universität wien

MASTER THESIS

Titel der Master Thesis / Title of the Master‘s Thesis

„Der Umgang mit Requirements-Engineering an wissenschaftlichen Bibliotheken

State of the Art und Zukunftsperspektiven“

verfasst von / submitted by

Mag. Dr. Susanne Blumesberger

angestrebter akademischer Grad / in partial fulfilment of the requirements for the degree of

Master of Science (Library and Information Studies) (MSc)

Wien, 2017 / Vienna 2017

Studienkennzahl lt. Studienblatt / Postgraduate programme code as it appears on the student record sheet:	A 992 600
Universitätslehrgang lt. Studienblatt / Postgraduate programme as it appears on the student record sheet:	Library and Information Studies (MSc)
Betreut von / Supervisor:	Ao. Univ. Prof. Dipl.-Ing. Dr. techn. Andreas Rauber

Abstract

Der Umgang mit Requirements-Engineering an wissenschaftlichen Bibliotheken. State of the Art und Zukunftsperspektiven

Die Arbeit beschäftigt sich mit dem Umgang mit technischen Anforderungen an wissenschaftlichen Bibliotheken. Während in der Wirtschaft Requirements-Engineering bereits seit langer Zeit eingesetzt wird um einen reibungslosen Ablauf von IT-Projekten zu garantieren, beschäftigen sich Bibliotheken bisher kaum mit diesem Thema, obwohl auch hier die Zusammenarbeit mit dem IT-Bereich immer wichtiger wird, z.B. durch die Bereitstellung von forschungsunterstützenden Services. Requirements-Engineers sind wie VermittlerInnen zwischen den AuftraggeberInnen und den IT-KollegInnen zu sehen. Sie bereiten zunächst das Projekt vor, nehmen Kontakt mit allen Stakeholdern auf, ermitteln die Anforderungen, formulieren und prüfen sie und begleiten den gesamten Prozess bis zur Umsetzung. Ein Ziel der Arbeit war anhand ausgewählter Literatur einen Überblick über den professionelle Umgang mit Anforderungen zu geben, ein Jobprofil für Requirements-Engineers an Bibliotheken zu entwerfen und ein Stimmungsbild, wie derzeit an österreichischen wissenschaftlichen Institutionen mit Anforderungen umgegangen wird, zu zeigen. Dazu wurden VertreterInnen – BibliothekarInnen und TechnikerInnen – aus insgesamt 13 Institutionen befragt. Das Ergebnis bestätigte die These, dass Requirements-Engineering in Bibliotheken noch weitgehend unbekannt ist, bzw. nicht eingesetzt wird. Allerdings werden bereits zahlreiche Projekte zwischen verschiedenen Abteilungen von Bibliotheken und IT-Abteilungen, bzw. Firmen erfolgreich durchgeführt. Beklagt wurden sehr oft die mangelhafte Ausbildung des Personals im technischen Bereich und die Tatsache, dass für Projekte allgemein zu wenig Geld und Personal vorhanden wären. Würde man hier mehr investieren, könnten sich Bibliotheken auf lange Sicht Zeit und Kosten sparen. Eine kurze Übersicht, was in Zukunft beachtet werden sollte, sowie die Empfehlung Requirements-Engineering in den Lehrplan der Bibliotheksausbildung aufzunehmen runden die Arbeit ab.

Requirements Engineering in academic libraries. State of the Art and Future Perspectives

This scientific work deals with the way technical requirements are handled in academic libraries. Whereas in the business community the Requirements-Engineering (RE) process has already been in use for a considerable length of time in order to guarantee smooth procedures in IT-projects, libraries, on the other hand, have scarcely dealt with the topic, despite the fact that cooperation with the IT sector is becoming ever more important, as, for example, by providing research supportive services. Thus Requirements Engineers can be regarded as the agents between the clients and the IT colleagues. They first devise the project, then get in touch with all the stake-holders, determine the requirements, formulate them and verify them and then accompany the entire process to the point of implementation. One of the aims of this work is, based on selected literature, to provide an overview of the professional handling of the requirements, to draft a job profile for Requirements Engineers in libraries, and to capture the mood of how the requirements are currently being handled at scientific institutions in Austria. To that end, interviews were carried out among representatives – librarians and technicians – at a total of 13 institutions. The result confirmed the thesis that Requirements Engineering remains uncharted territory in libraries or is simply not being implemented. It has to be said though that numerous projects are successfully being carried out among various library departments on the one hand and IT departments or companies on the other. A frequent complaint is the lack of training among staff members in technical fields and the fact that there is, generally speaking, insufficient funding and personnel. If more funds were to be invested in this field, the libraries could in the long term save time and money. This scientific paper concludes with a brief glance at what has to be given attention to in future and the recommendation that Requirements Engineering be added to the curriculum of Library Education.

INHALTSVERZEICHNIS

Vorbemerkungen 7

Untersuchungsgegenstand 9

Zu den Quellen 10

1. REQUIREMENTS-ENGINEERING UND REQUIREMENTS-MANAGEMENT 15

1.1 Begriffsdefinition und Aufgaben 15

1.2 Ausbildungsmöglichkeiten 20

1.3 Einbettung und Abgrenzung des Themas 21

1.4 Agile Vorgehensweisen und RE 21

1.5 Gründe für Requirements-Engineering 28

1.6 Grundlagen für ein gut funktionierendes Requirements-Engineering 30

2. PHASEN DES REQUIREMENTS-ENGINEERINGS 32

2.1 Glossar 33

2.2 Der RE-Leitfaden 34

2.3 Definition der Grenzen des Systems 35

2.4 Beschreibung der Ausgangssituation 37

2.5 Das Ermitteln von Anforderungen 39
2.5.1 Definition von Anforderungen 39
2.5.2 Arten an Anforderungen 42
2.5.3 Quellen der Anforderungen 52
2.5.4 Einbinden der Stakeholder 54
2.5.5 Ermittlungstechniken von Anforderungen 58

2.6 Attributierung von Anforderungen 73

2.7 Sichten auf Anforderungen 77

2.8 Priorisieren von Anforderungen 77

2.9 Verfolgbarkeit von Anforderungen 79

2.10 Strukturierung von Anforderungen 82

2.11 Spezifizieren von Anforderungen Requirements Specification **84**
2.11.1 Aufbau von Spezifikationen 84
2.11.2 Das Lasten- und Pflichtenheft 88

2.12 Dokumentation von Anforderungen **91**
2.12.1 Dokumentation von funktionalen Anforderungen 91
2.12.2 Dokumentation von nicht-funktionalen Anforderungen 98
2.12.3 Die Rolle der Sprache 106
2.12.4 Perspektiven der Anforderungen 108

2.13 Qualitätssicherung **112**
2.13.1 Qualitätsaspekte der Anforderungen 120
2.13.2 Qualitätssicherung des Anforderungsmanagements 123
2.13.3 Techniken zur Überprüfung der Anforderungen 123

2.14 Konflikte erkennen und lösen **131**

2.15 Change-Management – Verwaltung von Anforderungsänderungen **134**

2.16 Das Release **140**

2.17 Werkzeuge für das Anforderungsmanagement **140**

3. REQUIREMENTS-ENGINEERING AN BIBLIOTHEKEN **144**

3.1 Die Aufgaben von Requirements-Engineers in Bibliotheken **146**

3.2 Anforderungsprofil an Requirements-Engineers in Bibliotheken **148**

3.3 Einführen des Requirements-Engineerings in Institutionen **151**

4. EMPIRISCHER TEIL **155**

4.1 Zur Umfrage **155**

4.2 Die Ergebnisse der Umfrage **157**

4.3 Interpretation der Ergebnisse **182**
4.3.1 Zusammenarbeit zwischen der Bibliothek und der IT-Abteilung, bzw. IT-Firmen 182
4.3.2 Der Umgang mit Anforderungen im technischen Bereich 183
4.3.3 Das Wissen und der Einsatz von RE an wissenschaftlichen Bibliotheken 185
4.3.4 Subjektiver Eindruck vom Umgang mit Anforderungen 187

5. ZUSAMMENFASSUNG DER ERGEBNISSE **189**
6. GLOSSAR **193**
7. ABKÜRZUNGEN **196**
8. LITERATUR **197**

Vorbemerkungen

„Die Bedeutung des Requirements Engineering (RE) für die erfolgreiche, den Kunden zufriedenstellende Entwicklung von Systemen ist mittlerweile kaum mehr zu übersehen. In der Praxis ist es üblich, einen entsprechenden Aufwand für das Requirements Engineering einzuplanen.“ (Pohl/Rupp 2015, 1)

Inwieweit diese Aussage, dass Requirements-Engineering (RE), also der Weg von der Projektidee bis zu einem vollständigen Satz von Anforderungen, aktuell auch für wissenschaftliche Bibliotheken zutrifft, soll in der vorliegenden Arbeit diskutiert werden.

Die Aufgaben der Bibliotheken haben sich in den letzten Jahren stark geändert, es werden immer mehr Serviceleistungen angeboten, bei denen technische Dienste grundlegend sind und damit wird die Zusammenarbeit mit IT-Abteilungen oder IT-Firmen immer wichtiger. Bibliothekarisches Wissen im herkömmlichen Sinn reicht nicht mehr aus, um allen Herausforderungen, denen sich moderne wissenschaftliche Bibliotheken heute stellen, noch erfolgreich begegnen zu können. Forschungsunterstützende Services, die bereits in vielen Bibliotheken angeboten werden, wie etwa der Betrieb eines Repositoriums oder der professionelle Umgang mit Forschungsdaten, benötigen den Einsatz von Computertechnologie. Dazu bedarf es SpezialistInnen aus dem IT-Bereich, die mit dem Bibliothekspersonal eng zusammenarbeiten. Genau hier liegt die Herausforderung für wissenschaftliche Bibliotheken. BibliotheksexpertInnen treffen auf IT-ExpertInnen und sollen gemeinsame Projekte durchführen um den WissenschafterInnen technische Lösungen für Recherche, Dokumentation, Archivierung von Forschungsdaten usw. anbieten zu können. In der Praxis treffen hier nicht nur Personen aufeinander, die unterschiedliches Wissen besitzen, sondern auch oft eine unterschiedliche Sicht auf die diversen Aufgaben haben, diese auch mit anderen Werkzeugen beschreiben, bearbeiten und unterschiedlich benennen. Aus diesem Grund kann es zu Missverständnissen kommen, zu Umwegen und zu Verzögerungen von Projekten, die nicht nur Geld kosten, sondern auch das Arbeitsklima auf beiden Seiten belasten. „Etwa 60% der Fehler in Systementwicklungsprojekten entstehen bereits beim Umgang mit Anforderungen“. (Pohl/Rupp 2015, 1) Je später man diese Fehler entdeckt, desto höher steigt die

finanzielle Belastung, denn mit dem Projektfortschritt potenzieren sich die Fehler und dadurch auch die Kosten. Dieser Effekt ist umso stärker, je länger die einzelnen Entwicklungszyklen dauern und je stärker die einzelnen Phasen voneinander getrennt sind. (siehe auch Rupp/Sophisten 2014, 15). Für die Behebung eines Fehlers, der erst während des Programmierens offensichtlich wird, muss ein um ca. um den Faktor 20 höherer Aufwand berechnet werden, als wäre dieser Fehler bereits in der Phase des Requirements Engineering aufgefallen. (siehe Pohl/Rupp 2015, 2)
Je ähnlicher sich alle Beteiligten sind, je ähnlicher der Wissensstand, das verwendete Vokabular, die Ausbildung und der kulturelle Hintergrund sind, desto einfacher ist der Austausch von Informationen und deshalb auch die Verständigung im Allgemeinen, vor allem aber auch über Anforderungen im IT-Bereich. Dieser Idealfall ist jedoch meist bei den am Projekt beteiligten Personen, den Stakeholdern, nicht gegeben, da sich die Ausbildung, die Berufserfahrung und der Umgang mit Begriffen bei Personen die im IT-Bereich arbeiten und jenen die im Bibliotheksbereich tätig sind, stark unterscheiden. Dazu kommt noch, dass vielfach nicht persönlich kommuniziert wird. Kommunikation funktioniert jedoch bei persönlichen Kontakten meist besser als per Mail oder über eine andere Kommunikationsplattform, da sie stark redundant ist. Außer dem gesprochenen Wort hat man beim Gespräch noch Mimik und Gestik zur Verfügung. Es kommt zwischendurch zu Rückkoppelungen, Missverständnisse lassen sich sofort beheben. Bei schriftlicher Sprache fällt die Redundanz weg, man ist nur auf Worte angewiesen und diese können falsch oder anders interpretiert werden, als der Sender sie gemeint hat. Ein weiterer Faktor kommt noch erschwerend hinzu. Informationen werden auch im Alltag meist verkürzt weitergegeben. Eine Person nimmt Mitteilungen durch ihr Wissen gefiltert wahr und gibt Informationen in einer anderen Art und Weise weiter, als eine Person mit einem anderen Hintergrundwissen dies tun würde. Wissen kann auch nie im Gesamten weitergegeben werden sondern immer nur vereinfacht. In dieser Vereinfachung liegen jedoch weitere Fehlerquellen. Es wird beispielsweise ein Vorwissen vorausgesetzt, das die KommunikationspartnerInnen gar nicht besitzen, bzw. werden Informationen anderes interpretiert, weil das Vorwissen der einen Person von dem der anderen Person abweicht.

Untersuchungsgegenstand

Die Arbeit geht von der These aus, dass der professionelle Umgang mit technischen Anforderungen an den Bibliotheken noch nicht in den alltäglichen Bibliotheksbetrieb übernommen wurde, dass also RE im Regelfall noch nicht angewendet wird.
Wie BibliothekarInnen und IT-SpezialistInnen an wissenschaftlichen Institutionen derzeit Projekte zwischen der Bibliothek und der IT-Abteilung, bzw. IT-Firmen handhaben, wird im empirischen Teil mittels Fragebogen untersucht. Ziel dieser persönlichen, bzw. telefonischen Befragung ist es auch, herauszufinden, ob die befragten Personen mit der derzeitigen Situation zufrieden sind und für wie relevant sie den Einsatz von RE in der Zukunft halten. Um die oben vorgestellte These prüfen zu können, wurden 13 Institutionen in die Befragung einbezogen. Es wurde nicht nur konkret nach dem Bekanntheitsgrad, bzw. dem Einsatz von RE im alltäglichen Arbeitsprozess gefragt, sondern auch nach dem Umgang mit IT-Projekten insgesamt. Um einen Überblick über die derzeitige Situation zu erhalten, wurde auch erhoben, wie viele MitarbeiterInnen jeweils an Projekten beteiligt sind und wie die personelle Zusammensetzung jeweils aussieht. Auch subjektive Verbesserungsvorschläge und Wünsche konnten im Rahmen der Befragung genannt werden. Das daraus entstehende Stimmungsbild ist das erste Ergebnis der vorliegenden Arbeit.
Ein zweites Ergebnis soll eine kurze Einführung in RE und Requirements-Management (RM) sein, die wissenschaftlichen BibliothekarInnen einen ersten Einblick in das Thema bieten soll. Aus diesem Grund ist der theoretische Teil der Arbeit relativ umfangreich. Dieser kann und soll jedoch einschlägige aktuelle Handbücher, die bereits vorliegen, nicht ersetzen, jedoch LeserInnen aus dem Bibliotheksbereich auf das Thema neugierig und aufmerksam machen. Eine kurze Kritik der verwendeten Quellen soll den Überblick über weiterführende Literatur vereinfachen. Ein Glossar am Ende der Arbeit fasst alle relevanten Begriffe, die in der Arbeit verwendet werden, übersichtlich zusammen.
Ein drittes Ergebnis dieser Arbeit ist die Ausarbeitung eines Jobprofils eines/einer Requirements-Engineerin, der/die in Zukunft IT-Projekte professionell begleiten und unterstützen könnte und den jeweiligen Institutionen damit Zeit, Kosten und

Missstimmungen sparen würde. Zum Schluss finden sich Überlegungen zum Einsatz von RE in Bibliotheken und Empfehlungen RE in die Ausbildung zu integrieren.

Zu den Quellen

Hier sind die wichtigsten Quellen angeführt, die für die vorliegende Arbeit herangezogen wurden. Diese werden in chronologischer Abfolge nach ihren jeweiligen Erscheinungsdaten kurz charakterisiert.

Partsch, Helmut (Partsch 1991): Requirements Engineering. München, Wien: R. Oldenbourg Verlag 1991 (Handbuch der Informatik; Bd. 5.5.).

Die wissenschaftliche Literatur zum Thema RE, bzw. RM ist noch recht überschaubar. „Requirements Engineering“ von Helmut Partsch aus dem Jahr 1991 hat sich zum Ziel gesetzt, Grundwissen zu vermitteln und vor allem Interesse für das Thema zu wecken, es soll aber kein „Manual sein, das es erlaubt, eine der vorgestellten Methoden unmittelbar in der industriellen Praxis anzuwenden.“ (Partsch 1991, S. 22)
Obwohl das 335 Seiten umfassende Werk nicht mehr auf dem neuesten Stand der Technik ist, handelt es sich doch um eine gut lesbare und verständliche aufbereitete Einführung in das Thema. Man kann hier auch die klassischen Ansätze des RE nachlesen.

Neumann, Horst A. (Neumann 1998): Objektorientierte Softwareentwicklung mit der Unified Modeling Language (UML). München, Wien: Carl Hanser Verlag 1998.

Das Werk aus dem Jahr 1998 gibt auf 452 Seiten einen umfassenden Überblick über die Möglichkeit, Anforderungen grafisch zu notieren. Viele Beispiele und Abbildungen erleichtern den Umgang mit UML. „Das Buch richtet sich an alle, die eine sachliche, systematische, umfassende und verständliche Darstellung der objektorientierten Entwicklung von Softwaresystemen mit der UML suchen, weil sie das mächtige Potential besser verstehen und sich für die eigene Nutzung erarbeiten wollen“ . (Neumann 1998, VI) Hauptsächlich an Studierende der Informatik, Mathematik und Datenverarbeitung gerichtet, ist es auch für EntwicklerInnen, ProjektmanagerInnen und

Beauftragte der Qualitätssicherung gedacht. Die LeserInnen sollten jedoch bereits Erfahrung mit Software-Entwicklung haben.
Eine eingehende Beschäftigung mit der UML würde den Rahmen der Arbeit sprengen, deshalb werden LeserInnen, die sich in dieses Thema vertiefen möchten, auf diese Publikation und auf Rupp/Queins/Sohisten 2012 verwiesen.

Heche, Dirk; Oliver, Ollech (Heche/Ollech 2006): Projektbegleitendes Anforderungsmanagement: unnötige Kosten und Ärger vermeiden. In: ProjektMagazin. Das Fachportal für Projektmanagement. Ausgabe 21/2006.

Dirk Heche und Oliver Ollech geben in ihrem sechsseitigen Beitrag einen knappen aber informativen Ersteinblick in das Thema und fassen die Argumente, warum man mit Anforderungen sorgsam umgehen sollte, übersichtlich zusammen.

Moser, Julia (Moser 2007): Integration von Anforderungsverhandlungen in den Rescue Requirements Engineering Prozess. Masterarbeit, Universität Wien 2007.

Die Masterarbeit von Julia Moser thematisiert eine in London entwickelte Methode zur Erhebung von Anforderungen, den Rescue-Prozess, der in fünf Stufen und vier parallelen Streams abläuft. Ständige Checks garantieren präzise Anforderungen. Auf 87 Seiten werden hier die Grundlagen dieser Technik vermittelt.

Rupp, Chris; Christof Geißel (Rupp/Geißel 2007): Nichtfunktionale Anforderungen strukturiert erfassen und wieder verwenden. In: ProjektMagazin. Das Fachportal für Projektmanagement. Ausgabe 8/2007.

Chris Rupp und Christof Geißel beschäftigen sich in ihrem neun Seiten umfassenden Beitrag mit dem Umgang von nicht-funktionalen Anforderungen, also all jenen Forderungen, die die Qualität des Produktes bestimmen, bzw. alles, was das System benutzerfreundlich macht. Durch die Übersichtlichkeit und die anschaulichen Beispiele ermöglicht der Text einen guten Einstieg in das Thema.

Rupp, Christine; Stefan Queins & die SOPHISten (Rupp/Queins/Sophisten 2012): UML2 glasklar. Praxiswissen für die UML-Modellierung. 4. akt. und erw. Auflage, München: Carl Hanser 2012.

Der Band, der 560 Seiten umfasst, soll nicht nur fundierte Kenntnisse der UML vermitteln, sondern auch als Nachschlagewerk dienen. Eine übersichtliche Darstellung der Notationsmittel, zahlreiche Zusammenfassungen, Überblicke, Diagramme, Querverweise und Beispiele ermöglichen auch Laien einen einfachen Einstieg in das Thema. Die wichtigsten Notationselemente sind kompakt zusammengefasst und können aus dem Buch herausgetrennt werden. Ein Register ermöglicht das schnelle Auffinden bestimmter Themen. Ergänzungen zum Buch findet man unter http://www.sophist.de/uml/uml-2-glasklar/

Niebisch, Thomas (Niebisch 2013): Anforderungsmanagement in sieben Tagen. Der Weg vom Wunsch zur Konzeption. Berlin, Heidelberg: Springer Verlag 2013.

Das 243 Seiten umfassende Werk ist für folgende Adressaten gedacht: „für Fachexperten, Führungskräfte, Projektmanager und andere in IT-Projekte involvierte Rollen, die sich mit Anforderungen auseinandersetzen müssen, ohne selbst Spezialisten des Anforderungsmanagements zu sein." (Niebisch 2013, 1) Dieses Buch ist für alle EinsteigerInnen sehr zu empfehlen, da es im Bereich Anforderungsmanagement übersichtlich die einzelnen Schritte, aber auch mögliche Probleme und Hindernisse aufzeigt. Der Autor selbst meint:

„Das Buch behandelt alle Themen rund um das Anforderungsmanagement, die mir permanent, wenn auch in unterschiedlicher Intensität, in meinen IT-Projekten begegnet sind. Die daraus resultierenden, sich wiederholenden Gedankengänge habe ich niedergeschrieben in der Hoffnung, dass diese Ihnen in ähnlichen Situationen nützlich sein werden". (Niebisch 2013, 8)

Anhand zweier fiktiver Personen, die sich eben selbst ins Thema einarbeiten, können auch die LeserInnen sämtliche Gedankengänge mitverfolgen.

Rupp, Chris & die SOPHISten (Rupp/Sophisten 2014): Requirements-Engineering und –Management. Aus der Praxis von klassisch bis agil. 6., aktualisierte und erweiterte Auflage, München: Carl Hanser Verlag 2014.

Sehr zu empfehlen ist das 2014 in 6. Auflage erschienene Handbuch. Es führt nicht nur in logisch und übersichtlich dargestellten Schritten durch den gesamten Prozess des RE, sondern zeigt anhand von Beispielen aus dem Bibliotheksbetrieb sehr anschaulich und

zum Teil auch mit humorvollen Anmerkungen versehen, wie man mit RE umgehen kann. Das Standardwerk eignet sich auch um im Alltag praxisnahe Beispiele und Erklärungen zu finden. Sehr hilfreich sind auch die im Internet angebotenen zusätzlichen Materialien, Erläuterungen und Hinweise. Unter https://www.sophist.de/start/ kann man sich nicht nur über das Thema RE informieren, sondern sich auch praktische Tipps holen und z.B. Checklisten für die eigene Anwendung downloaden.

Kulke, Matthias (Kulke 2014): Wie Sie die natürliche Sprache bändigen. Die SOPHIST-Satzschablone: Funktionale Anforderungen präzise formulieren. In: ProjektMagazin. Das Fachportal für Projektmanagement. Ausgabe 15/2014.

Ebenfalls 2014 erschien der Beitrag von Matthias Kulke „Wie Sie die natürliche Sprache bändigen. Die SOPHIST-Satzschablone: Funktionale Anforderungen präzise formulieren“ im „ProjektMagazin. Das Fachportal für Projektmanagement“, in der Ausgabe 15/2014. Ausgehend vom Standardwerk von Rupp &Sophisten geht es darum, Standards für das Verfassen von Anforderungen zu verwenden. Auf 11 Seiten gelingt es Kulke übersichtlich darzustellen, warum Satzschablonen sinnvoll sind und wie sie optimal gestaltet werden.

Zastrow, Bettina; Elisabeth Wagner (Zastrow/Wagner 2014): Anforderungen an Softwareprodukte klar definieren. Requirements Engineering für Projektleiter Teil 1: Leistungsumfang und Rahmenbedingungen festlegen. In: ProjektMagazin. Das Fachportal für Projektmanagement. Ausgabe 23/2014.

Zastrow, Bettina; Elisabeth Wagner (Zastrow/Wagner 2015): Anforderungen an Softwareprodukte klar definieren. Requirements Engineering für Projektleiter Teil 2: Mit Use Cases den Funktionsumfang beschreiben. In: ProjektMagazin. Das Fachportal für Projektmanagement. Ausgabe 24/2015

Auf übersichtlichen 12 Seiten wird vor allem das Thema Leistungsumfang und Rahmenbedingungen festlegen behandelt. Der zweite Beitrag beschäftigt sich auf neun Seiten mit der Anleitung „Mit Use Cases den Funktionsumfang beschreiben“. Beide Beiträge sind auch für EinsteigerInnen in das Thema interessant

V-Modell XT. Das deutsche Referenzmodell für Systementwicklungsprojekte Version: 2.0. (V-Modell XT 2015) Herausgegeben vom Verein zur Weiterentwicklung des V-Modell XT e.V. (Weit e.V.) München 2015.

„Das V-Modell®XT ist ein Vorgehensmodell zum Planen und Durchführen von Systementwicklungsprojekten. Es ist ein Hilfsmittel, um allen Projektbeteiligten ein einheitliches und konsistentes Bild vom gemeinsamen Vorgehen zu vermitteln“. (V-Modell XT 2015, 6). Es soll dazu dienen die Kommunikation und die Qualität der Produkte zu verbessern und Kosten und Risiken zu senken. Das 300 Seiten umfassende Werk wendet sich an alle Projektbeteiligten, vor allem an jene, die in der Qualitätssicherung tätig sind. Es ist vor allem dann hilfreich, wenn man sich schon mit den Grundlagen von RE vertraut gemacht hat.

Pohl, Klaus; Chris Rupp (Pohl/Rupp 2015): Basiswissen Requirements Engineering. Aus- und Weiterbildung zum „Certified Professional for Requirements Engineering“. 4. Auflage, Heidelberg: dpunkt.verlag GmbH 2015.

Für alle, die einen kompakten aber fundierten Überblick suchen ist das 2015 in vierter Auflage erschienene Buch von Klaus Pohl und Chris Rupp, eigentlich konzipiert um auf die Zertifizierungsprüfung zum „Certified professional for Requirements Engineering“ vorzubereiten, ideal. Es bietet auf 171 Seiten einen gut strukturierten und mit zahlreichen Beispielen versehenen Überblick zum Thema.

Zastrow, Bettina; Elisabeth Wagner (Zastrow/Wagner 2016): Anforderungen an Softwareprodukte klar definieren Requirements Engineering für Projektleiter Teil 3: Anforderungen für die Software-Entwicklung ableiten. In: ProjektMagazin. Das Fachportal für Projektmanagement. Ausgabe 1/2016.

Auch in diesem Beitrag gelingt es den beiden Autorinnen einen praktischen Einblick in das Thema Umgang mit Anforderungen zugeben. Für eine Erstinformation ist dieser Text deshalb sehr hilfreich.

1. Requirements-Engineering und Requirements-Management

1.1 Begriffsdefinition und Aufgaben

Oft werden die Begriffe „Requirements-Engineering“ (RE) und „Requirements-Management“ (RM) unscharf verwendet. Unter RE versteht man das Aufstellen von Anforderungen. Darunter fallen die Bereiche Anforderungserhebung (Elicitation), Analyse, Dokumentation, bzw. Kommunikation, Prüfung (Validation) und Abstimmung (Negotiation). Unter RE wird der Weg von der Projektidee bis zu einem vollständigen Satz von Anforderungen verstanden. (siehe Moser 2007, 11)
Helmut Partsch definiert RE als Teilbereich der Informatik, der manchem Informatiker fremd oder bestenfalls nur vage bekannt ist. (Partsch 1991, 8) Warum das so ist, beantwortet Partsch folgendermaßen:

„Dies ist möglicherweise darauf zurückzuführen, daß viele Aspekte des Requirements Engineering eigentlich unmittelbar, allein mit gesundem Menschenverstand nachvollzogen werden können, so daß kein echtes, den Akademiker simulierendes Problembewußtsein geweckt wird. Hinzu kommt, daß die existierenden Lösungsansätze keine tiefgreifenden theoretischen Kenntnisse erfordern, und so der Reiz der Herausforderung‘ fehlt.“ (ebd.),

aber, so Partsch weiter:

„Requirements-Engineering ist viel schwieriger, als es oft den Anschein hat. Außerdem nimmt es im Rahmen der Softwareentwicklung eine Schlüsselposition ein. Viele fehlgeschlagene Projekte, unzufriedene Kunden oder frustrierte Benutzer sind letztlich auf eine Fehleinschätzung der Bedeutung des Requirements Engineering zurückzuführen.“ (ebd.)

Partsch unterscheidet beim RE zwei verwandte, aber doch zu unterscheidenden Bedeutungsvarianten: „Das Wort steht einmal für alle konkreten Aktivitäten am Beginn eines Softwareprojekts, die auf eine Präzisierung der Problemstellung abzielen, aber ebenso auch für eine ganze Teildisziplin der Informatik.“ (Partsch 1991, 25f) Er spricht von RE im weiteren Sinn „Als Disziplin zur systematischen Entwicklung einer vollständigen, konsistenten und eindeutigen Spezifikation, in der beschrieben wird, was ein Softwareprodukt tun soll (aber nicht wie) und die als Grundlage für Vereinbarungen zwischen allen Betroffenen dienen kann.“ (ebd., 26) Im engeren Sinn umfasst RE laut Partsch die Problemanalyse, die Anforderungsdefinition und die Anforderungsanalyse.

(ebd., 27f) Die Überprüfung und Einhaltung der Anforderungen sowie die Versionsverwaltung von Anforderungen werden bei dieser Definition ausgeklammert.

Klaus Pohl und Chris Rupp verstehen unter RE

„den systematischen und disziplinierten Ansatz zur Spezifikation und zum Management von Anforderungen mit folgenden Zielen:
(1) Die relevanten Anforderungen zu kennen, Konsens unter den Stakeholdern über die Anforderungen herzustellen, die Anforderungen konform zu vorgegebenen Standards zu dokumentieren und die Anforderungen systematisch zu managen.
(2) Die Wünsche und Bedürfnisse der Stakeholder zu verstehen, zu dokumentieren sowie die Anforderungen zu spezifizieren und zu managen, um das Risiko zu minimieren, dass das System nicht den Wünschen und Bedürfnissen der Stakeholder entspricht“. (Pohl/Rupp 2015, 4)

Ein Requirements-Engineer wird oft als „Mittler zwischen den Welten“ bezeichnet und in der Literatur auch oft „Systemanalytiker“, „Anforderungsanalytiker“ oder „Business Analyst“ genannt (Rupp/Sophisten 2014, 11)

„Unter RM wird der Prozess zum Managen der Anforderung verstanden, es umfasst Maßnahmen, welche die Anforderungsanalyse und die weitere Verwendung der Anforderungen unterstützen.“ (Rupp/Sophisten 2014, 368).
Somit ist RM ein Teilgebiet von RE.

RM ist eine der Haupttätigkeiten des Requirements-Engineering, es benötigt viel Zeit, da damit einige Aufgaben verbunden sind. Die Verwaltung von Anforderungen umfasst mehrere Techniken:

- Attributierung von Anforderungen
- Priorisierung von Anforderungen
- Verfolgbarkeit von Anforderungen
- Versionierung von Anforderungen
- Änderungsmanagement von Anforderungen
- Messung von Anforderungen (siehe auch Pohl/Rupp 2015, 119ff)

Beim RM ist es wichtig mitzudenken, dass sich Anforderungen ändern, was bei einer großen Anzahl an Anforderungen zu einem beträchtlichen Aufwand führen kann. Außerdem muss berücksichtigt werden, dass Anforderungen weiterverwendet werden. Das Dokumentieren von Anforderungen ist also kein Selbstzweck, sondern die

Dokumentation wird von anderen Personen gelesen und entsprechend genutzt, unter anderem auch von den Testpersonen. Man hat es dabei meist mit einer großen Menge an Informationen zu tun, die entsprechend übersichtlich verwaltet werden muss.
Für erfolgreiches RM ist es auch wichtig, dass sämtliche Anforderungen eigene eindeutige und permanente Identifikationsnummern haben. Das bedeutet, dass diese Nummer, auch wenn die ursprüngliche Anforderung gelöscht wurde, kein zweites Mal vergeben werden darf. Den Aufbau betreffend empfiehlt Rupp aufgrund der besseren Lesbarkeit nicht mehr als fünf Ziffern zu verwenden und um eine sinnvolle Sortierung zu ermöglichen, führende Nullen einzusetzen. Ein weiteres Kürzel kann dazu dienen, die ID's bestimmten Bereichen zuzuordnen, damit ist auf den ersten Blick erkennbar, in welchem Kontext die Anforderung verwendet wird. Die Beschreibung und der Umgang mit ID's sollte unbedingt im RE-Leitfaden festgehalten werden. Die Verwaltung von ID's ist am einfachsten mit einem speziellen RM-Tool möglich. Eine ID könnte beispielsweise wie folgt aussehen: LH-Bib-152. „LH" steht für „Lastenheft", „Bib" für „Bibliothekssystem" und 152 gibt die laufende Nummer der Anforderung an. (siehe Rupp/Sophisten 2014, 383f)
Das RM muss auch berücksichtigen, dass Anforderungen gewissermaßen „leben", d.h. sich verändern können. Deshalb sollte es möglich sein, den Bearbeitungsgrad einer Anforderung kenntlich zu machen. Damit verbunden muss auch festgelegt werden, ab welchem Bearbeitungsgrad eine Anforderung für andere Beteiligte sichtbar sein soll und wer zu welchem Zeitpunkt für die Anforderung verantwortlich ist. All dies sollte im RE-Leitfaden festgehalten werden. Auch die Entscheidungen bzgl. der Anforderungen sollten festgehalten und eventuell auch sichtbar gemacht werden. Ein Zustandsdiagramm kann diese Veränderungen festhalten.
Die Zustände einer natürlichsprachlichen Anforderung können unter anderem sein:

- gelöscht
- archiviert
- analysiert
- angelegt
- qualitätsgeprüft
- getestet
- abgenommen
- entworfen

- umgesetzt (siehe Rupp/Sophisten 2014, 390)

Da diese Bezeichnungen nicht selbsterklärend sind, ist es sinnvoll, sie genau zu definieren. Beispielsweise kann löschen bedeuten, die Anforderung vollständig und nicht wiederherstellbar aus dem System zu entfernen oder aber sie einfach unsichtbar zu machen.
Festgelegt werden sollte auch wie die Anforderungen von einem Zustand in einen anderen wechseln. Diese Zustandsübergänge sollten ebenfalls genau beschrieben und dokumentiert werden. Für die Dokumentation eignet sich in diesem Fall besonders gut die UML, es können aber auch Grafik-, Zeichen- und Präsentationstools genutzt werden.
Ändern sich Anforderungen müssen auch die damit zusammenhängenden Testfälle geändert werden. Wenn der Testdurchlauf erfolgreich war, muss auch die dazugehörige Anforderung mit der entsprechenden Information versehen werden. Erst wenn alle Testdurchläufe einer Anforderung erfolgreich waren, darf die Anforderung als „getestet“ bewertet werden. Auch die Testdurchläufe sollten mit Zuständen bewertet werden.

Nachdem der Lebenszyklus der Anforderungen beschrieben ist, müssen auch die einzelnen Tätigkeiten, die Abläufe untereinander und die unterschiedlichen Rollen identifiziert und beschrieben werden. Dabei kann man auf eine Stakeholdertabelle, die am Anfang des Projekts erstellt wurde, zurückgreifen. Wichtig ist es, die Rollen zu definieren:

- Der Projektleiter muss stets über den Stand des Projekts informiert sein und benötigt Informationen über Anforderungsänderungen, da Kosten und Aufwand abgeschätzt werden müssen.
- Die Autoren der Anforderungen benötigen Informationen, ob die Anforderungen akzeptiert wurden, wie sie priorisiert wurden und wie später die Umsetzung überprüft werden kann.
- Der RE hat die Aufgabe die Anforderungen zu formulieren und muss auf Vollständigkeit und Konsistenz achten. Bei Änderungen muss er wissen, welche Anforderungen betroffen sind um diese anzupassen.
- Der Architekt, der auf die Anforderungen hin das System plant, benötigt eine Gesamtübersicht über die essentiellen Funktionen, die Rahmenbedingungen, über nichtfunktionale Anforderungen und den Systemkontext.

- Der Entwickler benötigt einen freigegeben Satz an Anforderungen um mit der Umsetzung beginnen zu können
- Der Tester hat die Aufgabe die Testfälle zu schreiben und an die zugehörigen Anforderungen hängen zu können. Bei Änderungen muss er die Information erhalten, welche Anforderungen betroffen sind um die Testfälle adaptieren zu können
- Der Prüfer muss beurteilen können, ob die Anforderungen dem vorgegeben Qualitätsstandard entsprechen (siehe Rupp/Sophisten 2014, 399)

Wenn die Rollen definiert sind, benötigt man zusätzlich eine angepasste Rechteverwaltung. Dazu muss erstens geklärt werden, welche Informationen mit Rechten versehen werden müssen und welche Rolle wann welche Tätigkeiten durchführen darf. Mit einer Matrix kann dies übersichtlich dargestellt werden. Diese Auflistung sollte allen Beteiligten zur Verfügung stehen.

Bei jeder Anforderung sollte auch die Geschichte festgehalten werden, d.h. wer wann die Anforderung verändert hat. RM-Tools erledigen diese doch recht aufwändige Arbeit automatisch.

Für zahlreiche, vor allem kleinere wissenschaftliche Bibliotheken, ist der Umgang mit größeren Projekten allgemein noch ungewohnt, bei IT-Projekten, vor allem dann wenn die Zusammenarbeit zwischen den beiden Bereichen neu ist, entstehen zahlreiche Fragen, die schon mit der Wahl des geeigneten Kommunikationstools beginnen können. Inhaltlich profitieren beide Seiten, also Bibliothek und IT-Bereich bei einer engen Zusammenarbeit miteinander, wie auch die Ergebnisse des dreijährigen Projekts „e-Infrastructures Austria“ zeigen, das 2014 vom Bundesministerium für Wissenschaft, Forschung und Wirtschaft (BMWFW) initiiert wurde und sich zum Ziel gesetzt hatte, österreichweit den koordinierten Aufbau sowie die Weiterentwicklung von Repositorieninfrastrukturen zu fördern.[1]

[1] Siehe https://www.e-infrastructures.at/ (abgerufen am 19.2.2017)

1.2 Ausbildungsmöglichkeiten

2006 wurde im deutschsprachigen Raum das IREB, das International Requirements Engineering Board[2], gegründet, um RE auf ein professionelles Fundament zu stellen und um eine Standardisierung von RE-Fachwissen einzuführen. Ein Team aus weltweit anerkennten ExpertInnen aus den Bereichen Industrie, Beratung, Forschung und Lehre haben gemeinsam einen Lehrplan für den Bereich RE erarbeitet und das Zertifikat CPRE (Certified Professional for Requirements Engineering) entwickelt. Heute ist es zu einem wichtigen Expertengremium für die Personenzertifizierung von Fachkräften im RE geworden. „Seit 2007 haben mehr als 24.900 Personen in 61 Ländern die CPRE Foundation Level Prüfung bestanden. Dieses dreistufige Zertifizierungsmodell von IREB richtet sich an Personen, die sich mit Business Analyse, Requirements Engineering oder Testing befassen und einen hohen Anspruch an die Qualität ihrer Arbeit haben.“[3]

Ziel ist es, eine qualitätsgesicherte Standardisierung der Aus- und Weiterbildung zu erreichen. (Pohl/Rupp 2015, vii)

Auch das WIFI, das Weiterbildungsinstitut der Wirtschaftskammer, bietet einen Kurs zum Thema „Requirements Engineering für die agile Softwareentwicklung“ an[4]. An mehreren Fachhochulen und Universitäten werden bereits Lehrveranstaltungen zu diesem Thema angeboten, meist in den technisch orientierten Studienfächern. Helmuth Partsch, auf dessen Forschungen sich die vorliegende Arbeit zum Teil stützt, unterrichtet zum Beispiel an der Universität Ulm.[5] An der TU Wien ist RE Teil des Lehrplans. An der TU Graz fand in den Jahren 2011 bis 2013 sogar ein von der FFG gefördertes Forschungsprojekt zum Thema „IntelliReq - Intelligent Recommendation Technologies for Requirements Engineering“ statt.[6]

Im technischen Umfeld ist RE also zum Teil schon gut etabliert, in Bibliotheken – zumindest in Österreich – betritt man hier mehr oder weniger Neuland, wie auch die Ergebnisse der Umfrage für die vorliegende Arbeit zeigen werden.

---

[2] https://www.ireb.org/de/ (abgerufen am 19.2.2017)
[3] https://www.ireb.org/de/about/basics/ (abgerufen am 19.2.2017)
[4] siehe: http://www.wifiwien.at/eshop/bbdetails.aspx?bbnr=18267x (abgerufen am 19.2.2017)
[5] siehe: https://www.uni-ulm.de/in/pm/lehre/ss2015/re/ (abgerufen am 19.2.2017)
[6] siehe: https://online.tugraz.at/tug_online/fdb_detail.ansicht?cvfanr=F27146&cvorgnr=37&sprache=2 (abgerufen am 19.2.2017)

1.3 Einbettung und Abgrenzung des Themas

RE hängt eng mit Projektmanagement zusammen. Im Bereich der wissenschaftlichen Bibliotheken kann man gut beobachten, dass in den letzten Jahren die Projekte stark auf Technologie basieren. Ein Beispiel dafür ist der Umgang mit Forschungsdaten.
Seit relativ kurzer Zeit spielen Datenmanagementpläne (DMP) bei Drittmittelprojekten eine immer größere Rolle. Fördergeber erwarten, bzw. verlangen von den Forschenden, dass sie angeben, wie die in den Forschungsprojekten generierten Daten nach Projektabschluss nachgenutzt werden können. Um diesen Re-Use zu ermöglichen, müssen die Forschungsdaten aufbereitet und entsprechend langfristig und mit möglichst freien Lizenzen zur Verfügung gestellt werden. Für diesen freien und uneingeschränkten Zugang zu wissenschaftlicher Information im Internet werden das entsprechende Know-how, die nötige Infrastruktur und auch die professionelle Unterstützung durch TechnikerInnen benötigt. Nur so können beispielsweise Repositorien aufgebaut werden, wo Forschende ihre Daten archivieren können. Dafür muss ganz klar sein, welche Anforderungen die Forschenden an das jeweilige System haben um die Vorgaben der Fördergeber erfüllen zu können. Mit diesen Anforderungen sind wissenschaftliche Bibliotheken heute konfrontiert, wenn sie professionelle Forschungsunterstützung anbieten möchten. RE unterstützt die Umsetzung dieser Anforderungen.

1.4 Agile Vorgehensweisen und RE

RE ist eng verbunden mit der Idee der agilen Softwareentwicklung. Dabei stehen möglichst häufige Rückkoppelungsprozesse und zyklisches (iteratives) Vorgehen auf allen Ebenen im Vordergrund: sowohl bei der Programmierung, im Team als auch beim Management. Im Unterschied zu der klassischen Vorgehensweise wird bei der agilen Softwareentwicklung das neue System nicht in allen Einzelheiten bis zum Projektende genau durchgeplant und dann in einem einzigen langen Durchgang entwickelt, sondern es wird die Tatsache berücksichtigt, dass sich die Anforderungen während der Projektlaufzeit immer wieder ändern können, bzw. zu Projektbeginn noch gar nicht vollständig bekannt sein können. Beim iterativ-inkrementellen Vorgehen wird also die

gesamte Systementwicklung in zeitliche Abschnitte unterteilt (Iterationen) und das System in einzelnen Teilen (Inkrementen) realisiert. Bei der agilen Softwareentwicklung wechseln sich daher kurze Planungs- und Entwicklungsphasen ab. Zunächst wird eine Vision für das neue System entwickelt, Ziele festgelegt und priorisiert, ein Plan für eine erste Version ausgearbeitet und schließlich die Entwicklung begonnen. Danach werden notwendige Anpassungen vorgenommen. Durch ständige Prüfungen sind Fehlentwicklungen rasch erkennbar und es kann unmittelbar korrigierend eingegriffen werden. Somit werden Risiken minimiert.
Der agilen Methode steht das wasserfallartige Vorgehen, bei dem jede Tätigkeit in der Systementwicklung vollständig angeschlossen werden muss, bevor mit der nachfolgenden Tätigkeit begonnen werden kann, gegenüber. Verbesserungen an bereits freigegebenen Entwicklungen können hier nur noch über ein so genanntes Change-Management erfolgen.

Rupp spricht anschaulich über drei Gehirne als Speichermedien, die zusammenwirken müssen. Das Requirementsgehirn enthält Wissen über die Problemstellung, das Architekturgehirn ist mit Wissen um die Lösung gefüllt und das Managementgehirn enthält Projektwissen, wie Pläne und Schätzungen. (siehe Rupp/Sopisten 2014, 12) Diese drei Bereiche müssen stets vernetzt agieren, kein Bereich, bzw. in der Sprache Rupps „Gehirn“ kann getrennt von den anderen in einem Projekt erfolgreich arbeiten.
2001 wurde das „Manifesto for agile Software Development“ publiziert. Auf der in zahlreichen Sprachen verfügbaren Homepage kann man lesen:

„Wir erschließen bessere Wege, Software zu entwickeln, indem wir es selbst tun und anderen dabei helfen. Durch diese Tätigkeit haben wir diese Werte zu schätzen gelernt:

- Individuen und Interaktionen mehr als Prozesse und Werkzeuge
- Funktionierende Software mehr als umfassende Dokumentation
- Zusammenarbeit mit dem Kunden mehr als Vertragsverhandlung
- Reagieren auf Veränderung mehr als das Befolgen eines Plans.“[7]

Die 12 Prinzipien hinter dem Agilen Manifest lauten:

- Unsere höchste Priorität ist es, den Kunden durch frühe und kontinuierliche Auslieferung wertvoller Software zufrieden zu stellen.

[7] http://agilemanifesto.org/iso/de/manifesto.html (19.2.2017)

- Heiße Anforderungsänderungen sind selbst spät in der Entwicklung willkommen. Agile Prozesse nutzen Veränderungen zum Wettbewerbsvorteil des Kunden.
- Liefere funktionierende Software regelmäßig innerhalb weniger Wochen oder Monate und bevorzuge dabei die kürzere Zeitspanne.
- Fachexperten und Entwickler müssen während des Projektes täglich zusammenarbeiten.
- Errichte Projekte rund um motivierte Individuen. Gib ihnen das Umfeld und die Unterstützung, die sie benötigen und vertraue darauf, dass sie die Aufgabe erledigen.
- Die effizienteste und effektivste Methode, Informationen an und innerhalb eines Entwicklungsteams zu übermitteln, ist im Gespräch von Angesicht zu Angesicht.
- Funktionierende Software ist das wichtigste Fortschrittsmaß.
- Agile Prozesse fördern nachhaltige Entwicklung. Die Auftraggeber, Entwickler und Benutzer sollten ein gleichmäßiges Tempo auf unbegrenzte Zeit halten können.
- Ständiges Augenmerk auf technische Exzellenz und gutes Design fördert Agilität.
- Einfachheit – die Kunst, die Menge nicht getaner Arbeit zu maximieren – ist essenziell.
- Die besten Architekturen, Anforderungen und Entwürfe entstehen durch selbstorganisierte Teams.
- In regelmäßigen Abständen reflektiert das Team, wie es effektiver werden kann und passt sein Verhalten entsprechend an.[8]

Die agile Methode (Scrum) gibt folgende Rollen vor:

- Product-Owner (in wasserfallartigen Projekten der Produktmanager und der RE), der das Produktmanagement und das RE übernimmt, sowie die Kosten/Nutzen-Relation kontrolliert
- Scrum-Master (in wasserfallartigen Projekten der Projektleiter, Team-Lead, Gruppenleiter, bzw. Prozessmanager), der für das prozessorientierte Projektmanagement und für die Organisationsentwicklung zuständig ist
- Das Entwicklungsteam (in wasserfallartigen Projekten der Entwickler, Architekt, Tester usw.), das für alle operativen Aufgabenbereiche zuständig ist. (siehe auch Rupp/Sophisten 2014, 505f)

Der so genannte Scrum-Master achtet während des Projekts auf die Regeln. Der Product-Owner fungiert als Schnittstelle zwischen Stakeholdern und Entwicklerteam und sorgt für Qualität. Das gesamte Entwicklerteam ist für die Umsetzung

[8] http://agilemanifesto.org/iso/de/principles.html (19.2.2017)

verantwortlich, es arbeitet selbständig und kann bei Bedarf ExpertInnen hinzuziehen. Die für das zu entwickelnde System relevanten Anforderungen werden geordnet und bilden gesammelt das Product-Backlog. Für die Erstellung, Pflege und die Ordnung der Backlog-Items ist der Product-Owner verantwortlich, mit Unterstützung des Entwicklerteams. Das Product-Backlog ändert sich im Laufe eines Projekts. Die Anforderungen werden als User-Storys dargestellt. Unter User-Storys versteht man kurzgehaltene und in die Alltagssprache formulierte Software-Anforderungen. User-Storys beschreiben relevante Funktionalitäten und bestehen aus einer schriftlichen Beschreibung, Gespräche über die Funktionalität und Akzeptanzkriterien, die festlegen, wann eine User-Story umgesetzt ist. User-Storys können beliebig dokumentiert werden, z.B. mittels Aktivitäts- oder Zustandsdiagrammen. Mehrere User-Storys ergeben einen Use-Case. Um die weitere Planung, eine Sprint-Planung, zu ermöglichen, müssen die User-Storys weiter verfeinert werden, es muss auch darauf geachtet werden, dass sich innerhalb dieser User Storys keine weiteren User-Storys und keine Abhängigkeiten befinden.

Mittels Akzeptanzkriterien werden die Storys weiter präzisiert. So genannte „Themes“ strukturieren die User-Storys thematisch. Unter Epics versteht man grobgranulare Anforderungen, deren Umfang nicht abschätzbar ist. Eine Product-Vision bezeichnet eine abstrakte Darstellung des zu entwickelnden Systems und wird zu Projektbeginn entwickelt. Die User-Storys werden zu Sprint-Backlogs zusammengefasst. Jede dieser User Storys muss klar und verständlich sein und dem „Ready“ entsprechen. „Done“ bedeutet, was zusätzlich zu den User-Storys in den Sprints erfüllt sein muss, damit ein Inkrement als fertig gelten kann. (siehe Rupp/Sophisten 2014, 59) Wichtig dabei ist, jeweils eine Definition von „done“ und von „ready“ zu haben.

„Der Scrum-Guide definiert die Definition of Done als ein Artefakt, das bei allen Projektbeteiligten ein gemeinsames Verständnis dafür erzeugt, welche formalen Kriterien erfüllt sein müssen, damit die Arbeit an einem System- oder Produktinkrement als abgeschlossen gilt“ (ebd., 260)

Ein fertiges Produktinkrement muss sofort ohne weitere Arbeiten daran auslieferbar sein.

Die Definition of Done muss für jedes Unternehmen und sogar für jedes Projekt definiert werden, minimale Anforderungen wären:

- Akzeptanzkriterien erfüllt
- Code ist lauffähig
- Unit-Tests abgeschlossen
- Systemtest abgeschlossen
- Dokumentation vorhanden (ebd.)

Dazu kommen eventuell noch nicht-funktionelle Anwendungen, z.B. im Bereich der Performance. Die Definition of Done ändert sich unter anderem auch mit der Erfahrung des Entwicklerteams.
Die Definition of Ready wird verwendet, um zu prüfen, ob User-Storys klar und konkret genug sind, um vom Entwicklungsteam im Sprint umgesetzt werden zu können. (Rupp/Sophisten 2014, 261) So muss der Product-Owner in der Lage sein, sie dem Team so zu erklären, dass es weiß was zu tun ist. Das Entwicklerteam ist sogar berechtigt, User-Storys, die nicht ready sind, zurückzuweisen. Damit sollen unnötige Verzögerungen durch unklare Aufgabenstellungen oder unterschiedliche Vorstellungen vom Endprodukt vermieden werden. Die Stakeholder sind dabei aufgefordert, sich Gedanken über ihre Forderungen zu machen, der Product-Owner ist dafür verantwortlich, dass die User-Storys ausreichend verfeinert werden.
Eine Definition of Ready könnte folgende Punkte enthalten:

- User-Story ist definiert und dokumentiert
- User-Story lässt sich zu einem Quellendokument zurückverfolgen (falls vorhanden)
- Akzeptanzkriterien sind definiert
- Abhängigkeiten der User-Storys sind identifiziert
- Gegebenenfalls sind Benutzerszenarien vorhanden
- Der verantwortliche Stakeholder für die User-Story ist identifiziert
- Das Entwicklungsteam hat eine gute Vorstellung davon, wie die Demonstration für die User-Story aussehen könnte. (Rupp/Sophisten 2014, 262)

Die Inhalte der Definition of Ready werden gemeinsam von Entwicklungsteam und Product-Owner festgelegt, sie sind abhängig von der Erfahrung des Entwicklerteams, von der Vertrautheit mit dem Fach und der Technik und von der Art der Kommunikation zwischen Entwicklungsteam und Product-Owner. Auch die Definition of Ready sollte während des Projektverlaufs immer wieder angepasst werden.

Die Dokumentationstechniken hängen von menschlichen Einflussfaktoren, z.B. Erfahrung und zeitliche Verfügbarkeit der Stakeholder und des Product-Owner, organisatorischen Einflussfaktoren, wie Transparenz, Zeitrahmen, Ressourcen und fachlich-inhaltlichen Einflussfaktoren, wie beispielsweise von der Größe und Qualität der User-Storys, der Komplexität und dem Einsatz von neuen Technologien ab.
Ein Scrum-Projekt umfasst eine Iteration, die als „Sprint“ bezeichnet wird. Diese besitzen eine genau festgelegte Dauer, die während des Projekts nicht verändert werden sollte, häufig dauern diese Sprints zwischen einer und vier Wochen. Am Anfang steht das Sprint-Planning, auch das ist zeitlich genau begrenzt. Der Product-Owner hat schon zuvor die User-Storys aus dem Product-Backlog vorbereitet um sie dem Entwicklerteam mitteilen zu können. Er beantwortet auch Fragen aus dem Entwicklerteam. Gemeinsam werden die User-Storys priorisiert und es wird ein so genanntes „Sprint-Goal“ entwickelt, auf das hingearbeitet wird. Daily-Scrums,15-minütige Meetings, überprüfen jeden Tag die Fortschritte, das Entwicklerteam berichtet von den Erfolgen des Vortages, von den Vorhaben des jeweiligen Tages und von eventuell auftretenden Hindernissen. Am Ende des Sprints steht das Sprint-Review. Die Stakeholder und das Scrum-Team inspizieren das Inkrement, das entweder abgenommen oder nicht abgenommen wird, sowie ein überarbeitetes Product-Backlog. In der Sprint-Retrospective wird das bisherige Vorgehen beurteilt und eventuell Verbesserungsvorschläge eingebracht.

Betrachtet man RE anhand der vier Thesen des Agilen Manifests, ist es eigentlich immer agil:

1. Individuen und Interaktionen sind wichtiger als Prozesse und Werkzeuge
Kommunikation ist beim RE sehr wichtig. AnalytikerInnen müssen sowohl im Team als auch mit den Stakeholdern gut kommunizieren können.
2. Reagieren auf Veränderung ist wichtiger als das Befolgen eines Plans
Änderungen bzgl. der Detaillierungsgrad oder die Notwendigkeit neue Anforderungsquellen hinzuzuziehen, treten ständig auf und bestimmen den Alltag von AnalytikerInnen.
3. Funktionierende Software ist wichtiger als umfassende Dokumentation

An den Anforderungen zu arbeiten sollte Priorität vor der Verschriftlichung der Dokumentation haben.

4. Zusammenarbeit mit dem Kunden ist wichtiger als Vertragsverhandlung

Der gesamte RE-Prozess muss sehr eng mit allen Stakeholdern erfolgen.

Beim RE lässt sich aber auch gut hybrid vorgehen, hier wird die agile Phase in den traditionellen Gesamtprozess eingebettet.

