FernUniversität in Hagen

**Die Konzeption von Standardtypen zur Berücksichtigung von Risiken entlang eines Produktlebenszyklus im Release-Management**

Masterarbeit

Vorgelegt der Fakultät für Wirtschaftswissenschaft

der FernUniversität in Hagen

Lehrstuhl für Betriebswirtschaftslehre,

insbesondere Informationsmanagement

Von: Steve Lohr

Naumannstraße 10

01309 Dresden

Matrikelnummer: 9066454

Gutachter: Univ.-Prof. Dr. Ulrike Baumöl

Betreuer: Christian Grawe, M.Sc.

Abgabe am: 12.08.2016

Wintersemester 2015, 7. Studiensemester

Inhaltsverzeichnis

[Abbildungsverzeichnis II](#_Toc457246775)

[Tabellenverzeichnis III](#_Toc457246776)

[Abkürzungsverzeichnis IV](#_Toc457246777)

[1 Einleitung 1](#_Toc457246778)

[2 Grundlagen 3](#_Toc457246779)

[2.1 Innovation zur Existenzsicherung 3](#_Toc457246780)

[2.2 Innovationssteuerung durch Produkt-Management 6](#_Toc457246781)

[2.3 Der Produktlebenszyklus 7](#_Toc457246782)

[2.4 Produktveröffentlichung durch Standards im Release-Management 11](#_Toc457246783)

[3 Risiken im Produktlebenszyklus als Basis der Standardtypen 16](#_Toc457246784)

[3.1 Entwicklung 16](#_Toc457246785)

[3.2 Einführung 17](#_Toc457246786)

[3.3 Wachstum 18](#_Toc457246787)

[3.4 Reife 19](#_Toc457246788)

[3.5 Entsorgung 20](#_Toc457246789)

[3.6 Zusammenfassung der Risiken in den Phasen 20](#_Toc457246790)

[4 Faktoren des Release-Managements zum Umgang mit Risiken 24](#_Toc457246791)

[4.1 Transition Planning and Support 24](#_Toc457246792)

[4.2 Change Management 27](#_Toc457246793)

[4.3 Service Asset And Configuration Management 30](#_Toc457246794)

[4.4 Release And Deployment Management 30](#_Toc457246795)

[4.5 Service Validation And Testing 33](#_Toc457246796)

[4.6 Change Evaluation 37](#_Toc457246797)

[4.7 Knowledge Management 38](#_Toc457246798)

[4.8 Zusammenfassung der Faktoren des Release-Managements 40](#_Toc457246799)

[5 Ableitung der Standardtypen aus den Faktoren und Risiken 41](#_Toc457246800)

[5.1 Zeit 41](#_Toc457246801)

[5.2 Kosten 42](#_Toc457246802)

[5.3 Zusammenfassung der Standardtypen 44](#_Toc457246803)

[6 Kritische Würdigung der Standardtypen 46](#_Toc457246804)

[7 Evaluation der Zielerreichung und Ausblick 48](#_Toc457246805)

[8 Literaturverzeichnis 49](#_Toc457246806)

[Erklärung 51](#_Toc457246807)

Abbildungsverzeichnis

Abbildung 2.1: Zusammenhang Time to Market 5

Abbildung 2.2: Charakteristische Produktlebenszyklen 8

Abbildung 2.3: Zusammenhang Produkt zu Release 11

Abbildung 2.4: Überblick ITIL 15

Abbildung 4.1: Fehlerbehebungskosten 36

Tabellenverzeichnis

Tabelle 2.1: Zuordnung der verschiedenen Produktlebenszyklusphasen 10

Tabelle 3.1: Zuordnung Risikotypen zu Produktlebenszyklusphasen 21

Tabelle 3.2: Reduzierte Zuordnung Risikotypen zu Produktlebenszyklusphasen 22

Tabelle 4.1: Übersicht Prozessfaktoren und Ausprägungen 40

Tabelle 5.1: Zusammenfassung der Ausprägungen der Standardtypen 45

Abkürzungsverzeichnis

BEP Break-even-Point (deu.: Gewinnschwelle)

CAB Change Advisory Board

CHE Change Evaluation

CHM Change Management

CIO Chief Information Officer (deu.: Leiter Informationstechnik)

CMS Configuration Management System

COBIT Controlled Objectives for Information and Related Technology

Dev Development (deu.: Entwicklung)

DevOps Kofferwort aus den Abkürzungen Dev und Ops

IEC International Electrotechnical Commision

ISO International Organization for Standardisation

IT Informationstechnik (engl.: Information Technology)

ITIL IT Infrastructure Library

KLM Knowledge Management

Ops Operations (deu.: Betrieb)

RDM Release And Deployment Management

ROI Return-on-Investment

SACM Service Asset And Configuration Management

SVT Service Validation And Testing

TPS Transition Planning And Support

USP unique selling point (deu.: Alleinstellungsmerkmal)

>> Warum werden manche Wörter übersetzt und manche Nicht?

Klare Linie wäre, alle englischen ins Deutsche zu übersetzen, und die Deutschen nicht ins englische zu übersetzen

# Einleitung

„Nichts ist so beständig wie der Wandel“, sagte einst *Heraklit von Ephesos* (Ephesos kein Datum). Dieser Grundsatz gilt für jede Unternehmung, denn einerseits verändert sich stets die Umwelt und andererseits müssen sich Unternehmen fortlaufend selbst verändern, um Wettbewerbsvorteile gegenüber der Konkurrenz zu erarbeiten und dadurch langfristig erfolgreich zu sein (vgl. Kapitel 2.1). Diese eigenen Veränderungen erfolgen über Innovationen und werden über das Produkt-Management gesteuert und über Investitionen finanziert (vgl. Kapitel 2.2). Von entscheidender Bedeutung bei diesen Innovationen ist der Zeitpunkt, zu dem der Nutzen aus der Investition gezogen werden kann, welcher auch als „time to market“ bezeichnet wird (vgl. Kapitel 2.1). Aufgrund der anhaltenden Digitalisierung hat die Informationstechnik (IT) darauf einen immer größer werdenden Einfluss (vgl. Kapitel 2.4). Entscheidend für die schnelle Umsetzung der fachlichen Anforderungen ist die Flexibilität in der Softwareentwicklung (vgl. Kapitel 2.4). Aufgrund dieser Prämisse sind agile Vorgehensweisen, wie zum Beispiel Scrum, in der Softwareentwicklung mittlerweile etabliert (vgl. Kim, 2015, S. 2; Hammond, 2010; Lange & Diercks, 2015; Roock & Wolf, 2016, S. 446). Die Softwareentwicklung findet in vielen Fällen jedoch isoliert vom IT-Service-Management und damit dem Release-Management statt, da im Betrieb nicht die Innovation, sondern die Stabilität der Produktionsumgebung im Vordergrund steht (vgl. Kapitel 2.4). Diese Entkopplung führt letztendlich zu einer längeren „time to market“ und daher zu einer schlechteren Wettbewerbsfähigkeit (vgl. Puppet, 2016, S. 14 f.). Der Ansatz „DevOps“ versucht dieses Problem zu lösen, in dem die Isolation zwischen Entwicklung („Development“ – „Dev“) und IT-Service-Management („Operations“ – „Ops“) aufgehoben wird (vgl. Kapitel 2.4). Eine aktuelle Studie zeigt, dass mit dem DevOps-Ansatz die Vorlaufzeit für eine Änderung im IT-Betrieb, also einem Release, von Monaten zu Stunden verkürzt werden kann, wobei die Fehlerraten und die Ausfallkosten wesentlich niedriger ausfallen (vgl. Puppet, 2016, S. 14 f.; 48). Die typischen Fragestellungen im Release-Management bleiben jedoch trotzdem bestehen, denn die Balance von Kosten und Nutzen (vor allem Risikosenkung) ist auch hier entscheidend. Sollten Tests automatisiert werden? Für welche Änderungen sind Freigabeprozesse notwendig und welche können direkt umgesetzt werden? Kann jedes Release mittels derselben standardisierten Verfahren abgewickelt werden oder bedarf es individueller Prozesse? Sollte aufgrund der benannten Studie nun jedes Produkt gemäß dem DevOps-Ansatz behandelt werden, oder werden unterschiedliche Verfahrensweisen benötigt? Wenn ja, anhand welcher Kriterien lässt sich das ableiten? Das Standardrahmenwerk für das IT-Service-Management die IT Infrastructure Library (ITIL) und die internationale Norm ISO/IEC 20000 sind diesbezüglich zu generisch (vgl. Kapitel 2.4). Dadurch entsteht Unternehmen bei der Adaption von ITIL mittels Einführungsprojekten ein erhöhter Aufwand (vgl. Laser, 2012, S. 9; Lienemann, 2006, S. 57; Winniford, Conger, & Erickson-Harris, 2009, S. 158; Peasley & Fletcher, 2005, S. 5; Wonke-Stehle, 2012, S. 67 f.) (vgl. Newcombe, 2005; Pink Elephant, 2005, S. 1; itSMF e. V., 2007, S. 133; KBSt, 2007, S. 7). Durch die Konzeption von Standardtypen für Release-Management-Prozesse können diese Aufwände reduziert werden, da die Einführung vereinfacht und beschleunigt wird. Weiterhin können die Standardtypen bei der Identifizierung von Optimierungspotentialen für Unternehmen mit etablierten Prozessen hilfreich sein. Dies kann zu einer weiteren Standardisierung des IT-Service-Managements im Allgemeinen führen. Die Ableitung der Standardtypen kann von vielen verschiedenen Faktoren, wie zum Beispiel der Branche des Unternehmens, dem Einfluss des Produkts auf das Kerngeschäft, der Klassifizierung der verarbeitenden Daten oder der aktuellen Phase eines Produkts im Produktlebenszyklus, ausgehen. Die grundlegende Hypothese dieser Arbeit ist, dass aufgrund des direkten Zusammenhangs zwischen der Produktlebenszyklusphase und dem Innovationsbedarf eines Produkts, eine Unterscheidung für Release-Management-Prozesse ableitbar ist (vgl. Kapitel 2.2 f.). Ziel dieser Masterarbeit ist es daher, anhand der Phasen des Produktlebenszyklus Risiken abzuleiten, dazu passende Release-Management-Methoden zu finden und diese letztendlich zu Standardtypen zusammen zu fassen. Andere möglichen Faktoren neben dem Produktlebenszyklus finden keine Betrachtung und sollten daher Ziel weiterer wissenschaftlicher Betrachtungen innerhalb des Themenkomplex werden.

Um dieses Ziel zu erreichen, wird folgende Vorgehensweise angewendet. In Kapitel 2 erfolgt die Beschreibung der Grundlagen, damit ein einheitliches Verständnis der Thematik vorliegt. Innerhalb dieser Ausführungen wird auch die Ableitung eines Produktlebenszyklus als Basis für die weitere Bearbeitung vorgenommen. Die Phasen des abgeleiteten Produktlebenszyklus werden in Kapitel 3 hinsichtlich ihrer speziellen Risiken beleuchtet. Im darauffolgenden Kapitel 4 werden die unterschiedlichen Prozesse des Release-Managements vorgestellt und auf Faktoren zur Risikobehandlung untersucht. Die passende Zuordnung von Faktoren zu Risiken und deren Kombination zu Standardtypen erfolgt dann in Kapitel 5. Im Kapitel 6 werden die erarbeiteten Standardtypen bezüglich ihres Nutzens untersucht. Das beschriebene Vorgehen ist dem konstruktionsorientierten Forschungsansatz abgeleitet (vgl. Peffers, et al., 2006, S. 93). Im letzten Kapitel erfolgt eine Zusammenfassung der Ergebnisse, die Überprüfung der Erreichung der Zielstellung sowie die Beschreibung von möglichen Verbesserungs- und Forschungsansätzen.

# Grundlagen

Zur Einführung in die Thematik und zur Etablierung eines einheitlichen Begriffsverständnisses werden in den folgenden Unterkapiteln die Begriffe Innovation, Produkt-Management, Produktlebenszyklus und Release-Management erläutert und in Zusammenhang gebracht. Im Zuge dessen wird auch der für die spätere Risikoanalyse notwendige Produktlebenszyklus aus den unterschiedlichen vorliegenden Beschreibungen hergeleitet.

## Innovation zur Existenzsicherung

Zentrales Thema der Einleitung ist die Innovation. *Kairies* (vgl. 2004, S. 86) sieht Innovation als Erneuerung aus sich selbst heraus. *Schumpeter* (1997, S. 100 f.) beschreibt sie als Entwicklung und Durchsetzung von neuen Kombinationen aus Produktionsmitteln. Er unterscheidet dabei fünf verschiedene Fälle von Innovation (vgl. Schumpeter, 1997, S. 100 f.):

1. Herstellung eines neuen Guts oder eines Guts mit besserer Qualität
2. Einführung einer neuen Produktionsmethode
3. Erschließung eines neuen Absatzmarktes
4. Eroberung einer neuen Bezugsquelle von Produktionsmitteln
5. Neuorganisation durch Schaffung oder Aufbruch einer Monopolstellung

Die ersten beiden Fälle unterscheidet auch *Aumayr* (vgl. 2009, S. 322). Als Produktinnovation bezeichnet er neu entwickelte Produkte zur Befriedigung von Kundenbedürfnissen (vgl. Aumayr, 2009, S. 322). Die Prozessinnovation dagegen bezieht er auf die Leistungserstellungsprozesse, deren Innovation zu einer höheren Produktivität führt (vgl. Aumayr, 2009, S. 322). Prinzipiell zeichnet sich Innovation durch die Erreichung eines vorteilhafteren Zustands in mindestens einer Dimension im Vergleich zum Ausgangszustand aus, sei es durch eine höhere Nachfrage oder eine höhere Produktivität (vgl. Schumpeter, 1997, S. 16, 155, 208 ff.; Drucker, 2009, S. 32). Ohne diese Vorteile aus der Innovation, gäbe es letztendlich keinen Unternehmergewinn (vgl. Schumpeter, 1997, S. 236). Daher beschreibt *Drucker* (vgl. 2009, S. 20) Innovation auch als das Mittel der Unternehmer, um Nutzen aus Veränderungen zu generieren. Diese Fähigkeit unterscheidet Unternehmer von Managern, welche lediglich Bekanntes verwalten (vgl. Schumpeter, 1997, S. 216 f.). Die genannten geschaffenen Innovationsvorteile sind jedoch nicht von Dauer, da andere Unternehmer stets nachziehen (vgl. Schumpeter, 1997, S. 211). Dies führt zu einer hohen Dynamik auf den Märkten und zwingt Unternehmen permanent innovative und erfolgreiche Produkte zu vermarkten, um dauerhaft bestehen zu können (vgl. Kairies, 2004, S. 1 ff.; Lennertz, 2006, S. 11 f.; Aumayr, 2009, S. 11, 136, 325; Grawe, 2016, S. 133). Klassische Mittel zur Erfolgssicherung, wie Prozessbeschleunigung, Kostensenkung oder Organisationsveränderungen allein reichen dafür auf Dauer nicht aus (vgl. Lennertz, 2006, S. 7; Aumayr, 2009, S. 146). Innovation bedeutet jedoch im ersten Schritt immer eine Investition (vgl. Grawe, 2016, S. 133), entweder aus dem Unternehmen selbst heraus oder über eine Fremdfinanzierung, welche eine Priorisierung des Neuen über das Alte bedarf (vgl. Schumpeter, 1997, S. 103 ff., 148 ff.). Oftmals werden Innovationen daher von neuen Marktteilnehmern durchgesetzt, da die bestehenden Unternehmungen am altbekannten Status festhalten oder zu spät reagieren (vgl. Schumpeter, 1997, S. 101, 348; Drucker, 2009, S. 38; Christensen, 2000, S. 623). Die neuen Marktteilnehmer treten dann neben die etablierten Unternehmen und die Nachfrage verlagert sich nach und nach zu deren Vorteil (vgl. Schumpeter, 1997, S. 101, 155, 289, 312). So haben etablierte Unternehmen nur die Möglichkeit sich ebenfalls zu verändern oder aufzugeben (vgl. Schumpeter, 1997, S. 354 f.). Aufgrund der entwickelten Wertenetzwerke dieser etablierter Unternehmen sind Veränderungen allerdings nur bei sogenannten inkrementellen bzw. erhaltenden Innovationen wahrscheinlich (vgl. Christensen, 2000, S. 880). Für radikale bzw. revolutionäre Innovationen fehlt zu Beginn die Nachfrage der Bestandskunden und somit eine lukrative Marge (vgl. Christensen, 2000, S. 1629). Manager entscheiden daher im Sinne der inkrementellen Verbesserung (vgl. Christensen, 2000, S. 1689), wohingegen Unternehmer die Nachfrage mittels revolutionärer Innovation schaffen (vgl. Schumpeter, 1997, S. 100, 152). Möglicherweise müssen sie dafür einen hohen Widerstand überwinden und die Produkte dem Markt aufdrängen bzw. zu geringer oder negativer Marge anbieten (vgl. Schumpeter, 1997, S. 215). Trotz dieser Maßnahmen ist es möglich, dass die Neuerung keine Nachfrage erfährt, doch Innovation bedingt diese Initiative und dieses Risiko (vgl. Schumpeter, 1997, S. 213, 331). Dabei sind erfolgreiche Innovatoren nicht risikofreudig, denn „defending yesterday – that is, not innovating – is far more risky than making tomorrow“, wie *Drucker* (2009, S. 138) sagte. Unter einem Risiko wird dabei das Produkt von Eintrittswahrscheinlichkeit und der meist negativen Auswirkung eines Ereignisses bezeichnet (vgl. Chapman & Ward, 2011, S. 279 f.; Gassmann & Kobe, 2006, S. 10; Simon, 2010, S. 19; Proske, 2004, S. 27). *Festtag* (vgl. 2014, S. 5) unterscheidet die folgenden Möglichkeiten zum Umgang mit Risiken: Vermeidung, Reduktion, Optimierung, Transfer, Akzeptanz.

