötigte Ressourceneinteilung wird daraufhin der Teamleitung der verteilten SoftwareentwicklerInnen vorgeschlagen.

Ressourcenanforderung analysieren und Ressourceneinteilung im Team durchführen

TeamleiterInnen werden in der Aufbauorganisation als Vorgesetzte der verteilten SoftwareentwicklerInnen betrachtet (siehe Kapitel *3.3.1 Beschreibung des Problems*). Die Aktivitäten der verteilten SoftwareentwicklerInnen innerhalb eines Softwareprojektes werden in der Ablauforganisation beschrieben (siehe Kapitel *3.3.1 Beschreibung des Problems*). Für diese verteilten Softwareprojekte ist eine zeitliche Einteilung der verteilten SoftwareentwicklerInnen

erforderlich, welche in Form von Ressourcenplänen dokumentiert wird. Der vorgeschlagene Ressourcenplan der Projektleitung wird von dem/r TeamleiterIn der verteilten SoftwareentwicklerInnen genehmigt und wenn nötig, angepasst.

Ressourcenplanung kommunizieren und Projektplanung abschließen

TeamleiterInnen kommunizieren die finalen Ressourcenpläne an die ProjektleiterInnen. Darauffolgend gilt die Ressourcenplanung als abgeschlossen und die restlichen Planungen, wie z. B. Terminplanung, werden von den ProjektleiterInnen durchgeführt (siehe Kapitel *2.4.1 Projektmanagement*). Anschließend wird die Projektplanung als abgeschlossen angesehen und im nächsten Schritt die Umsetzung des verteilten Softwareprojekts initiiert.

3.4 Umsetzung und Controlling von verteilten Softwareprojekten

Dieses Kapitel beschreibt die Fragestellung *„Wie werden die Umsetzung und das Controlling bei Projekten mit verteilten SoftwareentwicklerInnen durchgeführt“* und liefert dahingehend ein Prozessmodell als mögliche Antwort. Zusätzlich wird das Prozessmodell anhand der darin inkludierten Aktivitäten beschrieben und eine Verbindung zur vorhergehenden theoretischen Recherche hergestellt.

3.4.1 Beschreibung des Problems

Nach der Projektplanung werden Prozesse hinsichtlich der Umsetzung und des Controllings angestoßen. Diese beiden Prozessgruppen verlaufen in Wechselwirkung untereinander. In der Projektumsetzung müssen die geplanten Arbeitspakete implementiert und abgeschlossen werden. Ein weiterer essenzieller Bestandteil innerhalb der Umsetzungsprozessgruppe ist das Koordinieren der verteilten SoftwareentwicklerInnen und die darauf basierende Zuweisung zu den verschiedenen Arbeitspaketen. Im Laufe der Umsetzung sammeln verteilte SoftwareentwicklerInnen Lernerfahrungen, welche wiederum in Wissensdatenbanken dokumentiert werden müssen und in späteren Prozessgruppen relevant werden. In der parallellaufenden Prozessgruppe „Controlling“ wird die Umsetzung regelmäßig von ProjektleiterInnen kontrolliert. Der Fokus liegt hierbei auf der aufgebrachten Leistung, den Ressourcen sowie der Termineinhaltung. Die Ziele hinter den regelmäßigen Kontrollen der genannten Teilbereiche sind zumal die Identifizierung des aktuellen Projektstatus und das rechtzeitige Erkennen eventueller Abweichungen. Auf Basis dieser Identifizierungen müssen ProjektleiterInnen vor Ort passende Maßnahmen einleiten. Die erkannten Abweichungen und eingeleiteten Maßnahmen müssen von ProjektleiterInnen in Statusberichten dokumentiert und kommuniziert werden. (siehe Kapitel *2.4.1 Projektmanagement*)

Für die Leistungskontrolle sind insbesondere die verbuchten Arbeitszeiten der verteilten SoftwareentwicklerInnen von Bedeutung, da mit diesen der aktuelle Projektstatus gemessen werden kann. (siehe Kapitel *2.6.2 Personalaufwandserfassung*)

Das fortdauernde Anforderungsmanagement ist ein fester Bestandteil der Projektumsetzung, da sich im Laufe des verteilten Softwareprojektes die KundInnenanforderungen ändern können und

ordentlich geführtes Anforderungsmanagement maßgeblich am Projekterfolg beteiligt ist (siehe Kapitel *2.4.2 Requirements Engineering*). Dies wird insbesondere bei Verrechnungsarten wie „Time & Materials“ ausschlaggebend (siehe Kapitel *2.3.3 Vertrags- und Abrechnungsmodelle*).

Die bereits erwähnte Sammlung an Lernerfahrungen bei verteilten SoftwareentwicklerInnen wird als Herausforderung identifiziert (siehe Kapitel *2.1.1 Herausforderungen der verteilten Softwareentwicklung*) und folglich in der Entwicklung des Prozessmodells stärker berücksichtigt. In der Prozessmodellentwicklung wird auf bestehende Erkenntnisse aus dem „Scrum“-Framework zurückgegriffen, um daraufhin die genannten Problematiken, wie z. B mangelnde Kommunikation, zu lösen (siehe Kapitel *2.4.3 Scrum und agile Softwareentwicklung*).

3.4.2 Prozessmodell

Angesichts der theoretischen Erkenntnisse wird folgendes Subprozessmodell „*Umsetzung von verteilten Softwareentwicklungsprojekten*“ entwickelt:

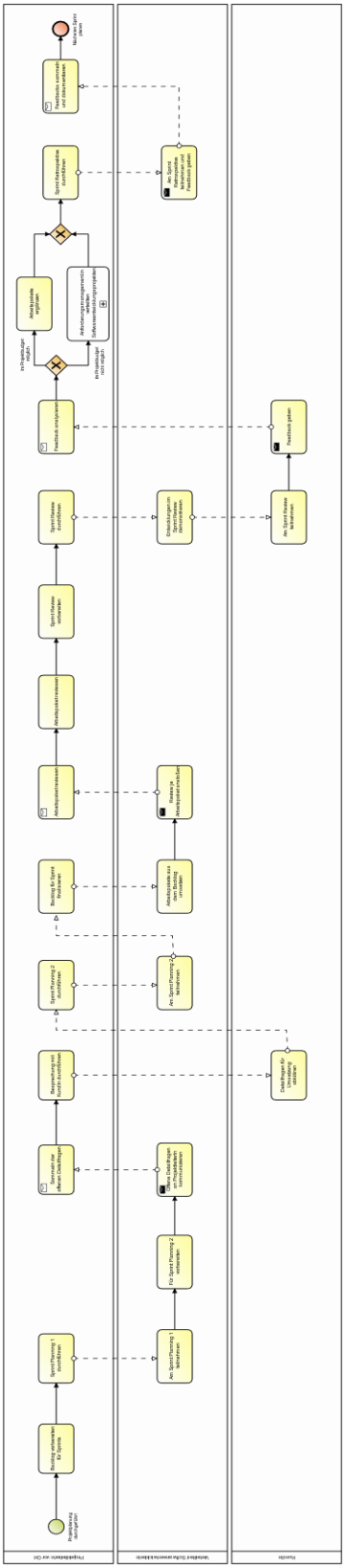

Abbildung 13 - P5: Umsetzung von verteilten Softwareentwicklungsprojekten

Angesichts der theoretischen Erkenntnisse wird folgendes Subprozessmodell „*Controlling von verteilten Softwareentwicklungsprojekten*“ entwickelt:

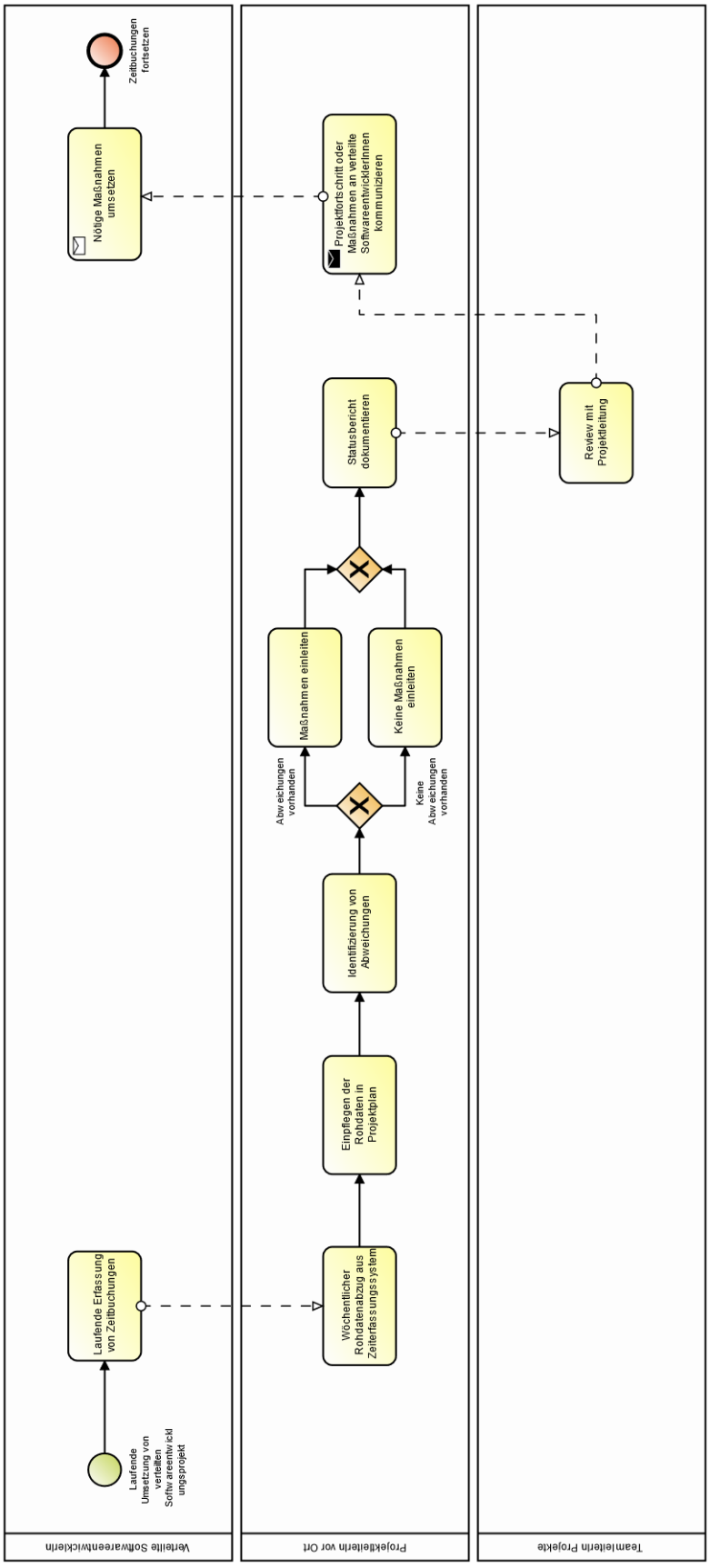

Abbildung 14 - P6: Controlling von verteilten Softwareentwicklungsprojekten

3.4.3 Beschreibung des Prozessmodells

Das entwickelte Subprozessmodell *„Umsetzung von verteilten Softwareentwicklungsprojekten“* wird im Folgenden durch die darin inkludierten Aktivitäten beschrieben.

Backlog für Sprints vorbereiten

Das Backlog ist eine Sammlung von User Stories, welche innerhalb des verteilten Softwareprojektes umgesetzt werden. Das Backlog wird vom verteilten Projektteam in sogenannten „Backlog Refinement“-Besprechungen aktualisiert. (siehe Kapitel *2.4.3 Scrum und agile Softwareentwicklung*)

Als Basis für das Backlog dienen hierbei die dokumentierte Leistungsbeschreibung aus der Projektinitiierung sowie Arbeitspakete aus der Projektplanung. (siehe Kapitel *2.4.1 Projektmanagement*).

Sprint Planning 1 durchführen

ProjektleiterInnen vor Ort planen grob die Arbeitspakete für den nächsten Sprint und kommunizieren dies in der „Sprint Planning 1“-Besprechung mit dem verteilten Softwareentwicklungsteams. (siehe Kapitel *2.4.3 Scrum und agile Softwareentwicklung*)

Für Sprint Planning 2 vorbereiten und offene Detailfragen mit der Kundin/dem Kunden abklären

Verteilte SoftwareentwicklerInnen analysieren nach Sprint Planning 1 die besprochenen User Stories im technischen Detail, um diese im Sprint Planning 2 genauer abschätzen zu können. Daraus ergeben sich offene Detailfragen hinsichtlich der KundInnenanforderungen (siehe Kapitel *2.4.2 Requirements Engineering*). Diese offenen Detailfragen werden von der Projektleitung gesammelt und in einem zusätzlichen KundInnengespräch geklärt.

Sprint Planning 2 durchführen und Sprint Backlog finalisieren

Die zusätzlichen Informationen der KundInnen werden nun im Sprint Planning 2 von der Projektleitung vor Ort und den verteilten SoftwareentwicklerInnen berücksichtigt. Aufgrund dessen ist der Großteil der Details geklärt und die User Stories können realistisch abgeschätzt werden. Auf Basis der Schätzungen kann das verteilte Projektteam den Umfang an User Stories für den nächsten Sprint Backlog realistisch eingrenzen. Das Backlog für den nächsten Sprint ist finalisiert und die Umsetzung des Sprints kann beginnen. (siehe Kapitel *2.4.3 Scrum und agile Softwareentwicklung*)

Arbeitspakete umsetzen und Reviews

ProjektleiterInnen müssen die Qualität der umgesetzten Arbeitspakete regelmäßig kontrollieren und prüfen, ob diese der dokumentierten Leistungsbeschreibung entsprechen (siehe Kapitel *2.4.1 Projektmanagement*). Hierfür wird eine „Review“-Besprechung zwischen ProjektleiterIn und verteiltem/r SoftwareentwicklerIn organisiert, wo zweitere/r die umgesetzten Arbeitspakete präsentiert. ProjektleiterInnen entscheiden in der Besprechung, ob Abweichungen vorhanden und Nachbesserungen nötig sind.

Sprint Review vorbereiten

Die umgesetzten Arbeitspakete müssen je Sprint in sogenannten „Sprint Review“-Besprechungen vor den KundInnen demonstriert werden (siehe Kapitel *2.4.3 Scrum und agile Softwareentwicklung*). Die Demonstrationen werden von verteilten SoftwareentwicklerInnen durchgeführt. Dennoch muss die Agenda je Sprint Review von der Projektleitung definiert und im Vorhinein an das verteilte Softwareentwicklungsteam kommuniziert werden. Auf diese Weise können die verteilten SoftwareentwicklerInnen im Voraus die zu demonstrierenden Entwicklungen auf dem entsprechenden System testen und relevante Testdaten organisieren.

Sprint Review durchführen und Feedback analysieren

Verteilte SoftwareentwicklerInnen demonstrieren am Ende des Sprints die Umsetzungen vor den KundInnen in „Sprint Review“-Besprechungen. Nach der Demonstration geben die KundInnen ihr Feedback zu den umgesetzten Arbeitspaketen hinsichtlich der erfüllten Erwartungen oder Abweichungen (siehe Kapitel *2.4.3 Scrum und agile Softwareentwicklung*). ProjektleiterInnen vor Ort analysieren die Feedbacks der KundInnen und gleichen ab, ob diese im Rahmen der Leistungsbeschreibung umgesetzt werden können. Ist eine Umsetzung möglich, werden die Arbeitspakete ergänzt und in das Backlog aufgenommen (siehe Kapitel *2.4.1 Projektmanagement*). Falls nicht, müssen die Zusatzanforderungen der KundInnen kommerziell separat betrachtet und via Anforderungsmanagement im Detail analysiert werden (siehe Kapitel *2.4.2 Requirements Engineering* und *3.5 Anforderungsmanagement in verteilten Softwareentwicklungsprojekten*).

Sprint Retrospektive und interne Feedbacks dokumentieren

Alle Mitglieder des verteilten Projektteams nehmen am Ende des Sprints an einer internen „Sprint Retrospektive“-Besprechung teil (siehe Kapitel *2.4.3 Scrum und agile Softwareentwicklung*). In dieser Besprechung listen alle Projektmitglieder die individuellen Pros und Kontras auf, welche im Laufe des aktuellen Sprints aufgefallen sind. Hierzu gehören auch Lernerfahrungen, welche in dieser Besprechung an das gesamte verteilte Projektteam kommuniziert werden sollen. Diese Lernerfahrungen müssen von ProjektleiterInnen in einer Wissensdatenbank gesammelt werden, um in zukünftigen Projekten auf bestehendes Wissen zugreifen zu können (siehe Kapitel *2.1.1 Herausforderungen der verteilten Softwareentwicklung* sowie *3.6 Lernerfahrungen mit verteilten SoftwareentwicklerInnen bei Projektabschluss*). Die erwähnten Kontras müssen in einem eigenen Speicher dokumentiert werden, damit in weiterer Folge Maßnahmen konzipiert werden können.

Nächsten Sprint planen

Der nächste Sprint wird analog zum bereits beschriebenen Vorgehen geplant. Das realisierbare KundInnenfeedback wird hierbei bereits im Backlog Sprint berücksichtigt.

Das entwickelte Subprozessmodell *„Controlling von verteilten Softwareentwicklungsprojekten“* wird im Folgenden durch die darin inkludierten Aktivitäten beschrieben.

Laufende Erfassung von Zeitbuchungen

Verteilte SoftwareentwicklerInnen müssen neben der Umsetzung ihre Zeitbuchungen je Arbeitspaket erfassen (siehe Kapitel *2.6.2 Personalaufwandserfassung*). Das Zeiterfassungssystem wird ebenso von ProjektleiterInnen vor Ort genutzt und dient zusätzlich als Projektmanagementwerkzeug.

Wöchentlicher Rohdatenabzug aus Zeiterfassungssystem und Pflege in Projektplan

ProjektleiterInnen vor Ort nutzen die Zeiterfassungen als relevante Information für das Projektcontrolling (siehe Kapitel *2.4.1 Projektmanagement*). Hierfür wird wöchentlich ein Rohdatenabzug aus dem Zeiterfassungssystem durchgeführt und im Projektplan aus der Prozessgruppe „Planung“ eingepflegt. Auf diese Weise ist ein Soll-/Ist-Vergleich möglich.