Die Dokumentation wird oft vernachlässigt, es ist jedoch sinnvoll zu dokumentieren – auch in agilen Projekten. Wird Wissen nur kurzzeitig benötigt, sollte der Stand auf der Dokumentation vermerkt sein. Empfohlen wird stets möglichst zeitnah zu dokumentieren um nichts zu vergessen, andererseits kann eine zu frühe Dokumentation dazu führen, dass diese immer wieder adaptiert werden muss. Es sollte jedoch auch die Frage gestellt werden, wie lange nach der Entstehung die Dokumentation benötigt wird, wann die Information dokumentiert werden soll und wie dokumentiert werden soll. (siehe Rupp/Sophisten 2014, 64) Dokumentation verhindert, dass Wissen durch den Weggang von MitarbeiterInnen wegfällt.

Die oben genannten vier Thesen des Agilen Manifests kann man als Reaktion auf die ständigen technischen Neuerungen sehen, die vor allem auch wissenschaftliche Bibliotheken betreffen, wenn man bedenkt, dass Bibliothekskataloge, Datenbanken und Suchmaschinen heute selbstverständlich über Smartphones und Tablets abgefragt werden. Eine lange Projektlaufzeit würde verhindern, rasch auf diese Veränderungen reagieren zu können. Der Zentrale Informatikdienst der Universität Wien hat 2012 auf die agile Vorgehensweise umgestellt, was beispielsweise für die Universitätsbibliothek Wien bedeutet, sich auf etwas andere Arbeitsabläufe und Ansprechpartner einstellen zu müssen.

1.5 Gründe für Requirements-Engineering

Im Laufe einer Systementwicklung kann, wie man aus dem Arbeitsalltag weiß, vieles schief laufen. RE soll für einen reibungslosen Weg vom Erstellen der Anforderungen bis zur Umsetzung sorgen.

Stolpersteine können sein:

- Erfolgt die Kommunikation beim Anforderungsmanagement über beliebige Kanäle, besteht die Gefahr, dass die Informationen unkoordiniert oder redundant weitergegeben werden oder sich widersprechen.
- Das führt zu weiterem Kommunikationsaufwand.
- Es werden unnötige Anforderungen umgesetzt.
- Aufgrund widersprüchlicher Informationsstände sind Korrekturen notwendig.
- Die Klärungsprozesse mit den Anforderern dauern lange, binden viele Ressourcen und erbringen unzureichende Ergebnisse, so dass weitere Klärungsgespräche notwendig sind.
- Änderungen des Projektablaufs werden schlecht koordiniert.
- Unter Zeitdruck und aufgrund des knappen Budgets werden „schnelle“ Lösungen entwickelt. Das führt später zu Mehrkosten in der Wartung.
- Es sind Nacharbeiten notwendig, um geforderte Eigenschaften des Projektergebnisses nachzuliefern.
- In Folgeprojekten entstehen Mehrkosten, weil veränderte Anforderungen aufgrund des Zeitdrucks nicht mehr konsistent in alle Projektergebnisse eingearbeitet wurden. In der Folge entsteht ein zusätzlicher Klärungsbedarf. So müssen z.B. Widersprüche in Dokumenten (Fachkonzept, Schnittstellenspezifikation, Benutzerhandbuch) aufgelöst werden. (Heche/Ollech 2006, 5)

Wenn beispielsweise die Anforderungen nur in Prosa verfasst sind, die Geschäftsprozesse kaum oder nicht bekannt sind, die Anforderungen zu abstrakt beschrieben und deshalb nicht greifbar sind, oder wenn Begriffe nicht klar definiert wurden, bzw. aus sprachlichen Gründen nicht immer die gleichen Begriffe verwendet werden, kommt es mit ziemlicher Sicherheit zu Verwirrungen. Nicht klare Anforderungen können auch nicht sinnvoll umgesetzt werden. Wenn auf der anderen Seite die Beschreibung auf einer rein technischen Ebene erfolgt, die AnwenderInnen dies aufgrund ihrer Ausbildung nicht verstehen, diese Tatsache aber nicht zugeben wollen, wird dem Gesamtprojekt ebenfalls kein Erfolg beschieden sein. Die TechnikerInnen setzen in diesem Fall Anforderungen um, die von den Stakeholdern

nicht oder nicht in dieser von den TechnikerInnen verstandenen Weise gewünscht waren. (siehe Rupp/Sophisten 2014, 15)
Gut definierte Anforderungen, die einer Qualitätsprüfung standhalten müssen, bestimmen über den weiteren Verlauf der Systementwicklung. Auf die Anforderungen baut die gesamte Kommunikation und Diskussion auf.
Die Dokumentation ist zeitaufwändige und wird deshalb oft mit keiner hohen Priorisierung versehen, da die Projektbeteiligten in vielen Fällen das Gefühl haben, ohnehin alles im Kopf zu haben. In Wirklichkeit gehen jedoch, nicht nur durch Personalwechsel, viele Inhalte verloren, wenn sie nirgends festgehalten sind. Vor allem Wissen über bereits erledigte Aufgaben, wird recht rasch vergessen. (siehe Rupp/Sophisten 2014, 20) Einen Sachverhalt zu dokumentieren kann auch dabei helfen, ihn besser zu verstehen, vor allem wenn nicht nur reiner Text sondern auch bildhafte Darstellungen verwendet werden. Auch um in der Lage zu sein, den Gesamtzusammenhang zu überblicken, ist Dokumentation unverzichtbar. Man vermeidet damit auch unterschiedliche Interpretationen und Missverständnisse. Selbstverständlich müssen alle Beteiligten die gleiche Version der Dokumentation erhalten.

Laut Partsch spricht gegen RE eigentlich vor allem eine „allgemeine latente Bequemlichkeit“ (Partsch 1991, 8)
Mit RE sollen Fehleinschätzungen minimiert werden. RE wirkt auch gegen fehlerhafte oder mangelnde Kommunikation von Anforderungen, unrealistische Kosten- und Zeitpläne, mangelndes Verständnis für die Bedürfnisse der NutzerInnen und unvollständige und inkonsistente Spezifikationen. (siehe auch Partsch 1991, 20)
Wenn die Anforderungsspezifikation nicht qualitativ hochwertig ist, sind meist die BenutzerInnen aus dem Prozess ausgeschlossen, das Management hat keine Informationen, was eigentlich produziert wird und es gibt auch keine Testphase, weil nichts Konkretes vorhanden ist. (siehe ebd., 20)
Die Gründe, die für RE sprechen, sind logisch und leicht nachvollziehbar, sie setzen allerdings gewisse Vorkenntnisse, beispielsweise im Projektmanagement voraus. Während sich in der Wirtschaft und vor allem in der Wissenschaft eine gut

funktionierende Projektkultur aufgebaut hat, war dies in Bibliotheken bis vor einigen Jahren noch relativ unbekannt, weil die Durchführung von größeren Projekten auch nicht wirklich erforderlich war. Sieht man sich jedoch die heutigen Handlungsfelder von Bibliotheken an, wie etwa Forschungsdatenmanagement oder digitale Langzeitarchivierung, an denen viele Personen beteiligt sind, sind gut geplante Projekte und ein professioneller Umgang mit Anforderungen unumgänglich. Damit RE jedoch wirkungsvoll eingesetzt werden kann, müssen gewisse Grundlagen geschaffen werden.

1.6 Grundlagen für ein gut funktionierendes Requirements-Engineering

Wichtig, damit RE gut funktionieren kann, ist der optimale Ablauf von Kommunikation zwischen RE und Stakeholder.
Der RE muss sozusagen Katalysator zwischen AuftraggeberInnen, TechnikerInnen und den einzelnen Stakeholdern sein. Die unterschiedlichen Wünsche der KundInnen müssen erkannt und extrahiert werden, eine gemeinsame Terminologie muss aufgebaut, gemeinsam verwendet und verwaltet werden. Dafür eignet sich ein Glossar.
Auch die Kluft zwischen den unterschiedlichen Detaillierungsniveaus muss überwunden werden. Die AuftraggeberInnen sind meist weniger an Details interessiert als die EntwicklerInnen. AuftraggeberInnen sehen ausgereifte Anforderungen meist nur als notwendiges Übel an – sie sind am Endprodukt interessiert, während EntwicklerInnen sie als Basis wahrnehmen um überhaupt mit der Arbeit beginnen zu können. Der RE hat die Aufgabe auch diese Perspektiven miteinander in Einklang zu bringen. Schließlich müssen auch die EndnutzerInnen im Auge behalten werden. Der gesamte Kommunikationsprozess und damit verbunden die Aufgabe, Beschreibungsmethoden konsistent zu verwenden, liegt ebenfalls im Bereich des RE.
Größere wissenschaftliche Bibliotheken haben unter anderem in den Bereichen Bibliometrie, Forschungsunterstützung und Publikationsförderungen AnsprechpartnerInnen aus sämtlichen wissenschaftlichen Fächern. Die Anforderungen und Erwartungen dieser heterogenen Gruppe, sind sehr unterschiedlich. Als Beispiel ist etwa der Umgang mit Forschungsdaten zu nennen. Für NaturwissenschafterInnen sind

dies eher Messdaten, für GeisteswissenschafterInnen sehr oft Bilder oder Texte. Daraus ergeben sich nicht nur völlig unterschiedliche Bedürfnisse beim technischen Aspekt der Archivierung, sondern auch hinsichtlich der juristischen Fragen. Die verschiedenen Bedürfnisse müssen möglichst präzise abgefragt werden, so kann Open Access für unterschiedliche Personen etwas ganz anderes bedeuten. Für RE bedeutet dies auch, sich in die diversen Fachgebiete zumindest so weit einzuarbeiten, dass optimal nach den Wünschen und Bedürfnissen der WissenschafterInnen gefragt werden kann.

2. Phasen des Requirements-Engineerings

Grundsätzlich müssen zunächst alle beteiligten Personen, also alle Stakeholder, ausgeforscht werden, ebenso alle Dokumente und Systeme, die direkt oder indirekt von den neuen Entwicklungen betroffen sind. Auch die Ist-Situation muss erhoben werden. Danach muss die Situation bewertet und Ziele abgeleitet werden. Wichtig dabei ist sämtliche Quellen, die Auskunft über Anforderungen geben können, zu nutzen. Neben Normen, Standards und Gesetzestexten können dies auch Fehlerberichte eines alten Systems, bzw. Schulungsunterlagen und Dokumentationen sein. Als hilfreich kann sich aber auch die Analyse von Konkurrenzprodukten oder anderen Systemen erweisen. Die unterschiedlichen Anforderungsquellen sollten auch entsprechend verwaltet werden und den EntwicklerInnen jederzeit zur Verfügung stehen. Die Suche nach Anforderungsquellen beschränkt sich jedoch nicht nur auf den Projektbeginn sondern sollte sich während der Projektlaufzeit fortsetzen.
Das Festlegen und die Benennung der Ziele ermöglichen erst die Formulierung von sinnvollen Anforderungen. Es sollten quantifizierbare Angaben aufgezählt werden, „denn Ziele sind Anforderungen auf einer abstrakten Ebene“. (Rupp/Sophisten 2014, 75). Ebenfalls zu bedenken ist, dass festgelegt werden sollte, wie Ziele dokumentiert werden. Die Dokumentation kann unterschiedlich formal erfolgen. Die Ziele müssen klar definiert, gut nachvollziehbar und auch gut kommunizierbar sein. Jede Anforderung, die später erstellt wird, muss mit dem Ziel in Einklang stehen. D.h. auch, dass jedes Ziel mit einem Stakeholder verknüpfbar sein muss. Die Frage, wie man Ziele erstellt, hängt von den Umständen ab. Wenn es sich um ein neues Produkt handelt, sind eher kreative Ansätze hilfreich, wenn es sich um die Optimierung eines bestehenden Systems handelt, ist es wichtig die Systemumgebung im Auge zu behalten.
Ziele und Umfang des Projekts müssen immer wieder aktualisiert werden. in welcher Reihenfolge diese vorbereiten Tätigkeiten durchgeführt werden ist laut Rupp nicht festgelegt, sie sollten jedoch nicht mehr als 5% des Gesamtaufwandes übersteigen. (siehe Rupp/Sophisten 2014, 74)

2.1 Glossar

Als Glossar wird eine – sinnvollerweise alphabetisch gereihte – Sammlung von Begriffsdefinitionen verstanden, das kontextspezifische Fachbegriffe, Akronyme und Abkürzungen, alltägliche Begriffe, die im Kontext eine spezifische Bedeutung haben, Synonyme (verschiedene Begriffe mit der gleichen Bedeutung) und Homonyme (Begriff mit verschiedenen Bedeutungen) enthält. Wenn man gleich zu Beginn ein Glossar aufbaut, das Fachbegriffe genau erklärt und eine für alle am Projekt Beteiligten verbindliche Definitionen liefert, sowie auch Abkürzungen auflöst, erleichtert dies die weitere Arbeit enorm. Vor allem sollte man auch auf Verben nicht vergessen, die auf den ersten Blick logisch wirken. Das Wort „archivieren“ kann zum Beispiel für eine Person „ablegen für eine bestimmte Zeit“ bedeuten, für eine andere „Langzeitarchivierung“ und eine dritte Person versteht darunter zusätzlich auch den raschen und gut visualisierten Zugriff auf die Daten. Wird hier keine genaue Definition festgelegt, kommt es zu falschen Erwartungen und Enttäuschungen.
Synonyme der Stakeholder sollten ebenso eingefügt werden, wie Eigenschaftswörter, die unterschiedlich interpretiert werden könnten. Auch Rollen sollten dort definiert werden, ebenso juristische Verbindlichkeiten. In der Literatur (wie etwa bei Pohl/Rupp 2015, 6) wird empfohlen, ein Glossar an den Anfang eines Dokuments zu stellen. Damit kann sichergestellt werden, dass für jede neue Spezifikation derselbe Begriff verwendet wird. Bei sehr ähnlichen Projekten bietet es sich an, projektübergreifende Glossars aufzubauen und wiederzuverwenden, da Begriffe dann nicht neu gelernt werden müssen und sich der Aufwand für die Erstellung eines Glossars verringert. Wenn vorhanden, ist es auch sinnvoll auf bereits bestehende Glossare zurückzugreifen, wie etwa auf das 1990 veröffentlichte „IEEE Standard Glossary of Software Engineering Terminology, den IEEE Standard 610.12-1990, der gebräuchliche Begriffe des Software Engineering definiert.
Das gewählte Glossar muss mit sämtlichen Stakeholdern abgestimmt werden und allen zur Verfügung stehen. Es sollte gepflegt, versioniert und explizit freigegeben werden. Zu beachten ist, dass jeweils nur eine gültige Version des Glossars zulässig ist und es auch eine zentrale Verwaltung und eine/en Verantwortlichen gibt. Eine einheitliche

Struktur der Einträge ist ebenso wichtig, wie die Angabe der Quellen für die einzelnen Begriffe.

2.2 Der RE-Leitfaden

Um die Übersicht nicht zu verlieren, empfehlen ExpertInnen einen so genannten RE-Leitfaden, den man auch als RE-Handbuch, RE-Konzept oder RE-Richtlinie bezeichnen kann. Eine Definition lautet etwa: „Der RE-Leitfaden ist das zentrale Artefakt, in dem alle Prozesse, Methoden, Artefakte, Tools, Rollen und Vorgehensweisen dokumentiert sind, die für die Durchführung der Anforderungsanalyse gebraucht werden.“ (Rupp/Sophisten 2014, 370)
Eine Aufgabe des RE ist es, diesen Leitfaden zu pflegen und die Anwendung zu überwachen.
Der Leitfaden soll beinhalten, wie die Anforderungsanalyse ablaufen soll, welche Verwaltungsaufgaben dafür nötig sind. Rupp empfiehlt dafür spezielle RE-Tools zu verwenden (Rupp/Sophisten 2014, 371). Am wichtigsten ist es jedoch möglichst früh damit zu beginnen, vor allem, wenn zahlreiche Anforderungen zu verwalten sind, wenn das zu erstellenden Produkt hochqualitativ und langfristig genutzt werden soll, wenn eine große Anzahl an Stakeholdern einzubeziehen ist und die Zusammensetzung inhomogen ist, wenn eine Wiederverwendung der Anforderungen zu erwarten ist, der Entwicklungsprozess komplex ist und viele Releases erwartet werden.
Hilfreich für einen RE-Leitfaden ist eine gute Struktur. Es gibt bereits mehrere Standardgliederungen auf die man zurückgreifen kann, wie etwa: Volere, V-Modell XT, IEEE 29148.

Unternehmen verwenden jedoch oft ihre eigenen Strukturvorlagen. Hilfreich ist dabei sich schon im Voraus zu überlegen, wie Kapitel sinnvoll in Unterkapitel eingeteilt werden, ohne dass Informationen evtl. an zwei Stellen auftauchen.
Gliederungsstrategien können sein:

- Definitionen: in lexikalische Gruppen einordnen (A-F, G-M, N-S, T-Z)
- Systemanforderungen: Anhand von Anwendungsfällen und Aktivitäten, Features, Signalen und Benutzergruppen
- Technologische Anforderungen: anhand von Hardwarekomponenten oder Schnittstellen

- Anforderungen an durchzuführenden Tätigkeiten: anhand von Verantwortungsbereichen oder Prozessschritten (Rupp/Sophisten 2014, 381f)

2.3 Definition der Grenzen des Systems

Bevor die Anforderungen an ein System ermittelt werden können, müssen die Grenzen klar gesteckt werden. Wo hört das System, auf das sich die Arbeit bezieht, auf und wo beginnt ein anderes System? Nur so können die Anforderungen klar bestimmt werden. Das zu analysierende System, auf das sich die Anforderungen bezieht, ist immer eingebettet in eine Umgebung. Am Anfang muss auch geprüft werden, wie sich das System in die Umwelt integrieren kann, um eventuelle Einflussbereiche erkennen zu können. Ziel dabei ist auch den so genannten Systemkontext definieren zu können. Als Systemkontext bezeichnet man jenen „Teil der Umgebung eines Systems, der für die Definition und das Verständnis der Anforderungen des betrachteten Systems relevant ist." (Pohl/Rupp 2015, 13). Dazu zählen alle Hardwaresysteme, Stakeholder, Organisationen, Prozesse, Ereignisse sowie Dokumente, wie zum Beispiel Standards oder Gesetze, die mit dem zu analysierenden System in irgendeiner Form in Verbindung stehen und eine Beziehung dazu haben. Werden hier Elemente übersehen, wie beispielsweise, dass Stakeholder bestimmte Standards bei einer Schnittstelle einfordern, hat dies später kosten- und zeitintensive Auswirkungen. Kontextvisualisierungen mit Hilfe von Use-Cases erleichtern diese Abgrenzung. Will man Abläufe gut sichtbar machen, erleichtern Kontextdiagramme die Arbeit.
Graubereiche, also nicht klar definierte Abgrenzungen zwischen zwei Systemen müssen eruiert werden. Die Abgrenzung des Systems kann unterschiedlich erfolgen:

- Bei der Systemabgrenzung wird die Systemgrenze bestimmt, es wird festgelegt, welche Aspekte durch das geplante System abgedeckt werden sollen.
- Bei der Kontextabgrenzung bestimmt man die Grenze zum irrelevanten Kontext, es wird geprüft, welche Aspekte in der Umgebung eine Beziehung zu dem geplanten System haben.

Systemgrenze und Kontextgrenze definieren gemeinsam den Systemkontext. Hier sind sämtliche Aspekte vereint, die für das geplante System relevant sind, aber nicht im Rahmen des Systems gestaltet werden können.
Alle Aspekte, die sich innerhalb der Systemgrenze befinden, können verändert oder gestaltet werden, das betrifft zum Beispiel Hard- oder Software, die ersetzt werden kann, bzw. auch Prozesse, die verändert werden können.
Das zu analysierende System selbst kann wieder aus mehreren Teilen, bzw. Bereichen bestehen, so kann es unterschiedliche IT-Systeme, Applikationen umfassen und über Organisationsgrenzen hinweg aufgebaut sein. Um diese zum Teil relativ komplexen Elemente und Abläufe zu visualisieren, bietet sich etwa ein Flussdiagramm an, in dem Prozesse und Schnittstellen beschriftet werden. Jede Änderung von Schnittstellen hat natürlich Einfluss auf das gesamte System. Änderungen des Kontextes bedeuten oft, dass eine Analyse wiederholt werden muss, da sich die Rahmenbedingungen ändern.
Die Kontextgrenze unterscheidet zwischen Kontextaspekten, die berücksichtigt werden müssen und jenen, die für das geplante System irrelevant sind. Nicht nur die Systemgrenzen können sich währen des Gesamtprozesses verschieben, sondern auch die Kontextgrenzen. Das kann zum Beispiel passieren, wenn während des Prozesses neue Vorschriften berücksichtigt werden müssen. Eine durchgängige Abgrenzung ist aufgrund der Komplexität jedoch in der Realität kaum möglich. So ist auch nicht immer feststellbar, ob und wie Aspekte einander beeinflussen. Die Grauzonen bei der Systemabgrenzung müssen immer aufgelöst werden, jene in der Kontextabgrenzung müssen dagegen nicht aufgelöst werden. Beim agilen Arbeiten sind zunächst viele dieser Grauzonen vorhanden und werden erst nach und nach aufgelöst.
Es kann aber im Laufe der Arbeit auch passieren, dass es zu weiteren Anforderungen an die benachbarten Systeme kommt. Die endgültige Systemabgrenzung ist deshalb oft erst am Ende des Projekts möglich. Auch durch diese Unschärfen können immer wieder Graubereiche entstehen, weil zum Beispiel während des Prozesses erkannt wird, dass gewisse Funktionalitäten nun doch innerhalb der Systemgrenze liegen und entwickelt werden sollen.

Für die Visualisierung bieten sich unterschiedliche Möglichkeiten an. Dazu zählen Datenflussdiagramme, Mind Maps und Anwendungsfalldiagramme, wenn nur ein Use Case dargestellt werden soll.

„Ein Use Case beschreibt Funktionen, die ein Akteur mithilfe des Systems ausführen möchte. Daraus abgeleitete Anforderungen werden als ‚funktionale Anforderungen' bezeichnet. Sie beschreiben jeweils einen der folgenden drei Fälle:

- eine Interaktion zwischen dem Benutzer und dem System
- eine Interaktion zwischen dem System und einem anderen System oder
- eine selbstständige Systemaktion aufgrund eines zeitlichen oder ereignisgesteuerten Auslösers." (Zastrow/Wagner 2016, 2)

Für die Dokumentation des Systemkontexts werden oft Use-Case-Diagramme oder Datenflussdiagramme verwendet. Sie zeigen den Datenfluss durch ein System und können allgemeine Bearbeitungsschritte, wie das Verändern von Daten darstellen. Der Vorteil ist, dass sie leicht zu erlernen sind und bei verschiedenen Abstraktionsebenen verwendet werden können. Sie bieten sich an um einen Überblick über den Informationsfluss im System zu geben. Um den Systemkontext darzustellen, werden auch oft UML-Klassendiagramme herangezogen.

Um festzuhalten, welche Teile nicht zum System gehören, empfiehlt sich eine Negativliste, die die Zieldefinition ergänzt.

Auch diese Vorgehensweise scheint auf den ersten Blick selbstverständlich zu sein. In größeren Institutionen fehlt jedoch manchmal die nötige Transparenz über die unterschiedlichen bereits eingesetzten Systeme. Eine mangelhafte Kommunikationskultur kann jedoch dazu führen, dass Dinge doppelt oder aneinander vorbei entwickelt werden. Regelmäßige Meetings mit den betroffenen Personen aus der Bibliothek und den IT-Services sind deshalb wichtig.

2.4 Beschreibung der Ausgangssituation

Bevor die Anforderungen eruiert werden können, empfiehlt es sich die Ausgangssituation der Institution zu analysieren und zu beschreiben. Dazu zählen auch die Geschäftsprozesse, die die Tätigkeiten von Personen, nicht aber die IT-Systeme berücksichtigen. Geschäftsprozess meint eine Bündelung an Einzeltätigkeiten, die dazu

führen, ein bestimmtes geschäftliches Ziel zu erreichen. Eine Ist-Analyse der Geschäftsprozesse deckt die Zusammenhänge auf und zeigt Probleme und Verbesserungspotentiale. Diese dient dazu sämtliche wichtige Aktivitäten und Funktionen berücksichtigen zu können. Eine Soll-Analyse beschäftigt sich damit, wie der Geschäftsprozess in Zukunft aussehen soll. Um geschäftliche Abläufe abzubilden, werden vor allem Business-Use-Case-Diagramme verwendet. Damit erhält man einen Überblick über die unterschiedlichen Systeme, Prozesse werden voneinander abgegrenzt und für alle Personen leicht verständlich aufbereitet, zeitliche Abfolgen von Prozessen können damit jedoch nicht dargestellt werden.
Kausale Zusammenhänge zwischen den einzelnen Schritten eines Verarbeitungsprozesses können durch Ablaufdiagramme, auch Flussdiagramme genannt, visualisiert werden.

2.5 Das Ermitteln von Anforderungen

Das Ermitteln von Anforderungen (Requirements Eliciation) nimmt im Gesamtprozess des RE viel Zeit in Anspruch und stellt sich in mehreren Schritten dar.

2.5.1 Definition von Anforderungen

Für das deutsche Wort „Anforderung“ werden unterschiedliche Begriffe verwendet, wie etwa „Forderung“, „Ziel“, „Eigenschaften“ oder auf Englisch z.B. „Feature“. Auch Wünsche werden fälschlicherweise oft als Anforderungen bezeichnet. Eine Anforderung kann man jedoch auch folgendermaßen definieren:

1. Eine Bedingung oder Fähigkeit, die von einem Benutzer (Person oder System) zur Lösung eines Problems oder zur Erreichung eines Ziels benötigt wird
2. Eine Bedingung oder Fähigkeit, die ein System oder Teilsystem erfüllen oder besitzen muss, um einen Vertrag, eine Norm, eine Spezifikation oder andere, formell vorgegebene Dokumente zu erfüllen
3. Eine dokumentierte Repräsentation einer Bedingung oder Eigenschaft gemäß (1) oder (2). (IEEE Std 830-1998, übers. von Pohl/Rupp 2015, 3)

Der Weg vom Zusammenstellen von Informationen bis zur fertigen Anforderung, die umgesetzt werden kann und den Bedürfnissen aller Stakeholder entspricht, ist fehleranfällig. Vor allem im Bereich der Anforderungsanalyse haben Fehler kostspielige und weitreichende Auswirkungen. Die häufigsten Probleme dabei sind:

- Unklare Zielvorstellungen
- Hohe Komplexität
- Sprachbarrieren
- Schlechte Qualität
- Veränderliche Anforderungen
- Informationsverlust (Rupp/Sophisten 2014, 24)

Im Laufe des Projekts kommen noch dazu:

- Mangelhafte Berücksichtigung des gesamten Entwicklungsprozesses eines Softwareprodukts
- Schwierigkeiten im Umgang mit Änderungen (Partsch 1991, 15)

Unklare Zielvorstellungen entstehen vor allem dann, wenn nicht alle Stakeholder miteinbezogen, bzw. nicht alle Erwartungen berücksichtigt werden. Der Komplexitätsgrad der Systeme, bzw. des Zusammenspiels zwischen unterschiedlichen

Systemen ist in den letzten Jahren stark gestiegen. Ohne umfangreiche und vollständige Dokumentation verliert man dabei leicht den Überblick. Für den Bibliotheksbereich hat zum Beispiel die Änderung der Bibliothekssoftware Auswirkungen auf mehrere andere Systeme, wie etwa auf den Hochschulschriftenserver oder das digitale Repository.
Die unterschiedliche Verwendung von Begriffen oder nicht klar definierte Aussagen verhindern eine konsistente gut verständliche und allen Anforderungen gerecht werdende Beschreibung von Sachverhalten. Werden Ziele und Anforderungen ständig verändert und werden diese Änderungen nicht professionell gemanagt, verlieren die Beteiligten den Überblick und die Kontrolle. Man spricht dabei auch von „requirements creeping“[9]. Eine mangelnde Dokumentation hat Informationsverlust zu Folge. Deshalb haben ExpertInnen des RE Standards aufgestellt, wie gute Anforderungen auszusehen haben.
Anforderungen existieren auf unterschiedliche Detaillierungseben, von sehr grob bis sehr fein. Anforderungen können an das System gerichtet sein, an das Produkt, die Software, Hardware, aber auch an Testfälle, Handbücher, Protokolle, Dokumente usw. Anforderungen an das fertige System umfassen alle Vorgaben und Eigenschaften, die das implementierte Zielsystem und seine Komponenten betreffen. Dazu gehören:

- Einfachheit des Systems
- Realisierung in Software und/oder Hardware
- Räumliche Verteilung von Komponenten
- Verfügbare oder zu verwendende Geräte
- Schnittstellen (mit anderen Teilsystemen)
- Verwendung vorgegebener Hilfsmittel (Programmiersprachen, Betriebssysteme, Rechner, sonstige Bausteine)
- Dokumentation des Systems
- Überlebensfähigkeit (bei Störungen, Katastrophen usw.)
- Sicherheit
- Physikalische Sicherheit (Grenzwerte, Standards für Anschlußverbindungen, abschaltbare Endgeräte, Verwahrung sicherheitsrelevanter Bänder und Disketten)
- Operationelle Sicherheit (Methoden, die für Verschlüsselung, Modularisierung, und Beschränkung von Datenübertragungen verwendet werden sollen oder die Verfügbarkeit sensitiver Daten betreffen)
- Menschliche Faktoren (Benutzerfreundlichkeit, Qualifikation des Bedienpersonals (Partsch 1991, 36)

[9] https://www.projektmagazin.de/glossarterm/schleichender-funktionszuwachs (abgerufen am 19.2.2017)

Partsch unterscheidet beim Ermitteln der Anforderungen folgende Teilbereiche:

- Durchführung von Bedarfs- und Ist-Analysen
- Definition der Benutzerschnittstelle im Dialog mit Auftraggeber und Benutzer
- Systematische Ermittlung der Information
- Festlegung der Qualitätsmerkmale
- Bestimmung von Entwicklungs- und Zielumgebung
- Beschaffung fehlender Informationen (Partsch 1991, 40f)

In einem ersten Schritt muss immer grundsätzlich gefragt werden, ob überhaupt ein Bedarf vorliegt. Soll ein bereits vorhandenes System durch ein neues ersetzt werden oder das bereits vorhandene System analysiert werden? Dabei ist es sinnvoll, sich auf unterschiedliche Aspekte zu konzentrieren:

- Statische Komponenten des Systems – Durch Beobachtung lässt sich ermitteln, welche Tätigkeiten ausgeführt, welche Gegenstände bearbeitet und welche Werkzeuge und Hilfsmittel dabei benutzt werden.
- Dynamisches Verhalten des Systems – Das dynamische Verhalten eines Systems kann man unter verschiedenen Gesichtspunkten untersuchen, etwa
  - Ort (Was geschieht hier während eines bestimmten Zeitraums?)
  - Personen (Welche Arbeit verrichten sie?)
  - Bearbeitetes Objekt (in welchen Schritten wird das Objekt be- oder verarbeitet?)
  - Informationen (Welche Kanäle durchlaufen sie?)
- Kausale Zusammenhänge im System: Neben den dynamischen Aspekten eines Systems ist es wichtig, das Systemverhalten daraufhin zu untersuchen, welche Ursachen und welche beabsichtigten und unbeabsichtigten Folgen ein Vorgang hat und in welchen ursächlichen zusammenhängen verschiedene Vorgänge stehen.
- Aspekte der Datenverarbeitung: Wichtig zur Erfassung der Anforderungen an ein System sind auch gewisse Aspekte der Datenverarbeitung, die sowohl die bearbeiteten Daten wie auch die datenverarbeitenden Tätigkeiten betreffen. (Partsch 1991, 41)

Diese Aspekte müssen natürlich noch weiter verfeinert werden, z.B. im Bereich der Benutzerschnittstelle, der Arten der Datenverarbeitung und der Benutzerprofile. Wichtig ist, dass diese Informationen systematisch ermittelt werden und nicht zufällig an die EntwicklerInnen weitergegeben werden.

2.5.2 Arten an Anforderungen

In der Literatur finden sich unterschiedliche Einteilungen von Anforderungen.
Oft werden funktionale und nicht-funktionale Anforderungen unterschieden. Erstere beziehen sich auf die funktionalen Aspekte des Systems, es wird die Frage gestellt: „Was tut ein System?“ Nicht-funktionale Anforderungen, auch „constraints“ genannt, lassen sich in quantifizierbar und nicht-quantifizierbar unterscheiden, hier ist zu fragen „Wie soll das geplante System die gestellten Aufgaben erfüllen?“ Es geht dabei um zusätzliche Eigenschaften wie Zuverlässigkeit, Robustheit, Wartbarkeit, das Verhalten beim Ausführen, Adaptierbarkeit, Kompatibilität usw. (siehe auch Partsch 1991, 35)
Wenn man sich mit nicht-funktionalen Anforderungen beschäftigt, ist jedoch ebenfalls Kenntnis über das funktionale Verhalten erforderlich.
IREB unterscheidet beispielsweise funktionale Anforderungen, Qualitätsanforderungen und Randbedingungen.
Rupp berücksichtigt nicht nur die Art der Anforderung, sondern auch den Inhalt und schlägt folgende Einteilung vor:

- Funktionale Anforderungen
- Technologische Anforderungen
- Qualitätsanforderungen
- Anforderungen an die Benutzungsoberfläche
- Anforderungen an sonstige Lieferbestandteile
- Anforderungen an durchzuführende Tätigkeiten
- Rechtlich-vertragliche Anforderungen (Rupp/Sophisten 2014, 17)

Funktionale Anforderungen

„Funktionale Anforderungen beschreiben die Fähigkeiten eines Systems, die ein Anwender erwartet, um mit Hilfe des Systems ein fachliches Problem zu lösen. Die Anforderungen werden aus den zu unterstützenden Geschäftsprozessen und den Ablaufbeschreibungen zur Nutzung des Systems abgeleitet“. (V-Modell XT 2015, 137)

Funktionale Anforderungen lassen sich in kleine Schritte zerlegen, der Ablauf kann beschrieben werden. Dazu zählen etwa Transaktionen, die durch den Anwender ausgelöst werden, wie beispielsweise die Ein- oder Ausgabe von Daten, Berechnungen

oder das Ausdrucken. Die Transaktionen können jedoch auch durch andere Systeme ausgelöst werden, wie etwas durch das Abfragen von Daten. Außerdem können Daten von außen archiviert werden. Aber auch sicherheitsrelevante Aktionen, wie etwa das automatische Logout oder die Passwortkontrolle zählen zu den funktionalen Anforderungen.

Nichtfunktionale Anforderungen

„Nicht-funktionale Anforderungen beschreiben Anforderungen an das System, die nicht-fachlicher Natur sind, jedoch entscheidend zur Anwendbarkeit des Systems beitragen. Sie definieren beispielsweise Qualitätsanforderungen, Sicherheitsanforderungen oder Performanceanforderungen. […] Nicht-funktionale Anforderungen definieren grundlegende Eigenschaften eines Systems, die im Architekturentwurf berücksichtigt werden müssen. Sie können zur Abschätzung der Entwicklungskosten herangezogen werden und sollten, soweit möglich, messbar beschrieben sein. Zur einfachen Strukturierung der Anforderungen werden diejenigen Anforderungen, die nicht eindeutig zu den funktionalen Anforderungen gehören, den nicht-funktionalen Anforderungen zugeordnet.“ (V-Modell XT 2015, 137)

Rupp und Geißel verweisen auf die Wichtigkeit immer auch großen Wert auf die nichtfunktionalen Anforderungen zu legen. Werden diese nicht berücksichtigt, ist die positive Umsetzung des Projekts gefährdet: „Nur eine umfassende Analyse der funktionalen UND der nicht-funktionalen Anforderungen wird zu einer erfolgreichen Systementwicklung führen!“ (Rupp/Sophisten 2014, 267)
Während funktionale Anforderungen beschreiben, was ein gewünschtes System können muss, versteht man unter nichtfunktionalen Anforderungen jene Anforderungen, die nicht durch funktionale Anforderungen abgedeckt sind, vor allem jene, die sich mit der Qualität der Anforderungen befassen, bzw. mit zusätzlichen Leistungen, wie beispielsweise Anleitungen. Sie definieren laut Rupp und Geißel „Randbedingungen, die den Entwicklern und Software-Zulieferern bei der Realisierung der funktionalen Anforderungen Freiheitsgrade entziehen“ (Rupp/Geißel 2007, 1), da sie zum Beispiel Funktionalitäten sehr genau beschreiben. Eine einheitliche Definition existiert jedoch nicht, da nichtfunktionale Anforderungen aus verschiedenen Perspektiven betrachtet werden können.

Nicht-funktionale Anforderungen richten sich nicht nur an das zu entwickelnde System selbst, sondern auch an Umgebungen und Bedingungen der Umwelt.
Oft werden diese nichtfunktionalen Anforderungen vernachlässigt, was unter anderem dazu führen kann, dass:

- das System zu umständlich zu bedienen ist,
- seine Performance zu schwach ist,
- es nicht in die bestehende Systemumgebung integriert werden kann, oder
- seine Architektur notwendige Änderungen am System nicht zulässt. (Rupp/Geißel 2007,1)

Diese Punkte entscheiden jedoch darüber, ob ein Kunde mit dem Produkt zufrieden ist oder nicht. So steht heute oft die intuitive Bedienbarkeit im Vordergrund. Im Bibliotheksbereich sind unter anderem die heutigen Suchgewohnheiten der BenutzerInnen zu bedenken. Eher komplexe Suchmöglichkeiten, die zwar ein exaktes Trefferergebnis liefern, treten zugunsten eines einfachen Suchschlitzes immer mehr in den Hintergrund, die Suche soll rasch und zwar von den unterschiedlichen mobilen Endgeräten aus funktionieren. Damit auch solche Vorgaben erfüllt werden können, müssen sie natürlich im Vorfeld definiert worden sein. Während die funktionalen Anforderungen oft genau definiert sind, werden die nichtfunktionalen Anforderungen oft „aus dem Bauch heraus“ beschrieben (Rupp/Geißel 2007, 2), sind damit wenig präzise formuliert und können danach kaum getestet werden. Da aber Usability und Performanz von den nichtfunktionalen Anforderungen abhängt und damit die Zufriedenheit der KundInnen gesichert ist, sollte viel Augenmerk darauf gelegt werden.
Auch Zastrow und Wagner verweisen auf die Wichtigkeit der nichtfunktionalen Anforderungen:

„Es zahlt sich aus, den nichtfunktionalen Anforderungen einen ebenso hohen Stellenwert einzuräumen wie den funktionalen Anforderungen. Erfüllt das fertig entwickelte Produkt z.B. die rechtlichen Vorgaben nicht, kann das Produkt im schlimmsten Fall nicht auf den Markt gebracht werden. (Zastrow/Wagner 2016, 10)

Zastrow und Wagner machen auch auf den so genannten „Joy of Use“ aufmerksam:

„Als Königsdisziplin der Benutzerfreundlichkeit gilt der ‚Joy of Use‘. Dieser Fachbegriff bedeutet, dass die Bedienung nicht nur angenehm sein, sondern Freude bereiten soll. Soll die Software diese Bedingung erfüllen, ist frühestmöglich der Rat von speziell geschulten Usability-Experten einzuholen. Diese kommen aus

unterschiedlichen Fachrichtungen und sind entweder im Bereich der Psychologie oder im Umfeld von Web- und Grafikdesignern zu finden.“ (Zastrow/Wagner 2016, 2)

Nichtfunktionalen Anforderungen, die sich aus dem Systemkontext und aus der Norm für Softwarequalität nach ISO/IEC 2500 ableiten lassen, können laut Zastrow und Wagner wie folgt gegliedert werden:

- Systemtechnische Anforderungen: Anforderungen aus bestehender oder genutzter Hardware, Software und Infrastruktur (z. B. Netzwerk)
- Anforderungen der Erfüllung bestimmter Vorschriften: Anforderungen aus externen und internen Regelwerken
- Softwarequalitäts-Anforderungen: Anforderungen, die die Performanz, Zuverlässigkeit, Usability, Stabilität und Angemessenheit betreffen
- Datenanforderungen: Anforderungen technischer Natur zur Datenspeicherung und –verarbeitung
- Archivierungsanforderungen: Anforderungen, die aus Vorgaben zur Archivierung hervorgehen
- Sicherheitsanforderungen: Anforderungen aus Datenschutz und Datensicherheit, sowie Arbeitssicherheit
- Betriebsanforderungen: Anforderungen für die Betriebsführbarkeit, beispielsweise Vorgaben zur Datensicherung und –wiederherstellung
- Lizenzierungsanforderungen: Anforderungen, die darauf abzielen, das Softwareprodukt lizenzierbar zu machen
- Portierungs- und Migrationsanforderungen: Anforderungen zur Plattformunabhängigkeit
- Rollout- und Schulungsanforderungen: Anforderungen, die die Einführung der Software in den Regelbetrieb ermöglichen oder erleichtern
- Dokumentationsanforderungen: Anforderungen an Entwicklungs-, Benutzer-, Administrations- und Betriebsdokumentation
- Sonstige Anforderungen: branchen- oder unternehmensspezifische Anforderungen, die nicht von den oben genannten Themengebieten abgedeckt wurden (Zastrow/Wagner 2016, 8f)

Technologische Anforderungen

Hier geht es vor allem um Anforderungen, die einen direkten Einfluss auf die in der Entwicklung zu verwendenden Technologien haben. Einerseits sind dies Anforderungen zur technologischen Einbettung des Systems in seine Umwelt und andererseits Vorgaben von Lösungen für die Realisierung. Dazu gehören unter anderem klimatische Umgebungen des Systems, elektrische Schnittstellen, mechanische Schnittstellen, Software-Schnittstellen oder Vorgaben an Komponenten, wie beispielsweise Hard- oder

Software. Um die technologischen Anforderungen ermitteln zu können, ist es sinnvoll, sich an jene Stakeholder zu wenden, die bereits Kenntnisse, bzw. Erfahrungen mitbringen. Auch Systemarchäologie kann dabei hilfreich sein. Für die Dokumentation bieten sich hier Tabellen an, da diese sehr übersichtlich sind und zum Teil auch weiter verwendet werden können. Auch technische Zeichnungen, Schaltpläne usw. können wertvoll für die Dokumentation sein.

Qualitätsanforderungen

Qualitätsanforderungen geben Auskunft über die Güte des Produkts, des Prozesses oder der beteiligten Personen. Laut der Norm ISO/IEC 2500 [ISO05][10] unterteilen sich diese Anforderungen in:

- Anforderungen an die Änderbarkeit: hier werden Vorgaben in Bezug auf den Aufwand und die Auswirkungen, die mit Änderungen oder dem Prozess verbunden sind, definiert
- Anforderungen an die Benutzbarkeit: Hier wird unter anderem der Aufwand für das Erlernen des Umgangs mit dem Systems vorgegeben, bzw. wird auch beschrieben, inwieweit Standards eingehalten werden müssen.
- Anforderungen an die Effizienz: Dabei steht das Leistungsniveau, das Zeitverhalten und das Verbrauchsverhalten im Vordergrund, zum Beispiel wird hier festgelegt, wie hoch der Stromverbrauch sein darf, bzw. wie viel Zeit zwischen Aktionen vergehen darf.
- Anforderungen an die Funktionalität: Darunter fällt beispielsweise die Angemessenheit, Richtigkeit, Interoperabilität, die Sicherheit und die Ordnungsgemäßheit, wie beispielsweise die Rechengenauigkeit
- Anforderungen an die Übertragbarkeit: Dabei wird nach der Anpassungsfähigkeit des Produkts gefragt
- Anforderungen an die Zuverlässigkeit (Reliability): Robustheit, Reife, Fehlertoleranz, Wiederherstellbarkeit und Konformität, also die Stabilität eines Systems, stehen hier im Vordergrund. Dabei wird auch festgehalten, wie das System mit Fehlern umgeht, dass es zum Beispiel nach einem Systemausfall wieder automatisch zur letzten stabilen Variante zurückführt, oder etwa wie genau Berechnungen durchgeführt werden. Das betrifft eine zuverlässige Eingabeverarbeitung, zuverlässige Ausführung und zuverlässige Diagnostik.

[10] Siehe http://iso25000.com/index.php/en/iso-25000-standards (abgerufen am 19.2.2017)

Für die Ermittlung dieser Anforderungen eignen sich Dokumente früherer Qualitätsanforderungen sehr gut. Qualitätsanforderungen können mittels UML, mathematischer Gleichungen oder Diagrammen dokumentiert werden.

Anforderungen an die Benutzeroberfläche

Bei diesen Anforderungen stehen die Optik, die Akustik und die Haptik im Vordergrund. Dazu zählen Bedienkonzepte, Rahmenbedingungen für die Gestaltung der Benutzeroberfläche, wie etwa wie flexibel das System auf Bildschirmgrößen, unterschiedliche Endgeräte, oder Corporate Design-Vorgaben sein muss und Bedienelemente, wobei auch zu prüfen ist, ob und wenn ja, welche Standards einzuhalten sind und ob z.B. gesetzliche vorgeschriebene Warnhinweise anzubringen sind. Hier ist es sinnvoll, zunächst zu überlegen, wie detailliert diese Anforderungen beschrieben werden sollen.

Für die Dokumentation eignen sich bildliche Darstellungen, Prototypen, Tabellen und natürlichsprachliche Dokumentationen. Dabei ist mit den AuftraggeberInnen zu vereinbaren, was als rechtlich verbindlich gilt und was nicht. Mittels Farbskalen, Bildern, Skizzen und Fotos kann die bildliche Gestaltung vorgegeben werden. Prototypen, wie beispielsweise anklickbare Software-Oberflächen sind für die meisten Stakeholder eingängig und direkt erfahrbar. Allerdings ist die Herstellung der Prototypen ziemlich aufwändig, auch sollte man darauf achten, dass die Features des Prototyps wirklich den Anforderungen entsprechen. Getestet werden sollte der Prototyp von jenen Personen, die das System später auch nutzen.

Anforderungen an sonstige Lieferbestandteile

Darunter versteht man sämtliche Produkte, die für den Betrieb und die Anwendung des Systems wichtig sind. Darunter fallen Dokumentationen, Benutzer-, Installations- und Wartungshandbücher, Schulungsunterlagen, Installationssoftware, bzw. Werkzeuge zur Montage von Komponenten, Trainingsmaterial uvm.

Anforderungen an durchzuführende Tätigkeiten

Darunter werden Entwicklungstätigkeiten, bzw. Tätigkeiten, die für die Einführung und den Betrieb des Produkts wichtig sind, verstanden. Dazu zählen auch Abnahmetests, Durchführung von Besprechungen, die fachliche und die betriebliche Abnahme und die Terminfestlegungen. Diese Anforderungen sind meist im Projekthandbuch festgeschrieben oder in Vereinbarungen zwischen Auftraggeber und Auftragnehmer. Tätigkeiten, die nach Projektabschluss zu tätigen sind, wie beispielsweise die Wartung oder der Support des Systems werden meist in einem eigenständigen Dokument festgehalten. Wichtig dabei ist, dass der Zusammenhang nicht verlorengeht, deshalb sollten diese Anforderungen immer gemeinsam dokumentiert werden.

Rechtlich-vertragliche Anforderungen

Dazu zählen Zahlungsmodalitäten, Vertragsstrafen, der Umgang mit Änderungen von Anforderungen, zusammengenommen sämtliche Rechte und Pflichten in Bezug auf Entwicklung und Verwendung des zu erstellenden Produkts, bzw. auch welche Konsequenzen das Nichteinhalten hat.
Die rechtliche Verbindlichkeit gibt den Grad der Bedeutung an, die den Anforderungen von den Stakeholdern beigemessen werden. Dabei handelt es sich im Wesentlichen darum, welche Teile eines Vertrags im Fall des Falles rechtlich einklagbar sind. Legt man fest, welche Anforderungen unbedingt sofort und welche evtl. in einer späteren Version umgesetzt werden müssen, fällt auch eine Priorisierung leichter. Die Internet Engineering Task Force (IETF), die sich um die sich um Standards im Internet bemüht, schlägt die Verwendung bestimmter Schlüsselwörter vor, wie beispielsweise „must“, „must not“, „requires“, „may“, „optional“ usw. Laut Rupp reichen drei Schlüsselwörter völlig aus: „shall“ („muss“) drückt eine Verpflichtung zur Umsetzung aus, „should“ („sollte“) zeigt den Wunsch auf und „will“ („wird“) zeigt die Absicht an. Eventuell wird auch „darf nicht“ nötig sein. (Rupp/Sophisten 2014, 18) Wichtig ist, dass die Bedeutung dieser Worte in der Anforderungsspezifikation dokumentiert ist. „May“ bedeutet etwa,

dass ein bestimmtes Verhalten auftreten darf. Diese Art von Anforderungen wird ebenfalls in einem eigenen Dokument niedergeschrieben. Rechtliche Anforderungen umfassen jedoch auch Punkte über Rechtseinräumungen, Gewährleistungen, Verschwiegenheitspflicht usw. Oft können diese Anforderungen gut wiederverwendet werden.

Es besteht auch die Möglichkeit Qualitätsanforderungen mit bestimmten funktionalen Anforderungen zu konkretisieren. Ein Beispiel dafür wäre die Barrierefreiheit eines Systems, die durch zusätzliche Steuerelemente mittels Tastatur erreicht werden würde. Die Anforderung, nicht nur mittels Maus navigieren zu können, würde mittels funktionaler Anforderungen beschrieben werden.
Da die Qualitätsanforderungen sehr oft mit funktionalen Anforderungen in Beziehung stehen, ist darauf zu achten, beide getrennt voneinander zu definieren und auch die Bezüge zwischen den beiden Anforderungsformen zu dokumentieren. (siehe auch Pohl/Rupp 2015. 11) Qualitätsanforderungen werden meist in einer natürlichen Sprache niedergeschrieben.
Randbedingungen, bzw. Rahmenbedingungen können von den Projektbeteiligten nicht beeinflusst werden, sie betreffen oft den Zeitraum oder die Art der Umsetzung und engen den Spielraum eines Entwicklungsprozesses ein. Randbedingungen beziehen sich auf Anforderungen, auf die beim Entwickeln des Systems zu achten ist, z.B. auf Gesetze und Normen, gewisse Qualitätssicherungsmaßnahmen usw.
Ein Klassifizierungsschema für Anforderungen ist das so genannte FURPS-Schema:

Functionality
Usability
Reliability
Performance
Supportability[11]

Unter Usability, bzw. Benutzerfreundlichkeit werden Eigenschaften wie Bedien- oder Benutzbarkeit verstanden. Dazu zählen Anforderungen, die eng mit e-Accessibility

[11] IT-Wissen. Das große Online-Lexikon für Informationstechnologie, http://www.itwissen.info/definition/lexikon/FURPS-funcionality-usability-reliability-performance-supportability.html (19.2.2017)

zusammenhängen, wie etwa dass die Funktionalitäten des Systems auch per Spracheingabe gesteuert werden können oder dass nicht nur die Maus sondern auch die Tastatur als Eingabegerät verwendet werden kann. Zur Usability zählen aber auch ästhetische Faktoren, wie beispielsweise die Farbgebung oder bestimmte Anforderungen an Konsistenz, wenn etwa Styleguides eingehalten werden müssen. Usability-Engineering beschäftigt sich vor allem wie Systeme brauchbar gestaltet werden können, wenn keine geeigneten Stakeholder zur Verfügung stehen. Man achtet dabei auf grafische Eigenschaften des Produkts, auf das Corporate Design, auf die Gestaltung der Hardware, greift auf Ergebnisse der Marktforschung und auf Branchentrends zurück. Vielfach kreiert man auch fiktive Personen, die Personas genannt werden. (siehe Rupp/Sophisten 2014, 525) Man erstellt einen Persona-Steckbrief für eine fiktive Person mit einem erfundenen Namen, der die Ziele, Einstellungen und Motivationen vermerkt. Ein erfundener Name, Alter, Familienstand, Verdienst, Wohnort und Herkunft werden ebenfalls angegeben, zusätzlich fiktive Aktivitäten und persönliche Einstellungen. Im Bibliotheksbereich könnte es sich beispielsweise um eine Person handeln, die von den neuen technischen Möglichkeiten fasziniert ist. Auch Fähigkeiten, das Bildungsniveau, Vorkenntnisse in unterschiedlichen Bereichen, bzw. ein mentales Modell einer zukünftigen Anwendung, wie sich die Persona dies vorstellen könnte, werden dokumentiert. Zuletzt wird auch der Nutzungskontext eingetragen. Davon lassen sich Anforderungen für das jeweilige Produkt ableiten. Somit können fehlende oder nicht verfügbare Stakeholder ergänzt werden. Die Persona wird wie alle anderen in die Stakeholderliste aufgenommen.
Aus dem Persona-Steckbrief werden Szenarien entwickelt, die daraus folgenden funktionalen und nichtfunktionalen Anforderungen werden verfeinert. Sämtliche Ergebnisse fließen in die jeweiligen Dokumentationen mit ein.
Beim Usability-Engineering werden so genannte Nutzungsabläufe aus der Perspektive des Users anhand von Aktivitätsdiagrammen erstellt. Bei der Qualitätssicherung sollten Personas durch reale Personen vertreten werden.

Bei der Performance (Geschwindigkeit) wird festgehalten, wie rasch das System arbeitet, was mit Daten passiert, die nicht mehr benötigt werden, wie rasch ein Datentransfer durchgeführt wird, wie rasch Antworten gegeben werden usw.
Zur Supportability (Wartbarkeit) zählen die Testbarkeit eines Systems, die Frage, inwieweit das System erweiterbar ist, wie es um die Anpassbarkeit und Konfigurierbarkeit steht, wie das System installiert werden kann und ob es lokalisierbar ist, also zum Beispiel in asiatische Sprachen übertragen werden kann.
Der Grund, warum nichtfunktionale Anforderungen oft von den Stakeholdern nicht oder nicht ausreichend beschrieben werden, liegt unter anderem auch darin, dass viele glauben, dass die EntwicklerInnen selbst am besten wissen, was beispielsweise Userfreundlichkeit bedeutet. Da dies für unterschiedliche Nutzergruppen verschieden aussehen kann, kommt es zu einer nicht genauen Spezifikation, bzw. die nichtfunktionalen Anforderungen fehlen gänzlich. Nichtfunktionale Anforderungen sind auch oft schwerer zu formulieren als funktionale Anforderungen. Auch lassen sich diese Anforderungen nicht so einfach testen, es müssen erst Prüfprozeduren definiert werden. Nicht zuletzt sind nichtfunktionale Anforderungen oft teuer, da sie die gesamte Architektur des Systems beeinflussen.
Oft werden nichtfunktionale Anforderungen auch zu spät genannt, was die Umsetzung ebenfalls erschwert. Rupp und Geißel empfehlen daher, bei jedem Projekt die Vollständigkeit der nichtfunktionalen Anforderungen zu prüfen, diese sehr präzise zu formulieren und in eine Referenzdatenbank, die bei einem späteren Projekt wiederverwendet werden kann, einzutragen.
Beim Wiederverwendungskonzept von Rupp und Geißel (Rupp/Geißel 2007, 3) werden nichtfunktionale Anforderungen gesammelt und in einer Referenzdatenbank abgelegt. Sie schlagen vor, einen „Wiederverwendungsmanager“ einzusetzen, der die Referenzdatenbank pflegt und die Anforderungen so ablegt, dass sie wieder verwendbar sind. Dazu gehört auch, dass Projektspezifika durch Platzhalter (tbd, to be defined) ersetzt werden. So wird bei einem neuen Projekt auf den ersten Blick ersichtlich, welche Teile der Anforderungen noch zu spezifizieren sind. Sinnvoll ist auch, nichtfunktionale Anforderungen zu klassifizieren. Damit sind jene leichter aufzufinden, die auch in einem neuen Projekt eine Rolle spielen. Dabei muss vor allem auf Redundanzen,

Überschneidungen und Inkonsistenzen geachtet werden. Inzwischen gibt es einige Rahmenwerke, die man dabei heranziehen kann. Bei der Wiederverwendung von nichtfunktionalen Anforderungen, wird in einem ersten Schritt in der Datenbank nach passenden Kategorien und Unterkategorien gesucht, die mit Platzhaltern versehenen Anforderungen werden für das aktuelle Projekt spezifiziert und mit dem Auftraggeber und den jeweiligen Stakeholdern besprochen. In dieser Phase kann es noch zu Anpassungen der Anforderungen kommen, etwa beispielsweise bei den Zugriffszeiten. Im einfachsten Fall reicht ein Kopieren der schon formulierten Anforderungen aus, wenn Anforderungen für jedes zu entwickelnde System gelten, kann darauf referenziert werden oder es wird spezifiziert. Diese Anforderungen landen erneut in der Referenzdatenbank, die somit immer größer und besser ausgearbeitet wird.
Erleichternd ist, dass sich nichtfunktionale Anforderungen bei unterschiedlichen Systemen oft gleichen und bei nächsten Projekten wieder übernommen werden können.
Eine Schwierigkeit bei Anforderungen aus dem nichtfunktionalen Bereich liegt darin, dass Anforderungen festgelegt werden, die die Zukunft betreffen und noch gar nicht testbar sind. (siehe auch Zastrow/Wagner 2016, 9)
Services im Bibliotheksbereich müssen nicht nur gut funktionieren, sie müssen auch so ansprechend gestaltet sein, dass sie von den BenützerInnen auch gerne verwendet werden. Denkt man etwa an Langzeitarchivierung stehen heute den WissenschafterInnen zahlreiche Möglichkeiten auch außerhalb der eigenen Institution zur Verfügung. Es herrscht also ein ziemlicher Konkurrenzkampf zwischen den einzelnen Systemen, den die eigene Institution nur gewinnen kann, wenn sie sich möglichst eng an die Wünsche und Bedürfnisse der eigenen Stakeholder orientiert und sich stets flexibel anpasst.