Langfristig setzen sich nur die Unternehmen durch, welche alle Phasen des Innovationsprozesses beherrschen und die Schlüsselfähigkeit besitzen, in kürzerer Zeit mehr erfolgreiche Produkte auf den Markt zu bringen (vgl. Kairies, 2004, S. 4 f.). Dies erklärt sich anhand folgendem Zusammenhang. Die hohe Dynamik auf den Märkten führt zur Verkürzung der Produktlebenszeit und zur Erhöhung der Entwicklungsaufwände (vgl. Kairies, 2004, S. 6; Aumayr, 2009, S. 325). Das bedeutet der Zeitraum in dem ein Unternehmen Gewinne aus der Produktvermarktung erwirtschaften kann sinkt und die Investitionskosten steigen (vgl. Kairies, 2004, S. 6). Dadurch steigt auch die Dauer der Pay-off-Zeit in der sich die Produktionskosten amortisieren, d. h. bis der Break-even-Point (BEP) erreicht wird. Somit wird die Zeit zwischen Produktidee und Markteintritt, die Time to Market, zum kritischen Wettbewerbsfaktor (vgl. Kairies, 2004, S. 6). Zur Veranschaulichung dieses zentralen Zusammenhangs dient Abbildung 2.1.



Abbildung .: Zusammenhang Time to Market

Quelle: in Anlehnung an (Kairies 2004, S. 7)

Der Faktor Time to Market wird außerdem durch das sogenannte Window of Opportunity entscheidend beeinflusst (vgl. Aumayr, 2009, S. 322). Nur wenn sich der Markt in der Zeit bis zum Markteintritt nicht weiterentwickelt, z. B. durch neue Technologien, andere Kundenbedürfnisse oder Konkurrenzprodukte ist die Produktveröffentlichung erfolgsbringend (vgl. Pichler, 2014, S. 89).

Innovation darf aufgrund dieser Bedeutung nicht durch Zufall entstehen, sondern bedingt ein systematisches Innovations-Management (vgl. Kairies, 2004, S. 86; Drucker, 2009, S. 34). Dies umfasst folgende Teilaufgaben (vgl. Kairies, 2004, S. 86):

* Schaffung eines Innovationsumfelds
* Auffinden innovativer Produkte
* erfolgreiche Realisierung und Vermarktung dieser Produkte

Quellen für Innovationschancen sind dabei laut *Drucker* (vgl. 2009, S. 34 ff.) in absteigender Reihenfolge gemäß ihrer Bedeutung:

1. unerwartete Dinge, wie Erfolg, Verlust oder Vorkommnisse
2. Inkongruenz zwischen Soll und Ist
3. Prozessveränderungsbedarfe
4. Branchen- und Marktveränderungen
5. Demografie
6. neues Wissen
7. veränderte Wahrnehmung und Bedeutung von Dingen

Die Steuerung des Innovation-Managements erfolgt im Unternehmen durch das Produkt-Management (vgl. Kairies, 2004, S. 1, 5, 87; Aumayr, 2009, S. 11, 322). Dies wird im nachfolgenden Kapitel betrachtet.

## Innovationssteuerung durch Produkt-Management

Im vorhergehenden Kapitel wurde die Bedeutung der Innovation für eine Unternehmung dargestellt. Produkte bzw. Güter haben dabei eine zentrale Rolle. *Schumpeter* (1997, S. 10, 14, 18 f.) unterscheidet zwei Typen von Gütern, Genussgüter zur Bedürfnisbefriedigung und Produktionsgüter zur Weiterverarbeitung. Wenn ein Gut das Ergebnis einer Produktion ist, wird es als Produkt bezeichnet (vgl. Schumpeter, 1997, S. 14). *Lennertz* (vgl. 2006, S. 14) definiert ein Produkt als Wirtschaftsgut, welches der Bedarfsdeckung des Nachfragers und der Existenzsicherung des Anbieters dient. Weiterhin unterscheidet er folgende Produktarten je nach Merkmal (vgl. Lennertz, 2006, S. 14):

* Substanz: materiell (z. B. Buch) und immateriell (z. B. Software)
* Verwendungshäufigkeit: Verbrauch und Gebrauch
* Nachfrager: Endverbraucher (Konsum) und Unternehmen (Investition)

Die gezielte Verwaltung von Produkten erfolgt durch das Produkt-Management (vgl. Pichler, 2014, S. 2). Es existiert keine einheitliche Definition, aber im Allgemeinen ist es verantwortlich für die Planung, Entwicklung sowie Pflege von Produkten (vgl. Lennertz, 2006, S. 7 ff.). Dabei steht das größtmögliche Wohle von Nachfrager und Anbieter im Mittelpunkt (vgl. Lennertz, 2006, S. 10). Es erfüllt die Rolle als Schnittstellenkoordinator zwischen den externen Kunden und den unternehmensinternen Bereichen wie Vertrieb, Produktion und Entwicklung (vgl. Kairies, 2004, S. 11, 15, 27; Aumayr, 2009, S. 18, 44 ff.). Daraus leiten sich die folgenden Aufgaben des Produkt-Managements ab (vgl. Kairies, 2004, S. 17, 26):

* Analyse von Markt- und Wettbewerbsinformationen
* Produktinnovation (Definition und Realisierung)
* Produktplanung und –controlling
* Produktpflege
* Vertriebsunterstützung

Ein Produkt-Manager agiert demnach als Unternehmer im Unternehmen (vgl. Kairies, 2004, S. 16 f.; Pichler, 2014, S. 6), wodurch der Produkterfolg auch bei unternehmensinterner Konkurrenz, gesteigert wird (vgl. Lennertz, 2006, S. 10). Er sorgt für einen verbesserten Informationsfluss sowie die optimale Planung, Koordination und Überwachung aller produktbezogenen Maßnahmen (vgl. Kairies, 2004, S. 18; Aumayr, 2009, S. 13). Der Produkt-Manager ist zentraler Ansprechpartner für alle Belange im Zusammenhang mit dem Produkt und ist daher auch verantwortlich für dessen Erfolg (vgl. Kairies, 2004, S. 18; Aumayr, 2009, S. 32). Von zentraler Bedeutung ist dafür eine durch ihn gesteuerte Produktplanung, da sonst Innovationsblockaden unvermeidlich sind (vgl. Kairies, 2004, S. 14). Auslöser für eine neue Produktplanung können Probleme (u. a. Reklamationen, neue Anforderungen, Krisen) oder Opportunitäten (z. B. neue Technologien oder Märkte) sein, wobei meist eine Kombination dieser Faktoren zu einer neuen Produktplanung führt (vgl. Lennertz, 2006, S. 52 f.). *Aumayr* (vgl. 2009, S. 325)nennt außerdem noch die Überalterung der Produktstruktur und die Änderung der Unternehmensstrategie sowie die strategische Lückenplanung als Auslöser. Die Gestaltung des Produktprogramms, die Summe aller Produkte, erfolgt über die Produktpolitik aus dem Marketing-Mix, welche die folgenden Gestaltungsmöglichkeiten bietet (vgl. Lennertz, 2006, S. 22 f.):

* Produktinnovation
* Produktmodifikation
* Produktelimination

Diese Gestaltungsmöglichkeiten sind abhängig vom jeweiligen Stand eines Produkts im Lebenszyklus. Der Produktlebenszyklus und seine Phasen werden im nachfolgenden Kapitel beschrieben.

## Der Produktlebenszyklus

Der Produktlebenszyklus beschreibt alle Phasen in der Entwicklung und Vermarktung eines Produkts von der Definition bis zur Absetzung (vgl. Kairies, 2004, S. 17). Die Gesamtverantwortung für den Lebenszyklus liegt beim Produkt-Management (vgl. Pichler, 2014, S. 2). Der Begriff selbst ist inhaltlich nicht korrekt, da die einzelnen Phasen nicht zyklisch aufeinanderfolgen (vgl. Lennertz, 2006, S. 19). Aufgrund der allgemeinen Bekanntheit und da im Falle einer Produktmodifikation durchaus ein Zyklus entstehen kann, wird er trotzdem in dieser Ausarbeitung verwendet. Die Dauer des Produktlebens und der Verlauf sind stark abhängig vom Produkttyp und der Branche (vgl. Aumayr, 2009, S. 294; Kairies, 2004, S. 6). So sind im Maschinenbau mittlere Lebensdauern von ca. 8 Jahren, in der Computertechnik dagegen nur von ca. 1 bis 2 Jahren typisch (vgl. Kairies, 2004, S. 6). *Kairies* (vgl. 2004, S. 60 f.) beschreibt unterschiedliche charakteristische Verläufe von Produktlebenszyklen in Abhängigkeit vom erwirtschafteten Umsatz. Neben der Normalform unterscheidet er den Flop mit kurzer Lebenszeit aufgrund eines schnellen Wachstums und Rückgangs des Umsatzes (vgl. Kairies, 2004, S. 60). Weiterhin nennt er ein langsam sterbendes Produkt, dessen Rückgang nicht entgegengewirkt wird und ein Produkt welches nach einer Modifikation erneut Wachstum generiert (vgl. Kairies, 2004, S. 61). Abschließend zeigt *Kairies* (vgl. 2004, S. 61) den Kurvenverlauf eines erfolgreichen Produkts mit langer Lebenszeit, welches einen schnellen Markteintritt gefolgt von starken Umsatzwachstum aufzeigt (vgl. 2004, S. 61). Zur Verdeutlichung der unterschiedlichen Verläufe wurden die genannten charakteristischen Produktlebenszyklen in Abbildung 2.2 in Bezug zueinander dargestellt.



Abbildung .: Charakteristische Produktlebenszyklen

Quelle: in Anlehnung an (Kairies 2004, S. 60 f.)

*Aumayr* (vgl. 2009, S. 295 f.) beschreibt einen direkten Bezug zwischen dem Produktlebenszyklusverlauf und den unterschiedlichen Elementen der Portfolioanalyse. So durchläuft ein erfolgreiches Produkt das Portfolio von Question Marks zu Stars über Cash Cows bis hin zu Dogs, wohingegen ein Flop direkt vom Fragezeichen zum Armen Hund übergeht (vgl. Aumayr, 2009, S. 295 f.). Die vorgestellten Verlaufsdarstellungen in Abbildung 2.2 zeigen, dass ein Produktleben sich in unterschiedliche charakteristische Phasen je nach Umsatzveränderung unterteilen lässt. *Kairies* (vgl. 2004, S. 60) unterscheidetdie folgenden Phasen im Produktlebenszyklus:

1. Einführungsphase
2. Wachstumsphase
3. Reifephase
4. Marktsättigungsphase
5. Marktrückgangsphase
6. Elimination (Outphasing)

In der Literatur werden aber auch andere Phasen und Bezeichnungen beschrieben. Diese werden nun kurz dargestellt und anschließend als Basis für die weitere Ausarbeitung zu einem Produktlebenszyklus zusammengefasst. Die Einteilung der Phasen sieht bei *Lennertz* (vgl. 2006, S. 19 ff.) wie folgt aus:

1. Entstehungsphase (pränatal)
   1. Produktplanung
   2. Produktentwicklung
   3. Produktfertigung
2. Lebensphase (vital)
   1. Einführung
   2. Wachstum
   3. Reife
   4. Rückgang (ggf. mit Verlängerung)
3. Entsorgungsphase (postmortal)
   1. Recycling
   2. Downcycling
   3. Abfall
   4. Emissionen

*Aumayr* (vgl. 2009, S. 33, 293, 322) unterteilt den Produktlebenszyklus dagegen in folgende Phasen:

1. Innovation
2. Einführung
3. Wachstum
4. Reife
5. Sättigung
6. Degeneration

Beim Vergleich der einzelnen Bezeichnungen und Beschreibungen sind viele Ähnlichkeiten auffallend. So finden sich 2 Phasen, nämlich Wachstum und Reife, wortwörtlich in allen 3 Einteilungen. Auch die Einführungsphase wird in allen 3 Modellen genannt, jedoch bei *Lennertz* (vgl. 2006, S. 19 ff.) und *Aumayr* (vgl. 2009, S. 33, 293, 322) erst als Folgephase zur Entstehung bzw. zur Innovation. Zu beachten ist dabei, dass *Kairies* (vgl. 2004, S. 7) die Produktentwicklung nicht als expliziten Teil des Lebenszyklus erwähnt, sie aber dennoch zeitlich vor der Einführungsphase ansetzt. Da der Begriff Entstehung eher einen passiven Prozess beschreibt und Entwicklung bzw. Innovation für die notwendige aktive unternehmerische Tätigkeit stehen (vgl. Kapitel 2.1), wird der Begriff Entwicklung für die erste Phase übernommen. Der generellen Reifephase schließt sich in zwei Modellen eine Sättigungsphase an, bei *Lennertz* (vgl. 2006, S. 19 ff.) hingegen folgt direkt die Rückgangsphase. Diese Rückgangsphase schließt sich bei *Kairies* (vgl. 2004, S. 60) der genannten Sättigung an. *Aumayr* (vgl. 2009, S. 33, 293, 322) bezeichnet diesen Rückgang synonym als Degeneration. Da die Sättigung den Übergangspunkt zwischen der Reifephase und der Wachstumsphase darstellt, erfolgt keine explizite Betrachtung als eigenständige Sättigungsphase (vgl. Aumayr, 2009, S. 295; Lennertz, 2006, S. 20). Der bereits genannten Rückgangsphase folgt bei der Beschreibung von *Kairies* (vgl. 2004, S. 60) die Elimination und bei *Lennertz* (vgl. 2006, S. 19 ff.) die Entsorgungsphase. Da die Elimination das abschließende Ergebnis der Entsorgungsphase ist und der Rückgang nur ihr Auslöser, werden diese drei unterschiedlichen Begriffe zusammengefasst und als Phase der Entsorgung übernommen. Das Ergebnis der soeben geschilderten Herleitung und somit der für die weitere Bearbeitung gültige Produktlebenszyklus, kann der nachfolgenden Tabelle (Tabelle 2.1) entnommen werden:

|  |  |  |  |
| --- | --- | --- | --- |
| *Kairies* | *Lennertz* | *Aumayr* | Produktlebenszyklus |
| (Entwicklung) | Entstehungsphase | Innovation | Entwicklung |
| Einführungsphase | Einführung | Einführung | Einführung |
| Wachstumsphase | Wachstum | Wachstum | Wachstum |
| Reifephase | Reife | Reife | Reife |
| Marktsättigungsphase | Rückgang | Sättigung |
| Marktrückgangsphase | Degeneration | Entsorgung |
| Elimination | Entsorgungsphase |

Tabelle .: Zuordnung der verschiedenen Produktlebenszyklusphasen

Quelle: eigene Tabelle

Die Vorstellung der konkreten Produktlebenszyklusphasen erfolgt in Kapitel 3. Eine besondere Bedeutung hat die Einführungsphase, da hier der Markteintritt erfolgt und somit die Pay-off-Zeit beginnt (vgl. Kapitel 2.1). Die Steuerung der Einführung erfolgt über das sogenannte Release-Management, welches im folgenden Kapitel erläutert wird.

