

# MASTERARBEIT

## Softwareteams im agilen Software Management – Handlungsempfehlungen für die Entscheidung der Team Organisation

ausgeführt am



Studiengang  
Informationstechnologien und Wirtschaftsinformatik

Von: Stefan Obendrauf, B.Sc.  
Pers. Kennz. 1710320008

Graz, am 2. Juli 2020

.....  
Stefan Obendrauf, B.Sc.

# 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.

.....  
Stefan Obendrauf, B.Sc.

# Kurzfassung

In den letzten Jahren ist ein stetiger Anstieg des Stellenwertes von agiler Softwareentwicklung zu beobachten. Im Speziellen ist Scrum eines der bedeutendsten und populärsten Frameworks für das Management agiler Entwicklungen. Im Zuge der agilen Entwicklung und auch im klassischen Projektmanagement, spielt das Team für die Entwicklung eine der größten Rollen. Dies führt dazu, dass auch die Strukturierung dieser Teams eine wesentliche Aufgabe spielt, da die Ressourcen meist knapp sind und die Arbeit der Teams eine möglichst hohe Effizienz aufweisen soll. In dieser Arbeit werden die gängigen klassischen, als auch die agilen Teamstrukturen vorgestellt und verglichen. Ergänzend dazu, werden einige agile Skalierungs-Frameworks betrachtet. Daraus resultierend, werden zwei Teamstrukturen in dieser Arbeit vorgestellt. Zum einen handelt es sich um Projektteams und zum anderen um Produktteams. Beide dieser Teams sollen eine effiziente Umsetzung im Unternehmen ermöglichen. In dieser Arbeit werden dazu Kriterien aufgezeigt, welche für die Entscheidung der Auswahl der Teamstruktur ausschlaggebend sind. Daraus resultieren wiederum Handlungsempfehlungen, welche sich an SoftwaremanagerInnen richten. Diese Handlungsempfehlungen zeigen, dass es möglich ist, aufgrund diverser Kriterien die Auswahl der Teamstruktur zu treffen. Letzten Endes gilt es jedoch die Vor- und Nachteile, welche sich durch die Teamstrukturen ergeben, zu priorisieren. Die Bildung der Teams soll somit, wie auch die agilen Methoden selbst, ein iterativer Lernprozess sein.

# Abstract

In recent years there has been a steady increase in the importance of agile software development. In particular, Scrum is one of the most important and popular frameworks for the management of agile developments. In the course of agile development and also in classic project management, the team plays the central role for development. As a result, the structuring of the teams also plays an increasingly important role, since resources are usually scarce and the teams should work as efficiently as possible. In this work the usual classic and agile team structures are presented and compared. Some agile scaling frameworks are also considered. As a result, two team structures are presented in this work, project and product teams, which should enable efficient implementation. Furthermore, criteria that are decisive for the decision on the selection of the team structure are developed. This results in recommendations for action that are particularly aimed at software managers. The recommendations for action show that it is possible to choose the team structure based on various criteria. Ultimately, however, it is always a prioritization of the advantages and disadvantages that result from the team structures. Eventually, the formation of the teams, like the agile methods themselves, should be an iterative learning process.

# Inhaltsverzeichnis

<b>1. Einleitung</b>	<b>1</b>
1.1. Ausgangssituation . . . . .	1
1.2. Problemstellung . . . . .	2
1.3. Zielsetzung & Forschungsfrage . . . . .	4
1.3.1. Forschungsfrage . . . . .	5
1.3.2. Hypothesen . . . . .	5
1.3.3. Ziele . . . . .	5
1.3.4. Nichtziele . . . . .	6
1.4. Vorgehensweise . . . . .	6
1.5. Aufbau der Arbeit . . . . .	7
<b>2. Teamstrukturen im klassischen Projektmanagement</b>	<b>8</b>
2.1. Funktionale Organisation . . . . .	8
2.2. Stab-Linienorganisation . . . . .	10
2.3. Matrixorganisation . . . . .	11
2.4. Multiprojektorganisation . . . . .	13
2.5. Reine Projektorganisation . . . . .	14
2.6. Objektorientierte Organisation . . . . .	16
2.7. Zusammenfassung . . . . .	17
<b>3. Team Strukturen der agilen Methoden</b>	<b>20</b>
3.1. Feature Teams . . . . .	21
3.2. Component Teams . . . . .	23
3.3. Zusammenfassung . . . . .	25
<b>4. Skalierungs-Frameworks</b>	<b>28</b>
4.1. Scrum of Scrums (SoS) . . . . .	28
4.2. Scaled Agile Framework (SAFe) . . . . .	30
4.3. Large Scale Scrum (LeSS) . . . . .	31
4.4. Nexus . . . . .	33
4.5. Zusammenfassung . . . . .	34
<b>5. Vergleich klassischer und agiler Teamstrukturen</b>	<b>35</b>
5.1. Klassische und agile Teamstrukturen . . . . .	35
5.2. Produktteams . . . . .	37
5.3. Projektteams . . . . .	38
5.4. Zusammenfassung . . . . .	40

<b>6. Vorgehensmuster</b>	<b>42</b>
6.1. Art des Projektes . . . . .	42
6.2. Technische Komplexität des Unternehmens . . . . .	43
6.3. Größe der Projekte beziehungsweise Teilaufgaben . . . . .	43
6.4. Zukünftiges Design der Schnittstellen . . . . .	44
6.5. Kompetenz der EntwicklerInnen . . . . .	44
6.6. Zusammenfassung & Hypothesenbildung . . . . .	45
<b>7. Evaluierung</b>	<b>47</b>
7.1. Vorgehensweise . . . . .	47
7.2. Vorbereitungen . . . . .	49
7.3. Ergebnisse . . . . .	49
7.3.1. InterviewpartnerIn . . . . .	50
7.3.2. Organisation von Teams . . . . .	51
7.3.3. Art des Projektes . . . . .	53
7.3.4. Technische Komplexität des Unternehmens . . . . .	54
7.3.5. Größe der Projekte beziehungsweise Teilaufgaben . . . . .	55
7.3.6. Zukünftiges Design der Schnittstellen . . . . .	56
7.3.7. Kompetenz der EntwicklerInnen . . . . .	56
7.4. Zusammenfassung . . . . .	56
<b>8. Handlungsempfehlungen</b>	<b>58</b>
8.1. Hypothesenprüfung . . . . .	58
8.2. Konkrete Handlungsempfehlungen . . . . .	60
<b>9. Zusammenfassung</b>	<b>61</b>
9.1. Kritische Reflexion . . . . .	63
9.2. Ausblick . . . . .	64
<b>A. Anhang</b>	<b>65</b>
A.1. Interviewleitfaden . . . . .	65
<b>Akronyme</b>	<b>67</b>
<b>Abbildungsverzeichnis</b>	<b>68</b>
<b>Tabellenverzeichnis</b>	<b>69</b>
<b>Literaturverzeichnis</b>	<b>70</b>

# 1. Einleitung

Laut dem Standish Group 2018 Chaos Report haben agile Projekte eine um 60% höhere Chance auf Erfolg, als andere Projekte. Etwas genauer betrachtet, lässt sich sogar sagen, dass „Wasserfall“ Projekte eine dreimal höhere Chance haben zu Scheitern, als agile Projekte (Mersino, 2018). Dadurch ist in den letzten Jahren ein stetiger Anstieg des Stellenwertes von agiler Softwareentwicklung zu beobachten. Damit ist ebenfalls das agile Software Management verbunden, welches sich im Bereich des Software Projektmanagements als Standard gefestigt hat. Wenn von „agil“ in Bezug auf Softwareentwicklung gesprochen wird, ist auch der Begriff „Scrum“ von Bedeutung. Scrum bezeichnet ein agiles Vorgehensmodell des Projekt- und Produktmanagements. Da Scrum zu den meist verbreiteten Modellen zählt, wird für diese Arbeit das Hauptaugenmerk auf Scrum gelegt.

Im Bereich des Software Managements stellt sich nach der Wahl des Vorgehensmodells die Frage, wie die Teams im Unternehmen zudem strukturiert sein sollen. Diese Frage stellt sich sowohl in den klassischen, als auch in den agilen Methoden, welche es in dieser Arbeit zu beantworten gilt.

## 1.1. Ausgangssituation

Agile Ansätze in der Softwareentwicklung treten bereits Anfang der 1990er Jahre in Erscheinung. Populär wird die agile Softwareentwicklung aber erst 1999, als das Buch zu Extreme Programming von Kent Beck veröffentlicht wird. Im Februar 2001 entsteht bei einem Treffen in Utah das agile Manifest. Dort sind 17 VertreterInnen agiler Methoden anwesend. Unter anderem auch die Erfinder von Scrum: Jeff Sutherland und Ken Schwaber (Bleek & Wolf, 2011).

Die agilen Methoden gewinnen in den letzten Jahren zunehmend an Bedeutung. Vor allem im Bereich der Softwareentwicklung ist die agile Vorgehensweise vermehrt als Standard als eine Ausnahme zu sehen (Hochschule Koblenz in Kooperation mit der GPM Deutsche Gesellschaft für Projektmanagement e. V., 2017). Auch die aktuelle Ausgabe der jährlichen Umfrage von CollabNet Inc. und VersionOne Inc. (2019), dem sogenannten „Annual State of Agile Report“ zeigt, dass 97% der befragten Personen in ihrem Unternehmen agile Methoden verwenden.

Es gibt verschiedene agile Methoden, wobei das am meisten verwendete Framework Scrum ist, weswegen auch das Hauptaugenmerk dieser Arbeit auf Scrum liegt (CollabNet Inc. und VersionOne Inc., 2019; Hochschule Koblenz in Kooperation mit der GPM Deutsche Gesellschaft für Projektmanagement e. V., 2017).

Scrum wird bereits seit Anfang der 1990er Jahre verwendet und dies nicht nur in der Softwareentwicklung sondern auch in verschiedensten Branchen. Es ist entwickelt, um komplexe Produkte besser managen zu können. Scrum selbst ist kein Prozess und auch keine Technik, sondern vielmehr ein Framework. In diesem Framework kommen unterschiedliche Prozesse und Techniken zum Einsatz. Mit Hilfe von Scrum sollen alle Bestandteile laufend verbessert werden. Darin enthalten sind das Produkt, das Team und das Arbeitsumfeld (Bittner, Kong & West, 2018; Schwaber & Sutherland, 2017).

Das Framework lässt sich in drei Hauptkategorien gliedern:

- Das Team mit den dazugehörigen Rollen
- Artefakte
- Ereignisse

Wie auch beim klassischen Projektmanagement hängt der Erfolg zum Großteil vom Team ab. Auch im agilen Manifest und den dazugehörigen Prinzipien wird das Team hervorgehoben.

Wenn entschieden wird, wie das Projekt umgesetzt werden soll, agil oder klassisch, stellt sich in beiden Fällen die Frage wie das Team für die Umsetzung strukturiert werden soll. Gerade im agilen Bereich ist die Frage noch nicht, beziehungsweise nur sehr beschränkt beantwortet. Das bildet die Problemstellung, mit der sich diese Arbeit auseinandersetzt. Im nächsten Abschnitt wird dies genauer erläutert.

## 1.2. Problemstellung

Es gibt nahezu gleich viele verschiedene Arten von Strukturen wie es Unternehmen gibt, jedoch stammen all diese Strukturen von zwei Basisstrukturen ab; der funktionalen und der Matrix Struktur. Die funktionale ist die originale Struktur, auf welche bereits Armeen und Industrie basieren. Die funktionale Struktur teilt Abteilungen aufgrund der primären Funktionen und des Zwecks auf. Somit würden zum Beispiel getrennte Teams für EntwicklerInnen, ProjektmanagerInnen und TesterInnen entstehen. Im Gegensatz zur funktionalen Struktur teilt die Matrix Struktur die Teams auf zwei Management Dimensionen auf. So werden zusätzlich zu den funktionalen Aufteilungen die einzelnen Teammitglieder auf verschiedene Teams verteilt, um in jedem Team eine Person des jeweiligen Fachbereiches zu haben. In dieser Arbeit wird

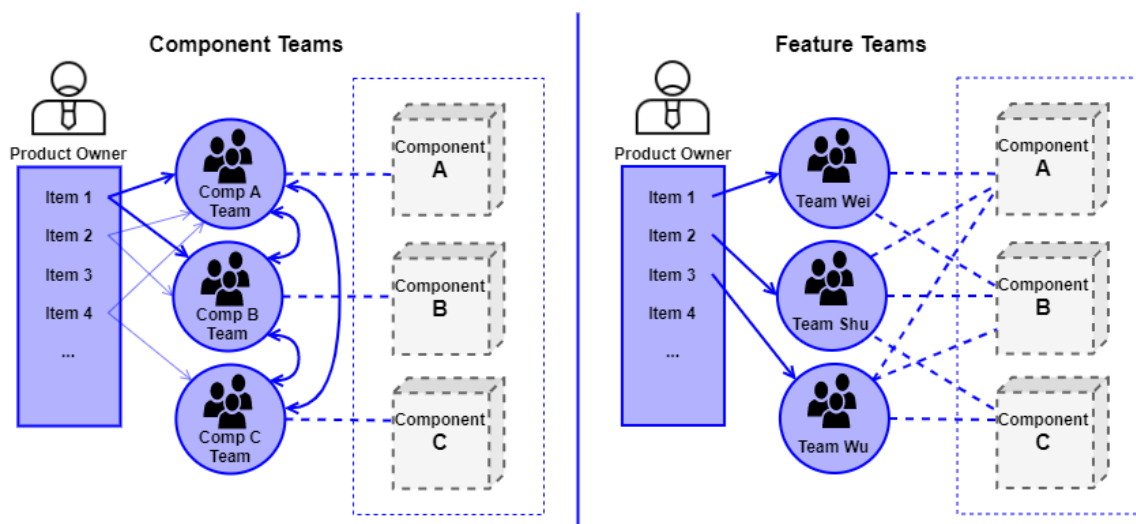


ein Fachbereich als Synonym für eine Komponente, Architektur-Schicht, Fachbereich, Technologie oder ähnliches verwendet. Im Grunde genommen alle Teile eines Systems, welche Integration mit anderen Teilen benötigt, um am Ende einen kompletten Kundennutzen zu generieren. Das Endresultat sieht hierbei so aus, dass in den jeweiligen Teams nicht nur EntwicklerInnen zusammenarbeiten, sondern sich jedes Team aus jeweils einer/einem ProjektmanagerIn, EntwicklerIn und TesterIn zusammensetzt (Abbot & Fisher, 2010).

Aufgrund der vielen unterschiedlichen Strukturen ist die/der SoftwaremanagerIn stets mit der Frage konfrontiert, wie Teams strukturiert werden sollen.

Scrum selbst sagt grundsätzlich nicht viel über die Strukturierung von Teams aus. Schwaber und Sutherland (2017) nennen für die Strukturierung eines Teams lediglich, dass ein Team selbstorganisiert und interdisziplinär sein soll. Ersteres bedeutet, dass das Team selbst über dessen Arbeiten entscheiden kann und dies managt. Zweiteres zielt darauf ab, dass im Team alle Fähigkeiten vorhanden sein müssen, welche zur Erstellung eines Produktinkrementes benötigt werden. Zudem schreiben sie auch über die optimale Größe des Teams, welche aber auch nicht konkret genannt werden kann. Aufgrund dieser Lücke für die Strukturierung haben sich in weiterer Folge auch Skalierungs-Frameworks entwickelt, welche in Kapitel 4 vorgestellt werden.

In den agilen Methoden haben sich zudem zwei Team-Strukturen etabliert. Die sogenannten Feature und Component Teams.

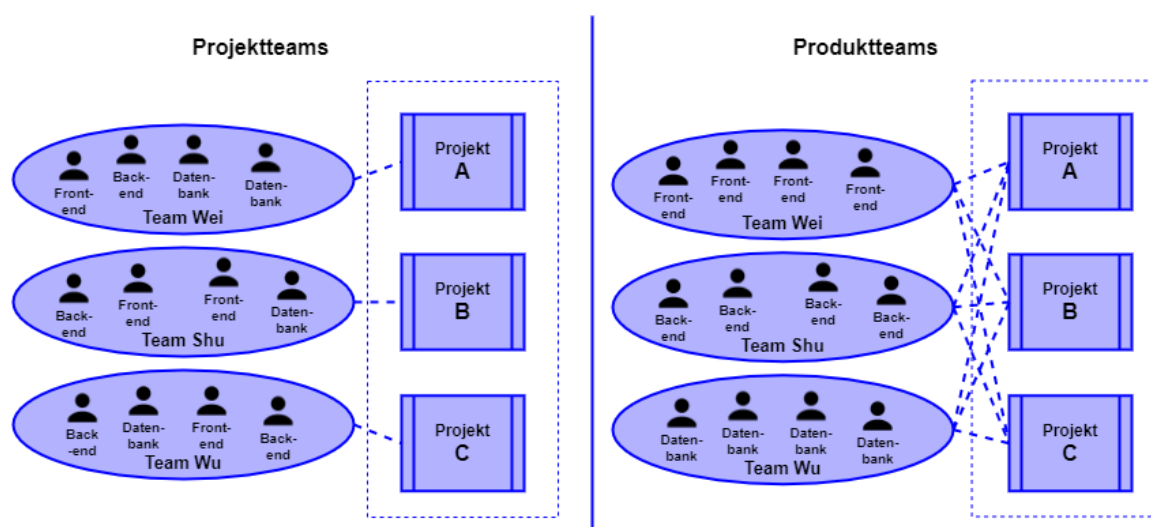


**Abbildung 1.1.:** Grafische Gegenüberstellung von Component und Feature Teams (vgl. The LeSS Company B.V., 2020a)

Diese unterscheiden sich hauptsächlich in der Zuständigkeit und Entwicklung der einzelnen Komponenten eines Produktes. Wobei ein Component Team exklusive Komponenten hat an denen es entwickelt und ein Feature Team potentiell an mehreren

Komponenten arbeitet und auch mehrere andere Teams gleichzeitig an dieser Komponente entwickeln können. Eine Komponente könnte zum Beispiel eine Architektur Schicht wie Benutzeroberfläche, Datenbank oder Anwendungslogik sein.

Da sich die Frage sowohl im klassischen als auch im agilen Bereich stellt, wird in dieser Arbeit stellvertretend auf zwei Teamstrukturen eingegangen. Zum einen dienen Teams, die zwar projektübergreifend jedoch im selben Fachbereich arbeiten und zum anderen Teams, die fachlich übergreifend am selben Projekt arbeiten, der Unterscheidung. Erstere Gruppe wird nachfolgend als Produktteam und letztere als Projektteam bezeichnet. Diese Team Strukturen werden in Kapitel 5 genauer vorgestellt.



**Abbildung 1.2.:** Grafische Gegenüberstellung von Projekt- und Produktteams (Quelle: eigene Darstellung)

Sowohl Scrum als auch die genannten etablierten Teamstrukturen von agilen Methoden beantworten somit nicht, beziehungsweise nur auf Komponenten-Ebene, die Frage wie Teams in einer Organisation strukturiert sein sollen. Scrum selbst beschäftigt sich lediglich mit einem Team und einem Produkt in Isolation. Dies ist in den meisten Fällen in der Realität nur selten anzutreffen.

### 1.3. Zielsetzung & Forschungsfrage

Dieser Abschnitt stellt die konkrete Fragestellung und die Zielsetzungen für diese Arbeit vor, welche sich aufgrund der zuvor genannten Ausgangssituation und Problemstellung ergeben.

### 1.3.1. Forschungsfrage

Im Software Management ist es eine entscheidende Frage, ob man ein Projekt mit Hilfe von klassischem Projektmanagement oder agilen Methoden umsetzen möchte. Hat man sich für eine der beiden Methoden entschieden, stellt sich sogleich die weitere Frage, wie man das Entwicklungsteam strukturieren soll. In dieser Arbeit wird der Fokus jedoch auf Scrum gelegt. Davon leitet sich für diese Arbeit folgende Forschungsfrage ab:

„Aufgrund welcher Kriterien kann die Entscheidung getroffen werden, ob Scrum-Teams als Produkt- oder als Projektteams strukturiert werden sollen?“

Die Beantwortung dieser Frage stellt das Hauptziel dieser Arbeit dar. Durch die Beantwortung sollen im Zuge dieser Arbeit neue Erkenntnisse für die Strukturierung von Teams im agilen Softwarebereich gewonnen werden.

### 1.3.2. Hypothesen

Zusätzlich zur Forschungsfrage wird zu Beginn der Arbeit eine Hypothese erstellt. Durch die Prüfung dieser Hypothese wird der weitere Forschungsprozess angeleitet. Dazu werden üblicherweise zwei Hypothesen aufgestellt: eine Nullhypothese  $H_0$  und eine Alternativhypothese  $H_1$ . Die Alternativhypothese  $H_1$  wird komplementär zu  $H_0$  aufgestellt. Das Verwerfen von  $H_0$  stellt das Ziel einer Hypothesenprüfung dar (Raithel, 2006). Folgende Hypothesen werden für diese Arbeit definiert:

$H_0$ : Die Entscheidung, ob ein Projekt agil oder klassisch gemanagt wird, *ist ausschlaggebend* für die Entscheidung der Strukturierung eines Teams.

$H_1$ : Die Entscheidung, ob ein Projekt agil oder klassisch gemanagt wird, *ist nicht ausschlaggebend* für die Entscheidung der Strukturierung eines Teams.

### 1.3.3. Ziele

Ziel dieser Arbeit ist es Faktoren zu benennen, welche für die Strukturierung eines Teams im Scrum Umfeld entscheidend sind. Dafür wird auch der Unterschied der Faktoren von klassischem Projektmanagement zu denen von agilen Methoden betrachtet. Die Unterschiede dienen dazu, um herauszufinden, ob die Wahl zwischen klassischem Projektmanagement und agilen Methoden einen Einfluss auf die Strukturierung der Teams hat. Das Endresultat dieser Arbeit ist eine Liste an Handlungsempfehlungen für SoftwaremanagerInnen. Diese Handlungsempfehlungen ermöglichen es, die Entscheidung zu treffen, aufgrund welcher Faktoren sie ihre Teams eher

als Projekt- oder als Produktteams strukturieren sollen. Hierfür werden durch diese Arbeit die fachlichen, strukturellen und organisatorischen Faktoren betrachtet.

### **1.3.4. Nichtziele**

Aufgrund der Popularität von Scrum in der Softwareentwicklung liegt das Hauptaugenmerk dieser Arbeit auf dieser Methode. Diese Arbeit bewertet somit die Faktoren nur für die Methode Scrum und nicht für etwaige andere agile Methoden. Des Weiteren wird auch der Unterschied der Branche Software zu anderen Branchen außen vorgelassen. Auch ein Nichtziel dieser Arbeit ist es, die sozialen Faktoren und Verhaltensweisen von Personen für die Teambildung zu berücksichtigen.

## **1.4. Vorgehensweise**

Zu Beginn dieser Arbeit wird im theoretischen Teil das Forschungsthema mit Hilfe von wissenschaftlicher Fachliteratur aufbereitet und diskutiert. Mit dem dadurch erlangten Wissen, wird die Arbeitshypothese geprüft. Auf Basis der Hypothese und den theoretischen Erkenntnissen können bereits erste Vorgehensmuster und dazugehörige Hypothesen für die weitere Bearbeitung erstellt werden. Um diese Vorgehensmuster in konkrete Handlungsempfehlungen überzuführen, wird zum theoretischen Wissen auch der praktische Aspekt benötigt. Dazu wird ein Fragebogen für Personen der Fokusgruppe Software Management und Software Projektmanagement erarbeitet. Durch die Befragung wird das Wissen des Autors mit den entsprechenden Einblicken der speziellen Anforderungen, Erwartungen und Bedürfnissen der Zielgruppen erweitert. Zudem wird das theoretische Wissen mit praktischen Aspekten ergänzt. Die Befragung erfolgt in Form von Leitfadeninterviews. Die Antworten der Befragung werden anschließend einer qualitativen Inhaltsanalyse unterzogen. Durch die Ergebnisse der qualitativen Analyse werden anschließend die Vorgehensmuster in konkrete Handlungsempfehlungen übergeführt.

Für die Zusammenstellung der Stichproben kommen für diese Arbeit nur Unternehmen in Betracht, welche zu dem Zeitpunkt bereits Scrum als agiles Vorgehensmodell verwenden. Zudem wird auch je Teamstruktur (Produkt- und Projektteam) ein Unternehmen vertreten sein. Die Stichprobe für die qualitative Befragung wird sich aus folgenden Personengruppen zusammensetzen:

- IT-LeiterIn (Chief Technology Officer (CTO), Chief Digital Officer (CDO), Chief Information Officer (CIO), Software-TeamleiterIn)
- Software ProjektmanagerIn

- Software EntwicklerIn

Endgültig wird die Stichprobe sechs Personen umfassen, wobei pro Unternehmen jeweils eine Person aus jeder der oben genannten Gruppen vertreten ist. Somit ergibt sich die Stichprobe aus zwei verschiedenen Unternehmen.

## **1.5. Aufbau der Arbeit**

Diese Arbeit gliedert sich in drei Hauptteile. Den ersten Teil stellt eine theoretische Ausarbeitung des Themas dar. Dieser Teil umfasst als erstes die Teamstrukturen im klassischen Projektmanagement, welche in Kapitel 2 abgebildet sind. Als nächstes werden zudem die agilen Teamstrukturen in Kapitel 3 beleuchtet. Zusätzlich werden zu den klassischen und agilen Methoden in Kapitel 4 die Skalierungs-Frameworks der agilen Methoden vorgestellt. Abschließend werden im ersten Hauptteil dieser Arbeit, in Kapitel 5, die klassischen und agilen Methoden verglichen und daraus zwei Teamstrukturen gebildet und vorgestellt. Aufgrund der dadurch erstellten Teamstrukturen und dazugehörigen Einsatzmöglichkeiten, deren Vor- und Nachteile, werden im zweiten Hauptteil erste Vorgehensmuster und dazugehörige Hypothesen erstellt, welche in Kapitel 6 abgebildet sind. Der letzte Hauptteil beinhaltet die Prüfung der Hypothesen und die Erstellung konkreter Handlungsempfehlungen. Um diese Hypothesen zu prüfen, werden Interviews durchgeführt und die Ergebnisse einer qualitativen Inhaltsanalyse unterstellt. Dies ist in Kapitel 7 dargelegt. Die konkreten Handlungsempfehlungen findet man in Kapitel 8. Abschließend wird die Arbeit zusammengefasst, eine kritische Reflexion erstellt und ein Ausblick in Kapitel 9 gegeben.