Identifizierung von Abweichungen und Maßnahmen

Anhand der verbuchten Arbeitsaufwände der verteilten SoftwareentwicklerInnen können ProjektleiterInnen vor Ort die Abweichungen im Vergleich zum Projektplan identifizieren. Daraus ergibt sich der Fortschrittsgrad im verteilten Softwareprojekt. Auf Basis dessen entscheiden ProjektleiterInnen vor Ort, ob Maßnahmen gesetzt werden müssen. (siehe Kapitel *2.4.1 Projektmanagement*).

Statusbericht dokumentieren und Review mit TeamleiterIn der Projektabwicklung

Unabhängig davon, ob Maßnahmen gesetzt werden, muss ein aktueller Statusbericht zum verteilten Softwareprojekt erstellt werden. Dieser Statusbericht dient dazu den höheren Führungsebenen einen aktuellen Einblick in das verteilte Softwareprojekt zu verschaffen (siehe Kapitel *2.4.1 Projektmanagement*). Hierfür wird ein wöchentliches „Review“-Meeting zwischen ProjektleiterInnen und TeamleiterInnen vor Ort durchgeführt. TeamleiterInnen bewerten in der Besprechung, ob ProjektleiterInnen die Projektsituation und den Fortschritt richtig einschätzen und organisieren gegebenenfalls benötigte Unterstützung.

Projektfortschritt kommunizieren und Maßnahmen umsetzen

Die ProjektleiterInnen müssen den verteilten SoftwareentwicklerInnen die Projektsituation sowie den Fortschritt transparent darstellen (siehe Kapitel *2.1.1 Herausforderungen der verteilten Softwareentwicklung*). Auf diese Weise ist dem gesamten verteilten Projektteam die aktuelle Situation klar ersichtlich und alle Beteiligten werden auf nötige Aktionen sensibilisiert – unter anderem auch Maßnahmen zur Beseitigung von Abweichungen (siehe Kapitel *2.4.1 Projektmanagement*).

3.5 Anforderungsmanagement in verteilten Softwareentwicklungsprojekten

Dieses Kapitel beschreibt die Fragestellung *„Wie können verteilte SoftwareentwicklerInnen ein besseres Verständnis für die KundInnen- und Softwareanforderungen im Projekt erlangen?“* und liefert dahingehend ein Prozessmodell als mögliche Antwort. Zusätzlich wird das Prozessmodell anhand der darin inkludierten Aktivitäten beschrieben und eine Verbindung zur vorhergehenden theoretischen Recherche hergestellt.

3.5.1 Beschreibung des Problems

Wie schon im Kapitel *3.4 Umsetzung und Controlling von verteilten Softwareprojekten* beschrieben, können während der Umsetzung des verteilten Softwareprojektes zusätzliche Softwareanforderungen von KundInnen gewünscht werden. ProjektleiterInnen vor Ort obliegt die Entscheidung, ob sich diese Zusatzanforderungen im Rahmen des Projektbudgets befinden oder diese Anforderungen separat bewertet werden müssen.

Das verteilte Softwareprojektteam muss flexibel genug arbeiten, damit derartige Zusatzanforderungen von KundInnen organisatorisch kein Problem darstellen. Dennoch muss das Anforderungsmanagement in derartigen Fällen strukturiert vorgehen. Hierfür bietet das Anforderungsmanagement folgende vier Teilprozesse (siehe Kapitel *2.4.2 Requirements Engineering*):

- Ermitteln der Anforderungen von KundInnen
- Vermitteln der Anforderungen an verteiltes Softwareentwicklungsteam
- Prüfen der Anforderungen (technisch, fachspezifisch und kommerziell)
- Verwalten der Anforderungen

Bei der Ermittlung der Anforderungen wird eine ergänzende Besprechung zwischen ProjektleiterInnen und KundInnen organisiert. Die ermittelten Anforderungen müssen dem verteilten Softwareentwicklungsteam entsprechend vermittelt und von den jeweiligen Personen verstanden werden (siehe Kapitel *2.1.1 Herausforderungen der verteilten Softwareentwicklung*). Diese Softwareanforderungen müssen im verteilten Projektteam geprüft und bewertet werden, um im Anschluss im Backlog verwaltet werden zu können. (siehe Kapitel *2.4.2 Requirements Engineering*)

3.5.2 Prozessmodell

Angesichts der theoretischen Erkenntnisse wird folgendes Subprozessmodell „*Anforderungsmanagement in verteilten Softwareentwicklungsprojekten*“ entwickelt:

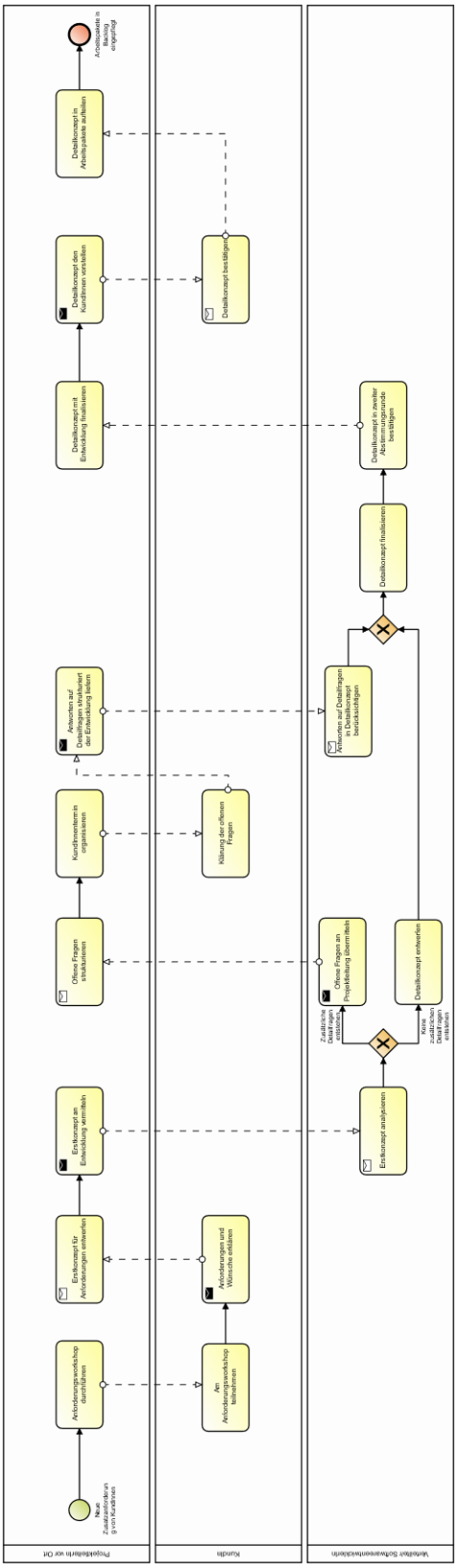

Abbildung 15 - P7: Anforderungsmanagement in verteilten Softwareentwicklungsprojekten

3.5.3 Beschreibung des Prozessmodells

Das entwickelte Subprozessmodell *„Anforderungsmanagement in verteilten Softwareentwicklungsprojekten“* wird im Folgenden durch die darin inkludierten Aktivitäten beschrieben.

Anforderungsworkshop durchführen

Bei zusätzlichen Softwareanforderungen von KundInnen, welche nicht im Rahmen des Projektbudgets abgedeckt werden können, müssen ProjektleiterInnen und KundInnen einen separaten Anforderungsworkshop durchführen. In diesem Anforderungsworkshop müssen KundInnen ihre Wünsche und Anforderungen beschreiben. In den vier Phasen des agilen Anforderungsmanagements wird dieser Schritt als „Ermitteln der Anforderungen“ beschrieben. (siehe Kapitel *2.4.2 Requirements Engineering*)

Erstkonzept entwerfen und an Entwicklung vermitteln

Die ermittelten Anforderungen und Wünsche der KundInnen müssen von ProjektleiterInnen vor Ort in einem Erstkonzept dokumentiert werden. ProjektleiterInnen berücksichtigen in diesem Schritt bereits die fachlichen Gegebenheiten und Umstände, ohne dabei tiefe technische Themen zu analysieren. Dieses Erstkonzept wird in einer zweistufigen Kommunikation an verteilte SoftwareentwicklerInnen vermittelt (siehe Kapitel *2.1.1 Herausforderungen der verteilten Softwareentwicklung*). In der ersten Besprechung muss der fachliche Hintergrund der KundInnenanforderungen an die verteilte Softwareentwicklung übermittelt werden. Darauffolgend wird das fachliche Erstkonzept von der Entwicklung aus technischer Perspektive analysiert. Im agilen Anforderungsmanagement wird die Vorstellung des Erstkonzepts als „Vermittlung von Anforderungen“ beschrieben (siehe Kapitel *2.4.2 Requirements Engineering*).

Erstkonzept analysieren und Detailkonzept entwerfen

Verteilte SoftwareentwicklerInnen prüfen das vermittelte Erstkonzept auf die technische Realisierbarkeit. Während der Analyse und Prüfung des Erstkonzepts können zusätzliche Detailfragen entstehen, welche in weiterer Folge durch ProjektleiterInnen vor Ort oder KundInnen beantwortet werden müssen. Falls keine offenen Detailfragen entstehen, können verteilte SoftwareentwicklerInnen das Detailkonzept weiterentwerfen und aus technischer Perspektive finalisieren. Die Prüfung des vermittelten Erstkonzepts wird im agilen Anforderungsmanagement als „Prüfung der Anforderungen“ beschrieben. (siehe Kapitel *2.4.2 Requirements Engineering*)

Offene Fragen für Detailkonzept

Verteilte SoftwareentwicklerInnen kommunizieren offene Fragen hinsichtlich des Detailkonzepts an die Projektleitung vor Ort. Daraufhin müssen ProjektleiterInnen vor Ort die offenen Fragen strukturieren, um diese entsprechend für ein KundInnengespräch aufzubereiten. Im nächsten Schritt organisiert die Projektleitung einen Besprechungstermin mit den KundInnen, in welchem die offenen Fragen gemeinsam geklärt werden müssen. Diese Abklärungen müssen immer im Einklang mit den fachlichen Gegebenheiten stehen. Die Antworten werden von ProjektleiterInnen vor Ort nachbearbeitet und in einer aufbereiteten sowie strukturierten Form an die verteilte Softwareentwicklung geliefert (siehe Kapitel *2.1.1 Herausforderungen der verteilten*

Softwareentwicklung). Die Antworten auf die entstandenen Detailfragen müssen im Detailkonzept berücksichtigt werden. Dadurch wird das Detailkonzept aus technischer Perspektive finalisiert. Die Klärung von offenen Detailfragen wird im agilen Anforderungsmanagement ebenso dem Teilprozess „Prüfung von Anforderungen“ zugeordnet. (siehe Kapitel *2.4.2 Requirements Engineering*)

Detailkonzept abstimmen und bestätigen

Verteilte SoftwareentwicklerInnen und ProjektleiterInnen vor Ort müssen das finalisierte Detailkonzept gemeinsam abstimmen – dazu zählt auch die Abschätzung des Entwicklungsaufwandes (siehe Kapitel *2.4.1 Projektmanagement*). Diese Abstimmung wird als zweite Besprechung im zweistufigen Kommunikationsprozess hinsichtlich der Anforderungsanalyse beschrieben. Nach der internen Abstimmung stellt die Projektleitung den KundInnen das Detailkonzept vor und kommuniziert ebenso den Preis für den Lösungsvorschlag auf Basis der Entwicklungsabschätzung. In weiterer Folge müssen KundInnen das Detailkonzept inklusive Preis im Rahmen eines Angebots bestätigen und bestellen (siehe Kapitel *2.3.3 Vertrags- und Abrechnungsmodelle*).

Detailkonzept in Arbeitspakete aufteilen

Nach der Bestätigung und Bestellung des Detailkonzepts durch KundInnen muss dieses in Arbeitspakete aufgeteilt werden. ProjektleiterInnen vor Ort transformieren diese Arbeitspakete in User Stories und pflegen diese abhängig der Priorität in das Backlog ein (siehe Kapitel *2.4.3 Scrum und agile Softwareentwicklung*). Im weiteren Verlauf werden diese User Stories in Sprint Plannings eingeplant sowie im Rahmen von Sprints umgesetzt (siehe Kapitel *3.4 Umsetzung und Controlling von verteilten Softwareprojekten*). Das Einpflegen der Anforderungen in das Backlog wird im agilen Anforderungsmanagement als „Verwaltung der Anforderungen“ beschrieben (siehe Kapitel *2.4.2 Requirements Engineering*).

3.6 Lernerfahrungen mit verteilten SoftwareentwicklerInnen bei Projektabschluss

Dieses Kapitel beschreibt die Fragestellung *„Wie kann man bei Abschluss eines verteilten Softwareentwicklungsteams die Lernerfahrungen mit verteilten SoftwareentwicklerInnen identifizieren und verwalten?“* und liefert dahingehend ein Prozessmodell als mögliche Antwort. Zusätzlich wird das Prozessmodell anhand der darin inkludierten Aktivitäten beschrieben und eine Verbindung zur vorhergehenden theoretischen Recherche hergestellt.

3.6.1 Beschreibung des Problems

In der letzten Prozessgruppe „Abschluss“ müssen die gesammelten Lernerfahrungen des verteilten Softwareprojektteams vollständig dokumentiert und in einer Wissensdatenbank für zukünftige MitarbeiterInnen und Projekte gesichert werden. Der Zweck ist, aus diesem verteilten Softwareprojekt und anderen Projekten für die Linearorganisation zu lernen. Die Erfassung von

Lernerfahrung kann in allen Projektmanagementprozessen erfolgen (siehe Kapitel *3.4 Umsetzung und Controlling von verteilten Softwareprojekten*), jedoch ist es wichtig, alle Erfahrungen spätestens zum Projektabschluss zu erfassen und entsprechend bereitzustellen. (siehe Kapitel *2.4.1 Projektmanagement*)

Das Wissensmanagement stellt in der verteilten Softwareentwicklung eine generelle Herausforderung dar. Die Erfahrungen, Methoden, Entscheidungen und Fähigkeiten der Teammitglieder müssen während des Entwicklungsprozesses durch einen effektiven Informationsaustausch gesammelt werden, damit Teammitglieder die Erfahrungen ihrer VorgängerInnen nutzen können. Dies führt folglich zur Reduktion von Kosten sowie überflüssiger Arbeit. Ohne effektive Mechanismen zum Informations- und Wissensaustausch können ManagerInnen die Vorteile der verteilten Softwareentwicklung nicht nutzen. Der Wissensaustausch wird erleichtert, indem ein Produkt- oder Prozess-Repository verwaltet wird – dieses beinhaltet verschiedene Quellen, wie z. B. E-Mail, Online-Diskussionen etc. (siehe Kapitel *2.1.1 Herausforderungen der verteilten Softwareentwicklung*)

3.6.2 Prozessmodell

Angesichts der theoretischen Erkenntnisse wird folgendes Subprozessmodell „*Lernerfahrungen mit verteilten SoftwareentwicklerInnen bei Projektabschluss*“ entwickelt:

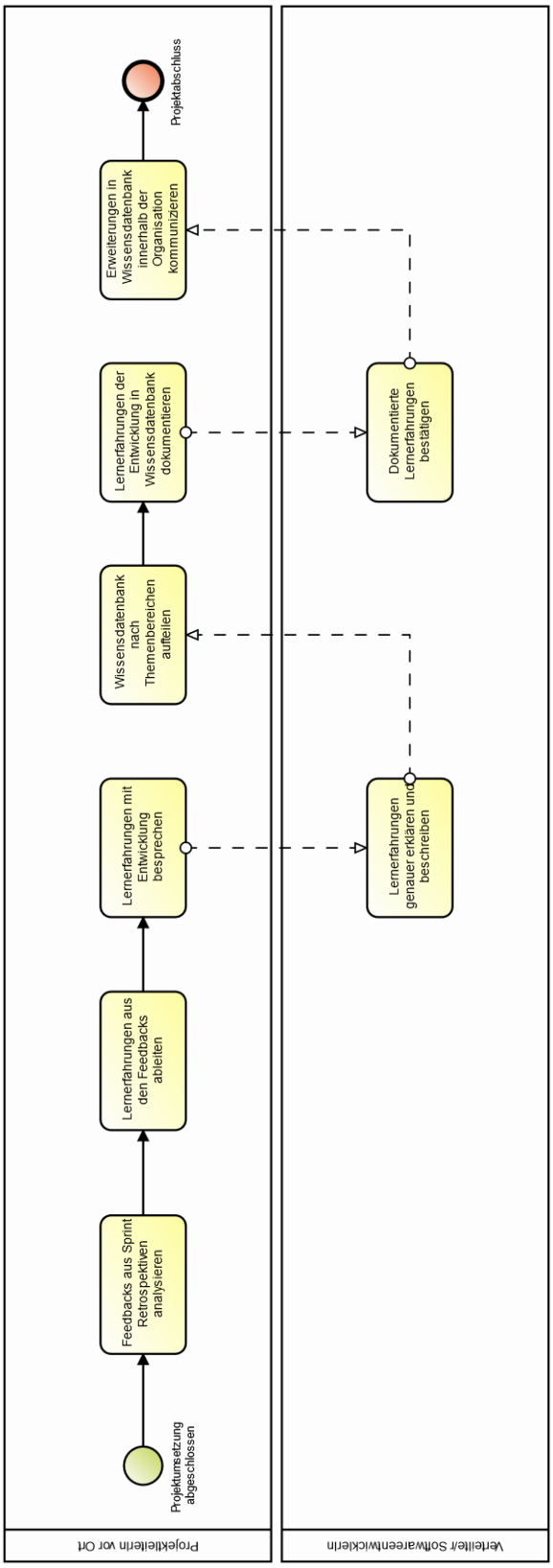

Abbildung 16 - P8: Lernerfahrungen mit verteilten SoftwareentwicklerInnen bei Projektabschluss

3.6.3 Beschreibung des Prozessmodells

Das entwickelte Subprozessmodell *„Lernerfahrungen mit verteilten SoftwareentwicklerInnen bei Projektabschluss“* wird im Folgenden durch die darin inkludierten Aktivitäten beschrieben.