## Produktveröffentlichung durch Standards im Release-Management

Um neue Produkte oder Änderungen an Produkten zu veröffentlichen bedingt es einer klaren Abwicklung über einen Prozess (vgl. Aumayr, 2009, S. 323). Prinzipiell könnte für jedes Produkt ein eigener Prozess gemäß den individuellen Anforderungen entworfen werden. Dies würde jedoch zu einer hohen Heterogenität in der Organisation führen und Mehrkosten verursachen, da die gleichen Dinge mehrfach erarbeitet werden würden. Andererseits könnte für jedes Produkt der gleiche Prozess (ein Standard) angewendet werden, der die individuellen Erfordernisse ignoriert und daher möglicherweise zu Verzögerungen und Mehrkosten führt, da unnötige Schritte durchgeführt werden. Besser ist es daher einen Kompromiss zwischen Individualität und Allgemeingültigkeit zu finden. Diesem Kompromiss kann durch die Etablierung von einigen wenigen Typen von Standardprozessen Rechnung getragen werden. Ein Standardtyp ist demnach im Rahmen dieser Ausarbeitung eine Prozesskonfiguration für eine bestimmte Gruppe von Produkten. Je nach Substanz eines Produkts (vgl. Kapitel 2.2) spielt die IT im Release-Management eine größere (immateriell) oder kleinere (materiell) Rolle. Im Fokus dieser Ausarbeitung stehen aufgrund der steigenden Bedeutung der IT für alle Produkttypen (vgl. AXELOS, 2011, S. 63; Baumann, 2014, S. 8; Baumöl, 2007, S. 71 f.) und der nur generisch beschriebenen Release-Management-Prozessstandards (vgl. AXELOS, 2011, S. 37) ausschließlich die für Software relevanten Prozesse. Der Änderungsprozess für Software erfolgt über die Veröffentlichung von einzelnen Releases, welche durch Projekte realisiert werden (vgl. Pichler, 2014, S. 5). Der Zusammenhang zwischen Produkt und Release wird durch die Abbildung 2.2 verdeutlicht:



Abbildung .: Zusammenhang Produkt zu Release

Quelle: in Anlehnung an (Pichler 2014, S. 5)

*Pichler* (vgl. 2014, S. 65, 93) beschreibt ein Release als ein Produktinkrement mit definierten Funktionsumfang, welches an den Kunden bzw. Anwender ausgeliefert wird. Gemäß *ITIL* (vgl. AXELOS, 2011, S. 325) ist ein Release, eine Änderung oder die Kombination mehrerer Änderungen an einem IT-Service, welche gemeinsam erstellt, getestet und installiert werden. Auf den ersten Blick scheinen diese beiden Definitionen nicht zusammen zu passen, da sich die Erste auf ein Produkt und die Zweite auf einen IT-Service bezieht. Ein IT-Service dient laut *ITIL* (vgl. AXELOS, 2011, S. 328) der Befriedigung von Kundenbedürfnissen. Diese Definition deckt sich mit der eines Produkts (vgl. Kapitel 2.2). Die unterschiedlichen Begriffe entstehen durch die verschiedene Perspektiven der Autoren. Während *Pichler* aus Sicht des Produkt-Managements arbeitet, bezieht sich *ITIL* auf die Sicht des IT-Service-Managements. Als IT-Service-Management bezeichnet *ITIL* (vgl. AXELOS, 2011, S. 18) die Bereitstellung und Verwaltung von qualitativen IT-Services durch IT-Service-Anbieter. Diese beiden Perspektiven zeigen, dass an der Entwicklung und Bereitstellung von Software mehrere unterschiedliche Bereiche beteiligt sind, welche in vielen Unternehmen strikt voneinander getrennt sind (vgl. Baumann, 2014, S. 69). Das Produkt-Management, auch Fachbereich oder Business genannt, hat als Ziel die stetige Innovation bei einem hohen Return-on-Investment (ROI) (vgl. Baumann, 2014, S. 69 f.). Die Entwicklung, als Teil der IT, setzt die Anforderungen des Produkt-Managements mittels Programmierung um und verfolgt dabei technische Innovationen, moderne Architekturen und häufige Änderungen (vgl. Baumann, 2014, S. 69). Der Betrieb, als zweiter Teil der IT, ist für die Bereitstellung der Infrastruktur und die Stabilität (vor allem) der Produktionsumgebung verantwortlich (vgl. Baumann, 2014, S. 69). Während Produkt-Management und Entwicklung demnach Veränderungen benötigen, strebt der Betrieb nach wenig Veränderung. Wie bereits in der Einleitung angedeutet, gibt es aber auch Ansätze diese Trennung aufzuheben. So ist beim DevOps-Ansatz ein Team Gesamtverantwortlich für das Produkt und alle notwendigen Aktivitäten (vgl. Baumann, 2014, S. 71 f.). Aber auch diese Teams brauchen abgestimmte Prozesse für die Zusammenarbeit. Der meist genutzte Ansatz im IT-Service-Management-Umfeld (Betrieb) ist die bereits genannte Best Practice Sammlung Information Technology Infrastructure Library (ITIL) (vgl. AXELOS, 2011, S. VIII). Sie dient als Hilfestellung bei der Etablierung von IT-Service-Management-Prozessen und lässt sich den individuellen Bedürfnissen anpassen (vgl. AXELOS, 2011, S. 3). Im Unterschied zur ISO/IEC 20000 Norm, welche einen formellen und universellen Standard beschreibt, ist ITIL eine Wissenssammlung, die beschreibt wie der ISO/IEC-Standard erreicht werden kann (vgl. AXELOS, 2011, S. 3). In ITIL wird ein Service Lebenszyklus mit fünf Stadien dargestellt (vgl. AXELOS, 2011, S. 3):

1. Service Strategy
2. Service Design
3. Service Transition
4. Service Operation
5. Continual Service Improvement

Im Zentrum steht die Service Strategy, welche die Regeln und Prinzipien für den kompletten Service-Lebenszyklus sowie das Service-Portfolio beschreibt (vgl. AXELOS, 2011, S. 32). Innerhalb des Stadiums des Service Design erfolgt die Zusammenfassung aller notwendigen Informationen für die Überführung und den Betrieb eines Service innerhalb des sogenannten Service Design Packages (vgl. AXELOS, 2011, S. 32). Die Service Transition stellt sicher, dass alle Erwartungen der Stakeholder eines Service bei Änderungen berücksichtigt werden und dabei das Risiko sowie die Beeinflussung dessen minimal sind (vgl. AXELOS, 2011, S. IX). Service Operation stellt den definierten Service letztendlich bereit (vgl. AXELOS, 2011, S. 32). Jederzeit präsent ist das Continual Service Improvement, welches die kontinuierliche Verbesserung der Organisation sicherstellt (vgl. AXELOS, 2011, S. 32). Jedes der genannten Stadien hat individuelle Prinzipien, Prozesse, Erfolgsfaktoren und Risiken (vgl. AXELOS, 2011, S. 3). Das Release-Management wird innerhalb von ITIL als Release and Deployment Management bezeichnet und ist ein Teilaspekt der Service Transition. Eine Definition bzw. Beschreibung des Begriffs ist in Kapitel 4.4 zu finden. Die Ziele der Service Transition lauten u. a. wie folgt (vgl. AXELOS, 2011, S. 4):

* effizienter und effektiver Umgang mit Änderungen („Changes“)
* Risiken gezielt verwalten und steuern
* erfolgreiche Bereitstellung von Releases
* Sicherstellung des Nutzens eines Service

Innerhalb der ITIL Service Transition Prinzipien werden verschiedene Richtlinien genannt, um diese Ziel zu erreichen (vgl. AXELOS, 2011, S. 37). Diese Richtlinien gelten unabhängig von der jeweiligen Organisation und müssen aufgrund dieses generischen Ansatzes an die vorherrschenden Bedingungen hinsichtlich beispielsweise Kultur und Größe angepasst werden (vgl. AXELOS, 2011, S. 37). Damit diese Richtlinien greifen, ist die Unterstützung durch die Unternehmensführung bei ihrer Durchsetzung entscheidend (vgl. AXELOS, 2011, S. 37). Die Richtlinien umfassen die folgenden Punkte (vgl. AXELOS, 2011, S. 37):

* Implementierung einer eigenen Service Transition Richtlinie
* alle Changes müssen die Service Transition Prozesse durchlaufen
* allgemeine Standards und Rahmenwerk sollen übernommen werden
* Wiederverwendung von bestehenden Prozessen
* Ausrichtung der Pläne der Service Transition an die Geschäftsbedürfnisse
* Stakeholder-Management betreiben
* ein Kontrollsystem soll aufgesetzt werden
* Systeme für Wissens- und Entscheidungs-Management bereitstellen
* Release-Pakete nutzen
* vorausschauend Änderungen einplanen
* proaktive Ressourcenverwaltung
* zeitige Integration der Service Transition im Lebenszyklus sicherstellen
* Qualitätssicherung für Services durchführen
* proaktive Qualitätsverbesserungen durchführen

Bei der eigenen Richtlinie zur Service Transition ist darauf zu achten, dass die Veröffentlichungstermine der Releases frühzeitig mit allen Stakeholdern abgestimmt werden (vgl. AXELOS, 2011, S. 37). Im Rahmen der Change-Abstimmung ist über die Bestätigung durch die Stakeholder weiterhin sicherzustellen, dass der veränderte Service den definierten Anforderungen entspricht (vgl. AXELOS, 2011, S. 40). Während und nach der Veröffentlichung soll der veränderte Service für einen definierten Zeitraum überwacht werden, um sicherzustellen, dass die Änderung wie geplant verläuft (vgl. AXELOS, 2011, S. 39). Wiederkehrende und fehleranfällige Aktivitäten, wie z. B. der Bau, der Test und die Installation der Software, sollten aus Gründen der Effektivität und der Effizienz automatisiert werden (vgl. AXELOS, 2011, S. 43). Bei der Durchführung von Qualitätssicherungsaktivitäten ist sicherzustellen, dass Testumgebungen der Wirkumgebung so ähnlich wie möglich sind und die Tester unabhängig vom Entwickler des Service sind (vgl. AXELOS, 2011, S. 44).

Neben den Richtlinien innerhalb der ITIL Service Transition werden auch die folgenden notwendigen Prozesse beschrieben (vgl. AXELOS, 2011, S. 5):

* Transition Planning and Support
* Change Management
* Service Asset and Configuration Management
* Release and Deployment Management
* Service Testing and Validation
* Change Evaluation
* Knowledge Management

Der Zusammenhang zwischen einem konkreten Release und den soeben beschrieben Service Lebenszyklusphasen sowie den Service Transition Prozessen wird in Abbildung 2.3 verdeutlicht.



Abbildung .: Überblick ITIL

Quelle: in Anlehnung an (AXELOS 2011, S. 5)

Neben ITIL werden auch im IT-Referenzmodell COBIT (Controlled Objectivies for Information and Related Technology), als Teil der Management-Domäne „Build, cquire and Implement“, Prozesse für die Durchführung von Änderungen beschrieben (vgl. Beims & Ziegenbein, 2015, S. 264). Der COBIT-Prozess „Management von Änderungen“ entspricht hierbei dem ITIL-Prozess Change Management (vgl. Beims & Ziegenbein, 2015, S. 267). Das Release and Deployment Management aus ITIL findet sich in COBIT im Prozess „Managen der Abnahme und Überführung von Änderungen“ wieder (vgl. Beims & Ziegenbein, 2015, S. 267). In der Norm ISO/IEC 20000 werden zu diesen beiden ITIL-Prozessen ebenfalls Vorgaben und Implementierungshinweise gegeben (vgl. ISO/IEC, ISO/IEC 20000-1, 2011, S. 23 ff.; ISO/IEC, ISO/IEC 20000-2, 2012, S. 69 ff.). Eine detaillierte Beschreibung der Prozesse erfolgt in Kapitel 3.2.2.

Nach diesen Ausführungen ist ein genereller Überblick in die Themenwelt sowie über die Zusammenhänge zwischen Innovation, Produkten und Release-Management gegeben. Im folgenden Kapitel erfolgt die Analyse der vorliegenden Risiken innerhalb der jeweiligen Produktlebenszyklusphasen.

# Risiken im Produktlebenszyklus als Basis der Standardtypen

Die geplanten Standardtypen für das Release-Management sollen auf einer Kombination von Risiken in den Produktlebenszyklusphasen und von Prozessbestimmungen des IT-Release-Managements basieren (vgl. Kapitel 1). Aufbauend auf den Ausführungen in Kapitel 2.3 erfolgt in diesem Kapitel die Untersuchung der einzelnen Phasen des Produktlebens auf spezifische Besonderheiten, um konkrete Risiken zu ermitteln. Charakteristisch für die einzelnen Phasen sind unterschiedliche Verläufe des Umsatzes (vgl. Kairies, 2004, S. 60 f.) und des Deckungsbeitrags (vgl. Aumayr, 2009, S. 322). Daher erfordert jede einzelne Phase ihre eigenen Aktivitäten in Bezug auf das Produkt und den Markt (vgl. Kairies, 2004, S. 61) und unterliegt damit unterschiedlichen Risiken (vgl. Aumayr, 2009, S. 323). Abgeschlossen wird das Kapitel mit einer Zuordnung der Phasen zu Risikotypen und einer möglichen Vereinfachung anhand von aufgefallenen Mustern. Diese Risikotypen bilden die Voraussetzung für die Untersuchung der Release-Management-Prozesse im nächsten Hauptkapitel.

## Entwicklung

Zu Beginn der Entwicklung eines Produkts erfolgt die Produktplanung, welche die Marktanalyse, die Wettbewerbsanalyse, die Ideenfindung und Produktbeschreibung umfasst (vgl. Lennertz, 2006, S. 19). Weiterhin werden die möglichen Kosten betrachtet und Studien bzgl. der einzusetzenden Komponenten und Verfahren durchgeführt (vgl. Lennertz, 2006, S. 19). Innerhalb eines Entwicklungsprojektes werden daraufhin die ersten Prototypen entwickelt und getestet (vgl. Lennertz, 2006, S. 19). Dabei ist eine sequentielle Arbeitsweise unbedingt zu vermeiden, da sie zu späten sowie teuren Änderungsbedarfen, Qualitätseinschränkungen und zur Demotivation der beteiligten Mitarbeiter führt (vgl. Kairies, 2004, S. 125). Stattdessen sollte die Entwicklung von einem dedizierten Team parallel und projektbezogen erfolgen (vgl. Kairies, 2004, S. 123 f.). Wichtig für den zukünftigen Erfolg ist eine hohe Produktqualität, welche die Grundlage für Verkaufbarkeit und Kundenzufriedenheit ist und durch eine prüfgerechte Produktentwicklung erreicht wird (vgl. Kairies, 2004, S. 9, 127, 131). Daher findet vor der Übergabe an die Serienfertigung der Test der Prototypen durch potenzielle Kunden statt (vgl. Lennertz, 2006, S. 19). Parallel zur Fertigung erfolgt die Vorbereitung für die Markteinführung (vgl. Lennertz, 2006, S. 19; Aumayr, 2009, S. 331 f.). Dies bedeutet vor allem umfangreiche Kommunikationsmaßnahmen, nach außen und innen, damit auch innerhalb des Unternehmens die Akzeptanz für das Produkt sichergestellt wird (vgl. Aumayr, 2009, S. 323). *Aumayr* (vgl. 2009, S. 323) bezeichnet diesen Ablauf als Innovationsprozess. Entscheidend ist dabei die Flexibilität, um auf unerwartete Entwicklungen reagieren zu können, da die Markt- und Wettbewerbssituation, vor allem bei Innovationen, mit einer hohen Unsicherheit verbunden ist (vgl. Aumayr, 2009, S. 323; Pichler, 2014, S. 39; Grawe, 2016, S. 133). Diese Unsicherheit entsteht vor allem durch einen Wissensmangel, welcher bei einer Innovation aber unvermeidlich ist (vgl. Pichler, 2014, S. 64). Aufgrund der empirischen Herangehensweise und der Ermöglichung des schnellen Kundenfeedbacks eignet sich daher Scrum als Projektvorgehen für IT-Produkte bzw. -Produktanteile besonders gut, da es effektiv Risiken reduziert (vgl. Pichler, 2014, S. S. XIX, 65). Besonders der Scrum-Aspekt des minimalen Produkts führt in dieser Situation zum größten Erfolg, da es aufgrund der Fokussierung auf Kernfunktionen günstiger und schneller erstellt sowie angepasst werden kann (vgl. Pichler, 2014, S. 39). Um den richtigen Zeitpunkt für die Veröffentlichung festzuhalten, eignet sich eine Release-Planung, welche außerdem noch ein gewisses Budgetziel und einen Funktionsumfang definiert (vgl. Pichler, 2014, S. 88). Entscheidend ist dabei, dass weder die Zeit noch die Qualität flexibel sind, sondern nur der Funktionsumfang (vgl. Pichler, 2014, S. 88). Denn ohne eine hohe Qualität ist ein schnelles und regelmäßiges ausliefern von Releases nicht effektiv (vgl. Pichler, 2014, S. 93). Die Alternative zum schnellen und regelmäßigen Ausliefern wäre ein sogenanntes „big bang“-Release, d. h. viel Funktionalität auf einmal, welches hohe Kosten sowie eine lange Projektlaufzeit und damit ein hohes Investitionsrisiko bedeutet (vgl. Pichler, 2014, S. 107).

Die Ausführungen aus der Literatur zeigen, dass die Entwicklungsphase von mehreren unterschiedlichen Risiken geprägt ist. Im Vordergrund stehen die Risiken einer zu langen Time to Market und das Verpassen des Window of Opportunity (vgl. Kapitel 2.1) durch fehlende Flexibilität, Schnelligkeit und daher lange Projektlaufzeiten. Dies birgt ein hohes Investitionsrisiko und zieht aufwendige Änderungsarbeiten nach sich. Die zeitliche Dimension allein ist aber nicht entscheidend, denn bei einer geringen Qualität fehlt die nötige Verkaufbarkeit des Produkts und damit die Kundenzufriedenheit. Die Risiken in dieser Phase konzentrieren sich demnach auf zeitliche und qualitative Aspekte. Die Kosten spielen hier eher eine untergeordnete Rolle (vgl. Roock & Wolf, 2016, S. 1249).