2.5.3 Quellen der Anforderungen

Zu den Quellen zählen Stakeholder, also Personen oder Organisationen dazu, wie beispielsweise spätere NutzerInnen, BetreiberInnen und EntwicklerInnen des Systems, sowie TesterInnen und AuftraggeberInnen, aber auch Dokumente, wie Visionsdokumente, Produktmarketingpläne, Handbücher und Dokumente von

Vorläufersystemen, Unternehmensrichtlinien, gesetzliche Regelungen, Normen, Standards und auch Literaturansätze und wissenschaftliche Arbeiten. Anforderungen können aber auch von anderen Systemen kommen, beispielsweise von Vorgängersystemen oder Konkurrenzprodukten, z.B. in Form von Service-Anfragen. Sie können aus bestimmten Prozessen, wie zum Beispiel einem Problem-Management, einem Change-Management, aus einer Geschäftsprozessanalyse oder aus einem Innovationsmanagement eines Systems hervorgehen.
Wichtig ist, dass zunächst die relevanten Anforderungsquellen identifiziert werden. In einem weiteren Schritt werden grobe Anforderungen definiert, die weiter verfeinert werden. Die funktionalen Anforderungen sollten danach von den nichtfunktionalen Anforderungen getrennt werden. Für die Anforderungen können Testfälle beschrieben werden. bei der agilen Vorgehensweise geht man vom so genannten „Test-Driven Development“ aus, die jeweils getesteten Software-Stände werden laufend in die Gesamt-Software integriert. (siehe Rupp/Sophisten 2014, 47) Für die Dokumentation werden meist unterschiedliche Tools verwendet, die miteinander kombiniert werden. Arbeitet man nach der agilen Methode, sollten die Informationen dann vorliegen, wenn sie benötigt werden, da zu frühe Informationen rasch veralten. Man kann dabei zwischen verschiedenen Reifegraden unterscheiden.

- Der Reifegrad 1 bedeutet, dass die Informationen nicht vollständig oder abgestimmt sein müssen.
- Der Reifegrad 2 sagt aus, dass die Informationen verlässlich und belastbar sein sollten
- Der Reifegrad 3 zeigt an, dass die Informationen allen Qualitätskriterien für gute Anforderungen entsprechen. Außerdem sollten sie vollständig und korrekt sein. (siehe Rupp/Sophisten 2014, 48)

Nicht in jedem Arbeitsschritt müssen die Anforderungen bereits im Reifegrad 3 vorliegen.
Nachdem die Rahmenbedingungen des jeweiligen Projekts geklärt sind, zum Beispiel wieviel Zeit und Geld zur Verfügung stehen, beginnt man mit der Ermittlung der Anforderungen, mit der Formulierung und schließlich mit der Prüfung und Bewertung.

Der Delta Ansatz

In der Realität, vor allem auch in Bibliotheken, kommt es oft vor, dass kein völlig neues System entwickelt werden soll, sondern, dass zwar ein Altsystem vorliegt, aber nur veraltete und eventuell unklare Anforderungen, die noch dazu kaum dokumentiert sind.

„Eine Delta-Anforderung ist eine funktionale oder eine nicht-funktionale Anforderung, welche die Lücke von Forderungen zwischen Alt- und Neusystem/erweitertem System schließt, ohne dass dabei die Funktionalitäten des Altsystems in seiner Gesamtheit verstanden und erfasst werden müssen.“ (siehe auch Rupp/Sophisten 2014, 514).

Es werden also zunächst nur Anforderungen dokumentiert, die das System ändern oder erweitern. Bereits umgesetzte Anforderungen, die von der Änderung nicht direkt betroffen sind, aber verstanden werden müssen, werden ebenfalls dokumentiert. Diese Methode bietet sich für nicht ausreichend dokumentierte Systeme an, die noch dazu unüberschaubar sind. Konkret muss zunächst überlegt werden, wie die ideale Dokumentenstruktur aussehen soll. In dieses Gerüst werden nach und nach die Änderungen, im Idealfall basierend auf einem Element der Altspezifikation, eingetragen. Dokumentiert kann in natürlicher Sprache, mittels Use-Case-Spezifikation oder Aktivitätsdiagramm, oft werden zunächst nur Fragmente dokumentiert.

2.5.4 Einbinden der Stakeholder

Dieser Schritt ist sehr wichtig für den Erfolg des Projekts. Zunächst muss genau überlegt werden, wer zu den Stakeholdern zählt. Hier darf kein Aspekt vernachlässigt werden. Jede Unschärfe, bzw. jede Nichtberücksichtigung würde später zu Verzögerungen und zu erhöhten Kosten führen. In der Literatur wird deshalb auch empfohlen, eine Checkliste an relevanten Stakeholdern anzulegen und sich von diesen weitere wichtige Personen nennen zu lassen. (siehe auch Pohl/Rupp 2015, 22).
Zu den Stakeholdern zählen auf jeden Fall die zukünftigen User, die Qualitätsmanager, die Projektverantwortlichen, Sicherheitsverantwortliche, andere Projekte, VertreterInnen vom Support oder Marketing. Dabei ist es von Vorteil, das zukünftige System und die damit befassten Personen, bzw. auch jene, die in benachbarten Bereichen arbeiten, zu identifizieren. Die Identität der Stakeholder kann über die Visualisierung des Systems geschehen, aus der Erfahrung aus vorangegangenen

Projekten stammen, bzw. aus Gesprächen. Sinnvoll ist es auch, schon gefundene Stakeholder nach weiteren maßgeblichen Personen in diesem Kontext zu befragen. Auch während des Arbeitsprozesses ist es wichtig, mögliche neue Stakeholder miteinzubeziehen. In manchen Fällen empfiehlt es sich eine Stakeholdervereinbarung abzuschließen. Dort werden die Aufgaben der Stakeholder definiert, die Pflichten und Rechte auf beiden Seiten, die Verantwortungsbereiche und die Weisungsbefugnisse. Es sollte jedoch auch überprüft werden, ob die gefundenen Stakeholder wirklich jene sind, die man für die Entwicklung benötigt, ob sie einen Bezug zum Produkt haben und die ihre Bedürfnisse auch klar kommunizieren können. Da nicht alle Personen, die irgendwie mit dem Produkt zu tun haben, in die Diskussionen einbezogen werden können, empfiehlt es sich, für jede Gruppe an Stakeholdern RepräsentantInnen auszuwählen.

Hilfreich ist, wie schon oben erwähnt, das Anlegen einer Stakeholderliste, die den Namen, die Rolle, zeitliche Verfügbarkeit während der Projektlaufzeit, Relevanz des Stakeholders, das Wissensgebiet und evtl. Ziele und Interessen, die sich auf das Projekt beziehen, umfassen sollen. Sinnvoll ist auch, die Entscheidungsbefugnis festzuhalten, Beziehungen zu anderen Stakeholdern, den Grund, warum diese Person als Stakeholder ausgewählt wurde, die Art der Kommunikation und die Mitwirkung während der Qualitätssicherung und Freigabe der Anforderungen (siehe auch Rupp/Sophisten 2014, 82). Um noch rascher die richtigen Stakeholder zur richtigen Zeit im Projekt einbinden zu können, wird auch oft empfohlen, die Stakeholder mit Klassifizierungen zu versehen, beispielsweise wie motiviert sie sind und welchen Einfluss sie haben. Zusätzlich kann man bei größeren Projekten auch Gruppen an Stakeholdern erstellen, die auch für zukünftige Projekte relevant sein können.

Damit der Umgang mit den Stakeholdern reibungslos verläuft, muss der Informationsfluss in Gang gehalten werden, es bieten sich dafür regelmäßige Statusbesprechungen an. Gleichzeitig sollte darauf geachtet werden, alle Stakeholder am Projekt zu beteiligen und ihnen ggf. auch Aufgaben zu geben und sie nicht nur als Projektbetroffene wahrzunehmen. So fühlen sich die Stakeholder auch mitverantwortlich und tragen aktiv zu einem guten Arbeitsklima und zu einem reibungslosen Ablauf bei. Man vermeidet dadurch auch, dass sich Personen kritisch

gegenüber dem Projekt äußern, bzw. hilft es auch, Ängste vor dem neuen System abzubauen, vor allem bei jenen Personen, die mit dem Vorgängersystem gut zurechtkamen oder negative Erfahrungen mit früheren Projekten gemacht haben. In dieser Phase ist es wichtig, dass der RE die Projektleitung gut unterstützt. Eine weitere Aufgabe des RE ist es, Formalitäten mit den Stakeholdern zu klären, beispielsweise Verantwortungsbereiche abzustecken oder abzuklären, welche Weisungsbefugnisse, die einzelnen Stakeholder haben. Auch die Kommunikationswege sollten geklärt werden, ebenso der Umgang mit Feedback. Dabei können, je nach Organisation mündliche Abmachungen ausreichen, es kann aber auch sein, dass schriftliche Vereinbarungen getroffen werden müssen. Wichtig ist jedoch, dass die Vorgesetzten den jeweiligen Vereinbarungen zustimmen.

RE bedeutet sich in das jeweilige Fachgebiet einzuarbeiten, die Sprache der Stakeholder zu sprechen, die Ergebnisse verständlich machen zu können, ein Anforderungsdokument erstellen und dafür zu sorgen, dass das System den Ansprüchen der der Stakeholder gerecht wird. Wichtig ist, dass ein respektvoller Umgang mit den Stakeholdern herrscht. Auch die Planung der Kommunikation und der Termine obliegt den RE. Für die Stakeholder bedeutet das, dass sie die Aufgabe haben, die RE in das Fachgebiet einzuführen, ihnen die Anforderungen mitzuteilen, die so gut wie möglich formuliert werden. Stakeholder sollten die Entscheidungen auch fristgerecht treffen, müssen aber die Einschätzung der RE die Kosten und die Machbarkeit respektieren. Stakeholder haben aber auch die Aufgabe, die Anforderungen zu priorisieren, Prototypen zu testen und zu dokumentieren, Änderungen so rasch wie möglich zu diskutieren und die vom RE vorgeschlagenen Prozess akzeptieren.

Für jeden Stakeholder soll außerdem festgehalten werden, welche Erhebungstechniken verwendet werden und wie der Erhebungsplan aussieht. Sämtliche Informationen, die mit dem Stakeholder und den Anforderungen in Zusammenhang stehen, sollten verwaltet werden. Sämtliche Protokolle und übergebene, bzw. verwaltete Dokumente müssen festgehalten werden.

Empfehlenswert ist auch zu klären, wer unter den Stakeholdern zu den Entscheidungsträgern zählt. Außerdem muss nachvollziehbar sein und festgehalten werden, von wem eine Anforderung stammt. Damit die Stakeholder einer Anforderung

zustimmen, müssen Reviews durchgeführt werden, evtl. mittels Prototypen. Die formale Abnahme von Anforderungen sollte mittels Unterschrift erfolgen.
Wie man sieht, ist die Arbeit mit den Stakeholdern sehr aufwändig und erfordert ein perfektes Zusammenspiel aller beteiligten Personen. Überträgt man dies nun auf die derzeitige Situation von Bibliotheken, genügt es nicht eine/n RE einzusetzen, sondern es ist maßgebend, auch das Umfeld, bzw. alle Beteiligten auf diese Arbeitsweise vorzubereiten.

2.5.5 Ermittlungstechniken von Anforderungen

> *Das Ermitteln von Anforderungen ist ein Akt der Verführung. Sie müssen Ihr Gegenüber dazu verführen, sein Wissen preiszugeben, seine Visionen mit Ihnen zu teilen. Es ähnelt damit einem Flirt, denn der Flirt ist ein Spiel, bei dem man nicht weiß, ob man noch in der Qualifikation ist oder schon im Finale*
> (Rupp/Sophisten 2014, 90)

RE hat die Aufgabe, die Stakeholder dabei zu unterstützen, sich über die Wünsche an die EntwicklerInnen klar zu werden und auch Wissen aus dem Unbewussten und Unterbewussten zutage zu fördern. Dafür ist Wissen über die Grundlagen von Kommunikation nötig, über die Sprache aber auch über geeignete Ermittlungstechniken.

Gelungene Kommunikation als Voraussetzung

Wenn man sich das Shannon-Weaver-Kommunikationsmodell[12] ansieht, erkennt man, dass zwischen Sender und Empfänger zahlreiche Störungen auftreten können. Dazu zählen Mehrdeutigkeiten, kulturelle Unterschiede, implizites Wissen, das fälschlicherweise vorausgesetzt wird, unterschiedliche Sprachen und einiges mehr. Deshalb empfiehlt es sich immer – auch bei der Generierung von Anforderungen – Feedbackschleifen einzubauen um solche Missverständnisse zu erkennen und sie aufzulösen.

Friedemann Schulz von Thun geht in seinem Vier-Seiten-Modell[13] davon aus, dass eine Nachricht nach folgenden vier Seiten interpretiert werden kann:

- Sachinhalt: Hier stehen Daten und Fakten, also Sachinformationen im Fokus
- Selbstkundgabe: Damit ist alles gemeint, was der oder die Sprechende absichtlich oder unbewusst über sich selbst preisgibt.
- Beziehung: Hier wird offenbart, wie der Sender den Empfänger einschätzt
- Appell: Hier wird alles transportiert, was der Sender beim jeweiligen Empfänger auslösen möchte.

---

[12] Siehe etwa https://www.mediamanual.at/mediamanual/workshop/kommunikation/bedeutung/modell01.php (19.2.2017)

[13] Siehe http://www.schulz-von-thun.de/index.php?article_id=71 (19.2.2017)

Es ist nie ausreichend nur auf der Sachebene zu kommunizieren, denn Kommunikation findet stets auf mehreren Ebenen statt. Dies sollte auch bei der Ermittlung von Anforderungen berücksichtigt werden.

Die Wahl der richtigen, bzw. der jeweiligen Situation angepassten Ermittlungstechnik ist sehr wichtig für den Erfolg des Projekts. Meist ist es zielführend, mehrere Ermittlungstechniken in einem Projekt einzusetzen. Welche Technik gerade passt hängt von den Rahmenbedingungen des Projekts ab, von den Stakeholdern, dem vorhandenen bewussten und unbewussten Wissen und den Vorerfahrungen des RE mit der entsprechenden Technik.

Bei der Kano-Methode wird zunächst ermittelt, welche Bedeutung, die Anforderung für die Zufriedenheit der Stakeholder hat. Man unterscheidet dabei drei Kategorien:

1. Selbstverständlich vorausgesetzte Systemmerkmale, bezeichnet man als Basisfaktoren (unterbewusstes Wissen)
2. Explizit geforderte Systemmerkmale werden Leistungsfaktoren genannt (bewusstes Wissen)
3. Systemmerkmale, die der Stakeholder vorerst nicht kennt und während des Prozesses als angenehm und nützlich erkennt, werden Begeisterungsfaktoren genannt. (unbewusstes Wissen) (siehe auch Pohl/Rupp 2015, 24)

Basisfaktoren müssen vom System auf jeden Fall erfüllt werden, da die Stakeholder sonst unzufrieden sind. Da die Basisfaktoren, die durch bereits vorhandene Systeme abgedeckt sind, als selbstverständlich angesehen werden, kann man mit der Berücksichtigung jedoch die Zufriedenheit nicht erhöhen. Fehlt ein Basisfaktor, ist die Unzufriedenheit des Kunden später groß. Ermittelt werden diese Basisfaktoren meist durch Beobachtungstechniken, bzw. durch Verfahren, bei denen man Dokumentationen heranzieht, wie etwa Benutzerhandbücher von vorherigen Systemen. Must-have-Eigenschaften werden vom Kunden oft implizit vorausgesetzt, werden, wenn sie vorhanden sind, kaum positiv bemerkt, führen jedoch zu Unzufriedenheit, wenn sie fehlen.

Zufriedenheit kann man mit der Erfüllung der Leistungsfaktoren erreichen, jeder Leistungsfaktor, der nicht erfüllt wird, lässt jedoch die Zufriedenheit sinken. Ermittelt werden diese Anforderungen meist mit Befragungstechniken, da sie den Stakeholdern bewusst sind.

Begeisterungsfaktoren werden erst vom Stakeholder erkannt, wenn er das System testen kann, oder wenn die RE diese Faktoren vorschlagen. Begeisterungsfaktoren ermöglichen den Vorsprung zu Konkurrenzprodukten. Wenn sich diese Begeisterungsfaktoren am Markt durchsetzen setzt nach und nach eine Gewöhnung ein, die Begeisterungsfaktoren werden zu Leistungsfaktoren und schließlich zu Basisfaktoren. Als Beispiel wäre hier wieder die Suchfunktion zu nennen, die sich in den letzten Jahren stark vereinfacht hat. Auch die Schnelligkeit von Systemen zählt dazu. Von linearen Begeisterungsfaktoren spricht man, wenn sie zu höherer Zufriedenheit führen je mehr sie vorhanden sind.

Eine universelle Technik, wie Anforderungen ermittelt werden sollen, existiert bislang nicht. Pohl und Rupp listen jedoch Einflussfaktoren auf die Wahl der Ermittlungstechnik auf:

- Die Unterscheidung nach bewussten, unbewussten und unterbewussten Anforderungen, die ermittelt werden sollen
- Die Termin- und Budgetanforderungen sowie die Verfügbarkeit relevanter Stakeholder
- Die Erfahrung des Requirements Engineer mit der entsprechenden Ermittlungstechnik
- Die Chancen und Risiken des Projekts (Pohl/Rupp 2015, 26)

Ein erster Schritt bei der Wahl der geeigneten Ermittlungstechnik sollte die Erhebung der Risikofaktoren sein. Diese können durch menschliche Einflüsse hervorgerufen werden, wie beispielsweise durch eine mangelhafte Kommunikation oder durch gruppendynamische Aspekte. Auch organisatorische Einflüsse müssen untersucht werden, dazu zählt z.B. ob es sich um eine Neuentwicklung eines Systems handelt, ob mittels Werkvertrag oder innerhalb einer Festanstellung gearbeitet wird, ob die Stakeholder jederzeit verfügbar sind oder nicht usw. Ist das zu entwickelnde System sehr komplex, empfiehlt sich darüber hinaus die Anwendung von strukturierenden Maßnahmen um die fachlichen Inhalte in leichter verständliche Einzelteile zerlegen zu können.
Wichtig scheint auch die Frage, wie detailliert die Anforderungen sind. Bei abstrakten Anforderungen bieten sich beispielsweise Kreativitätstechniken an, die gemeinsam mit

den Stakeholdern durchgeführt werden und bei denen wichtige Eigenschaften des Systems gemeinsam gesammelt werden können. Sind die Anforderungen weniger detailliert, könnte man mit Befragungs- und Beobachtungstechniken arbeiten. Bei sehr detaillierten Anforderungen bietet es sich eher an, Dokumentationen zu nutzen, bzw. sich bereits existierende Systeme anzusehen. ExpertInnen raten dazu, immer mehrere Techniken zu kombinieren um die Risiken zu vermindern und Schwächen der einzelnen Techniken ausgleichen zu können.
Neben Befragungstechniken und Beobachtungstechniken werden oft auch die Zustandsanalyse und szenariobasierte Techniken genannt. Welche Methode gewählt wird, hängt auch davon ab, ob die Stakeholder verfügbar sind, wieviel Vorwissen existiert und ob eine Dokumentation verfügbar ist.

Beobachtungstechniken

Beobachtungstechniken bieten sich an, wenn die Stakeholder zu wenig Zeit haben oder aus sonstigen Gründen nicht in der Lage sind, ihr Wissen an die RE weiterzugeben.
Bei den Beobachtungstechniken unterscheidet man die Feldbeobachtung, wo die Stakeholder mehr oder weniger passiv bleiben und die Teilnahme an der Arbeit der Stakeholder, wo diese den Beobachtenden auch Informationen geben. Sinnvoll ist dies dann, wenn es weder eine ausreichende Dokumentation gibt, noch eine Darstellung der Anforderungen durch die Stakeholder. Bei der Feldbeobachtung wird das System und die Umgebung beobachtet, wobei darauf zu achten ist, dass die Beobachtung das Verhalten des Systems nicht verändert. Bei dieser Methode müssen im Vorfeld Fragen formuliert werden, es muss genau beschrieben werden, was beobachtet werden soll, der Zeitraum und der Ort müssen definiert sein, bzw. die beteiligten Personen ausgewählt werden. Die Anforderungen müssen auch stets kritisch hinterfragt werden, da sonst eventuell überholte Anforderungen und suboptimale Prozesse festgehalten werden. Als externe Personen haben RE auch die Chance bessere Lösungen zu erkennen, außerdem können sie Basisanforderungen erkennen, die von den Stakeholdern eventuell stillschweigend vorausgesetzt und gar nicht explizit angegeben werden würden. Wichtig ist vor allem sich schon vorher zu überlegen, welches Medium für die Protokollierung verwendet werden soll. Ein Vorteil ist, dass RE die jeweiligen Fachbereiche besser

kennen lernen und davon für weitere Projekte profitieren. Nicht geeignet ist diese Technik, wenn es sich um ein innovatives System oder um Prozesse handelt die nur schwer beobachtbar sind. Diese Technik sollte immer mit anderen Techniken kombiniert werden, da hier nur der Ist-Zustand dokumentiert werden kann. Außerdem besteht die Gefahr, dass alleine das Beobachten schon den Zustand verändert. Wichtig ist hier, sehr feinfühlig zu sein, denn allzu leicht entsteht der Eindruck der Kontrolle. Allerdings kann mit der Beobachtungstechnik auch herausgefunden werden, dass ein System z.B. ineffizient ist. Ebenso können Faktoren ermittelt werden, die den Stakeholdern selbstverständlich sind, oder die sie nur unterbewusst kennen. Ungeeignet ist diese Technik bei Sonderfällen und schwer beobachtbaren Workflows.
Im Beobachtungsprotokoll empfiehlt es sich überraschende Ergebnisse, bzw. ungeklärte Punkte möglichst detailliert festzuhalten. Auch Video- oder Audioaufnahmen können dabei hilfreich sein. Vorstellbar ist diese Methode etwa, wenn man Personen beobachtet, die Daten in die Forschungsdokumentation eintragen.
Beim Teilnehmen am Prozess (Apprecenticing, bzw. „In die Lehre gehen“) unterstützt man die Stakeholder bei der Formulierung ihrer Wünsche, dabei ist es notwendig, dass sich die RE ExpertInnenwissen aneignet. Die Voraussetzung dafür ist natürlich, dass die Stakeholder bereit sind, das Wissen mit den ExpertInnen zu teilen und dass keine Sicherheitsbestimmungen oder rechtliche Gründe dagegen sprechen. Der Vorteil dieser Methode liegt darin, dass der RE nicht von oben herab beobachtet sondern sich mit den Stakeholdern auf eine Ebene begibt. Eingesetzt werden kann diese Technik natürlich nur dort, wo kein Schaden bei unsachgemäßer Bedienung auftreten kann, außerdem muss man viel Zeit und damit auch Kosten aufwenden.
Zusätzlich zu den oben bereits genannten Schritten ist hier natürlich wichtig, sämtliche Aufgaben und Methoden der ExpertInnen festzuhalten und diese, wann immer möglich, auch selbst durchzuführen. Ein Vorteil dieses Verfahrens ist, dass sich RE und Stakeholder auf Augenhöhe begegnen, bzw. die Stakeholder als Meister ihres Faches auftreten und ihr ExpertInnenwissen vermitteln können.

Artefaktbasierte Techniken – Dokumentenzentrierte Techniken

Bei dieser Technik verwendet man Dokumente wieder, die sich bewährt haben und beispielsweise Informationen über abzulösende Systeme bieten können. So kann sichergestellt werden, dass keine Funktionalitäten vergessen werden. Diese Art der Technik sollte jedoch immer mit anderen Techniken kombiniert werden, um keine neueren Anforderungen zu übersehen.
Anforderungen, die zu einem früheren Zeitpunkt erstellt wurden und gut dokumentiert sind, können evtl. wieder verwendet werden – was Zeit und Kosten einspart.
Wenn für das betreffende System bereits eine Lösung existiert, diese aber ersetzt werden soll, empfiehlt es sich eine Zustands- oder Ist-Analyse durchzuführen. Dabei kann man entweder Systemarchäologie betreiben, also Handbücher und Protokolle heranziehen oder bereits bestehende Lösungen wiederverwenden (Re-Use).

Systemarchäologie

Mittels der so genannten Systemarchäologie lassen sich Funktionen eines alten Systems, die nicht mehr bewusst sind, wieder erkennbar machen, indem der Code analysiert wird und geprüft wird, welche Funktionalitäten das System aufwies. Diese Verfahrenstechnik ist sehr aufwändig, führt aber zu detaillierten Anforderungen und garantiert als einzige Technik, dass wirklich sämtliche Funktionalitäten eines alten Systems in ein neues System übernommen werden kann. Bei der Systemarchäologie muss wenigstens auf den Source Code des bestehenden Systems zurückgegriffen werden können und es müssen Fachleute vorhanden sein, die diesen Source Code auch verwenden können. Der Vorgang bei der Systemarchäologie beginnt mit dem Sammeln aller Informationsquellen, wie Spezifikationen, Protokolle, Handbücher, Fehlermeldungen, Source Code usw. Danach wird die Anforderungsspezifikation festgelegt, wobei Berechnungen und interne Abläufe zu berücksichtigen sind. Abgesehen davon dass diese Methode sehr aufwändig ist, besteht auch die Gefahr, dass Fehler oder eher ungünstige Lösungen übernommen werden. Meist wird diese Technik mit anderen Methoden kombiniert.

Re-Use von Anforderungen

Bereits 1991 hatte Partsch gemeint:

„Aus Gründen der Qualitätsverbesserung, aber auch aus rein ökonomischen Erwägungen wird der Aspekt der Wiederverwendung im Rahmen des Requirements Engineering stärkere Berücksichtigung finden müssen. Viel Anforderungsdefinitionen stimmen in wesentlichen Teilen überein, so daß man – entsprechende Standardisierung vorausgesetzt – diese Teile nicht immer wieder neu erstellen müßte, sondern einfach wiederverwenden könnte.“ (Partsch 1991, 305)

Bei der Wiederverwendung werden Anforderungen, die auch für andere Fälle genutzt werden, gesammelt, beschrieben und nach Themen sortiert in einer Datenbank abgelegt. Vor allem nichtfunktionale Anforderungen, Modelle oder Glossareinträge bieten eine gute Basis. Günstig ist es, wenn Anforderungen zu diesem Zweck generalisiert wurden und wenn alles gut dokumentiert wurde, was im Projektalltag oft nicht einfach handhabbar ist. Wird die Chance auf bereits bestehendes Wissen zuzugreifen, nicht genützt, spricht man von „Not-Invented-Here-Syndrom“. (Rupp/Sophisten 2014, 111f) Re-Use bedeutet auch Kosten zu sparen und durch kürzere Entwicklungszeiten konkurrenzfähig zu bleiben. Rasches Feedback steigert wiederum die Qualität.
Die Frage wann eine Wiederverwendung sinnvoll ist, hängt von mehreren Faktoren ab:

- Von der Höhe des Spezifikationslevels. Eine Verfeinerung reduziert allerdings die mögliche Wiederverwendung
- Wenn die Produktpalette des Unternehmens eher eingeschränkt ist
- Wenn es sich nicht um einen fixen Ablauf handelt
- Wenn die Beschreibung technologieunabhängig ist, da sich Technologien sehr rasch ändern (siehe Rupp/Sophisten 2014, 462)

Eventuell müssen in einem Unternehmen neue Rollen geschaffen werden, um Wiederverwendungskonzepte einführen zu können. Auf jeden Fall muss eine mögliche Wiederverwendung rechtlich geprüft werden. Wenn es keine Einwände gibt, kann man folgende Schritte durchführen:

- Zugang zu sämtlichen in einer Datenbank abgelegten Wiederverwendungskandidaten geben. Hier muss eine Person verantwortlich sein
- Bestehende Wiederverwendungskandidaten analysieren, alle spezifischen Teile und konkrete Werte werden durch Platzhalter ersetzt
- Die ausgewählten Kandidaten organisieren. Hier wird nach Art der Information kategorisiert und strukturiert, sowie Dubletten ausgeschieden

- Zuletzt eine Synthese der bestehenden Wiederverwendungskandidaten durchführen: alles was für das neue System wiederverwendet werden kann, wird extrahiert. (siehe auch Rupp/Sophisten 2014, 465ff)

Befragungstechniken

Will man unverfälschte direkte Aussagen von den Stakeholdern erhalten, setzt man am besten Befragungstechniken ein. Diese setzen jedoch voraus, dass die interviewten Personen ihr Wissen gut vermitteln können und genügend Zeit zur Verfügung haben. Bei dieser Technik ist das Geschick der InterviewerInnen ausschlaggebend. Durch Anpassung an das Gegenüber und durch gezieltes Nachfragen können dabei auch Anforderungen ermittelt werden, die nicht offen ausgesprochen werden. Diese Technik ist sehr zeitaufwändig und benötigt viel Geduld von beiden Seiten.
Zastrow und Wagner schlagen vor:

„Als Diskussionsgrundlage kann eine Liste der Use Cases dienen. Projektleiter und Stakeholder nehmen dabei die gleiche Perspektive auf das System ein und erarbeiten gemeinsam die Anforderungen aus der Nutzerperspektive, z.B. indem sie die Workflows zu den jeweiligen Use Cases durchgehen: Was sind z.B. für den Use Case „Nutzer registriert sich am System“ die Voraussetzungen, was passiert im Hintergrund und was ist das Ergebnis?“ (Zastrow/Wagner 2016, 2)

Die Befragung mittels Fragebogen geht sehr rasch, vor allem, wenn er auch online ausgefüllt werden kann. Die Gefahr bei dieser Methode liegt jedoch darin, dass nur abgefragt werden kann, was bereits überlegt wurde und dass es zu Falschinterpretationen kommen kann. Die Formulierung von ausgereiften Fragen kostet im Vorfeld jedoch auch viel Zeit und auch Wissen. Befragungstechniken können überall dort eingesetzt werden, wo das Wissen explizit vorhanden ist. Man unterscheidet zwischen Fragebogen und Interview.
Der Fragebogen eignet sich vor allem dann, wenn eine große Gruppe an Stakeholdern einzubeziehen ist. Dabei ist wichtig zu wissen, dass die Art der Fragen das Ergebnis beeinflussen kann. Im Vorfeld müssen die Fragen gründlich vorbereitet werden, auf eine ausreichende Erläuterung muss ebenso geachtet werden. Geschlossene Fragen eignen sich vor allem für statistische Auswertungen. Eine Skala von 1-10 ermöglicht den Befragten die Einschätzung eines Sachverhalts und offene Fragen sind für

zusätzliche Anmerkungen anzuwenden. Der Fragebogen kann elektronisch oder auch auf Papier ausgefüllt werden, bzw. auch Online eingesetzt werden. Man erreicht mit dem Fragebogen eine große Gruppe an Personen. Die wichtigsten Fragen sollten am Anfang stehen, dann stehen die Chancen höher, dass sie beantwortet werden.
Wichtig dabei ist einerseits die gründliche Vorbereitung der Fragen und die Auswahl der Stakeholder. Die Fragebögen müssen eingesammelt und ausgewertet werden, danach werden Statistiken erstellt und Kommentare gesammelt und zum Schluss werden die Ergebnisse interpretiert. Für implizites Wissen sind Fragebögen nicht geeignet.
Das Interview bietet sich bei einer kleinen Gruppe an wichtigen Stakeholdern an, vor allem wenn noch Details geklärt werden müssen. Die Stakeholder werden bewusst ausgewählt und einzeln befragt, die Fragen vorher gut vorbereitet. Außerdem sollte noch eine protokollierende Person anwesend sein. Das Protokoll sollte danach mit den Stakeholdern abgestimmt werden, eventuelle Änderungen eingearbeitet werden.
Für einen guten Erfolg muss das Interview gut geplant werden, alle Beteiligten sollten genug Zeit zur Vorbereitung haben, auch für das Interview muss genügend Zeit eingeplant werden. Der Ort sollte sorgfältig gewählt werden damit keine Störungen das Interview unterbrechen. Eine gute Vertrauensbasis ist ebenfalls sehr wichtig. Für ein angenehmes Gesprächsklima ist es auch förderlich, den Blick auch auf jene Punkte zu lenken, die gut funktionieren.
Interviews sind recht zeitaufwändig, da es der Vorbereitung, der Durchführung und der Nachbereitung bedarf. Bei der Vorbereitung sollte man nach Möglichkeit folgende Punkte beachten:

- Vertrauen schaffen durch eine professionelle Selbstdarstellung
- Die rechtzeitige Ankündigung des Interviews und Darlegung des Themas
- Der Befragte sollte den Ort bestimmen
- Klarlegen, was mit den Antworten passiert, an wen und in welcher Form sie weitergegeben werden.
- Der Kontext sollte schon im Vorfeld klar abgegrenzt werden. (siehe auch Rupp/Sophisten 2014, 107)

Ein Gruppeninterview kann manchmal rascher die gewünschten Informationen schaffen, andererseits kann es passieren, dass durch gewisse gruppendynamische Prozesse Verzerrungen entstehen und damit das Ergebnis unstimmig ist.
Während des Interviews sollte man sich immer wieder rückversichern, alles richtig verstanden zu haben und ggf. das Verstandene mit eigenen Worten wiederholen, Wichtiges sollte von Unwichtigem getrennt werden.
Das Protokoll sollte möglichst rasch nach dem Interview erstellt werden, evtl. kann man noch Formulierungen verbessern oder Informationen ergänzen. Sämtliche Unterlagen, die verwendet wurden, sollten ins Protokoll aufgenommen werden. Ebenfalls möglichst zeitnah sollte das Protokoll den interviewten Personen zur Freigabe übergeben werden. Somit wird sichergestellt, dass keine Verständnisfehler enthalten sind, gleichzeitig hat der Interviewer das Einverständnis des Interviewten und darüber hinaus erfährt der Interviewte dadurch eine höhere Wertschätzung.
Der größte Vorteil beim Interview liegt wohl darin, dass gezielt auf die Gesprächspartner eingegangen werden kann, außerdem kann man sicher sein, dass die Fragen wirklich beantwortet werden. Es gibt aber auch eine Gefahr: „Die Schwierigkeit ist hier häufig, exotische Anforderungen einzudämmen und die maßgeblichen Aspekte im Auge zu behalten.“ (Zastrow/Wagner 2016, 2)

Szenariobasierte Techniken

Bei den szenariobasierten Techniken werden Storyboards oder Simulation unterschieden, bzw. Use Cases erstellt oder Geschäftsprozessmodellierung betrieben. Storyboards, bzw. Simulationen eignen sich vor allem für die Prüfung von Akzeptanz, bzw. für das Auffinden noch unbekannter Aspekte. Die darzustellenden Abläufe sind bekannt und werden evtl. mit Bildschirmmasken, Powerpointpräsentationen oder Skizzen vorgestellt. Die typischen Abläufe werden in Prosa und mit Skizzen dargestellt. Ein großer Vorteil dabei ist die allgemeine Verständlichkeit, ein Nachteil ist der große Aufwand. Eventuell werden auch Designentscheidungen vorweggenommen oder vom Stakeholder implizit vorausgesetzt.

Auch bei der Erstellung von Use Cases kann man herausfinden, ob es noch unbekannte Abläufe oder Zusammenhänge gibt.

Kreativitätstechniken

Auch Kreativitätstechniken sind manchmal sinnvoll, vor allem dann, wenn es um das Ermitteln von innovativen Anforderungen geht. Sie dienen vor allem dazu eine erste Version des neuen Systems zu entwickeln, eignen sich jedoch weniger für detaillierte Anforderungen.
In der Literatur werden mehrere Kreativitätstechniken unterschieden:

- Brainstorming
- Brainstorming paradox
- Perspektivenwechsel
- Analogietechniken

Brainstorming, eine Entwicklung aus den 1930er Jahren, bietet sich vor allem für eine Gruppe von 5 bis 10 Personen – wenn möglich aus unterschiedlichen Stakeholdergruppen – an. Ideen werden in einer Gruppe, ohne bewertet zu werden zunächst einmal nur von einem Moderator oder einer Moderatorin gesammelt und notiert. Dabei kann man auch über Videokonferenzen miteinander kommunizieren. Die Ideen der TeilnehmerInnen werden von anderen aufgegriffen und weiterentwickelt. Erst dann erfolgt eine Analyse. Bei dieser Technik gewinnt man in relativ kurzer Zeit und ohne viel Aufwand viele unterschiedliche Ideen, die einander evtl. unterstützen und zu neuen Ideen führen können. Vorsicht ist geboten, wenn es sich um eher schwierige Gruppen handelt, bzw. um TeilnehmerInnen, die unterschiedlich dominant auftreten, bzw. verschiedenen Hierarchien angehören. In diesen Fällen empfiehlt es sich die Gruppen zu teilen.
Beim **Brainstorming paradox** werden wie oben beschrieben Ideen gesammelt, mit dem Unterschied, dass darüber nachgedacht werden soll, was nicht zielführend ist. In einem weiteren Schritt wird gemeinsam überlegt, wie man diese Nichtziele vermeiden kann. Der Vorteil dieser Technik ist, dass den TeilnehmerInnen bewusst wird, welche Aktionen es zu vermeiden gilt um ein positives Ergebnis zu erzielen.

Bei der **Methode 6-3-5** entwickeln jeder für sich sechs Ideen und schreibt diese auf eine Karte. Die Karte wird an den Nachbarn weitergeleitet, dieser liest die Ideen durch, lässt sich inspirieren und entwickelt erneut drei Ideen, jede Person muss alle Karten einmal in der Hand gehabt haben. Danach werden die Ideen gesammelt und ausgewertet. Diese Methode vereinfacht die Arbeit der ModeratorInnen. Auch bzgl. Gruppendynamik hat diese Methode Vorteile, da alle TeilnehmerInnen gleich stark eingebunden werden. Diese Methode ist jedoch nicht ganz so spontan wie das Brainstorming, die TeilnehmerInnen können weniger rasch Themen aufgreifen.

Beim **Perspektivenwechsel** arbeitet man mit unterschiedlichen realen oder imaginären farbigen Hüten, die sich die TeilnehmerInnen aufsetzen um dann aus der Sicht eines anderen das Problem zu betrachten. Zielführend ist diese Technik bei Stakeholdern, die eine eingeengte Sichtweise haben.

Mögliche Rollen dabei sind:

- Objektivität und Neutralität (weiß): Fakten und Zahlen
- Persönliches Empfinden und subjektive Meinung (rot): Gefühle, Ängste, Hoffnungen
- Objektive, negative Argumente (schwarz): Zweifel, Bedenken, Risiken
- Objektive, positive Eigenschaften (gelb): Chancen, Pluspunkte, Ziele
- Neue Ideen (grün): beliebige Ideen, ähnlich wie beim Brainstorming
- Prozesskontrolle (blau): Moderation und Anleitung der Ideenfindung (Rupp/Sophisten 2014, 101)

Man erhält einen guten Überblick, für eine detaillierte Beschreibung ist sie jedoch nicht geeignet. Auch für eher introvertierte oder konservative Stakeholder ist sie nicht die richtige Wahl.

Bei der **Analogietechnik** sieht man sich andere Fachgebiete an und überträgt die dort gewonnen Lösungen auf das eigene Gebiet. Bei der verdeckten Variante ist nur den RE das reale Problem bekannt, bei der offenen Variante übertragen die TeilnehmerInnen die Lösung selbst in das reale Fachgebiet.

Um andere Perspektiven einnehmen zu können, hat sich auch die **6 Hüte Methode** bewährt. Sechs TeilnehmerInnen erhalten jeweils eine Rolle, bzw. einen Hut in unterschiedlichen Farben. Eine Farbe verkörpert z.B. das analytische Denken, ein anderer das emotionale Denken, ein dritter das kritische Denken usw. Auch

optimistisches Denken, ordnendes und kreatives Denken sollte vertreten sein. Bei dieser Methode gibt es mehrere Spielarten.

Unterstützende Techniken

Unterstützende Techniken werden eingesetzt um Schwächen der gewählten Methode auszugleichen und zu ergänzen. Pohl und Rupp stellen folgende Methoden vor:

Mindmapping, das Anlegen einer grafischen Darstellung um Abhängigkeiten zwischen den Begriffen darstellen zu können. Diese Methode wird vor allem beim Brainstorming angewandt. Dabei können gut Zusammengehörigkeiten dargestellt werden, man kann Ideen strukturieren und visualisieren. Für längerfristige Dokumentationen eignen sich Mindmaps jedoch nicht, da das Ergebnis nur vom Urheber interpretiert werden kann.

Workshops dienen dazu gemeinsam über evtl. Schnittstellen nachzudenken. Sie fördern die Kompromissbereitschaft, bei negativen gruppendynamischen Prozessen ist jedoch Vorsicht geboten.

Bei der so genannten **CRC-Karten-Technik** (CRC – CLASS Responsibility Collaboration) werden gewünschte Systemeigenschaften notiert und in einem weiteren Schritt zu Anforderungen umformuliert.

Audio- und Videoaufzeichnungen unterstützen vor allem die Feldbeobachtung, da die Befragungen aufgrund des Wegfallen des Protokollierens rascher ablaufen kann, können jedoch nur eingesetzt werden, wenn alle Beteiligten einverstanden sind und die Ergebnisse dadurch nicht verfälscht werden. Die Nachbereitung kostet allerdings viel Zeit.

Bei **Use-Case Modellierungen** wird zunächst die Außensicht des zu erstellenden Systems dokumentiert. Es wird ein rein fachlicher Blick auf das System geworfen, sämtliche Funktionalitäten werden logisch zergliedert. Wichtig ist jedoch, dass vorab klare Regeln erstellt werden um die Ergebnisse nachher vergleichen zu können.

Prototypen dienen einerseits dazu, bereits erarbeitete Anforderungen zu hinterfragen, bzw. evtl. auch neue Anforderungen zu finden (siehe auch Rupp/Sophisten 2014, 113ff; Pohl/Rupp 2015, 33)

Ergänzend kann hier auch noch das SOPHIST-*Re*gelwerk genannt werden, das sich des Neurolinguistischen Programmierens (NLP) bedient. Die Wirkung von Sprache kann hiermit optimiert werden, d.h. es ist auch für die Beschreibung von Anforderungen einsetzbar. (siehe Rupp/Sophisten 2014, 113) Richard Bandler und John Grinder entwickelten NLP ursprünglich um menschliche Kommunikation und damit unter

anderem auch die Psychotherapie zu verbessern. NLP ist aufgrund der teilweise aggressiven Herangehensweise nicht unumstritten. Bandler und Grinder gehen von folgender Prämisse aus:

„Es gibt keine zwei Menschen, die genau das gleiche erleben. Das Modell, das wir entwickeln, um uns in der Welt zu orientieren, basiert teilweise auf unseren Erlebnissen. Demzufolge kann jeder von uns ein anderes Modell der einen Welt, in der wir alle leben, entwickeln, was wiederum bedeutet, daß jeder von uns in einer etwas anderen Wirklichkeit lebt.“ (Bandler/Grinder 2012, 23)

Diese Aussage deckt sich mit den Grundsätzen von RE, da auch hier die unterschiedlichen Erfahrungswelten miteinander in Einklang gebracht werden müssen um erfolgreich Anforderungen zu generieren und zu managen.
Bei der sogenannten Essenzbildung lässt man sämtliche lösungsorientierte Aussagen beiseite und konzentriert sich auf konkrete Abläufe und Abhängigkeiten, damit wird die Komplexität reduziert, man kann sich mit dem eigentlichen Problem befassen. Der Nachteil ist, dass viele Stakeholder bereits eine Lösung im Kopf haben und diese auch kommunizieren möchten. (siehe auch Rupp/Sophisten 2014, 118)
Wenn der RE genügend Facherfahrung hat, kann er auch sein eigenes Wissen nutzen um Anforderungen zu generieren, er kann sozusagen Anforderungen erahnen. Der große Vorteil dabei ist, dass die Stakeholder zunächst nur kurz eingebunden sind. Es muss jedoch darauf geachtet werden, dass ihre Wünsche nicht unberücksichtigt bleiben und die Überprüfung der Anforderungen ist umso zeitaufwändiger. (siehe auch Rupp/Sophisten 2014, 119)

Welche Ermittlungstechnik für welche Fälle eingesetzt werden kann wurde in Form einer Matrix von Rupp und den Sophisten im Internet zur Verfügung gestellt und kann dort auch für den eigenen Zweck angepasst werden.[14]
Je nach Art des zu entwickelnden Systems, nach Vorwissen der Stakeholder und nach Struktur und Größe der Institutionen sind unterschiedliche Herangehensweisen an die Anforderungen in Bibliotheken zielführend.

[14] https://www.sophist.de/fileadmin/SOPHIST/Puplikationen/re6/Kapitel_6/Ermittlungstechniken_-_bearbeitbare_Matrix.pdf (abgerufen am 19.2.2017)

Beim Ermitteln von nicht-funktionalen Anforderungen empfiehlt es sich nach folgenden Schritten vorzugehen:

1. Nicht-funktionale Kategorien auswählen
2. Quellen suchen
3. Ermitteln
4. Dokumentieren (Rupp/Sophisten 2014, 271)

Dabei werden meist spezielle Stakeholder benötigt. Folgende Ermittlungstechniken bieten sich an: Befragung, der prototypische Ansatz, bzw. auch die Systemarchäologie, bzw. die Durchsicht bereits vorhandener Dokumente und die Wiederverwendung von Anforderungen.
Wichtig ist auf die Abhängigkeiten der Kategorien untereinander zu achten, bzw. auch auf eventuelle Widersprüche. Diese können zum Beispiel in folgenden Fällen auftreten:

- Zeitverhalten gegenüber Speicherverbrauch
- Archivierung gegenüber Speicherkapazität
- Sicherheit gegenüber Performanz
- Performanz gegenüber Wiederverwendung
- Einfachheit gegenüber Erweiterbarkeit
- Kosten gegenüber Qualität
- Termin gegenüber Qualität (Rupp/Sophisten 2014, 273)

Widersprüche beinhalten ein hohes Risiko für Projekte und sollten möglichst rasch mit den Beteiligten besprochen und aufgelöst werden.
Anhand der bereits vorhandenen Use-Cases für die Systemfunktionalitäten könnten auch die nicht-funktionalen Anforderungen und sämtliche Verfeinerungen geprüft werden. Beim agilen Vorgehen empfiehlt es sich zunächst eher grob zu beginnen um flexibel zu bleiben.

2.6 Attributierung von Anforderungen

Während des gesamten Lebenszyklus eines Systems müssen Informationen über Anforderungen festgehalten werden. Neben der permanenten ID ist auch ein eindeutiger Name für die Anforderung, Autor und Quelle sowie eine für die Anforderung verantwortliche Person unerlässlich.
Des Weiteren empfiehlt sich eine Beschreibung des Attributs. Die Bedeutung der Attribute muss klar sein, außerdem sollte festgelegt werden, welche Schritte zu unternehmen sind, wenn ein Attribut geändert wird. Muss etwa die Anforderung neu überprüft werden oder wer muss benachrichtigt werden?
Um die Übersicht nicht zu verlieren empfiehlt es sich, die Inhalte klar von den Attributen einer Anforderung zu trennen. Es existieren unterschiedliche Attributtypen, wie etwa **Textattribute** die es erlauben, jede Form von Text einzugeben und beispielsweise Kommentare oder getroffene Entscheidungen dokumentieren können. **Aufzählungsattribute** geben Auskunft über Eigenschaften einer Anforderung. Bei der Definition muss angegeben werden, welche Werte dieses Attribut annehmen darf, wer neue Werte anlegen darf und wer berechtigt ist zu verändern. Auch die möglichen **Abhängigkeiten der Attribute** müssen berücksichtigt werden. Welche Attribute jeweils wichtig sind, sollten schon vor dem Schreiben der Anforderungen festgehalten werden.
Ein Beispiel für ein Attribut ist der **Status.** Hier wird festgehalten, ob die Anforderung bereits ausgearbeitet oder getestet ist, ob sie von den Stakeholdern akzeptiert wurde oder sogar schon freigegeben ist. Auch **Prioritäten** zählen zu den Attributen, sie legen fest, welche Anforderung für das nächste Release für EntwicklerInnen, AnwenderInnen oder KundInnen welche Priorität hat.
Ebenso ist die Aussage über die **Stabilität** einer Anwendung ein Attribut, sie kann hoch, mittel oder gering sein. Die jeweils für die Anforderung relevanten **AnsprechpartnerInnen** sollen ebenfalls als Attribute behandelt werden. Unter Attribute fällt auch die Einschätzung eines Risikos, zum Beispiel das **Akzeptanz-Risiko** oder das **Fehler-Risiko**, bzw. auch die Einschätzung des erforderlichen **Aufwands.**

Für jedes Attribut muss schon zu Beginn der Name des Attributs feststehen, wer die Werte des Attributs verwaltet, wann sie im Projekt gesetzt werden müssen, ob Attribute verwendet werden können oder müssen.

Die Definition der Anforderungsattribute lässt sich gut anhand einer Tabelle, bzw. mit einer Schablone festhalten. In der Schablone sind die zu dokumentierenden relevanten Informationen definiert. Diese definierten Attribute, Attributtypen können sich je nach Anforderung unterscheiden, beispielsweise bei funktionalen Anforderungen sehen die Attribute anders aus als bei der Schablone für Qualitätsanforderungen.

Nimmt man sämtliche definierte Attribute einer Klasse von Anforderungen zusammen, z.B. all jene für funktionale Anforderungen, spricht man von einem Attributierungsschema. Diese Schemata werden je nach Projekt angepasst. Während des Projekts werden die Attribute mit Attributwerten belegt, z.B. Identifikator, Name, Kurzbeschreibung, die Stabilität, die jeweiligen Verantwortlichen, die AutorInnen und die Quellen.

LeserInnen der Anforderungen, wie beispielsweise AuftraggeberInnen, ProjektmanagerInnen oder EntwicklerInnen können rasch die wichtigsten Elemente erfassen und miteinander vergleichen.

In der Literatur und in den Standards des RE werden häufig bereits vordefinierte Attribute angeboten:

- Identifikator: Kurze, eindeutige Identifikation eines Anforderungsartefakts in der Menge der betrachteten Anforderungen
- Name: Eindeutig, charakterisierender Name
- Beschreibung: Beschreibt in komprimierter Form den Inhalt der Anforderungen
- Version: Aktueller Versionsstand der Anforderung
- Autor: Benennt den/die Autor(in) der Anforderung
- Quelle: Benennt die Quelle, bzw. Quellen der Anforderung
- Begründung: Beschreibt, weshalb diese Anforderung für das geplante System von Bedeutung ist.
- Stabilität: Benennt die voraussichtliche Stabilität der Anforderung. Stabilität ist dabei der Umfang, in dem künftig noch Veränderungen bzgl. dieser Anforderung erwartet werden. Mögliche Unterscheidung: „fest“, „gefestigt“, „volatil“.
- Risiko: Benennt das Risiko im Sinne einer Abschätzung der Schadenshöhe und Eintrittswahrscheinlichkeit, z.B. im Hinblick auf die Zeitplanung
- Kritikalität: Im Sinne einer Abschätzung der Schadenshöhe und Eintrittswahrscheinlichkeit

- Priorität: Benennt die Priorität der Anforderung hinsichtlich der gewählten Merkmale zur Priorisierung, z.B. „Bedeutung für die Akzeptanz am Markt“, „Reihenfolge der Umsetzung“, „Schaden bzw. Opportunitätskosten durch Nichtrealisierung“ (Pohl/Rupp 2015, 121)

Zu dieser Liste an Attributtypen können noch weitere hinzugefügt werden, die unter anderem auch die rechtliche Verbindlichkeit umfassen.

Pohl und Rupp schlagen weitere Attributtypen vor:

- Verantwortliche Priorität: Benennt die Person, Stakeholdergruppe bzw. Organisation(seinheit), die für diese Anforderung inhaltlich verantwortlich ist.
- Anforderungstyp: Benennt abhängig vom eingesetzten Differenzierungsschema den Typ der Anforderung (z.B. „funktionale Anforderung“, „Qualitätsanforderung“ oder „Randbedingung“).
- Status bzgl. des Inhalts: Benennt den aktuellen Status des Inhalts der Anforderung, z.B. „Idee“, „Konzept“, „detaillierter Inhalt“.
- Status bzgl. der Überprüfung: Benennt den Status der Validierung, z.B. „ungeprüft“, in Prüfung“, „überprüft“, „fehlerhaft“, „in Korrektur“.
- Status bzgl. der Einigung: Benennt den aktuellen Status der Abstimmung, z.B. „nicht abgestimmt, „abgestimmt“, „konfliktär“.
- Aufwand: Prognostizierter/tatsächlicher Umsetzungsaufwand dieser Anforderung.
- Release: Nummer des Releases, in dem die Anforderung umgesetzt werden soll.
- Juristische Verbindlichkeit: Gibt den Grad der juristischen Verbindlichkeit der Anforderung an.
- Querbezüge: Benennt die Beziehung zu anderen Anforderungen. Zum Beispiel wenn bekannt ist, dass die Realisierung dieser Anforderung die vorherige Realisierung einer anderen Anforderung voraussetzt.
- Allgemeine Informationen: In diesem Attribut können beliebige, für relevant erachtete Informationen zu dieser Anforderung dokumentiert werden. Zum Beispiel wenn die Abstimmung dieser Anforderung auf dem nächsten Treffen mit dem Auftraggeber vorgesehen ist. (Pohl/Rupp 2015, 122)

Die vorgeschlagenen Attributtypen sollen nur die Grundlage für ein Attributierungsschema bilden, die für spezielle Projekte herangezogen werden. Dabei spielen sowohl die Projektgröße, das Projektrisiko, bestimmte Vorgaben des Unternehmens, Standards und sämtliche Randbedingungen eine große Rolle. Das Attributierungsschema kann ggf. wiederverwendet werden, wenn es nicht nur für ganz spezielle Projekte angelegt wurde. Wenn kein Attributierungsschema zur Verfügung steht, sollte bereits zu Beginn des Projekts eines erstellt werden, es sollte eines für jede Art von Informationen zur Verfügung stehen, Abhängigkeiten und Kombinationen von

Attributen sollten möglichst rasch erkannt werden. Zu viele Attribute machen die Handhabung schwierig. Festgelegt werden sollte auch, wer berechtigt ist, Attribute zu lesen, zu befüllen, bzw. wer dafür verantwortlich ist. Um die Übersicht zu behalten empfiehlt es sich auch zu überprüfen, ob die Attribute noch benötigt werden.
Statt einer tabellarischen Auflistung der Attributtypen werden in der Praxis auch oft modellbasierte Definitionen eines Attributierungsschemas verwendet. Diese haben den Vorteil, zwischen den Attributtypen auch Beziehungen und Abhängigkeiten untereinander anzeigen zu können. Attribute sollten auch gefiltert werden können um erkennen zu können, ob noch Lücken vorhanden sind.

2.7 Sichten auf Anforderungen

Da die Anzahl der Anforderungen in Projekten immer mehr zunimmt und auch die Abhängigkeiten zwischen diesen Anforderungen immer komplexer werden, empfiehlt es sich, genau hier den Fokus zu legen. Man unterscheidet unter anderem die Sichten für die ArchitektInnen, die ProgrammiererInnen, ProjektmanagerInnen und TesterInnen. Diese rollenspezifische Sichtendefinition hat den Vorteil, dass einerseits mehrere Sichten für eine Rolle definiert werden können und andererseits kann auch eine Sicht für mehrere Rollen verwendet werden.
Eine Sicht beinhaltet dabei einen Teil der Anforderungsinformationen. Dabei können bestimmte Anforderungen selektiert werden und /oder bestimmte Attribute bewusst ausgeblendet werden.
Bei den Sichten auf die Anforderungsbasis kann auch mit verdichtenden Sichten gearbeitet werden, in dem Eigenschaften gruppiert, gemeinsam betrachtet und für den Gesamtaufwand eines Projekts addiert werden.