Feedbacks aus Sprint Retrospektiven analysieren

Während der Projektumsetzung (siehe Kapitel *3.4 Umsetzung und Controlling von verteilten Softwareprojekten*) sammeln ProjektleiterInnen vor Ort und verteilte SoftwareentwicklerInnen gemeinsam Feedbacks im Zuge der Sprint Retrospektiven (siehe Kapitel *2.4.3 Scrum und agile Softwareentwicklung*). ProjektleiterInnen vor Ort müssen diese Feedbacks nach der abgeschlossenen Projektumsetzung analysieren (siehe Kapitel *2.4.1 Projektmanagement*).

Lernerfahrungen aus den Feedbacks ableiten und mit Entwicklung besprechen

ProjektleiterInnen vor Ort können mit der Analyse der Feedbacks die Lernerfahrungen von Mitglieder des verteilten Softwareprojektes ableiten. Diese abgeleiteten Lernerfahrungen müssen in einer separaten Besprechung von den verteilten SoftwareentwicklerInnen im Detail erklärt und beschrieben werden. (siehe Kapitel *2.1.1 Herausforderungen der verteilten Softwareentwicklung*)

Lernerfahrungen in Wissensdatenbank dokumentieren

ProjektleiterInnen vor Ort müssen die von den verteilten SoftwareentwicklerInnen beschriebenen Lernerfahrungen in einer Wissensdatenbank pflegen. Hierfür muss die Projektleitung im Vorhinein eine geeignete Strukturierung hinsichtlich der Themenbereiche festlegen, um die Lernerfahrungen entsprechend zuordnen zu können. Die dokumentierten Lernerfahrungen in der Wissensdatenbank müssen im Anschluss von verteilten SoftwareentwicklerInnen bestätigt und gegebenenfalls ergänzt werden. (siehe Kapitel *2.1.1 Herausforderungen der verteilten Softwareentwicklung*)

Erweiterungen in Wissensdatenbank innerhalb der Organisation kommunizieren

Der Hauptgedanke hinter einer Wissensdatenbank liegt in der Erfahrungsnutzung von VorgängerInnen. Verteilte SoftwareentwicklerInnen sollen aus dem Wissen von Vorprojekten profitieren und somit Kosten und unnötigen Arbeitsaufwand sparen (siehe Kapitel *2.1.1 Herausforderungen der verteilten Softwareentwicklung*). Aus diesem Grund muss die Wissensdatenbank für eine gesamte Organisation, wie zum Beispiel eine Abteilung oder ein Unternehmen, zur Verfügung stehen. Darüber hinaus müssen ProjektleiterInnen die Erweiterungen in der Wissensdatenbank innerhalb der Organisation kommunizieren (siehe Kapitel *2.1.1 Herausforderungen der verteilten Softwareentwicklung*). Nach der erfolgten Kommunikation der ergänzten Lernerfahrungen können ProjektleiterInnen das verteilte Softwareprojekt im weiteren Verlauf abschließen (siehe Kapitel *2.4.1 Projektmanagement*).

4 VALIDIERUNG DES PROZESSMODELLS

Auf Basis des entwickelten Prozessmodells im Kapitel *3 Entwicklung des Prozessmodells* werden mehrere Hypothesen aufgestellt. Diese Hypothesen werden im weiteren Verlauf durch eine qualitative Inhaltsanalyse untersucht und daraus folgend die Antwort zur Forschungsfrage abgeleitet.

4.1 Hypothesenbildung

Anhand des entwickelten Prozessmodells (siehe Kapitel *3 Entwicklung des Prozessmodells*) werden mehrere Hypothesen aufgestellt. Je „ISO 21500"-Prozessgruppe (siehe Kapitel *2.4.1 Projektmanagement*) wird eine Hypothese aufgestellt, um auf diese Weise den gesamtheitlichen Projektverlauf abzudecken. Die Relevanz je Hypothese wird ebenso beschrieben, um dahingehend den Bezug zur Forschungsfrage aufzubauen.

a. Bei der Beschaffung von verteilten SoftwareentwicklerInnen sollen externe PersonaldienstleisterInnen hinsichtlich spezifischen Know-hows engagiert werden.

Die Einführung einer verteilten Softwareentwicklung für die Projektabwicklung erfordert folgende Prozesse:

- Initiierung eines verteilten Softwareentwicklungsprojektes (siehe Kapitel *3.1 Initiierung eines verteilten Softwareentwicklungsprojektes*)
- Beschaffung von verteilten SoftwareentwicklerInnen (siehe Kapitel *3.2 Beschaffung und Einführung von verteilten SoftwareentwicklerInnen*)
- Einführung von verteilten SoftwareentwicklerInnen (siehe Kapitel *3.2 Beschaffung und Einführung von verteilten SoftwareentwicklerInnen*)

Insbesondere der Prozess „*Beschaffung von verteilten SoftwareentwicklerInnen*" erfordert in der aktiven Personalbeschaffung die Beauftragung von externen PersonaldienstleisterInnen. Diese nutzen ihr spezielles Fachwissen, um die benötigten verteilten SoftwareentwicklerInnen zu finden. Die Unterstützung durch externe PersonaldienstleisterInnen wird als Supportprozess (siehe Kapitel *2.2.2 Geschäftsprozesse in Unternehmen*) wahrgenommen. Je Region wird mit einer/m anderen PersonaldienstleisterIn kooperiert, da diese je geografischem Standort unterschiedlich mit verteilten SoftwareentwicklerInnen vernetzt sind. (siehe Kapitel *2.5.1 Beschaffung* von Personal)

b. In der Aufbauorganisation sollen die Vorgesetzten der verteilten SoftwareentwicklerInnen am gleichen geografischen Standort arbeiten und mit diesen muss die Ressourcenplanung koordiniert werden.

Die Einführung und Steuerung einer verteilten Softwareentwicklung für die Projektabwicklung erfordert folgenden Prozess:

- Ressourcenplanung bei verteilten Softwareentwicklungsprojekten (siehe Kapitel *3.3 Ressourcenplanung bei verteilten Softwareentwicklungsprojekten*)

Auf Grundlage von Leistungs- und Terminplanungen kann die Ressourcenplanung erfolgen. Beim Schätzen des Umfangs der notwendigen Ressourcen wird für jedes Arbeitspaket der jeweilige Bedarf an Ressourcen (Personen, Infrastruktur etc.) analysiert. Zu diesem Zweck ist es hilfreich, Schätzungen aus bereits spezifizierten Arbeitspaketen heranzuziehen und diese für die Ressourcenplanung zu nutzen. (siehe Kapitel *2.4.1 Projektmanagement*)

In der definierten Aufbauorganisation arbeiten die Vorgesetzten der verteilten SoftwareentwicklerInnen am gleichen geografischen Standort, wie die verteilten SoftwareentwicklerInnen selbst. Auf diese Weise erhalten ProjektleiterInnen vor Ort eine zentrale Ansprechperson für die Ressourcenplanung der verteilten SoftwareentwicklerInnen, wodurch mögliche Synchronisierungsprobleme zwischen ProjektleiterInnen reduziert werden. (siehe Kapitel *3.3 Ressourcenplanung bei verteilten Softwareentwicklungsprojekten*)

c. Scrum bietet Vorteile für die Projektumsetzung mit verteilten SoftwareentwicklerInnen, da hierfür hoher Kommunikationsaufwand betrieben wird.

Die Steuerung einer verteilten Softwareentwicklung für die Projektabwicklung erfordert folgende Prozesse:

- Umsetzung von verteilten Softwareentwicklungsprojekten (siehe Kapitel *3.4 Umsetzung und Controlling von verteilten Softwareprojekten*)
- Anforderungsmanagement in verteilten Softwareentwicklungsprojekten (siehe Kapitel *3.5 Anforderungsmanagement in verteilten Softwareentwicklungsprojekten*)

In der Projektumsetzung müssen geplante Arbeitspakete implementiert und abgeschlossen werden. Ein weiterer essenzieller Bestandteil innerhalb der Umsetzungsprozessgruppe ist das Koordinieren der verteilten SoftwareentwicklerInnen und die darauf basierende Zuweisung zu den verschiedenen Arbeitspaketen. (siehe Kapitel *2.4.1 Projektmanagement*)

Die Umsetzung von verteilten Softwareentwicklungsprojekten basiert in dieser Arbeit auf „Scrum“ (siehe Kapitel *2.4.3 Scrum und agile Softwareentwicklung*). Diese Art der Projektumsetzung fordert durch die hohe Anzahl an Abstimmungen einen hohen Kommunikationsaufwand. Dies wird als Vorteil für die verteilte Softwareprojektabwicklung wahrgenommen, da eine zu geringe Kommunikation den Erfolg des Projektes gefährden kann (siehe Kapitel *2.1.1 Herausforderungen der verteilten Softwareentwicklung*).

In der Abwicklung von verteilten Softwareprojekten muss ebenso das agile Anforderungsmanagement berücksichtigt werden. Während der Umsetzung des verteilten Softwareprojektes können nämlich zusätzliche Softwareanforderungen von KundInnen gewünscht werden. (siehe Kapitel *2.4.2 Requirements Engineering*)

Bei der Ermittlung der Anforderungen wird eine ergänzende Besprechung zwischen ProjektleiterInnen und KundInnen organisiert. Die ermittelten Anforderungen müssen dem verteilten Softwareentwicklungsteam entsprechend vermittelt und von den jeweiligen Personen verstanden werden (siehe Kapitel *2.1.1 Herausforderungen der verteilten Softwareentwicklung*). Diese Softwareanforderungen müssen im verteilten Projektteam geprüft und bewertet werden, um im Anschluss im Backlog verwaltet werden zu können. (siehe Kapitel *2.4.2 Requirements Engineering*)

Das frühzeitige Erkennen von neuen KundInnenanforderungen wird durch den hohen Kommunikationsaufwand in Scrum erreicht. Den verteilten SoftwareentwicklerInnen werden die neuen Anforderungen dadurch frühzeitig und durch eine zweistufige Kommunikation verständlich vermittelt. (siehe Kapitel *3.5 Anforderungsmanagement in verteilten Softwareentwicklungsprojekten*)

d. ProjektleiterInnen sollen den verteilten SoftwareentwicklerInnen wöchentlich den Soll-/Ist-Vergleich im verteilten Softwareprojekt darlegen, um daraus folgend die aktuelle Projektsituation und notwendige Maßnahmen besser vermitteln zu können.

Die Steuerung einer verteilten Softwareentwicklung für die Projektabwicklung erfordert folgenden Prozess:

- Controlling von verteilten Softwareentwicklungsprojekten (siehe Kapitel *3.4 Umsetzung und Controlling von verteilten Softwareprojekten*)

In der parallellaufenden Prozessgruppe „Controlling“ wird die Umsetzung regelmäßig von ProjektleiterInnen kontrolliert. Der Fokus liegt hierbei auf der aufgebrachten Leistung, den Ressourcen sowie der Termineinhaltung. Die Ziele hinter den regelmäßigen Kontrollen der genannten Teilbereiche sind zumal die Identifizierung des aktuellen Projektstatus und das rechtzeitige Erkennen eventueller Abweichungen. Auf Basis dieser Identifizierungen müssen ProjektleiterInnen vor Ort passende Maßnahmen einleiten. Die erkannten Abweichungen und eingeleiteten Maßnahmen müssen von ProjektleiterInnen in Statusberichten dokumentiert und kommuniziert werden. (siehe Kapitel *2.4.1 Projektmanagement*)

Für die Leistungskontrolle sind insbesondere die verbuchten Arbeitszeiten der verteilten SoftwareentwicklerInnen von Bedeutung, da mit diesen der aktuelle Projektstatus gemessen werden kann. (siehe Kapitel *2.6.2 Personalaufwandserfassung*)

Im Prozess „Controlling von verteilten Softwareentwicklungsprojekten“ werden die Kennzahlen wöchentlich aktualisiert und auch an das verteilte Softwareentwicklungsteam kommuniziert. Dadurch soll die Notwendigkeit von neuen Maßnahmen effizienter vermittelt werden. (siehe Kapitel *3.4 Umsetzung und Controlling von verteilten Softwareprojekten*)

e. Lernerfahrungen von verteilten SoftwareentwicklerInnen sollen in einer organisationszentralen Wissensdatenbank dokumentiert werden und verteilte SoftwareentwicklerInnen sollen dabei aktiv mitwirken.

Die Steuerung einer verteilten Softwareentwicklung für die Projektabwicklung erfordert folgenden Prozess:

- Lernerfahrungen mit verteilten SoftwareentwicklerInnen bei Projektabschluss (siehe Kapitel *3.6 Lernerfahrungen mit verteilten SoftwareentwicklerInnen bei Projektabschluss*)

In der letzten Prozessgruppe „Abschluss“ müssen die gesammelten Lernerfahrungen des verteilten Softwareprojektteams vollständig dokumentiert und für zukünftige MitarbeiterInnen und Projekte in einer Wissensdatenbank gesichert werden. Der Zweck ist, aus diesem verteilten Softwareprojekt und anderen Projekten für die Linearorganisation zu lernen. Die Erfassung von Lernerfahrung kann in allen Projektmanagementprozessen erfolgen (siehe Kapitel *3.4 Umsetzung und Controlling von verteilten Softwareprojekten*), jedoch ist es wichtig, alle Erfahrungen spätestens zum Projektabschluss zu erfassen und entsprechend bereitzustellen. (siehe Kapitel *2.4.1 Projektmanagement*)

Das Wissensmanagement stellt in der verteilten Softwareentwicklung eine generelle Herausforderung dar. Die Erfahrungen, Methoden, Entscheidungen und Fähigkeiten der Teammitglieder müssen während des Entwicklungsprozesses durch einen effektiven Informationsaustausch gesammelt werden, damit Teammitglieder die Erfahrungen ihrer VorgängerInnen nutzen können. Dies führt folglich zur Reduktion von Kosten sowie überflüssiger Arbeit. Ohne effektive Mechanismen zum Informations- und Wissensaustausch können ManagerInnen die Vorteile der verteilten Softwareentwicklung nicht nutzen. Der Wissensaustausch wird erleichtert, indem ein Produkt- oder Prozess-Repository verwaltet wird – dieses beinhaltet verschiedene Quellen, wie z. B. E-Mail, Online-Diskussionen etc. (siehe Kapitel *2.1.1 Herausforderungen der verteilten Softwareentwicklung*)

Die Lernerfahrungen werden innerhalb der Projektumsetzung in „Sprint Retrospektiven“-Besprechungen gesammelt. Diese Erfahrungen basieren auf proaktiven Beiträgen der verteilten SoftwareentwicklerInnen im Zuge der Retrospektiven. (siehe Kapitel *3.4 Umsetzung und Controlling von verteilten Softwareprojekten*)

4.2 Operationalisierung

Die Operationalisierung oder Messbarmachung definiert, wie theoretische Strukturen beobachtbar und messbar gemacht werden können. Sie ist für empirische Arbeitswissenschaften von großer Relevanz, da sie die Basis für durchführbare Messungen ist. Zudem muss beim Testen von aufgestellten Hypothesen eine geeignete Art der Operationalisierung gefunden werden. (Bortz, 1984)

4.2.1 Qualitative Inhaltsanalyse

Die Messbarmachung beruht in dieser Arbeit auf einer qualitativen Inhaltsanalyse. Auf Basis dieser sollen die aufgestellten Hypothesen in Kooperation mit ExpertInnen untersucht werden.

Es bestehen mehrere Vorgehensarten der qualitativen Inhaltsanalyse – ein bekanntes Vorgehen ist nach dem deutschen Psychologen Philipp A. E. Mayring benannt. Generell werden anhand von qualitativen Inhaltsanalysen Gespräche strukturiert analysiert. Das Ziel von Inhaltsanalysen ist es, theoretische Themen nach definierten Regeln zu untersuchen und daraus Schlussfolgerungen aus durchgeführten Kommunikationen abzuleiten. (Mayring, 2015)

Eine qualitative Inhaltsanalyse muss folgende Eigenschaften enthalten (Mayring, 2015):

- Analog zu quantitativen Vorgehen müssen qualitative Inhaltsanalysen ein systematisches Schema besitzen.
- Analog zu quantitativen Vorgehen können auch bei qualitativen Inhaltsanalysen Kategorisierungen definiert werden, um Rohdaten jeweils einzuordnen.
- Die erhaltenen Materialen je Kommunikation müssen gesamtheitlich betrachtet werden. Die Rohdaten müssen in ein gemeinsames Kommunikationsmodell eingebettet werden, um übergreifende Verknüpfungen zwischen den Rohdaten erkennen zu können.

In der qualitativen Inhaltsanalyse wird vorbereitetes Sprachmaterial untersucht. Für die Interpretation muss das Quellmaterial detailliert analysiert werden. (Mayring, 2015)

Im Rahmen dieser Arbeit wird das Quellmaterial, wie folgt gewonnen (Mayring, 2015):

- **Festlegung des Materials**: Die ExpertInneninterviews werden aufgezeichnet. In weiterer Folge werden diese transkribiert und dienen als Basis für die Inhaltsanalyse. Die Auswahl der ExpertInnen kann dem Kapitel *4.2.2 Auswahl der ExpertInnen* entnommen werden.
- **Entstehungssituation analysieren**: Für die ExpertInneninterviews wurde die Kommunikationssoftware „Zoom“ verwendet. Mithilfe der Software konnten ebenso gleichzeitig Tonaufnahmen aufgenommen und archiviert werden.
- **Formale Eigenschaften des Materials**: Die Transkription erfolgt im Textverarbeitungsprogramm „Microsoft Word“. Die transkribierten Phrasen wurden von Sprach-, Grammatik- und Ausdrucksfehlern bereinigt. Auf diese Weise konnten umgangssprachliche Eigenschaften herausgefiltert werden und das Material wurde dadurch untersuchungskonform.