## Einführung

In dieser Lebenszyklusphase erfolgt der Markteintritt des Produkts und somit der eigentliche Lebenszyklus (vgl. Aumayr, 2009, S. 294). Ziele der Einführungsphase sind vor allem die Bekanntmachung des Produkts auf dem Markt, die Gewinnung von Referenzkunden sowie die Beseitigung von Fehlern (vgl. Kairies, 2004, S. 62; Aumayr, 2009, S. 304 f.). Die dadurch entstehenden hohen Einführungskosten und der aufgrund der geringen Bekanntheit geringe Absatz führen zu einem geringen oder sogar negativen Deckungsbeitrag (vgl. Aumayr, 2009, S. 294, 305). Um den Bekanntheitsgrad zu erhöhen ist die Kundenzielgruppe die der Innovatoren, welche aufgrund eines hohen Betreuungsbedarfs durch den Vertrieb jedoch kostenintensiv ist (vgl. Aumayr, 2009, S. 304). Anhand deren Adoption entscheidet sich die zukünftige Entwicklung des Produkts auf dem Markt, da die große Mehrheit diesen Referenzen folgt und das Produkt somit entweder angenommen oder ein Flop wird (vgl. Aumayr, 2009, S. 294, 304). Um das Feedback der Kunden so zeitig wie möglich zu erhalten und entsprechend reagieren zu können, ist analog der der Entwicklungsphase eine hohe Flexibilität nötig (vgl. Pichler, 2014, S. 44, 51). Besonders hervorzuheben ist hierbei die Rolle der Prozesse, Dienste und Werkzeuge, die für die schnelle Freigabe von neuen Versionen (Releases) notwendig sind (vgl. Pichler, 2014, S. 44). Positiv wirkt sich in dieser Phase der typischerweise geringe Konkurrenzdruck aus (vgl. Aumayr, 2009, S. 305). Dies ist allerdings auch abhängig von anderen Faktoren, wie z. B. dem Marktlebenszyklus, welcher den Lebenszyklus der gesamten Produktgruppe auf dem Markt darstellt und demnach vom Stadium des einzelnen Produkts abweichen kann (vgl. Aumayr, 2009, S. 297). Mit Erreichen des BEP (vgl. Kapitel 2.1), d. h. wenn die Erlöse die Kosten übersteigen, gilt die Produkteinführungsphase für beendet (vgl. Lennertz, 2006, S. 20). Sollte ein Produkt den BEP nie erreichen, gilt es als Flop (vgl. Kapitel 2.3).

Die Betrachtung dieser Produktlebenszyklusphase zeigt viele Ähnlichkeiten zur Entwicklungsphase. Auch hier liegt das größte Risiko auf zeitlichen Aspekten. Um Referenzkunden zu gewinnen, ist es nötig entdeckte Fehler schnell zu beseitigen und zeitnah auf das Feedback der innovationsbezogenen Kundengruppe mittels Produktverbesserungen einzugehen. Die geforderte zeitnahe Fehlerbeseitigung unterstreicht außerdem den weiterhin gültigen Risikoaspekt der mangelhaften Qualität.

## Wachstum

Für die Wachstumsphase ist der schnelle Anstieg des Absatzes und des Deckungsbeitrags aufgrund der Expansion am Markt charakteristisch (vgl. Aumayr, 2009, S. 294). Dies liegt an den geringeren Aufbaukosten und der Gewinnung der sogenannten frühen Mehrheit als Kundengruppe (vgl. Aumayr, 2009, S. 305). Unter Aufbaukosten werden die Aufwände für die Ergänzung und Modifikation des Produkts sowie die Veröffentlichungen von unterschiedlichen Produktoptionen zusammengefasst (vgl. Kairies, 2004, S. 62; Aumayr, 2009, S. 305). Durch die höheren Absatzzahlen lassen sich beim Einkauf und der Produktion Skaleneffekte realisieren, welche ebenfalls die Kosten senken (vgl. Kairies, 2004, S. 62). Sollte der Marktlebenszyklus gleichzeitig im Wachstum sein, wächst jedoch der Wettbewerbsdruck entsprechend (vgl. Aumayr, 2009, S. 305). Das Ende der Phase ist erreicht, sobald das Umsatzwachstum am Höhepunkt ist, spätestens dann sollten die Voraussetzungen für ein Nachfolgeprodukt geschaffen sein (vgl. Kairies, 2004, S. 62; Lennertz, 2006, S. 20).

Die Phasenbeschreibung des Wachstums zeigt, dass die Risiken bezüglich der zeitlichen Dimension im Gegensatz zu den vorherigen Phasen nicht mehr im Vordergrund stehen. Weiterhin relevant ist das Qualitätsrisiko, da es entscheidend für den Ausbau der Marktanteile ist. Dazu steigt die Bedeutung des Kostenrisikos, da im Fokus die Kostensenkung steht. Beide Risiken sind von entscheidender Bedeutung für die Durchsetzung gegen den Wettbewerb.

## Reife

Die Reife zeichnet sich durch die zunehmende Marktsättigung sowie durch stagnierende und ggf. rückläufige Wachstumsraten und Deckungsbeiträge aus (vgl. Aumayr, 2009, S. 295; Kairies, 2004, S. 62; Lennertz, 2006, S. 20). Nach und nach kann zwar die späte Mehrheit als Kunden gewonnen werden, doch das dauert länger als bei den vorherigen Kundengruppen (vgl. Aumayr, 2009, S. 305). Dafür fallen die Erhaltungskosten geringer aus, da der Änderungsbedarf aufgrund der eher vorhersagbaren Marktentwicklung niedriger ist (vgl. Aumayr, 2009, S. 305; Pichler, 2014, S. 39). Im Fokus der Phase stehen als Ziele die Sicherung der Marktanteile sowie die Gewinnoptimierung (vgl. Aumayr, 2009, S. 305). Für die Marktanteilsicherung ist die kommunikationspolitische Abgrenzung vom Wettbewerb durch Alleinstellungsmerkmale entscheidend, auch wenn der Konkurrenzdruck nicht mehr weiter steigt (vgl. Kairies, 2004, S. 63; Aumayr, 2009, S. 305). Aufgrund des rückläufigen Wachstums, sollte mit Erreichen der Sättigung das Nachfolgeprodukt fertig entwickelt sein und ersten Kunden verkündet werden (vgl. Kairies, 2004, S. 63).

Die Risikosituation in der Reifephase ist der der Wachstumsphase sehr ähnlich. Zeitliche Vorteile stehen nicht im Vordergrund, sondern die Gewinnoptimierung und die Sicherung der gewonnenen Marktanteile. Da das Wachstum rückläufig ist, steht die Kostensenkung im Vordergrund und damit wird das Kostenrisiko dominant. Zudem spielt die Qualität eine wichtige Rolle bei der Sicherung der Marktanteile, da sie auch als Alleinstellungsmerkmal gesehen werden kann.

## Entsorgung

Die Entsorgungsphase ist die letzte Phase im Produktlebenszyklus (vgl. Kapitel 2.3). Sie weist typischerweise einen negativen Deckungsbeitrag aufgrund der rückläufigen Nachfrage auf (vgl. Aumayr, 2009, S. 295). Ziel ist der Abbau des Produkts und die Kostensenkung, da aufgrund der fehlenden Konkurrenzfähigkeit nur noch Nachzügler als Kunden gewonnen werden können (vgl. Aumayr, 2009, S. 305; Kairies, 2004, S. 63). Versuche der Wiederbelebung, z. B. durch Facelifting, bedeuten weitere Kosten (vgl. Lennertz, 2006, S. 20 f.) und schlagen in dieser Phase häufig fehl (vgl. Aumayr, 2009, S. 295), daher sollte das Nachfolgeprodukt in dieser Phase eingeführt werden (vgl. Kairies, 2004, S. 63). Je nach Marktlebenszyklus nimmt auch der Wettbewerb ab (vgl. Aumayr, 2009, S. 305) oder im Falle einer Innovationswelle stärker zu, bis kein Marktanteil mehr übrig bleibt (vgl. Kapitel 2.1). Letztendlich wird das bestehende Produkt eliminiert (vgl. Kairies, 2004, S. 63). Um Restbestände des Produkts absetzen zu können, besteht die Möglichkeit mit einer Kostenführerstrategie auf andere Märkte zu schwenken (vgl. Kairies, 2004, S. 63 f.).

Die Entsorgungsphase weist erneut ähnliche Risikotypen wie die beiden Phasen davor auf. Aufgrund der bevorstehenden Elimination des Produkts erfolgt keine Weiterentwicklung mehr, welche zeitlichen Risiken ausgesetzt sein könnte. Das Qualitätsrisiko liegt in dieser Phase weiterhin vor, da die bestehenden Kunden solange wie möglich gehalten werden sollen und beim Produktwechsel nicht aufgrund negativer Erfahrung, z. B. aufgrund mangelhafter Verfügbarkeit oder Performanz, den Anbieter wechseln sollen. Eine schlechte Qualität würde auch die Kosten für notwendige Anpassungen (schwerwiegende Fehler, Sicherheitslücken oder Änderung von rechtlichen Rahmenbedingungen) erhöhen. Im Zentrum der Phase steht demnach die Einsparung von Kosten unter Beibehaltung eines akzeptablen Qualitätsniveaus.

## Zusammenfassung der Risiken in den Phasen

Die Betrachtung der unterschiedlichen Phasen in den vorherigen Kapiteln hat die Annahme der verschiedenen Risikotypen bestätigt. Während zu Beginn des Lebenszyklus das zeitliche Risiko dominiert und das Qualitätsrisiko im gesamten Verlauf relevant ist, sind die Kosten erst nach der Etablierung des Produkts am Markt ein Risiko, da sie vorher eine untergeordnete Rolle spielen. Diese Ausführungen zeigen weiterhin, dass anhand des Produktlebenszyklus drei verschiedene Risikotypen identifiziert werden können: die Zeit, die Qualität und die Kosten. Letztendlich lassen sich die Risiken der Zeit und der Qualität natürlich auch in Form von Kosten darstellen, da z. B. die Verzögerung einer Veröffentlichung zu weniger Umsatz führt (Opportunitätskosten) und schlechte Qualität zu höheren Kosten beispielsweise aufgrund einer höheren Nachbesserungsquote führt. Da sich der hier genannte Risikotyp Kosten aber eher auf direkten Aufwand als auf Folgekosten bezieht, ist die Unterscheidung trotzdem gerechtfertigt. Zusammenfassend für das Kapitel 3 erfolgt in Tabelle 3.1 eine Zuordnung dieser Risikotypen zu den einzelnen Produktlebenszyklusphasen.

|  |  |  |  |
| --- | --- | --- | --- |
| Produktlebenszyklusphase | Risikotyp | | |
| Zeit | Qualität | Kosten |
| Entwicklung | ja | ja | nein |
| Einführung | ja | ja | nein |
| Wachstum | nein | ja | ja |
| Reife | nein | ja | ja |
| Entsorgung | nein | ja | ja |

Tabelle .: Zuordnung Risikotypen zu Produktlebenszyklusphasen

Quelle: eigene Tabelle

Die Darstellung in der Tabelle 3.1 zeigt die Ähnlichkeit verschiedener Phasen hinsichtlich der korrespondierenden Risikotypen. Dies ermöglicht für die Standardtypdefinition die Reduktion der Komplexität, da nicht für jede Produktlebenszyklusphase eine eigene Ausgestaltung der Release-Management-Prozesse erfolgen muss, sondern ein Standardtyp für mehrere Phasen mit gleichem Risikotyp angewendet werden kann. Der Risikotyp der Qualität ist in jeder Phase relevant und kann daher für die weitere Betrachtung zur Unterscheidung eliminiert werden. In Tabelle 3.2 erfolgt daher die Zuordnung der Phasen zum jeweiligen Risikotyp ohne das Qualitätsrisiko.

|  |  |  |
| --- | --- | --- |
| Produktlebenszyklusphase | Risikotyp | |
| Zeit | Kosten |
| Entwicklung | ja | nein |
| Einführung | ja | nein |
| Wachstum | nein | ja |
| Reife | nein | ja |
| Entsorgung | nein | ja |

Tabelle .: Reduzierte Zuordnung Risikotypen zu Produktlebenszyklusphasen

Quelle: eigene Tabelle

Als Basis für die Ableitung der Standardtypen in Kapitel 5 erfolgt eine kurze Zusammenfassung der Risikotypen „Zeit“ und „Kosten“.

Der **Risikotyp „Zeit“** zeichnet sich gemäß der Ausführungen zu den Produktlebenszyklusphasen Entwicklung und Einführung (vgl. Kapitel 3.1 und 3.2) durch den Bedarf einer größtmöglichen Flexibilität aus, wobei so viele wie mögliche Aktivitäten parallel und nicht sequentiell erfolgen sollten. Dies ist vor allem durch die hohe Unsicherheit aufgrund des Innovationscharakters bedingt, wofür eine empirische Vorgehensweise empfehlenswert ist (vgl. Kapitel 3.1). Um diesen Ansatz zu verfolgen ist die Erstellung eines minimalen Produkts mittels vielen kleinen regelmäßigen Releases zur schnellen Anpassung an das Feedback der Kunden notwendig (vgl. Kapitel 3.1). Wichtig sind dabei vor allem die Prozesse, Dienste und Werkzeuge, welche zur Veröffentlichung von neuen Releases herangezogen werden, da Sie den Zeitpunkt der Veröffentlichung maßgeblich beeinflussen (vgl. Kapitel 3.2).

Im Gegensatz dazu ist der **Risikotyp „Kosten“** durch die folgenden Eigenschaften anhand der anderen 3 Phasen (vgl. Kapitel 3.3 bis 3.5) zu beschreiben. Im Fokus liegt das Einsparen von Kosten, durch reduzierte Aufwände im Aufbau und die Nutzung von Skaleneffekten, wodurch der Markt weiter durchdrungen und der Gewinn optimiert werden soll (vgl. Kapitel 3.3 und 3.4). Nicht unbedingt notwendige Ausgaben oder wenig erfolgreiche Maßnahme sind im Sinne der Kostensenkung zu vermeiden (vgl. Kapitel 3.5).

Nach diesen kurzen Zusammenfassungen der Risikotypen erfolgt nun die Erarbeitung der unterschiedlichen Prozessfaktoren des Release-Managements zum Umgang mit den Risikotypen.

# Faktoren des Release-Managements zum Umgang mit Risiken

Das Release-Management wurde bereits in Kapitel 2.4 vorgestellt. Ziel dieses Kapitels ist es, die unterschiedlichen Prozesse zur Steuerung des Release-Managements hinsichtlich der Möglichkeiten zur Behandlung der beiden Risikotypen (vgl. Tabelle 3.2) aus dem vorhergehenden Hauptkapitel zu untersuchen. Hierfür werden als Basis die Prozesse der ITIL Service Transition betrachtet, da es als de-facto Standard gilt (vgl. Kapitel 2.4). Die Erkenntnisse werden an den passenden Stellen um die Aspekte aus der ISO/IEC 20000 Norm ergänzt. Das COBIT-Framework findet keine eigenständige Betrachtung, da COBIT einen Fokus auf Governance-Aspekte hat, wogegen der Fokus bei ITIL auf Prozessen liegt und es sich in den wesentlichen Punkten nicht unterscheidet (vgl. Auner, 2008, S. 98). Für die Betrachtung werden die bereits in Kapitel 2.4 genannten Prozesse hinsichtlich dem Zweck, den Zielen, dem Mehrwert sowie den Grundgedanken beschrieben. Daraufhin erfolgt eine Betrachtung der Prozesse auf Faktoren, die je nach Risikotyp unterschiedlich definiert werden müssen. Eine detaillierte Vorstellung der einzelnen Prozesse kann aufgrund des Umfangs an dieser Stelle nicht erfolgen. Dazu kann die Quellliteratur herangezogen werden. Abschließend erfolgt im Kapitel eine Zusammenfassung der Ergebnisse als Grundlage für die Ableitung der Standardtypen im nächsten Hauptkapitel.

## Transition Planning and Support

Der Prozess Transition Planning and Support (TPS) dient der übergreifenden Planung und Koordination von Änderungsvorhaben sowie der nötigen Ressourcen (vgl. AXELOS, 2011, S. 51). Ziel ist es unter anderem die Änderung innerhalb der geplanten Vorgaben hinsichtlich Kosten, Zeit und Qualität durchzuführen (vgl. AXELOS, 2011, S. 51). Eine weitere Zielstellung ist die Identifizierung, Behandlung und Steuerung von Risiken, um die Fehlerwahrscheinlichkeit bei der Umsetzung zu senken (vgl. AXELOS, 2011, S. 51). Der Mehrwert des Services liegt in der Befähigung der Organisation zur Bearbeitung einer großen Anzahl von Änderungen unter Einhaltung der Vorgaben sowie der Ausrichtung des IT-Service-Managements anhand der Bedürfnisse der Stakeholder (vgl. AXELOS, 2011, S. 51). Ergänzend zu den übergreifenden Service Transition Richtlinien werden innerhalb von TPS Release-Richtlinien für einzelne Services oder Servicegruppen vorgeschlagen, welche folgende Aspekte betreffen (vgl. AXELOS, 2011, S. 53):

* Namens- und Identifizierungskonventionen
* Rollen und Zuständigkeiten für die einzelnen Release-Phasen
* Nutzung von Medienbibliotheken
* Release-Zyklen
* Methode zur Release-Bündelung
* Automatisierungsmechanismen
* Versionierung der zusammengehörigen Konfiguration
* Kriterien und Berechtigungen für die Freigabe in spezifische Umgebungen
* Kriterien für den Abschluss der Beobachtungsphase