2.8 Priorisieren von Anforderungen

Die Frage wann was Priorität hat, kann sehr heikel in einem Projekt sein, da unterschiedliche Stakeholder gänzlich verschiedene Meinungen darüber haben können. Beim Priorisieren werden zunächst die Ziele und die Randbedingungen der Priorisierung festgehalten, z.B. wann welche Stakeholdergruppe verfügbar ist oder wann auf welche Ressourcen zugegriffen werden kann.
Vom Ziel der Priorisierung ausgehend, werden die Priorisierungskriterien festgelegt. Dazu zählen:

- Kosten für die Umsetzung
- Risiko
- Schaden bei nicht erfolgter Umsetzung
- Volatilität
- Wichtigkeit
- Zeitdauer für die Umsetzung (Pohl/Rupp 2015, 126)

Die Auswahl der einzubeziehenden relevanten Stakeholder sichert das erforderliche Expertenwissen. Typisch für die auszuwählenden Stakeholder sind VertreterInnen des Entwicklerteams, des Projektmanagement, der KundInnen und der NutzerInnen.
In einem weiteren Schritt werden die zu priorisierenden Anforderungen ausgewählt. Diese sollten aus den gleichen Detaillierungsebenen stammen, da es sonst zu inkonsistenten und fehlerhaften Ergebnissen kommen kann, da Stakeholder Anforderungen auf höherem Abstraktionsniveau höher priorisieren als konkretere Anforderungen. (siehe Pohl/Rupp 2015, 127)
Danach wird eine geeignete Priorisierungstechnik gewählt, bzw. eine Kombination aus unterschiedlichen Techniken. Die einzelnen Techniken, die ein breites Spektrum umfassen, sind unterschiedlich aufwändig. Bei den meisten Projekten werden Ad-hoc-Priorisierungstechniken angewendet. Zu den weiteren Priorisierungstechniken zählen:

- Ranking: dabei wird von den Stakeholdern im Hinblick auf ein bestimmtes Kriterium eine Rangfolge er zu priorisierenden Anforderungen festgelegt
- Top-Ten-Technik: Für eine vorher festgelegte Anzahl werden im Hinblick auf das betrachtete Kriterium die wichtigsten Anforderungen ausgewählt. Für diese wird danach eine Rangfolge festgelegt
- Ein-Kriteriums-Klassifikation: Dabei steht die Wichtigkeit der Realisierung dieser Anforderungen für den Erfolg des Systems im Vordergrund. Jede Anforderung wird einer der drei Prioritätsklassen zugeordnet: Mandatory (unbedingt zu realisieren), optional (müssen nicht zwingend umgesetzt werden) und nice to have (diese gefährden den Erfolg eines Projekts nicht)
- Kano-Klassifikation: Hier werden die Anforderungen nach deren jeweiligen Marktwirkung klassifiziert und priorisiert. Die Anforderungen werden in drei Merkmalsklassen klassifiziert: Basismerkmale (Das System muss die Anforderung aufweisen, um in den Markt eintreten zu können), Leistungsmerkmal (die Kunden fordern dieses Merkmal bewusst, Leistungsmerkmale steigern die Kundenzufriedenheit)
- Begeisterungsmerkmal (den Stakeholdern war dieses Merkmal gar nicht bewusst und haben die Umsetzung gar nicht erwartet. Damit lässt sich die Kundenzufriedenheit überproportional steigern)
- Wiegers’sche Priorisierungsmatrix. Diese ist zwar sehr objektiv, erfordert jedoch einen sehr hohen Aufwand, da verschiedene Berechnungen stattfinden müssen um Nutzen, Nachteile, Kosten und Risiko einschätzen und miteinander in Verbindung bringen zu können. (siehe Pohl/Rupp 2015, 127ff)

2.9 Verfolgbarkeit von Anforderungen

Die Verfolgbarkeit, die so genannte Traceability wird in der Literatur als sehr wichtig beschrieben. (siehe u.a. Rupp/Sophisten 2014, 421 und Pohl/Rupp 2015, 130) Sämtliche Anforderungen sollten über den gesamten Lebenszyklus des Systems nachvollziehbar sein. Wobei für Rupp der Lebenszyklus schon in der Ideenfindung und in der Entwicklungsphase beginnt. Der Prozess kann auch mehrere Neusysteme überdauern. „Traceability ist die Fähigkeit, Verbindungen und Abhängigkeiten zwischen Informationen, welche während der Analyse, Entwicklung, Wartung und Weiterentwicklung eines Systems anfallen, jederzeit nachvollziehen zu können“ (Rupp/Sophisten 2014, 421)

Es gibt mehrere Gründe warum die Verfolgbarkeit von Anforderungen nützlich ist:

- Nachweisbarkeit: Die Verfolgbarkeit von Anforderungen unterstützt den Nachweis, dass eine Anforderung im System umgesetzt, d.h. durch ein Systemmerkmal realisiert wurde.
- Identifikation von Goldrandlösungen im System: Die Verfolgbarkeit von Anforderungen unterstützt die Identifikation von sogenannten Goldrandlösungen in dem entwickelten System und somit die Identifikation von nicht geforderten Eigenschaften. Hierzu wird für jedes Systemmerkmal (funktional oder qualitativ) nachgeprüft, ob dieses Merkmal zur Realisierung einer Anforderung des Systems beiträgt.
- Identifikation von Goldrandlösungen in den Anforderungen: Über die Verfolgbarkeit einer Anforderung und deren Ursprung können Anforderungen erkannt werden, die beispielsweise zu keinem Systemziel beitragen oder keiner Quelle zuordenbar sind. In der Regel gibt es für die Existenz dieser Anforderungen keinen Grund, weshalb diese Anforderungen auch nicht umgesetzt werden müssen.
- Auswirkungsanalyse: Die Verfolgbarkeit von Anforderungen unterstützt die Auswirkungsanalyse im Änderungsmanagement, in dem beispielsweise bei der Änderung einer Anforderung durch die Verfolgbarkeit die Entwicklungsartefakte identifiziert werden können, die diese Anforderung realisieren und somit möglicherweise von der Änderung der Anforderung betroffen sind.
- Wiederverwendung: Die Verfolgbarkeit von Anforderungen unterstützt die Wiederverwendung eines Vergleichs von Anforderungen eines vergangenen und des neuen Entwicklungsprojekts können über Verfolgbarkeitsbeziehungen ggf. Entwicklungsartefakte (z.B. Komponenten, Testfälle) identifiziert werden, die im neuen Entwicklungsprojekt (adaptiert) wiederverwendet werden können.
- Zurechenbarkeit: Die Verfolgbarkeit von Anforderungen unterstützt beispielsweise die nachträgliche Zuordnung von Entwicklungsaufwänden zu

einer Anforderung. Nach erfolgter Realisierung können z.B. alle Teilaufwände für die Entwicklungsartefakte aufaddiert werden, die der Anforderung zugeordnet sind.

- Wartung und Pflege: Die Verfolgbarkeit von Anforderungen unterstützt die Wartung eines Systems. Beispielsweise können Ursachen und Auswirkungen von Fehlern identifiziert, die von einem Fehler betroffenen Systemteile bestimmt und der Aufwand zur Beseitigung eines Fehlers prognostiziert werden. (Pohl/Rupp 2015, 131ff)

Es werden laut Rupp unterschiedliche Arten von Verbindungen zwischen Anforderungen unterschieden:

Die **Eltern Kind-Verbindung** besteht zwischen Anforderungen unterschiedlicher Detailebenen und bildet die Verfeinerungen ab. Zu einer groben Anforderung werden mehrere detaillierte Anforderungen erstellt. Die Abhängigkeiten untereinander müssen stets nachvollziehbar sein, denn bei Änderungen, beim Löschen oder Ergänzen müssen auch die „Kinder" geprüft werden. Eine konsistente Attributierung ist dabei sehr wichtig, sämtliche Attributwerte in den Eltern-Anforderungen müssen auch in den Kind-Anforderungen vorhanden sein und umgekehrt. Dieses Vorgehen muss auch im RE-Leitfaden festgehalten werden.

Verbindungen von **Anforderungen auf gleicher Ebene** können unterschiedliche Ausprägungen haben, es kann sich um eine fachliche Abhängigkeit handeln, eine funktionale Abhängigkeit, um Ausnahmen, Randbedingungen oder Realisierungsvorgaben. Dazu zählt auch die Verbindung einer Anforderung zur Vor- und Nachfolgerversion.

Es kann sich aber auch um **Verbindungen zwischen verschiedenen Informationsarten** handeln, wie Testfälle, Dokumente, Protokolle usw. Hier wäre zum Beispiel zu fragen, welches Ziel nicht erfüllt wird, wenn eine Anforderung nicht realisiert werden kann, bzw. welche Designänderungen oder angepasste Testfälle notwendig sind, wenn eine Anforderung verändert wird. (siehe Rupp/Sophisten 2014, 425f)

Die Beziehung zwischen Anforderungen und ihren Quellen, bzw. ihren Ursprung nennt man Pre-RS-Traceability. (Pre-Requirements – Specification-Traceability). Die Post-RS-Traceability umfasst die Verfolgbarkeitsinformationen von Anforderungen zu bestimmten Komponenten oder zu Testfällen, also zu allem, was den Anforderungen

nachgelagert ist. Die dritte Form ist die Abhängigkeit zwischen den Anforderungen. So kann eine Anforderung beispielsweise eine andere verfeinern oder sie ersetzen. Während die Anforderungen spezifiziert werden, muss zum Beispiel überprüft werden, ob alle wichtigen Wünsche und Bedürfnisse berücksichtigt wurden, bzw. wird nachvollziehbar, welche Anforderungen aus welchen Wünschen und Bedürfnissen abgeleitet wurden. Während der Verwaltung der Anforderungen kann man damit die Abhängigkeiten von Anforderungen bestimmen, was vor allem bei Änderungen sehr wichtig ist. Traceability hilft auch unnötige Eigenschaften im System zu filtern, die Wiederverwendbarkeit zu erhöhen und die Pflege und Wartung eines Systems zu unterstützen.

Dargestellt wird Traceability mittels Hyperlinks in Texten, Matrixen und Verfolgbarkeitsgraphen. Bei letzterer Methode werden unterschiedliche Knotentypen für Anforderungen und weitere Informationen definiert und diese mittels drei unterschiedlichen Kanten in Beziehung gesetzt. Diese wiederum drücken „realisiert durch“, „ist Ursprung“ oder „verfeinert“ aus. Um erkennen zu können, ob eine Anforderung die Verfeinerung einer anderen ist, wendet man am besten die Queins’sche Um-zu-Regel an. Wenn beide Anforderungen in einem Satz, in dem „um zu“ vorkommt, stehen können, handelt es sich um eine Verfeinerung. Bei der Präzisierung einer Anforderung wird dagegen eine Anforderung mit einer anderen, die mehr Informationswert besitzt, ersetzt. Damit wird eine Qualitätssteigerung erreicht. (siehe Rupp/Sophisten 2014, 38) Traces sollten immer in beide Richtungen angelegt werden um Inkonsistenzen zu vermeiden.

Spezifikationen sollten nicht in allen Bereichen gleich detailliert sein, das würde auf Ressourcenverschwendung hindeuten.

Bringt man sämtliche Verfolgbarkeitstypen von Anforderungen zusammen, kann man für die Anforderungen Verfolgbarkeitsketten erstellen. Diese bilden dann die Grundlage für die Erstellung einer Auswirkungsanalyse im Änderungsmanagement.

Während eines Projekts können jedoch nie sämtliche Informationen erfasst werden. Zielführend ist es, sich auf jene Informationen zu beschränken, für die eine klare Verwendung existiert.

In einem Verfolgbarkeitsmodell wird definiert, welche Traces benötigt werden, wer zu welchem Zeitpunkt und in welcher Weise dokumentiert. Dafür ist es notwendig das Ziel zu definieren, das mit der Traceability erreicht werden soll, welche Informationsarten miteinander verknüpft werden müssen und welche Bedeutung die Traces haben. Auch wie viele Traces vorhanden sein sollten, sollte festgelegt werden, ebenso die Art der Dokumentation.
Da die Pflege der Traces viel Zeit kostet, sollte man Traces immer sofort erstellen, sich auf das Wesentliche konzentrieren, wenn Eltern-Anforderungen verlinkt sind, müssen auch die Kind-Anforderungen mitbedacht werden. Auch die Zuständigkeiten müssen festgelegt werden.

2.10 Strukturierung von Anforderungen

Vor allem nichtfunktionale Anforderungen müssen gut strukturiert werden um nicht die Übersicht zu verlieren. Auch für eine spätere Wiedernutzung ist dies wichtig. Es gibt bereits fertige Gliederungen zum Download, die natürlich jeweils angepasst werden müssen.[15]
Bei funktionalen Anforderungen gestaltet sich die Strukturierung wesentlich komplizierter. Rupp schlägt dafür User-Storys und Diagramme aus dem Analysemodell vor. Die User-Storys dienen dabei der groben Strukturierung, die mit Hilfe von UML-Diagrammen verfeinert wird. Liegen zahlreiche User-Storys vor, wird empfohlen, diese thematisch zu so genannten „Themes“ zusammenzufassen. Für jedes Theme werden unter dem Kapitel „Funktionsspezifische Anforderungen“ weitere Unterkapitel angelegt. Der Kapitelname sollte jeweils dem Namen der Themes entsprechen. (siehe auch Kap. 1.4) Unterhalb der Themes werden dann weitere Unterkapitel angelegt. Darin werden die User-Storys weiter beschrieben – mit Akzeptanzkriterien, bzw. mit modellbasierten oder natürlichsprachlichen Feinanforderungen. In den Feinanforderungen sollten funktionale als auch nicht-funktionale Anforderungen enthalten sein. Hier kann auch eine weitere Untergliederung erfolgen und

[15] siehe www.sophist.de/re6/kapitel19 (abgerufen am 19.2.2017)

„Allgemeingültige Akzeptanzkriterien, bzw. „Allgemeingültige Anforderungen“ angelegt werden. Diese gelten dann für alle Anforderungen der User-Storys der Themes. In einem weiteren Schritt könnten auch die Kapitel „Wiederverwendete Akzeptanzkriterien“ bzw. „wiederverwendete Anforderungen“ angelegt werden. (siehe Rupp/Sophisten 2014, 433ff)

Alternativ zu den User-Storys kann man auch mit Use-Case-Diagrammen funktionale Anforderungen gliedern. Dabei wird ein Use-Case-Diagramm des gesamten Systems unterhalb des Kapitels „Funktionsspezifische Anforderungen“ abgelegt. Für jeden Use-Case wird ein Kapitel mit dem Namen des Use-Cases unterhalb der funktionsspezifischen Anforderungen abgelegt. Innerhalb des Use-Case-Diagramms bilden alle Use-Cases ein eigenes Kapitel, das aus dem Namen des jeweiligen Use-Cases besteht. Verfeinerte Use-Case-Diagramme werden als erstes Element in das Kapitel des verfeinerten Use-Cases eingefügt. Für jeden verfeinernden Use-Case wird ein Kapitel mit dem Namen des Use-Case-Diagramms eingefügt. Darunter werden die Kapitel Use-Case-Beschreibungen und Feinanforderungen angelegt. Vor den Use-Case-Kapiteln kann ein Kapitel „Allgemeingültige Anforderungen“ angelegt werden. Diese müssen dann bei jedem Use-Case beachtet werden. Unterhalb der Use-Case-Kapitel können auch noch „wiederverwendete Anforderungen“ angelegt werden. Bei den allgemeingültigen Anforderungen können auch wieder nicht-funktionale Anforderungen aufgenommen werden. (siehe Rupp/Sophisten 2014, 436ff)

Aktivitätsdiagramme können Anforderungen weiter strukturieren. In diesem Fall kann man unter dem Kapitel Feinanforderungen den verfeinerten Use-Case ablegen. Die Kapitelabfolge sollte dem logischen Fluss des Ablaufs folgen. Auch hier kann wieder ein Kapitel für „allgemeingültige Anforderungen“, bzw. „wiederverwendete Anforderungen“ angelegt werden. Wenn nicht-funktionale Anforderungen einem Use-Case, einer Aktion, oder einem Zustand zuordenbar sind, können auch nichtfunktionale Anforderungen beschrieben werden.

Redundanz sollte auf jeden Fall vermieden werden, viel besser ist es, Verweise anzubringen. Um rasch Informationen mit anderen tauschen zu können, ist es wichtig, mit einem Tool zu arbeiten, das Schnittstellen zulässt.

2.11 Spezifizieren von Anforderungen Requirements Specification

Wie oben beschrieben, nimmt das Ausarbeiten und Formulieren von Anforderungen relativ viel Zeit in Anspruch, die unterschiedlichen Bedürfnisse und Wünsche müssen in eindeutige Anforderungen umgewandelt werden. Günstig dafür ist, wenn bestimmte Lösungsmuster schon vorhanden sind und mit den einzelnen Wünschen in Einklang gebracht werden können.
Das Ziel zu definieren ist ein wichtiger Schritt auf dem Weg zur Anforderung. Dabei werden über- und untergeordnete Ziele unterschieden. Bei der sogenannten UND-Dekomposition müssen alle Teilziele erreicht sein um das Gesamtziel zu erreichen. Bei der Oder-Dekomposition muss mindesten ein Teilziel erreicht sein um das Gesamtziel zu erreichen.

2.11.1 Aufbau von Spezifikationen

Eine Anforderungsspezifikation ist ein Dokument, das sämtliche Anforderungen enthält, die ein System unbedingt erfüllen muss. Dieses Dokument ist Teil eines Vertrages zwischen Auftraggeber und Auftragnehmer. Partsch definiert eine Anforderungsdefinition oder Anforderungsspezifikation als „präzise Beschreibung des Funktions- und Leistungsumfangs an ein System zur Lösung eines Problems unter Berücksichtigung aller wichtigen Umgebungsbedingungen des Einsatzgebietes und der Softwareherstellung“ (Partsch 1991, 29f). Anders ausgedrückt: „Die Systemspezifikation beschreibt alle funktionalen und nicht-funktionalen Anforderungen an ein Systemelement (System, Unterstützungssystem oder Segment).“ (V-Modell XT, 2015, 144)
In agilen Projekten werden keine umfangreichen Spezifikationen verwendet, sondern eher Product Backlogs und User-Storys. Ein Product Backlog ist eine priorisierte Liste von Aufgaben für das Entwicklerteam, die von der Roadmap und ihren Anforderungen abgeleitet wird. Oder anders ausgedrückt: „Ein ordentliches Produkt-Backlog ist wie ein gesunder Mensch: gut gepflegt, organisiert und offen für Änderungen“.[16] Gut

[16] https://de.atlassian.com/agile/backlogs (abgerufen am 19.2.2017)

aufgebaute Storys erfüllen idealerweise die Kriterien von Bill Wakes Akronym „INVEST“ [17]: Das sogenannte INVEST-Prinzip sagt aus, dass User Storys independent, negotiable, valuable, estimable, small und testable sind, also unabhängig, verhandelbar, von Wert, schätzbar, klein und testbar (siehe Rupp/Sophisten 2014, 256):

- Independent (unabhängig): Wir wollen uns in jeder Phase entwickeln können.
- Negotiable (verhandelbar): Das Vermeiden zu vieler Details; bewahren Sie Flexibilität, damit das Team je nach Situation entscheiden kann, welche Bestandteile der Story umgesetzt werden.
- Valuable (wertschöpfend): Benutzer oder Kunden beziehen aus der Story einen Wert.
- Estimatable (abschätzbar): Das Team muss in der Lage sein, die Storys für die Planung zu nutzen.
- Small (klein): Große Storys sind schwerer abzuschätzen und zu planen. Für die Iterationsplanung sollte die Story eine geeignete Größe haben, sodass sie innerhalb der Iteration konzipiert, programmiert und getestet werden kann.
- Testable (überprüfbar): Durch das Dokumentieren von Akzeptanzkriterien oder das Beschreiben der Story als „abgeschlossen“, können Testfälle abgleitet werden.

Anforderungsdokumente enthalten sämtliche beschriebene Anforderungen sowie zahlreiche Informationen, deshalb ist es sehr wichtig, dass die Dokumente gut strukturiert sind. Darauf bauen sämtliche weitere Schritte auf. Der Ist-Zustand, wird gegen den Soll-Zustand (Spezifikation) getestet.

Anforderungen dienen als Grundlage für:

- Kommunikation
- Ermittlung von Rationalisierungspotentialen
- Optimierung des Kundennutzens
- Ausschreibung und Vertragsgestaltung
- Systemarchitektur
- Test und Abnahme
- Systemintegration und Wartung
- Fehlerbehebung und Weiterentwicklung
- Erhöhung der Mitarbeiterzufriedenheit (Rupp/Sophisten 2014, 16)

Schwachstellen eines Anforderungsdokuments können sein:

- Fehlen wichtiger Informationen
- Unzureichende Darstellung der Information
- Mehrdeutig beschriebene Information
- Falsche oder undurchführbare Anforderungsanweisungen

[17] Siehe https://www.agilealliance.org/glossary/invest/ (abgerufen am 19.2.2017)

- Nicht prüfbare Anforderungsanweisungen
- Überholte oder irrelevante Information (siehe Partsch 1991, 53)

In den Anforderungen können unterschiedliche Fehler auftreten:

- Unvollständigkeit
- Inkonsistenz
- Mehrdeutigkeit
- Redundanz
- Überspezifikation (Entwurf anstelle von Anforderungen)
- Anforderungen liegen außerhalb des relevanten Bereichs (ebd.)

Um diesen Fehlern vorzubeugen empfiehlt Partsch Zeit und Sorgfalt auf die Definition der Anforderung zu verwenden, Anforderungsdefinitionen methodisch zu erstellen, eine präzise Sprache und Standards zu verwenden, die Anforderungen gut zu dokumentieren und sämtliche Anforderungen einer Qualitätskontrolle zu unterziehen. (ebd., 56)
Anforderungen kommt auch eine große Bedeutung bei Systemänderungen und -erweiterungen zu, bzw. bei einer Integration mit anderen Systemen.
Meist enthalten Spezifikationen folgende Punkte:

- Einleitung (wofür dient das Dokument?)
- Geheimhaltung
- Standards, Normen und Vorschriften, die einzuhalten sind
- Der Kontext des Produkts (mittels Kontextdiagramm)
- Eine Kurzbeschreibung des Systems
- Die Anforderungen
  - Funktionale Anforderungen
  - Nichtfunktionale Anforderungen
  - Datenanforderungen
- Evtl. Traceability
- Regelungen für die Abnahme

Bei der Gliederung kann man auf bereits vordefinierte Modelle zurückgreifen oder selbst eine Struktur festlegen. Der Vorteil bei der Verwendung von Standards liegt darin, dass sich neue MitarbeiterInnen leichter einarbeiten könnten, dass die Inhalte leichter erfassbar sind und selektives Lesen einzelner Teile und eine automatisierte Überprüfung möglich ist. Außerdem können Standarddokumentationen leichter wiederverwendet werden. Die Strukturen sind natürlich je Projekt anzupassen.
Für die Dokumentation von Anforderungen stehen mehrere Standards zur Verfügung:

RUP (Rational Unified Process) ist ein kommerzielles Produkt von IBM, das erstmals 1998 vorgestellt wurde[18] wird meist für Softwaresysteme verwendet. Die AuftraggeberInnen leiten die Anforderungen aus dem zuvor erstellten Business-Modell ab.

Der IEEE Standard 830-1998 ist ein vom Institute of Electrical and Electronic Engineers entwickelter Standard zur Spezifikation von Software, der mehrmals überarbeitet wurde und sowohl das Pflichtenheft als auch das Lastenheft umfasst und unten aufgeführte Qualitätskriterien erarbeitet hat.

Der ISO/IEC/IEEE-Standard 29148:2011 gibt eine in fünf Blöcken unterteilte Gliederung für Softwareanforderungen vor und unterscheidet:

- Ein Kapitel mit einführenden Informationen, z.B. die Systemabgrenzung oder den Systemzweck betreffend und eine allgemeine Beschreibung der Software
- Ein Kapitel mit einer Auflistung aller referenzierenden Dokumente
- Ein Kapitel für spezifische Anforderungen, betreffend die Schnittstellen, die Performanz usw.
- Ein Kapitel mit sämtlichen geplanten Verifikationsmaßnahmen sowie
- Anhänge, beispielsweise bereits identifizierte Abhängigkeiten (siehe auch Pohl/Rupp 2015, 40f)

Volere – Requirements Resources bieten kostenpflichtig mehrsprachige Templates, Kurse und Videos an.[19]

Das V-Modell des Bundesministeriums des Inneren (BMI) ist ein Vorgehensmodell für die Durchführung von IT-Projekten, das erstmals 1992 von der Bundeswehr veröffentlicht wurde und sich an alle Projektbeteiligten richtet, gibt ebenfalls Strukturen vor. Es setzt sich aus modular aufeinander aufbauenden Vorgehensbausteinen zusammen, die projektspezifisch angepasst werden können. Im August 2015 wurde die neueste Version veröffentlicht.

Im nächsten Kapitel soll vor allem auf das Lasten- und Pflichtenheft eingegangen werden, denen im Gesamtprozess des RE eine sehr große Rolle zukommt.

[18] http://www.ibm.com/software/rational (abgerufen am 19.2.2017)
[19] http://www.volere.co.uk/ (abgerufen am 19.2.2017)

2.11.2 Das Lasten- und Pflichtenheft

Das Lastenhaft wird von den AuftraggeberInnen erstellt und beschreibt die Gesamtheit aller Forderungen bzgl. Lieferung und Leistungen an den Auftragnehmer. Meist werden hier auch alle Randbedingungen an das System und an den Entwicklungsprozess festgehalten.

„Das Produkt Lastenheft Gesamtprojekt enthält alle an das zu entwickelnde System verbindlich gestellten Anforderungen, die das Gesamtprojekt vollständig und konsistent beschreiben. Es ist Basis für die Aufteilung in Teilprojekte. Alle relevanten Anforderungen an das System werden vom Auftraggeber ermittelt und dokumentiert. Kern des Lastenhefts Gesamtprojekt sind die funktionalen und nicht-funktionalen Anforderungen an das System, sowie eine Skizze des Gesamtsystementwurfs. Der Entwurf berücksichtigt die zukünftige Umgebung und Infrastruktur, in der das System später betrieben wird, und gibt Richtlinien für Technologieentscheidungen. […] Zusätzlich werden die zu unterstützenden Phasen im Lebenszyklus des Systems identifiziert und als logistische Anforderungen aufgenommen. Ebenfalls Teil der Anforderungen ist die Festlegung von Lieferbedingungen und Abnahmekriterien. Die funktionalen und nicht-funktionalen Anforderungen dienen nicht nur als Vorgaben für die Entwicklung, sondern sind zusätzlich Grundlage der Anforderungsverfolgung und des Änderungsmanagements. Die Anforderungen sollten so aufbereitet sein, dass die Verfolgbarkeit (Traceability) sowie ein geeignetes Änderungsmanagement für den gesamten Lebenszyklus eines Systems möglich sind. Für die Erstellung des Lastenhefts Gesamtprojektes sowie für dessen Qualität ist der Auftraggeber alleine verantwortlich. Bei Bedarf kann er Dritte mit der Erstellung beauftragen. Das Lastenheft sollte im Allgemeinen keine technischen Lösungen vorgeben, um Architekten und Entwickler bei der Suche nach optimalen technischen Lösungen nicht einzuschränken. (V-Modell XT 2015, 136)

Das Pflichtenheft baut auf das Lastenheft auf, es enthält die von den AuftragnehmerInnen ausgearbeiteten Realisierungsvorgaben. Das Pflichtenheft ist damit die Konkretisierung der Inhalte, die im Lastenheft angeführt sind.

„Die Gesamtsystemspezifikation (Pflichtenheft) ist das Pendant zu dem Auftraggeberprodukt Anforderungen (Lastenheft) auf Auftragnehmerseite. Sie wird vom Auftragnehmer in Zusammenarbeit mit dem Auftraggeber erstellt und stellt das zentrale Ausgangsdokument der Systemerstellung dar. Wesentliche Inhalte der Gesamtsystemspezifikation sind die funktionalen und nicht-funktionalen Anforderungen an das zu entwickelnde Gesamtsystem. Die Anforderungen werden aus den Anforderungen (Lastenheft) übernommen und geeignet aufbereitet. Eine erste Grobarchitektur des Systems wird entwickelt und in einer Schnittstellenübersicht beschrieben. Das zu entwickelnde System sowie weitere zu entwickelnde Unterstützungssysteme werden identifiziert und den Anforderungen zugeordnet. Zusätzliche Anforderungen an die Logistik werden in Zusammenarbeit mit dem

Logistikverantwortlichen erarbeitet. Abnahmekriterien und Lieferumfang für das fertige Gesamtsystem werden aus den Anforderungen (Lastenheft) übernommen und konkretisiert. Um sicher zu stellen, dass alle Anforderungen berücksichtigt sind, wird eine Anforderungsverfolgung, sowohl hin zu den Anforderungen (Lastenheft) als auch zu System und Unterstützungssystemen, durchgeführt. Zur Erstellung der Gesamtsystemspezifikation (Pflichtenheft) sind Kenntnisse aus unterschiedlichen Disziplinen wie Systementwicklung, Sicherheit, Ergonomie und Logistik notwendig, die üblicherweise nicht von einer Person abgedeckt werden können. Da Anforderungen den Kern der Spezifikation darstellen, fällt dem Anforderungsanalytiker (AN) die verantwortliche Rolle für die Erstellung der Gesamtsystemspezifikation zu. Für die inhaltliche Ausarbeitung benötigt er jedoch intensive Unterstützung durch Experten der verschiedenen Disziplinen. Zu jedem in der Gesamtsystemspezifikation identifizierten System, Unterstützungssystem und Segment werden die entsprechenden Produkte wie Spezifikation und Architektur erstellt. Anforderungen an die Logistik werden in der Spezifikation logistische Unterstützung weiter verfolgt.“ (V-Modell XT 2015, 140)

In den meisten Fällen werden die Standards an die jeweiligen Projektbedingungen angepasst. Wichtig ist, dass die Anforderungen im Dokument klar als solche erkennbar sind und von Hinweisen, Notizen oder Kommentaren getrennt werden.

Die Struktur sollte jedoch auf jeden Fall folgende Punkte umfassen:

a. Einleitung
b. Allgemeine Übersicht
c. Anforderungen
d. Anhang
e. Index

a. Die Einleitung sollte einen raschen Überblick gewähren, den Zweck des Anforderungsdokuments beschreiben und die Zielgruppe angeben. Weiters sollte der Systemumfang angegeben werden, die Stakeholder sollten aufgelistet werden, die verwendeten Begriffe sollten definiert werden, sämtliche Dokumente aufgezählt werden und zum Schluss die Inhalte und der Aufbau des Dokuments erläutert werden.

b. Bei der Allgemeinen Übersicht werden zusätzliche fachliche Informationen angegeben, wie Hinweise zum Systemumfeld, Architekturbeschreibungen, wie etwa Schnittstellen oder Speicherbeschränkungen, Systemfunktionalitäten, Nutzer und Zielgruppen, Randbedingungen, Annahmen, wie Entscheidungen, bestimmte Teile des Systems nicht zu entwickeln oder bestimmte Annahmen über den Systemkontext.

c. Im Anhang können weiterführende Informationen angeführt werden, z.B. Standards, Konventionen oder Hintergrundinformationen.

d. Der Index enthält die Kapitelstruktur und das Indexverzeichnis. Da das Anforderungsdokument sehr dynamisch ist, ist dieser Punkt immer aktuell zu halten. (siehe auch Pohl/Rupp 2015, 41ff)

Um die Qualität des Dokuments zu überprüfen, sollten folgende Fragen gestellt werden:

- Ist das Dokument strukturiert?
- Sind die Anforderungen eindeutig und konsistent dargestellt?
- Ist das Dokument, wenn nötig erweiterbar, bzw. modifizierbar?
- Ist das Dokument vollständig?
- Können die Anforderungen verfolgt werden?

2.12 Dokumentation von Anforderungen

„Software ist nur Software, wenn es Dokumentation für die Menschen und Code für die Maschine gibt, und zwar in vollkommener Übereinstimmung. Sonst ist es entweder Belletristik oder Schrott“ (Rupp/Sophisten 2014/22)

Die Qualität der Dokumente ist deshalb so wichtig, weil sie für mehrere Gelegenheiten herangezogen werden. Zunächst für die Planung, indem konkrete Arbeitspakete definiert und Meilensteine gesetzt werden können. Die Anforderungsdokumente bilden jedoch auch die Grundlage für den Architekturentwurf und die Implementierung, aber auch für Tests, mit denen man später das System überprüfen kann. Die Dokumente sind auch für das Änderungsmanagement wichtig, denn man kann rasch erkennen, welche Teile des Systems von der Änderung betroffen sind. Auch für die Wartung und die Nutzung des Systems sind Anforderungsdokumente unverzichtbar, denn damit lassen sich im Betrieb konkrete Mängel herausfinden, ob es sich um Bedienfehler oder etwa um Implementierungsfehler handelt. Auch für das Vertragsmanagement zwischen Auftraggeber und Auftragnehmer sind Anforderungsdokumente grundlegend.

Grundsätzlich müssen Anforderungen so dokumentiert werden, dass sie für alle Beteiligten, von den Stakeholdern, AuftraggeberInnen, EntwicklerInnen und TesterInnen verständlich sind. Funktionale und nicht-funktionale Anforderungen sollten unterschiedlich dokumentiert werden.

2.12.1 Dokumentation von funktionalen Anforderungen

Dabei stehen mehrere Möglichkeiten zu Verfügung:

Anforderungen in natürlicher Sprache

Zunächst empfiehlt es sich die Anforderungen in natürlicher Sprache zu beschreiben. Das hat den Vorteil, dass alle Stakeholder, die über keine technischen Erfahrungen verfügen, die Anforderungen ebenfalls lesen und verstehen können. Außerdem können mit der natürlichen Sprache nahezu alle Sachverhalte beschrieben werden. Die

natürliche Sprache birgt aber auch Gefahren. Menschen unterschiedlicher Herkunft, mit verschiedenen Sprachkompetenzen und Wissensständen, Prägungen und Erfahrungen interpretieren Anforderungen auf unterschiedliche Arten. Daraus können Missverständnisse erfolgen, die dem Projekt Zeit und Geld kosten. Es treten gewisse sprachliche Effekte ein, wie beispielsweise der „Transformationseffekt“, bei dem die Aussage nicht wie vom Urheber eigentlich geplant beim Empfänger ankommt, sondern eine andere Bedeutung annimmt, also transformiert wird.

Pohl und Rupp haben fünf dieser für das RM relevanten Transformationsprozesse dargestellt:

1. Die Nominalisierung

Damit wird ein oft länger dauernder Prozess zu einem einmaligen Ereignis gemacht, wobei die Beschreibungen des Prozesses dabei verloren gehen. So wird das Substantiv „Eingabe“ nicht weiter hinterfragt oder als Prozess, der auf Einzelschritten aufbaut, gesehen. Verwendet man einen nominalisierten Begriff sollte man darauf achten, keinen Spielraum für Interpretationen zu lassen und den Begriff, bzw. den dahinter liegenden Prozess gut zu erklären. Die Anforderungsdokumente sollten daraufhin überprüft werden, ob wirklich alle Prozesse ausreichend definiert und erklärt und allen Stakeholdern auch verständlich sind. Empfehlenswert ist auch, diesen Begriff in das Glossar zu übernehmen.

2. Substantive ohne Bezugsindex

Dabei sind Substantive gemeint, bei denen der genaue Bezug fehlt. Spricht man in den Anforderungen zum Beispiel von „Daten“ ist genau zu klären, welche Daten hier gemeint sind.

3. Universalquantoren

Universalquantoren sind Angaben über Häufigkeiten, die eine Menge von Objekten zusammenfassen und eine Aussage über diese Menge treffen. Die Gefahr besteht darin, dass diese Aussage nicht für alle Objekte zutrifft. Es kann zum Beispiel vorkommen, dass Stakeholder Elemente in eine Gruppe einordnen, die einen Sonderfall darstellen, damit stimmt die Annahme über die gesamte Gruppe nicht mehr. Bei Anforderungen muss deshalb immer hinterfragt werden, ob das geforderte verhalten wirklich für alle Objekte gelten soll. Besondere Vorsicht ist deshalb geboten, wenn Wörter wie „nie“, „immer“, „alle“, „jeder“ usw. verwendet werden. Hier muss jedes Mal hinterfragt werden, ob wirklich alle oder jeder gemeint ist.

4. unvollständig spezifizierte Bedingungen

Anforderungen, die Bedingungen enthalten, müssen daraufhin geprüft werden, ob auch der Fall beschrieben wird, wenn die Bedingung nicht eintritt. Eine Entscheidungstabelle kann dabei helfen, nicht beschriebene Varianten von Bedingungen zu erkennen. Es handelt sich dabei um eine Matrix in der Bedingungen und Belegungen eingetragen werden können. Komplexe Bedingungen können damit übersichtlich dargestellt werden. Mit Vorsicht sind deshalb Wortkombinationen wie „wenn … dann“, „im Fall von“, „Falls“ usw. zu behandeln und zu überprüfen, ob wirklich alle Aspekte berücksichtigt wurden.

5. unvollständig spezifizierte Prozesswörter (Verben)
Manche Verben benötigen mehr als ein Substantiv um vollständig spezifiziert zu sein. Spricht man beispielsweise von „es werden Daten eingegeben“, ist nicht klar, wer, wie, wo und welche Daten eingeben werden. Klarer wird die Aussage, wenn man sie im Aktiv formuliert, z.B. „angemeldete User können ihr Passwort über die Tastatur eingeben …“. (siehe Pohl/Rupp 2015, 53ff)

Mehrdeutige Beschreibungen sind auf jeden Fall zu vermeiden, unterschiedliche Ebenen und Perspektiven dürfen nicht vermischt werden. Dabei hilft die Einhaltung von einigen Grundsätzen:

- Die Sätze sollten kurz gehalten sein
- Es sollte pro Anforderung nur ein Satz formuliert werden
- Eine aktive Sprache ist vorzuziehen
- Lücken sollten sofort gekennzeichnet werden. Dabei empfiehlt es sich „to be defined“ (tbd) zu verwenden.
- Anforderungen müssen klar verfasst werden, denn sie drücken eine Bedingung aus. Möglichkeitsformen haben hier keinen Platz. „Sollte“, „könnte“, „müsste“ dürfen nicht verwendet werden.
- Mittels Kontextdiagramm werden die Anforderungen beschrieben. (Bsp.: Das System macht X, wenn Y nicht zutrifft)
- Alle Anforderungen werden in der Gegenwartsform beschrieben
- Auch nicht Anforderungen werden explizit dargestellt.
- Anmerkungen, die keine Anforderungen sind, werden deutlich kenntlich gemacht, evtl. durch einen Zusatz oder durch eine farbliche Markierung

Diese Technik wird am Häufigsten eingesetzt, da sie für alle Beteiligten leicht verständlich ist, niemand muss sich neue Techniken aneignen und die Sprache ist an sich schon vielseitig einsetzbar. Der Nachteil bei der natürlichen Sprache sind einige Tücken, die auftauchen, wenn man „nur“ Sprache verwendet. Unterschiedliche Interpretationen von Begriffen, unklare Definitionen von Verben oder Unvollständigkeiten können dazu führen, dass Anforderungen nicht klar genug beschrieben werden.

Anforderungen werden mittels Szenarien dokumentiert

Szenarien können mittels natürlicher Sprache oder als Diagramme notiert werden. Meist erfolgt die Beschreibung in natürlicher Sprache, entweder als Freitext oder als Tabelle, bzw. mittels Formular. Szenarien beschreiben jene Situationen, in denen Menschen mit einem System interagieren, das betrifft jedoch nur etwa 90 % einer Spezifikation. (siehe

Rupp/Sophisten 2014, 187) Abläufe und komplexe Zusammenhänge lassen sich damit jedoch sehr gut und anschaulich darstellen. Sie erleichtern den Diskussionsprozess rund um Anforderungen und können auch später für Handbücher oder Tutorials gut eingesetzt werden. Allerdings ist es oft schwierig zu entscheiden, wann Szenarien eingesetzt werden sollten und wann nicht, bzw. wie abstrakt diese sein sollen.

Anforderungen als System- Use- Case-Diagramme

Diese fügen sich die zuvor erstellen Business-Use-Cases und Ablaufdiagramme ein und beschreiben die Architektur des neuen Systems. Dabei ist wichtig möglichst alle Entscheidungsträger miteinzubeziehen, wie etwa die Projektleitung und die Systemverantwortlichen. Ein System-Use-Case beschreibt eine Interaktion eines Anwenders mit einem System, dies wird mit Texten ergänzt. Um zu einem guten Ergebnis zu gelangen wird empfohlen, eine SWOT-Analyse durchzuführen (siehe Rupp/Sophisten 2014, 190) Eine SWOT-Analyse (engl. Akronym für Strengths (Stärken), Weaknesses (Schwächen), Opportunities (Chancen) und Threats (Bedrohungen), eine Stärken-Schwächen-Chancen-Risiken-Analyse, ist als Positionierungsanalyse, Teil der strategischen Planung. Dabei wird die Positionierungsanalyse der eigenen Aktivitäten gegenüber dem Wettbewerb dargestellt. Fragen, die gestellt werden sollten, sind:

- Welches System unterstützt diesen Schritt im Moment?
- Wie unterstützt dieses System den Schritt?
- Soll dieses System auch in Zukunft für die Unterstützung des Schritts genutzt werden oder nicht?
- Existiert bereits ein anderer System-Use-Case, der diesen Schritt unterstützt?
  Falls ja, wird der Schritt diesem System-Use-Case zugeordnet
  Falls nein, wird ein neuer System-Use-Case aufgenommen (Rupp/Sophisten 2014, 190)

Um diese Fragen beantworten zu können, benötigt man eine Übersicht über die Funktionalitäten des Systems, was am besten mit einem System-Use-Case-Diagramm dokumentiert wird. Dabei wird das gewünschte Systemverhalten grafisch dargestellt. Dies kann auch gemeinsam mit den Stakeholdern mittels Moderationskarten und einer Pinnwand geschehen. In einem weiteren Schritt werden diese Diagramme textlich beschrieben. Allerdings sollte man nicht zu sehr ins Detail gehen, warnt beispielsweise

auch Rupp (siehe Rupp/Sophisten 2014, 190). Use-Case-Diagramme sollten nur für die Darstellung der groben Hauptfunktionen eingesetzt werden. Die Diagramme sollten abgeschlossene Abläufe darstellen und sich nicht überschneiden. System-Use-Cases sind sehr anschaulich und leicht verständlich, es muss jedoch auf Redundanzen und Inkonsistenzen geachtet werden.

Die Use-Case-Beschreibung von Anforderungen

Bei diesen so genannten Anwendungsfallbeschreibungen handelt es sich um Dokumentation in natürlicher Sprache mittels Formularen. Diese Formulare haben den Vorteil, dass damit auch jene Informationen abgefragt werden können, auf die sonst leicht vergessen werden könnte. Die Beschreibungen können sowohl die Use-Case-Diagramme genauer beleuchten, aber auch für sich stehen und weitere Informationen, die im Diagramm nicht dargestellt werden, aufnehmen. Sie können je nach Bedarf mehr oder weniger detailliert sein. Use-Case-Beschreibungen sollten für sämtliche Use-Cases des Systems angefertigt werden. Wenn die Abläufe sehr verzweigt sind, wird empfohlen, zusätzliche Aktivitätsdiagramme zu erstellen. (siehe Rupp/Sophisten 2014, 193) bei jedem Anwendungsfall sollen auch neue technologische Möglichkeiten so gut wie möglich mitbedacht werden um sich bei der Umsetzung nicht zu sehr einzuengen. Zu empfehlen ist deshalb die konkrete technische Lösung auszusparen.
Bei Use-Case-Beschreibungen muss man darauf achten, dass der Text nicht unübersichtlich und damit inkonsequent wird. Außerdem müssen für die Beschreibung der Anforderungen auch noch andere Darstellungstechniken herangezogen werden.

Anforderungen als Aktivitätsdiagramm dokumentieren

Diese eigenen sich vor allem zur Darstellung von komplexen Abläufen. Für jeden zuvor erstellten Use-Case sollte ein Aktivitätsdiagramm erstellt werden, eventuell ist es sinnvoll, die Aktivitäten in Unteraktivitäten zu zerlegen und sämtliche Ausnahmen, Ablaufvarianten und Beschreibungen hinzuzufügen. Aktivitätsdiagramme lassen sich zu mehreren Zeitpunkten in Projekten einbinden und können unterschiedlich detailliert gestaltet sein. Als Notationssprache wird dabei UML eingesetzt. Fertige Aktivitätsdiagramme eignen sich auch gut zur Gliederung von Anforderungen. Wenn

die Aktivitätsdiagramme allerdings sehr detailliert sind, sind sie zugleich auch komplex und oft nicht mehr für alle Stakeholder interpretierbar.

Anforderungen als Sequenzdiagramme dokumentieren

Sequenzdiagramme, als eine Art grafische Tabellen konzipiert, lassen sich überall dort gut einsetzen, wo zeitliche Reihenfolgen, Ablaufbedingungen, Schleifen, Alternativen und Nebenläufigkeiten festgehalten werden sollen. (siehe auch Neumann 1998, 200ff) So kann die Kommunikation zwischen den einzelnen PartnerInnen, zum Beispiel NutzerInnen und Systemen gut präsentiert werden, aber auch die zeitliche Abfolge und die unterschiedlichen Verantwortlichkeiten lassen sich festlegen. Auch Use-Cases sind damit verfeinerbar. Bei der Darstellung bedient man sich unterschiedlicher Symbole. So zeigt ein Pfeil mit einer ausgefüllten Pfeilspitze zum Beispiel synchrone Kommunikation an, der Sender muss auf eine Antwort warten, bevor die nächste Aktivität begonnen werden darf. Nichtausgefüllte Pfeile zeigen asynchrone Nachrichten an, d.h. der Sender muss auf keine Reaktionen warten um weitere Schritte setzen zu dürfen. Das Sequenzdiagramm kann bei Use-Cases also immer dort sinnvoll eingesetzt werden, wo Aufgaben zwischen AkteurInnen und Systemen festgelegt werden sollen. Da die Darstellung ziemlich kompliziert und damit für die einzelnen Stakeholder unverständlich werden kann, empfiehlt es sich zusätzlich auch natürlichsprachliche Beschreibungen anzubieten. Schwierig wird es auch, wenn ein Sequenzdiagramm das gesamte System darstellen soll, denn dann wird es rasch unübersichtlich.

Die Dokumentation mittels Zustandsdiagramm

Damit kann das Verhalten eines Systems, bzw. einzelner Komponenten dargestellt werden, wie etwa der Lauf des Buches durch die Bibliothek. Hat man dies einmal sichtbar gemacht, lassen sich feste Regeln aufstellen und Inkonsistenzen erkennen. Das Zustandsdiagramm hat den Vorteil, dass es sehr detailliert sein kann und dass dabei die einzelnen Use-Cases nicht getrennt voneinander betrachtet werden, sondern im Zusammenspiel. Zustandsdiagramme können aber auch zum Prüfen von Anforderungen eingesetzt werden. Der hohe Abstraktionsgrad und der Schwerpunkt auf Technik

erschweren jedoch oft die Rezeption durch die Stakeholder, außerdem erfordert es profunde Kenntnisse des RE.

Die Dokumentation mittels Klassendiagramm

Das Klassendiagramm der UML ermöglicht es Begriffe und die jeweiligen Beziehungen zueinander zu beschreiben. Die einzelnen Begriffe werden dabei als Klassen dargestellt, so können zum Beispiel alle möglichen Aktivitäten zwischen NutzerInnen und System dargestellt werden. Das Klassendiagramm dient als Grundlage für ein Software-Designdiagramm und kann auch zum Prüfen von Anforderungen herangezogen werden. Zusammenhänge zwischen den Begriffen können visualisiert werden und Missverständnisse und Doppelbedeutungen von Begriffen werden erkennbar, allerdings liefert es selbst keine Begriffsdefinitionen.

Die Beschreibung von Systemregeln

Diese beschreiben die Regeln, nach denen Entscheidungen getroffen werden. Die Notation erfolgt entweder über Entscheidungstabellen oder natürlichsprachlich, was zwar allgemein verständlich ist, in komplexeren Fällen allerdings zu Unübersichtlichkeit führt. Entscheidungstabellen zeigen eindeutig die Kombination aller Bedingungen und der daraus resultierenden fachlichen Zuständen.
Mit Systemregeln lassen sich Abläufe eindeutig darstellen, allerdings bedarf es für die Stakeholder meist weiterer Erklärungen.

Verfeinerungen von Anforderungen

Um ein Diagramm detaillierter darzustellen kann man Use-Case-Diagramme und Aktivitätsdiagramme erstellen, bzw. auch Beschreibungen hinzufügen. Diagramme, die andere Diagramme beinhalten ermöglichen eine Strukturierung, außerdem werden Inkonsistenzen ersichtlich. Es wird jedoch empfohlen, sich an keine festen Regeln zu halten, sondern immer im Fall des Falles zu entscheiden, in manchen Fällen ist es hilfreicher, eine Prosabeschreibung anzugeben. (siehe auch Rupp/Sophisten 2014, 208)
Welche Dokumentationstechniken man verwendet, hängt von mehreren Faktoren ab. Einflussfaktoren auf die Wahl der Dokumentationstechniken sind unter anderem:

- Die Akzeptanz: zu komplizierte oder verwirrende Dokumentationstechniken können dazu führen, dass die Stakeholder die Anforderungen ablehnen
- Notationskenntnis: Hier ist zu prüfen, ob alle Beteiligten genügend Vorwissen besitzen um mit der gewählten Technik umgehen zu können
- Spezifikationslevel: Es sollte genau geprüft werden, für welchen Detailisierungsgrad sich welche Technik am besten eignet.
- Komplexität des Sachverhalts: Je nach unterschiedlicher Komplexität werden andere Notationen benötigt.
- Aufwand/Nutzen Verhältnis: man sollte sich fragen, ob nicht mit einer einfacheren Notation eventuell der gleiche Nutzen erzielt werden könnte
- Konsistenz: Da sich Anforderungen immer wieder ändern können, muss man darauf achten, dass diese Änderungen in allen Dokumentationen eingepflegt werden können
- Eindeutigkeit: Die Frage, die sich dabei stellt, ist, ob sich die gewählte Technik eignet, um Details festzuhalten. (siehe Rupp/Sophisten 2014, 211)

2.12.2 Dokumentation von nicht-funktionalen Anforderungen

Nicht-funktionale Anforderungen lassen sich schwerer mit Schablonen dokumentieren. Bei ihnen sind einerseits die Eigenschaften wichtig, die auch überprüfbar sein müssen, wichtig sind aber auch Umgebung und Kontext, die so beschreiben werden müssen, dass das System im Mittelpunkt steht, etwa wenn ein Gerät nur innerhalb eines gewissen Temperaturspektrums funktioniert, etwa: Das Gerät muss so gestaltet sein, dass es zwischen 5 Grad und 50 Grad betrieben werden kann.“ Ein weiterer wesentlicher Punkt sind Prozesse, beispielsweise klare Aussagen darüber, wie oft ein Gerät gewartet werden muss.

Sobald man nicht-funktionale Anforderungen gefunden hat, sollten diese auch entsprechend dokumentiert werden. Meist geschieht die in natürlicher Sprache, für die präzise Formulierung eignen sich auch hier Anforderungsschablonen. Bei der agilen Methode können diese auch in den Akzeptanzkriterien als Verfeinerungen der User-Storys in den Definitions of Done und in den Technical Storys niedergeschrieben werden. Hilfreich könnte auch das Heranziehen von Extrembeispielen sein um zum Beispiel den Begriff „gängige Versionen“ näher zu beschreiben. Auch der Einsatz von TBC’s (To Be Checked) und TBD (To Be Defined) kann hilfreich sein, da diese Platzhalter eine spätere Ausformulierung und Konkretisierung zulassen.

Die Qualitätsanforderungen werden oft nicht ausreichend dokumentiert und zwischen den Stakeholdern abgestimmt, was wiederum die spätere Akzeptanz des Systems negativ beeinflusst.

Das Sophist-Regelwerk

Das Sophist-Regelwerk basiert auf NLP und soll dazu dienen Verfälschungen und Tilgungen, wie es dort heißt aufzudecken. Zu Grunde liegt hier die Transformationsgrammatik von Noam Chomsky. Überträgt man dieses Konzept auf Anforderungen, wird deutlich, dass die Stakeholder, die Anforderungen formulieren, dazu neigen, in ihrer eigenen Sicht der Welt verhaftet bleiben, jede Person nimmt nur einen Bruchteil der Realität war. Durch die eingeschränkte persönliche Wahrnehmung kommt es zur Wahrnehmungstransformation. Jeder Mensch nimmt die Welt individuell wahr und macht sich ein eigenes Bild der Realität. Es entsteht die „persönliche Wirklichkeit“, der Einzelne nimmt somit nur einen Bruchteil der Realität wahr. Daraus folgt, dass es nicht ausreicht nur eine einzige Person zu befragen. (siehe auch Rupp/Sophisten 2014, 126) Wenn dieses persönliche Wissen in Sprache umgesetzt wird, kommt es zu einer weiteren Transformation, nämlich zur Darstellungstransformation. Das Wissen eines Menschen wird umgewandelt und vereinfacht, wenn er es sprachlich ausdrückt. In beiden Fällen gehen Informationen verloren, die für die Anforderungsanalyse wichtig wären. (siehe auch Rupp/Sophisten 2014, 127)
Um diesem Problem entgegenzuwirken, muss der RE diese Prozesse kennen und in der Folge sich nicht auf einen Stakeholder verlassen, sondern mehrere befragen.
Sprachliche Effekte werden auch von der Linguistik schon lange untersucht. Man geht dabei davon aus, dass der Mensch von seiner Wirklichkeit zunächst eine genaue sprachliche Repräsentation hat, die jedoch beim Schreiben oder Sprechen vereinfacht wird.
Sprachlichen Effekten kann man am besten begegnen indem man einige Regeln befolgt:

- Anforderungen sollten in vollständigen Sätzen niedergeschrieben werden

- Begriffe müssen konsequent verwendet werden, Synonyme und Homonyme sollen vermieden werden
- Ein Glossar erleichtert den Umgang mit definierten Begriffen
- Prozesse sollen durch Vollverben, also Verben, die ohne Verbindung mit anderen Verben das Prädikat bilden können, ausgedrückt werden. Diese Verben sollten ebenfalls im Glossar erklärt werden. Diese Verben dürfen aber auch nicht zu schwammig sein. (siehe Rupp/Sophisten, 131f)

Idealerweise geht man beim Prüfen der Anforderungen durch das Sophisten-Regelwerk in drei Schritten vor:

1. Man identifiziert sprachliche Effekte anhand von Signalwörtern
2. Verlorene oder verfälschte Informationen werden durch gezielte Fragestellungen analysiert
3. Sprachliche Mängel oder Inhaltliche Fehler werden bereinigt, indem die Anforderungen mittels der gegebenen Antworten umformuliert werden. (siehe Rupp/Sophisten, 133)

Dabei wird zunächst an einzelnen Wörtern geprüft, ob hier alle relevanten Informationen vorhanden sind. Wenn es beispielsweise um „archivieren“ geht, muss man erfahren, wer, wo, wie und wann archiviert. In einem zweiten Schritt wird der gesamte Anforderungssatz überprüft, dabei werden auch jene Floskeln gestrichen, die nicht zur Information beitragen. In einem weiteren Schritt wird das so genannte „Big Picture“ also das Gesamtbild dargestellt. Dabei kann es passieren, dass zuvor weitere Anforderungen versteckt waren, bzw. Informationen fehlen oder die Anforderungen noch nicht granular genug sind.

Prozesse können geprüft werden, indem man z.B. passive Formulierungen in aktive umwandelt. Dabei stellt sich oft heraus, dass der notwendige Akteur nicht genannt ist. Auch die geforderte Funktionalität muss identifiziert werden, schwammige Formulierungen müssen hinterfragt werden. Nominalisierungen müssen dann aufgelöst werden, wenn sie zur Verschleierung von Prozessen beitragen. Bei der Verwendung von Funktionsverbgefügen ist ebenfalls Vorsicht geboten. Bei der Formulierung „zur Verfügung stellen“ muss auf jeden Fall hinterfragt werden, was das bedeutet. Für jedes Prozesswort sollte es einen eigenen Satz geben. Besonders auf Unvollständigkeiten sollte geachtet werden.

Beim Gesamtbild muss auf Ausnahmen vom Normalverhalten geachtet werden, vor allem muss geklärt werden, ob es Situationen geben kann, in denen das System das

Normalverhalten nicht gewährleisten kann und worauf dann geachtet werden muss. Ebenso muss die Vollständigkeit bei den Bedingungen geprüft werden. Auch implizite Annahmen müssen in die Überprüfung miteinbezogen werden. Diese müssen irgendwo beschrieben werden, man kann nicht davon ausgehen, dass alle Beteiligten über dieses implizite Wissen verfügen.[20]

Satzschablonen

Satzschablonen wirken den oben beschriebenen sprachlichen Effekten entgegen und dienen der Verständlichkeit. Außerdem erhalten die AutorInnen bei der Formulierung von Anforderungen Unterstützung, diese in möglichst hoher Qualität zu verfassen. „Eine Satzschablone (Requirements Template) ist ein Bauplan für die syntaktische Struktur einer einzelnen Anforderung.“ (Pohl/Rupp 2015, 57) Satzschablonen verhindern, dass sprachliche Effekte erst gar nicht in den Anforderungen auftauchen. (siehe auch Rupp/Sophisten 2014, 217)

Pohl und Rupp schlagen folgende Schritte zum Aufbau einer Satzschablone vor:

1. Schritt 1: Die rechtliche Verbindlichkeit wird aufgebaut. Es wird unterschieden zwischen rechtlich bindenden, dringend empfohlenen, zukünftigen und wünschenswerten Anforderungen. Diese Unterscheidung wird entweder durch Attribute der Anforderung getroffen oder durch Formulierungen wie „Das System kann/soll/ muss … „ Diese Einteilung kann z.B. in Form einer Tabelle geschehen.