## 2. Teamstrukturen im klassischen Projektmanagement

Die Teamstrukturen im klassischen Projektmanagement sind durch die Organisationsstruktur abgebildet. Diese Struktur beinhaltet nicht nur die Aufteilung der MitarbeiterInnen in Teams, Abteilungen und Fachabteilungen, sondern auch die Management Hierarchie für das Steuern und Kontrollieren der MitarbeiterInnen. Es gibt eben so viele verschiedene Strukturen wie es auch Unternehmen gibt. Jedoch existieren zwei Strukturen von denen alle abstammen. Diese sind die funktionale und die Matrixorganisation (Abbot & Fisher, 2010).

Die funktionale Organisation wird oft auch als Linienorganisation beschrieben. Zusätzlich zur Linienorganisation gibt es auch noch eine Abwandlung, welche als Stab-Linienorganisation oder Einfluss-Organisation bezeichnet wird. Zudem werden in diesem Kapitel die Multiprojektorganisation, die reine Projekt-Organisation und die objektorientierte Organisation betrachtet.

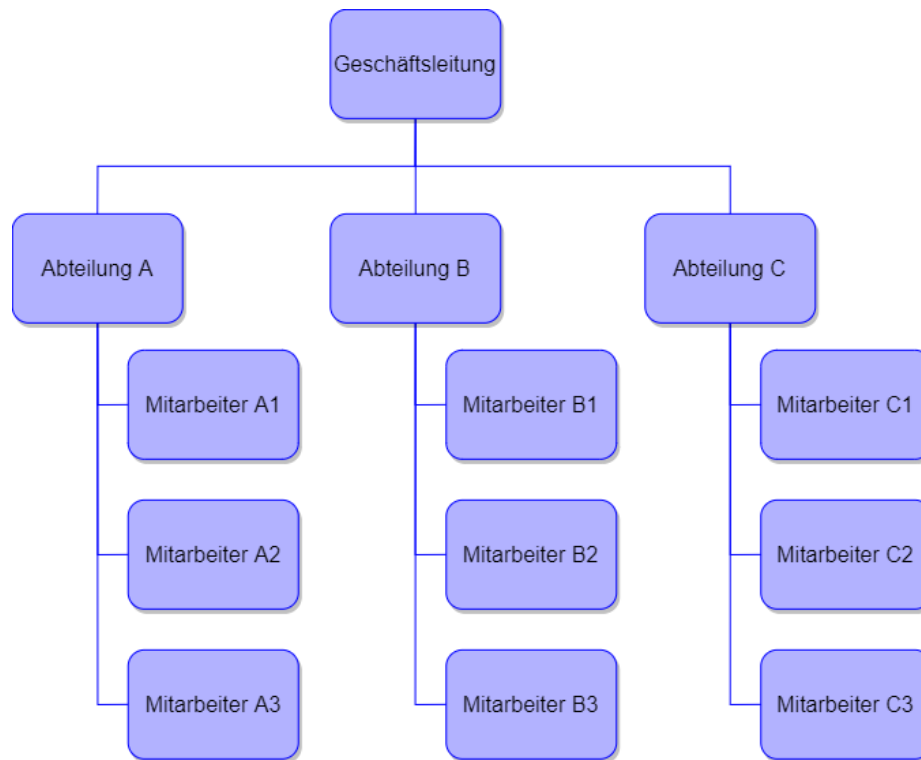
In den folgenden Abschnitten werden die genannten Strukturen vorgestellt, deren Vor- und Nachteile erwähnt und in welchen Szenarien sie zum Einsatz kommen.

### 2.1. Funktionale Organisation

Die funktionale Struktur ist die originale Struktur, auf welche bereits Armeen und Industrie basieren. Man nennt sie oft auch Linienorganisation. Die funktionale Organisation teilt die Abteilungen und Fachbereiche nach deren primären Funktionen ein (Bühner, 2004). Eine schematische Darstellung ist in Abbildung 2.1 zu sehen. Dadurch ergeben sich Gruppen von Personen, welche separiert sind, wie zum Beispiel Korn und Mais, wodurch es auch als Silo Methode bekannt ist. Das Resultat sind eigene Teams der Personengruppen TesterInnen, EntwicklerInnen und ProjektmanagerInnen (Abbot & Fisher, 2010).

Ebenso wie die einzelnen Fachbereiche, sind auch die Personal- und die Projektverantwortlichkeiten divisional oder funktional getrennt anhand technischer Zerlegung in z.B.: Produkte, Versionen oder Aufgaben. Dies schafft klare Organisationsstrukturen (Broy & Kuhrmann, 2013).

Jede Person in der Struktur bekommt folglich nur Anweisungen von einer/einem Vorgesetzten in seiner Linie. Sowohl die Befehlskette, als auch die Berichtkette ist so-



**Abbildung 2.1.:** Schematische Darstellung einer funktionalen Organisation (in Anlehnung an Abbot & Fisher, 2010)

mit linear und nur in einem Bereich vorhanden. Dadurch ergibt sich eine erlaubte Kommunikation nur über die Linienvorgesetzte/den Linienvorgesetzten zu anderen Linien (Bergmann & Garrecht, 2008). Für bereichsübergreifende Tätigkeiten und Kommunikation empfiehlt es sich bei größeren Unternehmen eine eigene Gruppe für die Querschnitts-Tätigkeiten zu etablieren (Broy & Kuhrmann, 2013).

Einer der stärksten Vorteile der funktionalen Organisation ist die transparente Struktur mit den klaren Verantwortlichkeiten. Dadurch ergibt sich auch eine klare „Kommandokette“. Dies erlaubt es, Arbeitspakete leicht auf die einzelnen MitarbeiterInnen zu verteilen. Zudem ergibt sich dadurch eine spezialisierte Einheit. Auch diese Gemeinsamkeit von Management und Mitarbeiterinnen/Mitarbeitern stärkt die Einhaltung von Standards, da diese von der Abteilung selbst getrieben werden (Abbot & Fisher, 2010; Broy & Kuhrmann, 2013; Vahs, 2007).

Die meisten Projekte benötigen nicht nur die Expertise von einer Abteilung, sondern sind meist fachbereichsübergreifend. So benötigt fast jedes Projekt zumindest Projektmanagement, Entwicklung und die Qualitätskontrolle. Durch die Linienorganisation sind diese Einheiten voneinander getrennt wodurch sich lange Kommunikationspfade für die abteilungsübergreifende Abstimmung ergeben. Somit sind auch die Verantwortlichkeiten für ein Projekt nicht in einer einzelnen Abteilung zu finden. Die erste Person, die eine gesamtheitliche Verantwortung hat, ist die Geschäftsleitung.

Dies kann schnell zu einer Überlastung der Geschäftsleitung führen (Abbot & Fisher, 2010; Broy & Kuhrmann, 2013; Probst, 1993).

Die funktionale Organisation eignet sich besonders für Unternehmen mit hoher Stabilität bei den Aufgaben. Dies ist oft der Fall bei Behörden. Unter Stabilität wird hier verstanden, dass die Aufgaben klar definiert, relativ gleichbleibend, sich wiederholend und mit geringem Erfordernis an Interaktion zwischen den Abteilungen gestaltet sind (Bergmann & Garrecht, 2008; Broy & Kuhrmann, 2013). Abbot und Fisher (2010) schreiben auch, dass die funktionale Organisation zum Einsatz kommen kann, wenn die Spezialisierung größer, beziehungsweise wichtiger ist, wie das Problem der Koordination und der Verantwortlichkeiten. Ein Beispiel hierfür ist die Skalierbarkeit einer Datenbank. Hier werden viele Tätigkeiten nur in der Entwicklung passieren und auch die Entscheidungen dafür getroffen. Der Produktbereich ist bei so einem technischen Projekt nicht so involviert, wie bei der Markteinführung eines neuen Produktes.

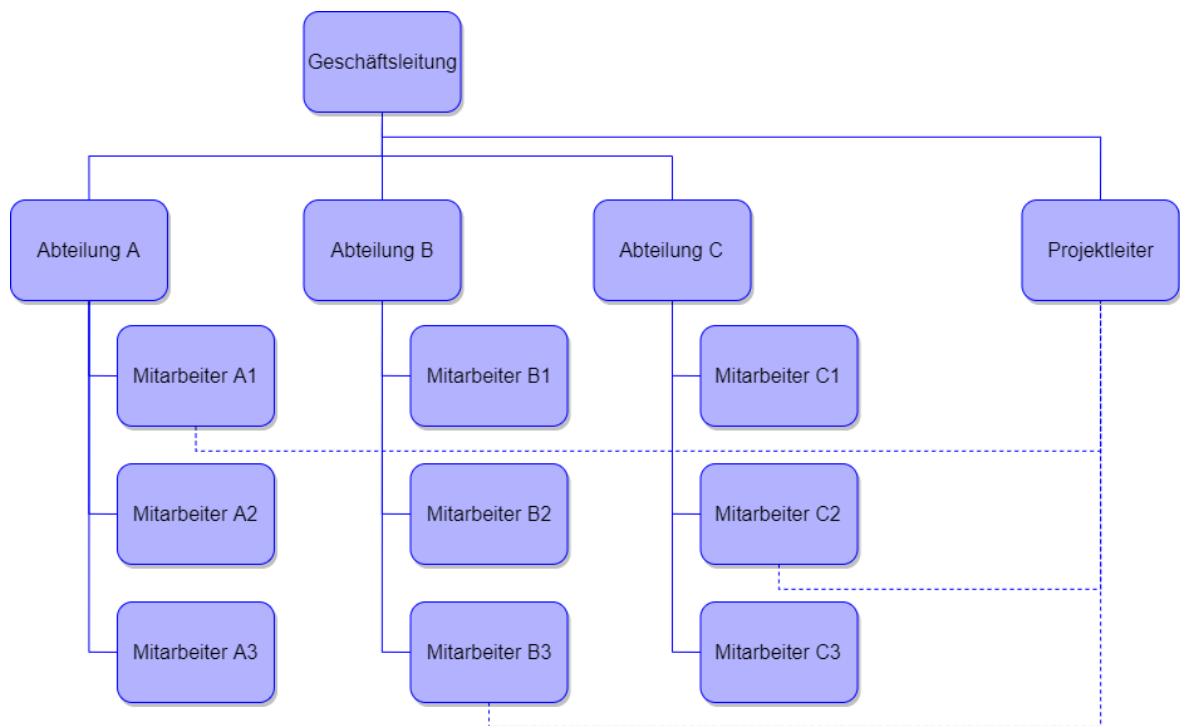
## 2.2. Stab-Linienorganisation

Die Stab-Linienorganisation ist eine leichte Abwandlung der funktionalen oder Linienorganisation. Man nennt sie auch Projektkoordination oder Einfluss-Organisation. Laut Bachmann et al. (2019) ist dies die minimale Form einer Projektorganisation. Hierbei wird, wie in Abbildung 2.2 dargestellt, die funktionale Organisation um eine Stabsstelle ergänzt. Die restliche originale Struktur bleibt unverändert. Dadurch bleibt die Verantwortung in den einzelnen Abteilungen und auch die ProjektmitarbeiterInnen bleiben fachlich und personell in ihrer Abteilung. Die Stabsstelle übernimmt lediglich Koordinations- und Überwachungsaufgaben. Sie fungiert somit als BeraterIn für die Linie (Broy & Kuhrmann, 2013).

Durch die Einführung einer Stabsstelle vermeidet man von den klassischen Nachteilen einer funktionalen Organisation zumindest die Überlastung und zu wenig unmittelbare Einwirkungsmöglichkeiten der Geschäftsleitung (Broy & Kuhrmann, 2013). Auch der Personaleinsatz wird flexibler, da die MitarbeiterInnen gleichzeitig in mehreren Projekten arbeiten können, ohne organisatorische Schwierigkeiten zu generieren. Aus diesem Grund ist auch nach Abschluss eines Projektes die Wiedereingliederung der ProjektmitarbeiterInnen in die Linie kein Problem (Bachmann et al., 2019; Kraus & Westermann, 2010; Winkelhofer, 2005).

Ein großer Nachteil dieser Organisationsform ist, dass die Projektverantwortung verloren geht, und niemand diese Aufgabe übernimmt. Auch die Reaktionszeit auf etwaige Abweichungen des Projektes sind sehr hoch. Das Team selbst, verbleibt in der Linie, wodurch sich die abteilungsübergreifende Zusammenarbeit schwierig gestaltet





**Abbildung 2.2.:** Schematische Darstellung einer Stab-Linienorganisation (in Anlehnung an Bachmann et al., 2019)

(Bachmann et al., 2019; Kraus & Westermann, 2010; Schulte-Zurhausen, 2013; Winkelhofer, 2005).

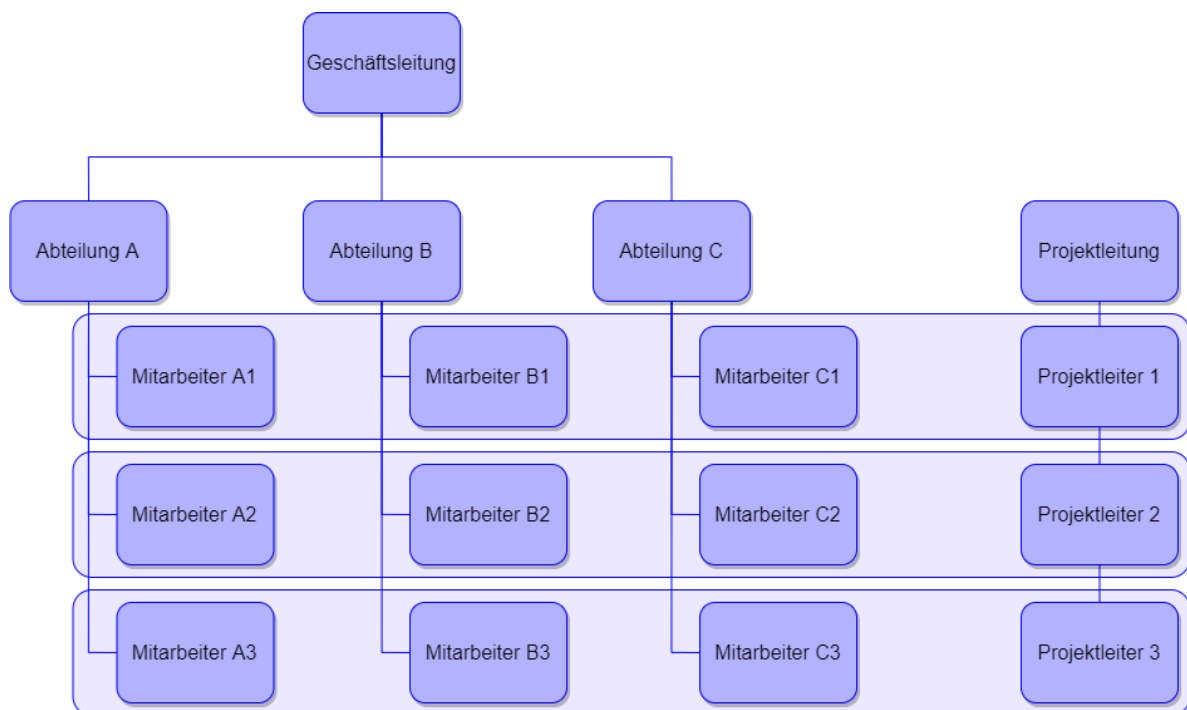
Diese Organisationsform eignet sich besonders für kleinere bis mittlere Projekte. Vor allem für jene Projekte, welche ein geringes Risiko aufweisen und mit überschaubarem Zeitdruck erledigt werden können (Kraus & Westermann, 2010; Winkelhofer, 2005). Auch eine Anwendung dieser Struktur kann man vornehmen, wenn die Projekte den Rahmen der herkömmlichen Aufgaben der Linien nicht wesentlich überschreiten, wie zum Beispiel bei Kundenaufträgen oder bei einfachen Produktentwicklungen (Bachmann et al., 2019).

### 2.3. Matrixorganisation

Die Matrixorganisation ist laut Bachmann et al. (2019); Kraus und Westermann (2010) die häufigste Organisationsform im klassischen Projektmanagement. Diese versucht die Nachteile der funktionellen Organisation auszumerzen und gleichzeitig neue Vorteile zu schaffen. Die Matrixorganisation führt zur vorhandenen Linie zumindest eine zweite Dimension ein. Hier teilen sich somit die Verantwortung und die Kompetenzen zwischen der Projektleitung und der Linienführung auf. So bleiben die Perso-

nalverantwortung und generelle administrative Aufgaben in einer Linie, während die/der ProjektleiterIn die volle Verantwortung über das gesamte Projekt in Bezug auf Kosten und Termin hat (Broy & Kuhrmann, 2013; Winkelhofer, 2005).

Die Koordination zwischen Projekt- und Personalverantwortung muss gut abgestimmt sein. Herrschen hier keine klaren Aufteilungen und keine klaren Abmachungen, stellt dies ein großes Problem für die MitarbeiterInnen dar. Diese haben zumindest zwei Vorgesetzte, deren Aufgaben sie erfüllen müssen und stehen bei schlechter Abstimmung zwischen den Meinungen (Bachmann et al., 2019; Kraus & Westermann, 2010). Im Vergleich zur funktionalen Struktur entstehen somit keine getrennten Teams, sondern gemischte Teams aus einer/einem ProjektleiterIn, EntwicklerIn und TesterIn. In Abbildung 2.3 ist die Zusammengehörigkeit der Team-Mitglieder durch die Schattierung dargestellt.



**Abbildung 2.3.:** Schematische Darstellung einer Matrixorganisation (in Anlehnung an Abbot & Fisher, 2010)

Wie bereits erwähnt, ist hier eines der größten Probleme die Regelung der Aufgaben und Kompetenzen. Somit ist die Matrixorganisation auch nur in jenen Unternehmen anzuwenden, welche dies klar regeln können. Auch Probleme und Konflikte, welche in Bezug darauf auftreten, müssen thematisiert werden und somit soll eine gut ausgeprägte Konfliktkultur herrschen damit diese Organisation funktionieren kann (Bachmann et al., 2019). Speziell eignet sich diese Struktur für Unternehmen mit zahllosen Projekten, welche gleichzeitig laufen, aber auch für stark abteilungsübergreifende Projekte, sowie Projekte bei denen der Ausgang kritisch hinsichtlich dem Zeitmanagement oder anderer Faktoren ist (Abbot & Fisher, 2010; Broy & Kuhrmann, 2013; Winkelhofer, 2005).

Laut Abbot und Fisher (2010) ist ein großer Vorteil der Matrixorganisation, dass diese zumindest die Nachteile der funktionalen Organisation verbessert. So liegen die eindeutige Verantwortung und Entscheidungskompetenz bei der/dem ProjektleiterIn und nicht mehr, als erste Person, bei der Geschäftsleitung. Zudem sind die Kommunikationswege der Teams untereinander kürzer, da diese nicht mehr über die Managementebene laufen, sondern direkt in den Projekt Besprechungen alle vertreten sind. Bachmann et al. (2019); Winkelhofer (2005) betonen auch, dass durch diese Organisation das Team und die/der ProjektleiterIn sich für das Projekt verantwortlich fühlen und sich somit besser damit identifizieren können. Einen weiteren positiven Aspekt bildet der flexible Personaleinsatz.

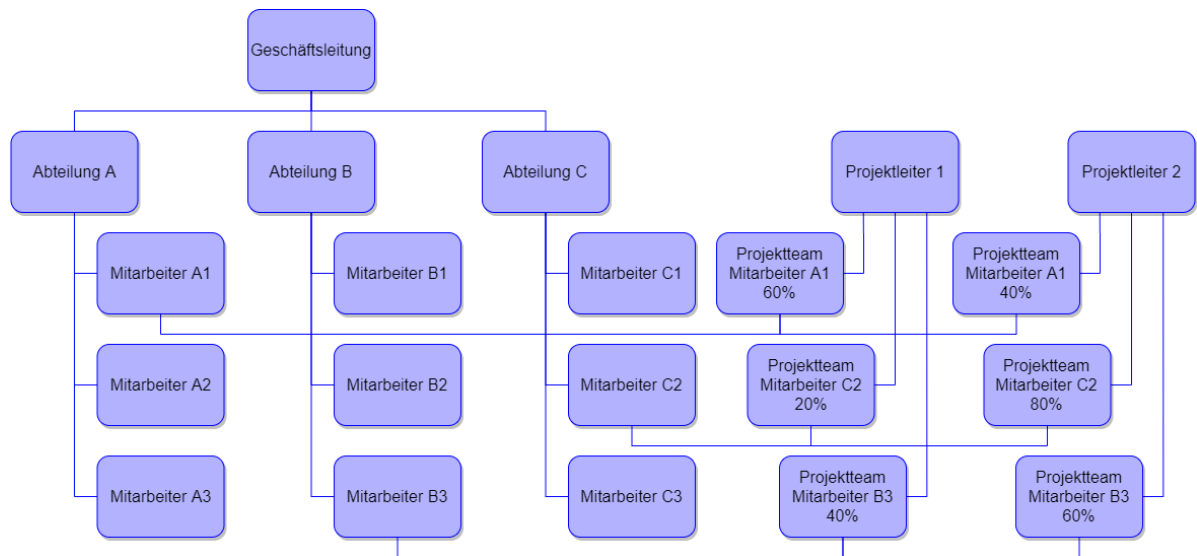
Der größte negative Aspekt ist jedoch die Gefahr von Kompetenzkonflikten zwischen den Linien- und Projektautoritäten. Damit geht auch ein erhöhter Koordinationsaufwand und Kommunikationsbedarf einher (Bachmann et al., 2019; Bergmann & Garrecht, 2008; Broy & Kuhrmann, 2013; Kraus & Westermann, 2010). Zudem zeigt auch die Tatsache, dass die Teammitglieder mehr als eine Vorgesetzte/einen Vorgesetzten zufrieden stellen müssen, eine negative Seite dieser Organisation (Abbot & Fisher, 2010; Bachmann et al., 2019; Broy & Kuhrmann, 2013).

### **2.4. Multiprojektorganisation**

Um noch mehr Flexibilität des Ressourceneinsatzes zu erlangen, beschreiben Broy und Kuhrmann (2013) die Multiprojektorganisation. Diese ist vom Prinzip her gleich wie die Matrixorganisation, mit dem Unterschied, dass die einzelnen MitarbeiterInnen nicht nur in einem Projekt, sondern gleichzeitig in mehreren arbeiten. Dies bedeutet für die Teammitglieder neben der/dem Personalvorgesetzten zusätzlich mehrere ProjektleiterInnen, welche als fachliche Vorgesetzte fungieren. Abbildung 2.4 zeigt eine beispielhafte Darstellung einer solchen Organisation.

Darüber hinaus beschreiben zur allgemeinen Struktur Broy und Kuhrmann (2013) auch die Vor- und Nachteile dieser Organisation. Die Multiprojektorganisation kann damit punkten, dass sie sehr flexible, kurzzeitige und partielle Kapazitätszuordnungen zulässt. Zudem werden die Projekte optimal mit der Expertise der Abteilungen versorgt. Außerdem, bleiben die einzelnen MitarbeiterInnen ihren Abteilungen zugeordnet, wodurch sie sich weiterhin stark auf ihre Kernaufgaben konzentrieren können. Durch die Arbeit, in mehreren gleichzeitigen Projektteams, kommt es auch zu einem schnellen Erfahrungsaustausch zwischen ähnlichen Projekten.

Auf der negativen Seite dieser Organisation, findet man unter den Projekten eine erhöhte Konkurrenz der Ressourcen. Aufgrund der vielen gleichzeitig laufenden Projekte besteht auch ein erhöhter Kommunikations- und Koordinationsaufwand. Wie auch bei der Matrixorganisation, ist es ein Nachteil für die MitarbeiterInnen, dass



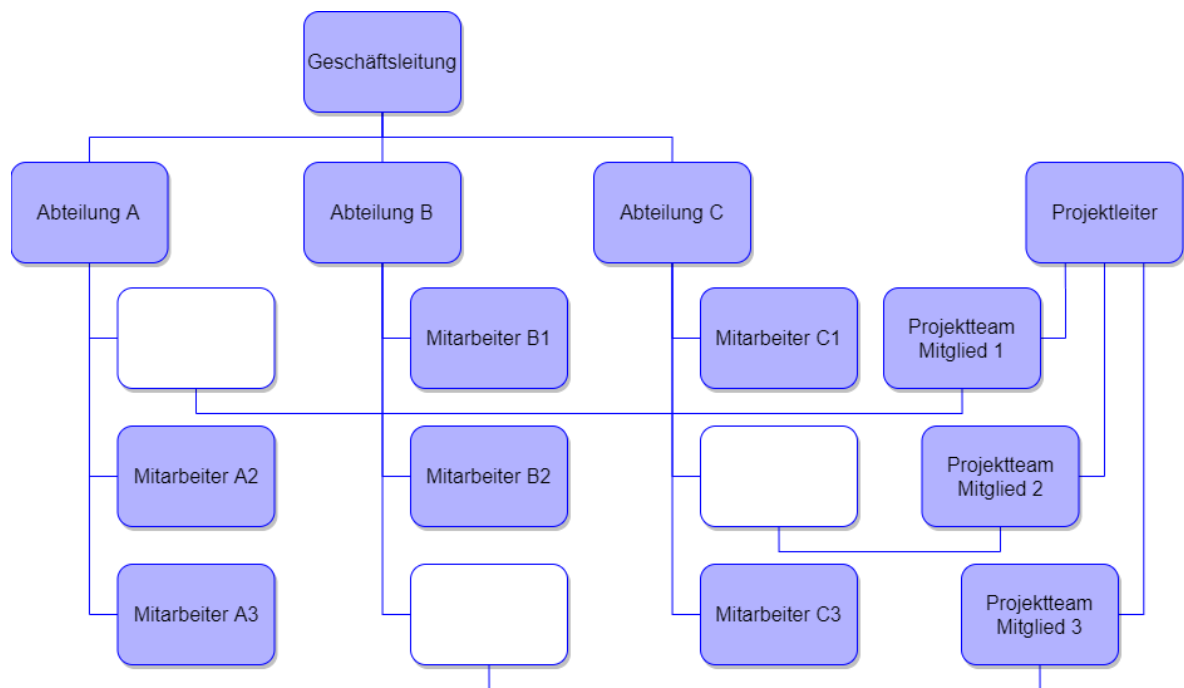
**Abbildung 2.4.:** Schematische Darstellung einer Multiprojektorganisation (Quelle: eigene Darstellung)

sie mehrere Vorgesetzte haben. Ergänzend ist es ein weiteres Problem, dass Plan-Abweichungen eines Projektes auf alle anderen Projekte übergreifen, in denen dieselben Mitglieder tätig sind (Broy & Kuhrmann, 2013).