Das gewonnene Quellmaterial muss durch eine wissenschaftliche Interpretation untersucht werden. Hierfür muss ein strukturierter und zielgerichteter Fragebogen erzeugt werden. Die Interpretation darf nicht objektiv vom Autor durchgeführt werden. Die Definition der Fragestellungen wird in zwei Schritten abgewickelt. (Mayring, 2015)

Die Fragen müssen auf dem Sinn dieser Arbeit basieren – also auf der Forschungsfrage: *„Welche Prozesse ermöglichen die Einführung und Steuerung einer verteilten Softwareentwicklung für die Projektabwicklung und wie sind diese in ein Prozessmodell überführbar?“*. Der Zweck hinter

dieser Forschungsfrage ist die Ermittlung eines Prozessmodells mit mehreren Prozessen, welche für die Einführung und Steuerung eines verteilten Softwareentwicklungsprojektes notwendig sind. Diese Prozesse decken spezifische Anforderungen ab, welche für verteilte Softwareentwicklungsprojekte notwendig sind. Diese Anforderungen sind bereits in der Hypothesenbildung (siehe Kapitel *4.1 Hypothesenbildung*) beschrieben und müssen durch die qualitative Inhaltsanalyse praxisnah untersucht werden.

Im darauffolgenden Schritt wird im Zuge der qualitativen Inhaltsanalyse die Entscheidung für eine Analysetechnik getroffen. Aufgrund der ausgewählten Technik wird die Analyse für die LeserInnen nachvollziehbar und intersubjektiv überprüfbar (Mayring, 2015). In dieser Arbeit wird der Ansatz zur deduktiven Kategorienbildung durchgeführt. In dieser wird jede Fragestellung jeweils einer oder mehreren Kategorien zugeordnet (Mayring, 2015). Die definierten Kategorien basieren auf den aufgestellten Hypothesen (siehe Kapitel *4.1 Hypothesenbildung*), welche wiederum aus den theoretischen Erkenntnissen (siehe Kapitel *2 Theoretischer Ansatz des Prozessmodells*) sowie dem entwickelten Prozessmodell (siehe Kapitel *3 Entwicklung des Prozessmodells*) dieser Arbeit abgeleitet wurden.

Folgende, auf den Hypothesen dieser Arbeit beruhenden Kategorien, werden definiert (siehe Kapitel *4.1 Hypothesenbildung*):

- **Personal:** beschreibt das Personal, welches beschafft werden muss.
- **Beschaffung:** beschreibt die Personalbeschaffung von verteilten SoftwareentwicklerInnen.
- **Externe PersonaldienstleisterInnen**: beschreibt das Engagieren von externen PersonaldienstleisterInnen hinsichtlich spezifischen Know-hows.
- **Land:** beschreibt den Staat oder die Region, aus welchem/r das Personal beschafft wird.
- **Aufbauorganisation:** beschreibt die Linienorganisation, in welcher verteilte SoftwareentwicklerInnen arbeiten.
- **Vorgesetzte:** beschreibt Vorgesetzte von verteilten SoftwareentwicklerInnen.
- **Standort:** beschreibt den geografischen Arbeitsort von verteilten SoftwareentwicklerInnen.
- **Ressourcenplanung:** beschreibt den Ressourcenplan für die Koordinierung von verteilten SoftwareentwicklerInnen.
- **Scrum:** beschreibt die Umsetzung von verteilten Softwareprojekten mit Scrum.
- **Projektumsetzung:** beschreibt die Projektumsetzung mit verteilten SoftwareentwicklerInnen.
- **Kommunikationsaufwand:** beschreibt den Kommunikationsaufwand, welcher durch Scrum entsteht.
- **Soll-/Ist-Vergleich:** beschreibt den Vergleich zwischen geplantem und aktuellem Fortschritt in verteilten Softwareprojekten.

- **Projektsituation:** beschreibt die aktuelle Situation im verteilten Softwareprojekt.
- **Maßnahmen:** beschreibt notwendige Maßnahmen im verteilten Softwareprojekt, welche an die verteilten SoftwareentwicklerInnen vermittelt werden müssen.
- **Lernerfahrung:** beschreibt Erfahrungen, welche verteilte SoftwareentwicklerInnen in Projekten sammeln.
- **Wissensdatenbank:** beschreibt eine organisationszentrale Wissensdatenbank, in der die Lernerfahrungen von verteilten SoftwareentwicklerInnen gesammelt werden.

Für die Kodierung der Rohdaten wird die Textanalysesoftware „MAXQDA“ genutzt. Im ersten Schritt werden die definierten Kategorien in der Software konfiguriert. Darauffolgend werden die transkribierten Interviews in die Software importiert. In der Textanalysesoftware werden die Rohdaten verarbeitet und die Aussagen der ExpertInnen den einzelnen Kategorien zugeordnet. Die Ergebnisse aus „MAXQDA“ sind in der beigelegten „Compact Disc“ (CD) dieser Arbeit archiviert – der Verzeichnisname lautet „Rohdaten“. Auf diese Weise wird den LeserInnen Zugriff auf die Kategorienzuordnungen geboten.

4.2.2 Auswahl der ExpertInnen

Die Antwort auf die Forschungsfrage soll einen qualitativ wertvollen Mehrwert liefern. Aus diesem Grund werden ExpertInnen für die Validierung des Prozessmodells herangezogen.

„Ein Experte oder eine Expertin zeichnet sich durch umfangreiches Wissen in einem bestimmten Bereich aus. Solche fachkundigen Menschen lösen Aufgaben und Herausforderungen zielorientiert, sachgerecht, methodengeleitet und selbständig.“ (Genau, 2020)

Im Rahmen der Validierung wurden drei ExpertInnen befragt. Alle drei ExpertInnen sind oder waren in der Guid.New GmbH im Managementbereich tätig. Die Guid.New GmbH selbst ist ein Dienstleistungsunternehmen im Bereich der Abwicklung und Entwicklung von Individualsoftwareprojekten in Österreich. Bereits vor der „COVID-19“-Pandemie beschäftige das Unternehmen hauptsächlich verteilte SoftwareentwicklerInnen für die Entwicklung von Individualsoftwareprojekten und besitzt aufgrund dessen langjähriges Know-how in Bezug auf derartige Prozesse. Wie bereits erwähnt, sind/waren die ausgewählten ExpertInnen im Management des Unternehmens tätig. Dies ist relevant für die Forschungsfrage dieser Arbeit, da es sich um Prozesse für die Einführung und Steuerung von verteilten Softwareprojekten handelt. Die ExpertInnen werden im Folgenden kurz vorgestellt.

Experte 1 – Dipl.-Ing. (FH) Stefan Schnuderl

Herr Schnuderl ist ein Mitgründer sowie Geschäftsführer der Guid.New GmbH. Bereits vor der Gründung des Unternehmens erlangte der Experte langjährige Erfahrung in der Leitung und Abwicklung von komplexen Softwareprojekten. Zudem verzeichnet er mehrere Unternehmensgründungen, wobei schon erfolgreiche Anteilsverkäufe seiner ehemaligen Unternehmungen durchgeführt wurden. Neben der Leitung und Abwicklung von verteilten Softwareentwicklungsprojekten ist er ebenso in der Personalplanung des Unternehmens involviert und führt Bewerbungsgespräche mit verteilten SoftwareentwicklerInnen durch.

Experte 2 – Dipl.-Ing. Harald Schaffernak

Herr Schaffernak ist ein Mitgründer der Guid.New GmbH. Als einer von vier Gesellschaftern des Unternehmens erlebte der Experte die Entstehung von Anfang an und liefert dadurch einen hohen Erfahrungswert für diese Untersuchung. In seiner operativen Tätigkeit im Unternehmen ist er als Projektleiter von verteilten Softwareprojekten beschäftigt und führt ebenso Bewerbungsgespräche mit verteilten SoftwareentwicklerInnen durch. Neben seiner operativen Tätigkeit wirkt er ebenso im Prozessmanagement des Unternehmens mit und leistet somit einen wesentlichen Beitrag zur Gestaltung der Arbeitsweise mit verteilten SoftwareentwicklerInnen.

Expertin 3 – Marina Opferkuch, MSc

Frau Opferkuch war mehr als vier Jahre als „Operations Managerin“ der Guid.New GmbH beschäftigt. Sie übernahm in ihrer ehemaligen Funktion die Agenden für die Personalplanung, Personalbeschaffung, Personalverrechnung sowie Verrechnung der verteilten Softwareentwicklungsprojekte. Die Expertin wirkte ebenso im Prozessmanagement des Unternehmens mit und leistete somit einen wesentlichen Beitrag zur Gestaltung der Arbeitsweise mit verteilten SoftwareentwicklerInnen.

4.2.3 Leitfaden

Der Leitfaden kann nach zwei unterschiedlichen Schemata aufgebaut sein: Als offene Fragen, welche in einer definierten Reihenfolge zu stellen sind, oder als narrative Fragen. Ebenso ist es möglich den Leitfaden auf Basis eines Mix aus beiden Schemata aufzubauen. Die Fragestellungen im Leitfaden müssen im Vorhinein ausformuliert werden, wobei es während der Befragung erlaubt ist, zusätzliche Fragen zu stellen. Der Leitfaden dient somit als roter Faden, von dem geringfügig abgewichen werden kann. (Helfferich, 2019)

Im Einstiegsteil des Leitfadens werden organisatorische Themen angesprochen. Dazu zählen:

- Zweck des Interviews
- Erlaubnis das Interview hinsichtlich Video und Ton aufzunehmen und zu archivieren
- Erlaubnis das Interview zu transkribieren und in der Arbeit auszuwerten
- Erlaubnis den Namen, akademischen Titel sowie kurze Vorstellung der Expertin/des Experten in der Arbeit verwenden zu dürfen
- Spezifische Frage an Experte 1: Erlaubnis die Guid.New GmbH als Referenzunternehmen im Rahmen dieser Arbeit zu verwenden

Nach den organisatorischen Abstimmungen wird den ExpertInnen die Forschungsfrage vorgestellt sowie der Hintergrund der Forschungsfrage näher beschrieben. Auf diese Weise wird den ExpertInnen vermittelt, auf welcher Problemstellung das durchzuführende Interview basiert.

Darauffolgend beginnt die Befragung in Bezug auf die Schlüsselfragen. Diese basieren auf den aufgestellten Hypothesen (siehe Kapitel *4.1 Hypothesenbildung*) und werden jeweils mit einer Begründungsaufforderung beantwortet. Hierdurch erhalten die Fragestellungen einen offenen

Charakter, damit die ExpertInnen im Interview zusätzliche Erfahrungen einbringen können und somit weitere Erkenntnisse für die Beantwortung der Forschungsfrage vermitteln.

Folgende Fragen werden im Leitfaden auf Basis der Hypothesen (siehe Kapitel *4.1 Hypothesenbildung*) integriert:

- *„Sollen bei der Beschaffung von verteilten SoftwareentwicklerInnen externe PersonaldienstleisterInnen hinsichtlich spezifischen Know-hows engagiert werden? Wie lautet die Begründung?“*
- *„Sollen in der Aufbauorganisation die Vorgesetzten der verteilten SoftwareentwicklerInnen am gleichen geografischen Standort arbeiten und mit diesen die Ressourcenplanung koordiniert werden? Wie lautet die Begründung?“*
- *„Bietet Scrum Vorteile für die Projektumsetzung mit verteilten SoftwareentwicklerInnen, da hierfür hoher Kommunikationsaufwand betrieben wird? Wie lautet die Begründung?“*
- *„Sollen ProjektleiterInnen den verteilten SoftwareentwicklerInnen wöchentlich den Soll-/Ist-Vergleich im verteilten Softwareprojekt darlegen, um daraus folgend die aktuelle Projektsituation und notwendige Maßnahmen besser zu vermitteln? Wie lautet die Begründung?“*
- *„Sollen die Lernerfahrungen von verteilten SoftwareentwicklerInnen in einer organisationszentralen Wissensdatenbank dokumentiert werden und sollen verteilte SoftwareentwicklerInnen dabei aktiv mitwirken? Wie lautet die Begründung?“*

Wie bereits am Anfang dieses Subkapitels beschrieben, können zusätzliche Fragen während des Interviews aufkommen (Helfferich, 2019). Für eine detailliertere Erklärung der Schlüsselfragen dürfen die ExpertInnen im Zuge des Interviews jederzeit Gegenfragen stellen.

Im Schlussteil des Leitfadens können die ExpertInnen Ergänzungen und zusätzliche Gegenfragen stellen. Außerdem wird ihnen ein Ausblick auf den weiteren Verlauf der Arbeit kommuniziert und die Möglichkeit gegeben, hinzukommende Wünsche mitzuteilen.

4.3 Ergebnisse

Die Ergebnisse der qualitativen Inhaltsanalyse werden im Folgenden dargestellt. Die Generalisierung dient zur Zusammenfassung der Paraphrasen der ExpertInnen und zur darauffolgenden Zuordnung je Kategorie aus der deduktiven Kategorienbildung. Diese generalisierten Objekte unterstützen bei der Beantwortung der Forschungsfrage.

Kategorie	Generalisierung
Personal	Arbeitgeberimage entwickeln
	Kulturelle Unterschiede
Beschaffung	Ansprechperson im Zielland
	Unterschiedliche Personalquellen

Externe PersonaldienstleisterInnen	Durch Know-how vernetzt
	Im eigenen Kerngeschäft bleiben
	Netzwerk aufbauen
Land	Je Land eigene/r PersonaldienstlerIn
	Ansprechperson im Zielland
	Kulturelle Unterschiede
	Keinen Ruf als Arbeitgeber
	Keine Mund-zu-Mund-Propaganda möglich in fremden Ländern
Standort	Zusammenarbeit mit Hauptstandort
Vorgesetzte	Erfahrung mit lokalen Gegebenheiten
	Benötigte Qualifikation
	Zukunftsaussichten geben
	Fachliche und disziplinarische Expertise
	Aktive Mitwirkung im Entwicklungsteam
Ressourcenplanung	Von der Größe des Standorts abhängig
	Verantwortung übergeben
	Zusammenarbeit mit Hauptstandort
Aufbauorganisation	Aufbauorganisation wird skalierbar
	Verantwortung übergeben
Scrum	Vorgegebene Termine
	Stärkere Kommunikation durch vorgegebene Dailies
	Probleme werden schneller angesprochen
	Besseres Miteinander im Team
	Soziale Bindung im Team
	Eigene Rolle als Kommunikationsträger
Projektumsetzung	Zeitnahe Abstimmung
	Keine Kontrolle möglich remote
Kommunikationsaufwand	Stärkere Kommunikation durch vorgegebene Dailies

	Besseres Miteinander im Team
	Eigene Rolle als Kommunikationsträger
Soll-/Ist-Vergleich	Jede zweite Woche
	Transparente Kommunikation
	Zufriedenheit nicht schmälern
Projektsituation	Serientermin im Kommunikationsregelwerk
	Interesse der verteilten SoftwareentwicklerInnen
	Stärkt das Teamgefühl
Maßnahmen	Interesse der verteilten SoftwareentwicklerInnen
	Individuelle Maßnahmen gemeinsam erarbeiten
Lernerfahrung	Absprache sorgt nicht für Lerneffekt
	Kollegiales Miteinander fehlt
Wissensdatenbank	Seltene Nutzung
	Aktive Mitwirkung der verteilten SoftwareentwicklerInnen
	Nicht jeder Wissenseintrag ist eine Lernerfahrung
	Motivation durch Unternehmen nötig
	Skalierbarkeit der Softwareprojektteams
	Aktive Mitwirkung sorgt für Lerneffekt

Tabelle 1 - Generalisierung in der qualitativen Inhaltsanalyse

Die Kategorien basieren auf den Hypothesen (siehe Kapitel *4.1 Hypothesenbildung*), welche wiederum die Anforderungen an das Prozessmodell beinhalten. Im nächsten Schritt werden, die zu den Kategorien zugeordneten Generalisierungen nach Häufigkeit gelistet. Die Paraphrasen der ExpertInnen zu den häufigsten Generalisierungsobjekten werden im Detail beschrieben. Auf diese Weise wird die Sichtweise der ExpertInnen hinsichtlich der spezifischen Anforderungen an das Prozessmodell aufgezeigt und Überschneidungen in den Aussagen hervorgehoben.

4.3.1 Personal

Die Generalisierung der Kategorie „Personal“ ergab folgende Objekte:

- Kulturelle Unterschiede (2/3)
- Arbeitgeberimage entwickeln (1/3)

Experte 2 und Expertin 3 teilen sich die Meinung, dass die kulturellen Unterschiede je Region eine Herausforderung in der Personalbeschaffung in fremden Ländern sind. Aus diesem Grund müssen externe PersonaldienstleisterInnen engagiert werden, da diese die kulturellen Gegebenheiten bereits kennen. Expertin 3 trifft die zusätzliche Aussage, dass es sogar Unterschiede je Stadt gibt und je nach Region eine Vollauslastung oder ein „9 to 5“-Modell von verteilten SoftwareentwicklerInnen bevorzugt wird.

4.3.2 Beschaffung

Die Generalisierung der Kategorie „Beschaffung“ ergab folgende Objekte:

- Ansprechperson im Zielland (2/3)
- Unterschiedliche Personalquellen (1/3)

Experte 1 und Expertin 3 gehen davon aus, dass je Zielland eine eigene Ansprechperson oder ein/e PersonaldienstleisterIn notwendig ist. Ergänzend meint Expertin 3, dass es sich über mehrere Jahre hinweg herauskristallisiert, mit welchen externen PersonaldienstleisterInnen die Zusammenarbeit am besten funktioniert. Basierend auf dieser Erfahrung wird es möglich, ein Netzwerk an PersonaldienstleisterInnen je Land aufzubauen.

4.3.3 Externe PersonaldienstleisterInnen

Die Generalisierung der Kategorie „Externe PersonaldienstleisterInnen“ ergab folgende Objekte:

- Im eigenen Kerngeschäft bleiben (1/3)
- Durch Know-how vernetzt (1/3)
- Netzwerk aufbauen (1/3)

In dieser Kategorie gab es zwischen den Aussagen der ExpertInnen keine Überschneidungen. Dennoch lieferte Experte 2 eine neue Erkenntnis: Externe PersonaldienstleisterInnen sollen angeheuert werden, damit sich ein Unternehmen auf sein Kerngeschäft konzentrieren kann.