2. Schritt 2: Der Kern der Anforderung wird festgelegt. Die gewünschte Funktionalität, bzw. der Prozess, wird hinzugefügt. Diese Prozesse werden ausschließlich durch Vollverben beschrieben. Diese müssen konkret und eindeutig definiert sein. Am besten ist es die Unterlagen nach Verben durchzusehen und zu schauen, ob diese eindeutig sind und erklärt wurden. Danach könnte man eine Tabelle anlegen, die einerseits die Verben enthält, daneben die Bedeutung und eventuell die Synonyme. Auch Homonyme sollten berücksichtigt werden. Wichtig ist, dass diese Liste ständig gepflegt und aktuell gehalten wird.

3. Schritt 3: Die Aktivitäten des Systems werden charakterisiert. Hier wird unterschieden in selbständige Systemaktivitäten, d.h. das System führt den Prozess selbständig durch, es werden Benutzerinteraktionen geschildert es wird dem User die Möglichkeit geboten etwas zu tun, oder es werden Schnittstellen miteinbezogen. Eine Schablone dafür wäre zum Beispiel „Das System

[20] Eine Checkliste für all diese Fragen findet sich unter: https://www.sophist.de/fileadmin/SOPHIST/Puplikationen/re6/Kapitel_7/Das_SOPHIST-Regelwerk_10.10.2014.pdf (abgerufen am 19.2.2017)

muss/sollte/wird/kann <Prozesswort>. <Prozesswort> bezeichnet die in Schritt 2 angegebene Funktion, wie zum Beispiel „drucken“. Will man die NutzerInnen einbinden, erweitert sich die Schablone: „Das System muss/sollte/wird/kann <wem?> die Möglichkeit bieten <Prozesswort>. Falls das System in Abhängigkeit von Nachbarsystemen agieren soll und auf Daten oder Nachrichten mit bestimmten Aktionen reagieren soll, wird folgende Formulierung vorgeschlagen: „Das System muss/sollte/wird/kann fähig sein <Prozesswort>.“

4. Schritt 4: Hier werden, wenn nötig, Objekte eingefügt. Zum Beispiel: Das System muss/sollte/wird/kann fähig sein, dem User eine Sortierung der Objekte zu erlauben.

5. Schritt 5: An dieser Stelle werden logische und zeitliche Bedingungen hinzugefügt. Die Funktionalitäten eines Systems werden meist nicht dauernd ausgeführt sondern nur zweitweise und unter bestimmten Bedingungen. Für zeitliche Bedingungen wird vorgeschlagen das Wort „sobald“ zu verwenden und für logische Bedingungen „falls“. Für zeitliche Bedingungen können auch noch weitere temporale Konjunktionen verwendet werden, wie etwa „nachdem“, „während“ oder „solange“. Bei der Konjunktion „wenn“ ist Vorsicht geboten, da sie sowohl bei zeitlichen als auch bei logischen Bedingungen eingesetzt werden kann. Qualitätsanforderungen, unter denen eine Anforderung erfüllt werden soll, werden mittels Nebensatzes an den Anfang der Anforderung gestellt. (siehe Rupp/Sophisten2014, 2019ff; Pohl/Rupp 2015, 57ff)

Satzschablonen erhöhen die Klarheit der Anforderungen, müssen aber von allen Stakeholdern akzeptiert werden. Das ist oft schwierig, da die stilistische Freiheit stark eingeschränkt ist.

Die Regeln bei Satzschablonen lauten:

- Regel 1: Anforderungen in vollständigen Sätzen beschreiben
- Regel 2: Anforderungen in kurzen Sätzen beschreiben
- Regel 3: Nur eine Anforderung pro Satz formulieren
- Regel 4: Begriffe konsistent verwenden (Kulke 2014, 10)

Anforderungen modellbasiert dokumentieren

Anforderungen können, wie oben beschrieben, mit sämtlichen Vor- und Nachteilen in natürlicher Sprache beschrieben werden, können aber auch in Formen von Modellen dokumentiert werden. Diese werden meist ergänzend zu den natürlichsprachlichen Ausführungen eingesetzt. „Ein Modell ist ein abstraktes Abbild einer existierenden oder einer noch zu schaffenden Realität“ (Pohl/Rupp 2015, 64)

Während des Gesamtprozesses müssen unterschiedliche Informationen dokumentiert werden, u.a. Protokolle von Interviews, Berichte über Abstimmungen und Änderungsanträge. Hauptaufgabe ist jedoch, die Anforderungen selbst langfristig zu archivieren. Die Herausforderung besteht unter anderem auch darin, dass zahlreiche Personen in den Prozess eingebunden sind.
Die Frage, wie man Anforderungen nun am besten dokumentiert, hängt davon ab, welche Perspektive man einnimmt. Zum Beispiel, ob man aus der Strukturperspektive agiert, sich also mit der Frage beschäftigt, wie beispielsweise Abhängigkeitsbeziehungen festgehalten werden können, ob man eher die Funktionen im Blick hat oder eher das Verhalten, wie zum Beispiel das neue System im Kontext mit bereits vorhandenen Systemen agiert.
Die Vorteile von Modellen sind:

- Modelle können die Realität abbilden, sowohl die bereits existierende Realität (deskriptiv) als auch die zu entwickelnde Realität (präskriptiv)
- Modelle verkürzen die Realität auf zwei Arten. Bei der Selektion werden nur bestimmte Aspekte herausgegriffen und im Modell abgebildet. Bei der Verdichtung werden alle Aspekte berücksichtigt, jedoch zusammengefasst.
- Modelle sind pragmatisch, d.h. sie enthalten jene Informationen, die für den jeweiligen Verwendungszweck (siehe Pohl/Rupp 2015, 64)

Die Sprache der Modelle

Meist werden grafische konzeptuelle Modelle eingesetzt. Konzeptuelle Modellierungssprachen werden durch Syntax und Semantik definiert. Die Syntax legt die zu verwendenden Modellelemente fest und definiert die gültigen Kombinationen der Sprachkonstrukte. Die Semantik definiert die Bedeutung der Modellelemente und definiert damit die Basis für die Interpretation von konzeptuellen Modellen der jeweiligen Modellierungssprache. Pohl und Rupp unterscheiden hinsichtlich des Formalisierungsgrades informelle, semiformale und formale Modellierungssprachen. (Pohl/Rupp 2015, 65) Der Formulierungsgrad ist abhängig vom Umfang in dem Syntax und Semantik der Sprache formal definiert sind.
„Konzeptuelle Modelle, die Anforderungen eines Systems dokumentieren, werden als Anforderungsmodelle bezeichnet.“ (Pohl/Rupp 2015, 65). Zur Konstruktion von

Anforderungsmodellen wird häufig die Unified Modeling Language (UML) eingesetzt. Sie besteht aus einer Menge von Modellierungssprachen, die vor allem im Bereich des RE eingesetzt werden, um Anforderungen aus unterschiedlichen Blickwinkeln betrachten zu können. Während konzeptuelle Modelle in der Systementwicklung Lösungsaspekte dokumentieren, dokumentieren Anforderungsmodelle spezifische Aspekte des Problems.
Anforderungsmodelle haben den großen Vorteil, dass sie mit Bildern arbeiten, die, wie man aus der Kognitionsforschung weiß, schneller erfasst und besser im Gedächtnis behalten werden können. Außerdem fokussieren sie auf bestimmte Aspekte und lassen andere Elemente, die für die jeweilige Darstellung derzeit nicht relevant sind, in den Hintergrund treten. Sie beschreiben die Realität aus einer bestimmten Perspektive und müssen nicht sämtliche Details widerspiegeln.
Kombiniert man Anforderungsmodelle mit natürlicher Sprache können die Vorteile beider Dokumentationsformen genutzt werden. Natürlichsprachliche Anforderungen können durch Anforderungsmodelle zusammengefasst werden, Anforderungsmodelle können eingesetzt werden um natürlichsprachliche Anforderungen detaillierter und präziser darzustellen und natürliche Sprache kann eingesetzt werden um zusätzliche Informationen anbieten zu können.

Use-Case-Spezifikationen verwenden unterschiedliche Schablonen zur textuellen Spezifikation von Use-Cases um Typen von Information zu definieren und diese Informationen zu strukturieren.

Pohl und Rupp legen ein Beispiel für eine Schablone eines Use Case vor:

Nr.	Abschnitt	Inhalt/Erläuterung
1	Bezeichner	Eindeutiger Bezeichner des Use Case
2	Name	Eindeutiger Name für den Use Case
3	Autoren	Namen der Autoren, die an dieser Use-Case-Beschreibung mitgearbeitet haben
4	Priorität	Wichtigkeit des Use Case gemäß der verwendeten Priorisierungstechnik
5	Kritikalität	Kritikalität des Use Case, z-B. hinsichtlich des Schadensausmaßes bei Fehlverhalten des Use Case
6	Quelle	Bezeichnung der Quelle ([Stakeholder I Dokument I System]), von der der Use Case stammt
7	Verantwortlicher	Der für diesen Use Case verantwortliche Stakeholder
8	Beschreibung	Komprimierte Beschreibung des Use Case
9	Auslösendes Ereignis	Angabe des Ereignisses, das den Use Case auslöst
10	Akteure	Auflistung der Akteure, die mit dem Use Case in Verbindung stehen
11	Vorbedingung	Eine Liste notwendiger Voraussetzungen, die erfüllt sein müssen, bevor die Ausführung des Use Case beginnen kann
12	Nachbedingung	Eine Liste von Zuständen, in denen sich das System unmittelbar nach der Ausführung des Hauptszenarios befindet.
13	Ergebnis	Beschreibung der Ausgaben, die während der Ausführung des Use Case erzeugt werden
14	Hauptszenario	Beschreibung des Hauptszenarios eines Use Case
15	Alternativszenarien	Beschreibung von Alternativszenarien des Use Case oder lediglich Angabe der auslösenden Ereignisse. Hier gelten oftmals andere Nachbedingungen als in (12) enthalten
16	Ausnahmeszenarien	Beschreibung von Ausnahmeszenarien des Use Case oder lediglich Angabe der auslösenden Ereignisse. Hier gelten oftmals andere Nachbedingungen als in (12) enthalten
17	Qualitäten	Querbezüge zu Qualitätsanforderungen

Quelle: Pohl/Rupp 2015, 72f

2.12.3 Die Rolle der Sprache

Die Art und Weise, wie Sprache eingesetzt wird, hat einen Einfluss auf die Verständlichkeit. Eine aktive Sprache ist einer passiven Sprache auf jeden Fall vorzuziehen. Auch exakte Angaben zu verwenden ist zielführender als einfach zu sagen, dass das System „besser“ funktionieren soll als das alte. Möglichkeitsformen sollten so weit wie möglich vermieden werden, da sie zu unklaren Äußerungen führen.
Vorsicht ist auch bei Generalisierungen geboten. Beim Satz „Niemand kann die Daten wieder löschen“ wäre beispielsweise zu klären, ob das einerseits so stimmt, bzw. auch so stimmen soll. Eventuell ist es nicht sinnvoll, bzw. auch rechtlich nicht möglich, dass niemand die Daten löschen darf.
Sämtliche Möglichkeiten, die in einem System zur Verfügung stehen sollen, müssen auch angemessen beschrieben werden. Es darf keine Möglichkeit übersehen werden.
Ebenso wichtig ist es Begriffe scharf voneinander zu trennen, z.B. zu klären, wer genau mit „Angehörige der Universität“ gemeint ist. Prozesse sollten möglichst genau beschrieben werden. „Der Text muss ausgedruckt werden können“ ist beispielsweise zu wenig.
Eine gute Spezifikationssprache sollte folgende Merkmale aufweisen:

- Den Erstellungsprozess unterstützen und vereinfachen
- Die Analyse der Spezifikation bezüglich interner Konsistenz, Präzision und Vollständigkeit zulassen
- Die Validation erleichtern
- Leicht erlernbar und gut anwendbar sein
- Präzise und eindeutig sein
- Eine ökonomische und kompakte Beschreibung ermöglichen
- Maschinell verarbeitbar
- Eine inkrementelle Erstellung ermöglichen (verschiedene Gesichtspunkte können nach und nach berücksichtigt und beschrieben werden)
- Konzeptionell erweiterbar
- Tolerant gegen zeitweiligen Unvollständigkeiten
- Die Formulierung von Nebenläufigkeiten zulassen
- Verschiedene Gesichtspunkte zulassen (Partsch 1991, 57)

Wenn Anwendungsfälle übersichtlich und einheitlich beschrieben werden sollen, empfiehlt es sich, Diagramme zu verwenden. Diese sind für Personen, die geübt sind, Diagramme zu lesen, einfacher zu interpretieren.
Mittels eines Use-Case-Diagrammes kann man zum Beispiel Funktionalitäten und Abhängigkeiten darstellen. Für detaillierte Beschreibungen ist diese Art der Dokumentation jedoch nicht geeignet. Klassendiagramme verwendet man um strukturelle Abhängigkeiten zwischen System und Systemkontext darzustellen und Aktivitätsdiagramme setzt man ein um Prozesse visualisieren zu können und den Ablauf eines Use-Cases zeigen zu können. Will man die unterschiedlichen Zustände eines Systems darstellen, verwendet man am besten Zustandsdiagramme.
Die Notation dafür ist die UML (Unified Modelling Language). Die UML ist ein anerkannter internationaler Standard für grafische Modellierung, bzw. „eine grafische Notation mit einfacher Syntax und weitgehend formal definierter Semantik“ (Neumann 1998, V). Die Vorteile von UML liegen einerseits in der eindeutigen Syntax, man kann genauer Sachverhalte definieren als mit natürlicher Sprache und muss auch nicht übersetzt werden. Durch UML wird eine Realität abgebildet, wobei man sich auf das Wesentliche konzentriert. Einer der Nachteile ist jedoch, dass viele nicht daran gewöhnt sind und deshalb diese Darstellungsweise nicht akzeptieren.

Folgendes Beispiel zeigt die Möglichkeiten, die UML bietet:

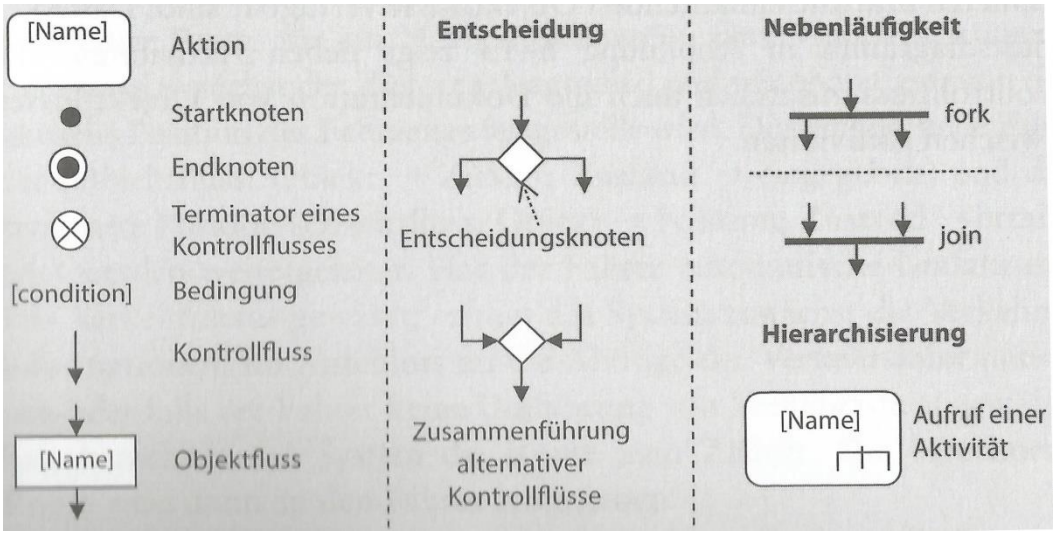


Abb. 1, Quelle: Pohl/Rupp 2014, 85

2.12.4 Perspektiven der Anforderungen

Anforderungen kann man aus unterschiedlichen Perspektiven betrachten:

a. Aus der Strukturperspektive wird auf die Struktur der Daten geachtet und die Nutzungs- und Abhängigkeitsbeziehungen dokumentiert. Einerseits berücksichtigt man dabei das vorgesehene Verhalten des IT-Systems, andererseits auch die Daten, die dabei verarbeitet werden.
b. Die Funktionsperspektive fokussiert darauf, welche Informationen aus dem Systemkontext durch das System manipuliert werden und welche Daten vom System in den Systemkontext fließen, zum Beispiel hinsichtlich der Verarbeitung von Eingabedaten.
c. Bei der Verhaltensperspektive achtet man auf das dynamische Verhalten eines Systems, also auf die unterschiedlichen Zustände, in denen sich ein System befinden kann, bzw. auf die Ereignisse, die diesen Zustandswechsel zugrunde liegen.
d. Aus der Ablauforientierten Perspektive stehen die Abläufe, bzw. das Zusammenwirken im Vordergrund
e. Bei der Ereignisorientierten Perspektive konzentriert man sich auf bestimmte Ereignisse, auf die reagiert werden muss
f. Aus der Perspektive des Datenflusses fokussiert man auf die zu erfassenden, zu verarbeitenden oder zu beeinflussenden Daten
g. Aus der Aufgabenbereich-orientierten oder Modul-orientierten Perspektive wird das Augenmerk auf logisch zusammenhängende Aufgabenbereiche gelegt
h. Bei der Anlagen/Geräte-orientierten Perspektive stehen die benötigten Gerätekomponenten im Fokus. (siehe Partsch 1991, 62f)

Meist werden die Perspektiven gemeinsam verwendet und – getrennt voneinander – auch mittels natürlicher Sprache, bzw. konzeptueller Modellierungssprachen beschrieben. Die Strukturperspektive wird oft mittels UML-Klassendiagramme beschrieben. Anforderungen in der Funktionsperspektive werden mit UML-Aktivitätsdiagrammen gezeigt und Anforderungen der Verhaltensperspektive oft mittels Statecharts dokumentiert.

Die Perspektiven sind nicht disjunkt (getrennt) zu sehen, deshalb können die drei Modelle wechselseitig auf Vollständigkeit und Konsistenz geprüft werden.

Modellierung von Anforderungen in der Strukturperspektive

Neben den bereits erwähnten Klassendiagrammen der UML werden auch Entity-Relationship-Modelle mit unterschiedlichen Erweiterungen herangezogen. Dabei unterscheidet man einzelne Elemente:

- Ein so genannter Entitätstyp, meist mit einem Rechteck dargestellt, definiert eine Menge von Entitäten innerhalb des Gegenstandbereiches. Diese Entitätstypen können zum Beispiel Personen sein, die ein gemeinsames Merkmal besitzen, wie etwas „angemeldete User“ alle einen Account besitzen.
- Ein Beziehungstyp klassifiziert eine Menge gleichartiger Beziehungen zwischen Entitätstypen innerhalb des Gegenstandsbereiches, etwa wenn sich „angemeldete User“ „einloggen“
- Attribute gibt es sowohl für Entitätstypen als auch für Beziehungstypen, sie definieren spezifische Eigenschaften, wie beispielsweise Vor- oder Nachnamen.
- Die Kardinalität definiert die Anzahl von Beziehungsinstanzen an denen eine Entität teilnehmen kann. Sind keine Kordialitäten angegeben, wird vorausgesetzt, dass beliebig viele Entitäten an einer Beziehung beteiligt sein können. (siehe Pohl/Rupp 2016, 77f)

So genannte ER-Modelle (Entity-Relationship-Modelle) bilden detailliert die jeweiligen Anforderung an die Datenbankmodellierung an.
Das ER-Modell geht von der Annahme aus, dass sich jede Situation in der realen Welt mit Hilfe von Entitäten und Beziehungen adäquat darstellen lässt. Entitäten werden auch Objekte genannt, sie sind gut unterscheidbare Dinge, eine Beziehung ist eine Assoziation zwischen Entitäten. Die Rolle einer Entität in einer Beziehung gibt Auskunft über die Funktion, die die betreffende Entität in der Beziehung spielt. Attribute geben eine Eigenschaft einer Entität oder Beziehung an. (siehe Partsch 1991, 95f) Visualisiert wird mittels Entity-Relationship-Diagrammen. Die Daten werden mittels Datenbank verwaltet.
Klassendiagramme erlauben auch Generalisierungsbeziehungen zwischen Klassen. Damit wird eine Beziehung zwischen spezielleren Klassen (Subtypen) und einer generellen Klasse (Supertyp) beschrieben, wobei der Subtyp die Eigenschaften des Supertyps erbt.

Damit werden unterschiedliche Ebenen von Klassen, Subklassen und Superklassen beschrieben, Assoziationen beschreiben die möglichen Relationen zwischen den Klassen.

Die Modellierung von Anforderungen in der Funktionsperspektive

Hier wird die Transformation von Eingabedaten durch das System bis zu den Ausgabedaten beschrieben. Die Modellierungsansätze liegen dabei in den Systemanalyseansätzen, die auf der Systemtheorie basieren. Systemanalysen gehen davon aus, dass man ein Modell des gewünschten Systems als Blackbox konstruiert. Man lässt dabei das eventuell komplexe Innere des Systems außer Acht und konzentriert sich auf In- und Output.
Bei der Anforderungsmodellierung in der Funktionsperspektive werden Funktionsmodelle, die Datenspeicher, Informationsflüsse und Quellen betreffen, beschrieben. Oft werden Datenflussmodelle herangezogen, mit denen man das System in verschiedenen Abstraktionsebenen beschreiben kann.
Datenflussdiagramme vernachlässigen die Bedingungen unter welchen die Prozesse durchgeführt werden. Diese können durch weitere Dokumentationsformen beschrieben werden, evtl. durch Spracherweiterungen, die es erlauben auch den Kontrollfluss zwischen den Funktionen zu beschreiben.
UML-Aktivitätsdiagramme eignen sich gut für die Modellierung von Abläufen, sie berücksichtigen auch den Kontrollfluss. Dabei wird zum Beispiel sichtbar gemacht, dass eine Folgeaktivität nur dann ausgeführt wird, wenn eine vorangegangene Aktivität in einer bestimmten Zeit abgeschlossen ist.

Die Modellierung von Anforderungen in der Verhaltensperspektive

Da hier das dynamische Verhalten eines Systems modelliert werden soll, verwendet man oft die Theorie endlicher Automaten. Diese endlichen Automaten, die auch Zustandsmaschinen genannt werden, sind Modelle eines Verhaltens, das aus Zuständen, Zustandsübergängen und Aktionen besteht. Die Menge an Zuständen, die ein endlicher

Automat einnehmen kann, ist eingeschränkt. Endliche Automaten lassen jedoch nur sehr begrenzt Abstraktion zu. Deshalb wurden die Automatenkonzepte weiterentwickelt und so genannte Statecharts entwickelt, die ebenfalls auf Automatenkonzepten beruhen, aber Möglichkeiten der Hierarchisierung von Zuständen ermöglicht. Auch die Darstellung von Nebenläufigkeiten ist bei Statecharts möglich.

Die wichtigsten Elemente von Statecharts sind:

- Ein Zustand, der einen Zeitraum definiert, in welchem das zu betrachtende System ein bestimmtes Verhalten zeigt und auf das Eintreten eines davor definierten Ereignisses wartet um einen definierten Zustandsübergang vornehmen zu können.
- Ein Zustandsübergang wird durch das Eintreten eines Ereignisses ausgelöst, es beschreibt den Wechsel des Ausgangszustandes zum Folgezustand. Zustandswechsel können dabei an bestimmte Bedingungen geknüpft sein
- Eine Hierarchisierung von Zuständen ermöglicht wiederum irrelevante Details wegzulassen und den übergeordneten Zustand (Superzustand) durch verfeinerte Methoden, etwa durch Automaten zu definieren.
- Ein Zustand kann auch durch mehrere Nebenläufigkeiten dargestellt werden. Ein Zustand kann also in mehrere nebenläufige Automaten zerlegt werden. (siehe Pohl/Rupp, 89f)

Bei der Erstellung von Anforderungen ist auch darauf zu achten, dass zwischen Problemstellung und Lösung strikt getrennt wird, weil Problemstellungen stabiler sind und es für jede Problemstellung möglicherweise mehrere Lösungen gibt.

2.13 Qualitätskriterien von Anforderungsdokumenten

Laut ISO/IEC/IEEE-Standard 29148:2011 sollte ein Anforderungsdokument klar strukturiert, vollständig und konsistent sein, einen angemessenen Umfang besitzen, gut lesbar sein, vollständig und konsistent sein.
Konsistenz ist dann gegeben, wenn die Anforderungen in sich eindeutig sind und einander nicht widersprechen. Wichtig ist auch, dass Anforderungsdokumente eindeutig identifizierbar sind. Eine klare Strukturierung ermöglicht es den LeserInnen sich rasch im Dokument zurechtzufinden und auch mal Teile zu überspringen.
Anforderungsdokumente sollten auch modifizierbar und erweiterbar sein und versioniert werden können. Dafür ist eine gute Versionsverwaltung nötig. Ein wesentliches Kriterium ist die Vollständigkeit der Dokumente, sowohl die Anforderungen selbst als auch die zusätzlichen Informationen müssen vollständig dokumentiert sein. Sämtliche gewünschte Funktionalitäten müssen beschrieben werden, aber auch alle Reaktionen, Ausnahmen und mögliche Fehler. Ebenso wichtig ist es, auch Qualitätsanforderungen zu dokumentieren sowie Hinweise zur Bedienbarkeit des Systems. Sämtliche Grafiken, Tabellen und Diagramme müssen beschriftet werden, es dürfen keine Quellen fehlen und auch eine Übersicht über Definitionen und das Abkürzungsverzeichnis müssen vorhanden sein. Die Beziehung zu sämtlichen weiteren Dokumenten, die mit den Anforderungsdokumenten in Beziehung stehen, muss klar definiert sein. Nur so ist bei Änderungswünschen sichergestellt, dass nichts vergessen wird. Im Projekt ist auch zu bedenken, dass das Erstellen, Ergänzen und die Verwaltung von Anforderungsdokumenten viel Zeit und Ressourcen kostet.
„Im Kontext der Anforderungsanalyse gibt Qualität […] an, in welchem Maß eine Anforderung oder ein Anforderungsdokument den objektiv messbaren und fest definierten Qualitätskriterien entspricht.“ (Rupp/Sophisten 2014, 302)
Grundsätzlich kann man drei Qualitätsaspekte unterscheiden:

- Inhalt: Prüfung auf Vollständigkeit der Anforderungsdokumente, Vollständigkeit der einzelnen Anforderungen, Verfolgbarkeit, Korrektheit, Konsistenz, Adäquatheit, Überprüfbarkeit, Notwendigkeit
- Dokumentation: Prüfung auf Mängel in der Dokumentation, Verständlichkeit, Eindeutigkeit

- Abgestimmtheit: Prüfung auf Mängel in der Anforderungsabstimmung (siehe Rupp/Sophisten 2014, 303)

Dazu muss jedoch erst mal definiert werden, was inhaltlich „gute“ Anforderungen sind.

Das Institute of Electrical and Electronic Engineers hat die Software Requirements Specification (SRS) entwickelt. Demnach sollten Anforderungen, orientiert am ISO/IEC/IEEE-Standard 29148:2011

- korrekt und mit den mit den Stakeholdern abgestimmt
- notwendig
- realisierbar (innerhalb der jeweiligen Randbedingungen)
- verständlich – die Stakeholder nötigen keine zusätzlichen Erläuterungen
- unzweideutig (eindeutig) – nicht auf unterschiedliche Art interpretierbar
- vollständig - alle notwendigen Anforderungen sind vorhanden
- konsistent – die Anforderungen widersprechen einander nicht
- bewertet nach Wichtigkeit und/oder Stabilität
- prüfbar
- modifizierbar – die abzuändernden Stellen können identifiziert werden
- verfolgbar (traceable) – über einen eindeutigen Identifikator sichergestellt sein.

Laut Niebisch sind dabei die folgenden beiden Punkte besonders hervorzuheben:

„Prüfbarkeit bedeutet, dass jene Person, die die Leistung abnehmen muss, dies anhand definierter Parameter tun muss. Nur so kann festgestellt werden, ob die Anforderungen erfüllt sind, oder nicht. Die Eindeutigkeit schließt Interpretationsmöglichkeiten aus, es darf kein Interpretationsspielraum offen bleiben. Niebisch schlägt aus diesem Grund vor, die Anforderungen in aktiver Form zu verfassen.“ (Niebisch 2013, 36)

Bei der Vollständigkeit ist darauf zu achten, auch jene Anforderungen zu berücksichtigen, die scheinbar ohnehin selbstverständlich sind. Außerdem müssen alle Funktionalitäten beschrieben werden, es darf nichts fehlen.

Zastrow und Wagner weisen darauf hin, dass die ersten drei Kriterien sich durch Verwendung der SOPHIST® Satzschablone erfüllen lassen (Zastrow/Wagner 2016, 4)

Dabei werden die Dokumente nach unterschiedlichen Kriterien überprüft, z.B. ob sie bestimmten organisatorischen oder projektspezifischen Richtlinien entsprechen, ob sie der vorgegeben Struktur folgen und auch ob sie verständlich sind.

Wird die Dokumentation nicht überprüft, kann dies längerfristig unangenehme Folgen nach sich ziehen:

- Die Behinderung von Entwicklungsaktivitäten: Entwicklungsaktivitäten, die auf einem festgelegten Dokumentationsformat beruhen, können nicht durchgeführt werden
- Unverständlichkeit von Anforderungen: Die Anforderungen sind für die Adressaten nicht verwendbar, bzw. nicht verständlich
- Unvollständigkeit: relevante Informationen sind nicht in den Anforderungen dokumentiert
- Übersehen von Anforderungen: Sind Anforderungen nicht an der vorgeschriebenen Stelle im Anforderungsdokument definiert, so können diese Anforderungen leicht von Stakeholdern der Folgeaktivität übersehen werden. (Pohl/Rupp 2015, 99)

Deshalb muss darauf geachtet werden, dass durch die Überprüfung von Dokumenten möglichst alle Fehler und Unzulänglichkeiten entdeckt werden. Es empfiehlt sich auf folgende Kriterien großen Wert zu legen:

- Konformität zum Dokumentationsformat und zur Dokumentenstruktur: Hier stellt sich die Frage, ob das vorgeschriebene Format zur Dokumentation, die Struktur, bzw. die vorgegebene Sprache verwendet wurde und ob alle Anforderungen an der zuvor festgelegten Stelle dokumentiert wurde.
- Verständlichkeit: An dieser Stelle ist zu fragen, ob die dokumentierten Anforderungen verstanden wurden und ob die verwendeten Begriffe in ein Glossar aufgenommen und entsprechend definiert wurden.
- Eindeutigkeit: Wichtig ist hier zu überprüfen, ob die Dokumentation auch in Bezug auf die verwendete Sprache nicht mehrdeutig zu verstehen ist, ob keine unterschiedlichen Interpretationsmöglichkeiten vorhanden sind.
- Konformität mit Dokumentationsregeln: Hier ist zu fragen, ob die verwendete Modellierungssprache den vorgegebenen Regeln entspricht und ob die zuvor bestimmten Dokumentationsregeln und –richtlinien beachtet wurden. (siehe Pohl/Rupp 2015, 99f)

Qualitätsaspekt Abgestimmtheit

Hier soll die Frage gestellt werden, ob es Mängel bei der Abstimmung der Anforderungen mit den Stakeholdern gibt. Da sich die Stakeholder während des Projektverlaufs immer mehr Wissen über das entstehende System aneignen, könnte es passieren, dass sich die Meinungen über bestimmte Anforderungen mit der Zeit ändern. Diesen Änderungswünschen muss man natürlich entgegenkommen, nur dürfen die weiteren Entwicklungsschritte dadurch nicht beeinträchtigt werden.
Hinsichtlich der Abgestimmtheit ist auf Folgendes zu achten:

- Abstimmung: dabei ist die Frage zu stellen, ob jede Anforderung mit allen relevanten Stakeholdern abgestimmt wurde
- Abstimmung nach Änderung: Dabei muss darauf geachtet werden, dass für jede geänderte Anforderung erneut die Zustimmung der Stakeholder eingeholt wurde
- Konflikte aufgelöst: Es muss überprüft werden, ob alle Konflikte bzgl. der Anforderungen aufgelöst werden konnten. (siehe Pohl/Rupp 2015, 100)

Folgende Prinzipien bei der Prüfung von Anforderungen sollten beachtet werden:

1. Prinzip: Beteiligung der richtigen Stakeholder: Wurden die richtigen Stakeholder an der Prüfung beteiligt?
2. Prinzip: Trennung von Fehlersuche und Fehlerkorrektur: Die Suche nach Fehlern und die Korrektur sollte nicht zum gleichen Zeitpunkt erfolgen
3. Prinzip: Prüfung aus unterschiedlichen Sichten: Die Prüfung der Anforderungen sollte aus unterschiedlichen Blickwinkeln erfolgen
4. Prinzip: Geeigneter Wechsel der Dokumentationsform: Wurde die Dokumentationsform angepasst?
5. Prinzip: Konstruktion von Entwicklungsartefakten
6. Prinzip: Wiederholte Prüfung: Eventuell sollte die Prüfung wiederholt werden. (siehe Pohl/Rupp 2015, 101ff)

Qualitätssicherung

Qualitätssicherung ist ein Teilbereich des RE und beschäftigt sich mit der Qualität von Anforderungen und den Anforderungsdokumenten.
Man unterscheidet konstruktive und analytische Qualitätssicherung. Konstruktive Qualitätssicherung begegnet Auffälligkeiten in den Anforderungen gleich zu Beginn, d.h. beim Ermitteln und Dokumentieren. Bei der analytischen Qualitätssicherung wird die Qualität bereites dokumentierter Anforderungen vor weiteren Entwicklungsphasen überprüft und gegebenenfalls verbessert. Zielführend ist es, so früh wie möglich festzulegen, welche Qualitätsziele erreicht werden müssen. Sobald die Anforderungen stabil sind, sollten sie getestet werden. Ein Qualitätssicherungsleitfaden hilft bei diesem Prozess. Bei einer agilen Vorgehensweise sollten die Qualitätssicherungsmaßnahmen in die Definitions of Ready und in die Definitions of Done aufgenommen werden. Im Qualitätssicherungsleitfaden sollten laut Rupp folgende Schritte unternommen werden (siehe Rupp/Sophisten 2014, 307ff):

1. Qualitätsziele festlegen

Diese sollten schon vor dem Schreiben der Anforderungen definiert sein. Die Qualitätssicherung sollte vor allem dort liegen, wo Qualität für das Projekt ausschlaggebend ist, dazu zählen fachliche Richtigkeit, Vollständigkeit und verständliche Struktur. Qualitätsansprüche dürfen jedoch nicht zu hoch angesetzt werden.

2. Qualitätssicherungsmethoden auswählen

Die formalen Kriterien, die hier geprüft werden sollen, sollten so auch schon in den Anforderungen genannt worden sein. Man kann sich dabei an den Prüftechniken für Anforderungen orientieren.

3. Qualitätszeitpunkte definieren

Es sollte möglichst früh mit der Qualitätssicherung begonnen werden. Eingeplant werden sollten mehrere Qualitätssicherungszyklen für die Funktionalitäten. In einem agilen Umfeld kann eine Spezifikationsphase eingeplant werden, bzw. kann parallel zum Software-Inkrement auch die Anforderungsdokumentation weitergeschrieben werden.

4. Prüfer ernennen

Zusätzlich zu den Stakeholdern, die das fachliche Wissen mitbringen, können auch externe Auditoren hinzugezogen werden, denen formale Fehler eventuell rascher auffallen. Sie haben im Allgemeinen auch einen guten Blick auf eine mögliche Prozessoptimierung. Teamgrößen von zwei bis drei Prüfern sorgen für die maximale Effizienz, bei vier bis fünf Prüfern erreicht man die höchste Effektivität. Es besteht die Möglichkeit, sowohl mit einer fixen als auch mit einer variablen Anzahl an Prüfern zu arbeiten, als auch mit wechselnden Prüfern zu arbeiten.

Die Qualitätssicherung wird vorbereitet, indem zunächst die Prüfbarkeit festgestellt wird. Dabei wird in einem ersten Schritt überprüft, ob das Anforderungsdokument den dem richtigen Format entspricht, bzw., ob alle wesentlichen Informationen enthalten sind. In einem weiteren Schritt wird dann der Prüfgegenstand definiert. Da der Aufwand, alle Anforderungen zu überprüfen zu hoch wäre, beschränkt man sich meist auf eine Stichprobe, d.h. man muss zunächst eine Wahl treffen, was überhaupt geprüft werden soll. Danach wird der Prüfgegenstand extrahiert und dokumentiert. Dabei wird für die zu prüfende Anforderung ein eigenes Dokument, eine so genannte Prüf- oder Messvorlage angelegt. Der Prüfer trägt in dieser Datenbank oder Tabelle sämtliche Auffälligkeiten, bzw. Messwerte ein. Wichtig dabei ist, dass diese Ergebnisse eine eigene ID und damit eine Referenz auf die Originalanforderungen erhalten. (siehe Rupp/Sophisten 2014, 313)

Bei der Durchführung der Qualitätsprüfung ist es wichtig eine einheitliche Vorlage und einen einheitlichen Qualitätssicherungsleitfaden zu verwenden. Der ausgewählte Teil der Anforderung wird nun nach vorgegebenen Kriterien geprüft und die Ergebnisse in das Dokument eingetragen. Parallel dazu werden Prüfberichte verfasst, die Informationen zum Ablauf und Kommentare zu Auffälligkeiten beinhalten. Auch Unklarheiten in Bezug auf die Vorgaben zur Qualitätsmessungen und Probleme, die während des Messens aufgetreten sind, werden protokolliert. All diese Anmerkungen helfen auch einzuschätzen, ob die Qualität der ausgewählten Stichprobe vom Gesamteindruck der Spezifikation abweicht. (siehe Rupp/Sophisten 2014, 314)

Am Ende des Prozesses werden die Ergebnisse beurteilt. Bei einem Scrum-Projekt sollte an dieser Stelle der Scrum-Master hinzugezogen werden. Bei der Beurteilung wird auch darauf geachtet, ob das Prüfergebnis mit dem Gesamteindruck übereinstimmt, bzw. sollte man das aktuelle Prüfergebnis mit den vorherigen Ergebnissen vergleichen. Danach erfolgt die Überlegung, wie man Verbesserungen erwirken könnte, das könnte sowohl die Notationstechniken betreffen als auch den Prozess der Anforderungsermittlung. Zum Schluss erhält man eine Liste von Mängeln, Fehlern und Anmerkungen zu den jeweiligen Anforderungen. Diese müssen in die Anforderungen eingearbeitet werden, Fehler müssen korrigiert werden. Die Stakeholder

werden in so einem Fall erneut herangezogen, bei divergierenden Stakeholdermeinungen hat eine Entscheidung zu fallen. Man sollte jedoch in dieser Phase aufpassen, nicht neue Fehler zu produzieren, indem man beispielsweise neue Widersprüche zwischen den Anforderungen erzeugt. Eine erneute Prüfung gibt dabei Klarheit. (siehe Rupp/Sophisten 2014, 315f)

Um die Qualität der Dokumentation zu prüfen, muss auf Folgendes geachtet werden: Die Dokumente müssen konform zum Format und zur Struktur der Dokumentation sein, sie müssen verständlich und eindeutig sein. Außerdem muss die Syntax der Modellierungssprache eingehalten werden.

Des Weiteren muss darauf geachtet werden, dass die initiale Abstimmung und dass eine Abstimmung nach einer Änderung durchgeführt wurden und auch dass mögliche Konflikte bei der Abstimmung aufgelöst wurden.

Da nicht alles zugleich umgesetzt werden kann, ist auch auf Priorisierung zu achten. Dabei ist alles, was zwingend umgesetzt werden muss, an den Anfang zu stellen.

Generelle Aussagen müssen immer auf eventuelle Ausnahmen überprüft werden, offene Punkte immer angesprochen werden. Anforderungen sollen neutral gehalten werden, sie sollen beschreiben, was verlangt wird, aber nicht den Weg dorthin.

Wichtig ist laut Zastrow und Wagner auch folgendes:

„Anforderungen, die mehrere Aspekte beinhalten, sollten in mehrere einzelne Anforderungen aufgeteilt werden. Das vereinfacht die Aufgaben der Entwickler wie auch der Tester. Erkennbar sind solche Anforderungen an dem Wort „und“, z.B.: „Der Kunde muss bei der Registrierung AGB und Datenschutzerklärung bestätigen.“ Hier ist es sinnvoll, zwei separate Anforderungen zu formulieren. Dadurch können verschiedene Teammitglieder mit der Umsetzung betraut werden und diese können die Aufgaben einzeln abarbeiten und als erledigt markieren. Dies erleichtert die Kontrolle des Projektfortschritts.“ (Zastrow/Wagner 2016, 5).

Für die Überprüfungen gibt es bereits mehrere Standards. Sehr bekannt ist die ISO 9000:2005 Normenfamilie, die grundlegende Techniken und Mindestanforderungen an Qualitätsmanagementsysteme vorschreibt. Da dieses Werk sehr abstrakt ist, muss es erst in Anweisungen für die Anforderungsanalyse umgewandelt werden. Rupp verweist jedoch auch auf andere Qualitätsstandards, zum Beispiel auf Bootstrap, Software Process Improvement and Capability Determination (SPICE), zum Durchführen von Bewertungen von Unternehmensprozessen, entwickelt, die Capability Maturity Model

Integration, das 1996 auf Initiative des US-Verteidigungsministeriums das Software Engineering Institute entstand und die IT infrastructure Library (ITIL), eine Sammlung von Best Practices zur Umsetzung eines IT-Service-Managements. (siehe Rupp/Sophisten 2014, 55)

Diese Standards konzentrieren sich jedoch nicht auf konkrete Handlungsanweisungen. Folgende Aspekte sind bei der Analyse von Anforderungen zu beachten:

Eine **Zuverlässigkeitsanalyse** sollte folgendes prüfen:

- Eine zuverlässige Eingabeverarbeitung
- Eine zuverlässige Ausführung
- Eine zuverlässige Fehlermeldung und Diagnostik
- Die Möglichkeit des Wiederaufsetzens im Fehlerfall

Eine **Durchführbarkeitsstudie** sollte prüfen, ob die Anforderungen an ein geplantes System unter technischen, personellen und ökonomischen Aspekten realisierbar und sinnvoll sind. Bei der Überprüfung der technischen Durchführbarkeit wird darauf geachtet, ob die gestellten Probleme überhaupt lösbar sind und wenn ja, ob sie unter den gegebenen Umständen lösbar sind. Bei einer personellen Durchführbarkeitsstudie wird geschaut, ob für die Herstellung, für die Einführung und für den Betrieb genügend Personal mit ausreichender Qualifikation zur Verfügung steht. Eine ökonomische Durchführbarkeitsstudie wird der erwartete Nutzen des geplanten Systems mit den zu erwarteten Projektkosten in Beziehung gesetzt. (Kosten-Nutzen-Analyse) Auch ob das Projekt im vorgesehenen Zeitrahmen unter akzeptierbaren Kosten durchführbar ist, ist zu klären. Risiken müssen ebenfalls abgeschätzt werden, z.B. technische Risiken, Kostenplanrisiken (dazu gehören die Verfügbarkeit, Zuverlässigkeit und Stabilität der verwendeten Software, die Verfügbarkeit von Personal), Umgebungsrisiken, wie etwa die Fähigkeiten der Nutzer und so genannte Interaktionseffekte, die sich aus dem Zusammenwirken verschiedener und verschiedenartiger Komponenten ergeben. (siehe Partsch 1991, 52)

Beim Prüfen von Anforderungen ist einerseits darauf zu achten, ob die Anforderungen den Qualitätsanforderungen entsprechen und andererseits, ob sie für die weiteren Schritte, z.B. Entwicklung oder Tests freigegeben werden können. Das ist oft auch Teil des Vertrags zwischen Auftragnehmer und Auftraggeber. Besonderes Augenmerk sollte

auf Mehrdeutigkeit, Widersprüchlichkeit und Unvollständigkeit der Anforderungen gelegt werden. Die Abstimmung von Anforderung, die während des gesamten Prozesses erfolgen muss und dementsprechend von Beginn an auch einkalkuliert werden sollte, erfordert unter anderem auch viel Feingefühl des RE, da es sich nicht nur um Konflikte zwischen den Anforderungen sondern auch zwischen den Stakeholdern handeln kann. Werden Konflikte nicht aufgelöst, ist die Akzeptanz des Systems gefährdet, bzw. kann das gesamte Entwicklungsprojekt an einzelnen Stakeholder, bzw. Stakeholdergruppen scheitern. Aus Widersprüchlichkeiten können jedoch auch neue, innovative Ideen entstehen. Wichtig ist, dass der Umgang mit Widersprüchlichkeiten offen und transparent erfolgt. Nur so wird das System akzeptiert.

2.13.1 Qualitätsaspekte der Anforderungen

Um die Qualität von Anforderungen objektiv und einheitlich überprüfen zu können, ist es wichtig, dass bestimmte Qualitätskriterien schon vorab konkretisiert und festgelegt wurden. Pohl und Rupp setzten drei Hauptziele bei der Überprüfung von Anforderungen fest:

- Inhalt: Wurden alle relevanten Anforderungen ermittelt und im erforderlichen Detaillierungsgrad erfasst?
- Dokumentation: wurden die Anforderungen gemäß der festgelegten Dokumentations- und Spezifikationsvorschriften dokumentiert?
- Abgestimmtheit: Stimmen alle Stakeholder mit den dokumentierten Anforderungen überein und sind alle bekannten Konflikte aufgelöst? (Pohl/Rupp 2015, 97)

Die Freigabe der Anforderungen sollte nur dann erfolgen, wenn alle drei Qualitätsaspekte geprüft wurden.

Qualitätsaspekt Inhalt

Auf dieser Ebene wird geprüft, ob es keine inhaltlichen Fehler in der Anforderung gibt, die die Entwicklung eines Systems negativ beeinflussen könnte. Pohl und Rupp schlagen vor, folgende Fehlerarten zu prüfen:

- Vollständigkeit (Menge aller Anforderungen): Sind alle relevanten Anforderungen an das System (für das nächste Systemrelease) erfasst?

- Vollständigkeit (einzelner Anforderung): Beschreibt jede Anforderung alle für diese Anforderung notwendigen Informationen? Sind die Anforderungen messbar? Unvollständige Anforderungen müssen als solche gekennzeichnet werden, am besten durch „tbd“ (to be done/tob e determined). Diese Marker können später in den Texten gesucht werden.
- Verfolgbarkeit: Sind alle relevanten Verfolgbarkeitsbeziehungen definiert (z.B. zu relevanten Anforderungsquellen)? Erleichtert wird dies durch die Vergabe einer eindeutigen Nummer, die die Anforderung den gesamten Lebenszyklus hindurch begleitet. Dadurch können Anforderungen verschiedener Spezifikationsebene miteinander verbunden werden
- Korrektheit/Adäquatheit: Spiegeln die Anforderungen die Bedürfnisse und Wünsche der Stakeholder in angemessener Weise wider?
- Konsistenz: Sind alle definierten Anforderungen an das geplante System gemeinsam erfüllbar bzw. stehen die Anforderungen nicht miteinander im Widerspruch? UML-Diagramme helfen dabei, die Übersicht zu wahren und Inkonsistenzen zu erkennen.
- Keine vorzeitigen Entwurfsentscheidungen: Wurden Entwurfsentscheidungen in den Anforderungen vorweggenommen, die nicht durch Randbedingungen induziert sind (z.B. Randbedingungen, die durch die Verwendung einer Client/Server-Architektur vorschreibt)?
- Überprüfbarkeit: können basierend auf den Anforderungen Abnahme- bzw. Prüfkriterien definiert werden bzw. sind diese bereits definiert? Jede Anforderung muss testbar sein.
- Notwendigkeit: Trägt jede Anforderung zur Erfüllung eines definierten Ziels bei? Ist die Anforderung gültig und aktuell? Können die Anforderungen zu den Zielen zurückverfolgt werden? Anforderungen, die nur bis zu einem bestimmten Zeitpunkt gültig sind, sollten als solche gekennzeichnet werden.
- Atomar: Jeder Anforderungssatz enthält genau eine Anforderung, ein Vollverb (Prozesswort). Konjunktionen, die mehrere Anforderungen in einem Anforderungssatz ermöglichen, werden nicht verwendet.
- Technisch lösungsneutral: Die Anforderung beschreibt nur, was gefordert wird, jedoch nicht, wie die Anforderung umgesetzt werden soll, es sollen keine unnötigen Einschränkungen für die Architektur und für die Entwicklung enthalten sein. Anforderungen sollten deshalb auf die Beschreibung des Problems konzentriert sein.
- Realisierbarkeit: Die Anforderung muss innerhalb der Grenzen des Systems und der Umgebung umgesetzt werden können.
- Eindeutigkeit: Die Anforderung muss klar und eindeutig formuliert sein und von allen Beteiligten verstanden werden. (Rupp/Sophisten 2014, 26-28 und Pohl/Rupp 2015, 98)

Auch wenn die einzelnen Anforderungen all diesen Kriterien entsprechen, kann es passieren, dass die Anforderungsspezifikation dennoch nicht den Qualitätskriterien entspricht. Die Spezifikation muss einigen Kriterien entsprechen:

- Vollständig: für jede gewünschte Funktionalität müssen sämtliche geforderten Reaktionen des Systems angegeben werden, auch hinsichtlich der Qualität, der Reaktionszeiten, der Verfügbarkeit und Bedienbarkeit. Es dürfen in der Spezifikation keine unvollständigen Anforderungen enthalten sein. Überall wo beispielsweise „tbd“ steht müssen Fragen geklärt und Lücken aufgefüllt werden. Sämtliche Grafiken und Tabellen müssen beschriftet sein, es müssen ein Quellen- und ein Abkürzungsverzeichnis sowie Begriffsdefinitionen vorhanden sein.
- Konsistent: es dürfen keine widersprüchlichen Anforderungen und keine Doppelungen enthalten sein. Begriffe müssen immer gleich verwendet werden, Glossare helfen dabei.
- Erschwinglich: Anforderungsspezifikationen sollten mit einer Lösung umsetzbar sein, die hinsichtlich der Randbedingungen vorgegeben sind, wie technische, rechtliche oder behördliche Vorgaben, Kosten- und Zeitpläne.
- Abgegrenzt: es sollten keine Anforderungen enthalten sein, die von den Stakeholdern nicht gefordert sind (siehe Rupp/Sophisten 2014, 29)

Zusätzlich zu diesen Kriterien erleichtern folgende Überlegungen den Umgang mit Anforderungen und Anforderungsspezifikationen:

- Bewertung: Wenn ein System relativ komplex und groß ist, sollten Anforderungen nach Wichtigkeit und Priorität bewertet werden.
- Rechtliche Verbindlichkeit: es muss deutlich gekennzeichnet werden, dass bestimmte Anforderungen juristisch einklagbar sind.
- Sortierbarkeit: Anforderungen sollten nach unterschiedlichen Kriterien gereiht werden können.
- Mehrbenutzerfähig: der gemeinsame Zugriff auf die Anforderungsspezifikation muss möglich sein und es muss sichergestellt werden, dass das Überschreiben von Informationen vermieden wird. Sämtliche Eintragungen und Änderungen sollten mit Angabe des jeweiligen verantwortlichen Autors protokolliert werden.
- Modifizierbar und Erweiterbar: Änderungen und Ergänzungen müssen möglich sein. Anforderungsspezifikationen sollten auch weiterverwendet werden können. (siehe Rupp/Sophisten 2014, 29f)

Rupp spricht auch von unterschiedlichen Ebenen auf denen sich Spezifikationen befinden können. Dabei ist jedoch zu beachten, dass die einzelnen Levels nicht streng voneinander abgetrennt sind, da sie je nach Situation mal verfeinert und mal abstrahiert werden.

- Spezifikationslevel 0: Hier findet man eine grobe Systembeschreibung, ein Mission Statement. In wenigen Sätzen wird hier der Gesamtumfang des zu entwickelnden Systems umrissen. Damit soll ein erster Überblick ermöglicht werden, es sind darin Absichtserklärungen und Ziele verpackt.
- Spezifikationslevel 1: Ein Anwendungsfall (Use Case) wird vorgestellt in dem die Teilschritte beschrieben werden.

- Spezifikationslevel 2: Anwenderforderungen, Nutzeranforderungen, Testfälle, User-Storys sind vorhanden. Beim agilen Vorgehen spricht man auf diesem Level auch vom Backlog Item. Hier werden fachliche Abläufe definiert.
- Spezifikationslevel 3: Dabei handelt es sich um detaillierte Anwenderforderungen, technische Anforderungen, Schnittstellenbeschreiben usw. Die Level-2-Anforderungen werden verfeinert, hier werden auch die nichtfunktionalen Anforderungen berücksichtigt.
- Spezifikationslevel 4: Dazu gehören Komponentenanforderungen, ein Pflichtenheft, Testfälle, … Das System wird in Soft- und Hardware getrennt bzw. in weitere Bestandteile. Dabei ist zu beachten, dass Anforderungen an einen Teil des Systems , die für das Gesamtsystem in Spezifikationslevel 4 definiert werden, mit Anforderungen auf Spezifikationsebene 1 für einen anderen Betrachtungsgegenstand gleichgesetzt werden können. (siehe Rupp/Sophisten 2014, 39ff)

Qualitätssicherung ist zwar aufwändig, bei größeren Projekten aber unerlässlich.

2.13.2 Qualitätssicherung des Anforderungsmanagements

Hier wird überprüft, ob mit genügend Sorgfalt gearbeitet wurde. Man benötigt dazu die Checklist der verwendeten Prozesse, gültige Versionen der Anforderungen, Ressourcen – und Zeitpläne. Meist wird bzgl. der Prozesse geprüft, ob ein Stakeholdermanagement durchgeführt wurde und ob man sich an Vorgaben gehalten hat. Bei der Anforderungsverwaltung wird geschaut, ob es eine klare Struktur in der Anforderungsspezifikation gibt, ob Unternehmensstandards eingehalten wurden und ob beispielsweise UML für die richtigen Fälle angewandt wird. Bzgl. der einzelnen Anforderungen wird geprüft ob die Anforderungen klar und konsistent beschrieben sind, ob die Anwendungsfälle in Schritten zerlegt sind, ob Aktivitätsdiagramme vorhanden sind, ob alles verständlich ist, ob beispielsweise mit kurzen und einfachen Satzstrukturen gearbeitet wurde und ob sich kein „sollte“ oder „könnte“ gehalten hat.

2.13.3 Techniken zur Überprüfung der Anforderungen

Im Leitfaden sollten die herangezogenen Prüftechniken kurz erklärt werden und die jeweiligen Verantwortlichen für die Auswahl, die Durchführung und die Nachbereitung der Prüfung festgelegt werden. In der derzeit vorhandenen Fachliteratur werden meist einige Prüftechniken vorgeschlagen (siehe auch Rupp/Sophisten 2014, 318ff):

Reviews

Dabei lassen sich drei Ausprägungsformen unterscheiden: Stellungnahme, Inspektion und Walkthrough.

Stellungnahme

Dabei wird der Prüfgegenstand von einer fachlich versierten bzw. einer analytisch begabten Person geprüft. Das erfordert relativ wenig Aufwand, das Ergebnis ist jedoch sehr stark vom Qualitätsbewusstsein der einzelnen Personen abhängig, Verständnisprobleme können das Ergebnis verzerren. Für sicherheitsrelevante Fragen ist diese Technik deshalb ungeeignet.
Der oder die AnforderungsautorIm übergibt diese einer dritten Person um von dieser eine fachliche Meinung über die Qualität zu erhalten. Diese Person überprüft die Anforderung mit dem Blick auf mögliche Qualitätsmängel, wie Fehler oder Mehrdeutigkeiten. Wesentlich ist, dass an dieser Stelle vorher festgesetzte Qualitätskriterien überprüft werden.