## 2.5. Reine Projektorganisation

Bei der reinen Projektorganisation wird temporär für die Dauer des Projektes eine eigene Organisationseinheit gegründet. In dieser Einheit ist die/der ProjektleiterIn sowohl fachlich, als auch personell für die Mitglieder zuständig. Dadurch ist die Einheit komplett von der Linie getrennt. Die/Der ProjektleiterIn trägt somit die volle Verantwortung über Zeit-, Kosten- und Ressourcenziele (Bachmann et al., 2019; Kraus & Westermann, 2010; Winkelhofer, 2005). Abbildung 2.5 zeigt die schematische Darstellung. Hierbei erkennt man, dass die Teammitglieder komplett aus der Linie gelöst werden. Dies führt zu Lücken, welche in Abbildung 2.5 als leere Blöcke dargestellt sind. Die reine Projektorganisation kann somit recht teuer sein, wenn die bisherigen Stellen in dieser Linie wieder nachbesetzt werden müssen. Ansonsten kann man diese Organisation wie eine „Task Force“ sehen, welche unabhängig und effizient ein Projekt erledigen kann (Bachmann et al., 2019).

Diese Organisationsform eignet sich besonders für Projekte, welche ein hohes Risiko aufweisen, zeitkritisch sind oder strategisch wichtige Projekte im Unternehmen sind. Zudem kann es auch bei Projekten Anwendung finden, welche wenig mit der Kernaufgabe dieser Linie zu tun haben, wie zum Beispiel eine neue Produktlinie ent-



**Abbildung 2.5.:** Schematische Darstellung einer reinen Projektorganisation (vgl. Bachmann et al., 2019)

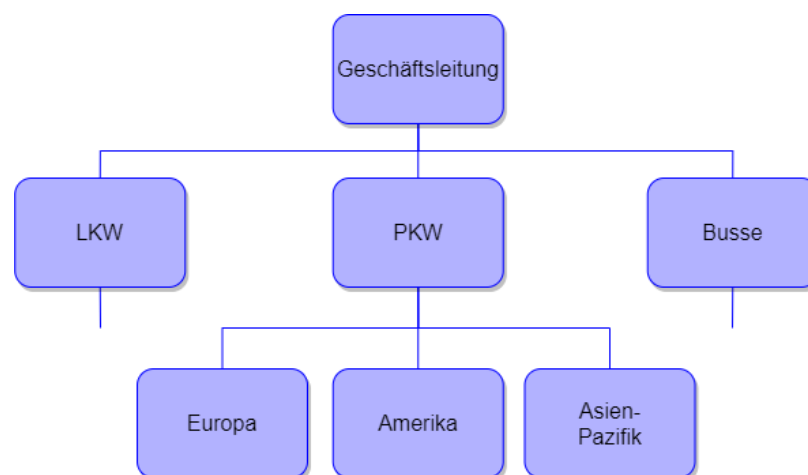
wickeln oder einen Neubau erstellen (Bachmann et al., 2019; Kraus & Westermann, 2010; Winkelhofer, 2005).

Wie bereits erwähnt, ist ein großer Vorteil dieser Organisation, dass das Team eigenständig, ohne jeglichen Einfluss der Linie, ein Projekt abwickeln kann. Dadurch ist die Fertigstellung der Projekte äußerst effizient. Auch bei Störungen kann das Team schnell reagieren. Ein Vorteil ist, dass die klare und eindeutige Verantwortung und Entscheidungskompetenz bei der/dem ProjektleiterIn liegen. Für das Team ist es auch ein Vorteil, dass es sich sehr gut mit dem Projekt identifizieren kann (Bachmann et al., 2019; Kraus & Westermann, 2010; Winkelhofer, 2005).

Einer der größten Nachteile dieser Organisation, ist die Wiedereingliederung der MitarbeiterInnen in die Linie nach Abschluss eines Projektes. Dies kann sich als schwierig erweisen und unter Umständen sehr teuer sein, vor allem wenn bereits Stellen nachbesetzt sind. Auch die Personalflexibilität zeigt sich bei dieser Organisation eher gering. Besonders wenn Spezialistinnen/Spezialisten nur zeitweise benötigt werden, ergibt sich hier eine massive Einschränkung. Zudem kann sein, dass die Linienorganisation die Ergebnisse des Projektes nicht akzeptiert (Bachmann et al., 2019; Kraus & Westermann, 2010; Winkelhofer, 2005).

## 2.6. Objektorientierte Organisation

Bergmann und Garrecht (2008) stellen die funktionale Organisation der objektorientierten Organisation gegenüber. Sie schreiben, dass es prinzipiell zwei Arten gibt ein Unternehmen zu gliedern. Entweder funktional oder nach Objekten. Bei allen bisherig genannten Strukturen gliedern sich alle Linien nach Funktionen. Es werden zusätzliche Strukturen und Einheiten geschaffen um Projekte auch abteilungsübergreifend abzubilden. Bei der objektorientierten Organisation, welche man auch Spartenorganisation nennt, teilt man die Abteilungen nach für das Unternehmen wesentlicher zusammenhängender Objekte. Diese Sparten oder Objekte können zum Beispiel, wie in Abbildung 2.6 dargestellt, Produktlinien sein, oder wie in der untersten Ebene, verschiedene Regionen. Es können aber auch Absatzkanäle sein wie Versand-, Direkt- und Onlinehandel oder auch geteilt nach Kundengruppen wie Groß-, Firmen- und Privatkunden. Besonders eignet sich diese Strukturierung für große Unternehmen, welche in mehreren Regionen präsent sind oder Produkte haben, welche schnell auf Änderungen der Umwelt reagieren müssen.



**Abbildung 2.6.:** Schematische Darstellung einer Spartenorganisation (vgl. Bergmann & Garrecht, 2008)

Als Vorteile dieser Organisation nennen Bergmann und Garrecht (2008) die gute Kenntnis der Umwelt, da das Unternehmen für spezielle Produkte und Märkte strukturiert ist. Zudem sehen sie es auch als Vorteil, dass die Verantwortung des operativen Geschäftes bei den Linienverantwortlichen liegt. Diese agieren somit als „UnternehmerIn im Unternehmen“. Aufgrund des gesamthaften Denkens der Linie, haben in der Regel auch die Entscheidungsprozesse eine gute Qualität.

Durch die Verteilung der operativen Verantwortung in die einzelnen Sparten, bedarf es an mehreren gut ausgebildeten Führungskräften in den verschiedenen Ebenen der Struktur. Zudem kommt auch ein erhöhter Koordinationsaufwand der einzelnen Produkt- beziehungsweise Marktbereiche, als negativer Aspekt hinzu. Dies kann

dazu führen, dass auch eigene Stellen für diese Koordination benötigt werden. Abschließend kann es durch das Ausmaß an Führungskräften zum Verlust der einheitlichen Führungsgrundsätze kommen (Bergmann & Garrecht, 2008).

## 2.7. Zusammenfassung

Wie bereits zu Beginn dieses Kapitels erwähnt, gibt es so viele Organisationsformen wie es auch unterschiedliche Unternehmen gibt. Jedoch stammen alle von der funktionalen und der Matrix Struktur ab. Dies zeigt sich auch im Laufe der weiteren Abschnitte in diesem Kapitel. Die meisten vorgestellten Organisationsformen sind sehr ähnlich zu den genannten zwei Hauptstrukturen. Jede von ihnen mit ihren Vor- und Nachteilen. Dadurch gibt es auch nicht die eine Organisationsform, die für alles perfekt funktioniert, sondern die Wahl ist immer situationsabhängig.

In dieser Arbeit sieht man die Stab-Linienorganisation beziehungsweise Einfluss-Organisation, die Matrixorganisation und die reine Projektorganisation, als die wichtigsten Organisationsformen für Projekte. Die Multiprojektorganisation wird nicht weiter betrachtet, da sie ein Spezialfall der Matrixorganisation ist und somit mit ähnlichen Kriterien ausgewählt wird. Die funktionale Organisation und Spartenorganisation werden auch außen vorgelassen, da diese hauptsächlich für Projekte beziehungsweise Arbeiten geeignet sind, welche eine hohe Stabilität aufweisen und relativ gleichbleibend sind. Da in dieser Arbeit der Fokus auf agilen Methoden liegt, sind schnelle Anpassung an Änderungen der Anforderungen, eine Voraussetzung und somit keine stabilen, gleichbleibenden Tätigkeiten, vor auszusehen.

Um die richtige Projektorganisation zu wählen, nennt Winkelhofer (2005) einige Kriterien zur Auswahl der Organisation. Diese sind in Tabelle 2.1 aufgelistet.

Aufgrund der Kriterien lassen sich folgende Einsatzmöglichkeiten der drei Projektorganisationsformen nennen:

### **Stab-Linienorganisation**

- Projekte mit kleinem bis mittlerem Umfang
- Projekte, welche die Kernaufgaben der Linie nicht wesentlich überschreiten z.B.: einfache Produktentwicklungen
- Projekte mit geringem Risiko
- Projekte mit geringem Zeitdruck

Kriterien	Stab-Linienorganisation	Matrixorganisation	Reine Projektorganisation
Bedeutung für das Unternehmen	Gering	Groß	Sehr groß
Umfang des Projektes	Gering	Groß	Sehr groß
Unsicherheit der Zielerreichung	Gering	Groß	Sehr groß
Technologie	Standard	Kompliziert	Neu
Zeitdruck	Gering	Mittel	Hoch
Projektdauer	Kurz	Mittel	Lang
Komplexität	Gering	Mittel	Hoch
Bedürfnis nach zentraler Steuerung	Mittel	Groß	Sehr Groß
MitarbeiterInnen-Einsatz	Nebenamtlich (Stab)	Teilzeit (variabel)	Hauptamtlich
ProjektleiterInnen-Persönlichkeit	Wenig relevant	Qualifizierte / Qualifizierter ProjektleiterIn	Sehr fähige / fähiger ProjektleiterIn

**Tabelle 2.1.:** Kriterien zur Auswahl der Organisationsform im klassischen Projektmanagement (vgl. Heinrich & Lehner, 2005)

- Projekte mit teamorientierten Führungsstrukturen

### Matrixorganisation

- Einer hohen Anzahl an laufenden Projekten
- Stark abteilungsübergreifende Projekte
- Projekte bei denen die Aufgaben und Kompetenzen zwischen Projekt und Linie klar geregelt sind
- Projekte bei denen die Ressourcen zwischen Linie und Projekt optimal koordiniert sind
- Projekte bei denen die Bereiche nicht miteinander konkurrieren, sondern zusammenarbeiten wollen

### Reine Projektorganisation

- Vollzeit-Projekte



- Projekte, welche wenig mit den herkömmlichen Aufgaben der Linie zu tun haben
- Projekte mit hohem Risiko
- Zeitkritische Projekte
- Strategisch wichtige Projekte
- Projekte, bei denen eine besonders hohe Effizienz der Ergebnisse gefordert ist

Durch die Kriterien aus Tabelle 2.1 und der genannten Einsatzmöglichkeiten, kann man die Wahl der richtigen Organisationsform in klassischen Projektmanagement Umgebungen bereits einigermaßen gut wählen (Bachmann et al., 2019; Kraus & Westermann, 2010; Winkelhofer, 2005). Diese Kriterien und Einsatzmöglichkeiten dienen in den weiteren Abschnitten als Diskussionsgrundlage. Diese werden mit ähnlichen Kriterien und Einsatzmöglichkeiten der agilen Methoden verglichen, um festzustellen, ob die Wahl, ob klassisch oder agiles Projektmanagement einen Einfluss auf die Strukturierung der Teams hat. Im Weiteren werden die Kriterien auch verwendet um die Strukturen auch mit den beiden, in dieser Arbeit vorgestellten Teams, zu vergleichen.

### 3. Team Strukturen der agilen Methoden

Bei den klassischen Projektteams wird ein Team zumeist nur temporär für einen speziellen Zweck zusammengestellt. Wenn dieser Zweck erledigt ist, meistens nach Projektabschluss, werden die Teammitglieder wieder in neue Teams für neue Zwecke gegliedert (Devine, 2002). Beim agilen Ansatz soll das Team möglichst effizient sein und daher auch langlebig zusammenarbeiten. Dies stellt den ersten großen Unterschied zu den klassischen Methoden dar (Roock & Wolf, 2018).

Scrum besagt, dass ein Team prinzipiell aus drei Rollen bestehen soll. Dem Product Owner, dem Scrum Master und dem Entwicklungsteam. Für diese Arbeit ist lediglich das Entwicklungsteam interessant (Schwaber, 2004). Schwaber und Sutherland (2017) nennen dafür zwei wichtige Aspekte: Das Team soll selbstorganisiert sein und es soll interdisziplinär aufgestellt sein. Das heißt, niemand sagt dem Entwicklungsteam wie es arbeitet und aus dem Product Backlog eine auslieferbare Software erstellt. Zudem muss in dem Team jede Fähigkeit besetzt sein, welche benötigt wird um das Produkt zu erstellen (Röpstorff & Wiechmann, 2016). Dies sind zwei weitere große Unterschiede zu klassischen Teams. Genauer zur Struktur des Teams wird jedoch in Scrum nicht beschrieben.

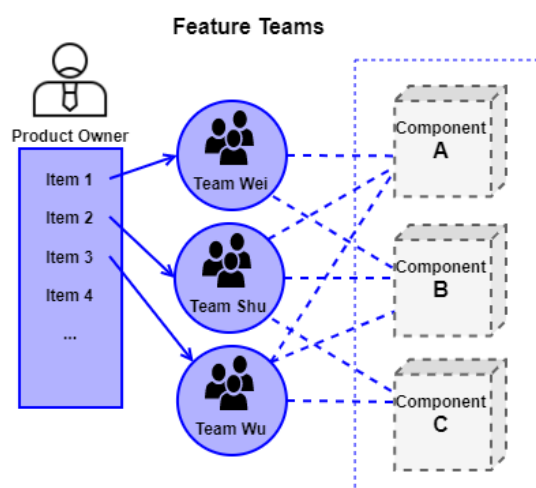
Der Product Backlog ist ein Artefakt von Scrum. Dieser stellt eine Liste aller Anforderungen an das Produkt dar, welches entwickelt werden soll. Dies ist die einzige Anforderungsquelle für die Entwicklung (Benefield, Deemer, Larman & Vodde, 2012). Zuständig für die Wartung, den Zugriff, die Erstellung der Tätigkeiten und das gesamte Management des Product Backlog ist der Product Owner. Dieser stellt sicher, dass der Product Backlog immer gut gefüllt ist, klar formuliert und priorisiert ist, sowie den Kundenanforderungen entspricht (Beedle & Schwaber, 2001). In diversen Scrum Meetings betrachtet man den Backlog vom Team und entscheidet, was in der nächsten Iteration erledigt werden kann. Dazu muss der Product Backlog immer priorisiert sein, damit das Team von Platz 1 bis n entscheiden kann, welche Arbeitspakete es schaffen wird. Der Scrum Master ist dafür verantwortlich, dass alle Beteiligten jegliche Aspekte von Scrum verstehen und der Prozess dementsprechend läuft. Zudem dient sie/er als ModeratorIn der einzelnen Meetings und hilft generell allen Beteiligten bei jeglichen Belangen bezüglich Scrum. Das Entwicklungsteam selbst ist dann für die Implementierung der geforderten Funktionen verantwortlich (Greene & Stelman, 2017; Project Management Institute Inc. und Agile Alliance, 2017).

Für das Entwicklungsteam selbst, gibt es, abgesehen von der Selbstorganisation und Interdisziplinarität, erst wenn mehrere Teams involviert sind, zwei Formen der Teamstruktur, welche am öftesten Verwendung finden. Die sogenannten Component Teams und Feature Teams (Rubin, 2013). Als Feature wird ein Verhalten eines Systems gesehen, welches direkt einen bestimmten Kundennutzen erfüllt. Eine Komponente stellt hingegen einen Teil des Systems dar, welcher allgemeine Funktionalitäten kapselt, die zur Implementierung von Features benötigt werden (Scaled Agile, Inc., 2020b).

In diesem Kapitel werden diese agilen Team Strukturen inklusive deren Szenarien, wann diese zum Einsatz kommen sowie deren Vor- und Nachteile vorgestellt.

### 3.1. Feature Teams

Wie es der Name suggeriert, arbeiten Feature Teams an kompletten Features. In der Schichten Architektur einer Software bedeutet das somit, dass das Team in allen Schichten eingreift, damit es das komplette Feature entwickeln kann. Dazu muss das Team zumindest pro Schicht eine Spezialistin/einen Spezialisten im Team inkludieren (Wintersteiger, 2018). In Abbildung 3.1 kann zum Beispiel eine Schicht mit einer Komponente gleichgesetzt werden. Das Gesamte Team bildet dann eine Sammlung an Spezialistinnen/Spezialisten ab, welche jedoch im Idealfall alle in allen Schichten entwickeln sollen und sich durch die Expertise der Mitglieder weiterbilden (Larman & Vodde, 2010). Feature Teams sollen zudem langlebig sein und interdisziplinär arbeiten. Sie entwickeln ein Feature nach dem anderen, welches sie aus dem Backlog abarbeiten (Larman & Vodde, 2008).



**Abbildung 3.1.:** Schematische Darstellung eines Feature Teams (vgl. The LeSS Company B.V., 2020a)

Feature Teams kann man in folgenden Szenarien gut einsetzen (KnowledgeHut, 2020b; Putman, 2020; Software Quality Lab GmbH, 2020):

- Wenn es nur ein Produkt gibt
- Wenn die Produkte sehr groß sind
- Wenn wenige unterschiedliche Technologien verwendet werden
- Wenn die EntwicklerInnen in allen Schichten gut arbeiten können
- Wenn die Features groß genug sind, um alle Schichten zu benötigen
- Wenn die Features klein genug sind, um von einem Team abgeschlossen zu werden
- Wenn das Hauptziel ist, die Einführungszeit zu minimieren
- Wenn das Design für zukünftige Komponenten nicht zuversichtlich erstellt werden kann
- Wenn die Hauptpriorität der Kundennutzen ist

Durch die Herangehensweise von Feature Teams, liegt der Hauptfokus hier auf dem Kundennutzen. Diese Teams generieren schneller einen Nutzen für die Kundin/den Kunden. Dies ist ein Vorteil von Feature Teams, der unter anderem durch die Unabhängigkeit der Teams entsteht (McKenna, 2016). Es muss keine Abstimmungen zwischen Teams erfolgen, wodurch hier Zeit gespart wird. Außerdem müssen die Teams nicht aufeinander warten, um ein Feature komplett fertig zu stellen. Durch diese Beschleunigung in den Entwicklungszyklen, ergeben sich auch schnellere Feedback-Schleifen mit der Kundin/dem Kunden (Emanuelsson, Lagerberg, Sandahl, Skude & Ståhl, 2013). Ein weiterer Vorteil von Feature Teams ist, dass alle Teammitglieder mehr lernen, da sie in allen Schichten mitarbeiten müssen und in ihrem Team eng mit anderen Spezialistinnen/Spezialisten zusammenarbeiten. Zudem stellt diese Form der Strukturierung auch eine hohe organisatorische Flexibilität dar, da selbst, wenn ein Feature nur eine Schicht betrifft, das gesamte Team daran arbeiten kann und keinen Stillstand erzeugt (Larman & Vodde, 2008).

Zu den Nachteilen dieser Form von Teams zählt vor allem die Tatsache, dass jedes Mitglied in allen Schichten arbeiten können soll. Somit muss jedes Mitglied das Können und auch den Willen dazu haben, dies zu tun. Dies stellt sich jedoch als schwierig heraus, dieserart EntwicklerInnen zu finden und einzustellen (Putman, 2020). Aufgrund der Aufteilung in Features, teilt sich auch die Verantwortung für das Design der einzelnen Schichten auf mehrere Teams auf. Zudem arbeitet man auch gleichzeitig an den verschiedenen Schichten, wodurch es zu Problemen kommen kann (The LeSS Company B.V., 2020a). Ein weiterer Nachteil ist, die ständige Konfrontation der

Teams mit neuen fachlichen Themen, da sie immer an den gesamten Features arbeiten und diese meist fachliche Unterschiede aufweisen. Dadurch fällt der Aufbau einer fachlichen Kompetenz schwer. Es kann auch dazu führen, dass die verschiedenen Teams unabsichtlich Funktionalitäten mehrfach entwickeln, da hier keine Absprache unter den Teams statt findet (Software Quality Lab GmbH, 2020).

## 3.2. Component Teams

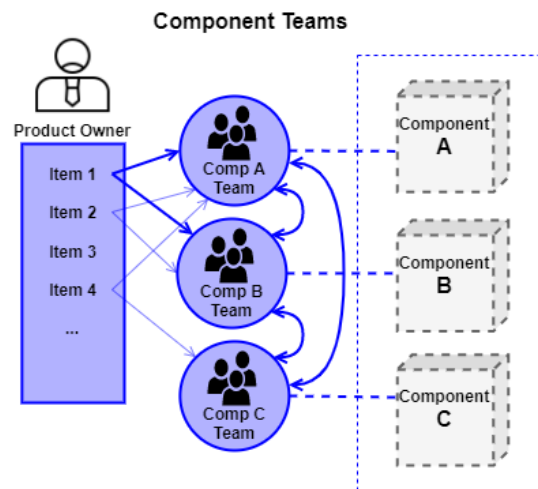
Im Vergleich zu Feature Teams, werden Component Teams nach einer oder mehreren Komponenten getrennt. Eine Komponente, ist wie bereits beschrieben, ein getrennter Teil eines Systems, welcher Funktionalitäten zur Implementierung eines Features beinhaltet. Dies kann auch eine ganze Schicht sein. So teilen sich Features auf mehrere Schichten und somit auch Component Teams auf. In Abbildung 3.2 ist das grafisch dargestellt. Hier sieht man auch die dadurch implizierte Abhängigkeit der einzelnen Teams untereinander (The LeSS Company B.V., 2020a).

Die Component Teams ähneln mehr dem klassischen Ansatz, als es Feature Teams tun (Visual Paradigm, 2020). Hier wird oft das Gesetz von Conway erwähnt, das besagt, dass die Organisation des Unternehmens mit der Struktur der Software korreliert (Baldwin, MacCormack & Rusnak, 2011).

Bei den Component Teams befinden sich in jedem Team nur Mitglieder, welche Expertinnen/Experten auf ihrem Gebiet sind und auf deren Komponente/n und deren Technologie/n spezialisiert sind. Daraus resultieren zumeist robuste, zuverlässige Komponenten, welche die Wiederverwendung und die Testbarkeit erhöhen (Visual Paradigm, 2020). Diese Komponenten sind jedoch völlig belanglos für die Kundin/-den Kunden, da diese/dieser Features und nicht Komponenten kauft. Dennoch sollen auch in Komponenten Teams die Mitglieder interdisziplinär arbeiten, damit sie den kompletten Lebenszyklus einer Komponente betreuen können (KnowledgeHut, 2020b).

Folgende Szenarien eignen sich besonders für Component Teams (KnowledgeHut, 2020b; Putman, 2020; Software Quality Lab GmbH, 2020):

- Wenn es wenig Produkte gibt, im Idealfall nur eines
- Wenn ein generelles Service entwickelt wird, welches von mehreren Teams und/oder Features verwendet werden kann
- Wenn die einzelnen Schichten eine hohe technische Komplexität aufweisen
- Wenn die Features so klein sind, dass sie nur eine Schicht betreffen



**Abbildung 3.2.:** Schematische Darstellung eines Component Teams (vgl. The LeSS Company B.V., 2020a)

- Wenn es viele unterschiedliche Technologien gibt, welche Spezialistinnen/Spezialisten erfordern
- Wenn die EntwicklerInnen nicht in allen Schichten arbeiten können beziehungsweise wollen
- Wenn die Einführungszeit passt
- Wenn das Design für die Integration zukünftiger Komponenten zuversichtlich erstellt werden kann
- Wenn man gut voraus planen kann, um eine gleichmäßige Arbeitslast sicher zu stellen

Einer der größten Vorteile der Component Teams ist jener, dass die Teams ausschließlich aus Expertinnen/Experten der einzelnen Schichten und dessen Technologien bestehen. Dadurch können sich die EntwicklerInnen auf einzelne Technologien und Schichten konzentrieren und daher kann eine effiziente Arbeitsweise ermöglicht werden (Software Quality Lab GmbH, 2020). In Folge dessen ergibt sich auch eine viel geringere Fehleranfälligkeit der Komponenten. Das ergibt sich daher, da Spezialistinnen/Spezialisten eher weniger dazu neigen, deren eigenen Code zu zerstören (Putman, 2020). Durch die Trennung der Teams in die einzelnen Schichten, identifizieren sich diese auch besser mit ihrer Schicht. Es kommt zu keinen Konflikten des Codes zwischen den Teams. Dadurch übernehmen sie dafür auch mehr Verantwortung. Es ist prinzipiell schwierig, die Verantwortung für etwas zu tragen, woran andere auch teilhaben und dies eventuell auch zerstören können (Larman & Vodde, 2008). Aus diesen Teams resultieren somit Komponenten beziehungsweise Schichten, dessen Funktio-

nalität leicht von anderen Teams wiederverwendet werden kann (Visual Paradigm, 2020).