Unabhängig vom Risikotyp ist es notwendig gewisse **Namens- und Identifizierungskonventionen** einzuhalten, um einen eindeutigen Bezug zum Änderungsvorhaben innerhalb der Organisation sicherzustellen und Missverständnisse zu vermeiden. Dieser Sachverhalt gilt ebenfalls für die Vorschriften zur **Versionierung**. Damit nicht nur der Name und die Version eines Releases eindeutig sind, sondern auch der Ablageort, auf denen verschiedene Personen wie Entwickler, Tester und Andere zugreifen müssen, empfiehlt sich die allgemeine Nutzung von zentralen **Medienbibliotheken**. Die Verwendung der genannten Richtlinien behandelt Risiken allgemeiner Natur, aber nicht die spezifisch genannten Risikotypen aus dem vorherigen Hauptkapitel. Von zentraler Bedeutung dagegen ist die Festlegung des **Release-Zyklus** aufgrund der direkten Auswirkung auf das zeitliche Risiko, welcher auch von der ISO/IEC 20000-1:2011 vorgeschrieben wird (vgl. ISO/IEC, 2011, S. 24). Je größer der Abstand zwischen den Releases wird, desto umfangreicher wird ein einzelnes Release hinsichtlich der umgesetzten Anforderungen und desto später kann das Feedback der Endkunden eingeholten werden – das Risiko einer Fehlentwicklung steigt (vgl. Kapitel 2.1). Auch die Dauer bis eine notwendige Fehlerbehebung zur Verfügung gestellt werden kann steigt. Je nach Fehlerauswirkung kann zu dem die Kundenzufriedenheit sinken bzw. können erhöhte Prozesskosten entstehen. Der Testaufwand sowie das Risiko einen Fehler zu übersehen steigen aufgrund der größeren Komplexität ebenfalls. Der beschriebene Umfang eines Release wird durch die **Release-Bündelung** bestimmt, d. h. welche Änderungen werden gemeinsam veröffentlicht (vgl. ISO/IEC, 2012, S. 72). Kurze Release-Zyklen reduzieren die Zeit zur Umsetzung neuer Änderungen und verringern damit die Release-Bündelung, so dass ggf. jede Änderung als eigenständiges Release veröffentlich wird. Aufgrund des direkten Zusammenhangs zwischen dem Release-Zyklus und der Release-Bündelung, wird die Bündelung als Nebeneffekt angesehen und nicht weiter gesondert betrachtet. Negativ kann sich weiterhin der Aspekt der **Rollen und Zuständigkeiten** auf das zeitliche Risiko auswirken. Mit der Anzahl der beteiligten Personen an den Prüfungen und Freigaben steigt die Dauer bis zur Freigabe aufgrund von Wartezeiten, in denen niemand aktiv daran arbeitet. Typischerweise bedeutet eine Aufteilung der Verantwortung, dass eine einzelne Person weniger Aufgaben pro Service für eine größere Anzahl von Services betreuen muss, um ausgelastet zu sein. Die dadurch häufiger notwendigen Themenwechsel führen zu längeren Bearbeitungszeiten, da jedes Mal eine gewisse Einarbeitungszeit benötigt wird (vgl. Roock & Wolf, 2016, S. 1839). Verstärkt wird dieser Effekt durch eine sinkende Motivation und Leistung der Mitarbeiter, aufgrund der eintönigen und repetitiven Aufgaben (vgl. Kelly, 1982, S. 204 ff.). Die Rollentrennung kann im Sinne der Qualität aber auch sinnvoll sein, vor allem wenn die Verantwortung mit dem Release von einer Abteilung zu einer anderen wechselt. Über eine zusätzlich Prüfung, kann dann sichergestellt werden, dass die Anforderungen der übernehmenden Abteilung eingehalten werden. Daher ist der Faktor der **Kriterien und Berechtigungen zur Freigabe** direkt abhängig von den entsprechenden Rollen und ein wichtiges Mittel um Qualität sicherzustellen. Die Forcierung der Einhaltung von Qualitätskriterien, z. B. durch den zwingenden Nachweis von Testergebnissen, stellt sicher, dass kein Release unterhalb des definierten Qualitätsniveaus veröffentlicht wird. Eine Ausnahme ist die bewusste Entscheidung trotzdem das Release zu veröffentlichen, aufgrund von anderen Risiken (z. B. Zeitdruck oder drohenden Kosten). In diesem Fall führt diese Richtlinie jedoch zu Transparenz und vermeidet die blinde Risikoakzeptanz. Ein positiver Aspekt, der die Durchlaufzeit begünstigt und Risiken senkt, ist die **Automatisierung** von Tätigkeiten (vgl. Baumann, 2014, S. 6). Dieser steht eng mit dem Release-Zyklus in Verbindung, denn bei einer kurzen Durchlaufzeit und bei häufiger Wiederholung ist eine Automation unabdingbar, um die Kosten gering zu halten und das Qualitätsniveau beizubehalten. Bei wenigen Releases mit größerem Änderungsumfang und daraus resultierenden größeren Anpassungsbedarfen der Automatisierung ist diese wiederum zu überdenken. Daher wird die Automatisierung nicht als einzelner Faktor des TPS-Prozess, sondern in Abhängigkeit vom Release-Zyklus gewertet. Die **Abschlusskriterien** der Beobachtungsphase sind von verschiedenen Faktoren abhängig. Entscheidend ist jedoch der Umfang der Änderung, denn je kleiner der Umfang, desto geringer das Risiko eines Problems und desto kürzer der Beobachtungszeitraum. Daher wird dieser Faktor ebenfalls in Abhängigkeit des **Release-Zyklus** betrachtet.

Zusammenfassend sind für den TPS-Prozess die Aspekte des **Release-Zyklus** sowie der **Freigaberollen** in Abhängigkeit des Risikotyps zu definieren, welche vor allem eine Auswirkung auf die Zeit haben.

## Change Management

Der Prozess Change Management (CHM) dient der Steuerung des Lebenszyklus aller Änderungen zur minimalen Beeinflussung der produktiven Services (vgl. AXELOS, 2011, S. 61). Der Fokus liegt demnach nicht auf organisatorischen Veränderungen, sondern auf technischen Änderungen an IT-Services. Ziele sind die Optimierung der Geschäftsrisiken unter Beachtung des potentiellen Nutzens, der Kundenanforderungen und der nötigen Dokumentation (vgl. AXELOS, 2011, S. 61). Der Mehrwert liegt daher in der Sicherstellung der Verlässlichkeit und der Verfügbarkeit der Services durch den Schutz der produktiven Systeme unter Beachtung der notwendigen Änderungsanforderungen (vgl. AXELOS, 2011, S. 61). Weiterhin lassen sich durch einen gut strukturierten CHM-Prozess erhebliche Kosteneinsparungen realisieren (vgl. AXELOS, 2011, S. 61). Für den CHM-Prozess werden folgende spezifische Regelungen empfohlen (vgl. AXELOS, 2011, S. 61):

* „für jede Änderung ein Change“ - Kultur schaffen
* Ausrichtung des Change Managements an den Stakeholdern
* Messen der Mehrwerte von Changes
* Priorisierung von Changes
* Verantwortlichkeiten definieren
* Verantwortung auftrennen
* Schaffung eines zentralen Anlaufpunkts
* Zugriffsbeschränkungen einrichten
* Nachverfolgbarkeit von Changes sicherstellen
* Wartungsfenster durchsetzen
* Leistungs- und Risikobewertung aller Changes
* Messung der Effizienz und Effektivität von Changes

Die meisten der genannten Regelungen sind unabhängig von der konkreten Risikosituation eines Service. So ist die Etablierung einer **Change-Kultur** ein genereller Ansatz und in jedem Fall notwendig um den beschriebenen Mehrwert des Change Managements zu erreichen und sicherzustellen, dass überhaupt eine Risikobewertung erfolgt. Ein Change-Prozess der die **Anforderungen der Stakeholder** nicht berücksichtigt, wird am ehesten umgangen (vgl. AXELOS, 2011, S. 89). Mit der richtigen kulturellen Einstellung sollten Stakeholder akzeptieren können, dass es gewisse Prozessregelungen für Änderungen einzuhalten gilt, da sie letztendlich nur in deren Interesse gestaltet wurden. Daher ist die Einhaltung dieser Regelung ebenfalls universell gültig. Hinsichtlich der **Mehrwertbemessung** lässt sich ebenfalls keine servicebezogene Unterscheidung vornehmen, da nicht per se davon ausgegangen werden kann, dass eine Änderung sinnvoll ist, nur weil ein Service bestimmte Merkmale aufweist. Abhängig von der Mehrwertbemessung ist die **Priorität** einer Änderung. Je höher der Nutzen, desto eher sollte die Änderung herbeigeführt werden ergo desto höher sollte die Priorität eingestuft werden. Potentiell sind Änderungen von Services mit hohem Geschäftsnutzen oder strategischer Bedeutung zwar wichtiger, da aber der Mehrwert nicht pauschal für alle Änderungen definiert werden kann, ist auch die Priorität nicht pauschal hoch oder niedrig. Bei jedem Change ist zudem das entsprechende Risiko zu beachten, so dass die Durchführung von **Leistungs- und Risikobewertungen** ebenfalls jederzeit zu erfolgen hat, da auch eine kleine Änderung an einem unkritischen Service zu größeren Problemen führen kann. Sehr eng miteinander verbunden sind die Aspekte der **Verantwortungsdefinition** und der **Verantwortungsauftrennung**. Generell muss es für jeden Service eine Verantwortungsdefinition geben, damit bei Entscheidungen die Kompetenzen klar sind. Die Auftrennung der Verantwortung kann jedoch bei jedem Service anders sein, da unterschiedliche Personen in unterschiedlichen Rollen mit individuellen Fähigkeiten miteinander arbeiten. Neben der individuellen Fähigkeit der handelnden Personen, können aber auch übergreifende Regelungen, wie z. B. ein generelles vier-Augen-Prinzip oder Budgetverantwortung eine Rolle spielen (vgl. AXELOS, 2011, S. 68). Um unnötige Verzögerungen zu vermeiden, sollten aber die Verantwortlichkeiten nicht zu stark verteilt werden (vgl. Kapitel 4.1). Verzögerungen können durch Wartezeiten, durch Abstimmungsprobleme und durch unterschiedliche Zielstellungen entstehen. Demnach ist abhängig von der zeitlichen Kritikalität die Verantwortungsdefinition zu gestalten. Da dieser Aspekt aber bereits aus dem TPS-Prozess (vgl. Kapitel 4.1) entnommen wurde, findet keine gesonderte Betrachtung im Rahmen vom CHM statt. Die gleiche Argumentation gilt auch für den Faktor der **Zugriffsbeschränkung**, da diese von der jeweiligen Verantwortung der Personen abhängig sind. Prinzipiell sollten jedoch für alle Services derartige Beschränkungen festgelegt werden. Ein **zentraler Anlaufpunkt** für alle Änderungen sollte analog der Kultur eine generelle Regelung für alle Services sein. Diese Funktion wird durch das Change Advisory Board (CAB) wahrgenommen (vgl. AXELOS, 2011, S. 80). Es prüft alle Änderungen und gibt diese frei bzw. lehnt sie mit Hinweisen zur Verbesserung ab (vgl. AXELOS, 2011, S. 67, 80). Diese Funktion ist gerade dann wichtig, wenn verschiedene Änderungsvorhaben in einer Dimension, z. B. hinsichtlich der benötigten personellen Ressourcen im Konflikt stehen oder andere Services von der Änderung betroffen sind (vgl. AXELOS, 2011, S. 64). Um den verschiedenen Priorisierung gerecht zu werden, werden mehrere Change-Typen unterschieden (vgl. AXELOS, 2011, S. 65). Emergency Changes sind Änderungen, die so schnell wie möglich durchgeführt werden müssen, um einen schweren Fehler zu beheben oder um ein Sicherheitsupdate einzuspielen (vgl. AXELOS, 2011, S. 65). Standard Changes dagegen sind Änderungen mit geringem Risiko, welche oft durchgeführt werden und als Routinetätigkeit angesehen werden können und demnach bevorzugt durchzuführen sind (vgl. AXELOS, 2011, S. 65; ISO/IEC, 2012, S. 72). Alle anderen Änderungen werden als Normal Changes bezeichnet (vgl. AXELOS, 2011, S. 65). Die zentrale Funktion des CAB führt eine weitere Verantwortungstrennung ein und birgt damit die Gefahr einer Zeitverzögerung. Jedoch bedeutet ein zentraler Anlaufpunkt nicht, dass dieser für jede einzelne Änderung aktiv werden muss (vgl. AXELOS, 2011, S. 80). Über das Konstrukt eines Standard Changes ist es möglich, die Freigabe für Änderungen, wie z. B. Releases, im Voraus zu erhalten, wenn dafür eine zuvor abgestimmtes Verfahren eingehalten wird (vgl. AXELOS, 2011, S. 69). Demnach sollte bei zeitkritischen Services das Verfahren für die Veröffentlichung von Releases standardisiert und als Standard Change registriert werden. Wie die Veröffentlichung prozessual und technisch konkret erfolgt, ist im Release and Deployment Management zu definieren (vgl. Kapitel 4.4) (vgl. AXELOS, 2011, S. 79). Das CAB als zentrale Anlaufstelle unterstützt weiterhin auch der **Nachvollziehbarkeit** aller Changes, da dort alle Änderungen geprüft bzw. registriert werden. Diese Regelung ist analog der Change-Kultur unabhängig von spezifischen Service-Merkmalen. Im Sinne der kontinuierlichen Verbesserung ist es weiterhin generell sinnvoll die **Effizienz und Effektivität** der Changes zu betrachten. Der Aspekt des definierten **Wartungsfensters**, d. h. der feste Zeitraum und Zyklus in dem Änderungen an produktiven Systemen durchgeführt werden, ist in mehrerer Hinsicht abhängig vom betroffenen Service. Einerseits bedeutet ein festes Wartungsfenster die zeitliche Fixierung eines Veröffentlichungstermins. Dieser muss zum Release-Zyklus aus dem TPS-Prozess passen (vgl. Kapitel 4.1). Andererseits ist bei den Wartungsfenstern darauf zu achten, dass je nach Änderungsverfahren und -vorhaben eine Serviceunterbrechung die Folge sein kann, welche entweder zu geschäftsarmer Zeit erfolgen sollte, oder über entsprechende Verfahren ganz zu vermeiden ist (vgl. AXELOS, 2011, S. 79; Baumann, 2014, S. 65; ISO/IEC, 2012, S. 72).

Relevant für die Standardtypkonstruktion im Sinne des Change Management ist damit der **Change-Typ** und die spezifische Definition von **Wartungsfenstern**. Diese Faktoren beziehen sich vorwiegend auf zeitliche, im Sinne der Freigabedauer und des Abwarten des Wartungsfensters, und qualitative Risiken, im Sinne der Serviceunterbrechung durch Wartungsfenster und der Standardisierung des Änderungsverfahrens (Change-Typ).

## Service Asset And Configuration Management

Mittels dem Service Asset And Configuration Management (SACM) Prozess wird sichergestellt, dass alle für die Serviceerbringung notwendigen Informationen zu allen Elementen verlässlich und akkurat verwaltet werden und bei Bedarf zur Verfügung stehen (vgl. AXELOS, 2011, S. 89). Diese Informationen umfassen einerseits spezifische Details der Elemente sowie deren Beziehungen zueinander (vgl. AXELOS, 2011, S. 90). Das Hauptziel des SACM ist somit die Sicherstellung, dass alle Elemente über den kompletten Lebenszyklus ordnungsgemäß identifiziert und gepflegt sind (vgl. AXELOS, 2011, S. 90). Dazu wird ein Configuration Management System (CMS) benötigt, welches alle diese Informationen speichert und anderen ITIL-Prozessen zur Verfügung stellt (vgl. AXELOS, 2011, S. 90). Der Mehrwert von SACM liegt in der Transparenz über die Servicelandschaft, welche vor allem im Falle von Audits zur Vermeidung von Mehrkosten und Strafzahlungen führt (vgl. AXELOS, 2011, S. 90). Weiterhin führt diese Transparenz zu einer besseren Planbarkeit von Änderungen sowie zu einer höheren Erfolgsquote bei deren Durchführung (vgl. AXELOS, 2011, S. 91). Zudem ermöglicht SACM die Nachverfolgbarkeit von Änderungen von der Anforderungserhebung bis zum veröffentlichten Release sowie die Kostenzuordnung auf Service-Ebene (vgl. AXELOS, 2011, S. 91). Die Regularien für das SACM hängen stark von den CHM und Release And Deployment Management (RDM) Prozessen ab und lassen sich nicht unabhängig verallgemeinern (vgl. AXELOS, 2011, S. 91). Aufgrund des hohen Aufwands für das SACM ist bei der Ausgestaltung darauf zu achten, welcher Bedarf dafür tatsächlich vorherrscht und ob dieser ggf. nur bestimmte Elementtypen oder Services betrifft (vgl. AXELOS, 2011, S. 91). Dieser Bedarf bezieht sich vorwiegend auf gesetzliche Regelungen in Abhängigkeit zur jeweiligen Branche bzw. zum Marktes und der genutzten Lizenzverträge (vgl. AXELOS, 2011, S. 90). Daher besteht kein erkennbarer direkter Bezug zum Produktlebenszyklus oder den genannten Risikotypen (vgl. Kapitel 3.6). Der Prozess wird demnach für die Standardtypen nicht weiter betrachtet, da er unabhängig davon zu gestalten ist.

