Technische Hochschule
Brandenburg
University of
Applied Sciences

Vorgehensmodell für CMS-Individualentwicklung in einer Webagentur am Beispiel von TYPO3-Extensions

Julian Fastnacht

Fachbereich Wirtschaft

Technische Hochschule Brandenburg

Masterarbeit
zur Erlangung des Akademischen Grades
Master of Science

27. Februar 2017

Vorgehensmodell für CMS-Individualentwicklung in einer Webagentur am Beispiel von TYPO3-Extensions

Masterarbeit zur Erlangung des Akademischen Grades Master of Science

Autor

Julian Fastnacht, geboren am 21.08.1990, Neu Kaliß, Deutschland
Studiengang Wirtschaftsinformatik
Fachbereich Wirtschaft
Technische Hochschule Brandenburg

Gutachter

Prof. Dr. Vera G. Meister, Technische Hochschule Brandenburg
Dipl.-Inf. (FH) Sebastian Kreideweiß, CPS-IT GmbH

Abgegeben am 27.02.2017, Brandenburg an der Havel

Brandenburg an der Havel, den 27. Februar 2017

Ich, Julian Fastnacht, Student im Studiengang Wirtschaftsinformatik der Technischen Hochschule Brandenburg, versichere an Eides statt, dass die vorliegende Masterarbeit selbstständig verfasst und nicht mit anderen als den angegebenen Hilfsmitteln erstellt wurde.

Sie wurde in dieser oder ähnlicher Form noch keiner Prüfungskommission vorgelegt.

Julian Fastnacht

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

CMMI Capability Maturity Model Integration

CMMI-DEV CMMI für Entwicklung

CPS-IT Consulting Piezunka & Schamoni - Information Technologies GmbH

IREB International Requirements Engineering Board

IEEE Institute of Electrical and Electronics Engineers

UML Unified Modeling Language

Abstract

The present work looked at how the process of CMS individual development at the example of TYPO3 extensions can be improved by applying requirements engineering.

First of all, the process was analyzed by means of interviews with employees, existing documents, and experiences from the everyday project work. The areas of process, transfer, project management and maturity were considered. It was found that there is, as yet, no detailed documentation of the existing process. The procedure is carried out in an agile manner, which is not defined in detail, and not according to a well-known software development approach. There are many implicit assumptions, rather than clearly defined requirements, and therefore no atomic requirement documentation. The project manager is responsible for the requirements management, besides his actual activity. A validation and verification of the requirements does not take place in an orderly manner. The different documents and storage locations in the projects, as well as unclear status and responsibilities, lead to uncertainty among the project participants.

Subsequently, solutions and possible solutions were identified for the existing problems. This includes the expansion of the roll concept by a product owner or requirements engineer who assumes the tasks of the requirements management from the project manager and the extension of the task spectrum of the technical project manager to include the tasks of the software engineer for the conceptual preparation and implementation of the business logic. In addition, the transfer of information and storage of files will continue to be directed to Confluence. Finally, the definition of short, medium and long-term necessary adaptations of the processes to meet the requirements of CMMI for Development (CMMI-DEV) in requirements development and requirements management.

1 Einleitung

Die Bereitstellung von Content Management Systemen für den Kunden erfordert oftmals die Erweiterungen um spezielle Kundenwünsche. Auch wenn bereits eine Vielzahl solcher Erweiterungen zur Verfügung stehen, welche sich über die Websites der jeweiligen Hersteller beziehen lassen, erfüllen diese oftmals nicht alle Anforderungen des Kunden und machen eine Eigenentwicklung unumgänglich. Des Weiteren sind die genauen Anforderungen des Kunden zu Beginn oftmals noch nicht klar, was bei einer unstrukturierten Vorgehensweise zu einem langwierigen Prozess von Versuch und Irrtum führt. Diese ständige Anpassung der Anforderungen weniger Erweiterungen kann die tatsächlichen Kosten des Gesamtprojektes und die damit verbundenen erwarteten Gewinne beeinträchtigen, was Projekte dieser Art weniger lukrativ erscheinen lässt. Neben dem Kostenfaktor spielt auch die Zufriedenheit der Mitarbeiter und natürlich des Kunden eine nicht weniger wichtige Rolle. Projekte die nicht abgeschlossen werden können, obwohl sie weit über dem Terminplan liegen, können bei allen Projektbeteiligten zu Unzufriedenheit und Frustration führen. Die Relevanz der Lösung dieses Problems ergibt sich also einerseits aus dem Kostenfaktor und andererseits aus der Zufriedenheit der Projektbeteiligten.

Bereits aus ersten Gesprächen mit Mitarbeitern handelt es sich hierbei um ein akutes Problem beim Auftraggeber, für das schnellstmöglich eine Lösung gefunden werden soll. Mögliche Lösungsansätze kommen dabei aus dem Bereich Requirements Engineering, welches sich mit dem Ermitteln, Beschreiben, Prüfen und Verwalten von Anforderungen befasst. Dabei existiert bisher jedoch keine explizit auf die Entwicklung von TYPO3 Extensions (Erweiterungen des CMS TYPO3) angepasste Variante. Adressaten dieser Lösung wären beim Auftraggeber vor allem Projektmanager und Entwickler, könnten aber auch unabhängig vom Auftraggeber für die TYPO3-Community interessant sein.

Aus der vorliegenden Problemstellung, sowie dem Erkenntnisinteresse ergibt sich folgende Leitfrage: Wie kann durch Requirements Engineering am Beispiel von TYPO3 Extensions der Prozess von Individualentwicklungen innerhalb der Webagentur verbessert werden? Weitere Fragestellungen, die sich an der Leitfrage orientieren, wären: “Was sind abnahmefähige Übergabeformen der Anforderungen und daraus abgeleiteter Aufgaben?”, “Was ist ein geeignetes Projektvorgehensmodell?” und “Welche Auswirkungen haben die angestrebten Veränderungen auf das Projektmanagement?”.