Durch die Strukturierung der Teams in Component Teams, verlangsamt sich vermeintlich die Entwicklung. Dies beruht auf der Tatsache, dass die einzelnen Teams für ein komplettes Feature sehr viel mehr Schnittstellen besprechen müssen und dadurch auch ein höherer Koordinationsaufwand entsteht (Bick, Heinzl, Scheerer & Schimmer, 2015). Zumeist führen diese Teams aufs Neue zur klassischen Wasserfall-Methode, bei der die einzelnen Teile der Features nacheinander bzw. sequentiell abgearbeitet werden. Dies führt schlussendlich auch zu Wartezeiten zwischen den Teams und erschwert die Ressourcenplanung, da es nicht immer möglich ist, alle Teams im selben Ausmaß auszulasten (Larman & Vodde, 2010). Zudem gibt es kein Team, welches die Hauptverantwortung über das gesamte Produkt hat. Auch der Hauptfokus liegt dabei nicht auf dem Kundennutzen, wodurch es oftmals dazu führt, dass vorrangig einfache Entwicklungen durchgeführt werden, als jene die der Kundin/dem Kunden einen Nutzen bringen. Außerdem ist es schwierig ein schichtenübergreifendes Design zu erstellen, was häufig zu schlechterer Code Qualität führt und auch zu doppelt entwickelten Funktionalitäten (Software Quality Lab GmbH, 2020).

### **3.3. Zusammenfassung**

Wie auch bei den klassischen Modellen, gibt es bei den agilen Methoden nicht die eine richtige Form, welche für alle Situationen passt. Hier nennt man zumindest zwei verschiedene Modelle. Durch die zuvor erwähnten Vor- und Nachteile und der Einsatzmöglichkeiten der verschiedenen Formen, kann man jedoch für sich selbst abwägen, welches Modell sich besser eignet. Larman und Vodde (2010) verfassen eine Gegenüberstellung beider Modelle, welche in Tabelle 3.1 dargestellt ist.

Feature Teams	Component Teams
Optimierung für die Lieferung von maximalem Kundennutzen	Optimiert für die Lieferung von einer maximalen Anzahl an Codezeilen
Fokussiert auf Features mit hohem Kundennutzen und System Produktivität (Nutzen Durchsatz)	Fokussiert auf erhöhte Produktivität des Einzelnen durch die Implementierung von „leichten“ Features mit weniger Nutzen
Verantwortlich für komplette kundenzentrierte Features	Verantwortlich für einen Teil eines kundenzentrierten Features
„Moderne“ Art Teams zu organisieren - vermeidet das Gesetz von Conway	Traditioneller Ansatz der Organisation von Teams - folgt dem Gesetz von Conway
Führt zu Kunden Fokus, Sichtbarkeit und kleineren Organisationen	Führt zu „erfundener“ Arbeit und einer für immer wachsenden Organisation
Minimiert die Abhängigkeiten zwischen den Teams, um Flexibilität zu erhöhen	Abhängigkeiten zwischen den Teams führt zu erhöhtem Planungsaufwand
Fokussiert auf mehrfache Spezialisierungen	Fokussiert auf eine Spezialisierung
Produkt Code ist geteiltes Eigentum	Code ist Eigentum einer/eines Einzelnen/Teams
Geteilte Teamverantwortlichkeiten	Klare Verantwortlichkeiten der/des Einzelnen
Unterstützt iterative Entwicklung	Resultiert in „Wasserfall“ Entwicklung
Beutet die Flexibilität aus; Kontinuierliches und breites Lernen	Beutet die bestehende Expertise aus; Niedrigeres Level neuer Fähigkeiten zu lernen
Erfordert erfahrene Entwicklungsmethoden - Auswirkungen sind weit verbreitet sichtbar	Funktioniert mit unordentlichen Entwicklungsmethoden - Auswirkungen sind lokal
Liefert die Motivation, Code einfach testbar und wartbar zu machen	Entgegen dem Glauben, führt es oft zu minderwertigem Code in den Komponenten
<i>Scheinbar</i> schwierig zu implementieren	<i>Scheinbar</i> einfach zu implementieren

**Tabelle 3.1.:** Feature Teams versus Component Teams (vgl. Larman & Vodde, 2010)

Zusätzlich zur Unterscheidung der beiden agilen Strukturen, nennen Larman und Vodde (2010) auch Unterschiede zwischen Feature Teams und konventionellen Projekt oder Feature Gruppen, welche in Tabelle 3.2 abgebildet sind.

Die zuvor genannten Teamstrukturen stellen jedoch nur die zwei am meisten verwendeten Strukturen dar. Software Quality Lab GmbH (2020) nennt zudem auch noch weitere Strukturen wie das Produktteam oder Mischformen. Wobei ersteres auch der



Feature Team	Feature Gruppe oder Projekt Gruppe
Stabiles Team welches für Jahre bestehen bleibt und an vielen Features arbeitet	Temporäre Gruppe von Personen für die Erstellung eines Features oder Projektes
Geteilte Teamverantwortlichkeiten für die gesamte Arbeit	Individuelle Verantwortlichkeiten für „dessen“ Aufgabe basierend auf der Spezialisierung
Selbst organisiertes Team	Kontrolliert von einer/einem ProjektmanagerIn
Resultiert in einer einfachen Ein-Linien Organisation	Resultiert in einer Matrix-Organisation mit Ressourcen Pools
Teammitglieder sind zu 100% dem Team zugehörig	Teammitglieder sind Teilzeit in mehreren verschiedenen Projekten aufgrund derer Spezialisierung

**Tabelle 3.2.:** Unterschiede von Feature Teams zu konventionellen Projekt beziehungsweise Feature Gruppen (vgl. Larman & Vodde, 2010)

Definition in dieser Arbeit vorgeschlagenen Modells entspricht und zweiteres eine Mischform aus verschiedenen agilen Strukturen darstellt. Die Mischform leitet auch weiter zu den agilen Skalierungs-Frameworks. In diesen wird beschrieben, wie man mit mehreren Teams effektiv planen und arbeiten kann. Diese werden in Kapitel 4 vorgestellt.

## 4. Skalierungs-Frameworks

Scrum hilft dabei, dass man mit relativ kleinen Teams in kurzen Entwicklungszyklen den maximalen Kundennutzen erzielen kann. Zudem ist auch ein wichtiger Aspekt, dass die Zeit bis zur Markteinführung des Produktes so kurz wie möglich sein soll. Das sind laut Scrum Alliance (2017) die wichtigsten Gründe, um Scrum zu verwenden. Wenn die Produkte oder Features jedoch zu groß werden, um von einem Team bearbeitet werden zu können, benötigt man übergeordnete Organisationen (The LeSS Company B.V., 2020c). Bittner et al. (2018) versucht mit dem Framework Nexus die Frage zu klären, wie man große Produkte mit Scrum sinnvoll, unter der Verwendung von mehreren Teams, entwickeln kann. Dies ist jedoch nur eines von vielen agilen Skalierungs-Frameworks. Laut CollabNet Inc. und VersionOne Inc. (2019), Scrum.org und Age of Product (2019) sind die am meisten verwendeten Frameworks:

- Scaled Agile Framework (SAFe)
- Large-Scale Scrum (LeSS)
- Nexus
- Scrum of Scrums (SoS)

Wobei letzteres kein dediziertes Framework ist, sondern eher eine Herangehensweise für die Kommunikation verteilt über mehrere Scrum Teams. Dies ist laut Agilest LLC (2020d) die älteste und meist verwendete Methoden einer Skalierung von Scrum. In den folgenden Abschnitten werden die zuvor genannten Frameworks vorgestellt. Ein besonderer Fokus liegt in dieser Arbeit auf deren Teamstrukturen.

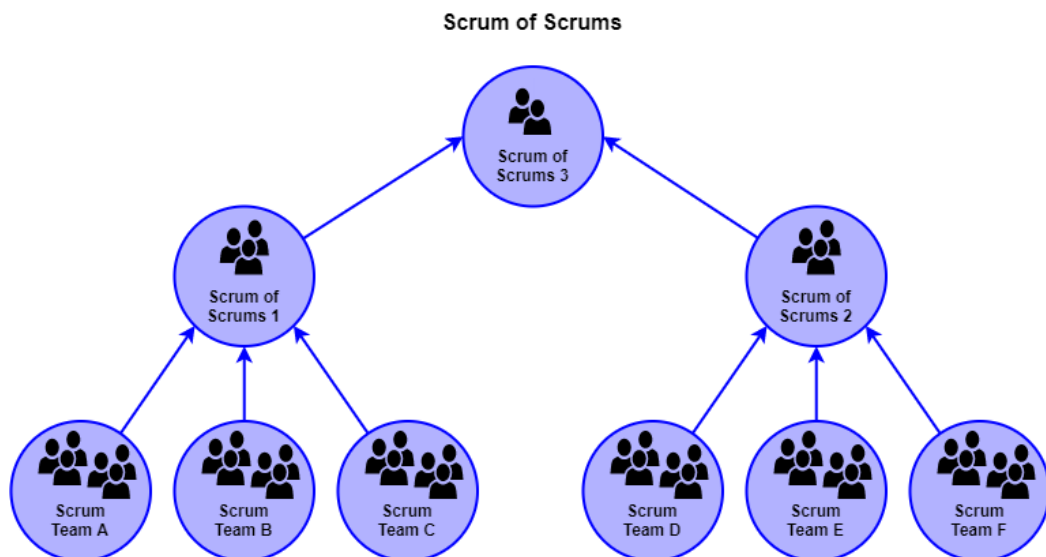
### 4.1. Scrum of Scrums (SoS)

SoS ist die älteste und am meisten angewandte Herangehensweise wie man agile Teams skalieren kann. Viele Unternehmen verwenden diesen Ansatz als ersten Schritt, um die Skalierung mehrerer agiler Teams zu bewerkstelligen und um die Erstellung von größeren und komplexen Produkten zu gewährleisten (Spanner, 2020).

Eines der Ereignisse, welche bei Scrum im Einsatz sind, ist das Daily Scrum. Dies ist ein Meeting bei dem sich die Teammitglieder regelmäßig, zumeist täglich, für maximal 15 Minuten treffen, um drei Fragen zu beantworten (Schwaber & Sutherland, 2017):

- Was habe ich seit dem letzten Meeting erledigt?
- Was werde ich bis zum nächsten Meeting erledigen?
- Wo habe ich Probleme oder bin ich blockiert?

SoS beschreibt schwerpunktmäßig ein Daily Scrum auf einem höheren Level, wie in Abbildung 4.1 dargestellt. So generiert man eine mehrstufige Hierarchie aus agilen Teams, wobei jedes Team seinen Scrum Master hat. Bei SoS gibt es, ähnlich wie bei Scrum, das Daily Scrum Meeting. Dies ist bei SoS ein regelmäßiges Meeting, wo sich die VertreterInnen der einzelnen Teams, zumeist die Scrum Master, treffen und versuchen dieselben drei Fragen zu beantworten, wie in den einzelnen Teams. Diese Fragen sind jedoch nicht auf jedes einzelne Individuum bezogen, sondern auf das gesamte Team, welches sie vertreten. Bei technischeren Belangen, kann es durchaus vorkommen, dass zusätzlich zum Scrum Master ein Teammitglied teil nimmt (Pries & Quigley, 2011).



**Abbildung 4.1.:** Schematische Darstellung von mehreren Teams in der Herangehensweise von Scrum of Scrums (vgl. Agilest LLC, 2020d)

Ziel von SoS ist es, die Hindernisse der Teams untereinander zu minimieren und einen Report auf Team Level zu erlangen, der dieselben Fragen wie ein normales Daily Scrum beantwortet. Dies soll dazu beitragen, die Koordination zwischen den Teams zu verbessern und somit als endgültiges Ziel, die Erstellung eines Produktes am Ende eines jeden Sprints zu bewerkstelligen. Dies nicht nur für die einzelnen Teams, sondern für alle Teams als gemeinsames Produkt (KnowledgeHut, 2020a).

Zusammengefasst lässt sich sagen, dass SoS beschreibt, wie man mehrere agile Teams koordinieren kann, sodass am Ende eines jeden Sprints ein potentiell fertiges Produkt aus der Zusammenarbeit aller Teams entsteht. Dies wird jedoch nur auf einer sehr rudimentären Ebene beschrieben, indem es ein wiederkehrendes Meeting gibt, in welchem sich VertreterInnen der einzelnen Teams abstimmen. SoS erwähnt jedoch nicht, wie sich die Teams zusammensetzen sollen. Weder die Struktur der einzelnen Teams noch das Gesamtbild werden beschrieben. So kann man von SoS weder ableiten ob Teams zum Beispiel als Feature Teams oder Component Teams strukturiert werden sollen, noch ob alle Teams derselben Struktur folgen müssen, oder eine Mischform mit einem eventuell bestimmten Verhältnis vorliegen soll.

### 4.2. Scaled Agile Framework (SAFe)

Im Vergleich zu SoS stellt SAFe nicht nur eine Herangehensweise dar, sondern ein komplettes Framework. Laut Böhm (2019) ist es das bekannteste Framework. Lang und Scherber (2019) schreiben, dass es ein großes gesamtes Bild liefert, mit einem sehr umfangreichen Werkzeugkoffer, welcher zu Gestaltung einer agilen Organisation verwendet werden kann. Es wird hauptsächlich von großen Konzernen und Unternehmen eingesetzt.

Im Kern teilt sich das Framework in drei Ebenen auf (Scaled Agile, Inc., 2020c). Die Portfolio Ebene, auf welcher die Vision, die Ziele und Strategien des Unternehmens gemanagt werden. Das Programm Level, in welchem eines der zentralen Konzepte von SAFe verwendet wird, die sogenannten „Agile Release Train (ART)“ (Leffingwell, 2007). Diese stellen ein Konzept dar, um ein Ergebnis zu liefern, welches von einem spezifischen Projekt erwartet wird. Dazu sind alle ART zu einem spezifischen „Value Stream“ zugeordnet, welcher ein Ziel darstellt (Agilest LLC, 2020c). Ein ART generiert alle drei Monate ein Produktinkrement. Innerhalb dieser drei Monate werden dennoch Sprints abgehalten, welche zumeist zwei Wochen dauern. Diese Sprints werden von mehreren Teams abgearbeitet. Jeder ART soll, wie auch die einzelnen Teams selbst, interdisziplinär aufgestellt sein um alle Tätigkeiten erledigen zu können um ein Feature für die Kundin/den Kunden am Ende eines ART liefern zu können. Dazu müssen alle Funktionen an Teams vorhanden sein um dies zu bewerkstelligen.

In der dritten und somit letzten Ebene beschreibt SAFe die Teamorganisation. In diesem Level ist das Framework annähernd gleich wie Scrum. Es beschreibt auch, dass ein Team aus drei Rollen besteht. Dem Scrum Master, dem Product Owner und dem Entwicklungsteam. Zudem beschreibt es auch für die Strukturierung der Teams hauptsächlich nur die Kriterien, welche auch Scrum nennt. Es soll zwischen drei und neun Mitglieder haben, interdisziplinär aufgestellt und selbstorganisiert sein (Scaled Agile, Inc., 2020a).

Knaster und Leffingwell (2019) beschreiben in ihrem Werk unter anderem wie man einen ART implementiert. Hier gehen sie auch auf die Organisation der agilen Teams ein. Sie nennen zwei Arten von Teams. Die Feature Teams und die Component Teams, welche sie gleich definieren wie auch Scrum sie definiert. Feature Teams sind auch bei SAFe die bevorzugte Variante der Strukturierung, da das Team fähig ist, einen kompletten Kundennutzen innerhalb des Teams zu liefern. Bei den Component Teams soll man sich laut Knaster und Leffingwell (2019) auf signifikante Wiederverwendungsmöglichkeiten, Bereiche welche höchst technisches Spezialwissen verlangen und kritische nicht funktionale Anforderungen limitieren. Knaster und Leffingwell (2019) erwähnen zudem, dass man vermeiden soll, Teams nach den Schichten der Architektur zu strukturieren.

Die meisten ART bestehen aus einem Mix der beiden genannten Strukturen. Bei SAFe wird oft eine Mischung von 75-80% Feature Teams und 20-25% Component Teams gesehen. Es gibt zwei Gründe, auf Basis derer diese Mischung beruht. Die technische Spezialisierung, welche von den einzelnen Komponenten benötigt wird und die Wirtschaftlichkeit einer möglichen Wiederverwendung. Für die genaue Verteilung ist es immer vom gesamten System abhängig, wie viele Feature Teams und wie viele Component Teams Sinn ergeben. Diese Aufteilung soll Teil des agilen Prozesses sein, welche man kontinuierlich inspiziert und dementsprechend anpassen kann, damit es am meisten Sinn ergibt (Scaled Agile, Inc., 2020b).

Zusammengefasst nennt SAFe die beiden Varianten der Teamstrukturen, welche auch Scrum nennt und die Einsatzmöglichkeiten dafür. Es wird auch erwähnt, dass in der Praxis zumeist eine Mischung der beiden Strukturen mit einem gewissen Verhältnis gesehen wird. Dennoch gibt es keine konkreten Angaben, sondern eher Empfehlungen und die Verteilung der einzelnen Teams soll im agilen Prozess kontinuierlich überwacht und angepasst werden.

### **4.3. Large Scale Scrum (LeSS)**

LeSS ist im Grunde genommen sehr ähnlich wie Scrum, nur etwas erweitert. So ist LeSS auch wie Scrum, ein Framework, welches mit Hilfe der Organisation die Prozesse inspizieren und adaptieren kann, damit es mit kleinen oder großen Gruppen arbeiten kann. LeSS ist ein Label für Scrum in Kombination mit einigen zusätzlichen Tipps, welche von Larman und Vodde (2010) während ihrer Arbeit mit Scrum in großen Multi-Team, Multi-Standort und auch ausländischen agilen Entwicklungen gesammelt werden.

Die Basis Variante von LeSS beschreibt die Skalierung für bis zu zehn Teams. Alles darüber ist in einem eigenen Framework „LeSS Huge“ abgebildet. Wie bereits erwähnt, funktioniert LeSS prinzipiell sehr ähnlich wie Scrum (Böhm, 2019). Ein Unterschied ist, dass alle Teams gemeinsam einen Product Owner haben, und nicht jedes

Team einzeln. Selbst wenn es mehrere Product Owner gibt, sollte es immer einen „Chef“ Product Owner geben, der die endgültige Entscheidung trifft. Das Ziel von LeSS ist es, am Ende eines Zyklus mit den Ergebnissen aus allen Teams ein großes auslieferbares Produkt Inkrement zu erstellen.

Auch bei den agilen Meetings ist es sehr ähnlich wie Scrum. Es gibt auch die Daily Scrum, Sprint Planning und Sprint Retrospektive Meetings, welche alle parallel in allen Teams durchgeführt werden. Wobei das Sprint Planning und das Sprint Retrospektive Meeting jeweils in zwei Stufen abgehalten wird. Ähnlich zum SoS gibt es hier bei den beiden Meetings auch einen zusätzlichen Teil, in dem die VertreterInnen der einzelnen Teams an den Meetings beteiligt sind und den normalen Teil von Scrum, bei dem nur das Team selbst beteiligt ist. So wird beim Sprint Planning zuerst mit den Vertreterinnen/Vertretern der Teams abgestimmt, welche Teile des Product Backlog im nächsten Sprint umgesetzt werden. Im zweiten Teil beschließt jedes Team einzeln wie die Anforderungen umgesetzt werden und bildet dessen eigenen Sprint Backlog. Bei der Retrospektive wird zuerst das Team-interne Meeting normal wie bei Scrum abgehalten. Danach gibt es eine Retrospektive, bei der wiederum die Vertretungen der Teams beteiligt sind. Hier werden Probleme identifiziert, welche nicht in den Teams selbst gelöst werden können (Agilest LLC, 2020b).

Die Rollen sind in LeSS auch dieselben wie in Scrum. So gibt es den Scrum Master, den Product Owner und das Entwicklungsteam. Alle haben prinzipiell dieselben Aufgaben und Verantwortungen wie auch in Scrum. Daher gibt es auch für die Strukturierung von Teams keine neuen Anforderungen durch LeSS (The LeSS Company B.V., 2020b). Larman und Vodde (2010) schreiben wie bei Scrum, dass ein Team prinzipiell selbstorganisiert, interdisziplinär, langlebig und an einem Standort sein soll. Bezüglich der Strukturierung als Feature Teams oder Component Teams plädieren sie stark darauf, von Component Teams weg und zu Feature Teams hin zu gehen. Laut (Larman & Vodde, 2008) haben Component Teams sehr viele Nachteile, welche durch Feature Teams beinahe auszumerzen sind. So ist es laut deren Meinung Pflicht, in Feature Teams zu organisieren damit die Vorteile, welche durch agile Herangehensweisen beschrieben werden, auch erreicht werden können.

Zusammengefasst ist LeSS eines der agilsten Skalierungs-Frameworks, welches sehr gleichartig zu Scrum ist. Es ist auch ähnlich wie Scrum sehr wenig präskriptiv, wodurch einige Lücken in der Regel von der Organisation gefüllt werden müssen. Dies betrifft unter anderem auch die konkrete Strukturierung der Teams. Hier tendieren sie stark zu den von Scrum vorgeschlagenen Feature Teams, da diese Teams End-to-End Features für die Kundin/den Kunden entwickeln können und das mit geradezu minimalem Aufwand an Kommunikation und Koordination (Lang & Scherber, 2019).

## 4.4. Nexus

Das Nexus Framework wird oft als „Exoskelett“ bezeichnet, welches sich um Scrum herum legt. Es soll dabei helfen, die schwierigen Herausforderungen und die komplexen Produkte zu bewegen, welche ein Scrum Team alleine nicht schafft (Böhm, 2019). Als zentrales Ziel setzt es sich die Verringerung der Abhängigkeiten zwischen den Teams. So funktioniert das Nexus Framework mit mindestens drei bis maximal neun Scrum Teams (Lang & Scherber, 2019).

Ähnlich wie LeSS versucht auch das Nexus Framework am Ende jeder Iteration ein gemeinsames Produkt über mehrere Teams hinweg zu erstellen. So bleibt auch hier der Aufwand für die Koordination der Integration der einzelnen Teams bestehen. Dies löst das Nexus Framework durch die Abänderung von Scrum. Es funktioniert im Kern genau wie Scrum und hat auch mehrere Scrum Teams. Zusätzlich gibt es ein Artefakt, den Nexus Sprint Backlog. Dieser macht die Zusammenarbeit aller Teams sichtbar und transparent. Außerdem werden fünf weitere Ereignisse hinzugefügt, welche statt den einzelnen Teams den Inhalt aller Teams gemeinsam betreffen. Es wird ein Ereignis entfernt, das Sprint Review. Dies wird somit nicht mehr von jedem Team einzeln abgehalten, sondern wird in einem gemeinsamen Nexus Sprint Review über alle Teams hinweg durchgeführt. Bei den Rollen gibt es auch im Nexus dieselben Rollen wie bei Scrum. So gibt es den Scrum Master, den Product Owner und das Entwicklungsteam (Bourk & Kong, 2016; Schwaber & Scrum.org, 2018). Zusätzlich führt Nexus noch ein sogenanntes „Nexus Integration Team“ ein. Dies ist nicht für die Integration verantwortlich, sondern dafür, allen Teams dabei zu helfen, die Integration selbstständig durchzuführen. Dieses Team besteht aus einem Scrum Master, einem Product Owner und einem Entwicklungsteam. Wobei alle der TeilnehmerInnen auch in anderen bestehenden Scrum Teams arbeiten (Agilest LLC, 2020a).

Für die Strukturierung von Teams fühlt es sich natürlich an, Teams aufgrund ihres Fachwissens auf spezielle Komponenten zu verteilen. Das ist jedoch nicht der einzige und laut Bittner et al. (2018) auch nicht der beste Weg, um Teams zu strukturieren. Teams können auf viele verschiedenen Ebenen strukturiert werden, wie zum Beispiel auf Grund von geografischen Orten, Features, Komponenten, Personas oder Technologie-Stacks. Im Idealfall, kann jedes Teammitglied an jeder Anforderung aus dem Product Backlog arbeiten. Zusätzlich ist es noch ideal, wenn jedes Teammitglied an jeglicher Art von Arbeit mitwirken kann. In der Realität sind die meisten Teams jedoch sehr weit von dieser Idealvorstellung entfernt. Somit ist es das Wichtigste, Teams so zu strukturieren, dass sie möglichst unabhängig und parallel zu anderen Teams arbeiten können, während sie kontinuierlich ihre Arbeit mit der Arbeit der anderen Teams zu einem großen gesamten Produkt integrieren (Bittner et al., 2018).

Für die konkrete Strukturierung liefert auch das Nexus Framework keine Antworten. Jedoch tendieren auch Bittner et al. (2018) zu Feature Teams, beziehungsweise Teams, welche je nach Persona aufgeteilt sind. Sie argumentieren dies damit, dass

wenn große Produkte nur eine bis wenige Plattformen und Technologien verwenden, es keinen Sinn ergibt, die Teams als Component Teams oder auf Basis der Technologie oder der Plattform zu strukturieren. Hauptsache bei der Strukturierung ist es, dass die Teams unabhängig und effizient mit wenig Koordinationsaufwand zwischen den Teams und einer schnellen Einführungszeit arbeiten können.