Inspektion

Bei der Inspektion von Software, auch technische Reviews genannt, werden Entwicklungsartefakte anhand eines Prozessschemas systematisch nach Fehlern durchsucht. Die Inspektion kann in die Phasen Planung, Übersicht, Fehlersuche, Fehlersammlung unterteilt werden. In einem weiteren Schritt sind dann noch Korrektur, Nachkontrolle und Reflexion nötig. In der ersten Phase, der Planung, wird das Ziel der Inspektion festgelegt, die zu überprüfenden Ergebnisse festgelegt, die Art der Durchführung gewählt und die Rollen, die die Teilnehmer an der Inspektion einnehmen sollen, festgelegt. Darauf folgt die Übersichtsphase, in der der Autor oder die Autorin der Anforderung die Anforderungen, die geprüft werden sollen, erklärt, damit alle TeilnehmerInnen am gleichen Stand des Wissens sind. In der Phase der Fehlersuche kann im Team oder individuell gearbeitet werden. Die Arbeit im Team kann zu Synergieeffekten führen, die individuelle Arbeit hat dagegen den Vorteil dass sich die

Personen, die nach Fehlern suchen, besser konzentrieren können. Wichtig ist auch hier, dass sämtliche Ergebnisse dokumentiert werden. Bei der Fehlersammlung, werden die Ergebnisse zusammengetragen. Hier ist eine Konsolidierung nötig um unechte Fehler, die durch falsche Kommunikation oder Fehlinterpretation als solche aufgefasst wurden, erkennen zu können oder auch doppelte Fehler zu identifizieren. Sämtliche Ergebnisse werden in eine Fehlerliste eingetragen.

Pohl und Rupp schlagen folgende Rollen bei einer Inspektion vor:

- Der Organisator plant und überwacht den Inspektionsprozess
- Der Moderator, einen möglichst neutrale Person, leitet die Sitzung und überwacht die Einhaltung des Prozessschemas
- Der Autor erläutert den Inspektoren, die von ihm erstellten Anforderungen in der Übersichtsphase und übernimmt später die Verantwortung für die Behebung von aufgedeckten Fehlern.
- Der Vorleser stellt die zu inspizierenden Anforderungen sukzessive vor und führt die Inspektion durch die zu untersuchende Anforderung. Diese Aufgabe sollte von einer neutralen Person übernommen werden, meist übernimmt der Moderator diese Rolle
- Die Inspektoren haben die Aufgabe, Fehler zu finden und diese dem Inspektionsteam zu kommunizieren.
- Der Protokollant führt ein Protokoll über die Ergebnisse der Inspektion. (siehe Pohl/Rupp 2015, 106)

Die Inspektion ist recht aufwändig aber effektiv, man findet recht viele Auffälligkeiten.

Walkthrough

Mit einem Walkthrough wird geprüft, inwieweit die Anforderungen akzeptiert werden. Ein Walkthrough ist weniger strikt angelegt als eine Inspektion, auch die Rollen sind nicht so vielfältig und nicht so strikt voneinander getrennt. Wichtig sind ReviewerInnen, AutorInnen und ProtokollantInnen. Ziel dieser Methode ist es, gemeinsam Qualitätsmängel in den Anforderungen zu entdecken und für alle ein akzeptierbares Verständnis für die perfekte Anforderung zu erzielen. In der Vorbereitungsphase werden die Anforderungen an die teilnehmenden Personen verteilt und von diesen auf Qualitätsmängel untersucht. In der Walkthrough-Sitzung wird schrittweise die Anforderung vorgestellt, die TeilnehmerInnen haben die Möglichkeit jederzeit Feedback zu geben und miteinander zu diskutieren, die Qualitätsmängel werden in der Gruppe diskutiert und vom Protokollanten festgehalten. Ein großer Nachteil ist, dass der

oder die AnforderungsautorIn von den Schwächen ablenken kann und dass es ohne Moderation oft zu wenig produktiven Diskussionen kommt.

Prototyp/Simulationsmodell

Dabei werden die Anforderungen bereits zum Teil umgesetzt um die Realisierbarkeit zu überprüfen. Fehlende, bzw. ungeeignete Funktionalitäten können so rasch aufgezeigt werden. Prototypen können in allen Phasen der Analyse verwendet werden. Für die Stakeholder sind die Ergebnisse direkt erlebbar, auch bei recht komplexen Systemen. Prototypen können auch für weitere Diskussionen gut eingesetzt werden. Sie sind jedoch sehr zeitaufwändig und man muss darauf achten, dass der Prototyp von den zukünftigen AnwenderInnen nicht mit dem fertigen Produkt gleichgesetzt wird, bzw. eine sehr rasche Umsetzung erwartet wird.

Testfälle

Mit einem Testfall schafft man die Basis für die Verifikation eines Prüfgegenstandes. Bei der Verifikation gilt es herauszufinden, ob eine Anforderung bestimmte Kriterien erfüllt. Testfälle dienen auch der Verbesserung der Anforderungen, denn der oder die ErstellerIn des Testfalls muss die Anforderungen gründlich lesen und damit auch prüfen. Testfälle sollten unbedingt im Rahmen des RE erstellt werden, denn so lassen sich gegebenenfalls Anforderungen noch ändern. In agilen Projekten spricht man dabei von Akzeptanzkriterien. Konkrete Testfälle helfen die Anforderungen zu veranschaulichen, abstrakte Testfälle sind dafür meist leichter verständlich. Natürlichsprachliche Testfälle beinhalten die Ausgangssituation, das Testereignis und das erwartete Ereignis. Mit Testfällen kann man rasch Lücken oder Fehler in den Anwendungen erkennen, sie sind jedoch recht aufwändig. Wichtig ist, dass die Testfälle von einer anderen Person stammen als die Anforderungen. Details zu Anforderungen können in Testfällen notiert werden. In Scrum-Projekten werden diese Akzeptanzkriterien als Verfeinerungen von User-Storys beschrieben. Testfälle lassen sich für alle Arten von Anforderungen einsetzen und können als Basis von Testdurchführungen dienen. Konsistenz und Notwendigkeit kann damit jedoch nicht überprüft werden. Für das Erstellen von Testfällen ist relativ viel Vorwissen nötig.

Analysemodell

Dieses besteht aus einem oder mehreren Diagrammen, z.B. UML-Klassendiagrammen oder Zustandsdiagrammen, man kann damit die statische, bzw. die dynamische Sicht eines Systems zeigen. Fehlendes oder Fehlerhaftes wird rasch erkannt. Durch die Anschaulichkeit eignet sich das Analysemodell auch gut als Diskussionsgrundlage mit den Stakeholdern. Inkonsistenzen werden damit rasch sichtbar. Auch das Glossar kann damit abgeglichen werden, natürlichsprachliche Anforderungen können besser strukturiert werden. Der Einsatz des Analysemodells setzt jedoch Fachwissen und Erfahrung voraus.

Perspektivenbasiertes Lesen

Gemeinsam mit der Stellungnahme und dem Walkthrough wird auch oft das perspektivenbasierte Lesen verwendet. Hier nimmt man unterschiedliche Rollen ein um eine ganz spezielle Sicht auf eine Anforderung zu erhalten und sich auf bestimmte Aspekte gut konzentrieren zu können. Perspektivenbasiertes Lesen bietet sich an, wenn man Dokumente aus einer bestimmten Sicht heraus interpretieren möchte. Meist nimmt man bei dieser Methode, jene Perspektiven ein, die mit den Adressaten der Anforderung übereinstimmen. Das sind vor allem die Perspektiven

- Kunde/Nutzer: Hier wird geprüft, ob die Anforderungen die gewünschte Funktionalität und Qualität beschreiben
- Softwarearchitekt: Hier wird vor allem überprüft, ob in der Anforderung alle für den Architektenentwurf benötigten Informationen enthalten sind (z.B. alle relevanten Performanzeigenschaften)
- Tester: Hier wird geprüft, ob sich Testfälle aus den Anforderungen ableiten lassen. (siehe Pohl/Rupp 2015, 107)

Ebenso können hier die drei bereits erwähnten Qualitätsaspekte als mögliche Perspektiven zur Überprüfung von Anforderungen herangezogen werden. Die Perspektive „Inhalt“ fokussiert auf die inhaltliche Qualität der Anforderung, die Perspektive „Dokumentation“ achtet auf die Einhaltung von Dokumentationsvorschriften für Anforderungen und für Anforderungsdokumente und bei der die Perspektive „Abgestimmtheit“ wird vor allem darauf geachtet, dass sämtliche Konflikte aufgelöst wurden.

Je nach Projekt können auch weitere Perspektiven eingenommen werden. Wichtig bei dieser Methode ist, dass alle Beteiligten ausreichend Informationen über die jeweiligen Perspektiven erhalten um entsprechend handeln zu können. Ein Fragenkatalog, der an die jeweiligen PrüferInnen gerichtet ist und von diesen beantwortet werden sollen, ist ebenso hilfreich, wie eine Checkliste, auf der die wichtigsten Inhalte vermerkt sind. Nach Abschluss des perspektivenbasierten Lesens werden die Ergebnisse gesammelt, miteinander verglichen und vorzugsweise im Rahmen einer Gruppensitzung konsolidiert.
Diese Technik kann eigenständig oder gemeinsam mit anderen Techniken verwendet werden, zum Beispiel mit Inspektion oder Review.

Prüfung durch Prototypen

Beim so genannten Prototyping wird die Akzeptanz spezieller Anforderungen mit den Stakeholdern überprüft. Der Vorteil dieser Methode liegt darin, dass Anforderungen erlebbar sind und direkt ausprobiert werden können. Damit werden Fehler am ehesten sichtbar. Außerdem erkennen die Stakeholder mögliche Diskrepanzen zwischen ihren Vorstellungen über die Anforderungen und der Realität. Diese Methode bietet sich vor allem für das Überprüfen von nichtfunktionalen Anforderungen an, wie z.B. bei der Prüfung der Usability.
Bei dieser Methode unterscheidet man zwischen Wegwerfprototypen, die nur für die Überprüfung verwendet werden und danach nicht weiterentwickelt werden und evolutionären Prototypen, die später weiterentwickelt werden.
Bevor ein Prototyp angefertigt werden kann, müssen jene Anforderungen ausgewählt werden, die damit auch überprüfbar sind und die im Rahmen der zeitlichen und budgetären Möglichkeiten liegen. Um einen möglichst hohen Nutzen aus dieser Methode zu gewinnen, müssen vor dem Testen gewisse Vorbereitungen getroffen werden.

- Die PrüferInnen müssen mit den notwendigen Informationen versorgt werden um den Prototyp optimal bedienen zu können. Hier sind ein Handbuch oder Schulungen notwendig
- Es empfiehlt sich, Prüfszenarien vorzubereiten, wie etwa Nutzungsabläufe, die die Prüfpersonen durchführen können

- Die Prüfpersonen sollten anhand einer Checkliste den Prototyp testen können. (siehe Pohl/Rupp 2015, 109f)

Bei dieser Methode ist wichtig, dass das Testen selbständig und ohne Beeinflussung von außen erfolgt. Ein Vorteil des Prototyping ist auch, dass die PrüferInnen experimentell mit dem Prototyp umgehen können und somit auch auf Abweichungen oder Fehler aufmerksam werden, die auf keiner Checkliste erfasst sind. Dafür ist natürlich wichtig, dass die prüfenden Personen über die Grenzen des Prototyps informiert sind, damit nicht Features geprüft werden, die (noch) gar nicht realisiert sind. Das Protokoll des Prüfers enthält die Ergebnisse der Erfahrung mit dem Prototyp, die z.B. anhand der Prüfszenarien und der Checkliste erstellt wurden.
Das Beobachtungsprotokoll sollte zusätzlich weitere Hinweise auf Anforderungen enthalten, die aus der Beobachtung des Prüfers entstanden ist. Zum Beispiel kann dort vermerkt werden, wenn eine Person zögert den Prototyp zu verwenden, das könnte beispielsweise bedeuten, dass die Erklärung nicht ausreichend ist. Je nach Sachlage könnte auch eine Videoaufzeichnung hilfreich sein, zum Beispiel, wenn die Usability eines Systems untersucht werden soll.
Zum Schluss werden die Ergebnisse ausgewertet und in Änderungsvorschläge für die Anforderungen übertragen.

Der Einsatz von Checklisten

Checklisten enthalten Fragen, die sich auf spezielle Anforderungen beziehen und im Rahmen des Prüfungsprozesses beantwortet werden sollen. Sie werden gemeinsam mit anderen Methoden eingesetzt und bedürfen selbstverständlicherweise einer gründlichen Vorbereitung. Nach Pohl und Rupp ist es zielführend, wenn die Erstellung von Checklisten auf folgende Quellen zurückgreift:

- Auf die drei Qualitätsaspekte Inhalt, Dokumentation und Abgestimmtheit
- Auf die Prinzipien der Prüfung von Anforderungen (Beteiligung der richtigen Stakeholder, Trennung von Fehlersuche und Fehlerkorrektur, Prüfung aus unterschiedlichen Sichten, geeigneter Wechsel der Dokumentationsform, Konstruktion von Entwicklungsartefakten, wiederholte Prüfung)
- Auf die Qualitätskriterien von Anforderungsdokumenten (Eindeutigkeit und Konsistenz, klare Struktur, Modifizierbarkeit und Erweiterbarkeit, Vollständigkeit, Verfolgbarkeit)

- Auf die Qualitätskriterien für die jeweiligen Anforderungen (Abgestimmtheit, Eindeutigkeit, Notwendigkeit, Konsistenz, Prüfbarkeit, Realisierbarkeit, Verfolgbarkeit, Vollständigkeit, Verständlichkeit)
- Auf die Erfahrung der Prüfer aus bisherigen Projekten
- Auf Fehlerstatistiken (siehe Pohl/Rupp 2015, 111)

Checklisten können auf unterschiedliche Arten eingesetzt werden, sie können der Prüfperson als Leitfaden dienen, an denen die Prüfung erfolgen kann, sie können aber auch als Vorgabe eingesetzt werden. Dann müssen die PrüferInnen sämtliche Fragen beantworten. Das ist zum Beispiel dann sinnvoll, wenn eine Vergleichbarkeit der Prüfungsergebnisse erzielt werden soll. Checklisten sollten nicht zu umfangreich sein und die Fragen sollten nicht zu offen formuliert werden um die Antworten vergleichen zu können.

Qualitätsmetriken

Die Qualität von Anforderungen und auch des Requirements-Engineering-Prozesses kann durch die Verwendung von Metriken gemessen werden. Daran lässt sich ein Indikator für die Produkt-, bzw. Prozessqualität ablesen. Beispielsweise lässt sich mit Prozessmetriken erkennen, wie viele Anforderungen, die schon abgestimmt waren in einem bestimmten Zeitrahmen geändert wurden. Produktmetriken dagegen liefern Zahlen darüber, wie viele Fehler beispielsweise in einem bestimmten Zeitrahmen identifiziert werden konnten. Diese Zahl lässt wiederum Rückschlüsse auf die Qualität des gesamten RE-Prozesses zu. (siehe auch Pohl/Rupp 2015, 146f)

2.14 Konflikte erkennen und lösen

Während des gesamten RE-Prozess kann es zu Konfliktsituationen kommen. RE kann dabei helfen diese Konflikte zu lösen. Werden sie nicht gelöst, drohen Akzeptanzprobleme, bzw. können nicht alle Anforderungen umgesetzt werden. Zunächst müssen die Konflikte identifiziert und analysiert werden. Nach der Konfliktauflösung muss eine Dokumentation der Lösung erfolgen. Mögliche Konflikttypen sind etwa:

- Sachkonflikte, wenn die Partner unterschiedliche Ansichten zu einem Thema haben, bzw. sich bzgl. des Ziels einig sind, nicht aber bei den Mitteln dieses zu erreichen.
- Benennungskonflikte entstehen durch unterschiedliches Verständnis über einen Begriff
- Interessenskonflikte, wenn die strategischen Ziele der Stakeholder nicht miteinander vereinbar sind
- Wertekonflikte, wenn unterschiedliche Wertesysteme der Stakeholder aufeinandertreffen.
- Beziehungskonflikte, bei persönlichen Differenzen, bzw. wenn hierarchische Strukturen nicht klar sind. Diese Konfliktart ist oft hinter Sachkonflikten versteckt
- Strukturkonflikte, zum Beispiel bei Macht- und Autoritätsverhältnissen.
- Rollenkonflikte, wenn die Rollen nicht klar genug definiert sind. (siehe Rupp/Sophisten 2014, 354ff und Pohl/Rupp 2015, 113f)

Oft sind diese Typen vermischt. Um einen Konflikt lösen zu können, muss man darin geübt sein und selbst neutral bleiben können. Bei länger andauernden Konflikten empfiehlt es sich, professionelles Konfliktmanagement einzubeziehen.

Die Identifikation von Konflikten erfolgt im Rahmen eines Konfliktmanagements

Laut Pohl und Rupp sind im Bereich des RE vier Aufgaben des Konfliktmanagements zu unterscheiden:

a) Konfliktidentifikation
b) Konfliktanalyse
c) Konfliktauflösung
d) Dokumentation der Konfliktauflösung (siehe Rupp/Sophisten 2014, 349ff und Pohl/Rupp 2015, 112ff)

a) Konfliktidentifikation

Konflikte treten zum Beispiel dann auf, wenn Stakeholder einander widersprechende Anforderungen formulieren. RE sollte während des gesamten Prozesses auf Konflikte achten und diese möglichst frühzeitig identifizieren. Nicht immer ist es offensichtlich, wo der Konflikt entstanden ist. Gründe für Konflikte können etwa sein, dass Begriffe unterschiedlich verwendet wurden, Stakeholder entgegengesetzte Meinungen vertreten, bzw. versuchen einander zu dominieren, eigene Aussagen werden „vergessen“, bzw. das Gesprächsklima wird aggressiv. Ein gut gepflegtes Glossar verhindert zumindest Unklarheiten bzgl. der Bedeutungen der verwendeten Begriffe. Kritische Szenarien können mit den Stakeholdern durchgegangen werden, auch so kann man Unklarheiten beseitigen. Widersprüche in den Anforderungen sollten so rasch wie möglich erkannt und aufgelöst werden.

b) Konfliktanalyse

Für den RE ist es wichtig, eine möglichst neutrale Haltung einzunehmen, den Stakeholdern genau zuzuhören und die sachlichen Argumente herauszufiltern. Wenn die emotionalen Aspekte überwiegen ist es sinnvoll einen Mediator hinzuzuziehen. Ein so genannter Konfliktsteckbrief hilft bei einem geordneten Vorgehen. Er beschreibt kurz den Konfliktgegenstand, die beteiligten Stakeholder sowie die einzelnen Konfliktpositionen und gibt an um welche Art von Konflikt es sich handelt. Außerdem sind darin mögliche Folgen und Risiken beinhaltet. Der Konfliktsteckbrief ist als eine Checkliste zu verstehen, die dabei hilft nichts Wesentliches zu vergessen (siehe Rupp/Sophisten 2014, 351f)

c) Konfliktauflösung

Der Konfliktauflösung kommt eine große Bedeutung zu, da die Art und Weise, wie diese durchgeführt ist und welchen Erfolg sie hat, Auswirkungen auf die Beziehungen zu den Stakeholdern hat. Wichtig ist, dass alle Stakeholder miteinbezogen werden. Als RE kann man beispielsweise versuchen, sämtliche Stakeholder bei einem Meeting bewusst zu machen, dass im Grunde genommen, alle das gleiche Ziel verfolgen.

Wichtig ist als RE neutral zu bleiben und bei schwerwiegenden Konflikten professionelle Unterstützung in Anspruch zu nehmen und so zur Deeskalation beizutragen.
Für das Gelingen eines Konfliktlösungsprozesses ist es sehr wichtig, dass die einzelnen Schritte gut nachvollziehbar dokumentiert werden. Sonst könnte es passieren, dass ein Konflikt immer wieder behandelt werden muss, was erstens zusätzliche Kosten verursacht und zweitens zu neuen Konflikten führen kann.
Hilfreich ist auch, wenn man den Eskalationsgrad eines Konflikts rasch erkennt. Ab einer bestimmten Eskalationsstufe kann man auf sachlicher Ebene kaum mehr gegensteuern.
Nicht unwesentlich ist es auch, sich als RE mit der so genannten „gewaltfreien Kommunikation“ nach Rosenberg[21] vertraut zu machen. Dabei ist das Ziel von einem aggressiven Konflikt zu einem offenen Konflikt zu gelangen. Auch Verhandlungstechniken können hier gewinnbringen eingebracht werden.
Je nach Zusammensetzung der Gruppe, nach der Anzahl der Stakeholder, nach der Kompliziertheit des Sachverhalts, den kommunikativen Fähigkeiten der Stakeholder, dem Zeitdruck und der Art des Konflikts können unterschiedliche Konsolidierungstechniken eingesetzt werden.
Sich mit Konfliktmanagement zu beschäftigen, ist für Bibliotheken sicher nicht nur im Bereich von RE hilfreich, sondern auch in anderen Zusammenhängen.

[21] Siehe etwa: Rosenberg, Marshall B.: Gewaltfreie Kommunikation: eine Sprache des Lebens. Paderborn: Junfermann 2011.

2.15 Change-Management – Verwaltung von Anforderungsänderungen

Eine Anforderung ist wie ein Chamäleon
(Rupp/Sophisten 2014, 35)

Change-Management bezeichnet den Prozess, der für die Steuerung des Lebenszyklus aller Änderungen verantwortlich ist. Je aktueller das System ist, je häufiger neue Systeme eingeführt werden und je mehr Kunden das System verwenden, desto mehr Änderungen sind zu erwarten. Anforderungen ändern sich im Verlauf eines Projekts, es werden Anforderungen gestrichen oder verändert oder es kommen neue hinzu. Diese Änderungen sind positiv zu sehen, denn sie hängen unter anderem auch mit dem Wissenszuwachs der Stakeholder während des Projekts zusammen. Außerdem können sich die Geschäftsprozesse eines Unternehmens ändern. Wird nach der agilen Methode gearbeitet, werden Stakeholder sogar ermuntert, Änderungen einzubringen, da der Stand der Anforderungen sich immer an den aktuellen Bedürfnissen der Stakeholder orientieren sollte. Wichtig ist, dass diese unterschiedlichen Versionen über den gesamten Zeitraum lückenlos unter Versionsnummern dokumentiert werden.

Die Versionierung von Anforderungen ist ein Teil des Konfigurationsmanagements. Wird das Datum und die Uhrzeit, die Änderungen und die Person mitgespeichert, wird der Prozess nachvollziehbar. Dabei ist es sinnvoll auch Versionsnummern zu vergeben. Bei großen Änderungen bezeichnet man die Versionen z.B. mit 1.0., 2.0. usw. bei kleineren Änderungen verwendet man so genannte Inkremente, z.B. 1.1., 1.2., 2.0., 2.1. usw. Die zusammengenommenen Anforderungen für den nächsten Entwicklungsschritt bezeichnet man als Baseline oder Anforderungsbasislinien. Die Baseline sollten stabile Versionen von Anforderungen enthalten und nach außen hin sichtbar sein, z.B. für den Auftraggeber. Für jeden Meilenstein im Projekt sollte es eine Baseline geben, vor allem vor dem Review der Anforderungen, für die Abschätzung des Realisierungsaufwandes, für den Vergleich mit Konkurrenzprodukten, nach der Einarbeitung der Anmerkungen aus dem Reviewprozess, wenn die Spezifikationen von den Stakeholdern akzeptiert und unterschrieben wurden, wenn die Anforderungen zum Testen oder zur Implementierung weitergegeben werden, bzw. für jene Anforderungen, die der Abnahme zugrunde

liegen. Parallelentwicklungen sind manchmal notwendig, erfordern jedoch eine besondere Aufmerksamkeit.

Jede der ausgewählten Anforderung muss genau in einer Version (durch eine Versionsnummer identifiziert) enthalten sein. Ist die Auslieferungsstufe eines Systems erreicht, spricht man von einem „Systemrelease“ oder „Anforderungsrelease“.
Change Management darf nicht nur am Anfang des Projekts stattfinden, sondern muss den gesamten Prozess durchlaufen. Vor allem in großen Projekten entwickeln und verändern sich Anforderungen im Laufe der Zeit.
Oft erhält man jedoch keine konkreten Änderungswünsche sondern vielmehr Informationen über Probleme oder Zwischenfälle. Ob es sich dabei wirklich um Wünsche nach Änderungen handelt, oder ob etwa Bedienfehler vorliegen oder die Probleme in einem anderen System angesiedelt sind, muss erst geprüft werden. Am Anfang werden sie als „Tickets“ behandelt, darunter versteht man Anfragen der Stakeholder, die an einen Helpdesk gerichtet sind. Es empfiehlt sich auch zu Beginn eines neuen Systems ein so genanntes „Incident –Management“ (Störungsmanagement) einzurichten. Mittels Kundendienst, auch häufig als Service Desk, User Help Desk, Hotline oder 1st-Level-Support bezeichnet werden Anfragen zur richtigen Bedienung des Systems behandelt. (siehe auch Rupp/Sophisten 2014, 448) Hier entstehen die Tickets, die eventuell Veränderbarkeitsbedarf anzeigen. Wenn es sich um ein neues System handelt, sollten nicht nur die EndanwenderInnen die Möglichkeit haben, es zu testen, sondern auch professionelle Tester. Auch diese Ergebnisse fließen in den Änderungsprozess ein.
Tickets können unter anderem folgende Inhalte haben:

- Bugs: Das System verhält sich nicht so, wie es in den Anforderungen beschrieben wurde.
- Defekt: Das System entspricht den dokumentierten Anforderungen, aber nicht den eigentlichen Anforderungen, hier müssen sowohl am System als auch in der Spezifikation Änderungen gemacht werden.
- Innovation: Hier handelt es sich um Wünsche von Stakeholdern, die neue Systemeigenschaften betreffen. Diese neuen Anforderungen müssen den gesamten, im RE-Leitfaden dokumentierten Prozess durchlaufen und werden evtl. erst in einem Folgeprojekt umgesetzt.

- Tuning-Tickets betreffen Änderungen, die aus der Anwendersicht nicht direkt ersichtlich sind, die aber das System stabilisieren oder die Performanz erhöhen (siehe auch Rupp/Sophisten 2014, 450)

Mit einem Ticket-Verwaltungstool kann man die Tickets verknüpfen, sortieren und filtern, jemandem zuweisen und den gesamten Lifecycle nachweisen.
Ein weiterer wesentlicher Schritt ist die Beschreibung der Tickets, die vom Verfasser meist nur aus einer Sicht und eher sparsam erstellt wurde, der Inhalt sollte verständlich und widerspruchsfrei sein. Im Rahmen des Change-Managements werden Lösungen grob skizziert und Aufwände abgeschätzt. Darauf folgt die Entscheidung, ob das Ticket umgesetzt wird.
„Die Erfahrung zeigt, dass fehlendes oder unzureichendes Anforderungsmanagement in der Regel schwer wiegende Konsequenzen nach sich zieht und die Projektkosten in die Höhe treibt“, meinen Heche und Ollech (Heche/Ollech 2006, 1)
Ein Änderungsanforderungsantrag (Change Request) kann den Source Code, die Anforderungen, die Architektur, oder das Design betreffen. Zuerst ist zu prüfen, ob überhaupt eine Anforderung betroffen ist, wenn ja, muss die Auswirkung auf das Gesamtprojekt ermittelt werden. Wenn die Änderungen akzeptiert werden, müssen erneut die Stakeholder miteinbezogen werden und die Änderung muss freigegeben werden.
Heche und Ollech sprechen im Idealfall davon, dass das Anforderungsmanagement wie ein „Leibwächter“ fungieren soll. (Heche/Ollech 2006, 2) Die Anforderungen müssen genau geprüft werden, was oft viel Zeit kostet und fachliche Kompetenz erfordert.

Wichtig ist, dass die Schritte überwacht werden. Dies kann zwar die Projektleitung übernehmen, ist das Projekt jedoch schon zu umfangreich, sollte diese Funktion eine andere Person oder eventuell ein CCB übernehmen (Heche/Ollech, 3 und Pohl/Rupp 2015, 141) Das CCB hat die Aufgabe, Änderungsanträge zu kanalisieren, den erforderlichen Aufwand bestimmen, oder jemanden damit beauftragen, über Annahme oder Ablehnung, bzw. Zurückstellung von Anforderungsänderungen zu entscheiden, es soll die Anforderungsänderungen definieren, die Priorität für die Änderung festsetzen und auch festlegen, mit welchem Release die Änderung erfolgen soll. Außerdem muss

das CCB die Entscheidung über den Änderungsantrag kommunizieren. Mitglieder des CCB können Auftraggeber, bzw. Geldgeber sein, Änderungsmanager, VertreterInnen der späteren NutzerInnen, Qualitätsbeauftragte, RE und ProjektleiterInnen.

Ein Änderungsantrag sollte im Idealfall folgende Komponenten beinhalten:

- Eine eindeutige Identifikationsnummer
- Einen eindeutigen Titel
- Das Datum
- Den Namen der AntragstellerInnen
- Eine Beschreibung
- Eine Begründung
- Die Priorität aus der Sicht des Antragstellers
- Eine Person, die nach der Umsetzung des Änderungsvertrages überprüft, ob die Änderung richtig umgesetzt wurde
- Eine Angabe, ob die Auswirkungen des Änderungsantrags schon analysiert wurde
- Die Angabe, ob über den Änderungsantrag schon entschieden wurde
- Die durch das CCB festgelegte Priorität
- Eine Person, die für die Umsetzung des Änderungsantrages verantwortlich ist
- Eine Angabe, in welchem Systemrelease der Änderungsantrag umgesetzt werden soll. (siehe Pohl/Rupp 2015, 143)

Das CCB, bzw. der dem Board vorsitzenden Änderungsmanager hat die Aufgabe, den Änderungsantrag zu klassifizieren, wenn notwendig anzupassen. Der Änderungsantrag wird in einem weiteren Schritt bei der nächsten Sitzung vorgestellt und genehmigt. Die/der ÄnderungsmanagerIn muss, wenn nötig, bei Unstimmigkeiten vermitteln und zwischen den Entscheidungen aller Beteiligten vermitteln. Außerdem hat sie oder er die Aufgabe Entscheidungen zu kommunizieren und zu dokumentieren.

Es werden oft drei Arten von Änderungen unterschieden:

- Korrektive Änderungen: Die Anpassung ist nötig, weil ein Fehler in den Anforderungen aufgetreten ist, der zu einem Fehlverhalten des Systems führen würde. Diese Änderungen müssen möglichst zeitnah durchgeführt werden.
- Adaptive Änderungen: Die Änderung ist nötig, wenn eine Anpassung des Systems erforderlich ist, z.B. bei geänderten Rahmenbedingungen oder wenn

eine neue Technologie verfügbar ist. Adaptive Änderungen werden oft gebündelt und zu einem späteren Zeitpunkt analysiert, bewertet und bei einem der nächsten Systemrelease umgesetzt.

- Ausnahmeänderungen: diese Änderungen werden auch „hotfix“ genannt und haben höchste Dringlichkeit. Sie können sowohl korrektiv als auch adaptiv sein. (siehe auch Pohl/Rupp 2015, 144)

Im Idealfall geht man bei korrektiven und adaptiven Änderungen schrittweise vor:

1. Auswirkungsanalyse: Dabei wird der Aufwand für die Durchführung der beantragten Änderungen ermittelt. Dabei werden alle Anforderungen ermittelt, die von der Änderung betroffen sind und die neu definierten Anforderungen ebenfalls berücksichtigt. Die nachfolgenden Entwicklungsratefakte werden identifiziert und müssen evtl. geändert oder neu entwickelt werden. Für jedes betroffene Artefakt muss der Aufwand für die Umsetzung der Änderung ermittelt werden und anschließend der Gesamtaufwand für alle beantragten Änderungen berechnet werden. Oft haben auch nur geringfügige Änderungen große Auswirkungen auf die nachgelagerten Entwicklungsartefakte. Um hier möglichst effektiv arbeiten zu können, ist es nötig, dass die Verfolgbarkeitsinformationen zur Verfügung stehen.
2. Anschließend an die Auswirkungsanalyse muss das CCB eine Beurteilung der beantragten Änderungen vornehmen. Dabei wird Kosten und Nutzen gegenüber gestellt und hinsichtlich der zur Verfügung stehenden Ressourcen bewertet. Der Nutzen kann in einem vermiedenen Prestigeverlust liegen, aber auch die Marktstellung verbessern oder sonst drohende Strafen vermeiden.
3. Werden die Anforderungsanträge genehmigt, wird durch das CCB eine Priorisierung vorgenommen und
4. zur Umsetzung einem der nächsten Release zugeordnet. (siehe auch Pohl/Rupp 2015, 145)

Entscheidungen über Anforderungsänderungen müssen allen Beteiligten kommuniziert werden.

Bei unkontrollierten Change-Requests spricht man auch von „requirements creeping“[22], also schleichenden Veränderungen, die die Projektziele verändern ohne die Stakeholder miteinzubeziehen. Bei einem Change-Request, der erstellt wurde, weil es eine Abweichung zwischen den Anforderungen und den Testergebnissen gibt, spricht man von einem „Bug-Report“.

[22] Siehe https://www.projektmagazin.de/glossarterm/schleichender-funktionszuwachs (abgerufen am 19.2.2017)

Der relativ hohe Aufwand, der damit verbunden ist, verhindert, dass es zu Verzögerungen und damit zu erhöhten Kosten bei der Umsetzung kommt.

2.16 Das Release

Testing a program is like walking on water,
it only works if it's frozen
Rupp/Sophisten 2014, 455

Das Release-Management bezeichnet den Prozess der sowohl für die Planung als auch für die Steuerung der Tests und die Verteilung der Releases zuständig ist. Vor allem die Einhaltung der festgesetzten Termine muss überwacht werden.
Um ein System releasen zu können, ist es wichtig, dass sämtliche Informationen zu einem bestimmten Zeitpunkt „eingefroren" werden. Man erhält damit eine Konfiguration. Erst danach kann man mit der Testphase beginnen. Wenn eine Konfiguration alle Anforderungen für ein Release enthält, spricht man von einer „Basislinie" oder „Baseline". (siehe Pohl/Rupp 2014, 455) Mit dieser Basislinie und einem Änderungsprotokoll kann man zu den nächsten Prozessschritten gehen.
Mit dem Release sollte man auch sogenannte „Release Notes", eine Auflistung aller in einem Release vorgenommenen Änderungen mitliefern.
Beim Auftreten kritischer Fehler kann auch ein Emergency Release sinnvoll sein, bei dem man sich wirklich nur um den aufgetretenen Fehler kümmert. Vor allem hier ist es wichtig, sehr strukturiert vorzugehen und alles gut zu dokumentieren, da sonst noch mehr Fehler entstehen. (siehe auch Rupp/Sophisten 2014, 453ff)

2.17 Werkzeuge für das Anforderungsmanagement

Wie aus den vorangegangenen Erläuterungen hervorgeht, handelt es sich beim Managen von Anforderungen um einen komplexen Prozess, der mehrere Arten an Werkzeugen benötigt, da er aus unterschiedlichen Aufgaben besteht. Dazu zählt das Verwalten von logischen Beziehungen, das eindeutige Identifizieren von jedem Artefakt, die Steuerung des Zugriffs, das Sichten und Organisieren, das Erstellen von Berichten und das Generieren von Dokumenten. (siehe auch Pohl/Rupp 2015, 151ff)

Bevor man ein neues Werkzeug einführt, empfiehlt es sich die Ressourcen der Einführung zu planen, mit Pilotprojekten die Risiken verringern, das neue Werkzeug aus unterschiedlichen Sichten beurteilen. Weiters sind nicht nur die Lizenzkosten zu berücksichtigen, sondern auch der Schulungsaufwand, die Integration in weitere Werkzeuge, der Aufwand in Anpassungen und für BenutzerInnenschulungen.

Bei den Werkzeugen lassen sich mehrere Kategorien unterscheiden (siehe auch Pohl/Rupp 2015, 152ff):

1. Standard-Bürowerkzeuge
MS-Word für die Textverarbeitung ist z.B. weit verbreitet und gut akzeptiert, das ID-Management, bzw. die Traceability ist jedoch schwer zu realisieren. Auch MS-Excel ist bekannt, Änderungsverfolgungen lassen sich damit jedoch kaum umsetzen, außerdem ist ein gemeinsames Arbeiten nicht möglich. Datenbanken, wie beispielsweise Access oder Oracle bieten zwar viele Möglichkeiten, bedeuten jedoch auch einen relativ hohen Programmieraufwand.

2. Anforderungsmanagementwerkzeuge
Sie sind meist proprietär, viele davon werden von IBM angeboten. Eine weitere Möglichkeit wäre die Entwicklung einer speziellen Datenbank.

3. Modellierungswerkzeuge
Für die Umsetzung von UML werden sowohl proprietäre Produkte als auch Open-Source-Produkte angeboten. Als Zeichenwerkzeug bietet sich z.B. Microsoft Visio an. Wichtig ist auch hier, dass jedes Modellelement eine eigene und eindeutige ID erhält.

4. Fehlermanagementwerkzeuge
Auch hier gibt es proprietäre Anwendungen, aber auch Open-Source-Produkte, wie z.B. Bugzilla[23], ein freies, webbasiertes Werkzeug zur Verwaltung von Fehlermeldungen.

5. Kollaborationswerkzeuge
Neben zahlreichen kostenpflichtigen Firmenprodukten, bietet sich hier ein Wiki oder Source Forge[24] an. Bei einer großen Anzahl von Stakeholdern könnte man auch ein Wiki einsetzen.

[23] https://www.bugzilla.org/ (abgerufen am 19.2.2017)
[24] Dabei handelt es sich um eine Open Source Community, siehe https://sourceforge.net/about (abgerufen am 19.2.2017)

6. Simulationswerkzeuge ermöglichen das neue System zu simulieren um das dynamische Verhalten überprüfen zu können

Allgemein lassen sich einige der Werkzeuge sowohl für die Systementwicklung als auch für das RE nutzen. Bei der Nutzung dieser Tools hat man den Vorteil, dass man die Handhabung des Werkzeuges nicht neu lernen muss und dass die einzelnen Prozesse gemeinsam verwaltet werden können. Mailsysteme, Chatsoftware, Terminplaner, Visualisierungstechniken können beispielsweise für die Projektplanung, das Controlling, die Kommunikation, die Planung und Verteilung von Aufgaben usw. eingesetzt werden.
Bei der Verwendung von Werkzeugen ist darauf zu achten, dass sie miteinander vernetzt sind und verfolgbar gehalten werden, bzw. dass es Schnittstellen zwischen den Werkzeugen gibt. Bei Änderungen muss etwa sichergestellt sein, dass dies auch an den betroffenen Modellelementen rasch durchgeführt werden können.

Werkzeuge, die für das RE[25] eingesetzt werden können, unterstützen vor allem:

- die Verwaltung von Anforderungen und Attributen auf der Basis von Informationsmodellen
- die Organisation von Anforderungen (mittels Hierarchieebenen)
- die Konfigurations- und das Versionsmanagement auf Anforderungsebene
- die Definition von Anforderungsbasislinien (Baselining)
- den Mehrbenutzerzugriff und –verwaltung (z.B. Zugriffskontrolle)
- das Verfolgbarkeitsmanagement (Traceability Management)
- die Konsolidierung der erfassten Anforderungen (z.B. Sichtenbildung)
- die Unterstützung des Änderungsmanagements (Änderungskontrolle) (Pohl/Rupp 2015, 152)

Anhand klar definierter Muss-Kriterien sollten die zur Auswahl stehenden Werkzeuge vorselektiert werden. Auch die zu erwartenden Kosten, die nicht nur Anschaffungs- oder Lizenzkosten, sondern auch Schulungskosten und eventuelle Anpassungskosten umfassen, spielen bei der Wahl eine große Rolle.

[25] Werkzeuge findet man u.a. unter http://www.incose.org/ und http://www.volere.co.uk/ (abgerufen am 19.2.2017)

Um die Werkzeuge möglichst objektiv auswählen zu können schlagen Pohl und Rupp vor, die Tools aus unterschiedlichen Blickwinkeln zu betrachten und aus diesen Sichtweisen heraus Kriterien zu entwickeln.

- Projektsicht: Wie kann das Werkzeug den Projektverlauf unterstützen?
- Benutzersicht: Wie sieht beispielsweise die BenutzerInnenverwaltung aus?
- Produktsicht: Wie können die Aktivitäten protokolliert werden?
- Anbietersicht: Welche Dienstleistungsmöglichkeiten sind vorhanden?
- Technische Sicht: Wie sieht es etwa mit verfügbaren Schnittstellen aus?
- Betriebswirtschaftliche Sicht: Welche Kosten fallen für die Einführung des Werkzeugs an, für die Beschaffung und die Wartung? (siehe Pohl/Rupp 2015, 155ff)

Bei der Wahl der Werkzeuge empfiehlt es sich für wissenschaftliche Bibliotheken eng mit den KollegInnen aus dem IT-Bereich zu kooperieren, damit die spätere Zusammenarbeit reibungslos funktioniert.

3. Requirements-Engineering an Bibliotheken

„In den letzten 15 Jahren hat sich das Aufgaben- und Serviceportfolio von Universitätsbibliotheken aufgrund der rasanten technischen Entwicklung und den damit verbundenen NutzerInnenanforderungen grundlegend geändert […] Die eigentliche Herausforderung für die nächsten Jahre wird sein, genau die richtigen Services zu entwickeln, die das Arbeiten, Forschen, Lehren und Studieren im digitalen Zeitalter optimal unterstützen, die Zukunft voranzutreiben, ohne die schriftliche Überlieferung und die Erhaltung des Kulturgutes zu vernachlässigen“ ,

meint die Leiterin der Universitätsbibliothek Wien Maria Seissl. (Seissl 2015, 73f)

Diese Forderungen entsprechen auch den Grundsätzen der Gründer von „Design Thinking“:

„Anwender, Verbraucher und Nutzerorientierung gehören ins Zentrum aller Aktivitäten von Wirtschaft und Staat. Es ist falsch, dass viele Unternehmen den Kunden jeweils nur als „End-User“ betrachten und nicht als Quelle für neue Ideen (Plattner, Meinel, Weinberg 2009, 56)

Dieses „nutzerorientierte, teambasierte Erfinden und Entwickeln“ (Plattner, Meinel, Weinberg 2009, 7) könnte auch für das gemeinsame Erstellen von Anforderungen in Bibliotheken eingesetzt werden, denn hier werden größere Zusammenhänge mitgedacht, multidisziplinäre Teams miteinbezogen und die unterschiedlichen Fähigkeiten der Teammitglieder berücksichtigt. Wie Personen, die im Bereich RE tätig sind, wird auch von einem Design Thinker Empathie, integratives Denken, Experimentierfreude, die Fähigkeit zur Zusammenarbeit und Optimismus verlangt. Auch hier werden verschiedene Kreativitätstechniken angewendet, der Prozess läuft in sechs Schritten ab:

1. Verstehen
2. Beobachten
3. Standpunkt definieren
4. Ideen finden
5. Prototypen entwickeln
6. Testen (Plattner, Meinel, Weinberg 2009, 113)

Die Schritte sind iterativ miteinander verbunden, es wird durch Versuch und Irrtum gelernt.

Vor allem bei wissenschaftlichen Bibliotheken wird die Zusammenarbeit mit dem IT- und dem Designbereich immer wichtiger. Auch die BenutzerInnen von Bibliotheken

erwarten eine optimale Aufbereitung des Serviceangebots, das von Recherchetools über Datenbanken mit einem möglichst einfachen Zugang zu Volltexten bis zu Repositorien mit entsprechend aufbereiteten Publikationen und Forschungsdaten reicht. Inhalte möglichst rasch und barrierefrei zur Verfügung zu stellen ist immer noch die Hauptaufgabe von Bibliotheken. Durch den raschen technischen Fortschritt, neue Tools und Technologien ist es erstens wichtig immer am Laufenden zu bleiben und zweitens dieses Wissen auch technisch umsetzen zu können. IT-Lösungen müssen ständig angepasst und optimiert werden, was bedeutet, dass anhand der Qualität der Anforderungen an Systeme und der reibungslose Prozess der Umsetzung heute die Qualität und die Innovativität von Bibliotheken gemessen wird. Wissenschaftliche Bibliotheken stehen in einigen Bereichen heute in starker Konkurrenz zu diversen Internetanbietern, vor allem bei der Dienstleistung für die Naturwissenschaften, die heute kaum noch gedruckte Bücher oder Zeitschriften verwenden oder darin publizieren. Forschungsunterstützende Services funktionieren jedoch nur dann gut, wenn die richtigen Anforderungen an diverse Systeme gestellt wurden und diese auch professionell umgesetzt werden können. RE könnte hier in Zukunft eine große Rolle spielen, denn bibliothekarisches und technisches Wissen könnte hier gebündelt werden um in der Lage zu sein effizient und effektiv auf die heutigen und zukünftigen Herausforderungen zu reagieren. Wissenschaftliche Bibliotheken haben vor allem die Aufgabe Forschende und Lehrende optimal zu unterstützen. Bei der Frage was wirklich in Forschung und Lehre benötigt wird, kann der hier in der Theorie gezeigte Prozess des RE Anforderungen von der Entstehung bis zur Umsetzung optimal begleiten - ohne Irr- und Umwege. Auf den ersten Blick scheint RE sehr aufwändig und teuer zu sein, bedenkt man aber wie viel Geld oft für Notlösungen, mit denen niemand so richtig zufrieden ist, ausgegeben wird, ist professionelles RE immer noch günstiger. Außerdem verbessert ein gut funktionierendes Projekt das Arbeitsklima, was sich wiederum positiv auf die Arbeitsabläufe auswirkt.

3.1 Die Aufgaben von Requirements-Engineers in Bibliotheken

Das, was Sie als Requirements-Engineer erstellen
ist […] ein Modell der Realität,
das den Lösungsmodellraum aufspannt,
indem sich später der Architekt austoben kann.

(Rupp/Sophisten 2014, 24)

Wie oben detailliert gezeigt werden konnte, sind die Aufgabenbereiche von RE und RM sehr vielfältig. In der Wirtschaft werden bereits Personen in dieser Funktion eingesetzt. In Bibliotheken steht diese Entwicklung mehr oder weniger noch aus. Doch auch wenn man die Methoden des RE erlernt hat,

„[…] das bloße Anwenden entsprechender Methoden reicht alleine nicht aus. Eine gute Kommunikation, das totale Verständnis des Sachverhalts und eine vermittelnde Position zwischen allen Beteiligten – auch das muss ein Anforderungsmanager mitbringen, um wirklich alle relevanten Informationen zu bekommen. (Duwe/Mégroz, 2009, 1)

Der verantwortungsvolle Umgang mit Anforderungen ist eine große Herausforderung. So meinen Pohl und Rupp:

„ein Anforderungsmanagement „nach Lehrbuch“ erfüllt nicht alle Bedingungen für einen reibungslosen Ablauf. Es muss nicht nur genügend Zeit eingeplant werden, sondern berücksichtigt werden, dass „der Requirements Engineer eine Rolle mit anspruchsvollen Tätigkeiten ist“. (Pohl/Rupp 2015, 1)

Die Aufgaben eines Anforderungsmanagers können unterschiedlich beschrieben werden:

„Bildlich gesprochen ist der Anforderungsmanager ein „Wanderer zwischen den Welten“ der IT-Seite und der fachlichen Nutzerseite. Seine Aufgabe ist es, die „Wissensschätze“ auf der Nutzerseite zu (er-) heben und sie so aufzubereiten, dass die Anwenderinnen sie später in der fertigen Anwendung wiederfinden. Dabei ist es nützlich, sich für die Rolle nicht als Vertreter einer der beiden Seiten zu positionieren, sondern eher die „Brückenfunktion“ hervorzuheben.“ (Duwe/Mégroz, 2009, 3)

Als „Mittler zwischen den Welten“ (Rupp/Sophisten 2014, 11) verbindet er/sie mehrere Bereiche innerhalb eines Projekts miteinander, verfügt über zahleiche Kontakte zu EntwicklerInnen, SystemarchitektInnen, dem Testteam, den späteren AnwenderInnen und den Stakeholdern und hat daher auch großen fachlichen Einfluss auf die

Systementwicklung. Daher kann der RE auch als „Katalysator“ (Rupp/Sophisten 2014, 11) für die Entscheidungen der Stakeholder bezeichnet werden.

All diese Eigenschaften werden auch an wissenschaftlichen Bibliotheken benötigt. Beim Aufbau oder beim Ankauf eines Systems, beispielsweise eines digitalen Repositoriums, sind die Bedürfnisse von zahlreichen unterschiedlichen Personen in verschiedenen Positionen zu berücksichtigen. WissenschafterInnen aus mehreren Fachdisziplinen, Verwaltungsangestellte, der Betriebsrat, Datenschutzbeauftragte, eventuell Studierende, die ihre Abschlussarbeit dort hochladen sollen, usw. müssen in den Prozess miteinbezogen werden.

Die Entscheidung, eine/n AnforderungsmanagerIn zu bestimmen, bringt viele Vorteile mit sich:

- Alle Aufgaben des Anforderungsmanagements sind bei einer Person gebündelt, die die Zusammenhänge und Wechselwirkungen im Projekt überblickt. Die anderen Projektmitglieder werden damit entlastet.
- Die diversen Stakeholder-Interessen werden proaktiv abgefragt und tief greifend analysiert.
- Änderungsanforderungen werden vor einer Entscheidung auf Auswirkungen auf das Projekt hin geprüft und aufgrund dieser Erkenntnisse eine solide und bedachte Entscheidung getroffen.
- Das Einführen eines Anforderungsmanagements ändert nichts an den sonstigen Abläufen. (siehe auch Heche/Ollech 2006, 4)

Die rasche technische Entwicklung und damit einhergehend das unterschiedliche NutzerInnenverhalten – so werden für viele Bibliotheksdienste Smartphones oder Tablets verwendet – erfordert eine ständige Anpassung an technische Neuerungen, was sowohl bei den BibliothekarInnen als auch bei den EntwicklerInnen bedeutet sich selbst mit den aktuellen Funktionalitäten diverser Endgeräte vertraut machen zu müssen. So können BibliothekarInnen im Auskunftsdienst nur dann adäquat beraten, wenn sie selbst mit den von den LeserInnen benutzten Geräten und Apps vertraut sind. Das hat auch zur Folge, dass die Enduser beispielsweise erwarten, dass sämtliche Funktionen von Datenbanken, Suchmaschinen, Katalogen oder Repositorien auch über mobile

Endgeräte gut funktionieren. Herauszufinden, wie dieses „gut funktionieren“ im Detail aussieht, ist Aufgabe des RE. Der ständige Kontakt zu den Endusern ermöglicht erst das Erkennen neuer Bedürfnisse. Diese müssen in eine optimale Form gebracht werden und als perfekte Anforderungen an die EntwicklerInnen adäquat weitergegeben werden. Lösungsmöglichkeiten müssen wieder zurück kommuniziert werden können. Vor allem in wissenschaftlichen Bibliotheken, wo Stakeholder oft hochspezialisierte WissenschafterInnen, die in zahlreichen Projekten involviert sind und kaum Zeit haben um sich um technische Belange ausreichend zu kümmern, ist es wichtig, Bedürfnisse rasch zu erkennen und zu übersetzen.
Auch beim Umgang mit Forschungsdaten, einem relativ neuen Aufgabengebiet von wissenschaftlichen Bibliotheken, wo auch die Bedürfnisse auf Seiten der Wissenschaft noch nicht ganz klar sind, ist es oft schwierig, rasch die richtigen Lösungen anzubieten. Es sind viele Gespräche nötig um die Anforderungen so zu erhalten, dass sie sinnvoll umgesetzt werden können.
Dieser Prozess funktioniert jedoch nur dann wirklich gut, wenn auch die Umgebung passt und diesen Workflow mitträgt.

3.2 Anforderungsprofil an Requirements-Engineers in Bibliotheken

Beispiele aus der Praxis zeigen, dass in der Wirtschaft eine starke Nachfrage an RE herrscht. Mit Bibliotheken wird das Thema heute kaum noch in Verbindung gebracht. Ein Vortrag im Rahmen des österreichischen Bibliothekartages in Wien 2015 beschäftigte sich mit dem Thema „Anforderungsanalysen bei Bibliotheksprojekten. Aus der Sicht Bibliothek und aus Sichte der IT“[26]. Die Reaktionen darauf zeigten einerseits das Interesse am Thema, andererseits aber auch, dass RE in der Praxis kaum Anwendung findet.

Eigenschaften, die RE an Bibliotheken mitbringen sollten wären:

[26] Blumesberger, Susanne; Raman Ganguly: Requirements-Management aus technischer und nichttechnischer Sicht. Vortragsfolien im Rahmen des 32. Österreichischen Bibliothekartages in Wien 2015. urn:nbn:de:0290-opus4-21005

- Er/sie muss bibliothekarisches Wissen besitzen und sich im IT-Bereich so weit auskennen, dass er/sie eventuelle Probleme nachvollziehen kann und in der Lage ist, mit den unterschiedlichen Stakeholdern entsprechend kommunizieren zu können.

- Er/sie muss gut im analytischen Denken sein, Sachverhalte rasch erfassen, auch komplizierte Zusammenhänge schnell erkennen und analysieren können. Da Stakeholder aufgrund ihrer oft mangelhaften IT-Kenntnisse Anforderungen oft nicht klar genug formulieren können, muss er/sie konkrete Aussagen abstrahieren können.

- Er/sie muss hohe kommunikative Fähigkeiten mitbringen, gut zuhören und zur richtigen Zeit die passenden Fragen stellen können. Außerdem ist es wichtig zu erkennen, ob Aussagen ausreichend Information enthalten und ob eventuell Rückfragen erforderlich sind. So werden unvollständige Anforderungen vermieden.

- Er/sie muss sprachliches Feingefühl mitbringen und sich bewusst sein, dass Personen, die schon lange in einem Bereich arbeiten, bei der Formulierung von Anforderungen möglicherweise Dinge voraussetzen und diese gar nicht mehr formulieren, weil sie sie als selbstverständlich erachten. Dabei besteht die Gefahr, dass gewisse Anforderungen, wie zum Beispiel, dass ein gewünschtes Feld Zahlen verarbeiten können muss, nur gedacht, aber nicht ausgesprochen, bzw. klar formuliert wird.

- Er/sie muss auch emotionale Intelligenz mitbringen, denn die Arbeitsbeziehung zwischen allen Beteiligten sollte so positiv wie möglich gestaltet sein. RE dürfen sich nicht nur auf den technischen Aspekt ihrer Rolle konzentrieren, sondern auch den prozessorientierten Aspekt nicht außer Acht lassen.

- Der oder die RE muss Konflikte möglichst rasch erkennen und sie auflösen können. Dabei ist es oft erforderlich zwischen Meinungen zu vermitteln und Diskussionen zu leiten. Moderationsfähigkeit ist also eine wichtige Eigenschaft für RE.

- Durch diese vermittelnden Aufgaben stehen RE oft im Mittelpunkt von Auseinandersetzungen. Es ist wichtig, in dieser Rolle genügend Kompetenz und Selbstbewusstsein mitzubringen und die Kritik niemals persönlich zu nehmen.

- Schließlich fällt den RE auch die Aufgabe zu, die Anforderungen der Stakeholder nach außen bei Besprechungen zu vertreten und andere von der Relevanz der Anforderungen zu vertreten.

- Nicht zuletzt muss auch das Umfeld so gestaltet sein, dass RE sich darin einfügen kann. Eine Person alleine kann hier nicht erfolgreich agieren.

Ein Beispiel für den Bibliotheksbereich könnte folgendermaßen aussehen:

Requirements-Engineer (m/w) für Bibliothek gesucht

Als professionelle und innovative Dienstleistungseinrichtung unterstützen wir die Ziele einer großen Universität. Wir versorgen die Forschenden, Lehrenden und Studierenden bedarfsorientiert mit Information und stellen für sie Lern- und Kommunikationsorte bereit. Darüber hinaus stehen wir unseren Nutzerinnen und Nutzern mit individueller Beratung zur Seite. Wir bieten Lösungen für das digitale Publizieren, das Management und die Langzeitarchivierung von Forschungsdaten sowie für die Sichtbarkeit von Forschungsergebnissen an. Sämtliche Services werden stets den Anforderungen unserer Nutzerinnen und Nutzern angepasst. Neue Dienste und Services, die den Lehr- und Forschungsbetrieb optimal unterstützen und den Forschungsoutput möglichst gut sichtbar machen, werden derzeit entwickelt.

Ihre Aufgaben:

- Sie machen sich ein deutliches Bild von den Anforderungen unserer User, die vor unterschiedlichen wissenschaftlichen Bereichen tätig sind und bereiten so die Grundlage von neuen Entwicklungen im Bibliotheksbereich vor.

- Auf technischer Ebene übernehmen Sie die Rolle einer direkten Schnittstelle zu Forscherinnen und Forschern und überlegen gemeinsam mit ihnen wie ihre Anforderungen bestmöglichst formuliert werden können.
- Sie übersetzen diese Anforderungen in klar formulierte Requirements und priorisieren diese.
- Sie führen den gesamten Verwaltungsprozess der Requirements durch
- Sie erarbeiten Vorschläge für Lösungen im Bibliotheksbetrieb gemeinsam mit Usern und der IT-Abteilung
- Sie übernehmen qualitätssichernde Aufgaben während des gesamten Prozesses bis zum Release.
- Sie führen das Projektmanagement durch.