Die Zielstellung der Arbeit ist ein einsatzbereites Vorgehensmodell für die CMS Individualentwicklung am Beispiel von TYPO3 Extensions zu schaffen, welche kostenef-

fizient ist und eine hohe Kunden- und Mitarbeiterzufriedenheit mit sich bringt. Der Anspruch an das Vorgehensmodell ist, dass es sich auf jede Kundendomäne und unabhängig von Spezifika einzelner Systeme anwenden lässt. Für die erfolgreiche Umsetzung soll zunächst eine Literaturrecherche zu den relevanten Themen erfolgen. Anschließend wird durch Experteninterviews mit Projektmanagern und Entwicklern der aktuelle Prozess ermittelt und auf Schwachstellen hinsichtlich der Durchführung von Requirements Engineering analysiert. Parallel wird als Teil der Beschäftigung im Unternehmen die Durchführung von Entwicklungsprojekten begleitet, in denen bereits gewonnene Erkenntnisse und daraus resultierende Verbesserungen angewandt werden. Aus diesen ergibt sich durch stetige Verbesserung das Vorgehensmodell.

2 Hintergrund

Dieses Kapitel behandelt die Hintergründe der Arbeit. Dafür wird zunächst eine Einordnung vorgenommen und anschließend die Grundlagen über Literaturarbeit und Stand des Wissens für das weitere Verständnis geschaffen. Zuletzt werden die verwendeten Methoden dargestellt.

2.1 Einordnung der Arbeit

Diese Arbeit entstand für das Unternehmen Consulting Piezunka & Schamoni - Information Technologies GmbH (CPS-IT) mit Sitz in Berlin. CPS-IT wurde 1998 von Henning Piezunka, Bungo Schamoni und Sascha Böttcher gegründet. Das Unternehmen ist auf den Einsatz des TYPO3 Content Management Systems spezialisiert. Im Jahr 2016 fusionierte das Unternehmen mit der familie redlich AG und verstärkt hierbei vor allem den Firmenteil der Web Entwicklung in der Agentur.

Die familie redlich AG ist eine 2001 durch Andre Redlich gegründete Agentur mit Sitz in Berlin. Mit über 100 Mitarbeitern und verschiedenen Agenturbereichen, deren Kompetenzen sich über Markenarbeit, Kampagnen, Veranstaltungsorganisation bis hin zu Online Projekten erstrecken.

2.2 Literaturanalyse

TYPO 3 ist ein quelloffenes Content Management System, dessen Entwicklung 1997 von Kasper Skårhøj [TYPO3 Association 2015] begonnen wurde und bis zur Erstellung dieser Arbeit weitergeführt wird. Mittlerweile hat sich eine Community um die Weiterentwicklung von TYPO3 und TYPO3 Extensions gebildet. Laut der Seite T3census gibt es weltweit etwa 250.000 Installationen [Krause 2016], die Seite builtWith berichtet sogar von knapp über 400.000 Installationen [BuiltWith Pty Ltd 2017]. Die TYPO3 Association beschreibt TYPO3 als Enterprise Open Source CMS, was eine Ausrichtung vermuten lässt, gibt aber in den Key Features [TYPO3 Association 2017] an, von kleinen Webseiten bis hin zu Unternehmen anwendbar zu sein. Für die Erweiterung des Funktionsumfangs nutzt TYPO3 sogenannte Extensions, welche sich in Core und User Extensions aufteilen, sowie in Plugins, Modules, Services und

Distributions [TYPO3 Documentation Team 2015, S. 7] typisieren lassen. Für die Erstellung der Extensions setzt TYPO3 konzeptionell auf das von Eric Evans entwickelte Domain-Driven Design.

Domain-Driven Design ist eine von Eric Evans entwickelte Methode zur Softwareentwicklung, welche in 3 Punkten zusammengefasst werden kann. Der erste Punkt ist die Fokussierung auf den Kern der Domain. Eine Domain bezeichnet dabei eine Sphäre von Wissen, Einfluss oder Aktivität im Fachgebiet der Software. Der zweite Punkt beschreibt die Zusammenarbeit von Domainexperten und Softwareentwicklern. Dabei lässt sich auch ein guter Übergang zum dritten Punkt bilden, dieser spricht sich explizit für die Schaffung einer gemeinsamen Sprachbasis, der Ubiquitous Language, innerhalb eines begrenzten Umfeldes aus. [Evans 2015, S. X-1] Auch im Umfeld von TYPO3 wird auf die Grundlagen von Domain-Driven Design eingegangen. Das auf den Seiten der TYPO3 Dokumentation bereit gestellte Buch zum Thema Extension Entwicklung beschreibt dabei auf wenigen Seiten die Prinzipien von Domain-Driven Design und wie diese im Umfeld von TYPO3 angewandt werden. [Kurfürst 2016]

Der Begriff Requirements Engineering wird sowohl von Hruschka [2014], als auch von Rupp [2014] auf Basis der Definition des International Requirements Engineering Board (IREB) beschrieben. Dabei handelt es sich um einen „systematische[n] und disziplinierte[n] Ansatz zur Spezifikation und zum Management von Anforderungen“. [Rupp 2014, S. 13] Ferner wird es in drei Ziele gegliedert:

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“ [Rupp 2014, S. 13]
2. „die Wünsche und Bedürfnisse der Stakeholder zu verstehen und zu dokumentieren“ [Rupp 2014, S. 13]
3. „die Anforderungen zu spezifizieren und zu managen, um das Risiko zu minimieren, ein System auszuliefern, das nicht den Wünschen und Bedürfnissen der Stakeholder entspricht“ [Rupp 2014, S. 13]

Aus der Definition und den Zielen geht bereits hervor, dass Requirements Engineering auch das Management von Anforderungen umfasst, welches unter dem Begriff Requirements Management auch als eigene Disziplin behandelt wird. Im Gegensatz zu dieser an das IREB angelehnten Definition, begrenzt [Grande 2014, S. 4] den Begriff des Anforderungsmanagement auf die Bereiche Anforderungs-Entwicklung und Anforderungs-Management, sowie in die Teilaufgaben „ermitteln, dokumentieren, abstimmen und verwalten“. In dieser Definition deckt sich Anforderungsmanagement eher mit dem Begriff Requirements Engineering, als Requirements Management, welches bei ihm als „Anforderungs-Management“ mit einem Bindestrich zu sehen ist. Mit ihrer Aufstellung (Abb. 2.1) zu den relativen Kosten für die Beseitigung eines

Softwarefehlers haben Leffingwell und Widrig gezeigt, wie gering die Kosten in der Konzeptionsphase im Vergleich späteren Wartungsarbeiten sind.