### 4.5. Zusammenfassung

Alle der genannten Skalierungs-Frameworks beziehen sich hauptsächlich darauf, wie man die Arbeit zwischen den einzelnen Teams am besten Koordinieren kann, um am Ende ein gemeinsames integriertes potentiell auslieferbares Produktinkrement zu erstellen. Dazu verwenden sie unterschiedliche Methodiken um dies zu bewerkstelligen. Für die konkrete Strukturierung der einzelnen Teams wird wenig, bis gar keine Information geliefert. Zumeist wird nur darauf verwiesen, dass die Teams unabhängig und mit wenig Koordinations- und Kommunikationsaufwand unter den Teams arbeiten können sollen. Ob man nun seine Teams nach Komponente, Architekturschicht, Feature oder ähnlichem strukturiert, wird nur mit den Vor- und Nachteilen der Organisationsformen argumentiert. Da die meisten Frameworks auch auf Basis von Scrum arbeiten, tendieren diese, wie Scrum selbst, hin zu Feature Teams. Dies wäre die Ideallösung wie man Teams Strukturieren kann. Ein Team, welches einen Kundennutzen von Anfang bis zum Ende eigenständig im Team umsetzen kann. Dies führt dazu, dass es nur wenig bis keine Koordination der einzelnen Teams untereinander benötigen würde. Bei dieser Ansicht wird jedoch fast immer erwähnt, dass solch ideale Teams schwierig zu bewerkstelligen sind, und es zumeist zu einer Mischung von verschiedenen Teams kommt. So beschreibt SAFe eine Mischung aus Component und Feature Teams. Zusammengefasst schreiben jedoch die meisten Frameworks, dass es wie bei Scrum, ein iterativer Lernprozess sein soll. Dieser soll nicht nur in den Teams intern vorherrschen, sondern im gesamten Unternehmen. Dadurch soll auch die Organisationsform der Teams inspiziert und bei Problemen adaptiert werden. Somit ergeben auch die Skalierungs-Frameworks keine neuen Erkenntnisse, durch welche die Frage der Team Struktur getroffen werden kann.



## 5. Vergleich klassischer und agiler Teamstrukturen

In diesem Kapitel werden die vorgestellten agilen und klassischen Teamstrukturen verglichen, um die Frage zu beantworten, ob die Auswahl der Methode (klassisch oder agil) einen Einfluss auf die Team Struktur hat. Durch den Vergleich wird somit die Hypothese der Arbeit beantwortet. Zusätzlich werden in diesem Kapitel die vorgeschlagenen Teamstrukturen dieser Arbeit definiert und vorgestellt. Mit Hilfe der Vergleiche und der vorgestellten Strukturen werden im nächsten Kapitel Vorgehensmuster erstellt und weitere Hypothesen gebildet.

### 5.1. Klassische und agile Teamstrukturen

Klassisches und agiles Projektmanagement unterscheidet sich in vielerlei Hinsichten. Es gibt aber auch Gemeinsamkeiten der beiden Formen (Preußig, 2018). So wie im Management gibt es auch in der Organisation einige Unterschiede, aber auch essentielle Gemeinsamkeiten der beiden Welten. Nachfolgend werden einige Unterschiede aber auch Gemeinsamkeiten beider Herangehensweisen erläutert.

Einer der ersten Unterschiede ist, dass bei den meisten vorgestellten klassischen Organisationsformen die Teams nur temporär vorhanden sind. So werden die Mitglieder dieser Teams nach Abschluss eines Projektes wieder in die Linien zurück eingegliedert oder wieder für neue Projekte formiert, wie zum Beispiel bei der reinen Projektorganisation (Bachmann et al., 2019). Beim agilen Ansatz soll das Team möglichst effizient sein und daher auch langlebig zusammenarbeiten (Scheerer, 2017). Es gibt jedoch auch beim klassischen Ansatz Strukturen, welche die Teams langlebig strukturieren, wie zum Beispiel die Funktionale oder Objektorientierte Organisation. Somit ist dieser Unterschied zwischen klassischen und agilen Teams auch nur bedingt gegeben.

Der größte Unterschied von klassischen zu agilen Strukturen ist die Zuteilung der Arbeitspakete. Dieser zieht sich durch die kompletten vorgestellten Teams im klassischen wie auch dem agilen Ansatz. So sind in der klassischen Welt alle Teams zumindest von einer Führungskraft organisiert. Diese gibt vor, was und wie die Teams es

zu entwickeln haben. Bei einigen der Strukturen sind dies dann je nach personeller und fachlicher Verantwortung mehrere verschiedene Personen. In Multiprojektorganisationen gibt es sogar fachlich mehrere Vorgesetzte, welche die Arbeitspaket Zuteilung für die einzelnen Teammitglieder vornehmen (Broy & Kuhrmann, 2013). In den agilen Strukturen ist eines der Prinzipien für die Teams, die Selbstorganisation. Das heißt, das Team entscheidet selbst, wie viel und was es aus dem Backlog für die nächste Iteration zur Entwicklung nimmt und auch wie es umgesetzt wird. Hierzu wird lediglich der Product Owner benötigt, der dafür sorgt, dass der Backlog stetig gefüllt und priorisiert ist (Schwaber & Sutherland, 2017).

Ein weiterer Unterschied ist die Eingliederung des Teams in die Organisation. So sind bei den klassischen Strukturen zumeist die Teams in Linien organisiert und es werden Teammitglieder aus den verschiedenen Linien für Projekte zusammengesammelt. Diese so gebildeten Teams bestehen jedoch zumeist nur virtuell und sind nicht als Organisationseinheit im Organigramm abgebildet. Wie bereits beschrieben, werden diese Teams nach einem Projekt zumeist wieder aufgelöst, wodurch es auch sinnvoll ist, dies so zu handhaben. In der agilen Welt wird selten erwähnt, wie die Organisation darüber selbst aussehen soll. Jedoch ist klar, die Teams sollen immer als eigenständige Organisationseinheit gesehen werden. So sind die Teams nicht nur virtuell und für ein Projekt vorhanden, sondern als eine Einheit und dies für den gesamten Lebenszyklus.

Die nun genannten Verschiedenheiten machen nur einen kleinen Teil der Unterschiede zwischen den agilen und klassischen Strukturen aus. Essentiell für die Frage dieser Arbeit ist jedoch auch, was die beiden Modelle gemeinsam haben. Hierfür könnte man auch mehrere Gemeinsamkeiten nennen, jedoch ist eine Gemeinsamkeit ausschlaggebend für die Beantwortung der Arbeitshypothese.

Bei den klassischen als auch bei den agilen Strukturen steht man zu Beginn mit einer Anzahl an Personen da, welche bestimmte Kenntnisse haben. Diese sind in der Regel Spezialistinnen/Spezialisten in zumindest einem Fachgebiet oder einer Technologie. So stellt sich sowohl bei den klassischen, als auch bei den agilen Strukturen die Frage, wie man die Personen gruppiert und diese nach Spezialgebieten verteilt. Diese Frage spiegelt die Forschungsfrage dieser Arbeit wider, ob Teams projektübergreifend jedoch im selben Fachbereich arbeiten sollen, oder ob die Mitglieder in Teams fachlich übergreifend am selben Projekt arbeiten sollen. Mit dieser Definition können somit Component Teams, die Linienorganisation und die Stab-Linienorganisation auf die eine Seite und Feature Teams und alle abteilungsübergreifenden Strukturen der klassischen Welt auf die andere Seite gestellt werden.

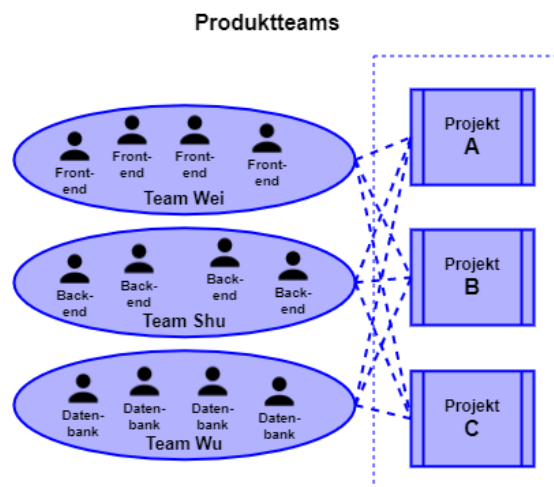
Aufgrund dieser Einschränkung wird somit die Hypothese  $H_0$ : „Die Entscheidung, ob ein Projekt agil oder klassisch gemanagt wird, ist *ausschlaggebend* für die Entscheidung der Strukturierung eines Teams“ verworfen.

Dadurch stehen die in Abschnitt 5.2 und 5.3 vorgestellten Teams nicht nur für die agilen sondern auch für die klassischen Modelle. In dieser Arbeit soll jedoch das

Hauptaugenmerk auf den agilen Strukturen bleiben. Anschließend werden die beiden Teamstrukturen für diese Arbeit im Detail vorgestellt.

## 5.2. Produktteams

Als Produktteam wird ein Team bezeichnet, welches projektübergreifend, jedoch im selben Fachbereich arbeitet. Dies kann ähnlich zu den Component Teams gesehen werden. In dieser Arbeit steht somit der Fachbereich als Synonym für Komponente, Architektur Schicht, Fachbereich, Technologie oder ähnliches. Im Grunde genommen alle Teile eines Systems, welche Integration mit anderen Teilen benötigt um am Ende einen kompletten Kundennutzen zu generieren. Hingegen steht ein Projekt in dieser Arbeit als Synonym für ein Produkt, Projekt, Feature oder ähnliches. Ein Teil eines Systems, welcher abgeschlossen ist und somit einen kompletten Kundennutzen alleinstehend erzeugt ohne zwingende Integration mit anderen Systemteilen. Abbildung 5.1 zeigt eine schematische Darstellung dieser Teamstruktur. Die Teams bestehen somit aus Mitgliedern, welche Spezialistinnen/Spezialisten in ihren Fachbereichen sind.



**Abbildung 5.1.:** Schematische Darstellung eines Produktteams (Quelle: eigene Darstellung)

In den Produktteams identifizieren sich die Teammitglieder gut mit ihrem Fachbereich (Larman & Vodde, 2008). Dies liegt unter anderem daran, dass die Teammitglieder lediglich in ihrem Fachbereich tätig sind. Das führt auch zu einer erhöhten Spezialisierung in den einzelnen Bereichen. Die Mitglieder sind somit hauptsächlich spezialisiert auf ihre Fachbereiche, im besten Fall ist es nur einer. Diese weisen hier jedoch eine enorme Expertise auf. Sie können somit sehr effizient in ihren Bereichen tätig sein. Aufgrund der Tatsache, dass die Fachbereiche Teile eines Systems entwickeln, welche später in ein gesamtes System integriert werden, können diese auch von mehreren anderen Bereichen verwendet werden. Dies fördert die Entwicklung

von wiederverwendbaren Teilen (KnowledgeHut, 2020b). Die Mitglieder in diesen Teams sollen somit Spezialistinnen/Spezialisten in ihren Bereichen sein, jedoch benötigen sie nicht zwingend die Fertigkeiten um in allen anderen Bereichen auch arbeiten zu können.

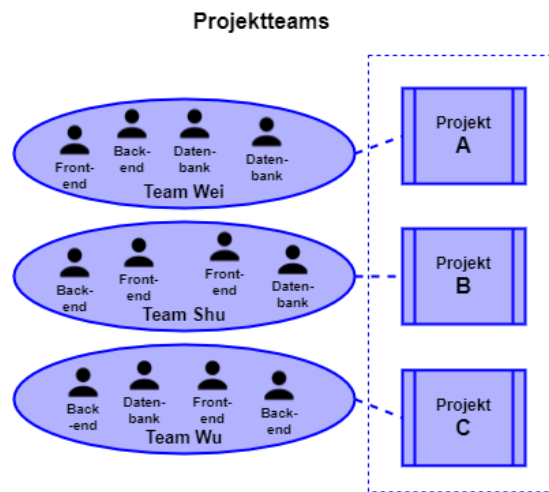
Negativ an den Produktteams ist, dass sie ständige Integration mit anderen Fachbereichen benötigen um einen Kundennutzen zu generieren (Bick et al., 2015). So liegt der Hauptfokus bei diesen Teams zumeist nicht auf dem Kundennutzen selbst, sondern auf deren eigenen Fachbereichen. Das kann dazu führen, dass Teile entwickelt werden, welche nicht wichtig für die Kundin/den Kunden sind, jedoch vom Team gerne entwickelt werden würden. Zudem sind alle EntwicklerInnen Spezialistinnen/Spezialisten in demselben Fachbereich, was dazu führt, dass sie wenig in neuen Fachbereichen lernen. Jedoch führt es zu einer noch besseren Expertise, da sich die Experten untereinander austauschen können.

Auch einer der Nachteile dieser Teams liegt in der Ressourcenauslastung. Wenn der Backlog nicht gleich verteilt gefüllt werden kann, dass alle Fachbereiche beschäftigt werden können, kann es zu Stehzeiten bei gewissen Teams kommen. Zudem besteht in dieser Struktur auch ein erhöhter Koordinations- und Kommunikationsaufwand zwischen den Teams da sich diese integrieren müssen, um am Ende ein Projekt zu entwickeln (Larman & Vodde, 2010).

Produktteams werden eingesetzt, wenn die Fachbereiche eine erhöhte Spezialisierung benötigen, oder sie eine hohe Komplexität aufweisen. Sie können auch zum Einsatz bei Projekten mit geringem Zeitdruck und geringem Risiko kommen. Zudem werden sie auch verwendet, wenn das Gesamtsystem aus vielen unterschiedlichen Technologien besteht. Auch die Größe der einzelnen Systemteile ist ausschlaggebend. So werden Produktteams dann eingesetzt, wenn die Teile zu klein sind um mehrere Schichten zu benötigen und wenn sie zu groß für ein einzelnes Team sind. Bezogen auf die Ressourcen werden Produktteams eingesetzt, wenn man gleichmäßig für alle Fachbereiche planen kann und die EntwicklerInnen selbst auch nur in ihren Fachbereichen arbeiten wollen beziehungsweise können (Larman & Vodde, 2010).

### 5.3. Projektteams

Teams, welche fachlich übergreifend am selben Projekt arbeiten, werden in dieser Arbeit als Projektteams gesehen. Diese können mit den Feature Teams verglichen werden. Durch die Definition von Fachbereich und Projekt in Abschnitt 5.2 steht somit ein Projektteam für jegliche Teile eines Systems, welche abgeschlossen einen Kundennutzen generieren. In Abbildung 5.2 wird diese Teamstruktur schematisch dargestellt. In den Projektteams soll somit von jedem Fachbereich zumindest eine/ein SpezialistIn vorhanden sein, um ein Projekt von Anfang bis zum Ende entwickeln zu können.



**Abbildung 5.2.:** Schematische Darstellung eines Projektteams (Quelle: eigene Darstellung)

Wie bei den Projektteams, identifizieren sich auch hier die Teammitglieder gut mit ihrer Arbeit. In diesem Fall ist es anstatt der einzelnen Fachbereiche jedoch der gesamte Teil des Systems, welcher implementiert wird um einen Kundennutzen zu generieren (Bachmann et al., 2019). In diesen Teams können die Mitglieder ein Domain Knowhow aufbauen, welches nicht nur auf ihren Fachbereich bezogen ist. Durch die Zusammensetzung aus Spezialistinnen/Spezialisten verschiedener Fachbereiche, können in diesen Teams die Mitglieder sehr viel voneinander lernen und in anderen Fachbereichen auch neues Wissen generieren. Diese Teams weisen eine hohe Effizienz bei der Entwicklung eines kompletten Projektes auf. Daher liegt auch der Fokus dieser Teams stark auf dem Projekt beziehungsweise auch folglich auf dem Kundennutzen. Des Weiteren liegt die hohe Effizienz unter anderem daran, dass das Team in keiner Abhängigkeit zu anderen Teams steht. Somit kommt es auch nicht zu diversen Wartezeiten bei etwaigen Übergaben oder Abstimmungen untereinander. Dies resultiert in einem geringen Aufwand für die Koordination und Kommunikation zwischen mehreren Teams (Emanuelsson et al., 2013). Abschließend ist auch die Ressourcenauslastung bei Projektteams sehr flexibel, da prinzipiell jedes Mitglied in diesem Team in allen Fachbereichen mitarbeiten können soll.

Der geringe Bedarf an Kommunikation zwischen den Teams ist nicht nur ein Vorteil, sondern kann auch dazu führen, dass Teile oder Funktionen mehrfach entwickelt werden. Durch die Aufteilung des Designs der einzelnen Fachbereiche auf mehrere Teams kann es auch zu Komplikationen kommen. Dies ist der Fall, wenn mehrere Teams gleichzeitig in einem Fachbereich arbeiten (Software Quality Lab GmbH, 2020).

Projektteams fordern Mitglieder, welche in jedem Fachbereich arbeiten können und auch den Willen dazu haben, dies zu tun. Solche Entwickler sind oft schwer zu finden. Es erfordert zudem auch noch, dass die Mitglieder sehr gute Entwicklungserfahrung haben, da die Effekte von Fehlern sich sonst auf viele weitere Projekte übertragen können, welche denselben Fachbereich verwenden.

Diese Art der Teamstruktur wird dann eingesetzt, wenn die Projekte stark fachbereichsübergreifend zu entwickeln sind und die einzelnen Fachbereiche eine geringe Komplexität aufweisen. Auch bei kritischen Projekten oder Projekten mit hohem Zeitdruck wird diese Form der Teamstrukturierung eingesetzt. Besteht ein Gesamtsystem aus wenigen verschiedenen Technologien, ist dies auch ein Indiz um diese Struktur zu verwenden. Wie bei den Produktteams ist auch hier die Größe der Systemteile entscheidend. Sie müssen groß genug sein, damit sie mehrere Schichten betreffen, aber klein genug um von einem Team entwickelt werden zu können. Hat man Entwickler, welche in allen Fachbereichen arbeiten können und dies auch wollen, so ist diese Struktur gut einsetzbar. Dadurch muss bei der Ressourcenplanung nicht darauf geachtet werden, dass alle Fachbereiche immer gleichmäßig ausgelastet sind (Larman & Vodde, 2010).

### 5.4. Zusammenfassung

Es gibt etliche Unterschiede und auch Gemeinsamkeiten, welche die klassischen und die agilen Methoden aufweisen. In dieser Arbeit werden bisher einige davon aufgezeigt. Der essentielle Teil dieser Arbeit ist jedoch die Gemeinsamkeit der beiden Modelle. So steht man zu Beginn in beiden Varianten mit der Frage da, ob man die Teams projektübergreifend im selben Fachbereich, oder besser doch fachbereichsübergreifend am selben Projekt arbeiten lässt. Hierfür werden in dieser Arbeit stellvertretend zwei Teamstrukturen vorgestellt, die Projekt- und die Produktteams. Jede dieser Strukturen hat, wie auch alle zuvor vorgestellten klassischen und agilen Strukturen, seine Vor- und Nachteile. Man kann immer abwägen was einem wichtiger ist. Dennoch gibt es keine konkreten Vorgaben wann welche Struktur gewählt werden soll. Die Tabelle 5.1 zeigt eine Zusammenfassung der Einsatzmöglichkeiten von Projekt- und Produktteams.

<b>Projektteams</b>	<b>Produktteams</b>
<i>Stark</i> fachbereichsübergreifende Projekte	<i>Wenig bis gar nicht</i> fachbereichsübergreifende Projekte
Fachbereiche erfordern <i>wenig bis keine</i> Spezialisierung	Fachbereiche erfordern <i>höchste</i> Spezialisierung
Projekte mit <i>hohem</i> Zeitdruck	Projekte mit <i>geringem</i> Zeitdruck
Projekte mit <i>hohem</i> Risiko	Projekte mit <i>geringem</i> Risiko
<i>Wenig</i> unterschiedliche Technologien in Verwendung	<i>Viele</i> unterschiedliche Technologien in Verwendung
Projekte umfassen <i>mehrere beziehungsweise alle</i> Schichten	Projekte umfassen nur <i>eine</i> Schicht
Projekt kann von <i>einem Team</i> bearbeitet werden	Projekt benötigt <i>mehrere Teams</i> zur Bearbeitung
Einführungszeit sollte <i>minimiert</i> werden	Einführungszeit ist <i>nicht kritisch</i>
Zukünftige Teile der Fachbereiche können <i>schlecht designed</i> werden	Zukünftige Teile der Fachbereiche können <i>gut und zuversichtlich designed</i> werden
Die einzelnen Bereiche weisen eine <i>geringe Komplexität</i> auf	Die einzelnen Bereiche weisen eine <i>hohe Komplexität</i> auf
EntwicklerInnen können und wollen in allen Fachbereichen arbeiten (Generalistinnen/Generalisten)	EntwicklerInnen können oder wollen <i>nicht</i> in allen Fachbereichen arbeiten (Spezialistinnen/Spezialisten)
<i>Nicht alle</i> Fachbereiche können gleichmäßig ausgelastet werden	<i>Alle</i> Fachbereiche können gleichmäßig ausgelastet werden

**Tabelle 5.1.:** Einsatzmöglichkeiten von Projekt- und Produktteams (Quelle: eigene Darstellung)

Mit Hilfe dieser Einsatzmöglichkeiten leiten sich die Kriterien ab. Diese werden anschließend dazu verwendet, um mögliche Vorgehensweisen der Strukturierung von Teams zu generieren. In Kapitel 6 werden die Vorgehensweisen detailliert und davon auch weitere Hypothesen abgeleitet, welche durch die folgenden Interviews geprüft werden.

## 6. Vorgehensmuster

Im Rahmen dieser Arbeit sollen konkrete Handlungsempfehlungen für ManagerInnen in der Software Branche erstellt werden, um die Entscheidung der Strukturierung von Teams in Produkt- oder Projektteams treffen zu können. Bevor die konkreten Handlungsempfehlungen abgeleitet werden, werden in diesem Kapitel Vorgehensmuster erstellt. Aus diesen Vorgehensmuster leiten sich Hypothesen ab, welche durch die folgenden Experteninterviews geprüft werden, um somit am Ende die konkreten Handlungsempfehlungen zu generieren.

In Kapitel 5 werden zuerst die Unterschiede und Gemeinsamkeiten von klassischen und agilen Methoden betrachtet. Anschließend werden in diesem Kapitel die in dieser Arbeit angestrebten Teamstrukturen vorgestellt und in Tabelle 5.1 deren Einsatzmöglichkeiten zusammengefasst. Auf Basis dieser Einsatzmöglichkeiten können die Kriterien für die Auswahl der Teams abgeleitet werden. Diese Kriterien werden in fünf Themengebiete gegliedert um daraus die Vorgehensweisen und dazugehörigen Hypothesen abzuleiten.

### 6.1. Art des Projektes

In Tabelle 2.1 nennt (Winkelhofer, 2005) Kriterien um die Entscheidung der Team-Organisation im klassischen Projektmanagement zu treffen. So nennt er den Zeitdruck und auch die Unsicherheit der Zielerreichung, welche mit dem Risiko eines Projektes gleichgestellt werden kann. Desto höher der Zeitdruck und größer das Risiko eines Projektes ist, umso weiter lenkt die Entscheidungsmatrix Richtung fachbereichsübergreifende Teamstrukturen, wie die Matrixorganisation und die reine Projektorganisation. Sind der Zeitdruck und das Risiko hingegen gering, so deutet dies auf eine Struktur eines einzelnen Fachbereichs hin, wie zum Beispiel die Stab-Linienorganisation.

Auch KnowledgeHut (2020b) beschreibt einen der Vorteile von Feature Teams als die Beschleunigung der Einführungszeit. Somit liegt auch in der agilen Welt bei zeitkritischen Projekten die Tendenz auf den fachbereichsübergreifenden Teams.

Zusätzlich zum Zeitmanagement und dem Risiko kommt es bei den Projekten auch allgemein auf den Umfang an, wie sehr dieser abteilungsübergreifend benötigt wird.



Zudem ist auch ein Kriterium, wie viele Schichten der Architektur durch das Projekt bearbeitet werden. So ist es in der klassischen als auch in der agilen Methode der Fall, dass umso mehr Schichten und Fachbereiche involviert sind, umso eher tendieren sie zu den fachbereichsübergreifenden Teamstrukturen (Larman & Vodde, 2010).

Die Tendenz hin zu fachbereichsübergreifenden Strukturen ergibt sich durch die diversen Vorteile dieser Strukturen. So beinhalten diese Teams alle notwendigen Kenntnisse um eine Aufgabe komplett innerhalb eines Teams zu lösen. Aus diesem Grund ergeben sich kurze Kommunikationswege innerhalb des Teams und wenig bis keine Abhängigkeiten zu anderen Teams. Dadurch sind die Teams äußerst effizient und haben keine Verzögerung durch etwaige Absprachen zwischen den Teams oder unnötiger Wartezeiten (Software Quality Lab GmbH, 2020).

## **6.2. Technische Komplexität des Unternehmens**

Abbot und Fisher (2010) schreiben, dass man die funktionale oder Linienorganisation einsetzt, wenn die Spezialisierung wichtiger ist, als das Problem der Koordination. In diesem Fall würde man sich somit gegen fachbereichsübergreifende Strukturen entscheiden und höchst spezialisierte Teams verwenden. Diese Teams werden allgemein verwendet, wenn Bereiche entweder eine hohe Spezialisierung erfordern oder wenn die Bereiche eine hohe Komplexität aufweisen.

Die Spezialisierung kann nicht nur in Fachbereichen benötigt werden, es kommt auch stark darauf an welche Technologien das Unternehmen verwendet. So deutet eine hohe Zahl an Technologien oder Technologien, welche wie die Fachbereiche höchste Spezialisierung benötigen darauf hin, dass man projektübergreifende Teams verwenden soll und keine fachbereichsübergreifenden Teams (Larman & Vodde, 2008).

Dies führt zwar zu Abhängigkeiten unter den Teams, jedoch fördert es unter anderem die Wiederverwendbarkeit der entwickelten Funktionen. Zudem bestehen diese Teams nur aus Expertinnen/Experten und dies resultiert zumeist in robusten und zuverlässigen Komponenten (Visual Paradigm, 2020).

## **6.3. Größe der Projekte beziehungsweise Teilaufgaben**

In vielen Unternehmen sind die Ressourcen zumeist knapp. Daraus ergibt sich meistens die Anforderung, diese optimal auszulasten. Dabei kann bereits eine passende Strukturierung der Teams Abhilfe schaffen (Strasser, 2020).