Sie bringen mit:

- Eine abgeschlossene bibliothekarische Ausbildung (ULG Library and Information Studies) oder eine gleichwertige bibliothekarische Ausbildung
- Erfahrung in der Softwareentwicklung, eine technische Ausbildung ist von Vorteil
- Eine langjährige Erfahrung im Projektmanagement
- Erfahrung im Umgang mit WissenschafterInnen
- Kommunikative und soziale Kompetenz
- Teambildungskompetenz
- Moderationsfähigkeit
- Sicheres Auftreten
- Eigenverantwortung
- Die Bereitschaft sich in andere Fachgebiete einzuarbeiten

3.3 Einführen des Requirements-Engineerings in Institutionen

Will man in der eigenen Institution RE einführen, muss man sich auch auf Widerstände gefasst machen. Neues wirkt oft bedrohlich, der Einzelne kann nicht sicher sein, ob das Neue nicht auch Auswirkungen auf seine persönlichen Bedürfnisse hat. Dazu zählen der eigene Status, gewisse Vorteile, wie zum Beispiel auch Machtpositionen. Aber auch die Befürchtung etwas Neues erlernen zu müssen und dabei zu scheitern, sollte man mitberücksichtigen. Vor allem, wenn in der eigenen Institution Fehler zu machen gleichbedeutend mit Versagen ist, kann dies zu starken Widerständen gegen sämtliche Neuerungen führen. So schreibt Rupp:

„Angst vor Veränderung entsteht vor allem dann, wenn etwas dem Menschen Vertrautes von anderen verändert wird. Wenn er allerdings Veränderungen selbst mitgestalten kann, wird das weniger der Fall sein.“ (Rupp/Sophisten 2014, 478)

Mit anderen Worten heißt das, dass man möglichst viele Personen in diesen Einführungsprozess miteinbeziehen muss. Dennoch werden sich einige überrumpelt oder bevormundet fühlen und deshalb den Veränderungen eher mit Trotz begegnen. Skepsis ist jedoch nicht grundsätzlich negativ zu sehen, sie bedeutet auch eine gewisse Stabilität, denn zu viele Änderungen innerhalb einer kurzen Zeitspanne bedeuten auch eine gewisse Gefahr. Wenn MitarbeiterInnen sachlich und konstruktiv Kritik üben, sollte man diese als wertvolle Hinweise werten und durchaus ernst nehmen. Widerstand ist jedoch nicht immer einfach zu erkennen. Wichtig ist, dass derjenige, der die Neuerungen einführt auch die Macht erhält, diese durchzusetzen, sonst könnten die Widerstände zu stark werden.
Im sogenannten Satir-Modell[27] wird beschrieben, welche Phasen die Neueinführung eines Systems durchläuft:

1. Befindet sich beispielsweise ein Unternehmen im Status Quo, existiert ein sich wiederholendes Muster von Erwartungen und Reaktionen, die Vorgänge sind vertraut und berechenbar.
2. Wird ein neuer Prozess oder ein neues Werkzeug eingeführt, müssen sich alle auf dieses Neue einstellen und es muss mehrheitlich akzeptiert werden.
3. Die dritte Phase wird auch als Chaos bezeichnet, denn man verlässt den sicheren Bereich und bewegt sich auf etwas Unbekanntes zu, Stabilität und Sicherheit gehen dabei verloren. Zugleich besteht die Möglichkeit, das Neue zu erforschen und es als neue Lösung für bestehende Probleme zu entdecken. Die Ängste der Beteiligten müssen in dieser Phase ernst genommen werden, sie benötigen Unterstützung, um sich auf die neue Situation einstellen zu können.
4. Darauf folgt die Phase der Integration. Die Zweifel und Ängste sind beseitigt, das Neue wurde angenommen und nach und nach in den Alltag integriert.
5. In einer weiteren Phase muss das Neue durch Übungen fest im Alltag verankert werden, damit es wieder vertraut werden kann und Sicherheit bietet.
6. In der letzten Phase hat das Neue einen Zustand erreicht, wo es einen neuen Status quo einnimmt. (siehe auch Rupp/Sophisten 2014, 479f)

Die Einführung des RE kann als Projekt gesehen werden – mit allen Stationen, die auch andere Projekte durchlaufen. Auf der strategischen Ebene sollte zunächst der Ist-Zustand ermittelt werden, d.h. wie sieht es in der Unternehmensstrategie in Bezug auf

[27] Satir, Virginia: Das Satir-Modell: Familie und ihre Erweiterung. 4. Aufl. Paderborn: Junfermann 2007.

RE aus? Welche Ziele erfolgt die Einführung, welche Rahmenbedingungen müssen beachtet werden. Auf der methodischen Ebene sind die erforderlichen Werkzeuge zu überlegen und die Art und Weise, wie die Ergebnisse messbar sein werden. Auf der menschlich sozialen Ebene soll überprüft werden, wo im Management oder bei den MitarbeiterInnen Probleme bei dem derzeitigen Ist-Zustand gesehen werden. Wie sieht es mit neuen Stellen oder Umbesetzungen während der Einführung aus? Ein weiterer wichtiger Punkt ist die Überprüfung der benötigten Kompetenzen und das Zeitbudget der MitarbeiterInnen.

Wie auch bei anderen Projekten sollte auch hier ein Pilotprojekt überlegt werden. Günstig ist, wenn es sich um einen erfahrenen Projektleiter handelt, wenn realistische Rahmenbedingungen, ein realistischer Kosten- und Zeitplan vorliegt, sowie interessiertes und innovationsfreudiges Team, wenn der Gegenstand zum Teil schon bekannt ist und repräsentativ für andere Projekte gesehen werden kann und wenn eine stufenweise Einführung möglich ist. Nicht geeignet dagegen sind Projekte für Pilotanwendungen, wenn ein hoher Druck von außen herrscht, wenn ein zu knapper Kosten- und Zeitplan vorliegt, wenn das Team wenig aufgeschlossen ist für neue Methoden, wenn es sich um ein kritisches Projekt handelt und zu viele Personen im Team mitarbeiten und auch wenn kein ausreichender Support zur Verfügung steht. (siehe auch Rupp/Sophisten 2014, 498f) Bei Pilotprojekten ist regelmäßiges Feedback zielführend.

Bei der Migration von neuen Methoden ist es auf jeden Fall wichtig das Management des Unternehmens miteinzubeziehen.

Möchte man RE im Bibliotheksbereich einführen, so müsste man an mehreren Stellen ansetzen. Man bräuchte Personal, das sich sowohl mit RE als auch mit dem Bibliotheksbetrieb auskennt und eine ausreichende Finanzierung. Ein erster Schritt könnte sein, RE in den Universitätslehrgang „Library and Information Studies“ einzuplanen, aber auch in das allgemeine Weiterbildungsprogramm der jeweiligen Institutionen. Damit würde man beginnen Kompetenzen in diesem Bereich aufzubauen. Gemeinsame Fortbildungsveranstaltungen, an denen sowohl IT-Personal als auch BibliotheksmitarbeiterInnen teilnehmen, könnten ein weiterer Schritt in die richtige

Richtung sein. Auch auf bibliothekarischen Veranstaltungen, wären Vorträge oder Workshops zum Thema RE hilfreich, um ein gewisses Bewusstsein dafür zu schaffen.

4. Empirischer Teil

4.1 Zur Umfrage

Ein Ziel der Arbeit war es, ein Stimmungsbild der aktuellen Situation bzgl. RE an wissenschaftlichen Bibliotheken zu erhalten.
Die Umfrage, die sich an MitarbeiterInnen in Schlüsselpositionen, bzw. bei kleineren Einrichtungen auch an die LeiterInnen direkt richtete, war zum Teil als quantitativer, als auch als qualitativer Fragebogen konzipiert. Soweit möglich wurde die Befragung persönlich durchgeführt, um Unklarheiten sofort beseitigen und Missverständnissen vorbeugen zu können und um eventuell auch zusätzliche Informationen zu erhalten. Diese Methode hat dazu geführt, dass wertvolle Hinweise, die über die eigentliche Befragung hinausging, gegeben wurden und somit das gewonnene Stimmungsbild weiter verdeutlicht werden konnte. Einige Personen wurden aufgrund der räumlichen Distanz telefonisch befragt, in wenigen Ausnahmefällen wurde der Fragebogen von den Befragten selbständig ausgefüllt, aber auch in diesen Fällen wurde telefonisch Kontakt aufgenommen. Alle Personen erhielten die Fragen vorab per Mail zugesandt um sich mit dem Thema vertraut machen zu können.
Allen wurde absolute Anonymität versprochen, in einem Begleitschreiben wurde klargestellt, dass es sich um eine nichtrepräsentative Umfrage im Rahmen einer Masterthesis an wissenschaftlichen Bibliotheken handelt, die unter anderem zeigen soll, inwieweit das professionelle Erstellen und Managen von Anforderungen in Bibliotheken bereits realisiert ist. Ausgehend von dem Gedanken, dass die Zusammenarbeit von wissenschaftlichen Bibliotheken mit IT-Abteilungen und/oder IT-Firmen immer mehr zunimmt, stellt sich die Frage, wie sich die gemeinsamen Projekte zwischen Bibliothek und IT heute gestalten, wie sie von den betroffenen Personen erlebt werden und welche Wünsche es bzgl. der zukünftigen Zusammenarbeit gibt.
Die ersten 11 Fragen beschäftigen sich mit der Art und der subjektiv empfunden Qualität der Zusammenarbeit zwischen IT und Bibliothek und mit dem Umgang mit Projekten im Allgemeinen, die Fragen 12 bis 27 zielen speziell auf den Umgang mit Anforderungen in der eigenen Institution und der subjektiven Zufriedenheit damit ab,

die Fragen 28 bis 35 sind auf den derzeitigen Umgang mit Projektmanagement, bzw. RE abgestimmt.

Sämtliche angefragte Personen waren erfreulicherweise bereit, die Fragen zu beantworten, wenn auch nicht immer alle in Details Auskunft geben konnten. Da die Umfrage sich vor allem auf den Bibliotheksbereich bezog, schien es angemessen, dass sechs Personen, also knapp ein Drittel der insgesamt 19 Befragten aus dem IT-Bereich – hier wurden nur IT-Abteilungen, keine Firmen berücksichtigt – stammen.

Folgende 13 Institutionen konnten in die Umfrage einbezogen werden:

- Akademie der Bildenden Künste
- Arbeiterkammer Wien
- Kunstuniversität Graz
- Kunstuniversität Linz
- Medizinische Universität Wien
- Österreichische Nationalbibliothek
- Technische Universität Wien
- Universität für Musik und darstellende Kunst Wien
- Universität Innsbruck
- Universität Padua[28]
- Universität Salzburg
- Universität Wien
- Veterinärmedizinische Universität Wien

[28] Die Universität Padua wurde als einzige nichtösterreichische Institution miteinbezogen, da es sich dabei um eine mitentwickelnde Partnerinstitution im Bereich der Digitalen Langzeitarchivierung der UB Wien handelt.

4.2 Die Ergebnisse der Umfrage

1. Sagt Ihnen der Begriff Requirements-Management/Requirements-Engineering etwas?

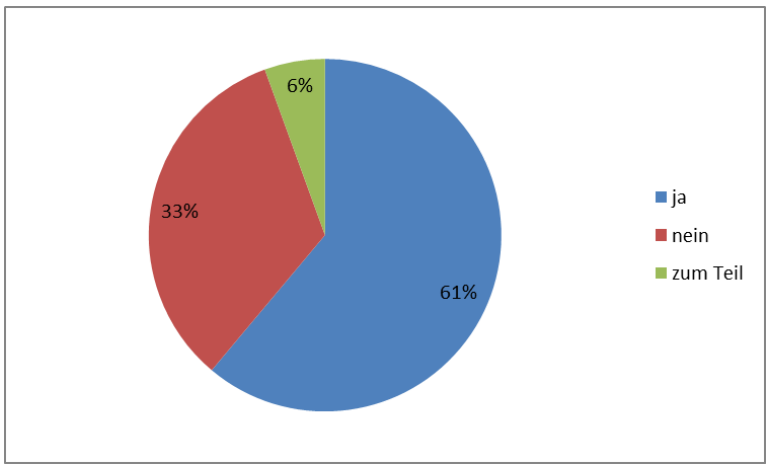


Abb. 2

Diese Frage wurde an den Anfang als Einstieg in das Umfragethema gestellt. Nur einem Drittel aller Befragten waren die Begriffe RE und RM völlig unbekannt, alle anderen hatten zumindest schon davon gehört.

2. Werden in Ihrer Einrichtung gemeinsame Projekte zwischen der Bibliothek und der IT-Abteilung, bzw. einer IT-Firma durchgeführt?

In allen Fällen werden Projekte sowohl mit der IT-Abteilung als auch mit Firmen durchgeführt, nur eine einzige interviewte Person gab an, dass kaum mit Personen aus dem IT-Bereich zusammengearbeitet wird.

3. Wenn ja, seit wann, gibt es diese Zusammenarbeit zwischen der Bibliothek und der IT-Abteilung, bzw. IT-Firmen?

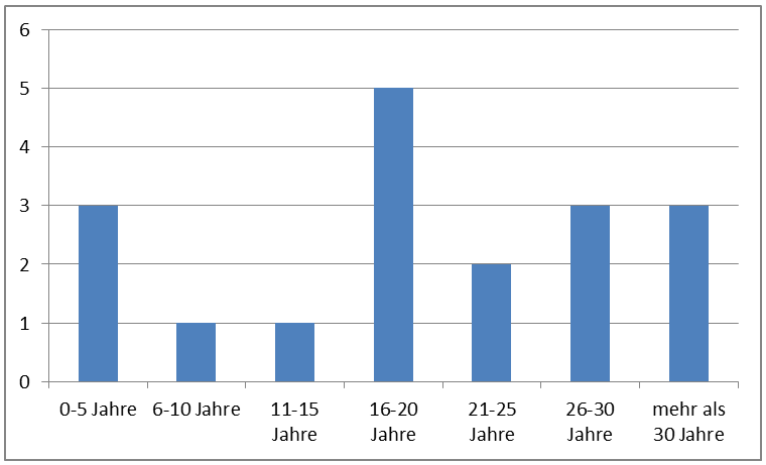


Abb. 3

Bei dieser Frage zeigte sich, dass die Dauer der Zusammenarbeit zwischen Bibliothek und IT-Abteilung, bzw. IT-Firma sehr unterschiedlich ist, am häufigsten reichen gemeinsame Projekte 16-20 Jahre zurück, was zeigt, dass sich da bereits eine große Erfahrung vorhanden ist.

4. Wie häufig arbeiten Sie zusammen?

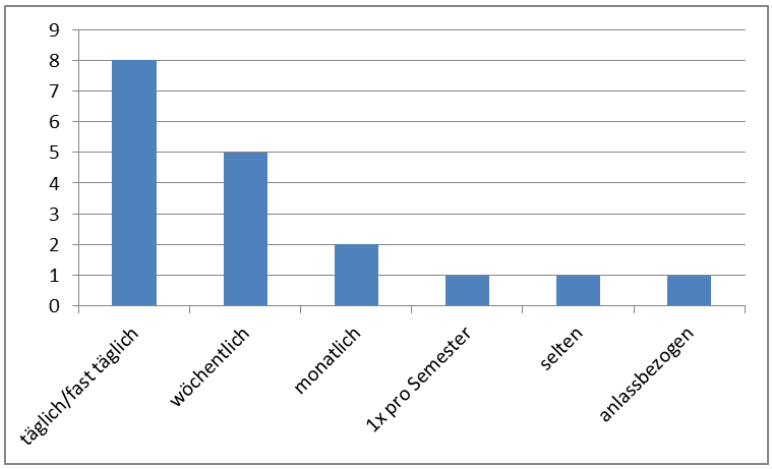


Abb. 4

Hier überraschten die Antworten insofern, als die meisten Befragten angaben täglich oder fast täglich zusammenzuarbeiten, man also von einem intensiven Kontakt ausgehen kann.

5. Welche Abteilungen der Bibliothek arbeiten eng mit der IT-Abteilung/mit der Firma zusammen?

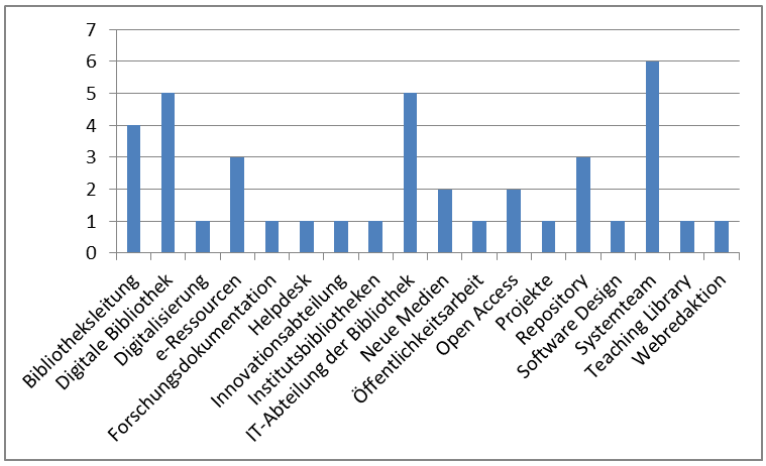


Abb. 5

Bei dieser Frage war eine Mehrfachauswahl möglich. Die einzelnen Institutionen sind hier schwer vergleichbar, da sich die Benennungen, bzw. auch die Aufgabenbereiche der Abteilungen stark überlappen. Außerdem gibt es zwischen den Abteilungen sicher auch Überschneidungen. Deutlich wird jedoch, wenig überraschend, dass vor allem die Digitale Bibliothek, die IT-Abteilung der Bibliothek und das Systemteam IT-Projekte durchführen.

6. Wie viele Mitarbeiterinnen und Mitarbeiter sind auf der Bibliotheksseite involviert?

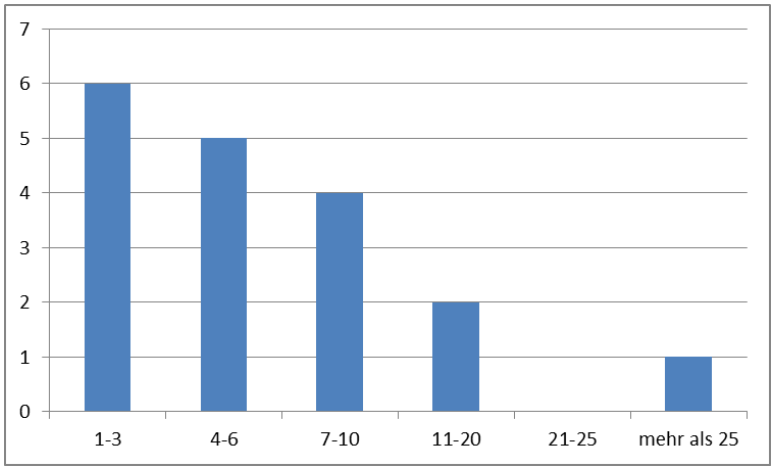


Abb. 6

Die Frage ergab, dass am häufigsten bis zu 10 BibliotheksmitarbeiterInnen in IT-Projekte eingebunden sind.

7. **Wie viele Mitarbeiterinnen und Mitarbeiter sind auf der IT-Seite involviert?**

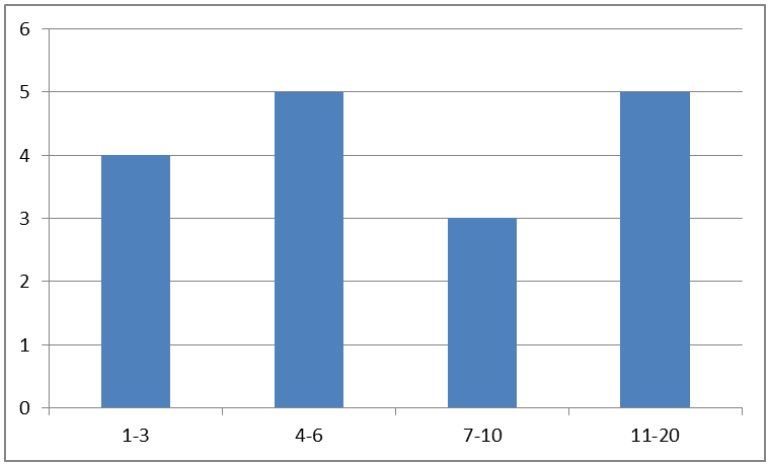


Abb. 7

Hier wird deutlich, dass auf der IT-Seite deutlich mehr MitarbeiterInnen, die in Bibliotheksprojekte eingebunden sind, als dies umgekehrt auf der Bibliotheksseite der Fall ist.

8. Um welche Projekte handelt es sich dabei?

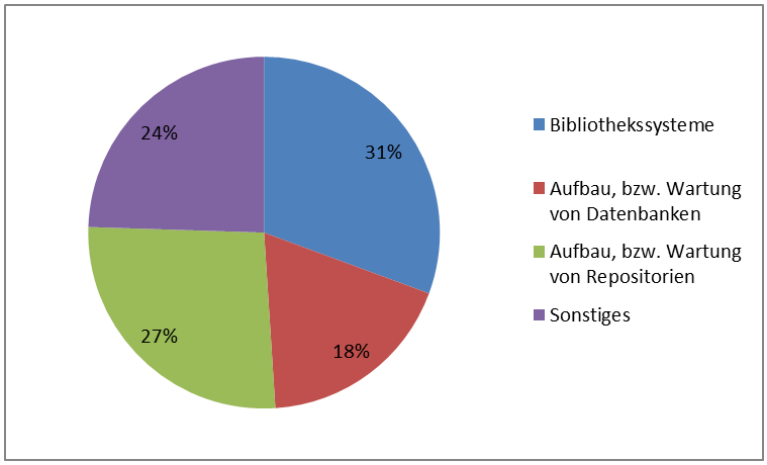


Abb. 8

Bei dieser Frage waren nur die angeführten vier Kategorien auswählbar, die sich nicht wie bei der Frage 5 auf Abteilungen, sondern auf konkrete Aufgaben bezogen.

Bibliothekssysteme nehmen dabei den größten Anteil ein, relativ dicht gefolgt vom Aufbau und von der Wartung von Repositorien. Unter „Sonstiges“ wurde in alphabetischer Reihenfolge genannt: Accountverwaltung, Applikationen, Digitalisierung, elektronische Zeitschriften, E-Mailserver, E-Ressourcen, Hardware, Hochschulschriftenserver, Intranet, Katalog, Netzinfrastruktur, Portale, Publikationssysteme, Schnittstellen, Share, Storage, Ticketsystem, Webserver, Website. Diese Angaben zeigen, mit wie vielen unterschiedlichen Themen sich heute Projekte zwischen IT-Abteilungen, bzw. Firmen und Bibliotheken beschäftigen.

9. Gibt es gemeinsame Treffen, bei denen über Anforderungen gesprochen wird?

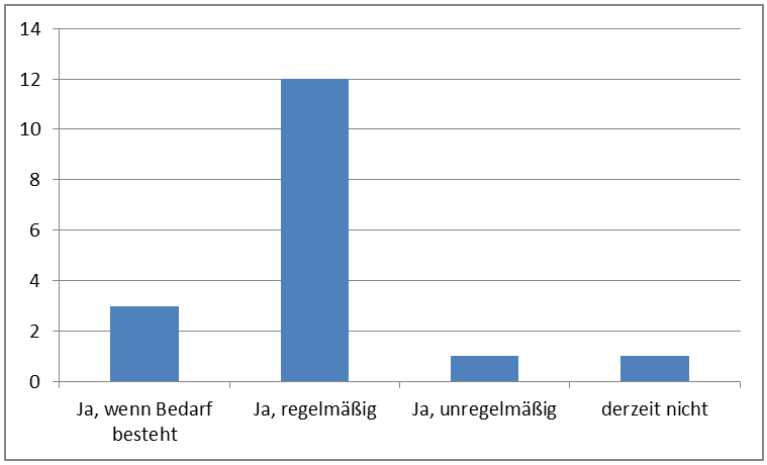


Abb. 9

Die Antworten haben gezeigt, dass Anforderungen in den meisten Institutionen regelmäßig bei gemeinsamen Treffen besprochen werden, bzw. wird ständig über ein Ticketsystem, telefonisch und auch auf anderen Wegen kommuniziert.

10. Gibt es in Ihrer Einrichtung eine bestimmte Person oder ein fixes Team, die oder das für das Projektmanagement zuständig ist, oder ändert sich die personelle Zusammensetzung von Projekt zu Projekt?

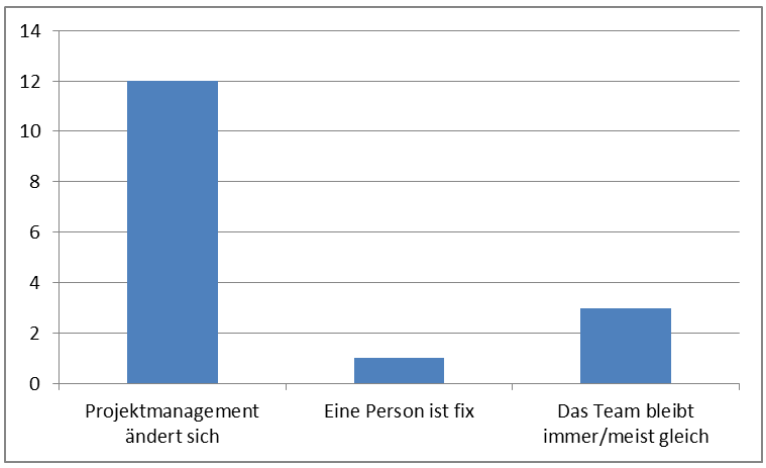


Abb. 10

Die Antworten hängen sicherlich auch mit der unterschiedlichen Größe der Institutionen zusammen. Kleinere agieren recht flexibel und setzen je nach Projekt ein unterschiedliches Projektmanagement ein.

11. Gibt es klar definierte Rollen?

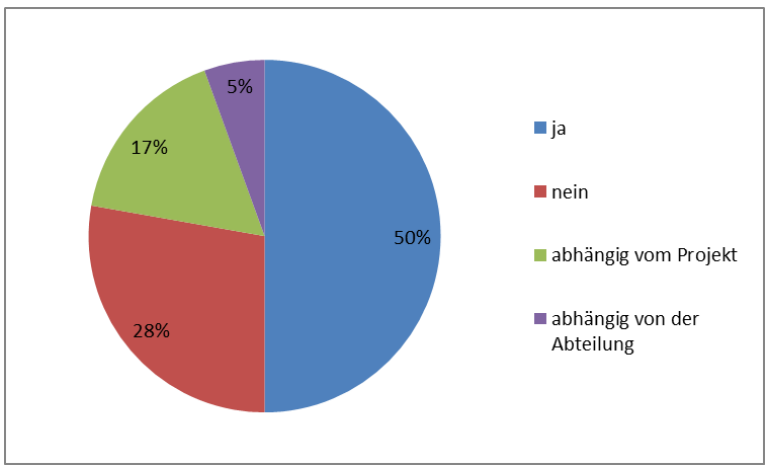


Abb. 11

Genau die Hälfte der Befragten bejahte diese Frage, wobei eine Peron meinte: „Ja, aber die Rollen werden nicht eingehalten!“. Laut den Empfehlungen des RE ist diese Situationen natürlich nicht optimal, da Rollen transparent für alle vergeben werden

sollen, mit gewissen Rechten und Pflichten ausgestattet sind und deshalb jede Änderung demnach sofort vermerkt und kommuniziert werden müsste. Mehr als ein Viertel der InterviewpartnerInnen merkte an, dass der Umgang mit Rollen je nach Projekt unterschiedlich gehandhabt wird und einige berichteten, dass die einzelnen Abteilungen einer Institution den Umgang mit Rollen eher flexibler halten.

12. Wird für das Managen von Anforderungen Zeit eingeplant?

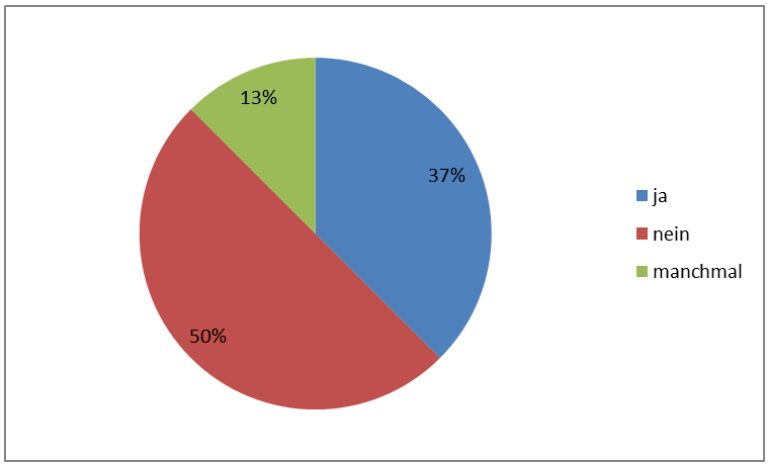


Abb. 12

Die Hälfte aller Befragten verneinte dies, immerhin 37% bestätigten, dass für das Managen von Anforderungen Zeit eingeplant wurde. Laut RE-Vorgaben ist das Erstellen und Einhalten eines Zeitplanes für die positive Abwicklung eines Projekts jedoch unerlässlich. Bei jenen, die Zeit einplanten, war das Managen, bzw. die Analyse und Konzeption von Anforderungen ein wichtiger Teil der Strategie.

13. Werden die Zuständigkeiten im Projekt schon zu Beginn festgelegt?

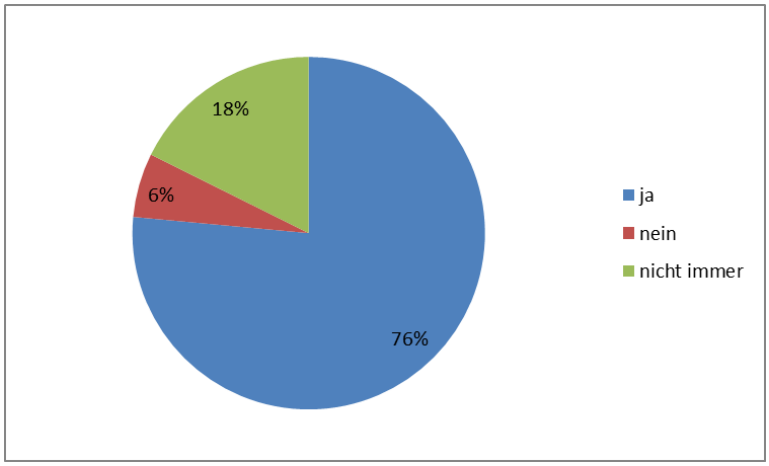


Abb. 13

Mehr als drei Viertel der Befragten gab an, dass die Zuständigkeiten zu Beginn des Projektes festgelegt werden würden. Auf eine vertiefende Frage, ob allen Beteiligten bekannt sei, wer für die Erstellung von Anforderungen, deren Prüfung und Dokumentation zuständig sei, wurde ähnlich geantwortet. Das heißt aber auch, dass bei ca. einem Viertel aller Fälle der Umgang mit den Anforderungen nicht optimal läuft, denn die Zuständigkeiten müssen laut Empfehlungen von RE-ExpertInnen von Anfang festgelegt und transparent sein.

14. Wird der Prozess rund um die Anforderungen dokumentiert?

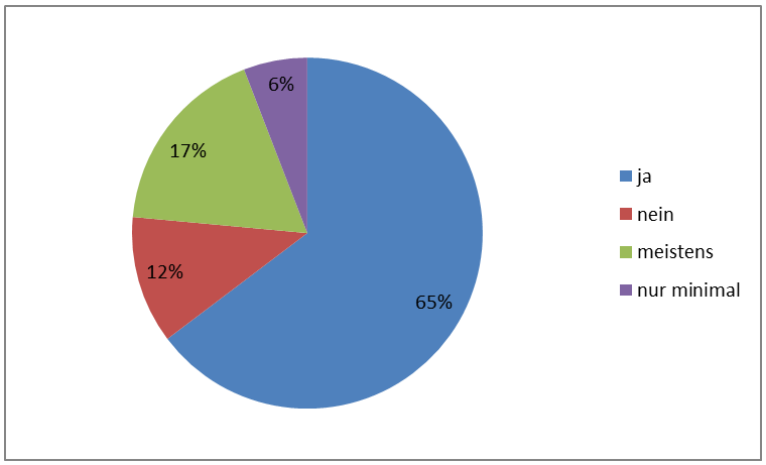


Abb. 14

Mehrheitlich wird eine Dokumentation vorgenommen, wenn sie auch nicht in allen Fällen als ausreichend empfunden wird. Bedenkt man, wie wichtig Dokumentation für das RE ist und wie vehement die ExpertInnen in der Literatur darauf hinweisen, kann man anhand dieser Antworten sehen, dass hier noch ein großes Verbesserungspotential vorhanden ist.

14 a. Wer dokumentiert?

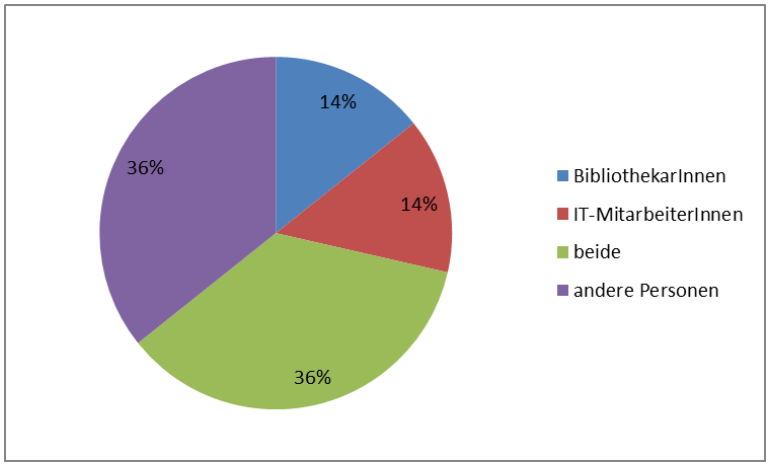


Abb. 15

Diese Frage wurde nur gestellt, wenn die obere Frage mit „ja“ beantwortet wurde. Die Dokumentation scheint in den jeweiligen Institutionen recht unterschiedlich vorgenommen zu werden. Häufig dokumentieren die Bibliotheken gemeinsam mit der IT-Abteilung, aber ebenso häufig übernehmen andere Personen diese Arbeit**.** Die Tatsache, dass nur in 36% der Fälle die Dokumentation gemeinsam zwischen der Bibliothek und den IT-Abteilungen vorgenommen wird, lässt ebenfalls darauf schließen, dass hier noch einiges verbessert werden sollte.

14. b. Wie wird dokumentiert?

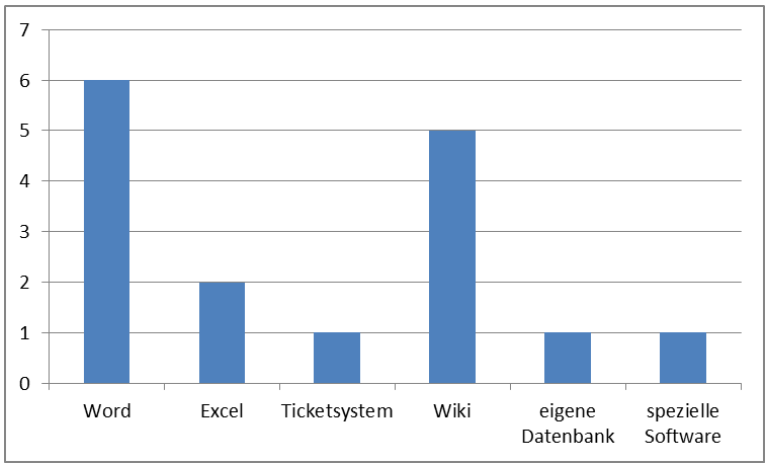


Abb. 15

Auch diese Frage wurde nur jenen Personen gestellt, die die Frage nach der Dokumentation des Prozesses bejaht hatten. Demnach wird immer noch am häufigsten mittels Worddokumenten dokumentiert, an zweiter Stelle folgt bereits ein Wiki. Eine spezielle Software oder ein Ticketsystem wird nur selten verwendet.

14.c. Wird diese Dokumentation inhaltlich überprüft?

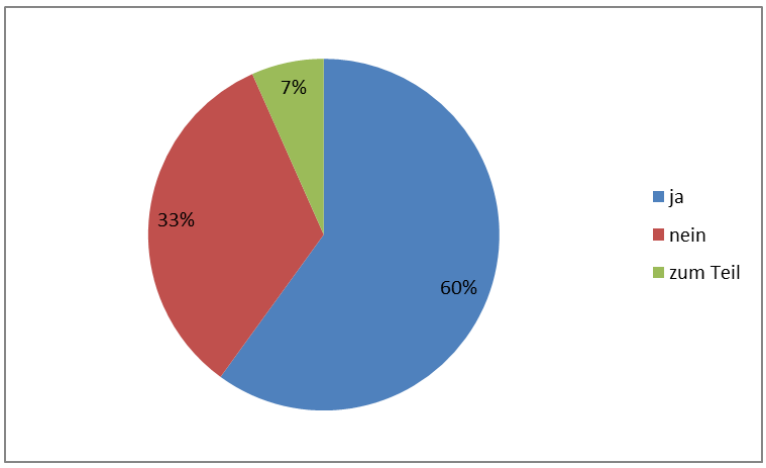


Abb. 16

Nur in 60% der Fälle wird diese Dokumentation überprüft. Laut Pohl und Rupp sollte die Konformität zum Dokumentationsformat und zur Dokumentenstruktur überprüft werden, die Verständlichkeit, die Eindeutigkeit, und die Konformität mit den Dokumentationsregeln. (siehe Pohl/Rupp 2015, 99f) Fehlen diese Schritte kann es im weiteren Projektverlauf zu Problemen kommen.

14.d. Wer überprüft die Dokumentation?

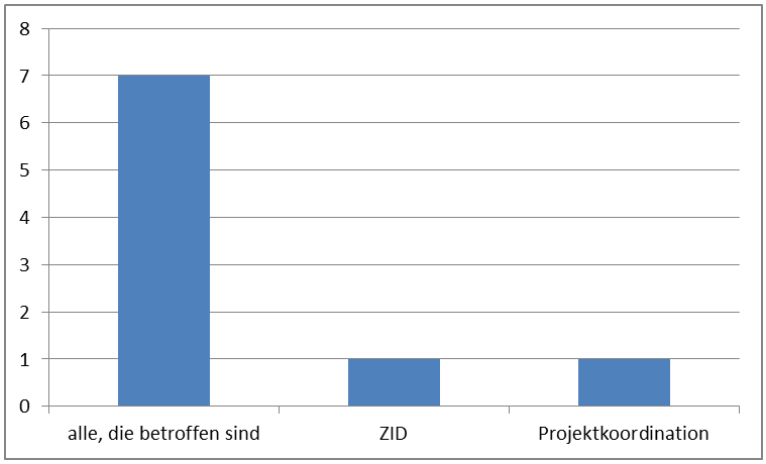


Abb. 17

Wenn die Dokumentation inhaltlich überprüft wird, geschieht das in den meisten Fällen von allen Stakeholdern, die von den Anforderungen betroffen sind. Nur in wenigen Fällen wird ausschließlich vom ZID oder von der Projektkoordination geprüft, was gemäß den Vorgaben der RE-Regeln nicht empfehlenswert ist.

14.e. Wird die Dokumentation regelmäßig aktualisiert?

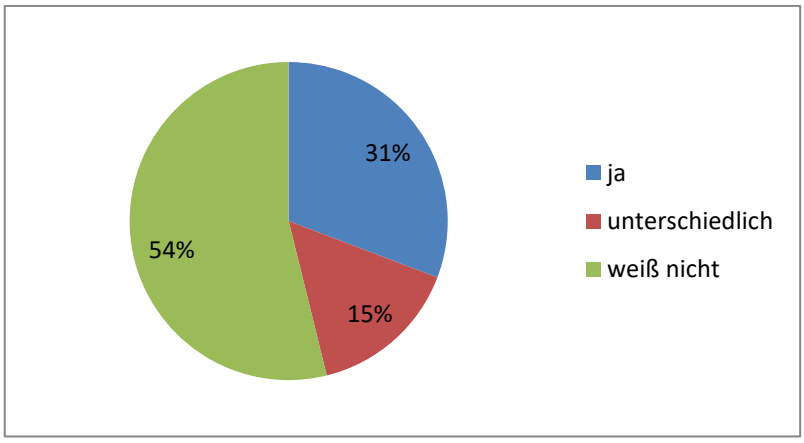


Abb. 18

Bei dieser Frage ist interessant, dass die meisten Befragten, wenn sie nicht direkt von den Anforderungen betroffen waren, nicht wussten, ob die Dokumentation regelmäßig aktualisiert wurde oder nicht. Das lässt darauf schließen, dass es zu wenig Transparenz in den Projekten gibt, bzw. dass diesem Punkt wenig Wichtigkeit beigemessen wird.

15. Wer erstellt die Anforderungen, die umgesetzt werden sollen?

Bei dieser offen gestellten Frage stellte sich heraus, dass in den meisten Fällen die Anforderungen gemäß den RE-Regeln gemeinsam im Team erstellt werden, in einem ersten Schritt oft von den BibliothekarInnen, in einem zweiten Schritt gemeinsam mit der IT-Abteilung, bzw. mit der Firma. In seltenen Fällen erstellen die BibliotheksmitarbeiterInnen die Anforderungen alleine, was natürlich, wie man an den theoretischen Ausführungen erkennen kann, nicht optimal ist.

16. Gibt es für das Eruieren von Anforderungen bereits Workflows?

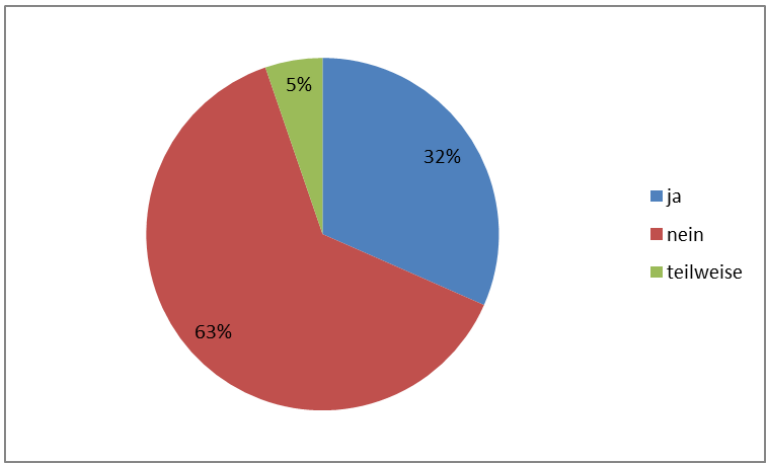


Abb. 19

Diese Frage wurde mehrheitlich verneint, nur knapp ein Drittel orientiert sich an bestimmten Abläufen, die unter anderem mittels Ticket-System abgebildet werden.

16.a. Werden für das Eruieren von Anforderungen bestimmte Methoden verwendet?

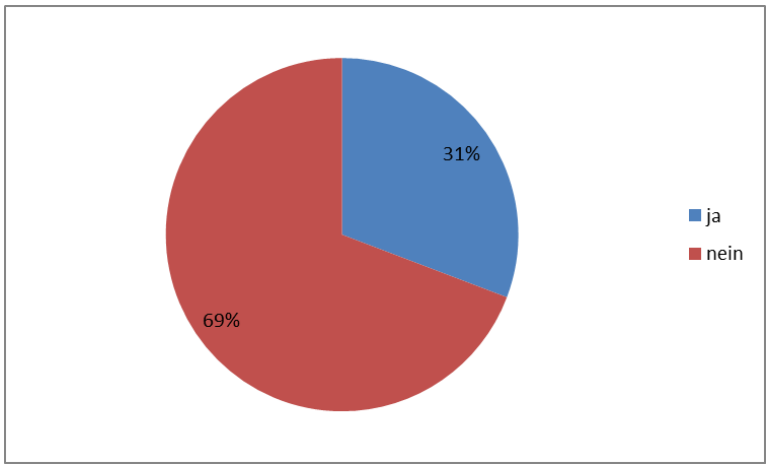


Abb. 20

Nur knapp ein Drittel der InterviewpartnerInnen konnte diese Frage positiv beantworten. Unter anderem wurden Interviews und Umfragen durchgeführt, aber auch beispielsweise Trends beobachtet. Bei den anderen ergaben sich Anforderungen in Gesprächen, Meetings oder wurden per Mail geschickt.

17. Werden die Anforderungen untereinander abgestimmt?

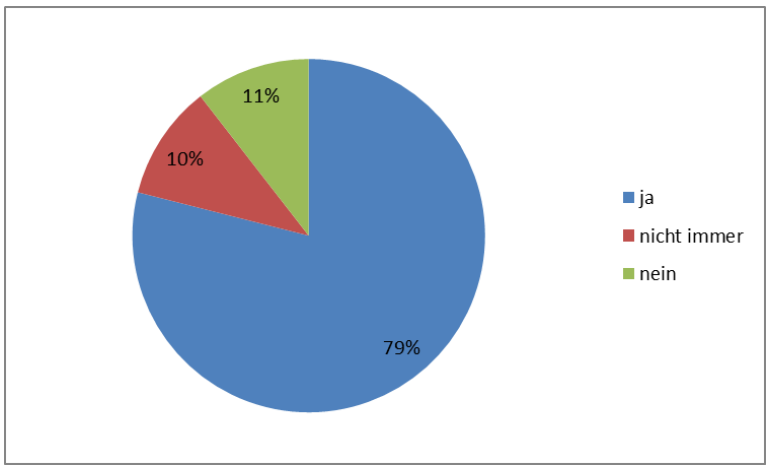


Abb. 21

Immerhin bejahten 79% der Befragten dies, zum Teil auch mit den EndnutzerInnen, wenn auch bei 11% keine Abstimmung vorgenommen wurde. Einige der Befragten gaben explizit an, dass agil gearbeitet werden würde. Einige beschäftigten auch explizit

mit den Konsequenzen, wenn die Anforderungen nicht umgesetzt werden würde, wurde genannt. Bei den anderen war zumindest das Bewusstsein vorhanden, dass diese Vorgangsweise nicht optimal ist.

18. Wenn ja, wer nimmt diese Abstimmung vor?

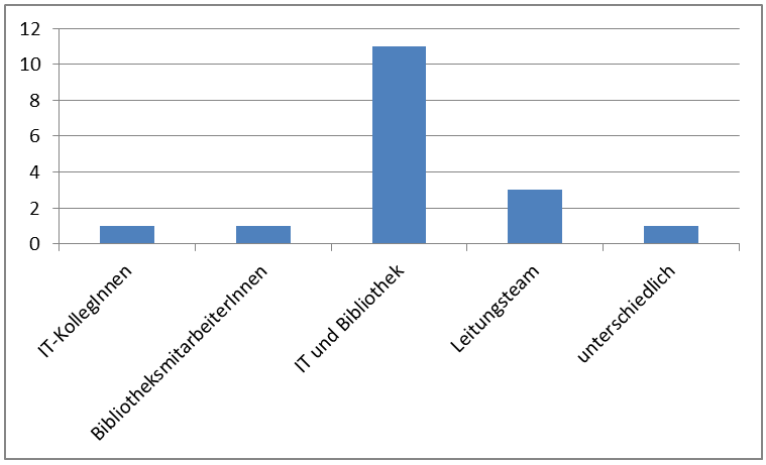


Abb. 22

Wenn es eine Abstimmung zwischen den Anforderungen gibt, wird diese meist, aber nicht immer, von KollegInnen der IT und der Bibliothek gemeinsam durchgeführt. In Einzelfällen erfolgt die Abstimmung auch nur durch die Leitungsebene, was laut RE-Regeln zu Akzeptanzschwierigkeiten führen könnte.

19. Wie werden die Anforderungen selbst dokumentiert?

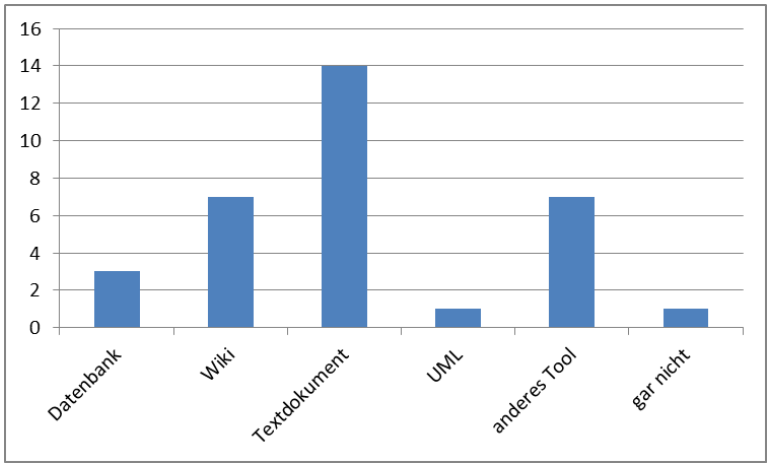


Abb. 23

Bei dieser Frage waren Mehrfachantworten möglich. Hauptsächlich werden die Anforderungen selbst noch als Textdateien dokumentiert. Unter der Rubrik „anderes Tool“ wurde relativ häufig Excel genannt. UML wird kaum eingesetzt, obwohl sie von ExpertInnen oft als die geeignetste Methode für die Dokumentation bezeichnet wird. (siehe etwa Rupp/Sophisten 2014, 394)

20. Kennen Sie Standards des Requirements-Engineerings?

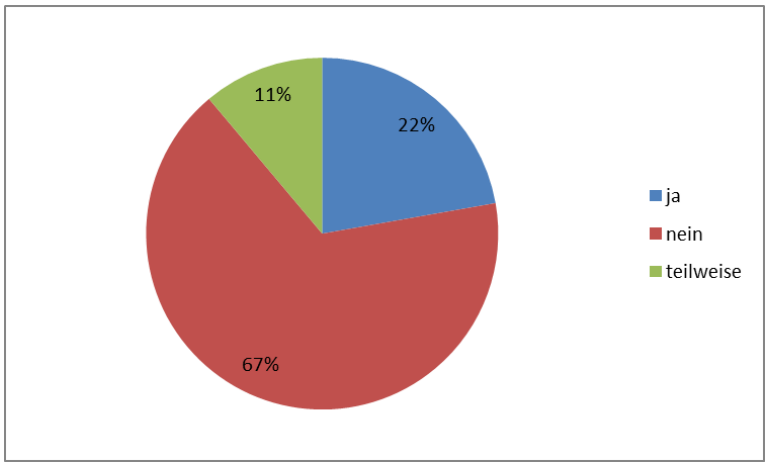


Abb. 24

Nicht einmal ein Viertel der Befragten kannte irgendwelche Standards des RE, was darauf schließen lässt, dass obwohl laut der ersten Frage, RE zwei Dritteln der Befragten bekannt ist, bisher keine intensivere Beschäftigung mit dem Thema stattgefunden hat.

20.a. Wenn ja, wird an Ihrer Institution darauf zurückgegriffen?

Von jenen, die vertraut mit RE-Standards sind, gaben die meisten an, dass diese in ihrer Institution, oft aus Kostengründen, bzw. weil sie die einzigen waren, die sich damit auskannten, nicht eingesetzt werden würden. In einem Fall war jedoch eine Stelle ausgeschrieben, die sich in Zukunft mit RE-Agenden befassen soll.

21. Werden die Anforderungen qualitätsgeprüft?

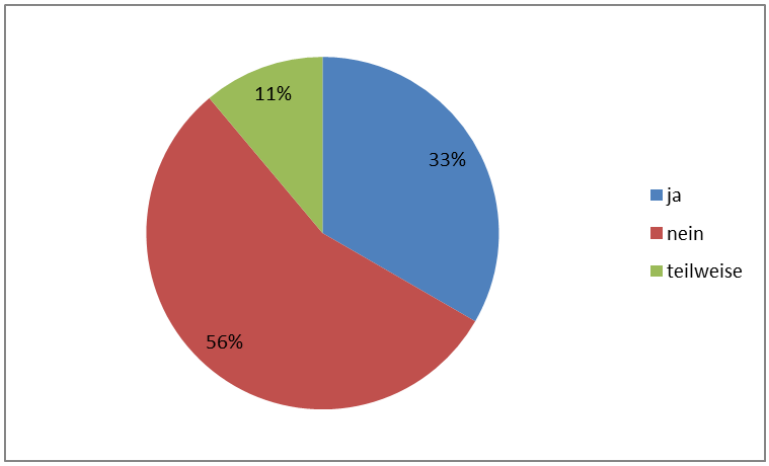


Abb. 25

Knapp mehr als die Hälfte verneinte diese Frage, nur ein Drittel antwortete mit einem klaren „Ja“, obwohl wie in Frage 17 ersichtlich, die meisten Anforderungen untereinander abgestimmt wurden. Das lässt darauf schließen, dass oft unfertige und eventuell sogar widersprüchliche Anforderungen der zuständigen IT-Person übergeben werden. Die, die die Frage bejahen konnten, wurden befragt, wer die Überprüfung vornimmt. Die Antworten fielen sehr unterschiedlich aus. Meist wurde gegenseitig im Team überprüft. Dabei wurde vor allem auf die Ressourcenplanung und Transparenz geachtet. Diese Prüfung wurde nur zu 50% dokumentiert, was laut den RE-Regeln nicht zufriedenstellend ist. Insgesamt wäre eine Qualitätsprüfung für die diversen Projekte sicher hilfreich.

22. Gibt es einen Feedbackprozess?

Ein Viertel der InterviewpartnerInnen gab an, dass es keinen formalisierten Feedbackprozess gibt, obwohl die Mehrheit berichtete (Frage 17), dass die Anforderungen zu Beginn des Prozesses untereinander abgestimmt werden. Klare Feedbackschleifen gibt es dann, wenn beispielsweise die Kommunikation über ein Ticket-System läuft, oder wenn etwas nicht funktioniert, bzw. natürlich dann, wenn agil gearbeitet wird. Oft wird dieses Feedback nicht explizit ausgewiesen.

23. Wie wird mit unzureichenden Anforderungen umgegangen?

In den allermeisten Fällen wird der Prozess erneut gestartet, einige nannten explizit ein Changemanagement. Bei wenigen ist so ein Fall noch nicht eingetreten. Eine Person meinte treffend: „Wichtig ist erst mal zu wissen, dass die Anforderungen unzureichend sind, sonst kommt es zu Problemen."

24. Wer hat Zugriff auf die erstellten Anforderungen?

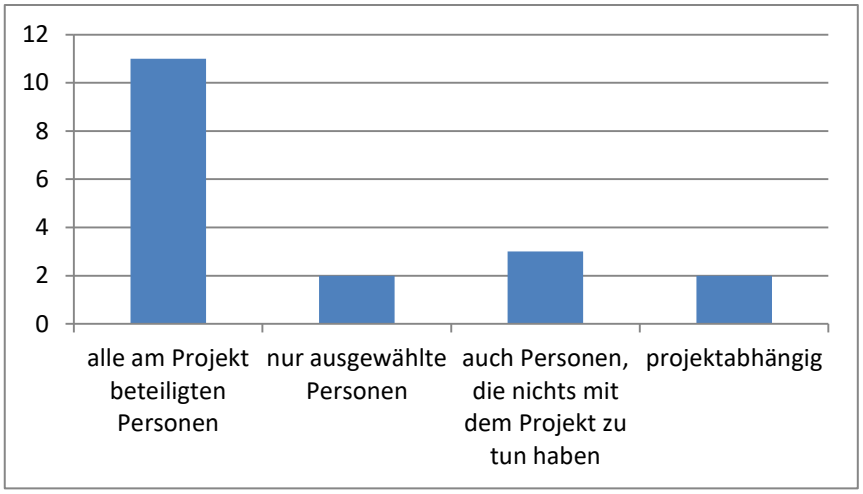


Abb. 26

Die Antworten zeigen, dass hier ziemlich viel Transparenz herrscht, wenn auch laut RE-Regeln der Zugriff noch nicht ausreichend für alle offen ist.

25. Was passiert, wenn im Nachhinein neue Anforderungen gestellt werden, bzw. die Anforderungen geändert werden?

Diese Frage war sehr offen gestellt, es gab keine vorgegeben Antwortmöglichkeiten. Für die meisten war das ein alltäglicher Fall. Interessant dabei ist, dass von einigen Befragten hier explizit die Vorteile des agilen Arbeitens herausgestrichen wurden. Von einigen wurde auf ein eher unkoordiniertes Vorgehen in ihrer Institution hingewiesen, auch ein fehlendes Changemanagement wurde vereinzelt kritisiert.

26. Wird die Umsetzung der Anforderungen überprüft?

Auf diese Frage antworteten alle Befragten mit einem klaren „Ja“. Am häufigsten wird diese Überprüfung von den zuständigen BibliotheksmitarbeiterInnen durchgeführt, manchmal auch von der IT-Abteilung und der Bibliothek gemeinsam.