## Release And Deployment Management

Das Release And Deploy Management ist für die Planung und Steuerung des Baus, des Tests und der Installation von Releases verantwortlich (vgl. AXELOS, 2011, S. 114). Es dient der Bereitstellung neuer Funktionalität gemäß der Geschäftsanforderungen, wobei die Integrität bestehender Services gesichert wird (vgl. AXELOS, 2011, S. 114; ISO/IEC, 2012, S. 72). Alle Änderung in der Produktivumgebung müssen über diesen Prozess erfolgen (vgl. ISO/IEC, 2011, S. 15). Daher ist es von zentraler Bedeutung für die Standardtypen des Release-Managements. Zielstellungen sind die Abstimmung von Release-Plänen, die Bereitstellung und Lagerung getesteter Release-Pakete, die Veröffentlichung der Releases auf die Produktionsumgebung und die Befähigung der Organisation die neuen Services zu betreuen (AXELOS, 2011, S. 114 f.). Die eigentliche Testdurchführung wird hier sichergestellt, aber über den Prozess der Service Validation And Testing (SVT) vorgenommen (vgl. AXELOS, 2011, S. 115). Der Mehrwert vom RDM-Prozess liegt in der schnellen Bereitstellung von Änderungen unter optimierten Kosten und Risiken sowie der Sicherstellung, dass dabei kein Schaden entsteht (vgl. AXELOS, 2011, S. 115). Dabei ist zu beachten, dass eine ausgewogene Balance zwischen Kosten, Stabilität und Agilität eingehalten wird (vgl. AXELOS, 2011, S. 115). Dieses Verhältnis kann sich von Service zu Service unterscheiden (vgl. AXELOS, 2011, S. 115). Für einen Service steht die Stabilität im Vordergrund für einen anderen Service dafür die Agilität (vgl. AXELOS, 2011, S. 115). Aufgrund der Individualität in Abhängigkeit zum konkreten Service werden in ITIL keine konkreten Regeln aber zwei Beispiele genannt (vgl. AXELOS, 2011, S. 115):

* Alle Änderungen und Releases müssen vor der Veröffentlichung auf ihre Belastbarkeit getestet werden.
* Alle Änderungen werden über ein jährliches Release veröffentlicht und zwischendurch dürfen nur kritische Fehler behoben werden.

Die beiden Beispiele zeigen einen engen Bezug zum TPS-Prozess. Beide beziehen sich auf **Kriterien zur Freigabe** eines Releases hinsichtlich der notwendigen Tests bzw. des konkreten Ziels (vgl. Kapitel 4.1). Das zweite Beispiel greift zudem noch den **Release-Zyklus** auf (vgl. Kapitel 4.1).

Neben diesen Beispielen werden bzgl. des RDM-Prozess noch weitere relevante Differenzierungsmerkmale vorgestellt, welche je nach Service unterschiedlich zu gestalten sind. Dazu gehört die Definition der **Release-Einheiten**, welche immer komplett veröffentlicht werden müssen (vgl. AXELOS, 2011, S. 116; ISO/IEC, 2011, S. 24). Ein kritischer Service darf möglicherweise nur vollständig verändert werden, damit keine internen Kompatibilitätsprobleme auftreten und besteht demnach nur aus einer Release-Einheit (vgl. AXELOS, 2011, S. 116). Für einen Internetauftritt könnte dagegen jede einzelne Webseite eine Release-Einheit darstellen (vgl. AXELOS, 2011, S. 116). Entscheidend für die Risikobetrachtung sind die Komplexität einer Release-Einheit und die Abhängigkeit zu anderen Komponenten. Je kleiner eine Release-Einheit ist, desto geringer ist der Anpassungs- und Testaufwand und desto schneller kann eine Änderung veröffentlicht werden. Dies bedingt jedoch, dass die Release-Einheiten voneinander unabhängig sind, da sonst die Auswirkungen auf die anderen Module zu prüfen ist. Die Abhängigkeit zwischen Release-Einheiten kann z. B. durch (temporäre) Abwärtskompatibilität mittels loser Kopplung aufgelöst werden (vgl. Baumann, 2014, S. 31). Neben der Release-Einheit werden für den RDM-Prozess zudem zwei unterschiedliche **Release-Methoden** beschrieben (vgl. AXELOS, 2011, S. 119). In der „big bang“-Variante erfolgt die Veröffentlichung des Releases auf alle Bestandteile und Nutzer gleichzeitig, um die Systemkonsistenz sicherzustellen (vgl. AXELOS, 2011, S. 119). Im Gegensatz dazu wird im Phasenansatz das Release stufenweise an die einzelnen Nutzer verteilt (vgl. AXELOS, 2011, S. 119). Beide Ansätze haben ihre individuellen Vor- und Nachteile. Das „big bang“-Release vermeidet die parallele Unterstützung von mehreren Versionen, wodurch Kosten reduziert werden, lässt sich je nach Änderungskomponente aber nicht immer realisieren und kann im Fehlerfall das komplette Unternehmen negativ beeinflussen (vgl. AXELOS, 2011, S. 120). Über das stufenweise Verfahren dagegen betreffen Fehler nur einen Teil der Nutzer, jedoch dauert es bei diesem Ansatz auch entsprechend länger bis das Release vollständig verteilt wurde (vgl. AXELOS, 2011, S. 120). Eine besondere Stufe im phasenweisen Release stellt der Pilot dar (vgl. AXELOS, 2011, S. 129). In der Pilotphase wird das neue Release nur einen definierten kleinen Nutzerkreis verteilt, um den Echteinsatz zu testen (vgl. AXELOS, 2011, S. 129). Prinzipiell eignet sich dieser Ansatz auch für den sogenannten A/B-Test. Denn dabei werden Änderungen, wie z. B. neue Funktionen oder eine neue Oberfläche, nur an einen Teil der Endnutzer gegeben, um zu prüfen, wie die Änderungen von den unterschiedlichen Nutzern angenommen werden (vgl. Kohavi & Longbotham, 2015, S. 4). Das dritte Differenzierungsmerkmal für den RDM-Prozess ist der **Automationsgrad** der Veröffentlichung (vgl. AXELOS, 2011, S. 121). Die Automatisierung stellt die Konsistenz und die Wiederholbarkeit einer Release-Veröffentlichung sicher, ist jedoch nicht immer möglich oder kosteneffizient (vgl. AXELOS, 2011, S. 121; Baumann, 2014, S. 50). Wenn jedoch zu viele manuelle Aktivitäten durchgeführt werden, ist dies ein Risiko für die Einhaltung der Zeit-, Qualitäts- und Kostenvorgaben, da sie länger dauern, fehleranfälliger sind und mehr Personalressourcen benötigen (vgl. AXELOS, 2011, S. 121). Die Automatisierung kann eine Vielzahl unterschiedlicher Aktivitäten unterstützen (vgl. AXELOS, 2011, S. 121 f.; 133; Baumann, 2014, S. 17):

* Übersichts-Scans der Systemumgebung
* Bau und Paketierung der Software
* Abgleich der Umgebung mit dem Sollzustand aus dem CMS
* Aktualisierung der Daten im CMS
* Installation der Software und Konfiguration der Systeme

Neben den genannten Entwurfsaspekten für RDM werden in ITIL noch zwei weitere Faktoren beschrieben, die **Remediation** und der **Early Life Support** (vgl. AXELOS, 2011, S. 143). Die Remediation dient dazu einen Service wieder in einen definierten Zustand zu überführen, nachdem ein Release fehlgeschlagen ist, weil z. B. ein Fehler aufgetreten ist (vgl. AXELOS, 2011, S. 143; ISO/IEC, 2011, S. 24). Typischerweise wird in solch einem Fall die Installation rückgängig gemacht - der sogenannte „back out“ (vgl. AXELOS, 2011, S. 143). Da früher oder später die Installation aber sowieso wiederholt werden muss und möglicherweise der Fehler nicht nachgestellt werden kann, bietet sich auch die Fehleranalyse und Behebung über einen Emergency Change an (vgl. AXELOS, 2011, S. 143), auch als „go forward“ bezeichnet. Die Entscheidung zwischen den beiden Richtungen ist von mehreren Faktoren abhängig. Einerseits führt der „back out“ zur Verzögerung in der Bereitstellung des Releases und damit zu einem zeitlichen Risiko, andererseits führt der Betrieb im Fehlerzustand zu einem Qualitäts- und Kostenrisiko. Je nachdem welches Risiko überwiegt, ist demnach eine Entscheidung zu treffen. Der zweite genannte Faktor, der Early Life Support, beschreibt eine zweistufige Überführung in den Wirkbetrieb (vgl. AXELOS, 2011, S. 143). In der ersten Stufe nach der Veröffentlichung erfolgt die Betreuung durch Spezialisten, z. B. die Entwickler des Releases und erst nach der Erreichung von definierten Kriterien erfolgt der Verantwortungsübergang an die Standardeinheiten (vgl. AXELOS, 2011, S. 143). Durch die Phasen ist sichergestellt, dass Fehler aufgrund des speziellen Wissens schneller behoben werden können und die Standardeinheiten nicht durch unerwartete Probleme überfordert werden (vgl. AXELOS, 2011, S. 143). Der Einsatz der Speziallisten kann jedoch zu erhöhten Kosten gegenüber dem direkten Einsatz der Standardeinheiten führen, da diese möglicherweise einen höheren Stundensatz haben. Weiterhin werden Entwickler möglicherweise von der Bearbeitung neuer Releases abgehalten, da sie Fehler im alten Release beheben müssen. Der Einsatz des Early Life Supports muss demnach ebenfalls je nach Risikosituation abgewogen werden.

Für die Standardtypkonstruktion im Kontext des RDM sind demnach die Aspekte der **Release-Einheit**, der **Release-Methode**, des **Automatisierungsgrads**, der **Remediation** und des **Early Life Supports** relevant, welche auf alle Risikotypen Auswirkungen haben.

## Service Validation And Testing

Der Grundgedanke des Prozesses der Service Validation And Testing ist die Qualitätssicherung (vgl. AXELOS, 2011, S. 150). Durch die Testdurchführung soll sichergestellt werden, dass ein Release bereit für den produktiven Einsatz ist (vgl. AXELOS, 2011, S. 150). Die Durchführung von Tests ist ebenfalls in der ISO/IEC 20000-1:2011 vorgeschrieben (vgl. ISO/IEC, 2011, S. 15). Die Tests führen zur Reduktion von Fehlern und damit zur Vermeidung von Kosten (vgl. AXELOS, 2011, S. 150). Zu beachten ist dabei, dass die Testdurchführung nur ein gewisses Vertrauen, aber keine Gewissheit schaffen kann, d. h. trotz aller Tests kann es im produktiven Einsatz zu Fehlern kommen (vgl. AXELOS, 2011, S. 151). Hinsichtlich des SVT-Prozess werden innerhalb von ITIL folgende Richtlinien vorgeschlagen (vgl. AXELOS, 2011, S. 151 f.):

* Die Testdurchführung darf nicht durch die Entwickler erfolgen.
* Die Testbewertungskriterien müssen vor den Tests definiert werden.
* Die Testumgebungen müssen vor der Testdurchführung immer auf den definierten Ausgangszustand zurückgesetzt werden.
* Alle Testartefakte sollten wiederverwendbar sein und wiederverwendet werden.
* Die Testdurchführung soll so zeitig wie möglich und im Entwicklungsprojekt integriert erfolgen.
* Die Testdurchführung soll risikobasiert ausgerichtet werden.
* An der Testdurchführung sollen auch Nutzer und andere Stakeholder beteiligt werden.
* Mess- und Überwachungssysteme sollen zur Effizienz- und Effektivitätssteigerung eingesetzt werden.
* Prüfung der Automatisierung der notwendigen Tätigkeiten bei komplexen und kritischen Systemen sowie bei tendenziell zeitkritischen Releases.

Die **Trennung der Testaktivitäten von den Entwicklungsaktivitäten** wird in ITIL als generelle Richtlinie beschrieben, da durch die unabhängige Prüfung eine höhere Qualität und Objektivität der Tests erreicht werden soll. Dies ist aber individuell zu betrachten, da es Einfluss auf die Dauer und Kosten eines Releases hat. Die Dauer steigt, da ein separater Tester sich erst mit dem Release vertraut machen muss und dafür von den Entwicklern geschult werden muss. Weiterhin kann der Tester Fehler nicht selbst beheben, so dass der Entwickler im Fehlerfall die Testdurchführung nachvollziehen können muss und sich daher trotzdem mit den Tests beschäftigen muss. Da der Tester das System möglicherweise trotz intensiver Schulung nicht vollständig versteht, kann es zur fehlerhaften Testdurchführung kommen, deren Testergebnisse nicht valide sind. Dazu kommt der Kommunikationsmehraufwand aufgrund der höheren Anzahl der Beteiligten. Sollte ein Service eher einem zeitlichen Risiko unterliegen, kann es demnach sinnvoll sein die Testdurchführung und Entwicklung nicht zu trennen. Die **Festlegung der Testbewertungskriterien** vor der Testdurchführung ist auch innerhalb der ISO/IEC 20000-1:2011 definiert (vgl. ISO/IEC, 2011, S. 15; 24). Sie ist unabhängig vom Risikotyp zu vollziehen, da ohne die Definition eines Solls und damit eines Ziels die Testaktivitäten wahllos und nicht valide sind. Dies gilt auch für die Regelung hinsichtlich der **Wiederherstellung des Ausgangszustands** vor jedem Testlauf, da die Ergebnisse sonst nicht vergleichbar und nachvollziehbar sind. Die **Wiederverwendbarkeit von Testartefakten** steht in starker Abhängigkeit zum Release-Zyklus des TPS-Prozess (vgl. Kapitel 4.1). Je kürzer der Release-Zyklus ist, desto geringer ist der Änderungsgrad und desto eher können bestehende Testartefakte ohne Anpassung wiederverwendet werden (vgl. AXELOS, 2011, S. 152). Bei einem hohen Release-Zyklus kann es demnach sein, dass die alten Testartefakte, z. B. die Testfallbeschreibungen, gar nicht mehr passen, da sich der komplette Geschäftsprozess verändert hat. Würde man hierbei versuchen die Testartefakte trotzdem wiederverwendbar zu gestalten, wäre der Aufwand entweder umsonst oder die Test wären so generisch, dass sie keine Aussagekraft besitzen. Das wäre z. B. der Fall, wenn im Test nur der Aufruf der Startseite eines Webshops erfolgen würde. Dieser ist zwar sicherlich auch für weitere Releases anwendbar, sagt aber über die Gesamtqualität wenig aus. Aufgrund des engen Bezugs zum Release-Zyklus erfolgt keine eigenständige Betrachtung der Wiederverwendbarkeit. Die Richtlinie zur **frühzeitigen** **Integration der Testaktivitäten in den Entwicklungsprozess** ist besonders bei Services mit einem hohen Qualitäts- und Zeitrisiko aber auch im Hinblick auf die Kosten relevant. Durch die frühzeitige Einbindung wird vermieden dass die Testaktivitäten, welche gewöhnlich bei einem wasserfallmodellbasierten Projektvorgehen am Ende stehen, aufgrund von Verzögerungen in vorherigen Phasen reduziert werden oder ganz wegfallen (vgl. AXELOS, 2011, S. 174). Dies kann sich dramatisch auf die Qualität auswirken, da der Endnutzer zum Tester des Releases wird. Zeitlich ist die frühzeitige Integration dann wichtig, falls in einer späten Phase im Projekt oder im Wirkbetrieb ein Fehler gefunden wurde, der zu Beginn des Projekts gemacht wurde und ggf. einen kompletten Neubeginn der Arbeiten erfordert. Solch eine Situation bedeutet dann natürlich auch erhöhte Kosten (vgl. Lennertz, 2006, S. 143 f.). Der Zusammenhang zwischen den Kosten zur Fehlerbehebung und dem Zeitpunkt der Fehlerentdeckung relativ zur Projektphase wird in Abbildung 4.1 zur Verdeutlichung dieses Sachverhalts dargestellt. In der Literatur gibt es jedoch auch kritische Stimmen, die diesen Zusammenhang aufgrund mangelhafter empirischer Nachweise in Frage stellen und als reine ideologische Auffassung bewerten (vgl. Bossavit, 2015, S. 87 ff.). Unabhängig von konkreten Verhältniswerten scheint der generelle Zusammenhang logisch, da die textuelle Fehlerbehebung durch eine Prüfung der Anforderungsdokumentation wesentlich kürzer scheint, als die Fehlerbehebung im produktiven System, welche ggf. Architektur- oder Technologieänderungen bedingt und alle durchgeführten Aktivitäten wie z. B. Tests erneut nach sich zieht sowie möglicherweise Umsatzverlust oder Entschädigungszahlungen bedeutet.



Abbildung .: Fehlerbehebungskosten

Quelle: in Anlehnung an (Boehm 1976, S. 1228)

Die **risikobasierte Ausrichtung der Tests** ist prinzipiell eine generelle Richtlinie, welche für alle Services gilt, wobei natürlich jeder Service einen entsprechenden Risikotypen vorweist. Diese Richtlinie unterstreicht zugleich den Bedarf von Standardtypen, da sie zu generisch ist, aber auch den Ansatz hinsichtlich der Risiken aus den Lebenszyklusphasen unterstützt, da es risikobasiert erfolgen soll. Die **Teilnahme der Nutzer an den Tests** dient der Tauglichkeitsbetrachtung des neuen Releases und wird auch von der ISO/IEC 20000-2:2012 empfohlen (vgl. ISO/IEC, 2012, S. 75). Einerseits bedeutet die Beteiligung weiterer Personen erhöhte Kosten, aufgrund des zeitlichen Aufwands, andererseits kann es eine höhere Qualität bringen, da die Nutzer möglicherweise eine andere Perspektive als z. B. ein Entwickler oder Tester haben. Zu beachten ist auf jeden Fall die zeitliche Dimension, da die Beteiligung weiterer Personen sich auf die Dauer der Tests negativ auswirken kann, wenn sie nicht parallelisiert werden können. Der **Einsatz von Mess- und Überwachungssystemen** bei den Tests ist in Abhängigkeit vom konkreten Test vorzusehen. Bei Lasttests ist die Nutzung für die Vergleichbarkeit von Testläufen entscheidend, da ohne Metriken, wie z. B. der Transaktionsdurchsatz oder die Netzwerkauslastung die Ergebnisse nicht objektiv auswertbar wären (vgl. AXELOS, 2011, S. 164). Aber auch bei funktionalen Tests, welche nur die fachliche Korrektheit einer Transaktion prüfen, können Messwerte wie diese hilfreich sein, um zum Beispiel eine Fehlerbehebung mit negativen Auswirkungen auf die Antwortzeit aufzudecken. Die Einhaltung der Richtlinie ist demnach nur indirekt abhängig vom Service, da es direkt von den durchzuführenden Tests abhängig ist, welche je nach Qualitätsverständnis und Vorgaben aus dem Risiko-Management des Unternehmens abgeleitet werden (vgl. AXELOS, 2011, S. 152). Neben den bereits genannten Testverfahren werden in ITIL noch viele andere genannt (vgl. AXELOS, 2011, S. 161). Eine Vorstellung der Genannten oder der Weiteren ist an dieser Stelle nicht wertstiftend und würde den Umfang dieser Ausarbeitung überschreiten, daher wird auf die Fachliteratur verwiesen. Die Vorgabe hinsichtlich der **Automation der Testaktivitäten** wird in ITIL selbst schon spezifisch an konkrete Risiken gebunden (vgl. AXELOS, 2011, S. 151). Demnach ist vor allem bei komplexen, geschäftskritischen oder bei zeitkritischen Services die Testautomatisierung einzusetzen (vgl. AXELOS, 2011, S. 151 f.). Hinsichtlich der Testautomatisierung gelten hierbei die gleichen Ausführungen wie bei der Wiederverwendbarkeit von Testartefakten oder der Automatisierung von Aktivitäten im RDM-Prozess (vgl. Kapitel 4.4).