Ist ein gesamtes Projekt oder sind Teilaufgaben davon klein genug, damit es ein einziges Team abarbeiten kann, deutet dies auf fachbereichsübergreifende Teams hin. Dadurch kann das Projekt beziehungsweise die Teilaufgabe möglichst effizient innerhalb eines Teams ohne große Verluste durch Koordination und Kommunikation zwischen den Teams erledigt werden (KnowledgeHut, 2020b).

Auch, wenn durch die Teilaufgaben nicht alle Fachbereiche gleichmäßig ausgelastet sind, spricht das für Projektteams. Hier beinhaltet jedes Team von jedem benötigten Fachbereich zumindest eine/einen MitarbeiterIn. Alle Teammitglieder sollen in den verschiedenen Bereichen arbeiten, so wird gewährleistet, dass in jenen Fachbereichen wo vermeintliche Stehzeiten auftreten, das gesamte Team beziehungsweise alle Teams gleichermaßen ausgelastet sind (Scaled Agile, Inc., 2020b).

Dadurch ergeben sich wie bereits erwähnt eine gleichmäßige Auslastung aller Teammitglieder, eine Vermeidung von Stehzeiten und eine effiziente Abarbeitung der einzelnen Teilaufgaben beziehungsweise der Projekte.

## **6.4. Zukünftiges Design der Schnittstellen**

Auch ein Aspekt, welcher sich auf die Strukturierung der Teams auswirkt, ist das Design zukünftiger Schnittstellen zwischen den einzelnen Fachbereichen oder Komponenten. So spricht es dafür, ein fachübergreifendes Team einzusetzen, wenn die zukünftigen Schnittstellen schlecht oder gar nicht designet werden können (Putman, 2020). In diesem Fall ist das Design der Schnittstellen auf alle Teams aufgeteilt und jedes Team kann das Design dann vornehmen, wenn es gerade in diesem Bereich etwas bearbeitet. Dadurch müssen zukünftige Schnittstellen nicht gut geplant sein. Zudem kommt es auch nicht zu erhöhtem Koordinationsaufwand zwischen den Teams und eventuellen Stillständen, wenn eine Schnittstelle geändert wird, aber das zuständige Team gerade keine Zeit dafür hat (Larman & Vodde, 2010).

## **6.5. Kompetenz der EntwicklerInnen**

Der letzte Aspekt der sich aus den Einsatzmöglichkeiten aus Tabelle 5.1 ergibt, ist die Kompetenz der EntwicklerInnen. Damit gemeint ist, ob die EntwicklerInnen es können und auch dazu gewillt sind, in mehreren Fachbereichen zu arbeiten. Sind die EntwicklerInnen reine Spezialistinnen/Spezialisten in einem Fachbereich, besteht zwar die Möglichkeit die EntwicklerInnen dahin zu motivieren sich langsam auch in andere Themenbereiche einzuarbeiten, jedoch kann dies die Motivation der MitarbeiterInnen senken, falls diese in ihrem Gebiet und ihren aktuellen Tätigkeiten zufrieden

sind. Wenn die EntwicklerInnen bereits in mehreren Fachbereichen arbeiten können, dies aber nicht wollen, ergibt sich ein ähnliches Szenario. Daher sollen die Teams, wenn es EntwicklerInnen gibt, welche nicht in mehreren Fachbereichen arbeiten können oder wollen, in Produktteams strukturiert sein.

Eine Strukturierung in Produktteams weist wieder den Vorteil der Spezialisierung der einzelnen Teams auf, im Verlust des Vorteils, dass die Teams unabhängig voneinander sind.

## 6.6. Zusammenfassung & Hypothesenbildung

Die Zusammenfassung dient dazu, die einzelnen Vorgehensweisen aufgrund der zuvor vorgestellten Themenbereiche zu definieren und dazugehörige Hypothesen abzuleiten. Daher werden folgend alle Vorgehensmuster übersichtlich dargestellt und deren Hypothesen dazu beschrieben. Die Arbeitshypothese wird bereits in Kapitel 5 geprüft. Diese Hypothese dient der Anregung für die Forschung und für die Richtungsvorgabe dieser Arbeit. Die nachstehend angeführten Hypothesen dienen der Ausarbeitung für die konkreten Handlungsempfehlungen. Sie beschreiben die Basis der Vorgehensmuster. Um aus den Vorgehensmustern die Empfehlungen abzuleiten, werden die Vorgehensmuster validiert. Dies geschieht durch eine Prüfung der definierten Hypothesen.

Das erste Vorgehensmuster (VM1) „Art des Projektes“ bezieht sich auf diverse Eigenschaften eines Projektes. So sind die zeitlichen Faktoren, das Risiko und die fachübergreifende Notwendigkeit eines Projektes für die Teamstrukturierung ausschlaggebend. Daher ergibt sich **VM1**: *„Ist ein Projekt stark fachübergreifend, benötigt mehrere Schichten, weist ein hohes Risiko oder einen hohen Zeitdruck auf oder Ziel des Projektes soll sein, die Einführungszeit zu minimieren, so soll das Team für das Projekt als Projektteam organisiert sein“*. Die dazugehörige Hypothese (**HT1**) wird wie folgt definiert: *„Die Eigenschaften eines Projektes bezüglich fachbereichsübergreifende Tätigkeiten, Zeit, Risiko und Einführungszeit sind ausschlaggebend für die Strukturierung von Scrum Teams.“*

Das zweite Vorgehensmuster (VM2) „Technische Komplexität des Unternehmens“ richtet das Augenmerk auf die Komplexität und die benötigte Spezialisierung des Unternehmens. Für dieses Vorgehensmuster sind die benötigte Spezialisierung der Fachbereiche, die Anzahl der verwendeten Technologien und die Komplexität der einzelnen Bereiche ausschlaggebende Kriterien für die Strukturierung der Teams. Somit ergibt sich **VM2**: *„Verwendet man viele verschiedene Technologien, weisen diese oder einzelne Bereiche eine hohe Komplexität auf oder benötigen höchste Spezialisierung, so sollen die Teams als Produktteams strukturiert sein“*. Aus diesem Vorgehensmuster lässt sich die Hypothese (**HT2**) ableiten: *„Die Anzahl an verwendeten Technologien, die Komplexität*

*des Systems und die benötigte Spezialisierung sind ausschlaggebend für die Strukturierung von Scrum Teams.“*

Im dritten Vorgehensmuster (VM3) „Größe der Projekte beziehungsweise Teilaufgaben“ betrachtet man die Ressourcenauslastung. Hierfür sind in diesem Fall die Größe der Projekte beziehungsweise deren Teilaufgaben und die gleichmäßige Auslastung der einzelnen Fachbereiche entscheidend. **VM3** wird definiert als: *„Kann man die Fachbereiche nicht gleichmäßig Auslasten und hat ein Projekt beziehungsweise dessen Teilaufgaben die Größe, dass es von einem einzelnen Team entwickelt werden kann, dann sollen die Teams als Projektteams strukturiert sein“*. Hierfür stellt man die Hypothese (**HT3**): *„Die Planung der gleichmäßigen Ressourcenauslastung, sowie die Größe der Projekte beziehungsweise Teilaufgaben sind ausschlaggebend für die Strukturierung von Scrum Teams“*.

„Zukünftiges Design der Schnittstellen“ stellt das vierte Vorgehensmuster (VM4) dar. Hierfür ist entscheidend, wie gut man die zukünftigen Schnittstellen der einzelnen Komponenten, Schichten oder ähnlichem untereinander planen und designen kann. Daraus ergibt sich das Vorgehensmuster **VM4** wie folgt: *„Können zukünftige Schnittstellen nicht gut vorab designet werden, sollen die Teams als Projektteams strukturiert werden“*. Daraus ergibt sich die Hypothese (**HT4**): *„Das Design von zukünftigen Schnittstellen ist ausschlaggebend für die Strukturierung von Scrum Teams“*.

Das letzte Vorgehensmuster (VM5) „Kompetenz der EntwicklerInnen“ bezieht sich auf die fachbereichsübergreifenden Kompetenzen und den Willen der EntwicklerInnen. Hierbei ist wichtig, ob die EntwicklerInnen das Können und auch den Willen haben, fachbereichsübergreifend zu arbeiten. Das Vorgehensmuster **VM5** lässt sich folglich so definieren: *„Haben EntwicklerInnen weder das Können noch den Willen dazu, fachbereichsübergreifend zu arbeiten, sollen die Teams als Produktteams strukturiert werden“*. Die letzte Hypothese (**HT5**) lautet: *„Das Können und der Wille der EntwicklerInnen, fachbereichsübergreifend zu arbeiten, ist ausschlaggebend für die Strukturierung von Scrum Teams“*.

Diese fünf aufgestellten Vorgehensmuster ergeben nur einen Entwurf der am Ende resultierenden Handlungsempfehlungen. Um diese Vorgehensmuster in konkrete Handlungsempfehlungen überzuführen, werden nachfolgend die dazugehörigen Hypothesen durch die Experteninterviews geprüft und die Vorgehensmuster dementsprechend angepasst, dass die konkreten Handlungsempfehlungen daraus erstellt werden können.

# 7. Evaluierung

Auf Basis der theoretischen Aufbereitung des Themas sind in Kapitel 6 bereits erste Vorgehensmuster definiert. Diese Vorgehensmuster werden mit Hilfe von Interviews validiert. Mit den Resultaten der Interviews und den Vorgehensmuster als Vorlage werden die konkreten Handlungsempfehlungen verfasst.

Die folgenden Abschnitte beschreiben die Vorgehensweise, die Vorbereitungen und die Ergebnisse der Interviews.

## 7.1. Vorgehensweise

Die Interviews finden auf Basis von Leitfadeninterviews statt. Diese Leitfadeninterviews stellen eine qualitative Methode dar. Hierbei ist bereits ein Leitfaden vorbereitet, welcher die Fragen in ausformulierter Form enthält. Es kann jedoch die Reihenfolge, aber auch der Inhalt während der Interviews individuell angepasst werden. Daher können auch weitere Inhalte hinzukommen (Oehrich, 2019).

Ein Vorteil von Leitfadeninterviews ist die Vergleichbarkeit und die Wiederholbarkeit der Antworten (Oehrich, 2019). Alle sechs InterviewpartnerInnen werden separat befragt, damit sie sich nicht gegenseitig beeinflussen. Dadurch wird dieser Vorteil von Leitfadeninterviews im Rahmen dieser Arbeit genutzt.

Ein weiterer Vorteil von Leitfadeninterviews besteht darin, dass die/der Interviewe-rIn die Möglichkeit hat, nicht nur die definierten Fragen zu stellen. So hat sie/er auch die Gelegenheit, durch weitere Rückfragen detaillierte Informationen zum befragten Thema zu generieren (Pohl & Rupp, 2015). Das in dieser Arbeit behandelte Thema kann sich in der Praxis stark von der Theorie unterscheiden. Daher sieht der Autor die Tatsache, Rückfragen stellen zu können, als einen der wichtigsten Vorteile um detailliertere Informationen von den befragten Personen zu erlangen.

Für diese Arbeit wird der Unterschied von verschiedenen Branchen nicht mitbetrachtet. Daher wird bei der Auswahl der InterviewpartnerInnen lediglich darauf geachtet, dass zumindest pro vorgeschlagener Teamstruktur ein Unternehmen vertreten

ist. Dieses Unternehmen soll bereits Scrum im Einsatz haben und in der Softwareentwicklung tätig sein. Die InterviewpartnerInnen setzen sich aus drei verschiedenen Positionen in den jeweiligen Unternehmen zusammen, um somit sowohl die Sicht der Entwicklerin/des Entwicklers, als auch der Managerin/des Managers und der Projektmanagerin/des Projektmanagers in Betracht zu ziehen. Zudem sollen die InterviewpartnerInnen selbst auch zumindest praktische Erfahrung mit Scrum haben. Jegliche Ausbildungen hinsichtlich Scrum sind nicht zwingend notwendig.

Der Fragebogen für das Leitfadeninterview setzt sich aus demografischen und praxisorientierten Fragen bezüglich der Strukturierung von Teams im agilen Umfeld zusammen. Dadurch gewinnt man zusätzlich zur bisherigen theoretischen Bearbeitung der Forschungsfrage auch die praxisorientierte Sicht, um ein gemeinsames Bild aus Theorie und Praxis erstellen zu können.

Die Interviewfragen werden den InterviewpartnerInnen bereits vorab übermittelt, damit sie sich vorbereiten können. Das Interview wird elektronisch aufgezeichnet und anschließend transkribiert. Die Transkription stellt somit die Voraussetzung für die inhaltliche Analyse und in weiterer Folge für die Erstellung der konkreten Handlungsempfehlungen dar.

Für die Auswertung wird die qualitative Inhaltsanalyse nach Philipp Mayring angewendet. Diese Methode ist die vorwiegend verwendete im deutschsprachigen Raum (Dunger, Kolbe, Schnell & Schulz-Quach, 2013). Ziel dieser Methode ist es, die protokollierten Daten zu reduzieren und zugleich ein überschaubares Ergebnis zu generieren und dabei alle wesentlichen Inhalte beizubehalten. Diese Methode gliedert sich in drei Schritte. Die Paraphrasierung, Generalisierung und Reduktion. Der erste Schritt, die Paraphrasierung, dient dazu, Textstellen, welche keinen oder wenig Inhalt bieten zu streichen. Als nächster Schritt werden in der Generalisierung alle Paraphrasen auf eine gemeinsame und einheitliche Abstraktionsebene verallgemeinert. Im letzten Schritt werden durch die Reduktion alle Paraphrasen gestrichen, welche bedeutungsgleich sind (Mayring, 2015).

Laut Vogt und Werner (2014) sind deduktive Kategorien relevant für die Erstellung der Fragen für ein Leitfadeninterview. Werden Kategorien vor der Befragung gebildet, dann spricht man von deduktiven Kategorien. Dies ist auch notwendig für die Inhaltsanalyse, welche ein Kategoriensystem nutzt, um ein Abstraktionslevel zu schaffen. Durch dieses Abstraktionslevel können die verschiedenen Textpassagen zusammengefasst werden (Mayring, 2015). Daher werden für die Befragung im Rahmen dieser Arbeit die Kategorien vorab gebildet.

## 7.2. Vorbereitungen

Für das Leitfadeninterview werden konkrete Fragestellungen auf Basis der in Kapitel 6 vorgestellten Vorgehensmuster und den dazugehörigen Hypothesen gebildet. Um den InterviewpartnerInnen einen Kontext zu bieten, werden zu Beginn die Forschungsfrage und die Arbeit kurz vorgestellt. Zudem werden auch jegliche Begriffe geklärt, welche unterschiedliche Interpretationen haben können. Beispiele für solch einen Begriff wären „Produktteam“ und „Projektteam“.

Die Fragestellungen für das Leitfadeninterview werden anhand deduktiv generierter Kategorien erstellt. Zu Beginn des Interviews gibt es ein paar Punkte, welche den Namen, das Alter, das Unternehmen, die Position im Unternehmen und die Größe des Unternehmens betreffen. Danach werden in Kategorie 1 noch weitere demographische Fragen gestellt. Die Kategorie 2 beinhaltet Fragen zur aktuellen Struktur der Teams im Unternehmen der befragten Person und aufgrund welcher Kriterien die Entscheidungen getroffen worden sind. In den Kategorien 3 bis 7 werden die fünf zuvor gebildeten Kategorien befragt, um die Vorgehensweisen und Hypothesen zu bestätigen. Die letzte Kategorie beinhaltet eine Abschluss Frage, durch welche die/der InterviewpartnerIn noch die Möglichkeit bekommt etwas anzumerken was ihr/ihm wichtig erscheint, aber nicht im Interview enthalten ist. Der komplette Fragebogen ist im Anhang A.1 dieser Arbeit abgebildet.

Die allgemeinen Fragen zur Person, die Kategorie 1 und 2 stellen zwingend abzufragende Kategorien dar. Bei den weiteren Kategorien kann die/der InterviewerIn entscheiden, ob durch die Beantwortung der jeweiligen Fragen relevante Informationen gewonnen werden können. Ist dies nicht der Fall, kann die/der InterviewerIn diese Fragen auch auslassen.

## 7.3. Ergebnisse

Für die Arbeit werden zwei Unternehmen mit jeweils drei Personen ausgewählt. Diese beiden Unternehmen sind jeweils einmal als Projektteam und einmal als Produktteam organisiert. Die InterviewpartnerInnen setzen sich aus einer/einem ProjektleiterIn, einer/einem EntwicklerIn und einer/einem IT-LeiterIn des jeweiligen Unternehmens zusammen. In diesem Abschnitt werden zuerst die InterviewpartnerInnen vorgestellt. Anschließend werden die Aussagen der einzelnen Interviews zusammengefasst, welche die Ergebnisse der Inhaltsanalyse darstellen.

### 7.3.1. InterviewpartnerIn

Die Tabelle 7.1 zeigt einen Überblick über die sechs InterviewpartnerInnen, die dazugehörigen Unternehmen sowie die Strukturierung der Teams in den Unternehmen.

InterviewpartnerIn	Unternehmen	Team Struktur
Interviewpartner 1	Unternehmen A	Produktteams
Interviewpartner 2	Unternehmen B	Projektteams
Interviewpartner 3	Unternehmen B	Projektteams
Interviewpartner 4	Unternehmen A	Produktteams
Interviewpartner 5	Unternehmen A	Produktteams
Interviewpartner 6	Unternehmen B	Projektteams

**Tabelle 7.1.:** Übersicht der InterviewpartnerInnen zu Unternehmen und Team Struktur (Quelle: eigene Darstellung)

Interviewpartner 1 ist eine männliche Person, welche im IT-Umfeld der Messtechnikbranche tätig ist. Der Interviewpartner ist in diesem Unternehmen seit zwei Jahren als Bereichsleiter tätig. Er hat bereits mehr als 15 Jahre Erfahrung mit agiler Entwicklung, welche er in einer kleinen Technologie Firma gesammelt hat. Mit dem Begriff Scrum ist die Person bereits seit dem Aufkommen des Begriffes konfrontiert. Auch das aktuelle Unternehmen arbeitet mit Scrum. Hinsichtlich Scrum hat die Person jedoch keine zusätzlichen Ausbildungen oder Zertifizierungen.

Der Interviewpartner 2 ist eine männliche Person, welche im IT-Umfeld in der Mode- und Einzelhandelsbranche tätig ist. Er arbeitet in seinem Unternehmen seit fünf Jahren als CTO. Er hat eine Ausbildung und Zertifizierung als „Certified Product Owner“ und arbeitet auch im Unternehmen seit vier Jahren mit Scrum.

Auch im IT-Umfeld in der Mode- und Einzelhandelsbranche ist Interviewpartner 3 tätig. Diese Person ist männlich und arbeitet als Softwareentwickler seit fünf Jahren in dem Unternehmen. Von den fünf Jahren arbeitet er seit drei Jahren aktiv mit Scrum, hat jedoch keine zusätzlichen Ausbildungen oder Zertifizierungen hinsichtlich Scrum.

Interviewpartner 4 arbeitet seit 20 Jahren als Softwareentwickler und ist männlich. Seit drei Jahren arbeitet er in seinem aktuellen Unternehmen im IT-Umfeld in der Messtechnikbranche. Er hat bereits zehn Jahre praktische Erfahrung mit Scrum, jedoch keine zusätzlichen Ausbildungen oder Zertifizierungen hinsichtlich Scrum.

Interviewpartner 5 ist ebenso im IT-Umfeld in der Messtechnikbranche tätig. Er arbeitet seit acht Monaten als Projektleiter. Diese Person ist männlich und hat bereits zwei bis drei Jahre praktische Erfahrung mit Scrum. Hinsichtlich Scrum hat er keine weiteren Ausbildungen oder Zertifikate vorzuweisen.

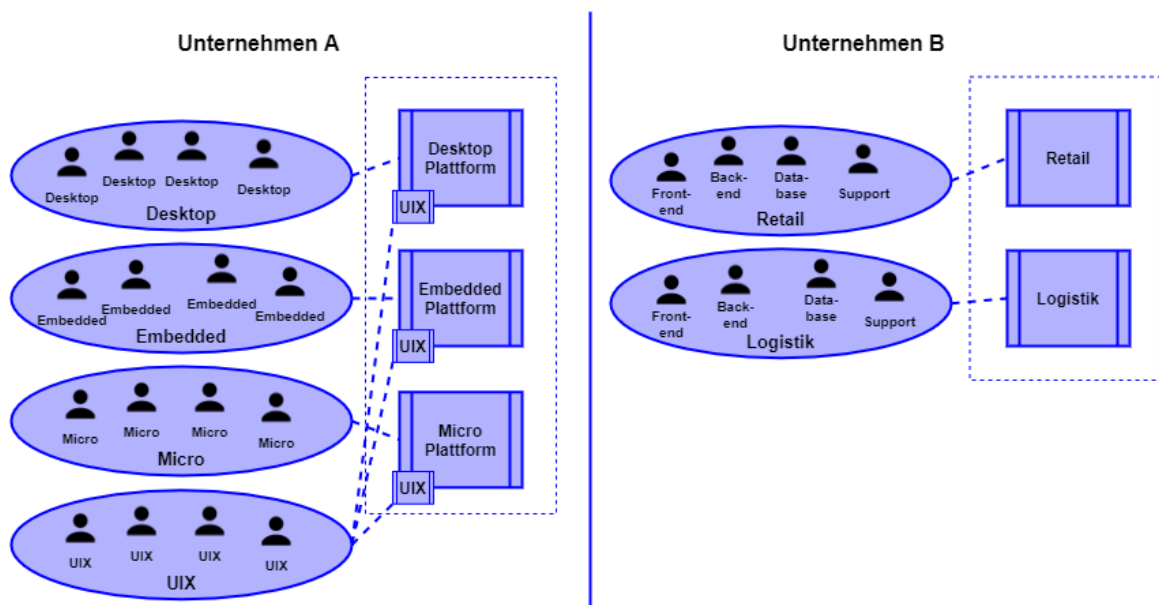


Der Interviewpartner 6 ist männlich und im IT-Umfeld der Mode- und Einzelhandelsbranche tätig. Er ist seit drei Jahren Projektmanager im Unternehmen und hat einen theoretischen Hintergrund von Scrum. Zudem hat er drei Jahre praktische Erfahrung mit Scrum. Hinsichtlich Scrum hat er keine weiteren Ausbildungen oder Zertifikate.

In Abschnitt 7.2 wird erwähnt, dass Kategorien der Fragen ausgelassen werden können. Bei allen sechs Interviews werden alle InterviewpartnerInnen bezüglich aller Kategorien befragt. Es gibt somit keine Kategorien, welche nicht beantwortet wurden.

### 7.3.2. Organisation von Teams

Die Fragen dieser Kategorie dienen dazu, um die aktuelle Organisation der Teams der InterviewpartnerInnen darstellen zu können. Dies ist auch in weiterer Folge wichtig, um die Aussagen der Personen der jeweiligen Team Struktur zuordnen zu können. Es wird versucht, die Anzahl der Teams, die Anzahl der Teammitglieder und die aktuelle Strukturierung zu befragen. Dadurch können die Aussagen der jeweiligen Personen immer auf die richtige Teamstruktur bezogen werden und somit die Vorgehensmuster evaluiert werden. Aufgrund der Interviews können in Abbildung 7.1 die aktuellen Strukturen der beiden Unternehmen dargestellt werden.



**Abbildung 7.1.:** Schematische Darstellung der Strukturierung der Teams im Unternehmen A und B der InterviewpartnerInnen (Quelle: eigene Darstellung)

Im Unternehmen A bestehen die Teams aus drei bis acht Personen und sind nach Fachbereich strukturiert. Die Plattformen, welche sie entwickeln, dienen als Basis für weitere Teams in der Produktentwicklung, welche in Abbildung 7.1 nicht abgebildet

sind. Somit entwickeln diese Teams Plattform-Komponenten, welche in die Produkte integriert werden und sind daher laut der Definition dieser Arbeit Produktteams.

Zusätzlich gibt es noch ein Team welches sich aus Produktmanagement, Projektleitung und Support zusammensetzt und ein Team welches sich mit neuen Technologien auseinandersetzt. Das erstere dieser Teams ist unter anderem verantwortlich für die Kommunikation zwischen den Teams. Das Technologie Team ist für die Integration der einzelnen Plattformen zuständig. Dies setzt es in einem eigenen „Cloud Use-Case“ um, dass es alle Plattformen in Form von Demonstratoren in ein großes Gesamtbild integriert. Außerdem, gibt es in jedem Team eine technische Leiterin/einen technischen Leiter. Durch die Einführung dieser zusätzlichen Teams und die technischen LeiterInnen hat man die Kommunikation unter den Teams simplifiziert und auch versucht, die Integration zu vereinfachen.

Im Vergleich zu Unternehmen A entwickelt Unternehmen B pro Team 1 Produkt, welches direkt an die Kundin/den Kunden verkauft wird. Hier gilt zu beachten, dass ein Produkt technisch aus einem bis mehreren verschiedenen Services zusammengesetzt ist. Die Produkte, Retail und Logistik, können sowohl einzeln als auch gemeinsam verkauft werden. Werden die Produkte einzeln verkauft, benötigen sie keine Integration zu anderen Produkten. Daher sind diese Teams per Definition als Projektteams zu sehen. Die einzelnen Teams bestehen im Unternehmen B aus drei bis zehn Mitarbeiterinnen/Mitarbeitern.

Eine Spezialität von Unternehmen B ist, dass jedes Team für den kompletten Lebenszyklus des Produktes zuständig ist. So sind diese von der Entwicklung bis hin zu Test, Produktivbetrieb und sogar Support zuständig. Daher gibt es in jedem der Teams auch eine Vertretung des Kundensupports. Zusätzlich gibt es im Unternehmen B noch ein drittes Team. Dieses Team befasst sich mit Datenanalyse und wird bei den Interviews und auch in der weiteren Arbeit nicht mit betrachtet.