4.3.4 Land

Die Generalisierung der Kategorie „Land“ ergab folgende Objekte:

- Kulturelle Unterschiede (2/3)
- Je Land eigene/r Personaldienstleister (2/3)

- Ansprechperson im Zielland (2/3)
- Keinen Ruf als Arbeitgeber (2/3)
- Keine Mund-zu-Mund-Propaganda möglich in fremden Ländern (1/3)

Die Generalisierungen innerhalb der Kategorie „Land“ überschneiden sich weitgehend mit der Kategorie „Beschaffung“ (siehe Kapitel *4.3.2 Beschaffung*). Eine weitere Information lieferten jedoch Experte 2 und Expertin 3 in Hinsicht auf die initiale Herausforderung in fremden Ländern: Als Unternehmen hat man am Arbeitsmarkt in einem fremden Land anfangs keinen Ruf – mit externen PersonaldienstleisterInnen kann man diese Hürde überwinden.

4.3.5 Standort

Die Generalisierung der Kategorie „Standort“ ergab folgendes Objekt:

- Zusammenarbeit mit Hauptstandort (3/3)

Alle ExpertInnen sind sich darüber einig, dass eine effektive Zusammenarbeit zwischen verteilten SoftwareentwicklerInnen und Hauptstandort nur möglich ist, wenn sich alle Personen an der Kooperation beteiligen. Der Vorgesetzte der verteilten SoftwareentwicklerInnen muss daher von seinen MitarbeiterInnen unterstützt werden, damit eine effektive Zusammenarbeit mit dem Hauptstandort gewährleistet werden kann.

4.3.6 Vorgesetzte

Die Generalisierung der Kategorie „Vorgesetzte“ ergab folgende Objekte:

- Benötigte Qualifikation (3/3)
- Erfahrung mit lokalen Gegebenheiten (2/3)
- Aktive Mitwirkung im Entwicklungsteam (2/3)
- Zukunftsaussichten geben (1/3)
- Fachliche und disziplinarische Expertise (1/3)

Alle ExpertInnen sind sich darüber einig, dass der/die Vorgesetzte der verteilten SoftwareentwicklerInnen die benötigten Qualifikationen erfüllen muss. Wobei Experte 2 dies als Herausforderung sieht, eine/n entsprechende/n SoftwareentwicklerIn mit diesen Qualifikationen zu finden und schlussfolgernd als Vorgesetzte/n zu ernennen. Experte 1 und Expertin 3 erwähnen als zusätzliche Voraussetzung, dass der/die Vorgesetzte sich mit lokalen Gegebenheiten auskennen muss – wie zum Beispiel steuerlichen oder unternehmerischen Themen.

4.3.7 Ressourcenplanung

Die Generalisierung der Kategorie „Ressourcenplanung“ ergab folgende Objekte:

- Verantwortung übergeben (3/3)
- Zusammenarbeit mit Hauptstandort (3/3)
- Von der Größe des Standorts abhängig (1/3)

Alle ExpertInnen sind sich darüber einig, dass verteilten SoftwareentwicklerInnen Verantwortung übergeben werden muss. Darunter fällt unter anderem die Ressourcenplanung. Die/der Vorgesetzte muss mit dem Hauptstandort aktiv zusammenarbeiten und die verteilten SoftwareentwicklerInnen koordinieren. Wie bereits in der Kategorie „Standort“ erwähnt, muss die/der Vorgesetzte auch in der Ressourcenplanung von seinen MitarbeiterInnen umfassend unterstützt werden.

4.3.8 Aufbauorganisation

Die Generalisierung der Kategorie „Aufbauorganisation“ ergab folgende Objekte:

- Verantwortung übergeben (3/3)
- Aufbauorganisation wird skalierbar (2/3)

Wie bereits in der Kategorie „Ressourcenplanung“ (siehe Kapitel *4.3.7 Ressourcenplanung*) erwähnt, muss verteilten SoftwareentwicklerInnen Verantwortung übergeben werden. Experte 1 und Expertin 3 beziehen sich zudem auch auf die Skalierbarkeit von Aufbauorganisationen mit verteilten SoftwareentwicklerInnen. Vor allem Experte 1 plädiert auf die Übergabe von Verantwortung, damit die Aufbauorganisation skalierbar wird und das Unternehmen langfristig wachsen kann.

4.3.9 Scrum

Die Generalisierung der Kategorie „Scrum“ ergab folgende Objekte:

- Besseres Miteinander im Team (3/3)
- Vorgegebene Termine (3/3)
- Probleme werden schneller angesprochen (2/3)
- Soziale Bindung im Team (2/3)
- Stärkere Kommunikation durch vorgegebene Dailies (1/3)
- Eigene Rolle als Kommunikationsträger (1/3)

Alle ExpertInnen sind sich darüber einig, dass das Kommunikationsregelwerk von Scrum hinsichtlich vorgegebener Serientermine für ein besseres Miteinander innerhalb des verteilten Softwareprojektteams sorgt. Laut Experte 1 und Experte 2 sind die Gründe dafür, dass Probleme

im verteilten Softwareprojektteam frühzeitig angesprochen werden und die soziale Bindung zwischen den verteilten SoftwareentwicklerInnen gestärkt wird.

4.3.10 Projektumsetzung

Die Generalisierung der Kategorie „Projektumsetzung“ ergab folgende Objekte:

- Keine Kontrolle möglich remote (1/3)
- Zeitnahe Abstimmung (1/3)

In dieser Kategorie gab es zwischen den Aussagen der ExpertInnen keine Überschneidungen. Dennoch lieferte Experte 1 die Erkenntnis, dass die Kontrolle der verteilten SoftwareentwicklerInnen während der Arbeitszeit nicht möglich ist, aber auch nicht gewünscht wird. Von verteilten SoftwareentwicklerInnen wird eine entsprechende Reife vorausgesetzt und somit davon ausgegangen, dass diese ihre Arbeitsleistung verlässlich und selbstverantwortlich erbringen.

4.3.11 Kommunikationsaufwand

Die Generalisierung der Kategorie „Kommunikationsaufwand“ ergab folgende Objekte:

- Besseres Miteinander im Team (3/3)
- Stärkere Kommunikation durch vorgegebene Dailies (1/3)
- Eigene Rolle als Kommunikationsträger (1/3)

Wie bereits in der Kategorie „Scrum“ (siehe Kapitel *4.3.9 Scrum*) erwähnt, sorgt der Kommunikationsaufwand in Scrum für ein besseres Miteinander im verteilten Softwareprojektteam. Der hohe Kommunikationsaufwand entsteht durch die vorgegebenen Serientermine im Kommunikationsregelwerk von Scrum.

4.3.12 Soll-/Ist-Vergleich

Die Generalisierung der Kategorie „Soll-/Ist-Vergleich“ ergab folgende Objekte:

- Transparente Kommunikation (3/3)
- Jede zweite Woche (3/3)
- Zufriedenheit nicht schmälern (1/3)

Alle ExpertInnen sind sich darüber einig, dass die transparente Kommunikation des Soll-/Ist-Vergleichs in verteilten Softwareprojekten wichtig und gut ist. Ebenso ist die Regelmäßigkeit der Kommunikation des Soll-/Ist-Vergleichs ausschlaggebend. Ansonsten besteht die Gefahr, dass die verteilten SoftwareentwicklerInnen das Interesse an der Projektsituation und den vermittelten Maßnahmen verlieren. Die ExpertInnen schlagen einen zweiwöchigen Serientermin Wochen vor.

4.3.13 Projektsituation

Die Generalisierung der Kategorie „Projektsituation“ ergab folgende Objekte:

- Interesse der verteilten SoftwareentwicklerInnen (3/3)
- Stärkt das Teamgefühl (2/3)
- Serientermin im Kommunikationsregelwerk (2/3)

Alle ExpertInnen sind sich darüber einig, dass bei der Kommunikation des Soll-/Ist-Vergleichs das Interesse der verteilten SoftwareentwicklerInnen vorhanden sein muss. Die Regelmäßigkeit der Kommunikation des Soll-/Ist-Vergleichs hat einen wesentlichen Einfluss auf das Interesse. Zudem erwähnt Experte 1, dass das Interesse einzelner SoftwareentwicklerInnen während der transparenten Kommunikation der Projektsituation mitverfolgt werden soll.

4.3.14 Maßnahmen

Die Generalisierung der Kategorie „Maßnahmen“ ergab folgende Objekte:

- Interesse der verteilten SoftwareentwicklerInnen (3/3)
- Individuelle Maßnahmen gemeinsam erarbeiten (1/3)

Alle ExpertInnen sind sich darüber einig, dass sich verteilte SoftwareentwicklerInnen für die vermittelten Maßnahmen interessieren müssen. Hinzukommend trifft Expertin 3 die Aussage, dass die Maßnahmen mit den jeweiligen verteilten SoftwareentwicklerInnen individuell erarbeitet werden müssen, um die Projektatmosphäre nicht negativ zu beeinflussen.

4.3.15 Lernerfahrung

Die Generalisierung der Kategorie „Lernerfahrung“ ergab folgende Objekte:

- Absprache sorgt nicht für Lerneffekte (2/3)
- Kollegiales Miteinander fehlt (1/3)

Experte 1 und Expertin 3 gehen davon aus, dass die Dokumentation von Lernerfahrungen in einer Wissensdatenbank für einen nachhaltigen Lerneffekt bei verteilten SoftwareentwicklerInnen sorgt. Die einfache Absprache dieser Erfahrungen im Nachgang des verteilten Softwareprojektes entwickelt das Know-how der verteilten SoftwareentwicklerInnen nicht weiter.

4.3.16 Wissensdatenbank

Die Generalisierung der Kategorie „Wissensdatenbank“ ergab folgende Objekte:

- Aktive Mitwirkung der verteilten SoftwareentwicklerInnen (3/3)
- Aktive Mitwirkung sorgt für Lerneffekt (2/3)
- Skalierbarkeit von Softwareprojektteams (1/3)

- Motivation durch Unternehmen nötig (1/3)
- Seltene Nutzung (1/3)
- Nicht jeder Wissenseintrag ist eine Lernerfahrung (1/3)

Alle ExpertInnen sind sich darüber einig, dass verteilte SoftwareentwicklerInnen aktiv an der langfristigen Pflege einer Wissensdatenbank beteiligt sein müssen. Experte 1 und Expertin 3 fügen hinzu, dass diese aktive Mitwirkung für einen nachhaltigen Lerneffekt und den Aufbau von Know-how unter den verteilten SoftwareentwicklerInnen sorgt. Des Weiteren trifft Expertin 3 die Aussage, dass eine qualitative Wissensdatenbank eine Voraussetzung für die Skalierbarkeit von verteilten Softwareprojektteams ist. Vor allem neu eingestellte SoftwareentwicklerInnen brauchen initial eine derartige Wissensdatenbank.

4.4 Beantwortung der Forschungsfrage

Die definierte Forschungsfrage (siehe Kapitel *1.1 Forschungsfrage*) wird nach der Entwicklung (siehe Kapitel *3 Entwicklung des Prozessmodells*) und Validierung des Prozessmodells (siehe Kapitel *4 Validierung des Prozessmodells*) folgenderweise beantwortet:

Auf Basis des entwickelten Prozessmodells und der untersuchten Anforderungen im Rahmen der qualitativen Inhaltsanalyse mit ExpertInnen ermöglichen die Prozesse *„Initiierung eines verteilten Softwareprojektes“*, *„Beschaffung von verteilten SoftwareentwicklerInnen“*, *„Einführung von verteilten SoftwareentwicklerInnen“*, *„Ressourcenplanung bei verteilten Softwareentwicklungsprojekten“*, *„Umsetzung von verteilten Softwareentwicklungsprojekten“*, *„Controlling von verteilten Softwareentwicklungsprojekten“*, *„Anforderungsmanagement in verteilten Softwareentwicklungsprojekten“* sowie *„Lernerfahrungen mit verteilten SoftwareentwicklerInnen bei Projektabschluss“* die Einführung und Steuerung einer verteilten Softwareentwicklung für die Projektabwicklung und sind in ein gesamtheitliches Prozessmodell überführbar.

Die in der Antwort erwähnten Anforderungen an das Prozessmodell sind folgende:

- Kulturelle Unterschiede je Zielland werden berücksichtigt.
- Übergabe von Verantwortung an verteilte SoftwareentwicklerInnen.
- Besseres Miteinander im verteilten Softwareprojektteam durch Kommunikationsaufwand anhand vorgegebener Termine.
- Transparente Kommunikation von Projektsituation und Maßnahmen an verteilte SoftwareentwicklerInnen in gemäßigten Zeitabständen.
- Aktives Mitwirken von verteilten SoftwareentwicklerInnen in der Pflege von Wissensdatenbanken sorgen für einen nachhaltigen Lerneffekt und den Aufbau von „Know-how“.

Die Aktivitäten des Prozessmodells werden im Kapitel *3 Entwicklung des Prozessmodells* detailliert beschrieben und grafisch dargestellt.

5 CONCLUSIO

In diesem Kapitel wird die Arbeit schlussendlich zusammengefasst sowie ein Ausblick auf weitere Forschungsmöglichkeiten gegeben.

5.1 Zusammenfassung

In dieser Arbeit wurde im ersten Schritt die theoretische Grundlage für die Problemstellung aufbereitet. Darauf basierend wurden Prozesse hinsichtlich Einführung und Steuerung einer verteilten Softwareentwicklung für die Projektabwicklung entwickelt und in ein gesamtheitliches Prozessmodell überführt.

Ein verteiltes Softwareprojekt wird in fünf Prozessgruppen aufgeteilt: Initiierung, Planung, Umsetzung, Controlling sowie Abschluss. Das entwickelte Prozessmodell basiert auf allen fünf Prozessgruppen und umfasst dahingehend alle Prozesse von Anfang bis Ende eines verteilten Softwareprojektes. Es handelt sich um folgende Prozesse des Modells: *„Initiierung eines verteilten Softwareprojektes“, „Beschaffung von verteilten SoftwareentwicklerInnen“, „Einführung von verteilten SoftwareentwicklerInnen“, „Ressourcenplanung bei verteilten Softwareentwicklungsprojekten“, „Umsetzung von verteilten Softwareentwicklungsprojekten“, „Controlling von verteilten Softwareentwicklungsprojekten“, „Anforderungsmanagement in verteilten Softwareentwicklungsprojekten“* sowie *„Lernerfahrungen mit verteilten SoftwareentwicklerInnen bei Projektabschluss“*.

Im weiteren Verlauf wurde das Prozessmodell nach abgedeckten Anforderungen untersucht. Diese Untersuchung wurde im Rahmen einer qualitativen Inhaltsanalyse mit ExpertInneninterviews durchgeführt. Die Hypothesen für die qualitative Inhaltsanalyse basierten wiederum auf den fünf Prozessgruppen der Projektabwicklung. Je Prozessgruppe wurde eine Hypothese definiert. Die Fragestellungen für die ExpertInneninterviews wurden aus den ermittelten Hypothesen abgeleitet und je ExpertIn aufgearbeitet. Die Befragung der ExpertInnen sowie die darauffolgende qualitative Inhaltsanalyse lieferten die abgedeckten Anforderungen an das Prozessmodell: *„Kulturelle Unterschiede je Zielland werden berücksichtigt“, „Übergabe von Verantwortung an verteilte SoftwareentwicklerInnen“, „Besseres Miteinander im verteilten Softwareprojektteam durch Kommunikationsaufwand anhand vorgegebener Termine“, „Transparente Kommunikation von Projektsituation und Maßnahmen an verteilte SoftwareentwicklerInnen in gemäßigten Zeitabständen“* sowie *„Aktive Mitwirkung von verteilten SoftwareentwicklerInnen in der Pflege von Wissensdatenbanken sorgen für einen nachhaltigen Lerneffekt und den Aufbau von „Know-how“*.

Schlussfolgernd liefern die beschriebenen und grafisch dargestellten Prozesse des Modells sowie die ermittelten Anforderungen eine Antwort auf die Forschungsfrage.

5.2 Weitere Forschungsmöglichkeiten

Neben dem entwickelten Prozessmodell sowie den ermittelten Anforderungen ergaben sich in dieser Arbeit weitere Erkenntnisse hinsichtlich verteilter Softwareprojekte. So wurde in den ExpertInneninterviews das Thema *„Soziale Bindung bei verteilten SoftwareentwicklerInnen“* angesprochen, wobei dieser Punkt nicht näher betrachtet wurde. Des Weiteren wurden Schwierigkeiten in der Strukturierung von Wissensdatenbanken erwähnt und die Eingrenzung, ab wann eine Sammlung von Know-how als Wissensdatenbank gilt, wurde als schwierig angesehen. Zudem wurde die Frage gestellt, ob es eine ideale Wissensdatenbank für verteilte Organisationen gibt.

Darüber hinaus wurde erwähnt, dass vor allem die Bereiche Personal- sowie Projektverrechnung in verteilten Softwareprojekten eine große Herausforderung für die involvierten Personen darstellen. Entsprechend könnte evaluiert werden, wie ein derartiger Abschlussprozess hinsichtlich Projektabrechnung im bestehenden Modell aussehen könnte.

Es können weitere Prozesse, welche die Einführung und Steuerung von verteilten Softwareentwicklungen für die Projektabwicklung ermöglichen, auf dieselbe Herangehensweise entwickelt und untersucht werden. Zudem können mögliche Qualitätsmerkmale und Kennzahlen für das bereits bestehende Prozessmodell implementiert und ausgewertet werden.

Diese Arbeit kann somit als Grundlage für weiterführende Forschungen dienen.