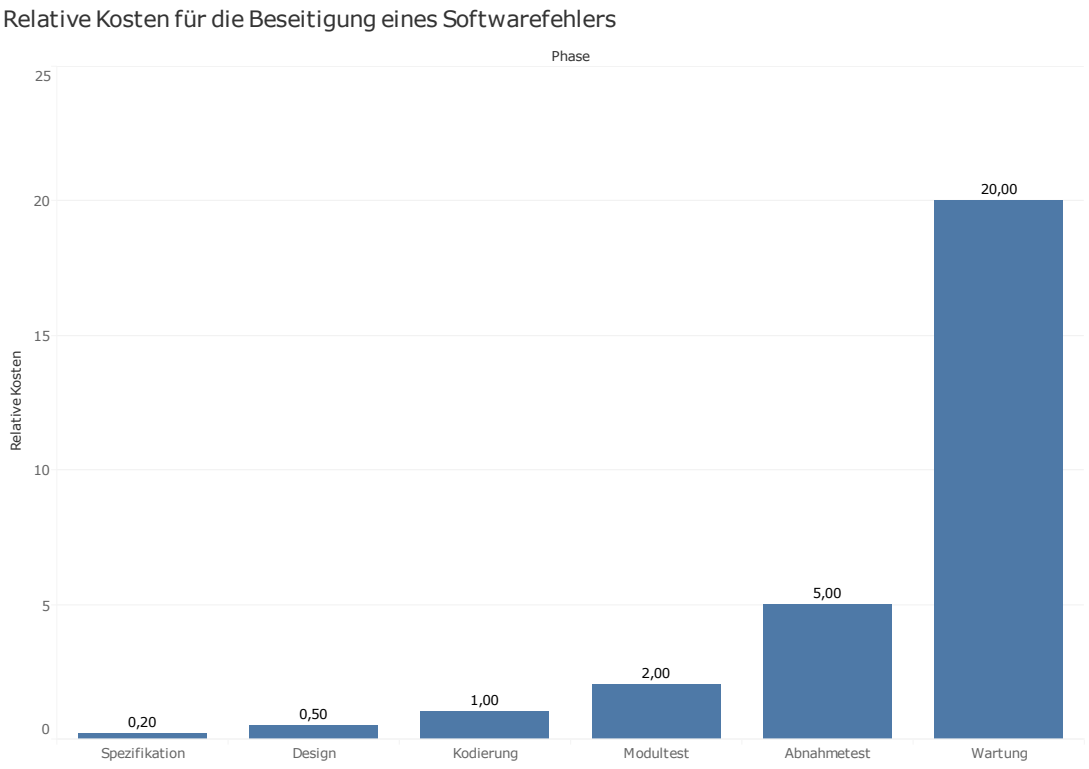

Abbildung 2.1: Relative Kosten für die Beseitigung eines Softwarefehlers [Leffingwell 2000]

[Grande 2014, S. 11] beschreibt es treffender damit, dass „Professionelles AM [Sie dabei unterstützt], möglichst wenig oder optimalerweise keine Fehler in der frühen Phase der Anforderungsermittlung in die Anforderungsspezifikationen und damit in das Produkt hinein zu bringen“. Auch Casper Jones hat in der 4th World Conference for Software Quality gezeigt, dass sich gutes Requirements Engineering auch in anderen Metriken widerspiegelt. Dabei hatte er ermittelt, dass gutes Requirements Engineering die Fehler pro Function Point von 0,23 auf 0,08 reduzieren kann. [Hruschka 2014, S. 3] Function Points sind eine formale, algorithmische Methode zur Aufwandsschätzung, dabei werden Funktionseinheiten, wie Datenein- und ausgaben, Anfragen, Schnittstellen zu externen Datenquellen und interne Logik, werden gezählt und entsprechend der Komplexität und definierter Einflussfaktoren bewertet“. [Schatten 2010, S. 93] Bei der Definition einer Anforderung bedient sich Rupp ebenfalls einer bestehenden akzeptierten Definition des Institute of Electrical and Electronics Engineers (IEEE). Demnach ist eine Anforderung:

1. „Eine Eigenschaft oder Fähigkeit, die von einem Benutzer (Person oder System)

zur Lösung eines Problems oder zur Erreichung eines Ziels benötigt wird.“ [Rupp 2014, S. 13]

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.“ [Rupp 2014, S. 13]
3. „Eine dokumentierte Repräsentation einer Eigenschaft oder Fähigkeit gemäß (1) oder (2).“ [Rupp 2014, S. 13]