Zur Strukturierung werden zusätzlich auch die Kriterien, welche zur Auswahl der aktuellen Struktur betrachtet werden, abgefragt. Auch die Vorteile welche man sich durch die Strukturierung erhofft hat, sind Teil dieser Kategorie. Dadurch können erste Bestätigungen oder Unstimmigkeiten der vorgeschlagenen Vorgehensmuster identifiziert werden.

Im Unternehmen A besteht Konsens darüber, dass die Struktur historisch gewachsen ist. Man hat bereits damals die Trennung der Fachbereiche von Micro und Desktop und hat mit der Erweiterung des Bereiches diese Struktur nicht angepasst sondern um das Embedded und User Interaction and Experience (UIX) Team erweitert. Ein Vorteil den Interviewpartner 1 für die Strukturierung nennt ist, dass sich durch das Heraustrennen des Technologie Teams dieses Team mit neuen Technologien auseinandersetzen kann ohne gleichzeitig an kundenrelevanten Themen arbeiten zu müssen.

Im Unternehmen B wird vermehrt erwähnt, dass der große Vorteil durch diese Strukturierung jener ist, dass die Integration unter den Teams sehr gering ist. Dies entspricht auch dem Vorteil wie er in dieser Arbeit bei den Projektteams in Abschnitt 5.3 erwähnt wird, dass die Projektteams keine Abhängigkeiten zu anderen Teams haben. Dies erlaubt es in weiterer Folge auch, dass die einzelnen Teams räumlich getrennt arbeiten können.

Abschließend beinhaltet diese Kategorie die Frage, ob die/der InterviewpartnerIn die Struktur aufgrund des heutigen Wissensstandes anders strukturieren würde und warum sie/er das machen würde. Es halte fast alle InterviewpartnerInnen fest, dass sie mit der aktuellen Struktur zufrieden sind und diese gut in ihrem Unternehmen funktioniert. Interviewpartner 2 erwähnt zwar, dass die Integration einer Support Mitarbeiterin/eines Support Mitarbeiters in ein Team nicht zufriedenstellend funktioniert, er dennoch die Struktur nicht ändern würde. Lediglich Interviewpartner 5 erwähnt den Plan in Zukunft die Teams anders zu strukturieren. Er erzählt, dass es mit der aktuellen Struktur Probleme gibt, welche man hofft mit einer neuen Struktur, welche fachbereichsübergreifend gegliedert ist zu beheben. Vor allem erhofft er sich dadurch, flexibler auf Kundenanforderungen reagieren zu können.

### 7.3.3. Art des Projektes

Diese Kategorie stellt konkrete Fragen für das erste Vorgehensmuster (VM1). Im VM1 geht es darum, dass sich die Eigenschaften der Projekte auf die Auswahl der Team Struktur auswirken. Hierzu ermitteln diese drei Fragen die Themen Risiko und Zeitdruck, die Minimierung der Einführungszeit und wie fachbereichsübergreifend die Projekte sind.

Die erste Frage beschäftigt sich mit dem Risiko und dem Zeitdruck. Wie in Abschnitt 6.1 beschrieben, deutet ein hohes Risiko und ein hoher Zeitdruck darauf hin, dass die Teams als Projektteams strukturiert werden sollen. Bei den Interviews erwähnen jedoch alle InterviewpartnerInnen, dass ihre Projekte hohem Zeitdruck und Risiko unterliegen. Somit sind in beiden Teamstrukturen die Projekte hinsichtlich dieser Kriterien nicht unterschiedlich.

Die zweite Frage gilt der Minimierung der Einführungszeit. Laut KnowledgeHut (2020b) ist es ein definitiver Vorteil von fachbereichsübergreifenden Teams, dass sie sehr effizient arbeiten. Dies ergibt sich vor allem aufgrund der geringen Integration und Koordination unter den Teams. Daher ist eines der Kriterien für die Wahl von Projektteams, das Ziel die Einführungszeit zu minimieren. Im Unternehmen B ist es ein konkretes Ziel, durch die Strukturierung als Projektteams die Einführungszeit zu minimieren und somit auch kürzere Feedbackschleifen mit den Kundinnen/Kunden zu generieren. Dies führt dazu, dass in jedem Team Spezialistinnen/Spezialisten jedes Fachbereiches vorhanden sind und zu geringem Abstimmungsaufwand unter

den Teams und resultiert dadurch in schnelleren Entwicklungszeiten. Im Unternehmen A hingegen, ist es kein Ziel die Einführungszeit der Komponenten zu minimieren. Hier ist es lediglich ein Ziel, die Einführungszeit der resultierenden Produkte zu minimieren, was durch die erhöhte Wiederverwendbarkeit der Plattform Komponenten geschaffen wird.

Die letzte Frage dieser Kategorie beschäftigt sich damit, wie fachbereichsübergreifend die Projekte sind. Dahingehend wird gefragt, ob ein Kundennutzen rein durch einen Fachbereich generiert werden kann. In Bezug auf das Unternehmen A ist das definitiv der Fall, da diese als Produktteams strukturiert sind und somit je nach Fachbereich eine Komponente für die unternehmensinternen Kundinnen/Kunden zur Verfügung stellen kann. Im Unternehmen B ist das nicht der Fall. Hier können zwar die einzelnen Teams komplette Kundennutzen generieren, jedoch sind die Teams fachbereichsübergreifend strukturiert, da das Produkt aus allen Fachbereichen zusammengesetzt ist. Auf die Frage, ob ein Kundennutzen durch ein Team generiert werden kann, antworten jedoch alle sechs Interviewpartner mit ja.

### **7.3.4. Technische Komplexität des Unternehmens**

Diese Kategorie befasst sich mit Fragen bezüglich der Komplexität des Unternehmens. Dazu zählen die Anzahl der Technologien, welche in Verwendung sind, die Spezialisierung der Fachbereiche und die Komplexität der einzelnen Fachbereiche.

Im Unternehmen A sind die Teams sehr divers. Hier reicht es von recht trivialen Teams bis hin zu sehr komplexen und auch spezialisierten Teams. Es sind auch viele verschiedene Technologien in Verwendung, wobei diese auf die einzelnen Teams verteilt sind. Somit deutet beides, die Spezialisierung beziehungsweise die Komplexität der Teams und auch die Anzahl an Technologien, eindeutig auf Produktteams hin. So sind sie aktuell auch strukturiert. Zudem ist es ein klares Ziel, die Wiederverwendbarkeit im Unternehmen zu erhöhen. Dies resultiert auch in der aktuellen Struktur um die Plattform Komponenten für die Produkte zu entwickeln. Visual Paradigm (2020) erwähnt, dass es durch diese Trennung zu Abhängigkeiten zwischen den Teams kommt. Im Fall des Unternehmens A trifft das nicht zwingend auf den eigenen Bereich zu, aber zumindest auf die weiteren Teams, welche die finalen Produkte entwickeln. Jedoch ist ein Vorteil solcher Strukturen, die Wiederverwendbarkeit zu erhöhen und gleichzeitig robuste und zuverlässige Komponenten zu implementieren, was ein Ziel von Unternehmen A ist.

Über Unternehmen B ist es schwer eine konkrete Aussage zu treffen, da sowohl bei der Anzahl der verwendeten Technologien, als auch bei der Komplexität die Meinungen sehr divers sind. Interviewpartner 2 schätzt sowohl die Komplexität des Unternehmens als auch die Anzahl an verwendeten Technologien als gering ein. Zudem verweist er auch auf die Ähnlichkeit der Domänen in der die beiden Teams arbeiten.

Beide arbeiten im Bereich der Mode- und Einzelhandelsbranche jedoch getrennt in den Retail und in den Logistik Bereich. Hingegen sind sich die Interviewpartner 3 und 6 einer Meinung, dass die einzelnen Produkte und somit auch Teams sehr komplex sind. Wobei sich auch hier die Meinungen bezüglich der Anzahl an Technologien von wenig bis hin zu vielen Verschiedenen bewegt. Was man dennoch sagen kann ist, dass die einzelnen Fachbereiche nicht sehr spezialisiert und komplex sind, sondern die Produkte, welche eine Zusammenstellung aller Fachbereiche ergeben.

### **7.3.5. Größe der Projekte beziehungsweise Teilaufgaben**

Ein Aspekt bei Projekten sind immer die Ressourcen. Diese sind zumeist in den Unternehmen knapp und sollen daher möglichst effizient eingesetzt werden. Um dies zu bewerkstelligen, sollen alle MitarbeiterInnen gut ausgelastet sein und den maximalen Kundennutzen dadurch generieren und nicht mit Tätigkeiten beschäftigt werden, welche nur dazu dienen, um die Ressourcen auszulasten. Daher ist das Ziel der Fragen die gleichmäßige Auslastung der einzelnen Fachbereiche und auch die Größe der Projekte oder Teilaufgaben abzufragen.

Wie bereits in Abschnitt 7.3.3 erwähnt, kann in beiden Unternehmen ein Kundennutzen rein durch ein Team generiert werden. Dadurch wird angenommen, dass die einzelnen Projekte und auch Arbeitspakete klein genug sind, um von einem Team bearbeitet zu werden. Dies resultiert vermutlich aus der Tatsache, dass die Projekte angepasst an die Teamstruktur erstellt werden und nicht die Projekte zuerst vorhanden sind und man sich dafür passende Teams zusammenstellen muss.

Im Unternehmen A besteht auch bei der Auslastung Konsens darüber, dass der Bereich allgemein sehr gut ausgelastet ist und es somit keine Probleme gibt, die einzelnen Fachbereiche und somit auch Teams auszulasten. Es wird erwähnt, dass es in diesem Unternehmen immer in einer Priorisierung endet und am Ende nicht alles, was eigentlich gefordert ist, gemacht werden kann. Somit sind alle Teams immer gut ausgelastet.

Bei den Teams im Unternehmen B gibt es auch keine Probleme, die einzelnen Fachbereiche und somit Teammitglieder auszulasten. Was jedoch ein Problem darstellt, ist die einzelnen Teams als Einheit auszulasten. Durch die Ähnlichkeit der Technologien, der Domäne und auch durch die Strukturierung als Projektteams können jedoch Tätigkeiten von Team A auch an Team B übergeben werden. Dies ist jedoch mit dem negativen Aspekt behaftet, dass diese Tätigkeiten dann nicht so effizient waren, wie im eigentlichen Team.

### **7.3.6. Zukünftiges Design der Schnittstellen**

Im Zuge dieser Frage wird geklärt, wie gut können Designs von Schnittstellen vorab erstellt werden und wie gut können zukünftige Kundenanforderungen geplant werden. Diese Frage wird in beiden Unternehmen sehr divers beantwortet.

Interviewpartner 1 argumentiert, dass das Design zukünftiger Schnittstellen leicht zu bewerkstelligen sei, wenn allerdings die Plattformen bereits vor den Produktentwicklungen fertig sind. Da dies aktuell jedoch nicht der Fall ist, kann diese Aussage schlecht verwertet werden. Interviewpartner 4 erwähnt zwei Arten von Schnittstellen. Einerseits die Schnittstellen zwischen den Teams und andererseits jene zu den Produktbereichen nach außen. Erstere können gut designet werden und zweitere stellen eine Herausforderung dar. Auch Interviewpartner 5 sagt, dass die Schnittstellen vorab schlecht geplant werden können.

Auch im Unternehmen B gibt es zwei unterschiedliche Meinungen. So tendieren Interviewpartner 2 und 6 dazu, dass zukünftige Schnittstellen schlecht vorab geplant werden können. Interviewpartner 3 hingegen appelliert auf die Erfahrungswerte und meint somit, dass die zukünftigen Schnittstellen gut planbar sind.

### **7.3.7. Kompetenz der EntwicklerInnen**

In dieser Kategorie geht es darum, ob die EntwicklerInnen des Unternehmens das Können und auch den Willen besitzen, um fachbereichsübergreifend zu arbeiten. Dahingehend wird gefragt, ob die EntwicklerInnen aktuell in mehreren Fachbereichen tätig sind. Diese Frage wird in beiden Unternehmen sehr eindeutig beantwortet.

Im Unternehmen A arbeitet jede/jeder EntwicklerIn nur in ihrem/seinem Fachbereich und somit auch in ihrem/seinem Team. Es gibt auch sehr viele Spezialistinnen/Spezialisten der einzelnen Fachbereiche in diesem Unternehmen. Was beides für Produktteams spricht. Im Gegensatz dazu, sind im Unternehmen B alle EntwicklerInnen in fachbereichsübergreifenden Teams strukturiert und arbeiten somit auch in mehreren Fachbereichen. Zudem können hier Kundennutzen aufgrund der ähnlichen Technologien in den Teams auch teamübergreifend entwickelt werden.

## **7.4. Zusammenfassung**

Ziel der Interviews ist es, zuerst einen Überblick über die aktuellen Teamstrukturen der Teams zu erstellen. Zudem sollen auch jegliche Kriterien, welche die Auswahl

beeinflussen und auch die Motivationen dahinter abgefragt werden. Anschließend wird in den Interviews auf die einzelnen Kategorien, welche sich aufgrund der Vorgehensmuster bilden, eingegangen, um hier die spezifischen Antworten zu erlangen. Teilweise werden die Fragen sehr eindeutig in den Unternehmen beantwortet, aber teilweise auch sehr divers. Außerdem stellt sich heraus, dass nicht alle Kategorien wie sie aktuell aufgebaut sind, konkret für entweder Produkt- oder Projektteams sprechen, sondern dadurch die Vorgehensmuster angepasst oder eventuell sogar verworfen werden müssen.

Prinzipiell sind alle Interviews sehr gut verlaufen und es können in allen sechs Interviews stets alle Kategorien befragt werden. Somit ergeben sich hier keine Lücken bezüglich der Informationen. Die bisher definierten Vorgehensmuster stellen gemeinsam mit den Ergebnissen dieses Kapitels die weitere Diskussionsgrundlage für die Erstellung der Handlungsempfehlungen dar.

## 8. Handlungsempfehlungen

Aufgrund der in Kapitel 6 definierten Vorgehensmuster und der Ergebnisse in Abschnitt 7.3 werden in diesem Kapitel die Hypothesen des Abschnittes 6.6 geprüft. Anschließend werden durch die Hypothesenprüfung und die Ergebnisse der Interviews die Vorgehensmuster in Handlungsempfehlungen übergeführt.

### 8.1. Hypothesenprüfung

Aufgrund der Aussagen der InterviewpartnerInnen und die darauffolgende Inhaltsanalyse können in diesem Abschnitt die einzelnen Hypothesen und Vorgehensmuster geprüft werden.

HT1 besagt, dass die Zeit, Risiko, Einführungszeit wie auch die Eigenschaften zur fachbereichsübergreifenden Entwicklung ausschlaggebend für die Strukturierung der Teams sind. Diese Hypothese wird verworfen. Dies beruht darauf, dass in beiden Unternehmen weder der Zeitdruck noch das Risiko ein Kriterium für die Strukturierung der Teams sind. In beiden Fällen arbeiten die Unternehmen mit Projekt- oder Produktteams an Projekten mit hohem Zeitdruck und auch hohem Risiko. Was dennoch festgehalten werden kann ist, wenn ein Projekt fachbereichsübergreifende Tätigkeiten erfordert um einen Kundennutzen zu generieren, soll man das Team definitiv als Projektteam strukturieren. Auch die Minimierung der Einführungszeit stellt bei der Befragung ein eindeutiges Kriterium für die Auswahl dar. Somit muss VM1 nicht komplett verworfen, jedoch dementsprechend angepasst werden.

Sowie HT1 wird auch HT2 in ihrer aktuellen Definition verworfen. Die Anzahl der verwendeten Technologien kann nicht als eindeutiges Kriterium für die Wahl der Teamstruktur im Zuge der Interviews festgestellt werden. Die Frage bezüglich der Anzahl der verwendeten Technologien wird in beiden Unternehmen sehr divers beantwortet und somit kann es keiner einzelnen Teamstruktur zugewiesen werden. Hingegen die Komplexität und Spezialisierung der einzelnen Fachbereiche oder Technologien kann als Kriterium für die Auswahl dienen. So weist Unternehmen A einzelne höchst spezialisierte Fachbereiche auf und hat auch die Teams dementsprechend als Produktteams strukturiert. Im Unternehmen B sind die Teams als Projektteams



strukturiert. Auch hier weisen die einzelnen Fachbereiche keine hohe Spezialisierung oder Komplexität auf.

HT3 besagt, sowohl die Größe der Projekte beziehungsweise Teilaufgaben als auch die gleichmäßige Auslastung der Ressourcen sind ausschlaggebend auf die Wahl der Teamstruktur. Bei den befragten Unternehmen trifft keines der beiden Kriterien auf die Auswahl der Struktur zu. So sind beide Unternehmen generell so gut ausgelastet, dass es kein Problem darstellt, die einzelnen Fachbereiche auch dementsprechend gleichmäßig auszulasten. Zudem wird angenommen, dass die Projekte den Teams angepasst werden und somit auch die Größe der Projekte beziehungsweise der Teilaufgaben auch kein Kriterium für die Wahl der Teamstruktur darstellt. Es wird vermutet, dass diese Kriterien dennoch ausschlaggebend sind für die Wahl der Struktur, jedoch in einem anderen Stadium der Entwicklung. Beide befragten Unternehmen stehen nicht am Anfang der Entwicklung und haben bereits beide gut eingespielte Teams sowie laufende Projekte. Wenn die Unternehmen am Beginn der Reise stehen, ein Projekt vorgegeben haben und müssen dementsprechend die Teams strukturieren, dann kann die Größe und auch die Auslastung ausschlaggebend sein. Auf die Frage der Gründe für die Strukturierung, wird dies jedoch von keinem der beiden Unternehmen genannt. Dadurch werden für diese Arbeit sowohl HT3 als auch das gesamte Vorgehensmuster VM3 verworfen.

HT4 lautet: „Das Design von zukünftigen Schnittstellen ist ausschlaggebend für die Strukturierung von Scrum Teams“. Um diese Hypothese zu prüfen wird im Laufe der Interviews die Frage gestellt, wie gut zukünftige Schnittstellen vorab designet werden können. Diese Frage wird sowohl im Unternehmen A als auch im Unternehmen B divers beantwortet. Lässt man die Aussage von Interviewpartner 1 weg, welche sich auf den Idealzustand in der Zukunft bezieht, dann müssten im Unternehmen A die Teams eher als Projektteams und nicht als Produktteams strukturiert sein. Dies ist jedoch nicht der Fall. Auch im Unternehmen B wird die Frage divers beantwortet. Wenn man bei drei Antworten zwei als Mehrheit zählt, kann man sagen, dass das Design vorab eher schlechter als besser getroffen wird. Aufgrund des schlechten Designs vorab, ist somit die Projektteam Struktur für Unternehmen A die passendere Struktur. Dennoch wird auch das Design nicht von den Interviewpartnerinnen/Interviewpartnern als Kriterium erwähnt und durch die diversen Aussagen kann dadurch keine konkrete Handlungsempfehlung erstellt werden. Daher werden auch HT4 und VM4 verworfen.

Die letzte Hypothese, HT5 wird beibehalten. In beiden Unternehmen zeigt sich deutlich, dass das Können und auch der Wille der EntwicklerInnen, fachbereichsübergreifend zu arbeiten, für die Strukturierung ausschlaggebend ist. So hat Unternehmen A hauptsächlich Spezialistinnen/Spezialisten und auch die Teams dementsprechend nach Fachbereichen getrennt. Hingegen hat Unternehmen B hauptsächlich EntwicklerInnen, welche fachbereichsübergreifend arbeiten können und auch wollen und somit die Teams als Projektteams strukturiert. Natürlich haben beide Teamstrukturen ihre Vor- und Nachteile, welche nichtsdestotrotz abgewogen werden müssen. Es

lässt sich aber sagen, wenn man EntwicklerInnen hat, welche fachbereichsübergreifend arbeiten können und wollen, macht es definitiv Sinn, die Teams als Projektteams zu strukturieren. Wenn man nur Spezialistinnen/Spezialisten hat, macht es keinen Sinn die EntwicklerInnen in das fachbereichsübergreifende Denken und Handeln aufgrund von Projektteams zu zwingen.

## 8.2. Konkrete Handlungsempfehlungen

Resultierend aus den erstellten Vorgehensmustern und den zuvor geprüften Hypothesen, werden in diesem Abschnitt die Handlungsempfehlungen definiert. Diese sind folgend als Auflistung zur übersichtlichen Darstellung angeführt:

- Ist ein Projekt stark fachbereichsübergreifend und benötigt somit mehrere Fachbereiche, um einen Kundennutzen zu generieren, soll das Team als Projektteam strukturiert werden.
- Wenn es ein Hauptziel ist, die Einführungszeit zu minimieren, soll das Team als Projektteam strukturiert sein.
- Weisen Technologien oder einzelne Bereiche eine hohe Komplexität auf oder benötigen höchste Spezialisierung, so sollen die Teams als Produktteams strukturiert sein.
- Haben EntwicklerInnen das Können und auch den Willen dazu, fachbereichsübergreifend zu arbeiten, sollen die Teams als Projektteams strukturiert sein.

Diese Handlungsempfehlungen ermöglichen es SoftwaremanagerInnen, die Entscheidung zu treffen, ob sie ihre Teams eher als Projekt- oder als Produktteams strukturieren sollen. Die Handlungsempfehlungen umfassen jedoch nur vier Kriterien, welche im Zuge dieser Arbeit erarbeitet werden. Sie stellen somit zwar eine Hilfestellung, aber keine vollständige Liste zur Entscheidung der Teamstruktur dar.

## 9. Zusammenfassung

Die Beliebtheit agiler Entwicklung steigt in den letzten Jahren stetig, besonders in der Softwareentwicklung. Im speziellen ist Scrum eines der bedeutendsten und populärsten Frameworks für das Management agiler Entwicklungen. Im Zuge der agilen Entwicklung und auch im klassischen Projektmanagement spielt das Team für die Entwicklung eine der größten Rollen. Dies führt dazu, dass auch die Strukturierung der Teams eine immer wichtigere Rolle spielt, da die Ressourcen meist knapp sind und die Teams so effizient wie möglich arbeiten sollen. In dieser Arbeit werden Teamstrukturen vorgestellt, welche die effiziente Umsetzung ermöglichen sollen. Für die Entscheidung der Auswahl der Teamstruktur werden in dieser Arbeit Kriterien erarbeitet, welche dafür ausschlaggebend sind. Daraus resultieren Handlungsempfehlungen, welche sich besonders an SoftwaremanagerInnen richten.

Es gibt bereits im klassischen Projektmanagement viele verschiedene Varianten wie man die Teams strukturieren kann. Zudem gibt es auch in den agilen Methoden zwei Strukturen, welche sich etabliert haben. Im Zuge dieser Arbeit werden die Gemeinsamkeiten und auch Unterschiede der klassischen Strukturen mit den agilen verglichen, um somit neue allgemeingültige Strukturen zu erzeugen. Zusätzlich zu den verschiedenen Strukturen gibt es auch Frameworks, welche die Skalierung in den agilen Methoden unterstützen sollen. Auch diese werden im Zuge dieser Arbeit betrachtet.

Bei den klassischen Teamstrukturen leiten sich alle von den zwei Hauptstrukturen, der funktionalen und der Matrix Struktur, ab. Dadurch werden für den weiteren Vergleich in dieser Arbeit die Stab-Linienorganisation, die Matrixorganisation und die reine Projektorganisation als die wichtigsten Strukturen definiert. Aufgrund dieser Einschränkung können bereits Kriterien für die Auswahl der klassischen Strukturen genannt werden und auch die dazu passenden Einsatzmöglichkeiten. Zusätzlich zu den klassischen Strukturen werden auch die agilen Strukturen vorgestellt. Auch hier ergeben sich durch die Vor- und Nachteile der beiden Strukturen bereits Einsatzmöglichkeiten, woraus Kriterien abgeleitet werden können.

Die Betrachtung der Skalierungs-Frameworks zeigt, dass diese sich hauptsächlich auf die Arbeit zwischen den Teams fokussieren und wie die Koordination der Teams am besten funktioniert. Dies dient dazu, um am Ende ein gemeinsames integriertes potentiell auslieferbares Produktinkrement erstellen zu können. Für die konkrete

Entscheidung einer Teamstruktur findet man in den Skalierungs-Frameworks wenig Informationen. Diese verweisen zumeist nur darauf hin, dass die Teams wenig Koordinations- und Kommunikationsaufwand aufweisen sollen. Die meisten der betrachteten Frameworks bauen auf Scrum auf, wodurch diese auch wie Scrum selbst, hin zu Feature Teams tendieren. Diese Form der Strukturierung erlaubt es dem Team eigenständig Kundennutzen umzusetzen und minimiert den Koordinationsaufwand zwischen den Teams. Für die konkrete Entscheidung schreiben die meisten jedoch, dass es ein iterativer Lernprozess ist, wie man die Teams am besten strukturiert.

Beim Vergleich der klassischen und agilen Methoden stellt sich heraus, dass es einige Unterschiede zwischen den beiden Methoden gibt. Diese sind nicht alle in dieser Arbeit erwähnt und auch für die Beantwortung der initialen Forschungsfrage nicht relevant. Vielmehr ist die Gemeinsamkeit der Strukturen der beiden Methoden ausschlaggebend. So stellt sich in beiden Bereichen die Frage, ob man Teams projektübergreifend im selben Fachbereich oder besser fachbereichsübergreifend im selben Projekt beschäftigen soll. Daher kann die initial erstellte Arbeitshypothese verworfen werden. Aus dieser Erkenntnis werden zwei Teamstrukturen stellvertretend für die agilen und klassischen Methoden vorgestellt. Die Projektteams und die Produktteams. Erstere arbeiten fachbereichsübergreifend im selben Projekt und zweitere arbeiten projektübergreifend im selben Fachbereich. Beide dieser Strukturen haben somit auch ihre Vor- und Nachteile. Dadurch kann eine Tabelle mit Einsatzmöglichkeiten für die beiden Strukturen erstellt werden.