ANHANG A - 1. Anhang

Person	Aussage
Haris Bećirević	Hallo Stefan! Danke für deine Zeit. Bei meiner Masterarbeit geht es um das Thema *„Ein Prozessmodell zur Einführung und Steuerung einer verteilten Softwareentwicklung für die Projektabwicklung am Beispiel eines österreichischen Individualsoftwaredienstleisters“*. Es geht um ein Prozessmodell, welches die Einführung von verteilten SoftwareentwicklerInnen beschreibt, sowie die Steuerung von verteilten Softwareprojekten – es geht um das ganze Konzept. Die Forschungsfrage lautet: *„Welche Prozesse ermöglichen die Einführung und Steuerung einer verteilten Softwareentwicklung für die Projektabwicklung und wie sind diese in ein Prozessmodell überführbar?“* Ich habe Prozesse mittels BPMN konzipiert und daraus Prozessmodelle gebildet. Diese Prozessmodelle besitzen bestimmte Eigenschaften. Die Fragen im Hauptteil dieses Interviews enthalten diese Eigenschaften und genau um diese geht es hauptsächlich im Interview. Ich möchte diese Eigenschaften mit Experten, wie dir, validieren.
Stefan Schnuderl	Mhm, gut.
Haris Bećirević	Passt. Bevor ich das Interview starte, möchte ich noch folgende Erlaubnisabfrage durchführen: Dieses Interview hinsichtlich Video und Ton aufzunehmen und zu archivieren Transkription des Interviews und Auswertung in der Masterarbeit Verwendung der Guid.New GmbH als Referenzunternehmen im Rahmen der Masterarbeit Verwendung deines Namens sowie akademischen Titels in der Masterarbeit Vorstellung deiner Person in der Masterarbeit
Stefan Schnuderl	Ja, alles gut.
Haris Bećirević	Die erste Frage lautet: *„Sollen bei der Beschaffung von verteilten SoftwareentwicklerInnen externe PersonaldienstleisterInnen hinsichtlich spezifischen Know-hows engagiert werden? Wie lautet die Begründung?“* Die Frage lautet konkret, ob es Sinn macht, je Region, externe PersonaldienstleisterInnen zu beschäftigen. Sei es in Weißrussland oder auch, wie bei meinem aktuellen Arbeitgeber, Bosnien und Herzegowina. Jeder externe Personaldienstleister besitzt, je Land, das jeweilige Know-how

	für die Beschaffung von Personal. Neben dem Know-how besitzt der externe Personaldienstleister ebenso die „Connections“ am Personalmarkt. Was denkst du darüber?
Stefan Schnuderl	Wir haben schon vor Guid.New Kontakte in Weißrussland geknüpft – also schon vor sehr langer Zeit. Wirklich funktioniert die Beschaffung von Personal erst, wenn man eine Ansprechperson im Zielland hat. Diese Kontaktperson muss mit den Gegebenheiten im Land vertraut sein und wissen, wo man Personal finden kann. Ohne diese Eigenschaften ist die Beschaffung von Personal im Zielland nicht möglich. Ansonsten müsste man selbst im Zielland sein und Personal finden – aber keiner möchte wirklich hinziehen. Mit unseren aktuellen externen Personaldienstleisterin funktioniert dies sehr gut, da sie durch ihr Know-how sehr vernetzt ist. Ich wüsste nicht, wie es anders funktionieren könnte, außer mit Personaldienstleister vor Ort.
Haris Bećirević	Und wenn du mehrere Zielländer hättest, würdest du je Land einen eigenen externen Personaldienstleister beschäftigen? Ist das die Schlussfolgerung?
Stefan Schnuderl	Ja. Je Land und möglichst dort vor Ort.
Haris Bećirević	Passt, nächste Frage: *„Sollen in der Aufbauorganisation die Vorgesetzten der verteilten SoftwareentwicklerInnen am gleichen geografischen Standort arbeiten und mit diesen die Ressourcenplanung koordiniert werden? Wie lautet die Begründung?“* Aus meiner Erfahrung bei der Guid.New GmbH weiß ich, dass die Ressourcenplanung allein von unserem Standort in Graz durchgeführt wurde – ohne einem Miteinbeziehen von verteilten SoftwareentwicklerInnen. Würde es Sinn machen, je geografischen Standort, einen eigenen Vorgesetzten aus den Reihen der verteilten SoftwareentwicklerInnen zu ernennen? Diesen würden wir in der Ressourcenplanung miteinbeziehen und dieser Person kommunizieren, wie viele Personalressourcen wir in welchem Zeitraum benötigen.
Stefan Schnuderl	Hm, ich denke das hängt primär von der Größe des jeweiligen Standorts ab. Ab einer gewissen Größe sollte ein Vorgesetzter ernannt werden, der die koordinative Oberhand am Standort behält. Bis jetzt haben wir es nicht vernünftig geschafft, da bedingt durch COVID-19 kein großes Team entstanden ist. Aber wenn wir in Weißrussland wieder mehr als fünf Entwickler haben, würde ich stark dafür plädieren, dass ein dortiger Vorgesetzter die Ressourcenplanung verantwortet. Das gesamte Entwicklerteam sollte zusammenarbeiten und den Vorgesetzten unterstützen. Auf diese Weise würde die Arbeit mit Graz viel effizienter laufen und die Aufbauorganisation wird skalierbar.

Haris Bećirević	Ok. Also würdest du ab einer bestimmten Größe je Standort raten, dort einen Vorgesetzten für die Ressourcenplanung einzusetzen.
Stefan Schnuderl	Ja, genau. Dieser Vorgesetzter sollte sich ebenso mit den lokalen Gegebenheiten auskennen. Zum Beispiel Arbeitsrecht, Steuerrecht und weiteren Themen. Wir haben bereits zehn Jahre Erfahrung mit verteilten Softwareentwicklern und es ist nicht gut, wenn bei einer großen Entwickleranzahl die Ressourcenplanung weiterhin in Graz koordiniert wird.
Haris Bećirević	Danke, dann hängt es zusätzlich von der Anzahl der EntwicklerInnen am Standort ab. Die nächste Frage lautet: *„Bietet Scrum Vorteile für die Projektumsetzung mit verteilten SoftwareentwicklerInnen, da hierfür hoher Kommunikationsaufwand betrieben wird? Wie lautet die Begründung?*“. Scrum beschreibt ein Regelwerk, welches von einem hohen Kommunikationsaufwand geprägt ist. Ich meine dabei die Scrum Besprechungen, wie zum Beispiel Sprint Plannings, Sprint Retrospektiven, Dailies, etc. In der Theorie der verteilten Softwareentwicklung wird das Problem beschrieben, dass unzureichende Kommunikation eine Herausforderung darstellt. Schafft Scrum eine Abhilfe hinsichtlich der Kommunikation?
Stefan Schnuderl	Ich denke die Kommunikation wird über die Jahre immer besser. Vor allem durch COVID-19 werden Videokonferenzen immer mehr toleriert und ebenso lokal in Österreich üblicher. In Microsoft Teams kannst du während der Projektumsetzung verteilte Entwickler direkt anrufen, um Themen zeitnah abzustimmen. Ich sehe den Vorteil im Kommunikationsframework von Scrum, dass einzelne Termine fix vorgegeben und der Zweck dieser Besprechungen beschrieben wird. Es gibt zum Beispiel die Sprint Retrospektiven, in der Punkte angesprochen werden, die in der Projektumsetzung letztens nicht gut funktionieren haben. Diese Punkte werden in Scrum vorab abgefangen, damit es später nicht sofort zur Eskalation kommt. Durch die Vorgabe einer derartigen Besprechung in Scrum fallen offene Probleme nicht einfach unter den Tisch und können zeitnah gelöst werden. Das „Daily“ halte ich für ein nützliches Werkzeug in der Projektumsetzung. Aus dem Grund, weil man jeden Kollegen mindestens einmal am Tag sieht und das Gefühl entsteht man arbeitet „nah“ miteinander. Wir haben ebenso dediziert vorgegeben, dass Webkameras eingeschaltet werden müssen während „Dailies“ – das stärkt die Kommunikation. Es setzt eine gewisse Reife der Entwickler voraus, dass diese auch wirklich verteilt arbeiten können. Sie müssen das Homeoffice als wirkliche Arbeit ansehen, da wir während der Projektumsetzung die Tätigkeiten nicht kontrollieren können und wollen.

Haris Bećirević	Ok, danke. Du siehst also einen Vorteil in den vordefinierten Serienterminen für die Kommunikation in verteilten Softwareprojekten im Regelwerk Scrum.
Stefan Schnuderl	Mhm, ja.
Haris Bećirević	Gut. Die nächste Frage lautet: *„Sollen ProjektleiterInnen den verteilten SoftwareentwicklerInnen wöchentlich den Soll-/Ist-Vergleich im verteilten Softwareprojekt darlegen, um daraus folgend die aktuelle Projektsituation und notwendige Maßnahmen besser zu vermitteln? Wie lautet die Begründung?“* Hier geht es darum, dass verteilte SoftwareentwicklerInnen kein Gefühl für die aktuelle Projektsituation besitzen – sei es abhängig von der angespannten Stimmung, budgetären Situation oder auch Nichterreichung von Deadlines. Würde es helfen den verteilten SoftwareentwicklerInnen einen Soll-/Ist-Vergleich wöchentlich zu präsentieren? Auf der einen Seite hätten wir die Planung und auf der anderen Seite den Fortschrittsgrad? Durch diese wöchentliche Besprechung könnten wir die SoftwareentwicklerInnen daraufhin sensibilisieren. Was haltest du von der Idee?
Stefan Schnuderl	Grundsätzlich finde ich einen Serientermin hinsichtlich Soll-/Ist-Vergleich für sinnvoll. Ob dieser Termin wöchentlich stattfinden muss, sei dahingestellt – ich denke es reicht einmal in zwei Wochen. Im Kommunikationsregelwerk sollte dezidiert Platz gemacht werden für eine Besprechung in Bezug auf den Soll-/Ist-Vergleich, um die aktuelle Projektsituation zu vermitteln. Darauffolgend sollte man auch mitverfolgen, wie die verteilten Softwareentwickler darauf reagieren. Aus diesem Grund plädiere ich stark für „Dailies“, da man recht schnell erkennt, ob sich die Beteiligten noch für das Projekt interessieren. In einem Serientermin, wo der Soll-/Ist-Vergleich dargestellt wird, spricht man Maßnahmen an und erkennt gleich, ob sich die Entwickler überhaupt dafür interessieren. Wenn solche Themen, wie die Projektsituation oder Maßnahmen, gar nicht angesprochen werden, dann tut man sich schwer zu erkennen, ob das nötige „Commitment“ zum Projekt noch vorhanden ist. Falls von den verteilten Softwareentwicklern gar nie ein Feedback zu den vermittelten Maßnahmen kommt, dann sieht es eher so aus, als ob sich kaum jemand für das Projekt interessiert.
Haris Bećirević	Mhm, ok. Du meinst, dass die Vermittlung des Soll-/Ist-Vergleichs in einem fixen Serientermin notwendig sind und man dadurch ziemlich schnell erkennt, ob die verteilten SoftwareentwicklerInnen noch am Projekt interessiert sind?
Stefan Schnuderl	Genau. Ich als Projektleiter würde die Projektsituation immer transparent vermitteln und Gegenfragen mitverfolgen. Gerade die Softwareentwicklung verlangt, dass alle Beteiligten mitdenken und sich nicht „babysitten“ lassen. Die Entwicklung muss sich für das Gesamtbild interessieren und mit der

	aktuellen Projektsituation und Maßnahmen umgehen können. Ansonsten stellt sich die Frage, ob der Entwickler vielleicht fehl am Platz ist.
Haris Bećirević	Super, danke für die umfangreiche Antwort. Nun kommen wir zur letzten Frage: *„Sollen die Lernerfahrungen von verteilten SoftwareentwicklerInnen in einer organisationszentralen Wissensdatenbank dokumentiert werden und sollen verteilte SoftwareentwicklerInnen dabei aktiv mitwirken? Wie lautet die Begründung?“* In Scrum gibt es zum Beispiel die Sprint Retrospektiven, in denen aufgetauchte Probleme während des Sprints angesprochen werden. Die Lösung dieser Probleme kann als Lernerfahrung betrachtet werden, aber sollen diese auch aktiv von den verteilten SoftwareentwicklerInnen in eine zentrale Wissensdatenbank strukturiert dokumentiert werden?
Stefan Schnuderl	Mhm. Einerseits finde ich es für wichtig, dass Lernerfahrungen oder Lösungswege, gemeinsam aufgearbeitet und dokumentiert werden müssen. Andererseits denke ich, dass derartige Wissensdatenbanken im Nachgang selten genutzt oder gelesen werden. Insofern ist es dieser „Journaling“-Effekt, bei dem Entwickler Freiraum gegeben wird, um folgende Fragen für sich selbst zu beantworten: „Welche Lernerfahrungen habe ich gemacht? Wie habe ich mich organisiert?“, sehr wichtig. Die Aufarbeitung sollte im Team erfolgen und jede Person nimmt Themen für sich selbst mit. Die Wissensdatenbank wird im späteren Verlauf eher nicht genutzt, aber der eigentliche Lerneffekt geschieht in der gemeinsamen Aufarbeitung.
Haris Bećirević	Ok, gut. Du meinst, dass die Wissensdatenbank in drei, vier Jahren kaum jemand lesen wird, aber der Lerneffekt wird durch die gemeinsame Aufarbeitung verstärkt.
Stefan Schnuderl	Mhm, ja genau.
Haris Bećirević	Danke, das war schon die letzte Schlüsselfrage. Sobald ich mein Masterstudium abschließe, erhält ihr eine gebundene Version meiner Masterarbeit. Das wird ungefähr im Februar 2022 der Fall sein. Vielen Dank für deine Zeit, Stefan!
Stefan Schnuderl	Sehr gerne! Danke dir!

ANHANG B - 2. Anhang

Person	Aussage
Haris Bećirević	Hallo Harald! Danke für deine Zeit. Bei meiner Masterarbeit geht es um das Thema *„Ein Prozessmodell zur Einführung und Steuerung einer verteilten Softwareentwicklung für die Projektabwicklung am Beispiel eines österreichischen Individualsoftwaredienstleisters“*. Es geht um ein Prozessmodell, welches die Einführung von verteilten SoftwareentwicklerInnen beschreibt, sowie die Steuerung von verteilten Softwareprojekten – es geht um das ganze Konzept. Es handelt sich hierbei nicht nur um einen geografischen Standort, wie zum Beispiel Weißrussland bei Guid.New, sondern um mehrere verteilten Standorte.
Harald Schaffernak	Ja, ok.
Haris Bećirević	Ein weiterer Teil dieser Einleitung ist die Erlaubnisabfrage von folgenden Punkten: Dieses Interview hinsichtlich Video und Ton aufzunehmen und zu archivieren Transkription des Interviews und Auswertung in der Masterarbeit Verwendung deines Namens sowie akademischen Titels in der Masterarbeit Vorstellung deiner Person in der Masterarbeit
Harald Schaffernak	Ja, passt.
Haris Bećirević	Super, danke. Die Forschungsfrage lautet: *„Welche Prozesse ermöglichen die Einführung und Steuerung einer verteilten Softwareentwicklung für die Projektabwicklung und wie sind diese in ein Prozessmodell überführbar?“* Ich habe Prozesse mittels BPMN konzipiert und daraus Prozessmodelle gebildet. Diese Prozessmodelle besitzen bestimmte Eigenschaften. Die Fragen im Hauptteil dieses Interviews enthalten diese Eigenschaften und genau um diese geht es hauptsächlich im Interview. Ich möchte diese Eigenschaften mit Experten, wie dir, validieren. Hast du die Fragestellungen und Agenda vorher durchgelesen? Können wir mit dem Hauptteil dieses Interviews starten?
Harald Schaffernak	Ja, habe ich mir vorher kurz durchgelesen. Wir können starten.

Haris Bećirević	Die erste Frage lautet: *„Sollen bei der Beschaffung von verteilten SoftwareentwicklerInnen externe PersonaldienstleisterInnen hinsichtlich spezifischen Know-hows engagiert werden? Wie lautet die Begründung?“* Die Frage lautet konkret, ob es Sinn macht, je Region, externe PersonaldienstleisterInnen zu beschäftigen. Sei es in Weißrussland oder auch, wie bei meinem aktuellen Arbeitgeber, Bosnien und Herzegowina. Jeder externe Personaldienstleister besitzt je Land das jeweilige Know-how für die Beschaffung von Personal. Neben dem Know-how besitzt der externe Personaldienstleister ebenso die „Connections“ am Personalmarkt. Was denkst du darüber?
Harald Schaffernak	Das ist, finde ich, eine aufgelegte Frage. Natürlich wird ein externer Personaldienstleister herangezogen, weil du gerade am Anfang keinen Ruf im jeweiligen Land besitzt. Am Anfang weiß man nicht, wie man Menschen ansprechen soll und wie die kulturellen Unterschiede sind. Ohne externen Personaldienstleister wird man initial kein Personal finden. Außer wenn das Unternehmen bekannter wird im Laufe der Zeit – dann kann man darüber nachdenken, die Personalbeschaffung ohne externen Personaldienstleister durchzuführen. Zusätzlich muss ich sagen, man sollte im eigenen Kerngeschäft bleiben – also in unserem Fall die Softwareentwicklung.
Haris Bećirević	Also würdest du die Personalbeschaffung im Laufe der Zeit im jeweiligen Land selbst übernehmen? Also ohne externe PersonaldienstleisterInnen und dafür je Land eine eigene Personalabteilung aufstellen?
Harald Schaffernak	Eine eigene Personalabteilung je Land würde ich nicht aufstellen. Genau aus dem Grund, weil ich in meinem Kerngeschäft bleiben möchte.
Haris Bećirević	Passt, nächste Frage: *„Sollen in der Aufbauorganisation die Vorgesetzten der verteilten SoftwareentwicklerInnen am gleichen geografischen Standort arbeiten und mit diesen die Ressourcenplanung koordiniert werden? Wie lautet die Begründung?“* Aus meiner Erfahrung bei der Guid.New GmbH weiß ich, dass die Ressourcenplanung allein von unserem Standort in Graz durchgeführt wurde – ohne einem Miteinbeziehen von verteilten SoftwareentwicklerInnen. Würde es Sinn machen, je geografischen Standort, einen eigenen Vorgesetzten aus den Reihen der verteilten SoftwareentwicklerInnen zu ernennen? Diesen würden wir in der Ressourcenplanung miteinbeziehen und dieser Person kommunizieren, wie viele Personalressourcen wir in welchem Zeitraum benötigen.
Harald Schaffernak	Das haben wir ganz am Anfang versucht. Wir hatten zwei verteilte Softwareentwickler im Fokus für die Aufgabe als Vorgesetzten, aber so weit ist es nie gekommen. Das Problem war, dass die Softwareentwickler am weißrussischen Standort diese Verantwortung nie bei sich selbst gesehen