Auch hier fässt sich Grande wesentlich kürzer und beschreibt eine Anforderung als „Eigenschaften, Funktionalitäten und Qualitäten, die ein Produkt bekommen soll“ [Grande 2014, S. 5], stimmt jedoch auch mit dem dritten Punkt der IEEE Definition überein, welche nur schriftlich dokumentierte Anforderungen als diese anerkennen. Anforderungen lassen sich weiter unterteilen. Während Grande die Anforderungen lediglich in funktionale und nicht-funktionale Anforderungen unterteilt, bei denen die nicht-funktionalen nur in Qualität und Randbedingungen weiter untergliedert werden [Grande 2014, S. 37ff], teilt Rupp die Anforderungen in „funktionale Anforderungen, technologische Anforderungen, Qualitätsanforderungen, Anforderungen an die Benutzungsoberfläche, Anforderungen an sonstige Lieferbestandteile, Anforderungen an durchzuführende Tätigkeiten und rechtlich-vertragliche Anforderungen“. [Rupp 2014, S. 17] Dabei stellen alle Gliederungspunkte, außer den funktionalen Anforderungen, die nicht-funktionalen Anforderungen dar. Die übergeordneten Schritte teilt auch Rupp in ermitteln, dokumentieren, prüfen und verwalten ein. Bei der Ermittlung von Anforderungen sind zunächst die Stakeholder zu ermitteln. Diese definiert Hruschka als „alle Personen oder Organisationen, die direkt oder indirekt Einfluss auf die Anforderungen des Systems haben“ [Hruschka 2014, S. 34], weiterhin beschreibt er die Wichtigkeit von Stakeholdern in der Anforderungsermittlung „Verpasste Stakeholder sind verpasste Requirements!“ [Hruschka 2014, S. 34]. Außerdem stellt er 4 wichtige Faktoren von Stakeholdern dar, Interesse, Beitrag, Einfluss und Anspruch. [Hruschka 2014, S. 36] Nach der Auswahl der Stakeholder folgt die Ermittlung der Anforderung. Dies geschieht über unterschiedliche Ermittlungstechniken, welche Rupp ausführlicher behandelt. Die Tabelle 2.1 stellt eine Übersicht der Ermittlungstechniken dar, die Rupp beschreibt.

Nach der Ermittlung der Anforderungen müssen diese in ein geeignetes Format übertragen werden. Dabei unterscheidet man zwischen verschiedenen Arten der Dokumentation, aber auch formalen Dokumenten, die entweder selbst Anforderungsdokumente darstellen oder Abschnitte mit definierten Anforderungen enthalten. Als Beispiele für Arten von Anforderungen stehen User Story und das User Story Mapping, UML Klassen-, Aktivitäts- und Sequenzdiagramme, aber auch Use Cases und Anforderungsschablonen. Während die Diagramme aus der Unified Modeling Language (UML) eher eine vereinfachte grafische Darstellung der Anforderung darstellen, sind User Stories, Use Cases und über Anforderungsschablonen definierte Anforderungen eine textuelle

Kreativitätstechniken	Brainstorming Methode 6-3-5 Wechsel der Perspektive
Beobachtungstechniken	Feldbeobachtung Apprenticing
Befragungstechniken	Fragebogen Interview
Artefaktbasierte	Systemarchäologie Reuse
Unterstützende Techniken	SOPHIST-REgelwerk Workshop Mind Mapping

Tabelle 2.1: Ermittlungstechniken für Anforderungen [Rupp 2014, S. 99ff]

Beschreibung. Von den formalen Dokumenten sind vor allem Lasten- und Pflichtenheft die gängigsten Varianten. Speziell für die Erfassung von Anforderungen wurde von James und Susanne Robertson das Volere-Schema entwickelt. [Hruschka 2014, S. 57] Auch das IEEE hat mit der Software Requirements Specification [IEEE 1993] für den Bereich der Softwareentwicklung eine Vorlage zur Anforderungsdokumentation geschaffen. Das Prüfen von Anforderungen dient der Qualitätssicherung und soll einerseits sicherstellen, dass die beschriebenen Anforderungen der Definition einer Anforderung gerecht werden, aber auch eine Prüfung hinsichtlich ihrer Umsetzung erfolgt. Bei der Umsetzung wird geprüft, ob das entwickelte Produkt mit den zuvor definierten Anforderungen überein stimmt. Als Prüftechniken gibt Rupp dafür beispielsweise Reviews und Prototypen an. [Rupp 2014, S. 318ff] Den letzte Bereich im Requirements Engineering stellt das Verwalten von Anforderungen dar. Hierbei wird zunächst jede Anforderung mit einer eindeutigen Identifikation ausgestattet. Diese bezeichnet Rupp als „Objekt-ID“ [Rupp 2014, S. 383] und geht im späteren Verlauf auf weitere Attribute ein [Rupp 2014, S. 409], die für die genauere Beschreibung von Anforderungen notwendig sein könnten. Grande gibt diese Attribute direkt als „FAPA, das Fünfeck der Anforderungs-Pflicht-Attribute“ [Grande 2014, S. 40] vor, welches auch die Identifikation enthält, sowie weiterhin Name, Beschreibung, Status und Version. [Grande 2014, S. 41] Zusammen mit der Versionierung wird auch eine Änderungshistorie der Anforderung erstellt, welche die Nachverfolgung von Änderungen transparent machen soll. Konfigurationen, Releases und Baselines sind nach Rupp ein Teil des Change und Release-Management. Dabei stellen Konfigurationen eine Auswahl von Anforderungen dar. Wenn diese Konfiguration alle Anforderungen für ein Release umfasst, wird sie als Baseline bezeichnet. [Rupp 2014, S. 455] Grande definiert den Begriffe Baseline als „eine definierte Konfigurationsbasis[, mit] definierte[n] Versionen der darin enthaltenen Artefakte“ [Grande 2014, S. 101], sowie Release als „eine Baseline, die eine Auslieferung an den Kunden darstellt“ [Grande 2014, S. 101]. Während der Verwal-

tung von Anforderungen soll auch die Nachverfolgbarkeit (Traceability) gewährleistet werden. [Grande 2014, S. 87] Grande beschreibt die Nachverfolgbarkeit als wichtige Voraussetzung „für die Qualitätssicherung und für Analysen, wie zum Bespiel die Einfluss-Analyse“ [Grande 2014, S. 90].