Aufgrund der zuvor erstellten Tabelle der Einsatzmöglichkeiten von Produktteams und Projektteams können erste Vorgehensmuster erstellt werden. Diese versuchen Kriterien für die Einsatzmöglichkeiten zu definieren. Diese Kriterien werden kategorisiert, gruppiert und in Vorgehensmuster zusammengefasst. Dazugehörig werden verschiedene Hypothesen aufgestellt, welche es zu prüfen gilt.

Aufgrund der erstellten Hypothesen und Vorgehensmuster wird ein Leitfragebogen erstellt. Dieser dient dazu, um Interviews mit Expertinnen/Experten in Scrum und Softwareentwicklung zu führen. Aufgrund der Interviews wird eine qualitative Analyse der definierten Kriterien durchgeführt. Durch die Ergebnisse werden die Hypothesen geprüft und die dazugehörigen Vorgehensmuster entweder verworfen oder angepasst. Es stellt sich heraus, dass die meisten Vorgehensmuster mehrere Kriterien in einem Vorgehensmuster zusammengefasst haben, was sich als nicht tauglich herausstellt. Aufgrund dieser Erkenntnis werden aus den Vorgehensmuster und den neu gewonnen Erkenntnissen konkrete Handlungsempfehlungen erstellt. Somit sind aus den erstellten Vorgehensmuster VM1, VM2, VM3, VM4 und VM5 die Handlungsempfehlungen HE1, HE2, HE3 und HE4 entstanden. Diese können von Softwaremanagerinnen/Softwaremanagern für die Auswahl der Teamstruktur als Projektteam oder Produktteam genutzt werden.

## 9.1. Kritische Reflexion

Im Zuge dieser Arbeit werden sechs Interviews mit Personen aus zwei unterschiedlichen Unternehmen geführt. Um die externe Validität zu steigern, werden die Unternehmen so gewählt, dass zumindest je Teamstruktur ein Unternehmen vorhanden ist. Zusätzlich zu den Teamstrukturen, sind auch die Branchen der Unternehmen unterschiedlich, wodurch branchenspezifische Eigenschaften keinen starken Einfluss haben können. Außerdem wird versucht, die InterviewpartnerInnen von den verschiedenen beteiligten Jobpositionen zu befragen. So wird, angefangen von der/dem IT-LeiterIn bis hin zu der/dem EntwicklerIn, deren Meinung zum Thema befragt, um auch hier die unterschiedlichen Sichtweisen zu betrachten. Qualitative Methoden, wie zum Beispiel Interviews, orientieren sich eher an Einzelfällen (Mayring, 2015). Um die externe Validität weiter zu verbessern und die Ergebnisse nicht nur auf diese zwei Unternehmen zu beziehen, benötigt es eine größere Anzahl an Interviewpartnerinnen/Interviewpartnern.

Im Zuge der Interviews hat sich herausgestellt, dass es nicht sinnvoll ist, die Hypothesen und somit auch die Vorgehensmuster auf mehr als ein Kriterium zu beziehen. Es wird versucht, die Kriterien in Kategorien zu gruppieren und aus diesen Kategorien die Vorgehensmuster zu erstellen. Bei der Befragung stellt sich heraus, dass teilweise Kriterien ausschlaggebend sind und andere nicht. Dadurch müssen die meisten Hypothesen und auch Vorgehensmuster entweder ganz verworfen oder zumindest angepasst und einige der Kriterien herausgenommen werden. Zu den Kriterien, welche sich als nicht ausschlaggebend herausstellen, kommen auch neue Kriterien im Zuge der Interviews hinzu. Diese werden teilweise nur in einem Unternehmen erwähnt, wodurch sie nicht mit in die Handlungsempfehlungen mit aufgenommen werden können. Diese können jedoch sehr wohl für weitere Unternehmen für die Strukturierung ausschlaggebend sein, werden aber in dieser Arbeit nicht weiter bearbeitet.

Einige der Fragen des Interviewleitfadens können je nach InterviewpartnerIn unterschiedlich interpretiert und auch beantwortet werden. So ist zum Beispiel die Frage nach dem Risiko und Zeitdruck eine sehr subjektive Frage. Damit solche Kriterien sinnvoll verglichen werden können, müssen konkrete Vergleichswerte geschaffen werden, um diese auch zu quantifizieren. Mit den aktuellen Aussagen wird zwar gesagt, der Zeitdruck ist hoch oder niedrig, aber was hoch und niedrig bedeutet ist je nach InterviewpartnerIn vermutlich unterschiedlich.

Die Handlungsempfehlungen, ob ein Team als Projektteam oder Produktteam strukturiert werden soll, beruhen in dieser Arbeit hauptsächlich auf fachlichen Kriterien. Im Zuge dieser Arbeit werden jegliche persönliche Faktoren der Teammitglieder nicht mitbetrachtet. Diese spielen jedoch für die Teambildung eine wesentliche Rolle. Zudem befinden sich auch die Unternehmen in verschiedenen Stadien. Es kann sein, dass ein Unternehmen erst bei der Gründung ist und ein weiteres Unternehmen bereits seit einigen Jahren erfolgreich entwickelt und umstrukturieren möchte. Dies

betrifft nicht nur die befragten Unternehmen, sondern auch die Unternehmen, für welche das Ergebnis dieser Arbeit dienen soll. Auch das macht einen großen Unterschied für die Entscheidung der Strukturierung aus.

## 9.2. Ausblick

Die Entscheidung ob ein Team als Projektteam oder Produktteam strukturiert werden soll, kann durch die Handlungsempfehlungen dieser Arbeit getroffen werden. Diese Handlungsempfehlungen erfassen nicht alle Kriterien, welche für die Auswahl entscheidend sind. Daher ist naheliegend, dass man eine weiterführende Erörterung der weiteren Kriterien auf Basis dieser Arbeit erstellen kann. Dies kann dadurch passieren, dass man mehr Unternehmen befragt, welche sich in anderen Branchen, anderen Team Konstellationen und anderen Stadien befinden. Die Fragen sollen offen zu beantworten sein, damit man neue Kriterien direkt von den Unternehmen bekommt. Diese können in weiterer Folge in Handlungsempfehlungen strukturiert werden, welche man mit einer quantitativen Forschung, wie zum Beispiel einem Fragebogen, versucht zu quantifizieren und auch zu validieren.

Zusätzlich zur Vollkommenheit der Kriterien geben die aktuellen Handlungsempfehlungen auch eher eine Tendenz, als eine klare Entscheidung vor. Eine weitere Möglichkeit auf dieser Arbeit aufzusetzen stellt daher dar, dass man einen Entscheidungsbaum ausarbeitet, wodurch am Ende eine konkrete Auswahl getroffen wird. Dafür müssen die Kriterien und dazugehörigen Vorteile der Teamstrukturen priorisiert und in eine Reihenfolge gebracht werden. Mit den aktuellen Handlungsempfehlungen ist es möglich, durch die Beantwortung zweier Fragen, auch zwei unterschiedliche Strukturen vorgeschlagen zu bekommen. Die Priorisierung der Themen soll somit die Fragen aneinander Reihen und in einer einzigen Teamstruktur enden. Aktuell ist es immer ein abwägen der Vor- und Nachteile der einzelnen Strukturen für Teams.

# A. Anhang

## A.1. Interviewleitfaden

---

**Name:**

**Alter:**

**Unternehmen:**

**Position im Unternehmen:**

**Größe des Unternehmens:**

---

### 1. Demografisch

- (a) Wie viele Jahre sind Sie in Ihrer Position bereits tätig?
- (b) Welche Ausbildung besitzen Sie hinsichtlich Scrum? Besitzen Sie eine Zertifizierung für Scrum und falls ja, welche?
- (c) Wie lange arbeiten Sie bereits mit Scrum?

### 2. Organisation von Teams

- (a) Wie viele Teams gibt es in Ihrem Unternehmen / Ihrem Bereich?
- (b) Wie groß sind die Teams?
- (c) Wie sind diese Teams strukturiert beziehungsweise aufgeteilt?

- (d) Aufgrund welcher Kriterien wurde diese Aufteilung erstellt beziehungsweise was war die Motivation dahinter? Gibt es bestimmte Vorteile die man sich dadurch erhofft hatte?
- (e) Würden Sie die Entscheidung treffen können die Teams erneut zu strukturieren, würden Sie die Teams aus heutiger Sicht anders aufteilen? Falls ja, warum?

### 3. Art des Projektes

- (a) Arbeiten die EntwicklerInnen an Projekten mit hohem Risiko oder Zeitdruck?
- (b) Ist eines der Ziele die Einführungszeit neuer Releases zu verkürzen?
- (c) Kann ein Kundennutzen rein durch einen Fachbereich generiert werden?

### 4. Technische Komplexität des Unternehmens

- (a) Arbeiten die Teams in sehr spezialisierten Fachbereichen?
- (b) Wie viele unterschiedliche Technologien haben Sie in Verwendung?
- (c) Wie komplex würden Sie sagen, sind die einzelnen Fachbereiche in denen Ihre EntwicklerInnen tätig sind?

### 5. Größe der Projekte beziehungsweise Teilaufgaben

- (a) Kann ein kompletter Kundennutzen durch ein einzelnes Team generiert werden oder benötigen Sie dazu mehr als ein Team?
- (b) Wie gut können die einzelnen Fachbereiche ausgelastet werden?

### 6. Zukünftiges Design der Schnittstellen

- (a) Wie gut können zukünftige Kundenanforderungen geplant und somit auch die Design Entscheidungen getroffen werden?

### 7. Kompetenz der Entwickler

- (a) Sind die EntwicklerInnen Cross-Functional und arbeiten in mehreren Fachbereichen? Falls ja, wie viele davon?

### 8. Abschluss

- (a) Gibt es etwas zur Strukturierung von Scrum Teams, dass Sie noch sagen möchten und noch nicht erwähnt wurde?



# Akronyme

ART	Agile Release Train
CDO	Chief Digital Officer
CIO	Chief Information Officer
CTO	Chief Technology Officer
LeSS	Large-Scale Scrum
SAFe	Scaled Agile Framework
SoS	Scrum of Scrums
UIX	User Interaction and Experience

# Abbildungsverzeichnis

1.1. Grafische Gegenüberstellung von Component und Feature Teams (vgl. The LeSS Company B.V., 2020a) . . . . .	3
1.2. Grafische Gegenüberstellung von Projekt- und Produktteams (Quelle: eigene Darstellung) . . . . .	4
2.1. Schematische Darstellung einer funktionalen Organisation (in Anlehnung an Abbot & Fisher, 2010) . . . . .	9
2.2. Schematische Darstellung einer Stab-Linienorganisation (in Anlehnung an Bachmann et al., 2019) . . . . .	11
2.3. Schematische Darstellung einer Matrixorganisation (in Anlehnung an Abbot & Fisher, 2010) . . . . .	12
2.4. Schematische Darstellung einer Multiprojektorganisation (Quelle: eigene Darstellung) . . . . .	14
2.5. Schematische Darstellung einer reinen Projektorganisation (vgl. Bachmann et al., 2019) . . . . .	15
2.6. Schematische Darstellung einer Spartenorganisation (vgl. Bergmann & Garrecht, 2008) . . . . .	16
3.1. Schematische Darstellung eines Feature Teams (vgl. The LeSS Company B.V., 2020a) . . . . .	21
3.2. Schematische Darstellung eines Component Teams (vgl. The LeSS Company B.V., 2020a) . . . . .	24
4.1. Schematische Darstellung von mehreren Teams in der Herangehensweise von Scrum of Scrums (vgl. Agilest LLC, 2020d) . . . . .	29
5.1. Schematische Darstellung eines Produktteams (Quelle: eigene Darstellung) . . . . .	37
5.2. Schematische Darstellung eines Projektteams (Quelle: eigene Darstellung) . . . . .	39
7.1. Schematische Darstellung der Strukturierung der Teams im Unternehmen A und B der InterviewpartnerInnen (Quelle: eigene Darstellung) .	51

# Tabellenverzeichnis

2.1. Kriterien zur Auswahl der Organisationsform im klassischen Projektmanagement (vgl. Heinrich & Lehner, 2005) . . . . .	18
3.1. Feature Teams versus Component Teams (vgl. Larman & Vodde, 2010)	26
3.2. Unterschiede von Feature Teams zu konventionellen Projekt beziehungsweise Feature Gruppen (vgl. Larman & Vodde, 2010) . . . . .	27
5.1. Einsatzmöglichkeiten von Projekt- und Produktteams (Quelle: eigene Darstellung) . . . . .	41
7.1. Übersicht der InterviewpartnerInnen zu Unternehmen und Team Struktur (Quelle: eigene Darstellung) . . . . .	50

# Literaturverzeichnis

- Abbot, M. L. & Fisher, M. T. (2010). *The art of scalability - scalable web architecture, processes, and organizations for the modern enterprise*. Boston: Addison-Wesley.
- Agilest LLC. (2020a). *Scaled professional scrum - nexus*. Zugriff am 17. May. 2020 auf <https://www.agilest.org/scaled-agile/nexus-framework/>
- Agilest LLC. (2020b). *Scaling agile with less (large-scale scrum)*. Zugriff am 17. May. 2020 auf <https://www.agilest.org/scaled-agile/large-scale-scrum-less/>
- Agilest LLC. (2020c). *Scaling agile with safe (scaled agile framework)*. Zugriff am 17. May. 2020 auf <https://www.agilest.org/scaled-agile/safe-framework/>
- Agilest LLC. (2020d). *Scaling agile with scrum of scrums (sos)*. Zugriff am 17. May. 2020 auf <https://www.agilest.org/scaled-agile/scrum-of-scrums/>
- Bachmann, C., Huber, E., Hubmann, M., Kuster, J., Lippmann, R., Schneider, E., ... Wüst, R. (2019). *Handbuch projektmanagement - agil - klassisch - hybrid*. Berlin: Springer Gabler.
- Baldwin, C., MacCormack, A. & Rusnak, J. (2011). *Exploring the duality between product and organizational architectures. a test of the "mirroring" hypothesis*. Zugriff am 17. May. 2020 auf [https://www.hbs.edu/faculty/Publication%20Files/08-039\\_1861e507-1dc1-4602-85b8-90d71559d85b.pdf](https://www.hbs.edu/faculty/Publication%20Files/08-039_1861e507-1dc1-4602-85b8-90d71559d85b.pdf)
- Beedle, M. & Schwaber, K. (2001). *Agile software development with scrum*. Pearson.
- Benefield, G., Deemer, P., Larman, C. & Vodde, B. (2012). *The scrum primer version 2.0: A lightweight guide to the theory and practice of scrum*. Zugriff am 21. May. 2020 auf [https://scrumprimer.org/scrumprimer20\\_small.pdf](https://scrumprimer.org/scrumprimer20_small.pdf)
- Bergmann, R. & Garrecht, M. (2008). *Organisation und projektmanagement*. Heidelberg: Physica-Verlag.
- Böhm, J. (2019). *Erfolgsfaktor agilität - warum scrum und kanban zu zufriedenen mitarbeitern und erfolgreichen kunden führen*. Wiesbaden: Springer Vieweg.
- Bühner, R. (2004). *Betriebswirtschaftliche organisationslehre*. Oldenburg.
- Bick, S., Heinzl, A., Scheerer, A. & Schimmer, T. (2015, 03). The effects of team backlog dependencies on agile multiteam systems: A graph theoretical approach. , 2015, 5124-5132.
- Bittner, K., Kong, P. & West, D. (2018). *The nexus framework for scaling scrum - continuously delivering an integrated product with multiple scrum teams*. Scrum.org.
- Bleek, W.-G. & Wolf, H. (2011). *Agile softwareentwicklung - werte, konzepte und methoden*. Heidelberg: dpunkt.verlag GmbH.

- Bourk, S. & Kong, P. (2016). *An introduction to the nexus framework*. Zugriff am 17. May. 2020 auf [https://scrumorg-website-prod.s3.amazonaws.com/drupal/2016-06/An%20Introduction%20to%20the%20Nexus%20Framework%20-%20June%202016\\_0.pdf](https://scrumorg-website-prod.s3.amazonaws.com/drupal/2016-06/An%20Introduction%20to%20the%20Nexus%20Framework%20-%20June%202016_0.pdf)
- Broy, M. & Kuhrmann, M. (2013). *Projektorganisation und management im software engineering*. Berlin: Springer Vieweg.
- CollabNet Inc. und VersionOne Inc. (2019). *13th annual state of agile report*. Zugriff am 17. May. 2020 auf <https://www.stateofagile.com/#ufh-i-521251909-13th-annual-state-of-agile-report/473508>
- Devine, D. (2002, 12). A review and integration of classification systems relevant to teams in organizations. *Group Dynamics: Theory, Research, and Practice*, 6, 291-310.
- Dunger, C., Kolbe, H., Schnell, M. & Schulz-Quach, C. (2013). *Der patient am lebensende: Eine qualitative inhaltsanalyse*.
- Emanuelsson, P., Lagerberg, L., Sandahl, K., Skude, T. & Ståhl, D. (2013, 10). The impact of agile principles and practices on large-scale software development projects: A multiple-case study of two projects at ericsson. In (S. 348-356).
- Greene, J. & Stellman, A. (2017). *Head first agile - a brain-friendly guide*. Sebastopol: O'Reilly Media, Inc.
- Heinrich, L. J. & Lehner, F. (2005). *Informationsmanagement - planung, Überwachung und steuerung der informationsinfrastruktur*. München: Oldenburg.
- Hochschule Koblenz in Kooperation mit der GPM Deutsche Gesellschaft für Projektmanagement e. V. (2017). *Status quo agile*. Zugriff am 17. May. 2020 auf [https://www.gpm-ipma.de/fileadmin/user\\_upload/GPM/Know-How/Studie\\_Status\\_Quo\\_Agile\\_2017.pdf](https://www.gpm-ipma.de/fileadmin/user_upload/GPM/Know-How/Studie_Status_Quo_Agile_2017.pdf)
- Knaster, R. & Leffingwell, D. (2019). *Safe distilled - applying the scaled agile framework for lean enterprises*. Addison-Wesley.
- KnowledgeHut. (2020a). *Scrum of scrums*. Zugriff am 17. May. 2020 auf <https://www.knowledgehut.com/tutorials/scrum-tutorial/scrum-of-scrums>
- KnowledgeHut. (2020b). *Scrum tutorial*. Zugriff am 17. May. 2020 auf <https://www.knowledgehut.com/tutorials/scrum-tutorial/feature-teams-vs-component-teams>
- Kraus, G. & Westermann, R. (2010). *Projektmanagement mit system - organisation, methoden, steuerung*. Wiesbaden: Gabler.
- Lang, M. & Scherber, S. (2019). *Der weg zum agilen unternehmen - wissen für entscheidender*. Hanser.
- Larman, C. & Vodde, B. (2008). *Scaling lean & agile development: Thinking and organizational tools for large-scale scrum*. Addison-Wesley.
- Larman, C. & Vodde, B. (2010). *Practices for scaling lean & agile development - large, multisite, and offshore product development with large-scale scrum*. Boston: Addison-Wesley.
- Leffingwell, D. (2007). *Scaling software agility: Best practices for large enterprises*. Addison-Wesley Professional.
- Mayring, P. (2015). *Qualitative inhaltsanalyse - grundlagen und techniken*. Weinheim

- und Basel: Beltz Verlag.
- McKenna, D. (2016). *The art of scrum: How scrum masters bind dev teams and unleash agility*. Texas: The Pragmatic Bookshelf.
- Mersino, A. (2018). *Agile project success rates are 2x higher than traditional projects (2019)*. Zugriff am 28. Jun. 2020 auf <https://vitalitychicago.com/blog/agile-projects-are-more-successful-traditional-projects/>
- Oehrich, M. (2019). *Wissenschaftliches arbeiten und schreiben: Schritt für schritt zur bachelor- und master-thesis in den wirtschaftswissenschaften*. Springer Gabler.
- Pohl, K. & Rupp, C. (2015). *Basiswissen requirements engineering: Aus- und weiterbildung nach ireb-standard zum certified professional for requirements engineering foundation level*. Heidelberg: dpunkt.verlag GmbH.
- Preußig, J. (2018). *Agiles projektmanagement - scrum, use cases, task boards & co*. Haufe Lexware GmbH.
- Pries, K. H. & Quigley, J. M. (2011). *Scrum project management*. CRC Press.
- Probst, G. J. B. (1993). *Organisation. strukturen, lenkungsinstrumente, entwicklungsperspektiven*. Landsberg/Lech: Verlag Moderne Industrie.
- Project Management Institute Inc. und Agile Alliance. (2017). *Agile practice guide*. Pennsylvania: Project Management Institute Inc.
- Putman, D. (2020). *Component teams vs feature teams*. Zugriff am 17. May. 2020 auf <https://www.agil8.com/blog/component-teams-vs-feature-teams/>
- Raithel, J. (2006). *Quantitative forschung*. VS Verlag für Sozialwissenschaft/GWV Fachverlag.
- Roock, S. & Wolf, H. (2018). *Scrum - verstehen und erfolgreich einsetzen*. dpunkt.verlag GmbH.
- Röpstorff, S. & Wiechmann, R. (2016). *Scrum in der praxis: Erfahrungen, problemfelder und erfolgskfaktoren*. dpunkt.verlag GmbH.
- Rubin, K. S. (2013). *Essential scrum: A practical guide to the most popular agile process*. Michigan: Addison-Wesley.
- Scaled Agile, Inc. (2020a). *Agile teams*. Zugriff am 17. May. 2020 auf <https://www.scaledagileframework.com/agile-teams/>
- Scaled Agile, Inc. (2020b). *Organizing by feature or component*. Zugriff am 17. May. 2020 auf <https://www.scaledagileframework.com/features-and-components/>
- Scaled Agile, Inc. (2020c). *Scaled agile framework (safe)*. Zugriff am 17. May. 2020 auf <https://www.scaledagileframework.com/>
- Scheerer, A. (2017). *Coordination in large-scale agile software development - integrating conditions and configurations in multiteam systems*. Baden-Württemberg: Springer.
- Schulte-Zurhausen, M. (2013). *Organisation*. Vahlen.
- Schwaber, K. (2004). *Agile project management with scrum*. Microsoft Press.
- Schwaber, K. & Scrum.org. (2018). *Nexus guide - der gültige leitfaden zur skalerung von scrum mit dem nexus: Die spielregeln*. Zugriff am 17. May. 2020 auf [https://scrumorg-website-prod.s3.amazonaws.com/drupal/2018-01/2018-Nexus-Guide-English\\_0.pdf?nexus-file=https%3A%2F%2Fscrumorg-website-prod.s3.amazonaws.com%2Fdrupal%](https://scrumorg-website-prod.s3.amazonaws.com/drupal/2018-01/2018-Nexus-Guide-English_0.pdf?nexus-file=https%3A%2F%2Fscrumorg-website-prod.s3.amazonaws.com%2Fdrupal%2F)

- 2F2018-01%2F2018-Nexus-Guide-English\_0.pdf
- Schwaber, K. & Sutherland, J. (2017). *Der scrum guide*. Zugriff am 17. May. 2020 auf <https://scrumguides.org/docs/scrumguide/v2017/2017-Scrum-Guide-German.pdf>
- Scrum Alliance. (2017). *State of scrum 2017-2018 - scaling and agile transformation*. Zugriff am 17. May. 2020 auf [https://www.scrumalliance.org/ScrumRedesignDEVSite/media/ScrumAllianceMedia/Files%20and%20PDFs/State%20of%20Scrum/2017-SoSR-Final-Version-\(Pages\).pdf](https://www.scrumalliance.org/ScrumRedesignDEVSite/media/ScrumAllianceMedia/Files%20and%20PDFs/State%20of%20Scrum/2017-SoSR-Final-Version-(Pages).pdf)
- Scrum.org und Age of Product. (2019). *Scrum master trends 2019*. Zugriff am 17. May. 2020 auf <https://age-of-product.com/scrum-master-trends-report-2019-free-download/>
- Software Quality Lab GmbH. (2020). *Feature-teams oder komponenten-team? oder doch ganz anders?* Zugriff am 17. May. 2020 auf <https://www.software-quality-lab.com/wissen/knowledge-letter/veroeffentlichte-knowledge-letter/>
- Spanner, C. (2020). *Scrum of scrums - how to scale scrum*. Zugriff am 17. May. 2020 auf <https://www.atlassian.com/agile/scrum/scrum-of-scrums>
- Strasser, J. (2020). *Ressourcenmanagement erfolgreich einführen*. Zugriff am 17. May. 2020 auf <https://www.business-wissen.de/artikel/projektmanagement-ressourcenmanagement-erfolgreich-einfuehren/>
- The LeSS Company B.V. (2020a). *Less framework: Feature teams*. Zugriff am 17. May. 2020 auf <https://less.works/less/structure/feature-teams.html>
- The LeSS Company B.V. (2020b). *Less framework: Teams*. Zugriff am 17. May. 2020 auf <https://less.works/less/structure/teams.html>
- The LeSS Company B.V. (2020c). *Less framework: Why less?* Zugriff am 17. May. 2020 auf <https://less.works/less/framework/why-less.html>
- Vahs, D. (2007). *Organisation: Einführung in die organisationstheorie und -praxis*. Schäffer-Poeschel.
- Visual Paradigm. (2020). *Feature team vs component team in agile*. Zugriff am 17. May. 2020 auf <https://www.visual-paradigm.com/scrum/feature-team-vs-component-team-in-agile/>
- Vogt, S. & Werner, M. (2014). *Forschen mit leitfadeninterviews und qualitativer inhaltsanalyse*. Zugriff am 21. May. 2020 auf [https://www.th-koeln.de/mam/bilder/hochschule/fakultaeten/f01/skript\\_interviewsqualinhaltsanalyse-fertig-05-08-2014.pdf](https://www.th-koeln.de/mam/bilder/hochschule/fakultaeten/f01/skript_interviewsqualinhaltsanalyse-fertig-05-08-2014.pdf)
- Winkelhofer, G. (2005). *Management- und projekt-methoden - ein leitfaden für die it organisation und unternehmensentwicklung*. Berlin: Springer.
- Wintersteiger, A. (2018). *Scrum: Schnelleinstieg*. entwickler.press.