27. Wird der Umsetzungsprozess dokumentiert?

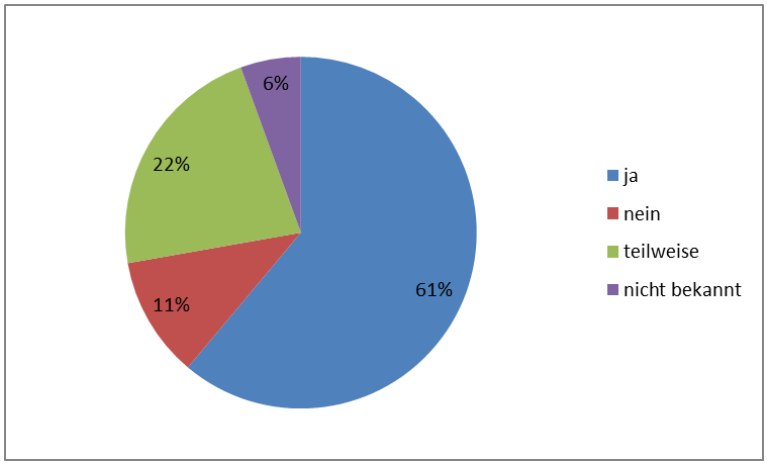


Abb. 27

In den meisten Fällen wurde der Prozess konform der RE-Regeln per Tickets, Berichten oder E-Mail dokumentiert, in seltenen Fällen wurde angegeben, dass nur die Abweichungen dokumentiert werden, was natürlich für eine gute Projektplanung nicht ausreichend ist.

28. Sind Sie im Allgemeinen zufrieden mit dem Umgang mit Anforderungen?

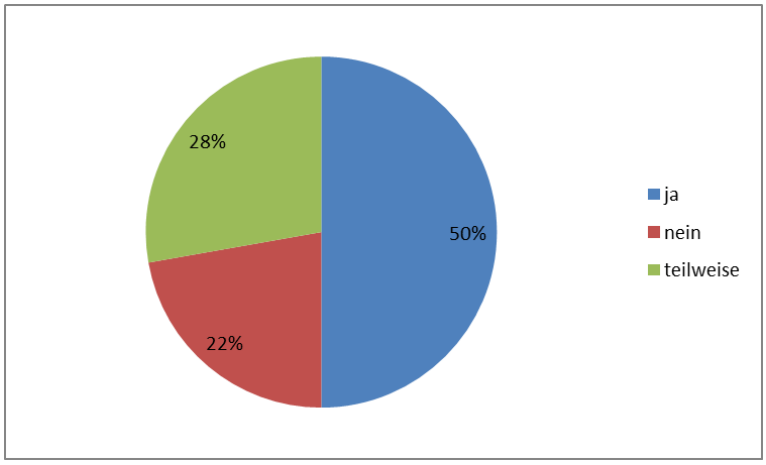


Abb. 28

Die Hälfte der Befragten gab an, zufrieden zu sein, andere trafen Aussagen, wie: „Zu viel Formalismus kann auch einschränkend sein“, „Eine engere Zusammenarbeit wäre hilfreich“. „Die Rollen sind nicht immer klar definiert“, „Man sollte nicht schon zu früh an die Umsetzung denken“, „Es herrscht keine Kultur in Sachen Projektmanagement“, „Es wäre besser, wenn strukturierter gearbeitet werden würde“, „Ein wenig professioneller wäre besser“, „Es sollten weniger unterschiedliche Tools verwendet werden“ aber auch „Der Ablauf von Projekten funktioniert viel besser als früher“. Insgesamt kann man bei dieser Frage erkennen, dass es im Bereich des Umgangs mit Anforderungen noch vieles zu optimieren gibt.

29. Steht Ihnen für die Umsetzung dieser Projekte genügend Personal und Zeit zur Verfügung?

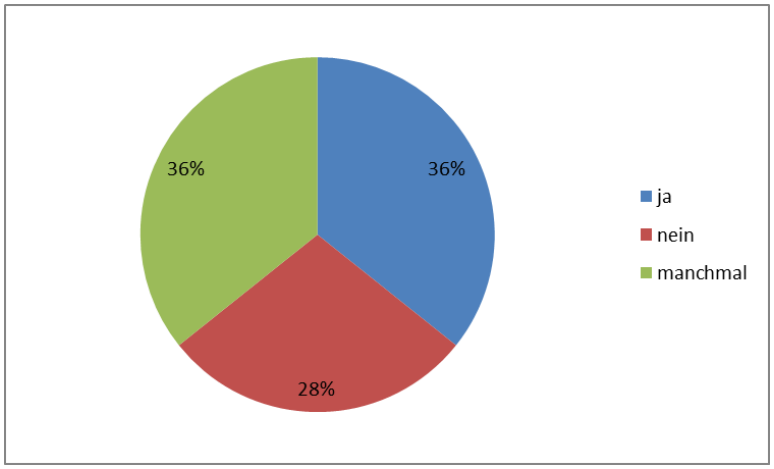


Abb. 29

Hier waren die Antworten recht ausgewogen, mit einer Tendenz zu ja. Von denen, die nicht oder nicht ganz zufrieden waren, meinte die Hälfte, dass nicht genug Personal zur Verfügung stünde, die andere Hälfte gab an, dass die Zeit oft knapp werden würde. Befolgt man den Rat der ExpertInnen sollte sowohl genügend Zeit als auch Personal vorhanden sein, um ein möglichst gutes Ergebnis zu erzielen. In der Realität sind wissenschaftliche Bibliotheken finanziell oft von der Trägerinstitution abhängig.

30. Wenn Sie nicht zufrieden sind, wo sehen Sie Verbesserungsmöglichkeiten?

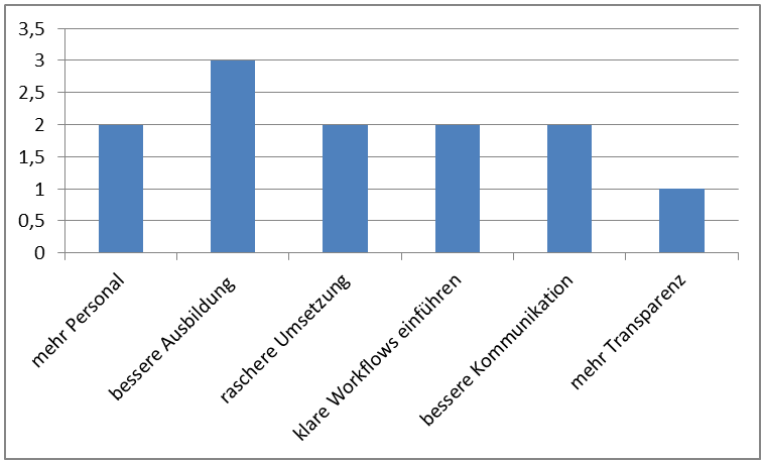


Abb. 30

An erster Stelle der geäußerten Verbesserungswünsche steht der Wunsch nach besser ausgebildetem Personal im Bereich des Aufbaus von e-Infrastrukturen, eine Forderung, der auch die Ausbildungsinstitutionen von BibliothekarInnen nachkommen sollten. Mehr Personal für diese neuen Aufgaben einzusetzen ist natürlich ebenso wichtig, aber auch eine gut funktionierende Kommunikation zwischen allen Beteiligten ist dringend nötig. Im Bereich der Workflows und in der Umsetzungsgeschwindigkeit wäre ebenfalls noch einiges zu verbessern.

31. Ist es Ihrer Meinung nach wichtig, dass es auch bei kleineren Projekten ein Management gibt, das sich um gewisse Dinge kümmert?

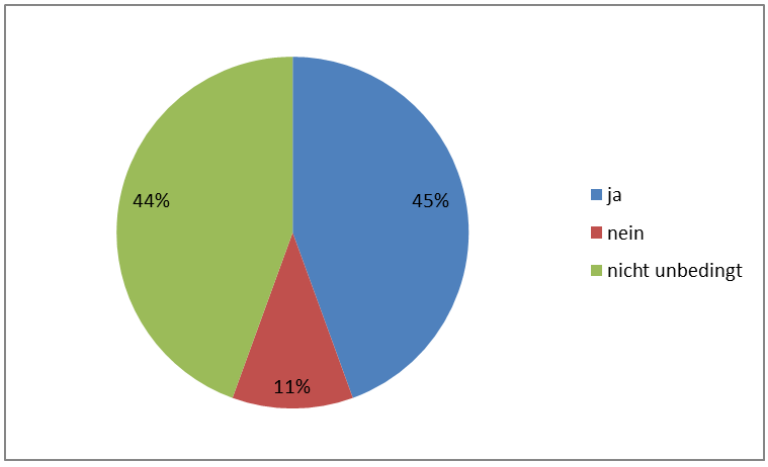


Abb. 31

Die Mehrheit der Befragten sprach sich zwar in jedem Fall für ein Projektmanagement aus, es wurden jedoch auch zahlreiche Gründe angeführt, die dagegen sprechen.
Eine Sorge war, dass das Durchführen von kleineren Projekten dann zu kompliziert werden könnte und dass zu viel Formalismus den Projektfortgang bremsen könnte.
Viele gaben an, dass es auf die Komplexität der Projekte ankäme, das Projekt dürfe durch den Mehraufwand nicht ineffizient werden. Bei den Befürwortern wiesen einige auf die Wichtigkeit eines klaren Wordings hin.

32. Ist Requirements-Engineering in Ihrer Einrichtung ein Thema?

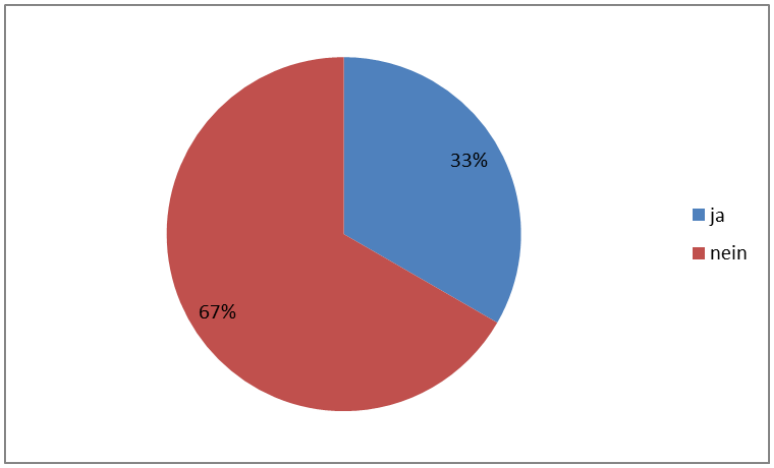


Abb. 32

Für zwei Drittel war RE kein Thema. Bei jenem Drittel, die angaben, dass RE in ihrer Institution thematisiert werden würde, gaben einige an, dass es vor allem im Umgang mit externen Kooperationspartnern sehr wichtig wäre, vor allem auch zur gegenseitigen Absicherung. Wirklich in den Alltag integriert ist RE aktuell in Bibliotheken kaum.

33. Gibt es ausgebildete Personen im Bereich Requirements-Engineering? Wenn ja, welche Ausbildung haben sie?

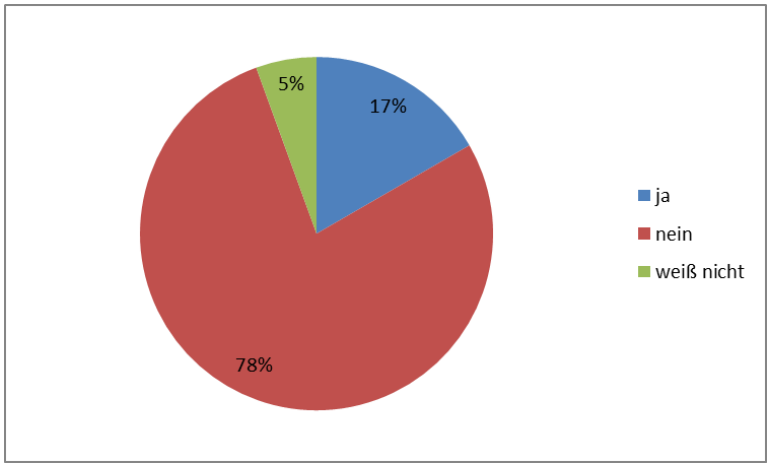


Abb. 33

In nur 17% der Fälle gab es ausgebildete MitarbeiterInnen in den jeweiligen Institutionen, in zwei Fällen besaßen diese das Zertifikat „Certified Professional for Requirements Engineering (CPRE)“

34. Werden in Ihrer Institution Weiterbildungen im Bereich Requirements-Engineering angeboten?

Auf diese Frage antworteten alle Personen, bis auf eine, die sich nicht sicher war, mit „nein“. Dass dieses Thema auch in großen Institutionen noch nicht im Weiterbildungsprogramm zu finden ist, ist erstens bedauerlich und zweitens auch verwunderlich, wenn man sich beispielsweise das breite Angebot an Weiterbildungsmöglichkeiten der Universität Wien ansieht.

35. Sehen Sie einen Bedarf von Requirements-Engineering in der Zukunft?

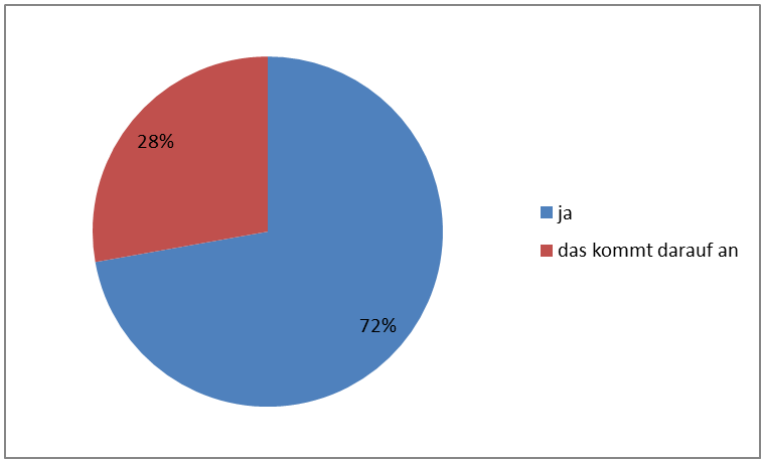


Abb. 34

72% aller InterviewpartnerInnen sprachen sich dafür aus. Als Gründe wurden genannt:

- Für die gesamte Institution wäre sicher Bedarf vorhanden
- Sicher ein wichtiges Thema zur Professionalisierung von Abläufen
- In Zukunft braucht es mehr Schnittstellenpersonen, die sich auf gewisse Themen spezialisieren, da die Anforderung der Fachbereiche immer komplexer werden und ohne eine tiefgreifendes Verständnis für den Fachbereich ist es nicht mehr möglich die Anforderung wirklich zu verstehen, da oft vom Gleichen gesprochen wird aber etwas anderes gemeint wird.
- Für größere Organisationen auf jeden Fall, da mehr Koordinationsarbeit zwischen verschiedenen Abteilungen und Stakeholdern erforderlich ist.
- Die IT-Zusammenarbeit wird immer wichtiger
- Ist eine alltägliche Anforderung an alle
- Ja, das Thema wird immer stärker werden, da immer mehr Projekte in immer kürzeren Zeitabständen bewältigt werden müssen
- Wenn es in Zukunft mehr Projekte gibt, wovon auszugehen ist, wäre es sinnvoll
- Man muss eigentlich nur Leute zusammenbringen. Es handelt sich eigentlich um eine Mediatoraufgabe, man ist ein Übersetzer zwischen den Welten.

Jene, die sich nicht sicher waren, meinten:

- Das Problem liegt darin, dass die Ausbildung Zeit kostet. Dabei ist auch der Kosten-Nutzen-Faktor sehr wichtig.
- Ja, wenn sich jemand zusätzlich darum kümmern kann, sonst zu viel Verwaltungsaufwand.
- Wichtig wären Erfahrungen in diesem Bereich und die Persönlichkeit muss stimmen, z.B. kommunikative Fähigkeiten
- Es soll keinen Overkill an Management geben. Eine professionelle Projektabwicklung ist jedoch sehr wichtig.

4.3 Interpretation der Ergebnisse

4.3.1 Zusammenarbeit zwischen der Bibliothek und der IT-Abteilung, bzw. IT-Firmen

Fast alle Befragten gaben an, Projekte sowohl mit der eigenen IT-Abteilung als auch mit IT-Firmen Projekte durchzuführen. (Frage 2) In den meisten Fällen reicht diese Zusammenarbeit schon 16 bis 20 Jahre zurück, bei vielen sogar 26 bis 30 Jahre und bei manchen sogar mehr als 30 Jahre. (Frage 3) Bei vielen hat die Zusammenarbeit begonnen, als die EDV in den Bibliotheken eingeführt wurde. Die Frage, wie häufig diese Zusammenarbeit erfolgt (Frage 4), zeigt, dass in den meisten Fällen täglich, bzw. fast täglich oder zumindest wöchentlich zusammen gearbeitet wird. Man kann also von einer intensiven Zusammenarbeit sprechen. Bemerkenswert ist, dass bereits sehr viele Abteilungen, bzw. Bereiche von Bibliotheken IT-Projekte durchführen. Bei der Frage 5, bei der eine Mehrfachauswahl möglich war, wurden am häufigsten das Systemteam, die IT-Abteilung und die Digitale Bibliothek genannt, was wenig überraschend ist. Interessant ist, dass bei kleineren Bibliotheken oft die Bibliotheksleitung selbst in Projekte involviert ist. Die Bandbreite der Antworten zeigt zusätzlich, wie viele unterschiedliche Aufgaben wissenschaftliche Bibliotheken heute wahrnehmen, bei denen die IT eine wesentliche Rolle spielt. Digitalisierung, der Umgang mit e-Ressourcen, die Forschungsdokumentation, die Einrichtung von Open Access-Abteilungen, der Aufbau und Betrieb von Repositorien und weitere Handlungsfelder wie beispielsweise die Teaching Library zählen dazu. In den meisten Fällen waren bei den IT-Projekten auf der Bibliotheksseite nur ein bis drei MitarbeiterInnen involviert, wobei die Angabe der Zahlen natürlich sehr stark von der Größe der Bibliothek abhängt. (Frage 6) Auf der IT-Seite wurden gleich häufig vier bis sechs und sogar 11-20 MitarbeiterInnen genannt, auch hier natürlich abhängig von der Größe und Struktur der Organisation, insgesamt wurde jedoch eine deutlich höhere Personenanzahl als auf der Bibliotheksseite genannt. Die Frage um welche Projekte es sich dabei handelt (Frage 8) zielte nicht auf konkrete Abteilungen, sondern auf unterschiedliche Aufgabenbereiche ab. Zur Auswahl standen Bibliothekssystem, der Aufbau, bzw. Wartung von

Datenbanken und von Repositorien und sonstiges. Am häufigsten wurden Bibliothekssysteme genannt, gefolgt von der Wartung von Repositorien und von Datenbanken. Unter sonstiges wurden unter anderem Digitalisierung, der Umgang mit elektronischen Zeitschriften, das Intranet, Schnittstellen, Ticketsysteme angegeben.
Als Ergebnis dieses Fragenblocks ist festzuhalten, dass es für wissenschaftliche Bibliotheken heute Alltag ist, sehr eng und häufig in IT-Projekten involviert zu sein und zwar in zahlreichen, wenn nicht inzwischen in sämtlichen Bereichen der Bibliothek.

4.3.2 Der Umgang mit Anforderungen im technischen Bereich

Der nächste Fragenblock befasste sich konkreter mit dem Umgang mit Anforderungen im technischen Bereich. Es stellte sich heraus, dass in fast allen Fällen regelmäßige Treffen mit der IT-Abteilung organisiert werden, wo über Anforderungen gesprochen wird. (Frage 9) Die Frage nach der Zusammensetzung des Projektmanagements ergab, dass sich dieses – abhängig von der Größe der jeweiligen Institution – in den meisten Fällen je nach Projekt ändert (Frage 10), dass es in der Hälfte der befragten Institutionen eine klare Rollenverteilung gibt (Frage 11), wobei hier anzumerken ist, dass der Umgang mit Rollen von Abteilung zu Abteilung stark unterschiedlich sein kann. Der Vorteil der persönlichen, bzw. telefonischen Umfrage war, dass auch zusätzliche Kommentare zu den Fragen abgegeben werden konnten. So merkte eine Person bei der Frage „Gibt es klar definierte Rollen?“ an: „Ja, aber die Rollen werden nicht eingehalten“, was darauf schließen lässt, dass das Projektmanagement da und dort noch ausbaufähig ist und dies den Betroffenen auch deutlich bewusst ist.
Wie im theoretischen Teil der Arbeit gezeigt wurde, nimmt professionelles RE viel Zeitressourcen in Anspruch. Die Frage ob für das Management von Anforderungen im Vorfeld Zeit eingeplant wird (Frage 12) verneinte die Hälfte aller Befragten. Für 37% der InterviewpartnerInnen wird genügend Zeit eingeplant und für 13% nur hin und wieder, das ist nicht ausreichend für eine professionelles RE. Günstiger, wenn auch nicht wirklich zufriedenstellend, fielen die Antworten bei der Frage aus, ob die Zuständigkeiten im Projekt schon zu Beginn festgelegt werden würden (Frage 13). Hier antworteten mehr als drei Viertel mit „ja“. Auch eine vertiefende Frage nach der Transparenz der Zuständigkeiten wurde ähnlich beantwortet. Wie oben gezeigt, ist die

Dokumentation ein wesentlicher Bestandteil von RE. Bei der Frage ob der Prozess rund um die Anforderungen dokumentiert wird (Frage 14) antworteten 65 % der Befragten mit „ja“, 17% mit „meistens“, 6% mit „nur minimal“ und immerhin 12% mit „nein“, was auf einen deutlichen Mangel in der Projektplanung hinweist. Wenn dokumentiert wird, wird diese Aufgabe je nach Institution von unterschiedlichen Personen wahrgenommen. (Frage 14a) In 36% der Fälle dokumentieren Bibliothek und IT-Abteilung/IT-Firma gemeinsam, jeweils 14% nur die Bibliothek oder die IT-Abteilung/IT-Firma. In 36% der Fälle dokumentieren auch andere Personen den Prozess, wie beispielsweise andere Abteilungen, wie etwa die Controlling-Abteilung. Die Frage, wie dokumentiert wird, ergab, dass in den meisten Fällen eine Word-Datei erstellt wird, an zweiter Stelle wurde ein Wiki genannt, bzw. eine Excel-Datei. (Frage 14b) Nur sehr wenige der Befragten setzten eine spezielle Software ein. 60% der befragten Personen gab an, dass diese Dokumentation auch inhaltlich überprüft wird, 33% verneinten dies. (Frage 14c). Auch hier ist also noch Verbesserungspotential vorhanden. Wenn Dokumentationen geprüft werden, wird dies in den meisten Fällen gemeinsam von allen Betroffenen durchgeführt (Frage 14d). Ob die Dokumentationen regelmäßig aktualisiert werden, war den meisten, nämlich 54% unbekannt (Frage 14 e) Nur 31% bejahten die Frage, der Rest gab an, dass in Projekten unterschiedlich mit der Aktualisierung von Dokumentation umgegangen wird. Daraus kann man schließen, dass die Transparenz in den Projekten zu wünschen übrig lässt.

Wie in der Theorie gezeigt werden konnte, ist die Erstellung der Anforderungen für ein gelungenes RE ausschlaggebend. Auf die offen gestellte Frage, wer die Anforderungen erstellt (Frage 15) antworteten die meisten, dass diese gemeinsam von der Bibliothek und der IT-Abteilung, bzw. der IT-Firma erstellt werden und zwar oft in einem ersten Schritt von der Bibliothek alleine und erst in einem zweiten Schritt gemeinsam mit den TechnikerInnen. Für das Eruieren von Anforderungen (Frage 16) gab es in 63% der Fälle noch keine Workflows. Einige wiesen im Gespräch darauf hin, dass Anforderungen eher unkoordiniert verfasst werden. Nur knapp ein Drittel der Befragten gab an, dass für das Eruieren der Anforderungen bestimmte Methoden verwendet werden würden. (Frage 16a) Nur 79% der Interviewten bestätigten, dass die Anforderungen untereinander abgestimmt werden, woraus man schließen kann, dass

eventuell widersprüchliche Anforderungen weitergegeben werden. (Frage 17) Wenn es eine Abstimmung gibt, erfolgt diese meistens gemeinsam mit der IT-Abteilung, bzw. der IT-Firma. (Frage 18). Die Dokumentation der Anforderungen selbst erfolgte meist in einem Textdokument, an zweiter Stelle standen Wikis oder andere Tools, auch UML wurde vereinzelt genannt, allerdings fand in wenigen Fällen auch keine Dokumentation der Anforderungen statt, was sich laut den RE-Regeln sehr ungünstig auf den weiteren Projektverlauf auswirkt. (Frage 19)
Zusammenfassend zu diesem Frageblock lässt sich sagen, dass sich der Umgang mit Anforderungen im technischen Bereich recht unterschiedlich gestaltet, in einigen Fällen schon weitgehend professionell abläuft, aber bei den meisten Institutionen noch stark verbessert werden könnte.

4.3.3 Das Wissen und der Einsatz von RE an wissenschaftlichen Bibliotheken

Nachdem bereits zu Beginn der Befragung geprüft wurde, ob der Begriff RE überhaupt bekannt ist (Frage 1) und nur ein Drittel angab, damit nichts zu verbinden, wurde im nächsten Abschnitt dieses Thema vertieft. Die Frage, ob Standards des RE bekannt seien, wurde nur von 22% aller Befragten bejaht. 11% wusste ein wenig darüber und 67% sind Standards unbekannt. (Frage 20) Von jenen, die den Begriff kannten, gaben 93% an, dass diese in ihrer jeweiligen Institution nicht eingesetzt werden würden. Das bestätigt, die eingangs formulierte These, dass RE aktuell in wissenschaftlichen Bibliotheken noch nicht berücksichtigt wird.
Mehr als Hälfte der InterviewpartnerInnen verneinten die Frage, ob die Anforderungen qualitätsgeprüft werden. (Frage 21) Daraus lässt sich der Schluss ziehen, dass oft unfertige oder fehlerhafte Anforderungen an die TechnikerInnen weitergegeben werden. Wenn eine Qualitätsprüfung stattfindet, wird sie meist im Team vorgenommen, wobei am häufigsten auf Ressourcenplanung und Transparenz geachtet wird. Eine Dokumentation der Qualitätsprüfung erfolgte nur in 50% der Fälle.
Ein Viertel der Befragten gab an, dass es keinen Feedbackprozess hinsichtlich der Anforderungen gibt. (Frage 22). Wenn Anforderungen als unzureichend angesehen werden, startet in den meisten Institutionen der Prozess erneut, einige nannten explizit ein bereits existierendes Changemanagement. (Frage 23) In einigen Fällen war den

ProjektpartnerInnen der Umgang mit unzureichenden Anforderungen nicht bekannt, was auf mangelnde Transparenz schließen lässt, bzw. ist das Problem noch nie aufgetreten. Eine befragte Person merkte hier treffend an, dass es zunächst einmal wichtig ist, überhaupt zu erkennen, dass die Anforderung unzureichend ist. Mehrheitlich haben alle am Projekt beteiligten Personen Zugriff auf die erstellen Anforderungen. (Frage 24) Von einigen wurde auch angegeben, dass selbst Personen, die nichts mit dem Projekt zu tun haben, die Anforderungen einsehen können, was eine sinnvolle und erfreuliche Offenheit beweist.

Auf die Frage, wie reagiert wird, wenn im Nachhinein neue Anforderungen gestellt, bzw. die Anforderungen geändert werden (Frage 25), antworteten fast alle, dass dies ein alltäglicher Fall sei. An dieser Stelle verwiesen einige auf die Vorteile des agilen Arbeitens. Diese offene Frage führte auch dazu, dass auf derzeitige Mängel in der eigenen Institution hingewiesen wurde, auf unkoordiniertes Vorgehen, bzw. auf fehlendes Changemanagement. Dies wiederum zeigt, dass vielen diese Unzulänglichkeiten bewusst sind.

Sämtliche Befragte gaben an, dass die Umsetzung der Anforderungen überprüft werden würde, in den meisten Fällen von zuständigen BibliotheksmitarbeiterInnen, manchmal auch von der Bibliothek gemeinsam mit der IT-Abteilung, bzw. IT-Firma. (Frage 26) In 61% der Fälle wurde dieser Umsetzungsprozess dokumentiert, in 22% nur zum Teil. (Frage 27) Immerhin gaben 11% an, dass der Prozess überhaupt nicht dokumentiert wird, was sowohl für das Projekt als auch für die mögliche Wiederverwendung von Anforderungen kontraproduktiv ist.

Auch die Antworten auf diesen Fragenblock zeigten einerseits die Unterschiede in den einzelnen Institutionen, so wird zum Teil sehr professionell mit Anforderungen umgegangen, während es an anderen Einrichtungen noch am Wissen und am Umsetzen mangelt, andererseits sieht man auch, dass den meisten Befragten diese Mängel durchaus bewusst sind. Die Einführung von RE würde hier sicherlich auf fruchtbaren Boden fallen.

4.3.4 Subjektiver Eindruck vom Umgang mit Anforderungen

Bei den letzten Fragen war vor allem der subjektive Eindruck der Befragten zum Thema „Umgang mit Anforderungen“ gefragt. Genau die Hälfte gab an, mit der derzeitigen Situation, wie an ihrer Institution mit Anforderungen umgegangen wird, zufrieden zu sein. (Frage 28) Jene, die nicht oder nur teilweise zufrieden waren, gaben meist auch den Grund dafür an. So wurde beispielsweise eine mangelnde Kultur in Sachen Projektmanagement kritisiert, eine engere Zusammenarbeit, eine klare Rollenzuweisung und ein strukturierteres Arbeiten wurden gefordert. Von den zufriedenen UmfrageteilnehmerInnen waren einige der Meinung, dass sich der Umgang mit Projekten in den letzten Jahren wesentlich verbessert hat. Kein eindeutiges Ergebnis ergab die Frage, ob für die Umsetzung von Projekten genügend Personal und Zeit zur Verfügung steht. Von denen die nicht, oder nicht ganz zufrieden waren, meinte die Hälfte, dass zu wenig Personal vorhanden wäre, die andere Hälfte merkte an, dass es an Zeit mangle. (Frage 29) Beides stört den Prozess des Anforderungsmanagements. Auf die vertiefende Frage, wo Verbesserungsmöglichkeiten gesehen werden würden, antworteten die meisten, dass das Personal eine bessere Ausbildung im Bereich des Aufbaus von e-Infrastrukturen benötigen würde, dass mehr Personal nötig wäre, eine raschere Umsetzung von Projekte, klarere Workflows, eine besser funktionierende Kommunikation und mehr Transparenz. (Frage 30) Ein Management auch bei kleineren Projekten wurde nicht nur positiv bewertet. Es sprachen sich zwar 45% dafür aus, 44% waren jedoch der Meinung, dass es nicht unbedingt notwendig sei und 11% sahen keinen Sinn auch bei kleinen Projekten ein Management einzuführen. (Frage 31) Eine offensichtliche Sorge war, dass sich damit alles verkomplizieren würde und durch den Mehraufwand Projekte ineffizient werden könnten.

In 33% aller befragten Institutionen ist RE bereits ein Thema, wenn es auch nicht konsequent umgesetzt werden würde. (Frage 32) Einige der Befragten gaben an, dass dies vor allem im Umgang mit den externen Partnern wichtig sei und auch der Absicherung diene. Ausgebildete RE gibt es derzeit jedoch kaum an den Institutionen. (Frage 33). Unter den ausgebildeten Personen besaßen zwei ein entsprechendes Zertifikat. Keine der befragten Institutionen bietet derzeit Weiterbildungsmöglichkeiten im Bereich RE an. (Frage 34)

72 % der InterviewpartnerInnen sehen in der Zukunft Bedarf an RE. (Frage 35) Als Gründe wurden die vermehrte Zusammenarbeit von Bibliotheken mit dem IT-Bereich und die erstrebte Professionalisierung von Abläufen genannt. Der Rest, der sich nicht sicher war, ob Bedarf an RE besteh, äußerte Bedenken hinsichtlich des Kosten-Nutzen-Faktors, da die Ausbildung Zeit und Geld kostet und es eventuell zu einem erhöhten Verwaltungsaufwand kommen könnte. Ebenfalls genannt wurde, dass für RE vor allem die richtige Personen eingesetzt werden müssten, da die Persönlichkeit und die kommunikativen Fähigkeiten für die angestrebte Wirkung ausschlaggebend wären. Eine Einführung in das Thema RE, eventuell auch auf Bibliothekartagen oder Tagungen, bzw. Workshops oder Vorträge an den einzelnen Bibliotheken könnte hier Klarheit bringen und im besten Fall dazu führen, dass sich einige BibliothekarInnnen intensiver damit beschäftigen. So wie es heute selbstverständlich ist, wenigstens die Grundzüge des Projektmanagements zu verstehen, könnte in Zukunft auch RE zum Basiswissen gehören.

5. Zusammenfassung der Ergebnisse

Die Literaturrecherche hat ergeben, dass bereits einige Handbücher am Markt sind, die sich auch für den Bibliotheksbereich gut für einen Einstieg in das Thema RE eignen. Es hat sich aber auch gezeigt, dass es sich um einen sehr komplexes Bereich handelt, der bis heute fast nur im IT-Bereich bekannt ist und dort auch selbstverständlich Anwendung findet. Sich das Wissen über RE anzueignen kann sowohl über Literatur, aber auch über diverse Kurse, die in Weiterbildungsinstitutionen angeboten werden, erfolgen. Auch wenn in der Wirtschaft RE zum Teil schon fixer Bestandteil von Unternehmen ist und Jobprofile entwickelt wurden, wird es im bibliothekarischen Umfeld, zumindest in Österreich, noch nicht in der Ausbildung angeboten. Weder im Grundlehrgang des Universitätslehrganges „Library and Information Studies“, noch beim Masterlehrgang findet RE auch nur Erwähnung. Das ist umso bedauerlicher, als in der Zukunft nahezu alle BibliothekarInnen mit IT-Projekten beschäftigt sein werden, wie auch die Umfrage deutlich gezeigt hat.

Von vielen wird die Zusammenarbeit zwischen IT und Bibliothek oft als nicht sehr kompetent und zielführend bewertet. Das liegt oft auch daran, dass zu wenig Zeit, Geld und Personal eingeplant wird und dass deshalb das gesamte Projektmanagement nicht optimal erfolgen kann. Der Umgang mit Anforderungen wird einerseits aus diesen Gründen nicht immer als zufriedenstellend erlebt, andererseits fehlt aber auch oft das Wissen über RE, es gibt derzeit kaum Personen, die bibliothekarisches Wissen und IT-Wissen besitzen und gleichzeitig auch noch über die erforderliche kommunikative Kompetenz verfügen um als RE agieren zu können.

Für die Zukunft scheint es ratsam zu sein, den BibliotheksmitarbeiterInnen Zugang zu diesem Wissen in speziellen Kursen zu ermöglichen, bzw. die Grundzüge des RE auch in den Grundlehrgang „Library and Information Studies“ einzuplanen. Ein Modul mit RE kann selbstverständlich eine fundierte Ausbildung nicht ersetzen, jedoch das nötige Bewusstsein schaffen um in der eigenen Arbeitsumgebung den Umgang mit Anforderungen zu optimieren. Viele der in den Handbüchern beschriebenen Schritte werden auch jetzt schon im Arbeitsalltag intuitiv ausgeführt, mit ein wenig Übung und den richtigen Tools lassen sich diese Ansätze verbessern und vor allem bewusst machen. Eine Möglichkeit wäre etwa im Modul RE Rollenspiele einzuplanen, die den

Umgang mit Anforderungen im Spannungsfeld User, TechnikerInnen und AuftraggeberInnen einüben.[29] Auf lange Sicht gesehen ersparen sich Bibliotheken, wenn sie gut ausgebildetes Personal zur Verfügung haben, Umwege, Kosten und Unzufriedenheit bei ihren BenützerInnen.
Darüber hinaus wäre es auch empfehlenswert, wenn die jeweiligen Trägerinstitutionen ebenfalls Weiterbildungsmöglichkeiten im Bereich des RE anbieten würden, für all jene, die schon mitten im Berufsalltag stehen und die Bibliotheksausbildung schon hinter sich haben.
An dieser Stelle soll eine (nicht vollständige) Auflistung von Hinweisen dabei helfen Stolpersteine zu erkennen und eine erste Orientierung im Bereich RE zu erhalten.

- Beim Erstellen von Anforderungen sollte man sich nicht nur auf das Wissen von IT-ExpertInnen verlassen. Vielfach legt man sich nicht genau fest, um später keine Änderungen mehr machen zu müssen. IT-ExpertInnen können aber nicht sämtliche Bedürfnisse ihrer Kunden kennen. Auch auf der Seite des Auftragnehmers sollte man sich nicht ausschließlich auf die Wünsche des Kunden verlassen. Auch hier fehlt vielfach Wissen, nämlich das Wissen um die technischen Möglichkeiten, die vielleicht schon existieren.
- Keine Angst vor Änderungen! Anforderungen ändern sich während eines Projekts, das agile Arbeiten im IT-Bereich hat auf diese Tatsache bereits reagiert. Wichtig ist, kompetent und sorgsam mit diesen Änderungswünschen umzugehen.
- Alles sollte so gut und lückenlos wie möglich dokumentiert werden. So werden Anforderungswünsche für alle nachvollziehbar, Anforderungen können eventuell wiederverwendet werden und Unklarheiten wie Anforderungen entstanden sind, werden vermieden.

[29] Siehe auch Birkner, Michael; Susanne Blumesberger: Requirements-Management. Vortragsfolien im Rahmen des Fortbildungsseminars des Projekts e-infrastructures Austria am 7. Juni 2016. http://phaidra.univie.ac.at/o:441305

- Rasch Ergebnisse erzielen zu wollen ist nicht immer der beste Weg. Eine gut strukturierte Planung führt kostensparender und oft rascher zum gewünschten Erfolg als vorschnelles Umsetzen von halbfertigen Anforderungen.
- Alle betroffenen Personen, also alle Stakeholder, müssen von Beginn an in den Prozess eingebunden sein. Werden Entscheidungsträger nicht oder zu spät eingebunden, kann das Projekt scheitern oder sich zumindest das Arbeitsklima verschlechtern.
- Bei der Analyse von Anforderungen darf nicht an Kosten gespart werden. Anforderungen sollten nur dann, wenn sie vollständig vorliegen sowie klar und deutlich formuliert sind, umgesetzt werden.
- Es dürfen keine Anforderungen, scheinen sie auch auf den ersten Blick banal zu sein, fehlen. Auch scheinbar Selbstverständliches, wie beispielsweise, dass in einer Datenbank nach bestimmten Kriterien gesucht werden kann, muss explizit formuliert werden.
- Anforderungen dürfen sich nicht gegenseitig widersprechen. Das Gesamtbild darf nie aus den Augen verloren werden, sonst schleichen sich rasch Widersprüchlichkeiten ein.
- Änderungen dürfen nicht schleichend passieren. Sollen Anforderungen geändert werden, muss dieser Prozess transparent gehalten werden, alle Phasen des RE durchlaufen und vor allem auch dokumentiert werden.
- Wichtig ist, dass alle Beteiligten nicht nur zu Projektbeginn sondern während der gesamten Laufzeit gut informiert sind und sich niemand übergangen fühlt. Auch sollten die Key-User einen Gesamtüberblick haben um evtl. auf Wünsche von anderen Personen angemessen reagieren zu können. Regelmäßige Informationsveranstaltungen beugen hier Problemen vor, evtl. kann auch ein so genanntes Sounding-Board helfen, zusätzlich Sicherheit zu geben, in dem auf atmosphärische und emotionale Rückmeldungen geachtet wird.
- Werden neue Rollen eingeführt, so muss dies an alle kommuniziert werden um die Beteiligten nicht mit neuen Regelungen zu überfordern.
- Der Kontakt zu den Stakeholdern darf nie abbrechen.
- Der/die RE sollte bei Bedarf Coaching erhalten.

- Die Rollen und Abläufe müssen allen vertraut sein
- Ein laufendes gepflegtes Glossar, das allen zugänglich ist, verhindert, dass Begriffe unterschiedlich verwendet werden und dadurch Missverständnisse entstehen.
- Anforderungsmanagement sollte nicht jedes Mal anders ablaufen um die Beteiligten nicht zu verunsichern, d.h. es ist sinnvoll, sich gewisse Workflows zu überlegen und diese beim nächsten Mal erneut aufzugreifen.
- Sinnvoll ist auch, wenn allen Beteiligten die Grundlagen von RE bekannt sind.
- Bereits Bekanntes und Bewährtes sollte wiederverwendet werden.

6. Glossar

Anforderung	Eine Anforderung ist 1. Eine Eigenschaft oder Fähigkeit, die von einem Benutzer (Person oder System) zur Lösung eines Problems oder zur Erreichung eines Ziels benötigt wird. 2. Eine Eigenschaft oder Fähigkeit, die ein System oder Teilsystem erfüllen oder besitzen muss, um einen Vertrag, eine Norm, eine Spezifikation oder andere, formell vorgegebene Dokumente zu erfüllen 3. Eine dokumentierte Repräsentation einer Eigenschaft oder Fähigkeit gemäß (1) oder (2).	Rupp/Sophisten 2014, 14
Anforderungskonfiguration	besteht aus einer Menge an Anforderungen mit der zusätzlichen Bedingung, dass jede der ausgewählten Anforderungen in genau einer Version in der Anforderungskonfiguration enthalten ist.	Pohl/Rupp 2015, 139
Anforderungsspezifikation	eine präzise Beschreibung des Funktions- und Leistungsumfangs an ein System zur Lösung eines Problems unter Berücksichtigung aller wichtigen Umgebungsbedingungen des Einsatzgebietes und der Softwareherstellung.	Partsch 1991, 29f
Attribut	eine Eigenschaft, bzw. Zustandsgröße von Objekten	Neumann 1998, 419
Baseline (Basislinie)	umfasst eine Konfiguration aller Anforderungen für ein Release	Pohl/Rupp 2014, 455
Change-Request	Änderungsanforderung, Anpassungen an Leistungen	Klett/Hilberg 2010, 421
Incident -Management	Störungsmanagement, auch Kundendienst, 1 st-Level-Support, Hotline, Service Desk, kümmert sich um Anfragen der User	Rupp/Sophisten 2014, 448
Kanban	Ein Modell der Softwareentwicklung, bei der parallele Workflows vermindert werden um Engpässe rasch sichtbar machen zu können.	Das Kanban-Plakat[30]
Klassendiagramm	Ist ein Diagramm, das eine bestimmte Menge von abstrakten Modellelementen (Klassen und Typen), ihren Inhalt und	Neumann 1998, 424

[30] http://www.kanban-plakat.de/ (abgerufen am 19.2.2017)

	ihre statischen Beziehungen sichtbar macht	
Lastenheft	Ein Lastenheft ist die vom Auftraggeber festgelegte Gesamtheit aller Forderungen innerhalb eines Auftrages.	
Objekt	Eine in der Datenverarbeiten-gerechte Repräsentation - Für einen realen Gegenstand - Für eine Person bzw. für die Rolle einer Person - Für ein abstraktes Konzept, z.B. ein Konto, das zur Verwaltung von Geldwerten eingesetzt wird - Für einen Prozess oder Vorgang - Für ein gedankliches Konzept zur Unterstützung eines Vorgangs oder zur Erfüllung von irgendwelchen Anforderungen an ein Software-System	Neumann 1998, 2f
Pflichtenheft	Ein Pflichtenheft wird vom Auftragnehmer erarbeitet und beschreibt das Realisierungsvorhaben anhand des Lastenheftes, es zeigt also wie die Umsetzung stattfinden wird.	
Requirements creeping	Schleichender Funktionszuwachs. Ziele und Anforderungen werden ohne Einbindung der Stakeholder und ohne Management verändert.	ProjektMagazin. Das Fachportal für Projektmanagement[31]
Requirements-Engineering (RE)	Unter RE wird der Weg von der Projektidee bis zu einem vollständigen Satz von Anforderungen verstanden.	Moser 2007, 11
Requirements-Management (RM)	RM umfasst die Maßnahmen, welche die Anforderungsanalyse und die weitere Verwendung der Anforderungen unterstützen.	Rupp/Sophisten 2014, 368
Scrum	Scrum stammt aus der agilen Softwareentwicklung und ist seit 1995 bekannt. Man geht dabei davon aus, dass die Entwicklung eines Systems nicht vorhersehbar ist und deshalb in kleine Schritte zerlegt werden muss. Der Begriff stammt ursprünglich vom Gedränge im Rugby (englisch Scrum): Kleine Teams arbeiten sehr effektiv und selbstbestimmt, wobei nur die Richtung von außen angegeben wird, nicht aber die Art und Weise der Umsetzung. Statt	Rupp/Sophisten 2014, 64

[31] https://www.projektmagazin.de/glossarterm/schleichender-funktionszuwachs (abgerufen am 19.2.2017)

	Arbeitsanweisungen erhalten die Teams Freiheit um ihr Wissen und Können einsetzen zu können.	
Sequenzdiagramm	betont die zeitliche Sequenz der Interaktionen von Objekten	Neumann 1998, 430
Spezifikation	eine verständliche, eindeutige, widerspruchsfreie, möglichst formale Beschreibung eines bestimmten Sachverhalts im Zusammenhang mit der Entwicklung eines Software-Systems	Neumann 1998, 431
Stakeholder	alle Personen und Institutionen, die von der Systementwicklung und natürlich auch vom Einsatz und Betrieb des Systems oder Produkts betroffen sind. Beispiele für Stakeholder sind Nutzer des Systems, Betreiber des Systems, Entwickler, Architekten, Auftraggeber und Tester – also auch Personen, die nicht in der Entwicklung mitwirken.	Rupp/Sophisten 2014, 79
Statecharts	Statecharts sind ein Bestandteil der UML und dienen der Beschreibung des Systemverhaltens. Sie basieren auf Zustandsautomaten, die das reaktive Verhalten des Systems beziehungsweise einer Komponente darstellen.	http://www.re-wissen.de/opencms/Wissen/Techniken/State_Charts.html
System	umfasst die Gesamtheit der zur Erfüllung eines Aufgabenkomplexes erforderlichen Hardware- und Software-Komponenten.	Neumann 1998, 432
Traceability	Traceability ist die Fähigkeit, Verbindungen und Abhängigkeiten zwischen Informationen, welche während der Analyse, Entwicklung, Wartung und Weiterentwicklung eines Systems anfallen, jederzeit nachvollziehen zu können.	Rupp/Sophisten 2014, 421
Unified Modeling Language (UML)	Grafische Notation mit einfacher Syntax und weitgehend formal definierter Semantik. Die UML bildet das einheitliche und eindeutige Mittel der Verständigung von Anwendern und Entwicklern	Neumann 1998, V
Use Case	wird auch Anwendungsfall genannt. Ein Use Case beschreibt Funktionen, die ein Akteur mithilfe des Systems ausführen möchte. Daraus abgeleitete Anforderungen werden als „funktionale Anforderungen“ bezeichnet.	Zastrow/Wagner 2016, 2
User Storys	User Storys beschreiben relevante Funktionalitäten und bestehen aus einer	Rupp/Sophisten 2014, 256

	schriftlichen Beschreibung, Gespräche über die Funktionalität und Akzeptanzkriterien, die festlegen, wann eine User-Story umgesetzt ist.	
Validation	Die gezielte und systematische Prüfung von Analyseergebnissen. Die Prüfung soll sicherstellen, daß letzten Endes da Ergebnis der Entwicklung in dem Maße nutzbar ist, wie es der Auftraggeber bzw. der spätere Nutzer erwartet. Die typische Fragestellung lautet deshalb: „Are we building the right product?“	Neumann 1998, 265
Verifikation	Die gezielte und systematische Überprüfung der aktuellen Entwurfsergebnissen, die sicherstellen soll, daß das geplante Software-System mit dem geforderten Verhalten ohne Einschränkungen realisierbar ist. Die typische Fragestellung lautet deshalb: „Are we building the product right?“	Neumann 1998, 326

7. Abkürzungen

CCB	Change Control Board
CPRE	Certified Professional for Requirements Engineering
ER	ER-Modelle (Entity-Relationship-Modelle)
IREB	International Requirements Engineering Board
RE	Requirements-Engineering/Requirements-Engineer
RM	Requirements-Management/Requirements-ManagerIn
UML	Unified Modeling Language

8. Literatur

Bandler, Richard; John Grinder (Bandler/Grinder 2012): Metasprache und Psychotherapie. Die Struktur der Magie I. Ein Buch über Sprache und Therapie. 12., neu übersetzte Auflage. Paderborn: Junfermann Verlag 2011.

Blumesberger, Susanne; Michael Birkner: Requirements-Management. Vortragsfolien im Rahmen des Fortbildungsseminars des Projekts e-infrastructures Austria am 7. Juni 2016. http://phaidra.univie.ac.at/o:441305

Blumesberger, Susanne; Raman Ganguly: Requirements-Management aus technischer und nichttechnischer Sicht. Vortragsfolien im Rahmen des 32. Österreichischen Bibliothekartages in Wien 2015. urn:nbn:de:0290-opus4-21005

Dreyfus, Hubert; Stuart Dreyfus: Mind over Machine. The Power of Human Intuition and Expertise in the Era oft he Computer. New York: Fireside Books 2000.

Duwe, Peter; Michèle Mégroz (Duwe/Mégroz 2009): Methodeneinsatz alleine ist nicht genug. Der Anforderungsmanager als Vermittler zwischen den Welten. In: ProjektMagazin. Das Fachportal für Projektmanagement. Ausgabe 4/2009.

Heche, Dirk; Oliver Ollech (Heche/Ollech 2006): Projektbegleitendes Anforderungsmanagement: unnötige Kosten und Ärger vermeiden. In: ProjektMagazin. Das Fachportal für Projektmanagement. Ausgabe 21/2006.

IEEE Standard Glossary of Software Engineering Terminology. New York: The Institute of Electrical and Electronics Engineers. 28. September 1990.

Institute of Electric and Electronic Engineers: IEEE Recommendes Practice for Software Requirements Specifications (IEEE Std 830-1998). IEEE Computer Society. New York 1998.

Klett, Detlef; Söntje Julia Hilberg (Klett/Hilberg 2010): Die neue DIN-Spezifikation für das Outsourcing–Inhalt und praktische Anwendung. In: Computer und Recht. Zeitschrift für die Praxis des Rechts für Informationstechnologien, Heft 7 2010, S. 417-421.

Kulke, Matthias (Kulke 2014): Wie Sie die natürliche Sprache bändigen. Die SOPHIST-Satzschablone: Funktionale Anforderungen präzise formulieren. In: ProjektMagazin. Das Fachportal für Projektmanagement. Ausgabe 15/2014.

Moser, Julia (Moser 2007): Integration von Anforderungsverhandlungen in den Rescue Requirements Engineering Prozess. Masterarbeit, Universität Wien 2007.

Neumann, Horst A. (Neumann 1998): Objektorientierte Softwareentwicklung mit der Unified Modeling Language (UML). München, Wien: Carl Hanser Verlag 1998.

Niebisch, Thomas (Niebisch 2013): Anforderungsmanagement in sieben Tagen. Der Weg vom Wunsch zur Konzeption. Berlin, Heidelberg: Springer Verlag 2013. DOI 10.1007/978-3-642-34857-0_1.

Partsch, Helmut (Partsch 1991): Requirements Engineering. München, Wien: R. Oldenbourg Verlag 1991 (Handbuch der Informatik; Bd. 5.5.).

Plattner, Hasso/Christoph Meinel/Ulrich Weinberg (Plattner, Meinel, Weinberg 2009): Design Thinking. Innovation lernen – Ideenwelten öffnen. München: by mi-Wirtschaftsbuch, FinanzBuch Verlag GmbH 2009.

Pohl, Klaus; Chris Rupp (Pohl/Rupp 2015): Basiswissen Requirements Engineering. Aus- und Weiterbildung zum „Certified Professional for Requirements Engineering“. 4. Auflage, Heidelberg: dpunkt.verlag GmbH 2015.

Rosenberg, Marshall B.: Gewaltfreie Kommunikation. Eine Sprache des Lebens. Paderborn: Junfermann 2011.

Rupp, Chris & die SOPHISten (Rupp/Sophisten 2014): Requirements-Engineering und –Management. Aus der Praxis von klassisch bis agil. 6., aktualisierte und erweiterte Auflage, München: Carl Hanser Verlag 2014.

Rupp, Chris; Christof Geißel (Rupp/Geißel 2007): Nichtfunktionale Anforderungen strukturiert erfassen und wieder verwenden. In: ProjektMagazin. Das Fachportal für Projektmanagement. Ausgabe 8/2007.

Rupp, Christine; Stefan Queins & die SOPHISten (Rupp/Queins/Sophisten 2012): UML2 glasklar. Praxiswissen für die UML-Modellierung. 4. akt. und erw. Auflage, München: Carl Hanser 2012.

Satir, Virginia: Das Satir-Modell: Familie und ihre Erweiterung. 4. Aufl. Paderborn: Junfermann 2007.

Seissl, Maria: (Seissl 2015): Viel mehr als früher! Das Aufgaben- und Serviceportfolio der Universitätsbibliothek der Universität Wien. In: Niedermair, Klaus, Dietmar Schuler (Hg.): Die Bibliothek in der Zukunft Regional – Global: Lesen, Studieren und Forschen im Wandel. Festschrift für Hofrat Dr. Martin Wieser anlässlich seiner Versetzung in den Ruhestand. Universität Innsbruck: innsbruck university press, 2015, S. 73-78.

Smith R. Craig: Using a Quality Model Framework to Strengthen the Requirements Bridge. In: Proceedings of IEEE International Symposium on Requirements Engineering: RE '98, 0 1998, pp.118-125. DOI: 10.1109/ICRE.1998.667816

V-Modell XT. Das deutsche Referenzmodell für Systementwicklungsprojekte Version: 2.0. (V-Modell XT 2015) Herausgegeben vom Verein zur Weiterentwicklung des V-Modell XT e.V. (Weit e.V.) München 2015.
Zastrow, Bettina; Elisabeth Wagner (Zastrow/Wagner 2014): Anforderungen an Softwareprodukte klar definieren. Requirements Engineering für Projektleiter Teil 1: Leistungsumfang und Rahmenbedingungen festlegen. In: ProjektMagazin. Das Fachportal für Projektmanagement. Ausgabe 23/2014.

Zastrow, Bettina; Elisabeth Wagner (Zastrow/Wagner 2015): Anforderungen an Softwareprodukte klar definieren. Requirements Engineering für Projektleiter Teil 2: Mit Use Cases den Funktionsumfang beschreiben. In: ProjektMagazin. Das Fachportal für Projektmanagement. Ausgabe 24/2015.

Zastrow, Bettina; Elisabeth Wagner (Zastrow/Wagner 2016): Anforderungen an Softwareprodukte klar definieren Requirements Engineering für Projektleiter Teil 3: Anforderungen für die Software-Entwicklung ableiten. In: ProjektMagazin. Das Fachportal für Projektmanagement. Ausgabe 1/2016.

Internetquellen (alle zuletzt abgerufen am 19.2.2017)

Agile Alliance. Glossary
https://www.agilealliance.org/glossary/invest/

Atlassian. Das Product Backlog. Die ultimative To-Do-Liste
https://de.atlassian.com/agile/backlogs

Bugzilla
https://www.bugzilla.org/

Bundesministerium des Innern. V-Modell XT
http://www.cio.bund.de/Web/DE/Architekturen-und-Standards/V-Modell-XT/vmodell_xt_node.html

Gabler Wirtschaftslexikon
http://wirtschaftslexikon.gabler.de/

IBM Rational Software
http://www.ibm.com/software/rational

INCOSE
http://www.incose.org/

IREB
https://www.ireb.org/de/

ISO 25000
http://iso25000.com/index.php/en/iso-25000-standards

IT-Wissen. Das große Online-Lexikon für Informationstechnologie
http://www.itwissen.info/definition/lexikon/FURPS-funcionality-usability-reliability-performance-supportability.html

Kanban
http://www.kanban-plakat.de/

Karriere.at
www.karriere.at

Manifest für Agile Softwareentwicklung
http://agilemanifesto.org/iso/de/manifesto.html

mediamanual archiv
https://www.mediamanual.at/mediamanual/workshop/kommunikation/bedeutung/modell01.php

Prinzipien hinter dem Agilen Manifest
http://agilemanifesto.org/iso/de/principles.html

ProjektMagazin. Das Fachportal für Projektmanagement
https://www.projektmagazin.de/

re-wissen.de
http://www.re-wissen.de/opencms/Wissen/Techniken/State_Charts.html

Schulz von Thun. Institution für Kommunikation
http://www.schulz-von-thun.de/index.php?article_id=71

Sophist
https://www.sophist.de/start/

SourceForge
https://sourceforge.net/about

StepStone
https://www.stepstone.at

TU Graz online
https://online.tugraz.at/tug_online/fdb_detail.ansicht?cvfanr=F27146&cvorgnr=37&sprache=2

Universität Ulm, Institut für Softwaretechnik und Programmiersprachen
https://www.uni-ulm.de/in/pm/lehre/ss2015/re/

Volere. Requirements Resources
http://www.volere.co.uk/

WIFI Kursangebot
http://www.wifiwien.at/eshop/bbdetails.aspx?bbnr=18267x