Das Capability Maturity Model Integration (CMMI) ist ein Reifegradmodell mit verschiedenen Sammlungen von Best Practice Modellen und hilft bei der Einführung dieser. Es dient der Einschätzung des Reifegrades und der Verbesserung der eigenen Prozesse. Um den Grad der Reife in einem Prozessgebiet zu beschrieben, gibt CMMI sogenannte Fähigkeitsgrade vor. Diese gliedern sich nach Unnvollständig (0), Durchgeführt (1), Geführt (2) und Definiert (3). [CMMI Product Team 2011, S. 36] Das CMMI für Entwicklung (CMMI-DEV) stellt dabei das Modell für die Software-Entwicklung dar. Das Modell basiert auf den Erkenntnissen von Teams aus der Softwareentwicklung, dem Software Engineering Institute, sowie Industrie und Regierung. [CMMI Product Team 2011, S. 4] CMMI ist eine umfangreiche Sammlung, daher wurde der Fokus auf die Kapitel Anforderungsentwicklung (Requirements Development, RD) und Anforderungsmanagement (Requirements Management, REQM) gelegt. Dabei werden diese Kapitel wiederum in spezifische Ziele (Specific Goals, SG) und spezifische Praktiken (Specific Practice, SP) unterteilt. [CMMI Product Team 2011, S. 27] Die spezifischen Praktiken werden weiterhin durch Beispiele für Arbeitsergebnisse, sowie Subpraktiken ergänzt.

SG 1 - Kundenanforderungen entwickeln	SP 1.1 Bedürfnisse herausfinden
	SP 1.2 Bedürfnisse der Stakeholder in Kundenanforderungen überführen
SG 2 - Produktanforderungen entwickeln	SP 2.1 Anforderungen an Produkte und Produktbestandteile etablieren
	SP 2.2 Anforderungen an Produktbestandteile zuweisen
	SP 2.3 Schnittstellenanforderungen identifizieren
SG 3 - Anforderungen analysieren und validieren	SP 3.1 Betriebskonzepte und Anwendungsszenarien etablieren
	SP 3.2 Definition erforderlicher Funktionalität und Qualitätsattribute etablieren
	SP 3.3 Anforderungen analysieren
	SP 3.4 Anforderungen analysieren und abgleichen
	SP 3.5 Anforderungen validieren

Abbildung 2.2: Anforderungsentwicklung (RD), eigene Darstellung auf Basis von [CMMI Product Team 2011, S. 337]

Die Abbildungen 2.2 und 2.3 zeigen jeweils die spezifischen Ziele und Praktiken für die beiden Kapitel Anforderungsentwicklung und Anforderungsmanagement. Die Anforderungsentwicklung befasst sich mit der Entwicklung, Analyse und Validierung von Kunden- und Produktanforderungen. Wichtige Arbeitsergebnisse in der Anforderungsentwicklung sind Ergebnisse der Anforderungsermittlung, Einschränkung von Verifizierung und Validierung durch den Kunden, Produktanforderungen, Schnittstellenanforderungen, Betriebskonzept, Funktionsumfang und Qualitätsattribute, sowie

SG 1 - Anforderungen verwalten	SP 1.1 Anforderungen verstehen
	SP 1.2 Zusagen zu Anforderungen einholen
	SP 1.3 Anforderungsänderungen verwalten
	SP 1.4 Bidirektionale Nachverfolgbarkeit von Anforderungen aufrechterhalten
	SP 1.5 Abstimmung zwischen Projektarbeit und Anforderungen sicherstellen

Abbildung 2.3: Anforderungsmanagement (REQM), eigene Darstellung auf Basis von [CMMI Product Team 2011, S. 352]

Fehlerbericht für Anforderungen. [CMMI Product Team 2011, S. 339-347] Das Anforderungsmanagement befasst sich mit der Verwaltung von Anforderungen. Wichtige Arbeitsergebnisse in im Anforderungsmanagement sind genehmigte Anforderungen, Zusagen zu Anforderungen und deren Änderung, sowie Dokumentation dieser.

In den meisten Bereichen der Softwarebranche hat sich das Software Engineering als fester Begriff für den Prozess der Ermittlung, Planung, Modelierung, Entwicklung, Implementierung und Verwaltung [Foster 2014, S. 4] bereits etabliert, wie man an Jobbörsen wie Monster.de (1000+ Stellenangebote) oder StepStone (5675 Stellenangebote) anhand der Einträge erkennen kann. Im Bereich der Web-Entwicklung wurde dafür bereits 1996 ein anderer Begriff geprägt, der des Web Engineers. [Mendes 2006, S. 2] Zum Vergleich listen Monster.de (562 Stellenangebote) und StepStone (18 Stellenangebote) hierbei wesentlich weniger Treffer unter dem Begriff „Web Engineer“, die außerdem oftmals auch als „Software Engineer“ mit dem Zusatz „Web“ ausgeschrieben sind. Die Definition von Murugesan [1999] weicht nicht sonderlich von Foster [2014] ab, hat jedoch noch den Zusatz mit Verweis auf Web-Anwendungen: „Web Engineering uses scientific, engineering, and management principles and systematic approaches to successfully develop, deploy, and maintain high-quality Web systems and applications“ [Murugesan 1999]. Die Abbildung 2.4 zeigt, dass auch bei Google Trends die Suche nach Software Engineering und Software Engineer eine wesentlich größere Rolle spielt, als die nach Web Engineering und Web Engineer.

Zieht man dabei auch noch den Vergleich zu den beiden Bereichen Software Development und Software Developer, sowie Web Development und Web Developer heran, wie in Abbildung 2.5 zu sehen ist, so ergibt sich jedoch ein wesentlich anderes Bild. Das Web Development ist dabei näher am Trend des Software Development.