	haben, und dadurch ist die Steuerung und Ressourcenplanung in Österreich geblieben. Den Begriff „Aussteuern“ finde ich nicht gut, weil dieser Begriff „Geringschätzung von irgendjemandem“ enthält. Ich würde es aber trotzdem gut finden, wenn eine Person in der Aufbauorganisation die Rolle als Vorgesetzten übernehmen würde. Diese Person würde uns in Österreich aktiv hinsichtlich Personalplanung, Ressourcenplanung etc. unterstützen. Das würde mehr Sinn machen, da dieser Vorgesetzte verknüpft mitdenken würde und nicht einfach die Implementierungen „runterprogrammiert“. Ansonsten liegen alle Überlegungen abseits der Implementierungen in Österreich und nicht am Standort in Weißrussland.
Haris Bećirević	Also ist es für dich wünschenswert eine Person in der Aufbauorganisation am geografischen Standort der verteilten SoftwareentwicklerInnen zu haben, welche den organisatorischen Aufwand abnimmt und als Ansprechperson dient?
Harald Schaffernak	Hm, ob das absolut beantwortbar ist – das weiß ich derzeit nicht. Es ist wünschenswert, aber findet man eine Person mit der nötigen Qualifikation als Vorgesetzten in der Aufbauorganisation? Es hat praktisch auch so funktioniert, dass die Ressourcenplanung in Österreich liegt und in Weißrussland nur die Ausführung erfolgt. Das wirft ein schlechtes Licht auf den Standort in Weißrussland, aber es hat funktioniert. Im Endeffekt hat jeder Prozess Vor- und Nachteile und für einen muss man sich entscheiden. Ob, der eine Prozess besser ist als der andere, das kann man nicht sofort beantworten.
Haris Bećirević	Ok. Ist also nicht eindeutig beantwortbar, aber wir haben in jedem Prozess die jeweiligen Vor- und Nachteile. Die nächste Frage lautet: *„Bietet Scrum Vorteile für die Projektumsetzung mit verteilten SoftwareentwicklerInnen, da hierfür hoher Kommunikationsaufwand betrieben wird? Wie lautet die Begründung?“*. Scrum beschreibt ein Regelwerk, welches von einem hohen Kommunikationsaufwand geprägt ist. Ich meine dabei die Scrum Besprechungen, wie zum Beispiel Sprint Plannings, Sprint Retrospektiven, Dailies, etc. In der Theorie der verteilten Softwareentwicklung wird das Problem beschrieben, dass unzureichende Kommunikation eine Herausforderung darstellt. Schafft Scrum eine Abhilfe hinsichtlich der Kommunikation?
Harald Schaffernak	Im Vergleich zu was?
Haris Bećirević	Zum Beispiel im Vergleich zu Wasserfallmodellen. Der Kommunikationsaufwand ist bei Scrum viel höher, da die Anzahl der Besprechungen und Abstimmungen höher ist als bei Wasserfallmodellen. Bei

	Wasserfall hast du zum Beispiel vielleicht in der Projektumsetzung nur ein Jour Fixe wöchentlich. Durch den hohen Kommunikationsaufwand entstehen weniger Probleme in der Projektumsetzung.
Harald Schaffernak	Das läuft dann wieder auf den Vergleich: „Was ist besser? Scrum oder Wasserfall?“ Scrum bietet in unserem Fall einen sozialen Vorteil, da sich verteilte Softwareentwickler nicht immer persönlich sehen. In der Theorie sind User Stories fertig definiert und es sind keine zusätzlichen Abstimmungen notwendig. Aber bei Scrum ist es so, dass durch den hohen Kommunikationsaufwand eine soziale Bindung innerhalb Teams entsteht. Das ist gut, da sich das Team näher kennenlernt und die Personen sich dadurch im Regelfall eher mögen. Ein Team, welches sich mag, arbeitet in der Projektumsetzung besser – hier sehe ich eher den Vorteil von Scrum.
Haris Bećirević	Mhm. Du siehst also den Vorteil eher in der sozialen Bindung innerhalb des Teams infolge von Scrum?
Harald Schaffernak	Ja, genau. Ich würde aber nicht sagen, dass Scrum generell besser ist als Wasserfallmodell oder V-Modell. In vielen Fällen kann es durchaus sein, dass das Wasserfallmodell in bestimmten Projekten besser funktioniert als im Vergleich zu Scrum. Als Bedingung gilt hier, dass das Projekt sauber aufgesetzt wird und die Projektumsetzung ordentlich erfolgt. Das ist also, meiner Meinung nach, nicht absolut beantwortbar. Ein Unterschied, der mir persönlich ins Auge sticht, ist, dass bei Scrum das Team viel mehr miteinander reden muss und sich bei der Projektumsetzung selbst organisieren muss. Durch diesen hohen Kommunikationsaufwand lernt sich das geografisch verteilte Softwareprojektteam besser kennen und dies kann in Scrum ein Vorteil sein im Vergleich zu anderen Modellen. Bei anderen Modellen wird zum Beispiel der hohe Kommunikationsaufwand nicht gefordert.
Haris Bećirević	Passt, verstehe – danke. Die nächste Frage lautet: *„Sollen ProjektleiterInnen den verteilten SoftwareentwicklerInnen wöchentlich den Soll-/Ist-Vergleich im verteilten Softwareprojekt darlegen, um daraus folgend die aktuelle Projektsituation und notwendige Maßnahmen besser zu vermitteln? Wie lautet die Begründung?“* Hier geht es darum, dass verteilte SoftwareentwicklerInnen kein Gefühl für die aktuelle Projektsituation besitzen – sei es abhängig von der angespannten Stimmung, budgetären Situation oder auch Nichterreichung von Deadlines. Würde es helfen den verteilten SoftwareentwicklerInnen einen Soll-/Ist-Vergleich wöchentlich zu präsentieren? Auf der einen Seite hätten wir die Planung und auf der anderen Seite den Fortschrittsgrad? Durch diese wöchentliche Besprechung könnten wir die SoftwareentwicklerInnen daraufhin sensibilisieren. Was haltest du von der Idee?

Harald Schaffernak	Wo liegt der Unterschied zwischen verteilt und nicht verteilt?
Haris Bećirević	Würdest du es generell bei Softwareprojekten raten?
Harald Schaffernak	Es macht durchaus Sinn unter Einbeziehung von allen Personen, die relevanten Informationen aus dem Soll-/Ist-Vergleich transparent zu liefern. Natürlich besteht die Gefahr, dass bestimmte Personen in einer derartigen Besprechung gelangweilt werden und nicht richtig zuhören. Grundsätzlich erkennt man in den ersten beiden Terminen, ob die Personen am Status der Projektsituation interessiert sind oder nicht. Ich bin für eine transparente Kommunikation der Maßnahmen, da hierdurch die Softwareentwickler sich mehr miteingebunden fühlen. Auf diese Weise fühlen sich Entwickler als Teil eines Teams und nicht nur als temporärer Ausführer der Implementierung.
Haris Bećirević	Ok, verstehe. Du würdest also die transparente Kommunikation für Softwareprojekte generell empfehlen?
Harald Schaffernak	Ja, würde ich generell empfehlen. Ich sehe hier keinen Unterschied zwischen verteilten und nicht verteilten Softwareprojekten.
Haris Bećirević	Ok. Die letzte Frage lautet: *„Sollen die Lernerfahrungen von verteilten SoftwareentwicklerInnen in einer organisationszentralen Wissensdatenbank dokumentiert werden und sollen verteilte SoftwareentwicklerInnen dabei aktiv mitwirken? Wie lautet die Begründung?“* In Scrum gibt es zum Beispiel die Sprint Retrospektiven, in denen aufgetauchte Probleme während des Sprints angesprochen werden. Die Lösung dieser Probleme kann als Lernerfahrung betrachtet werden, aber sollen diese auch aktiv von den verteilten SoftwareentwicklerInnen in eine zentrale Wissensdatenbank strukturiert dokumentiert werden?
Harald Schaffernak	Was zählt als Wissensdatenbank? Microsoft Teams? Wissen wird generell schnell erzeugt. Ob das erzeugte Wissen gut oder schlecht ist, liegt im Auge des Betrachters. Wenn jemand zum Beispiel in Microsoft Teams, im sogenannten „Offtopic“-Kanal, eine Information hinterlässt, dann ist das eigentlich auch Wissen. Nur zählt das halt meistens nicht als Lernerfahrung.
Haris Bećirević	Mit Dokumentation in der Wissensdatenbank ist hier eher ein strukturiertes Vorgehen hinsichtlich Speicherung von Lernerfahrungen gedacht. Hierfür könnte man zum Beispiel ein WIKI in Azure DevOps nutzen. Kann man in Microsoft Teams überhaupt Lernerfahrungen strukturiert dokumentieren?
Harald Schaffernak	Man kann zum Beispiel je Projekt einen eigenen Kanal erzeugen und innerhalb dieses Kanals „Threads“ erzeugen. Auf diese Weise erhält man eine Wissensdatenbank, wobei nicht immer neue Lernerfahrungen beschrieben werden. Eine Wissensdatenbank muss nicht immer ein WIKI

	sein. Damit wir jetzt wieder zurück zur Frage kommen: Ja, es wäre ziemlich einseitig, wenn immer nur der „Product Owner“ die Wissensdatenbank pflegen würde. Die Softwareentwickler sollen in der Wissensdatenbank aktiv mitwirken und ebenso ihre Lernerfahrungen dokumentieren. Ansonsten wäre die Wissensdatenbank nutzlos, weil sie keiner lesen würde.
Haris Bećirević	Gut, danke für deine Antworten. Möchtest du die Masterarbeit bei Fertigstellung erhalten?
Harald Schaffernak	Die PDF-Datei reicht mir.

ANHANG C - 3. Anhang

Person	Aussage
Haris Bećirević	Hallo Marina! Danke für deine Zeit. Bei meiner Masterarbeit geht es um das Thema *„Ein Prozessmodell zur Einführung und Steuerung einer verteilten Softwareentwicklung für die Projektabwicklung am Beispiel eines österreichischen Individualsoftwaredienstleisters“*. Es geht um ein Prozessmodell, welches die Einführung von verteilten SoftwareentwicklerInnen beschreibt, sowie die Steuerung von verteilten Softwareprojekten – es geht um das ganze Konzept. Die Forschungsfrage lautet: *„Welche Prozesse ermöglichen die Einführung und Steuerung einer verteilten Softwareentwicklung für die Projektabwicklung und wie sind diese in ein Prozessmodell überführbar?“* Ich habe Prozesse mittels BPMN konzipiert und daraus Prozessmodelle gebildet. Diese Prozessmodelle besitzen bestimmte Eigenschaften. Die Fragen im Hauptteil dieses Interviews enthalten diese Eigenschaften und genau um diese geht es hauptsächlich im Interview. Ich möchte diese Eigenschaften mit Experten, wie dir, validieren.
Marina Opferkuch	Ok, alles gut.
Haris Bećirević	Passt. Bevor ich das Interview starte, möchte ich noch folgende Erlaubnisabfrage durchführen: Dieses Interview hinsichtlich Video und Ton aufzunehmen und zu archivieren Transkription des Interviews und Auswertung in der Masterarbeit Verwendung deines Namens sowie akademischen Titels in der Masterarbeit Vorstellung deiner Person in der Masterarbeit
Marina Opferkuch	Ja, passt alles.
Haris Bećirević	Die erste Frage lautet: *„Sollen bei der Beschaffung von verteilten SoftwareentwicklerInnen externe PersonaldienstleisterInnen hinsichtlich spezifischen Know-hows engagiert werden? Wie lautet die Begründung?“* Die Frage lautet konkret, ob es Sinn macht, je Region, externe PersonaldienstleisterInnen zu beschäftigen. Sei es in Weißrussland oder auch, wie bei meinem aktuellen Arbeitgeber, Bosnien und Herzegowina. Jeder externe Personaldienstleister besitzt, je Land, das jeweilige Know-how für die Beschaffung von Personal. Neben dem Know-how besitzt der externe

	Personaldienstleister ebenso die „Connections“ am Personalmarkt. Was denkst du darüber?
Marina Opferkuch	In der Guid.New GmbH war es sehr wichtig externe PersonaldienstleisterInnen zu beschäftigen. Für die Beschaffung von Personal ist es generell wichtig derartige Personalexperten anzuheuern, da auch durch die Etablierung einer eigenen Arbeitgebermarke es heutzutage nicht mehr möglich ist, externe Personaldienstleister außen vor zu lassen. Als Arbeitgeber muss man für die Beschaffung von Personal möglichst viele Plattformen nutzen, wie zum Beispiel auch die Mund-zu-Mund-Propaganda. Aufgrund dessen ist es enorm wichtig in einem fremden Land einen externen Personaldienstleister zu engagieren, da es initial dort noch schwieriger ist über Mund-zu-Mund-Propaganda potenzielles Personal zu erreichen. Im Laufe der Jahre sammelt das Unternehmen Erfahrung mit welchen externen Personaldienstleistern die Vermittlung am geeignetsten funktioniert. Zudem muss ich als Unternehmen mein Arbeitgeberimage positiv entwickeln, damit ich auch neues potenzielles Personal anziehe. Ein wesentlicher Punkt ist ebenso das Aufbauen eines Netzwerks mit mehreren externen Personaldienstleistern, da man hierdurch ein breiteres Spektrum an möglichen verteilten Softwareentwickler erhält.
Haris Bećirević	Also würdest du vorschlagen, je Land einen eigenen externen Personaldienstleister zu engagieren?
Marina Opferkuch	Tendenziell ja! Je Land oder Stadt gibt es Unterschiede hinsichtlich der Beschaffung von Personal. In den meisten Städten sind zum Beispiel die Universitäten die Hauptquelle hinsichtlich potenziellen Personals. In anderen Städten haben sich die geeignetsten Softwareentwickler diese Fachexpertise selbst beigebracht. Ein weiterer Knackpunkt ist der kulturelle Hintergrund je Land: Gewisse Regionen bevorzugen mehr Arbeitsstunden und Auslastung, andere wiederum bevorzugen ein „9 to 5“-Modell.
Haris Bećirević	Ich habe gar nicht gewusst, dass es Unterschiede zwischen den Städten je Land gibt – danke für die Info! Passt, nächste Frage: *„Sollen in der Aufbauorganisation die Vorgesetzten der verteilten SoftwareentwicklerInnen am gleichen geografischen Standort arbeiten und mit diesen die Ressourcenplanung koordiniert werden? Wie lautet die Begründung?“* Aus meiner Erfahrung bei der Guid.New GmbH weiß ich, dass die Ressourcenplanung allein von unserem Standort in Graz durchgeführt wurde – ohne einem Miteinbeziehen von verteilten SoftwareentwicklerInnen. Würde es Sinn machen, je geografischen Standort, einen eigenen Vorgesetzten aus den Reihen der verteilten SoftwareentwicklerInnen zu ernennen? Diesen würden wir in der Ressourcenplanung miteinbeziehen und

	dieser Person kommunizieren, wie viele Personalressourcen wir in welchem Zeitraum benötigen.
Marina Opferkuch	Hier muss man basierend auf die Ausgangslage differenzieren. Wenn das verteilte Softwareprojektteam nicht an einem gemeinsamen Standort arbeitet, stelle ich mir es schwierig vor einen Vorgesetzten zu ernennen. Falls die verteilten Softwareprojektleiter an einem gemeinsamen Standort arbeiten, dann macht es meiner Meinung schon Sinn einen Vorgesetzten zu bestimmen. Dieser Vorgesetzte muss sich nicht fachlich auskennen, aber sollte auf jeden Fall ein Gespür für disziplinarische Tätigkeiten besitzen. Die Person sollte nicht nur die Ressourcenplanung durchführen, sondern sollte auch aktiv im Entwicklungsteam mitwirken. Ansonsten besteht die Gefahr, dass keine „Verbindung“ zwischen Projektleitung in Österreich und Entwicklungsteam am weißrussischen Standort entsteht.
Haris Bećirević	Ist ein Vorgesetzter also nicht nötig, wenn die Softwareentwickler nicht am gleichen geografischen Standort arbeiten?
Marina Opferkuch	In der Softwarebranche geht es in die Richtung, dass man als Mitarbeiter innerhalb der Aufbauorganisation immer mehr Tätigkeiten übernehmen kann. Aus Personalentwicklungssicht muss man den verteilten Softwareentwicklern Zukunftsaussichten geben, wie zum Beispiel die Rolle als Vorgesetzten oder die Verantwortung über die Ressourcenplanung. Also macht es auch bei verteilten Standorten Sinn einen Vorgesetzten zu ernennen.
Haris Bećirević	Ok, danke. Die nächste Frage lautet: „*Bietet Scrum Vorteile für die Projektumsetzung mit verteilten SoftwareentwicklerInnen, da hierfür hoher Kommunikationsaufwand betrieben wird? Wie lautet die Begründung?“.* Scrum beschreibt ein Regelwerk, welches von einem hohen Kommunikationsaufwand geprägt ist. Ich meine dabei die Scrum Besprechungen, wie zum Beispiel Sprint Plannings, Sprint Retrospektiven, Dailies, etc. In der Theorie der verteilten Softwareentwicklung wird das Problem beschrieben, dass unzureichende Kommunikation eine Herausforderung darstellt. Schafft Scrum eine Abhilfe hinsichtlich der Kommunikation?
Marina Opferkuch	Ich glaube ein Prozessmodell mit einem hohen Kommunikationsaufwand ist sehr wichtig und dabei kann Scrum eine große Rolle spielen. Vor allem die dedizierte Funktion des Scrum Masters unterstreicht die Relevanz der Kommunikation in Scrum. Diese Person ist in allen Besprechungen dabei und kontrolliert laufend, ob die Kommunikation ordnungsgemäß abläuft. Wir wissen alle, welcher wirtschaftlicher Druck auf den Projektleitern oder Product Owner liegt: Auf der einen Seite sind die Kunden und auf der anderen Seite die Entwickler. Projektleiter oder Product Owner besitzen einen gänzlich anderen Fokus in der Projektumsetzung und dabei können Themen, wie die