Die relevanten Faktoren des SVT-Prozesses sind demnach die **Trennung von Test und Entwicklung**, die **frühzeitigen Tests**, die **Endnutzertests** und die **Testautomatisierung,** aufgrund deren Bezug zu den Risikotypen.

## Change Evaluation

Der Change Evaluation (CHE) Prozess dient der Bereitstellung eines konsistenten und standardisierten Verfahrens zur Bestimmung von potentiellen Auswirkungen sowie dem Abgleich mit den tatsächlichen Auswirkungen durch einen Change (vgl. AXELOS, 2011, S. 175). Die Zielstellung umfasst die Bereitstellung von akkuraten Informationen an das Change Management sowie an andere Stakeholder zur Entscheidungsfindung (vgl. AXELOS, 2011, S. 175). Die Informationen entstehen durch die Evaluierung der gewünschten und ungewünschten Effekte bei der Durchführung der geplanten Änderung (vgl. AXELOS, 2011, S. 175). Der Mehrwert liegt in der Beschaffung einer Entscheidungsgrundlage sowie in der Erarbeitung von Ansätzen zur zukünftigen Verbesserung im Rahmen des ITIL Continual Service Improvement (vgl. AXELOS, 2011, S. 175). Die Richtlinienbeispiele für den CHE-Prozess innerhalb von ITIL umfassen die folgenden vier Elemente (vgl. AXELOS, 2011, S. 175):

* Jede Änderung muss vor der Durchführung evaluiert werden.
* Nur kritische Änderungen müssen den formalen Prozess durchlaufen.
* Die Evaluierung umfasst auch andere von der Änderung betroffene Services.
* Bei Abweichungen von den Planeffekten muss der Kunde bzgl. des weiteren Vorgehens befragt werden.

Wie den bisherigen Ausführungen entnommen werden kann, herrscht eine enge Verbindung zum CHM-Prozess (vgl. AXELOS, 2011, S. 180). Hinsichtlich der zwingenden **Evaluierung für jeden Change** wird auf die bereits genannte Change-Kultur verwiesen, welche diese Evaluierung zur Risikobewertung benötigt und damit ebenfalls allgemein einzuhalten ist (vgl. Kapitel 4.2). Die Anwendung eines **formellen Prozesses für kritische Änderungen** unterstützt die unterschiedlichen Risikotypen ähnlich der unterschiedlichen Change-Typen des CHM-Prozess (vgl. Kapitel 4.2). Der formelle Prozess ist aufgrund des Bezugs auf kritische Änderungen nur bei einem Normal Change bzw. Emergency Change anzuwenden, da ein Standard Change nur ein geringes Risiko haben darf (vgl. Kapitel 4.2). Die beiden genannten Punkte bieten aufgrund der engen Integration in den CHM-Prozess keinen zusätzlichen Mehrwert für die Standardtypenbetrachtung. Die **Evaluierung anderer betroffener Services** durch eine Änderung sollte prinzipiell immer durchgeführt werden, da die Änderung am eigenen Service möglicherweise klein ist, aber einen kritischen Service negativ beeinflusst. Die Evaluierung kann entfallen, wenn der Service keinen Bezug zu anderen Services hat oder Maßnahmen zur Vermeidung von Fremdbeeinflussungen vorgenommen werden. Da die Evaluierung bei einer größeren Anzahl von abhängigen Services aufwendig werden kann, sollten bei zeitkritischen Services demnach solche Maßnahmen vorgenommen werden. Diese beziehen sich jedoch auf Gestaltungsfaktoren aus dem RDM-Prozess und wurden dort bereits betrachtet (vgl. Kapitel 4.4). Die **Befragung des Kunden bei Planabweichungen** weist keine risikobezogene Unterscheidung auf, da unabhängig von Zeit, Kosten oder Qualität in einer derartigen Situation nur der Kunde das weitere Vorgehen bestimmen kann, es sei denn es wurde vorab definiert. Innerhalb der ISO/IEC 20000-1:2011 wird zudem die Bereitstellung eines Berichts, welcher die erreichten und die erwarteten Ergebnisse vergleichend darstellt, für die Stakeholder vorgeschrieben (vgl. ISO/IEC, 2011, S. 15). Dies ist jedoch ebenfalls unabhängig vom Service oder Risikotyp zu sehen, so dass der Prozess CHE für die Standardtypbetrachtung nicht relevant ist.

## Knowledge Management

Der letzte Prozess in der ITIL Service Transition ist das Knowledge Management (KLM) (vgl. AXELOS, 2011, S. 181). Es dient dem Austausch von Informationen sowie Erfahrungen und stellt sicher, dass diese mit minimalen Aufwand gefunden werden können (vgl. AXELOS, 2011, S. 181 f.). Die Ziele sind die Bereitstellung von verlässlichen Informationen für Entscheidungen und eine qualitative Service-Erbringung (vgl. AXELOS, 2011, S. 182). Der Mehrwert liegt demnach in der Möglichkeit des einfachen Zugriffs auf die notwendigen Informationen für jeden Service für jeden Mitarbeiter zur richtigen Zeit am richtigen Ort (vgl. AXELOS, 2011, S. 181 f.). Die Beispielrichtlinien innerhalb von ITIL für den KLM-Prozess sind sehr generisch und erfassen nur die Anforderung der allgemeinen Bereitstellung von Informationen, der regelmäßigen Aktualisierung und der Einhaltung eines abgestimmten Prozesses (vgl. AXELOS, 2011, S. 183). Die Regelungen sind nicht abhängig von konkreten Services sondern eher von der Kultur im Unternehmen (vgl. AXELOS, 2011, S. 182). Daher erfolgt für die Erarbeitung der Standardtypen keine weitere Betrachtung dieses Prozesses.

## Zusammenfassung der Faktoren des Release-Managements

Die Untersuchung der einzelnen Prozesse der ITIL Service Transition zeigt eine Vielzahl von unterschiedlichen zu beachtenden Faktoren. Wobei die Prozesse SACM, CHE und KLM für diese Arbeit keine Relevanz haben. Als Zusammenfassung der relevanten Prozesse, deren Prozessfaktoren und deren Ausprägungsmöglichkeiten dient die Tabelle 4.1:

|  |  |  |
| --- | --- | --- |
| Prozess | Prozessfaktor | Ausprägungen |
| TPS | Release-Zyklus | ad-hoc bis selten |
| Freigaberollen | keine bis viele |
| CHM | Change-Typ | Standard oder Normal |
| Wartungsfenster | notwendig oder nicht |
| RDM | Release-Einheit | klein bis groß (eine) |
| Release-Methode | „big bang“ oder Phasenansatz |
| Automatisierungsgrad | keine bis hoch |
| Remediation | „back out“ oder „go forward“ |
| Early Life Support | ja oder nein |
| SVT | Trennung Test/Entwicklung | ja oder nein |
| Frühzeitige Tests | ja oder nein |
| Endnutzertest | ja oder nein |
| Testautomatisierung | keine bis hoch |

Tabelle .: Übersicht Prozessfaktoren und Ausprägungen

Quelle: eigene Tabelle

Im nachfolgenden Hauptkapitel erfolgt nun die Zusammenstellung der Standardtypen auf Basis der Erkenntnisse zu den Prozessfaktoren und den Risikotypen (vgl. Kapitel 3.6).

# Ableitung der Standardtypen aus den Faktoren und Risiken

In Kapitel 3 erfolgte die Vorstellung der Risiken in den einzelnen Produktlebenszyklusphasen und die Zusammenfassung zu zwei unterschiedlichen Risikotypen. Die relevanten Prozessfaktoren des Release-Managements wurden in Kapitel 4 vorgestellt. Dieses Kapitel beschäftigt sich mit der Ableitung der Standardtypen. Dazu wird für die beiden Risikotypen, abgeleitet aus den Zusammenfassungen (vgl. Kapitel 3.6), jeweils in einem eigenen Kapitel die entsprechende Ausprägung der Prozessfaktoren beschrieben und abschließend eine Zusammenfassung der Standardtypen aufgestellt. Die Bezeichnung der Standardtypen leitet sich konsequenterweise aus denen der korrespondierenden Risikotypen ab.

## Zeit

Die zusammenfassende Beschreibung des Risikotyps „Zeit“ (vgl. Kapitel 3.6) ist die Basis für die folgende Ableitung der nötigen Ausprägung der Prozessfaktoren (vgl. Kapitel 4.8). Für den Release-Management-Prozessstandardtyp „Zeit“ ist der **Release-Zyklus** so kurz wie möglich zu halten, damit zeitnah auf Kundenfeedback eingegangen werden kann (vgl. Kapitel 3.6). Typischerweise liegen die Zyklen zwischen 2 bis 4 Wochen (vgl. Baumann, 2014, S. 6; Roock & Wolf, 2016, S. 1888), wobei prinzipiell auch öfter bzw. ad-hoc veröffentlich werden könnte (vgl. Puppet, 2016, S. 15). Hinsichtlich der **Freigaberollen** ist insbesondere die Ausführung in Kapitel 4.1 zu beachten, welche darauf hinweist, dass mit der Anzahl der Beteiligten die Durchlaufzeit steigt, so dass prinzipiell so wenig wie möglich unterschiedliche Personen in die Freigabe involviert sein sollten. Theoretisch könnten über automatisierte Prüfverfahren die nötigen formalen Prüfungen und Freigaben sichergestellt werden, so dass eine Änderung keine explizite manuelle Freigabe benötigt (vgl. Humble & Farley, 2010, S. 1100). Hinsichtlich des **Change-Typs** ist aufgrund der zeitlichen Risikosituation der Standard-Change zu bevorzugen, da er aufbauend auf den Erläuterungen zu den Freigaberollen, die Durchlaufzeit verkürzt (vgl. Kapitel 4.2). **Wartungsfenster** sind im Standardtyp „Zeit“ mittels passenden Installationsverfahren zu vermeiden, da Sie die ad-hoc Ausführung von Releases unterbinden und eine Serviceunterbrechung bedeuten (vgl. Kapitel 4.2). Ist dies nicht möglich sein, sollten die Wartungsfenster gemäß dem genannten Release-Zyklus ausgelegt werden (vgl. Kapitel 4.2). Die **Release-Einheiten** sollten so klein wie möglich gestaltet werden, damit der Änderungsumfang minimal ist und somit das Release schnell zur Verfügung gestellt werden kann (vgl. Kapitel 4.4). Dafür sind entsprechende Vorkehrungen in der Architektur des Systems und Arbeitsweisen in der Software-Entwicklung von Nöten (vgl. Kapitel 4.4). Die **Veröffentlichungsmethode der Releases** entspricht dem Phasenansatz, da hier am schnellsten und besten das Feedback der Kunden eingeholt werden kann (vgl. Kapitel 4.4). Verstärkt wird dieser Effekt mit dem benannten A/B-Tests (vgl. Kapitel 4.4). Der **Automatisierungsgrad** ist einer der wichtigsten Prozessfaktoren für diesen Standardtypen, da hierdurch die sonst vielen notwendigen manuellen Schritte im Voraus automatisiert werden, so dass sie im Release-Fall schnell und gesichert erfolgen können (vgl. Kapitel 4.4). Je höher die Automation demnach ist, desto schneller kann ein Release veröffentlich werden (vgl. Kapitel 4.4). Im Falle eines Fehlers bei einem Release, sollten im Standardtyp „Zeit“ die „go forward“-Variante des Prozessfaktors „**Remediation**“ gewählt werden, damit die neuen Änderungen trotz des Fehlers eher beim Kunden sind (vgl. Kapitel 4.4). Der **Early Life Support** sollte nur in begründeten Fällen genutzt werden, da sonst die Ressourcen für die Weiterentwicklung fehlen (vgl. Kapitel 4.4). Anders ist dies im sogenannten DevOps-Ansatz, wo ein Team für das Gesamtprodukt und damit alle Tätigkeiten gemeinsam verantwortlich ist, d. h. auch der Support und die Wartung durch die Entwicklung erfolgen (vgl. Kapitel 2.4). Die vorgeschlagene **Trennung von Test und Entwicklung** ist bei diesem Standardtyp aufgrund des damit einhergehenden Zeitverzugs nicht zu empfehlen (vgl. Kapitel 4.5). Die Integration von **frühzeitigen Tests** ist aufgrund der zeitlichen Kritikalität unverzichtbar (vgl. Kapitel 4.5). Im DevOps-Ansatz sollten ohnehin bei jedem Release die gleichen Tests durchgeführt, damit trotz der Geschwindigkeit die Qualität sichergestellt ist (vgl. Baumann, 2014, S. 50). **Endnutzertests** sind auch im zeitkritischen Standardtyp sinnvoll, jedoch nur zu empfehlen, wenn diese parallel bzw. integriert erfolgen und keine zusätzliche Verzögerung bedingen (vgl. Kapitel 4.5). Analog der Ausführungen zum Automationsgrad ist eine hoher Grad an **Testautomatisierung** ein wichtiger Prozessfaktor für diesen Standardtypen, um trotz der Schnelligkeit die Qualität nicht zu vernachlässigen (vgl. Kapitel 4.5).

## Kosten

Basierend auf der Beschreibung des Risikotyps „Kosten“ (vgl. Kapitel 3.6) erfolgt analog zum Standardtyp „Zeit“ die Ableitung der passenden Ausprägungen der Prozessfaktoren (vgl. Kapitel 4.8) für das Release-Management. Der **Release-Zyklus** ist bei diesem Standardtyp nicht durch die Anforderung an Schnelligkeit und Flexibilität geprägt, sondern durch das Kostenoptimierungsziel (vgl. Kapitel 3.6). Daraus lässt sich aber nicht ableiten, dass der Release-Zyklus lang sein muss, um kostengünstig zu sein, da dann der Aufwand pro Release steigt (vgl. Kapitel 4.1). Typischerweise ergeben sich in solchen Fällen jedoch Release-Zyklen von durchschnittlich 12 bis 2 Releases pro Jahr (vgl. Puppet, 2016, S. 15), wobei es auch nur bei Bedarf Releases geben kann, falls selten am System etwas verändert wird. Die **Freigaberollen** sollten aufgrund der negativen Effekte auf die Organisation (vgl. Kapitel 4.1) ebenfalls nicht zu sehr verteilt sein, wobei der Aufwand für eine automatische Prüfung jedoch entfallen kann. Aufgrund des langsameren Release-Zyklus steigt die Wahrscheinlichkeit, dass für die Veröffentlichungen unterschiedlichen Verfahren und Abläufe angewendet werden müssen, wodurch ein Standard-Change als **Change-Typ** nicht genutzt werden kann (vgl. Kapitel 4.2). Aufgrund der fehlenden zeitlichen Risikos sind die Durchlaufzeiten eines Normal-Changes jedoch akzeptabel (vgl. Kapitel 4.2). Sollte einer Unterbrechung der Service-Verfügbarkeit aufgrund von anderen Faktoren, wie z. B. vertraglich zugesicherte Verfügbarkeiten, nichts entgegensprechen, können aufgrund der Kosten für die Einrichtung und Pflege eines derartigen Verfahrens stattdessen **Wartungsfenster** genutzt werden (vgl. Kapitel 4.2). Die **Release-Einheiten** verhalten sich wie der Release-Zyklus. Aus der Kostenperspektive lässt sich nicht allgemein ableiten, wie viele Einheiten optimal sind. Viele kleine Einheiten sind wahrscheinlich aufgrund der Komplexität durch externe Abhängigkeiten so teuer, wie eine große Einheit aufgrund der inneren Komplexität und demnach liegt die Wahrheit dazwischen. Aus Sicht der Aufwände für die parallele Betreuung unterschiedlicher Versionen empfiehlt sich die Verwendung der „big bang“ – **Release-Methode**, wobei auch hier bei bestimmten kritischen Änderungen eine Pilotphase im Sinne der Fehlerkostenvermeidung sinnvoll ist (vgl. Kapitel 4.4). Der **Automatisierungsgrad** lässt sich ebenfalls nicht pauschal bestimmen, da dies abhängig vom Erstellungsaufwand und der tatsächlichen Wiederverwendung ist. Sollte aufgrund der langen Release-Zyklen der Wiederverwendungsgrad niedrig sein, lohnt es sich nur bei wenigen Aktivitäten aufgrund eines geringen Umsetzungsaufwands, diese zu automatisieren. Im Fehlerfall ist bei einem Release die „back out“ – **Remediation** zu wählen, da es kein Zeitrisiko gibt, welches potentielle Mehrkosten durch den Betrieb im Fehlerzustand rechtfertigt (vgl. Kapitel 4.4). Auf einen **Early Life Support** ist aufgrund der erhöhten Kosten prinzipiell auch zu verzichten, wobei es auch Fälle geben kann, bei denen durch die schnellere Fehlerlösung mehr Kosten vermieden werden, als dafür angefallen sind (vgl. Kapitel 4.4). Eine **Trennung von Test und Entwicklung** führt gemäß der Ausführungen in Kapitel 4.5 zu erhöhten Kosten und ist demnach zu vermeiden. Im Sinne der Vermeidung von späten Folgekosten sind bei diesem Standardtyp ebenfalls die **Test frühzeitig** zu integrieren. Die Involvierung von Endnutzer bei diesen Tätigkeiten ist nur bedingt zu empfehlen und dann sinnvoll, wenn dadurch späte Folgekosten vermieden werden können (vgl. Kapitel 4.5). Die **Testautomatisierung** im Release-Management ist analog des allgemeinen Automatisierungsgrad zu betrachten.