Eine mögliche Erklärung für diese Diskrepanz wäre der Ansatz, dass Engineering innerhalb der Web Technologien nur von einer sehr begrenzten Anzahl an Personen durchgeführt wird. In einem Gespräch mit Dan Bechard (Gesprächsprotokoll im Anhang) brachte dieser die Aussage, dass der Terminus „Web Engineering“ eher nach Personen klingt, die das Web und deren grundlegende Technologien und nicht einzelne Webanwendungen entwickeln. Ein weiterer Punkt wäre die viel frühere Verbreitung des Begriffs Software Engineering, welche die Etablierung eines weiteren Begriffs, der sich lediglich auf eine spezielle Form von Anwendungen bezieht, verhindert hat. Die

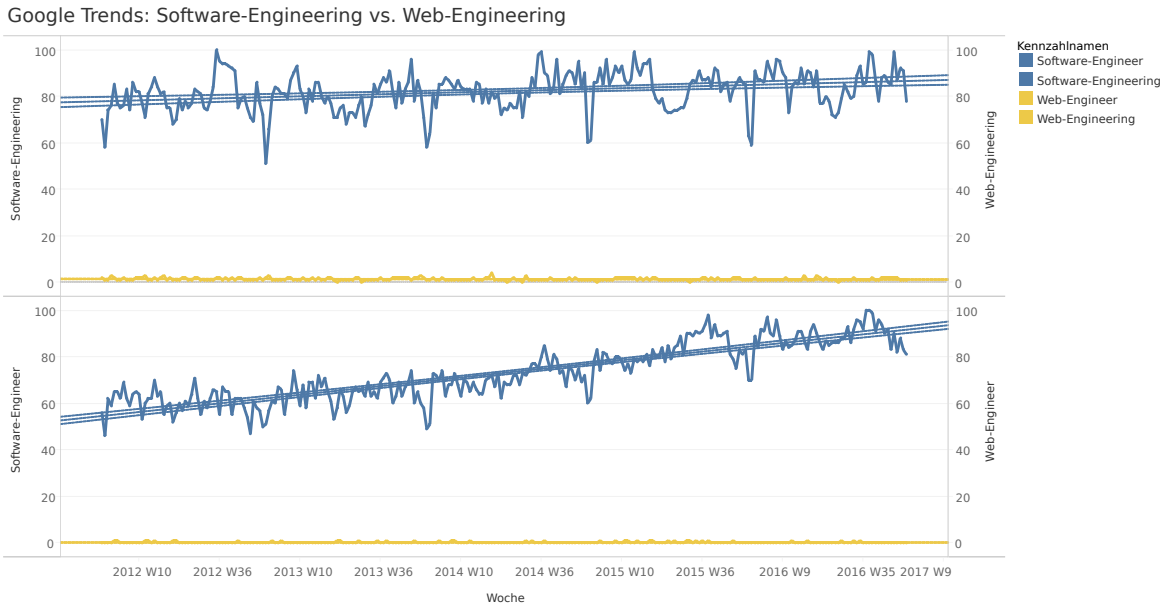

Abbildung 2.4: Vergleich von Software-Engineering und Web-Engineering bei Google Trends, Datenquelle: Google Trends (www.google.com/trends)

größten Unterschiede zwischen dem Web Engineering und der Entwicklung konventioneller Software wurden von Mendes [2006] zusammengetragen. Dazu zählen aus ihrer Sicht Application Characteristics, Primary Technologies Used, Approach to Quality Delivered, Development Process Drivers, Availability of the Application, Customers (Stakeholders), Update Rate (Maintenance Cycles), People Involved in Development, Architecture and Network, Disciplines Involved, Legal, Social, and Ethical Issues, sowie Information Structuring and Design. [Mendes 2006, S. 5]

2.3 Stand des Wissens

Der Stand des Wissens soll bereits durchgeführte Arbeiten mit Themenbezug und deren wesentlichen Inhalte darstellen.

Bereits in einem Artikel aus dem Jahr 2000 beschäftigt sich Overmyer mit der Frage, inwiefern sich das herkömmliche Requirements Engineering in der Softwareentwicklung von der Erstellung von Websites unterscheidet. Dabei sieht er im Wesentlichen 3 Hauptunterschiede. Die Gestaltung von Websites hat einen anderen Fokus, dabei sind Websites eher wie Magazine oder Broschüren zu sehen, dessen Fokus im Design und im Marketing liegt. [Overmyer 2000, S. 63] Daraus lässt sich bereits der zweite Punkt ableiten, denn die Gestaltung hat auch einen anderen disziplinarischen Schwerpunkt, welcher sich in die Bereiche Funktionalität, Usability und Grafikdesign unterteilt.

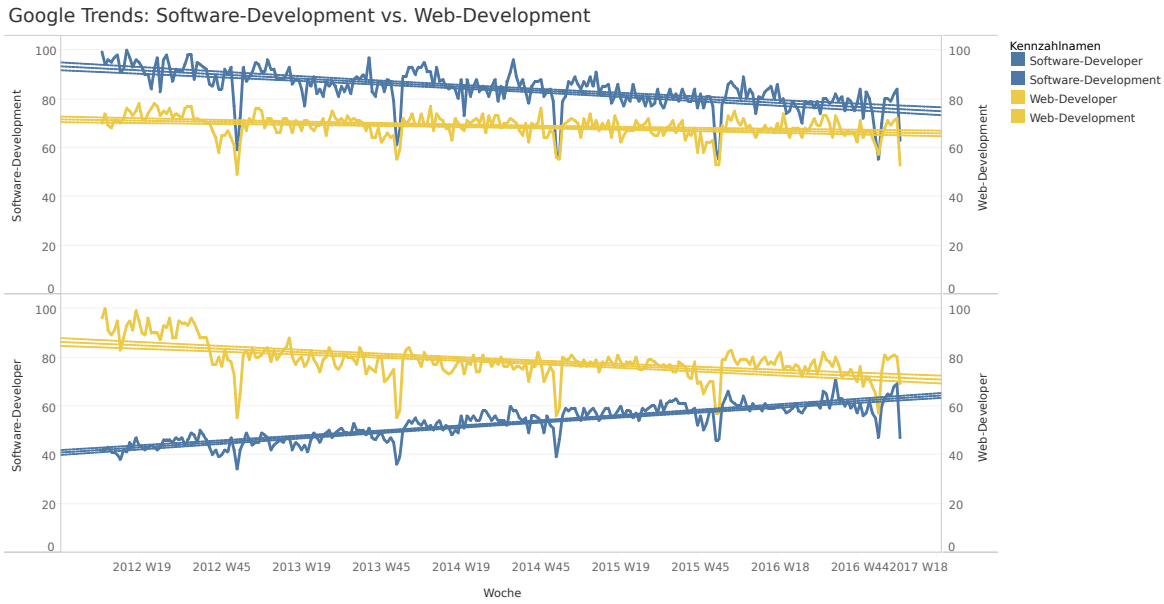

Abbildung 2.5: Vergleich von Software-Development und Web-Development bei Google Trends, Datenquelle: Google Trends (www.google.com/trends)