	Kommunikation, auf der Strecke bleiben. Hierbei können Scrum Master die Projektleitung in der Projektumsetzung hinsichtlich Kommunikation unterstützen. Das Sammeln von Lernerfahrungen kann beispielsweise vom Scrum Master übernommen werden, damit die Projektleitung sich beispielsweise auf den wirtschaftlichen Erfolg des Projektes konzentriert. Durch die explizite Zuweisung einer derartigen Rolle kann verhindert werden, dass während der Projektumsetzung Informationen verloren gehen.
Haris Bećirević	Du meinst, dass die vordefinierten Serientermine im Kommunikationsregelwerk von Scrum in verteilten Softwareprojekten für Abhilfe sorgen können?
Marina Opferkuch	Genau. Nicht nur die Sprint Plannings unterstützen bei der Projektumsetzung, sondern auch die Sprint Retrospektiven. Der große Vorteil von Scrum ist die Trennung zwischen Product Owner und dem Scrum Master, welcher während der Projektumsetzung als Kommunikationsträger dient. Der Scrum Master muss sich nicht fachlich einmischen oder disziplinär mitwirken – diese Person dient hauptsächlich als Moderator in Besprechungen. Diese Trennung zwischen Projektleitung und Kommunikationsträger führt einen positiven Effekt in der Projektumsetzung von verteilten Softwareprojekten herbei.
Haris Bećirević	Danke! Die nächste Frage lautet: *„Sollen ProjektleiterInnen den verteilten SoftwareentwicklerInnen wöchentlich den Soll-/Ist-Vergleich im verteilten Softwareprojekt darlegen, um daraus folgend die aktuelle Projektsituation und notwendige Maßnahmen besser zu vermitteln? Wie lautet die Begründung?“* Hier geht es darum, dass verteilte SoftwareentwicklerInnen kein Gefühl für die aktuelle Projektsituation besitzen – sei es abhängig von der angespannten Stimmung, budgetären Situation oder auch Nichterreichung von Deadlines. Würde es helfen den verteilten SoftwareentwicklerInnen einen Soll-/Ist-Vergleich wöchentlich zu präsentieren? Auf der einen Seite hätten wir die Planung und auf der anderen Seite den Fortschrittsgrad? Durch diese wöchentliche Besprechung könnten wir die SoftwareentwicklerInnen daraufhin sensibilisieren. Was haltest du von der Idee?
Marina Opferkuch	Diese Frage kann ich nicht pauschal mit „Ja“ oder „Nein“ beantworten – ich tendiere aber eher zu „Nein“. Die wöchentliche Aufarbeitung eines Soll-/Ist-Vergleichs ist sehr aufwändig und ein weiterer Grund ist die Tatsache, dass wir uns in einem Arbeitnehmermarkt befinden. In solch einem Markt, wo es ohnehin schwierig ist verteilte Softwareentwickler zu finden, sollte man diese nicht mit Soll-/Ist-Vergleiche „bombardieren“. In jedem Soll-/Ist-Vergleich schwirrt im Hintergrund die Botschaft: „Du warst zu langsam. Das war zu wenig. etc.“. Der Projektleiter soll den Überblick über die Projektsituation besitzen und individuell Maßnahmen setzen, falls ein

	verteilter Softwareentwickler die geforderte Leistung nicht erbringt. Zudem sollten die Soll-/Ist-Vergleiche in einem größeren Zeitintervall stattfinden – zum Beispiel einmal in zwei Wochen. Auf jeden Fall soll die Zufriedenheit der verteilten Softwareentwickler durch Soll-/Ist-Vergleiche nicht geschmälert werden. Hierfür muss der Projektleiter die „goldene Mitte“ finden und durch eine saubere Vermittlung der Projektsituation eine Verschlechterung der Atmosphäre vermeiden. Auf der anderen Seite ist es dennoch wichtig individuelle Maßnahmen zu treffen und gemeinsam mit den jeweiligen Softwareentwicklern Verbesserungskonzepte zu erarbeiten. Als Fazit kann ich sagen: Ein Soll-/Ist-Vergleich kann in einer anderen Regelmäßigkeit hilfreich sein und Maßnahmen müssen individuell gesetzt werden.
Haris Bećirević	Du betrachtest die Soll-/Ist-Vergleiche also erst für sinnvoll, wenn diese in einem größeren Zeitintervall stattfinden?
Marina Opferkuch	Ja, so ist das gemeint. Die Soll-/Ist-Vergleiche sollten zudem nicht zu feingranular sein. Einem „Frontend“-Entwickler interessiert beispielsweise die detaillierte wirtschaftliche Projektsituation des Gesamtprojekts nicht. Dennoch sollte er sich für die grobe Projektsituation interessieren und verstehen wofür einzelne Maßnahmen dienen.
Haris Bećirević	Perfekt! Nun kommen wir zur letzten Frage: *„Sollen die Lernerfahrungen von verteilten SoftwareentwicklerInnen in einer organisationszentralen Wissensdatenbank dokumentiert werden und sollen verteilte SoftwareentwicklerInnen dabei aktiv mitwirken? Wie lautet die Begründung?“* In Scrum gibt es zum Beispiel die Sprint Retrospektiven, in denen aufgetauchte Probleme während des Sprints angesprochen werden. Die Lösung dieser Probleme kann als Lernerfahrung betrachtet werden, aber sollen diese auch aktiv von den verteilten SoftwareentwicklerInnen in eine zentrale Wissensdatenbank strukturiert dokumentiert werden?
Marina Opferkuch	Ich finde die Betrachtung des Projektabschlusses im Rahmen der verteilten Softwareentwicklung für sehr interessant! Das Sammeln von Lernerfahrungen während den Sprint Retrospektiven halte ich in verteilten Softwareprojekten für gut und wichtig. In der Absprache der Lernerfahrungen im Nachgang sehe ich allerdings keinen nachhaltigen Lerneffekt. In der verteilten Softwareentwicklung besteht das Problem, dass das kollegiale Miteinander fehlt und somit das „Untereinander helfen“ untertags nicht existiert. Aufgrund dessen denke ich, dass bei verteilten Softwareprojekten kleinere Lernerfahrungen komplett verloren gehen. Eine Wissensdatenbank je Projekt oder Organisation würde dabei helfen, dass auch kleinere Lernerfahrungen nicht in Vergessenheit geraten. Die aktive Mitwirkung je verteilten Softwareentwickler beim Pflegen der Wissensdatenbank sorgt für einen nachhaltigen Lerneffekt.

Haris Bećirević	Soll die aktive Mitwirkung deshalb kontrolliert werden?
Marina Opferkuch	Die verteilten Softwareentwickler sollen auf jeden Fall gefördert und motiviert werden die Wissensdatenbank langfristig zu pflegen. Die Pflege der Wissensdatenbank kostet Zeit und Geld, aber das Unternehmen muss den Mitarbeitern proaktiv ein Zeitkontigent dafür zur Verfügung stellen. Die aktive Dokumentation der Lernerfahrungen sorgt für einen nachhaltigen Lerneffekt im Vergleich zur „normalen“ Absprache der aufgetretenen Probleme im Projekt.
Haris Bećirević	Sehr gut, danke! Noch eine kleine Frage zu diesem Thema, bevor wir den Hauptteil abschließen. Denkst du irgendjemand wird diese Wissensdatenbank im Nachgang nutzen und lesen?
Marina Opferkuch	Hm. Das ist das „A und O“, dass die Wissensdatenbank auch genutzt wird. Vielleicht würde es helfen die Wissensdatenbank unternehmenszentral aufzustellen und nicht projektspezifisch. Die Qualität und Einfachheit der Beschreibungen hinsichtlich Lernerfahrungen tragen einen wesentlichen Beitrag zur Attraktivität der Wissensdatenbank bei. Eine qualitative Wissensdatenbank ist auch eine Voraussetzung für die Skalierbarkeit eines Unternehmens. Neu eingestellte, verteilte Softwareentwickler im „Junior“-Level benötigen solch eine Wissensdatenbank gerade am Anfang sehr oft und auf diese Weise können die verteilten Softwareprojektteams langfristig wachsen.
Haris Bećirević	Dankeschön, das war schon der Abschluss des Hauptteils. Sobald ich mein Masterstudium abschließe, erhältst du via E-Mail meine Masterarbeit als PDF. Zusätzlich bekommst du auch noch die Vektorgrafiken zum Prozessmodell. Das wird ungefähr im Februar 2022 der Fall sein. Vielen Dank für deine Zeit, Marina!
Marina Opferkuch	Ja, bitte und danke! Ich habe dich sehr gerne unterstützt.

ABKÜRZUNGSVERZEICHNIS

AG	Aktiengesellschaft
BMI	Bundesministerium für Inneres
BPD	Business Process Diagram
BPMN	Business Process Model and Notation
CD	Compact Disc
COVID-19	Corona Virus Disease 2019
GmbH	Gesellschaft mit beschränkter Haftung
IT	Informationstechnologie
PDF	Portable Document Format
RUP	Rational Unified Process
T&M	Time & Materials
XP	Xtreme Programming

ABBILDUNGSVERZEICHNIS

TABELLENVERZEICHNIS

LITERATURVERZEICHNIS

Allweyer, T. (2005). *Geschäftsprozessmanagement.* Bochum: W3L GmbH.

Allweyer, T. (2015). *BPMN 2.0 Business Process Model and Notation. Einführung in den Standard für die Geschäftsprozessmodellierung.* Kaiserslautern: BoD – Books on Demand.

Becker, J., Kugeler, M., & Rosemann, M. (2005). *Prozessmanagement - Ein Leitfaden zur prozessorientierten Organisationsgestaltung.* Berlin-Heidelberg: Springer Verlag.

Blum, N. (2010). *Erfolgsfaktor Inplacement: Neue Mitarbeiter systematisch und zielgerichtet integrieren.* Hamburg: Diplomica Verlag.

BMI, D. (Februar 2018). *Organisationshanduch des Bundesministerium des Innern, Bau und Heimat, Republik Deutschland.* Von https://www.orghandbuch.de/OHB/DE/ohb_pdf.pdf?__blob=publicationFile&v=29 abgerufen

Bortz, J. (1984). *Lehrbuch der empirischen Forschung für Sozialwissenschaftler.* Berlin: Springer-Verlag.

Brenner, D. (2014). *Onboarding: Als Führungskraft neue Mitarbeiter erfolgreich einarbeiten und integrieren.* Wiesbaden: Springer.

Carmel, E. (1999). *Global Software Teams - Collaborating Accross Borders and Time-Zones.* New Jersey: USA: Prentice Hall.

Casey, V., & Richardson, I. (2006). *Project Management within Virtual Software Teams.* Florianopolis: Brazil: IEEE Software.

DIN e.V. (2013). *Projektmanagement: Netzplantechnik und Projektmanagementsysteme.* Berlin: Beuth.

Dingsøyr, T., Dybå, T., & Moe, N. B. (2010). *Agile Software Development.* Berlin: Springer.

Drucker, P. F. (1977). *People and Performance: The Best of Peter Drucker on Management.* New York: Harper's College Press.

Engel, R. (2014). *Personalbeschaffung im Web 2.0.* Mittweida: Hochschule Mittweida - University of Applied Sciences.

Falls, M. (2004). *Inside the Minds - The Software Business: How Top Companies Design, Develop & Sell Successful Products & Applications.* Boston, Massachusetts: Aspatore.

Fischermanns, G. (2013). *Praxishandbuch Prozessmanagement.* Gießen: Dr. Götz Schmidt.

Gareis, R., & Stummer, M. (2007). *Prozesse & Projekte.* Wien: Manz Verlag.

Genau, L. (21. August 2020). *Scribbr*. Von https://www.scribbr.de/methodik/definition-experte/ abgerufen

Helfferich, C. (2019). *Leitfaden- und Experteninterviews - Methoden der empirischen Sozialforschung.* Wiesbaden: Springer-Verlag.

Herbsleb, J. D., & Moitra, D. (2001). *Global Software Development.* USA: IEEE Software.

Holden, N. J. (2002). *Cross-Cultural Management: A Knowledge Management Perspective.* Toronto: Pearson Education Canada.

Holtbrügge, D. (2017). *Personalmanagement.* Wiesbaden: Springer.

Hubert, W., & Pionczyk, A. (2006). *StartUp in den Job.* Berlin: VDE Verlag.

Humberg, D. (2002). *Projektcontrolling und IT-Leistungsverrechnung.* Siegen: Diplomarbeiten Agentur diplom.de.

International Standard Organization (ISO). (September 2012). *ISO.org*. Von ISO.org: https://www.iso.org/standard/50003.html abgerufen

International Standards Organization (ISO). (15. Juli 2013). *Beuth publishing DIN*. Von Beuth publishing DIN: https://www.beuth.de/de/norm-entwurf/din-iso-21500/187128218 abgerufen

Jimenez, M., Piattini, M., & Vizcaino, A. (2009). *Challenges and Improvements in Distributed Software: A System Review.* Albacete: Spain: Hindawi Publishing Corporation.

Korotia, Y. (31. Januar 2017). *Time-and-Materials vs Fixed Price: Which to Choose for Your Project?* Von https://medium.com/@Eugeniya/time-and-materials-vs-fixed-price-which-to-choose-for-your-project-11dc6adc758b abgerufen

Kuhrmann, M., & Linssen, O. (2013). *Welche Vorgehensmodelle nutzt Deutschland?* München: Technische Universität München, Fakultät für Informatik.

Lehtonen, I. (2009). *Communication Challenges in Agile Global Software Development.* Helsinki: Finland: University of Helsinki, Department of Computer Science, Faculty of Science.

Lindner, D. (01. Mai 2016). *Agiles Requirements Engineering*. Von https://agile-unternehmen.de/agiles-requirements-engineering-re/ abgerufen

Lindner-Lohmann, D., Lohmann, F., & Schirmer, U. (2016). *Personalmanagement.* Wiesbaden: Springer.

Marquardt, M. J., & Horvath, L. (2001). *Global Teams: How Top Multinationals Span Boundaries and Cultures with High- Speed Teamwork.* Palo Alto, CA: USA: Davies-Black.

Martin, R. C. (2002). *Agile Software Development. Principles, Patterns, and Practices.* Upper Saddle River: Prentice Hall.

Marxer, M. (21. 12 2016). *Projektabrechnung: Wie sich die Geschwindigkeit durch geeignete Software erhöhen lässt*. Von Handelsblatt: https://unternehmen.handelsblatt.com/projektabrechnung.html abgerufen

Mayring, P. A. (2015). *Qualitative Inhaltsanalyse: Grundlagen und Techniken.* Weinheim: Beltz Verlag.

Möller, T., & Dörrenberg, F. (2010). *Projektmanagement.* Berlin: Walter de Gruyter Verlag.

Nagura, M., & Iida, H. (2007). *Correlation Analysis for Distributed Development based on Configuration Management and Bug Report.* Japan: Nara Institute of Science Technology.

Nerdinger, F., Blickle, G., & Schaper, N. (2011). *Arbeits- und Organisationspsychologie.* Wiesbaden: Springer.

Olfert, K. (2012). *Personalwirtschaft.* Herne: Kiehl.

Opelt, A., Gloger, B., Pfarl, W., & Mittermayr, R. (2012). *Der agile Festpreis: Leitfaden für wirklich erfolgreiche IT-Projekt-Verträge.* München: Carl Hanser Verlag GmbH & Co. KG.

Oshri, I., Kotlarsky, J., & Willcocks, L. P. (2008). *Outsourcing Global Services: Knowledge, Innovation and Social Capital.* London: Palgrave Macmillan.

Permana, P. A. (2015). *Scrum Method Implementation in a Software Development Project Management.* Denpasar, Bali: International Journal of Advanced Computer Science and Applications.

Pham, A., & Pham, P.-V. (2011). *Scrum in Action: Agile Software Project Management and Development.* Boston, Massachusetts: Cengage Learning, Inc;.

Pongratz, H., Tramm, T., & Wilbers, K. (2009). *Prozessorientierte Wirtschaftsdidaktik und Einsatz von ERP-Systemen im kaufmännischen Unterricht.* Aachen: Shaker Verlag GmbH.

Prikladnicki, R., & Yamaguti, M. H. (2004). *Risk Management in Global Software Development: A Position Paper.* Rio Grande do Sul: Brazil: School of Computer Science, Pontifícia Universidade Católica do Rio Grande do Sul.

Prikladnicki, R., Audy, J. L., & Evaristo, R. (2006). *A Reference Model for Global Software Development: Findings from a Case Study.* Florianopolis: Brazil: IEEE Software.

Schatten, A., Demolsky, M., Winkler, D., Biffl, S., Gostischa-Franta, E., & Östreicher, T. (2010). *Best Practice Software-Engineering: Eine praxiserprobte Zusammenstellung von komponentenorientierten Konzepten, Methoden und Werkzeugen.* Wiesbaden: Springer Spektrum.

Schiele, M. (2009). *Standardsoftware vs. Individualsoftware*. Von http://www.dermarki.de/wissen/standard-individualsoftware-vor-nachteile.php abgerufen

Schirmeyer, P. (7. Mai 2020). *Softwareentwicklung in Zeiten von Corona - "Business as usual?"*. Von https://blogs.zeiss.com/digital-innovation/de/softwareentwicklung-in-zeiten-von-corona/ abgerufen

Schmelzer, H. J., & Sesselmann, W. (2008). *Geschäftsprozessmanagement.* München: Hanser Verlag.

Tripathi, V., & Goyal, A. K. (2014). *Agile Requirement Engineer : Roles and Responsibilities.* Indore, Indien: School of computer science and IT, Devi Ahilya University.

Vaher, L. (2003). *Potenziale und Risiken von Standard- und Individualsoftware.* Hannover: Institut für Wirtschaftsinformatik Universität Hannover.

Van Cauwenberghe, P. (2003). *Agile Fixed Price Projects Part 1: "The Price Is Right".* Sterrebeek: Belgien: Nayima bvba.

Wagner, R., & Grau, N. (2014). *Basiswissen Projektmanagement - Prozesse und Vorgehensmodelle.* Düsseldorf: Symposion Publishing GmbH.

Woodward, E., & Surdek, S. (2010). *A Practical Guide to Distributed Scrum.* Upper Saddle River: IBM Press.

Züllighoven, H. (2004). *Object-Oriented Construction Handbook: Developing Application-Oriented Software with the Tools & Materials Approach.* Burlington, Massachusetts: Morgan Kaufmann Verlag.