## Zusammenfassung der Standardtypen

In den vorhergehenden Kapiteln wurden die beiden Standardtypen „Zeit“ und „Kosten“ vorgestellt, welche die Adaption und Definition von Release-Management-Prozessen aufgrund der hier geleisteten Vorüberlegungen vereinfachen. Für ein Produkt des Risikotyps „Zeit“ (vgl. Kapitel 3.6) kann demnach der Release-Management-Prozess-Standardtyp „Zeit“ als Vorlage herangezogen und je nach Bedarf modifiziert werden. Selbiges gilt für Produkte des Risikotyps „Kosten“ (vgl. Kapitel 3.6) und dessen korrespondierenden Standardtyp. Eine Übersicht zu den beiden Standardtypen und deren Ausprägungen der Prozessfaktoren ist abschließend als Zusammenfassung in Tabelle 5.1 dargestellt.

|  |  |  |  |
| --- | --- | --- | --- |
| Prozess | Prozessfaktor | Standardtypen | |
| Zeit | Kosten |
| TPS | Release-Zyklus | aller 2 bis 4 Wochen bzw. ad-hoc | 12 bis 2 pro Jahr oder bei Bedarf |
| Freigaberollen | keine bzw. wenige | wenige |
| CHM | Change-Typ | Standard | Normal |
| Wartungsfenster | keine bzw. passend zu Release-Zyklus | ja |
| RDM | Release-Einheit | klein | mittel |
| Release-Methode | Phasenansatz mit A/B-Test | big bang |
| Automatisierungsgrad | hoch | niedrig |
| Remediation | go forward | back out |
| Early Life Support | nein bzw. DevOps | nein |
| SVT | Trennung Test/Entwicklung | nein | nein |
| Frühzeitige Tests | ja | ja |
| Endnutzertest | ja | bedingt |
| Testautomatisierung | hoch | niedrig |

Tabelle .: Zusammenfassung der Ausprägungen der Standardtypen

Quelle: eigene Tabelle

Im Anschluss an die Herleitung der Standardtypen und deren Ausprägung erfolgt im folgenden Kapitel die kritische Betrachtung der Erkenntnisse.

# Kritische Würdigung der Standardtypen

Die Zielsetzung dieser Ausarbeitung war die Untersuchung einer Möglichkeit zur Vereinfachung der Implementierung von Release-Management-Prozessen (vgl. Kapitel 1). Es sollte überprüft werden, ob die unterschiedlichen Risiken innerhalb der Phasen des Produktlebenszyklus eine Unterscheidung von Prozessvarianten für das Release-Management ermöglichen und wie die Prozessfaktoren zu Standardtypen zusammengefasst werden können (vgl. Kapitel 1). Aufgrund der Art der Zielstellung ist eine konkrete Messung der Zielerreichung nicht möglich. Es lässt sich aber sehr wohl sagen, dass die Zielsetzung erreicht wurde. Die Ausführungen in Kapitel 3 zeigen die unterschiedlichen Risikotypen innerhalb der Lebensphasen eines Produkts. Passend zu den Risikotypen konnten in Kapitel 4 Faktoren gefunden werden, welche je nach Ausgestaltung, besser oder schlechter zum Risikotyp passen. Die Standardtypableitung in Kapitel 5 fasst dann abschließend die unterschiedlichen Prozessfaktoren in den jeweils zum Risikotyp passenden Ausgestaltungen zusammen. Demnach wurde auch die grundlegende Hypothese, dass aufgrund des direkten Zusammenhangs zwischen der Produktlebenszyklusphase und dem Innovationsbedarf eines Produkts, eine Unterscheidung für Release-Management-Prozesse ableitbar ist, bestätigt. Da dieser Zusammenhang naheliegend ist, ist das Ergebnis zu erwarten gewesen. Überraschend dabei war jedoch die Möglichkeit, die 5 unterschiedlichen Lebenszyklusphasen eines Produkts mit 2 Standardtypen abdecken zu können (vgl. Kapitel 3.6). Eine detailliertere Unterscheidung der konkreten Risiken bzw. eine graduelle Abstufung der Betroffenheit von einem Risikotyp, hätte vermutlich zu weiteren Standardtypen geführt, wäre aber konträr zum Ziel der Vereinfachung und Standardisierung gewesen. Zu beachten ist beim Einsatz der Standardtypen die fehlende Betrachtung des konkreten Kosten-Nutzen-Verhältnisses, da dies nur individuell bestimmt werden kann. Unerfüllt bleiben weiterhin konkrete Betrachtungen bei den Prozessfaktoren, welche nicht binär zu beantworten sind, wie z. B. der Release-Zyklus, die Freigaberollen und der Testautomatisierungsgrad (vgl. Kapitel 5). Dies liegt vor allem daran, das deren Ausgestaltung auch von anderen Dimensionen als dem Risikotyp abhängt, z. B. der Aufteilung der Verantwortung in der Organisation (vgl. Kapitel 4.1). Zur Erweiterung der Standardtypen bietet sich daher die Betrachtung von weiteren Dimensionen an. Diese könnten zum Beispiel die Marktstellung (z. B. Monopol oder Monopson), die Organisationsform (z. B. zentralisiert oder dezentralisiert), die Branche (z. B. Logistik, Medizin oder Rahmfahrt) oder der Kundentyp (z. B. Business-to-Business oder Business-to-Customer) sein. Trotzdem sind die vorliegenden Standardtypen aufgrund ihrer Einfachheit als Leitfaden bei der Implementierung von Release-Management-Prozessen von Bedeutung, da ein aufwendiges Studium der gängigen Rahmenwerke entfallen kann. Möglicherweise führt die Betrachtung von weiteren Dimensionen zum Verlust der Einfachheit und letztendlich zu einem ähnlichen Gebilde wie ITIL, da es versucht allgemeingültig zu sein und dabei die gesuchte Individualität verloren geht. Offen ist letztendlich, wie genau eine Umstellung zwischen den beiden Standardtypen aussehen kann. Diese wäre nach Abschluss der Einführungsphase und mit Beginn der Wachstumsphase durchzuführen, wurde allerdings in dieser Ausarbeitung nicht beschrieben.

# Evaluation der Zielerreichung und Ausblick

Die steigende Bedeutung der IT, der hohe Marktdruck nach Innovationen und die generischen Vorgaben zur Gestaltung von Release-Management-Prozessen waren Anlass dieser Ausarbeitung (vgl. Kapitel 1). Zu Beginn wurde anhand der unterschiedlichen Beschreibungen in der Literatur ein Basisproduktlebenszyklus abgeleitet (vgl. Kapitel 2.3). Daraufhin wurde ermittelt, wie die generischen Vorgaben zum Release-Management anhand der Risiken innerhalb der Produktlebenszyklusphasen unterschieden werden können (vgl. Kapitel 3). Diese Untersuchung offenbarte zwei wesentliche Risikotypen „Zeit“ und „Kosten“ (vgl. Kapitel 3.6). Anhand dieser Unterscheidung wurden 13 konkrete Release-Management-Prozesskonfiguration in jeweils zwei Standardtypen abgebildet (vgl. Kapitel 5.3) und nachfolgend evaluiert (vgl. Kapitel 6). Diese Standardtypen können Unternehmen bei der Implementierung des Release-Managements helfen, da sie schneller zu erfassen und bezogen auf die Fragestellung konkreter als Rahmenwerke, wie z. B. ITIL, sind. Die Standardtypen helfen aber auch etablierten Unternehmen in der Überprüfung der bestehenden Prozesse und darauffolgender Optimierungen. Aufgrund der Fokussierung auf den Produktlebenszyklus fehlt die Betrachtung anderer Dimensionen, welche ebenfalls Einfluss auf die Release-Management-Prozesse haben. Daher sollten zukünftige Forschungsarbeiten die hier definierten Standardtypen aufgreifen und um weitere Dimensionen ergänzen. Weiterhin sollten Untersuchungen zur Erlangung von empirischen Nachweisen erfolgen, um eine Konkretisierung der nicht binären, aber auch zur Bestätigung der binären Prozessfaktoren zu erreichen. Eine andere Untersuchungsrichtung ist die Betrachtung des Übergangs zwischen den beiden Standardtypen. Ein letzter Ansatz für die weitere Forschung ist der Versuch der allgemeinen Ermittlung des Kosten-Nutzen-Verhältnisses, welche aktuell nur individuell erfolgen kann.

# Literaturverzeichnis

Aumayr, K. J. (2009). *Erfolgreiches Produktmanagement. Tool-Box für das professionelle Produktmanagement und Produktmartketing* (2. Auflage Ausg.). Wiesbaden: Gabler.

Auner, K. (2008). *ITIL-COBIT-Mapping: Gemeinsamkeiten und Unterschiede der IT-Standards* (1. Ausg.). Düsseldorf: Symposium Publ.

AXELOS. (2011). *ITIL Service Transition.* o. O.: TSO.

Baumann, J. (2014). *Continuous Delivery* (1. Ausg.). Heidelberg: dpunkt.Verlag.

Baumöl, U. (April 2007). Business-IT-Alignment durch Projektportfolio-Management und -Controlling. *HMD - Praxis der Wirtschaftsinformatik*(254), 71-81.

Beims, M., & Ziegenbein, M. (2015). *IT-Service-Management in der Praxis mit ITIL* (4. Auflage Ausg.). München: Carl Hanser Verlag.

Boehm, B. W. (1976). Software Engineering. *IEEE Transactions On Computers*(12), 1226-1241.

Bossavit, L. (2015). *The Leprechauns of Software Engineering.* o. O.: Leanpub.

Chapman, C., & Ward, S. (2011). *How to Manage Project Opportunity and Risk* (3. Ausg.). Orchester: John Wiley & Sons Ltd.

Christensen, C. M. (2000). *The Innovator's Dilemma: When New Technologies Cause Great Firms to Fail.* o. O.: Harvard Business Review Press.

Drucker, P. F. (2009). *Innovation and Entrepreneurship* (Reprint 2009 Ausg.). o. O.: HarperCollins e-books.

Ephesos, H. v. (kein Datum).

Festtag, S. (2014). *Umgang mit Risiken* (1. Ausg.). Berlin: Beuth Verlag GmbH.

Gassmann, O., & Kobe, C. (2006). *Management von Innovation und Risiko* (2. Ausg.). Berlin: Springer-Verlag Berlin Heidelberg.

Grawe, C. (2016). Die betriebswirtschaftliche Abbildung von Innovationsprojekten. *Controlling*(2), 133-135.

Hammond, J. S. (05. 05 2010). *The Forrester Wave*. Abgerufen am 24. 10 2015 von https://www.forrester.com/The+Forrester+Wave+Agile+Development+Management+Tools+Q2+2010/fulltext/-/E-RES48153

Humble, J., & Farley, D. (2010). *Continuous Delivery: Reliable Software Releases Through Build, Test, and Deployment Automation.* Upper Saddle River: Addison Wesley.

ISO/IEC. (15. 04 2011). ISO/IEC 20000-1. o. O.

ISO/IEC. (15. 02 2012). ISO/IEC 20000-2. o. O.

itSMF e. V. (2007). *ITIL in der öffentlichen Verwaltung.* Düsseldorf: Symposion Publishing GmbH.

Kairies, P. (2004). *Professionelles Produkt-Management für die Investitionsgüterindustrie.* Renningen: expert-Verlag.

KBSt. (April 2007). *cio.bund.de.* Abgerufen am 6. Juli 2016 von http://www.cio.bund.de/cae/servlet/contentblob/79996/publicationFile/4113/studie\_itil\_erfahrungsberichte\_download.pdf

Kelly, J. (1982). *Scientific Management, Job Redesign and Work Performance.* London: Academic Press.

Kim, D. (2015). *The State of Scrum.* Abgerufen am 24. Juni 2016 von https://www.scrumalliance.org/scrum/media/scrumalliancemedia/files%20and%20pdfs/state%20of%20scrum/scrum-alliance-state-of-scrum-2015.pdf

Kohavi, R., & Longbotham, R. (25. April 2015). *Cross entries for A/B Tests, Split Tests, and Randomized Experiments.* Abgerufen am 1. Mai 2016 von http://bit.ly/onlineControlledExperiments

Lange, B., & Diercks, J. (04 2015). In 28 Artikeln: So gelingen agile IT-Projekte. *Agil Software entwickeln*, S. 8-12.

Laser, T. (2. November 2012). *10 Mythen und Missverständnisse um ITIL*. Abgerufen am 6. Juli 2016 von http://de.slideshare.net/TorstenLaser/10-mythen-und-missverstndnisse-um-itil

Lennertz, D. (2006). *Produktmanagement: Planung, Entwicklung und Vermarktung* (1. Auflage Ausg.). Frankfurt am Main: Frankfurter Allgemeine Buch.

Lienemann, G. (2006). *ITIL - Change Management* (1. Ausg.). Hannover: Heise Zeitschriften Verlag GmbH & Co KG.

Newcombe, T. (31. Januar 2005). *gt*. Abgerufen am 6. Juli 2016 von http://www.govtech.com/magazines/pcio/100560679.html

Peasley, J., & Fletcher, J. (5-8. April 2005). *IT service management - and beyond.* Abgerufen am 6. Juli 2016 von minerva.mq.edu.au: http://hdl.handle.net/1959.14/9465

Peffers, K., Tuunanen, T., Gengler, C. E., Rossi, M., Hui, W., Virtanen, V., et al. (2006). The Design Science Research Process: A Model For Producing And Presenting Information Systems Research. *DESRIST* (S. 84-106). Claremont: CGU.

Pichler, R. (2014). *Agiles Produktmanagement mit Scrum.* Heidelberg: dpunkt.

Pink Elephant. (2005). *JDCMG.* Abgerufen am 6. Juli 2016 von http://jdcmg.isc.ucsb.edu/docs/ITIL/ITIL-Top%20Things%20Managers%20must%20Know%20when%20Implementing%20ITIL.pdf

Proske, D. (2004). *Katalog der Risiken.* Dresden: Eigenverlag Dirk Proske.

Puppet. (22. Juni 2016). *puppet.com.* Abgerufen am 8. Juli 2016 von https://puppet.com/resources/white-paper/2016-state-devops-report

Roock, S., & Wolf, H. (2016). *Scrum verstehen und erfolgreich einsetzen.* Heidelberg: dpunkt.verlag.

Schumpeter, J. A. (1997). *Theorie der wirtschaftlichen Entwicklung: Eine Untersuchung über Unternehmergewinn, Kapital, Kredit, Zins und den Konjunkturzyklus* (9. Auflage Ausg.). Berlin: Duncker & Humblot.

Simon, F. (2010). *Qualitäts-Risiko-Management.* Berlin: Logos Verlag Berlin GmbH.

Winniford, M., Conger, S., & Erickson-Harris, L. (2009). Confusion in the Ranks: IT Service Management Practice and Terminology. *Information Systems Management*(26), 153-163.

Wonke-Stehle, J. (2012). ITIL in Bibliotheken. *Berliner Handreichungen zur Bibliotheks- und Informationswissenschaft*(331), 0-92.

Erklärung

**Name:** Lohr, Steve

**Matrikel-Nr.:** 9066454

**Fach:** Wirtschaftsinformatik

Ich erkläre, dass ich die Masterarbeit selbstständig und ohne unzulässige Inanspruchnahme Dritter verfasst habe. Ich habe dabei nur die angegebenen Quellen und Hilfsmittel verwendet und die aus diesen wörtlich, inhaltlich oder sinngemäß entnommenen Stellen als solche den wissenschaftlichen Anforderungen entsprechend kenntlich gemacht. Die Versicherung selbstständiger Arbeit gilt auch für Zeichnungen, Skizzen oder graphische Darstellungen. Die Arbeit wurde bisher in gleicher oder ähnlicher Form weder derselben noch einer anderen Prüfungsbehörde vorgelegt und auch noch nicht veröffentlicht. Mit der Abgabe der elektronischen Fassung der endgültigen Version der Arbeit nehme ich zur Kenntnis, dass diese mit Hilfe eines Plagiatserkennungsdienstes auf enthaltene Plagiate überprüft und ausschließlich für Prüfungszwecke gespeichert wird.

Dresden,   
Ort, Datum Unterschrift