[Overmyer 2000, S. 63] Weiterhin haben Websites einen kürzeren Lebenszyklus als andere Softwareprodukte. Die Erstellung von einfachen Websites erfolgt in wesentlich kürzerer Zeit, wobei auch die Schwierigkeit im Vergleich zur Softwareentwicklung geringer ist. [Overmyer 2000, S. 63] Er kommt zu dem Schluss, dass abhängig von der zu entwickelnden Website die Ansprüche an das Requirements Engineering signifikant anders sein können. [Overmyer 2000, S. 64] Die Besonderheit dieses Artikels liegt auf der Abgrenzung zu tiefergehenden Themen der Web Entwicklung, jedoch lässt sich der disziplinarische Schwerpunkt von Funktionalität, Usability und Grafikdesign, sowie die höhere Relevanz von Marketing im Vergleich zu herkömmlichen Softwareprodukten auch auf komplexe Web Anwendungen übertragen.

Im Jahr 2004 haben Koch und Escalona eine Vergleichsstudie zu Requirements Engineering für Web Anwendungen durchgeführt. Dabei haben sie zunächst einige Grundlagen für die drei Bereiche Ermittlung, Spezifizierung und Validierung von Anforderungen behandelt. Dabei geht es vor allem um Techniken und Methoden, die später zum Vergleich heran gezogen werden sollten. Bei der Ermittlung von Anfordrungen sind es Interviews, JAD (Joint Application Development), Brainstorming, Concept Mapping, Sketching, Storyboarding, Use Case Modeling, Fragebögen, Checklisten und Terminology Comparison. [Koch 2004, S. 196f] Für die Spezifizierung von Anforderungen nutzen sie Natürliche Sprache, Glossare, Ontologien, Vorlagen, Szenarios, Use Case Modellierung, formale Beschreibungen und Prototypen. [Koch 2004, S. 198] Auch für die Validierung wurden einige Techniken dargestellt, dazu zählen Review, Walk-through, Audit, Nachverfolgbarkeitsmatrix und Prototypen. [Koch 2004, S. 199]

In der Web Entwicklung gibt es viele verschiedenartige Stakeholder, wie Analysten, Kunden, Nutzer, Grafikdesigner, aber auch Marketing-, Multimedia- und Sicherheitsexperten, die alle relevante Anforderungen an das Produkt haben. [Koch 2004, S. 199] Die Hauptfunktionen des Systems sind dabei Navigationsstruktur, Benutzerschnittstelle und die Personalisierbarkeit, wobei auch hier in der Gestaltung auf Multimedia- und Marketingaspekte geachtet werden muss. [Koch 2004, S. 199] In ihrer Vergleichsstudie schlagen sie eine eigene Unterteilung für funktionale Anforderungen vor, die sich an den Eigenheiten der Web Entwicklung orientiert. Die Datenanforderungen beschreiben, wie Informationen gespeichert und verwaltet werden. Die Schnittstellenanforderungen beschreiben, wie die Nutzer mit der Web Anwendung interagieren. Die Navigationsanforderungen beschreiben, wie Nutzer durch die Web Anwendung geführt werden sollen. Die Personalisierbarkeitsanforderungen beschreiben, wie sich die Web Anwendung dynamisch an das Nutzer- oder Umgebungsprofil adaptiert. Die Transaktionsanforderungen beschreiben, wie die Web Anwendung intern Informationen austauscht und Daten verarbeitet. [Koch 2004, S. 200] Zuletzt wurden für die Grundlage des Vergleiches Methoden aus der Web Entwicklung betrachtet, welche bereits einige der zuvor beschrieben Anforderungen abdecken: [Koch 2004, S. 200ff]

- „WSDM: Web Site Design Method“
- „SOHDM: Scenario-based Object-Oriented Hypermedia Design Methodology“
- „RNA: Relationship-Navigational Analysis“
- „HFPM: Hypermedia Flexible Process Modeling“
- „OOHDM: Object Oriented Hypermedia Design Model“
- „UWE: UML-based Web Engineering“
- „W2000“
- „WebML: Web Modeling Language“
- „NDT - Navigational Development Techniques“
- „Design-driven Requirements Elicitation“

Aus der Abbildung 2.6 lässt sich das Ergebnis des Vergleiches von Anforderungen und Methoden sehen. Insbesondere die Navigational Development Techniques und das Design-driven Requirements Elicitation decken dabei alle erforderlichen Arten von Anforderungen ab.

bei der Auswahl von Software Bezug auf die „Vergleichsstudie zu Werkzeugen für das Anforderungsmanagement“ von Studenten der Wirtschaftsinformatik an der DHBW-Stuttgart [Hahn 2014]

	Data Req.	User Interface Req.	Navigational Req.	Adaptive Req.	Transactional Req.	Non-Functional Req.
WSDM	✓			✓		✓
SOHDM	✓	✓			✓	
RNA	✓	✓	✓		✓	
HFPM	✓	✓	✓			✓
OOHDM	✓	✓	✓			
UWE	✓	✓	✓	✓		✓
W2000			✓	✓	✓	
WebML	✓	✓		✓		✓
NDT	✓	✓	✓	✓	✓	✓
DDDP	✓	✓	✓	✓	✓	✓

Abbildung 2.6: Vergleich von Anforderungen und Vorgehensweisen aus dem Requirements Engineering für Web Anwendungen [Koch 2004, S. 204]

2.4 Methoden

Die Arbeit stützt sich zum größten Teil auf die Untersuchung einer Fallstudie zur Entwicklung von TYPO3 Extensions. Dazu wurde vor allem mit induktiven Rückschlüssen aus den Interviews auf den Gesamtprozess geschlossen. Während der Soll-Prozess Konzeption fand jedoch auch deduktiv unter Einfluss der Literaturanalyse eine Überführung von allgemeinem Wissen in den speziellen Prozess statt. Auf der untersten Ebene wurde zunächst mit einer Literaturrecherche breites Wissen zum Themengebiet aufgebaut. Dazu zählen Software/Web Development, Requirements Engineering, TYPO3 und Domain-Driven Design. Auch die Mitarbeit in Projekten vor Ort, im Rahmen des Apprenticing brachte Erkenntnisse mit sich. Als größte Informationsquelle dienten jedoch die Interviews, auf denen ein Großteil der Ist-Analyse basiert, aber auch Wünsche und Vorstellungen für den zukünftigen Prozess enthielt. Aufbauend auf den Interviews wurde mit Hilfe von Modellierung der Prozess aufbereitet.

In Abbildung 2.7 ist das Methodenspektrum der Wirtschaftsinformatik mit Zuordnung in verschiedene Bereiche abgebildet. Daran lässt sich erkennen, dass innerhalb der Arbeit alle Bereiche abgedeckt sind, sowohl qualitativ-behavioristische (Fallstudie) und -konstruktivistische Methoden (Aktionsforschung), als auch quantitativ-behavioristische (Quantitative Querschnittsanalyse) und -konstruktivistische Methoden (Referenzmodellierung).

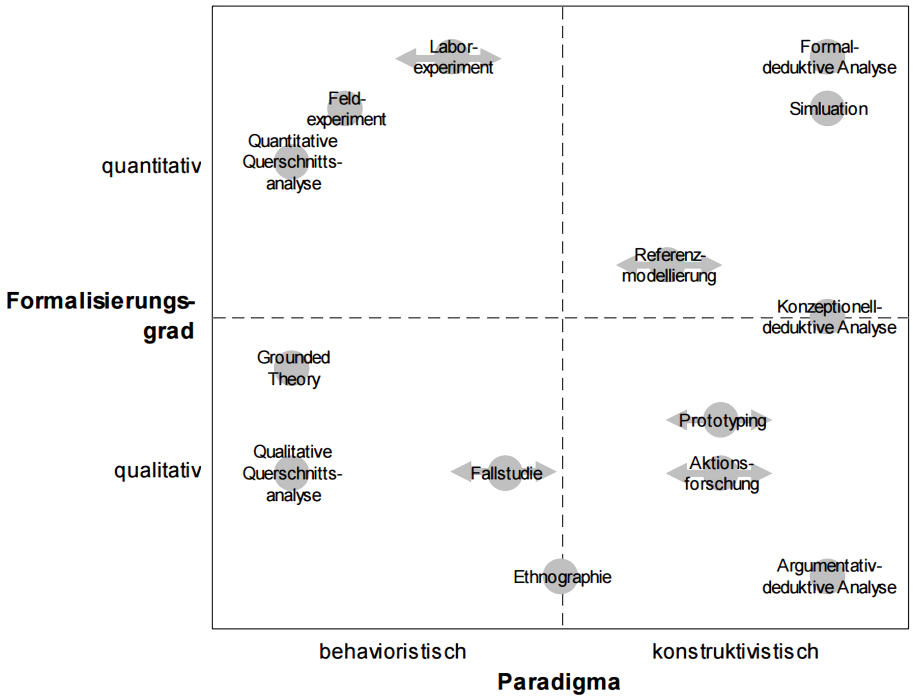

Abbildung 2.7: „Einordnung der Methoden im Portfolio“ [Wilde 2006, S. 14]

3 Stand des Vorgehens

Dieses Kapitel befasst sich mit einer eingängigen Betrachtung des derzeitigen Zustands in den Bereichen Prozess, Übergabeformen, Projektmanagement und Reifegrad. Die Ermittlung der Informationen erfolgte über Interviews. Die Fragen basierten auf einer Sammlung von Notizen, die sich aus Gesprächen mit Mitarbeitern im Vorfeld ergeben haben, sowie Fragestellungen aus der Literatur. Die Interviewvorlage mit den Fragen befindet sich im Anhang.

Es wurden insgesamt 10 Interviews geführt, darunter Teilnehmer aus den einzelnen Gewerken Frontend, Backend und Integration, sowie Design und Projektmanagement. Im Durchschnitt waren die Interviews etwa 1 Stunde lang. Die Schwierigkeit lag in den Interviews darin, eine konkrete Trennung von Gesamtprojekten und der Umsetzung einzelner Extensions vorzunehmen. Das war nicht immer möglich.

3.1 Prozess

Die Prozesse sind bisher nirgendwo in ausführlicher Form beschrieben worden. Es gibt eine Projektphasenwand (Abbildung 3.1), welche iterativ durch die Mitarbeiter von CPS-IT erstellt wurde und grob den Prozess umreißt. Auf der Darstellung sind auch weitere Informationen zu erkennen, jedoch existiert keine Abbildung mehr, welche die Details um den Hauptprozess in erkenntlicher Weise darstellt.

In der Abbildung lässt sich jedoch eine grobe Struktur erkennen, welche den Gesamtprozess in 11 Schritte einteilt.

1. Ziel definieren
2. Wer ist der User?
3. Content analysieren
4. Design Atmosphäre
5. Prototyp
6. CMS Integration
7. Programmierung
8. Redaktion

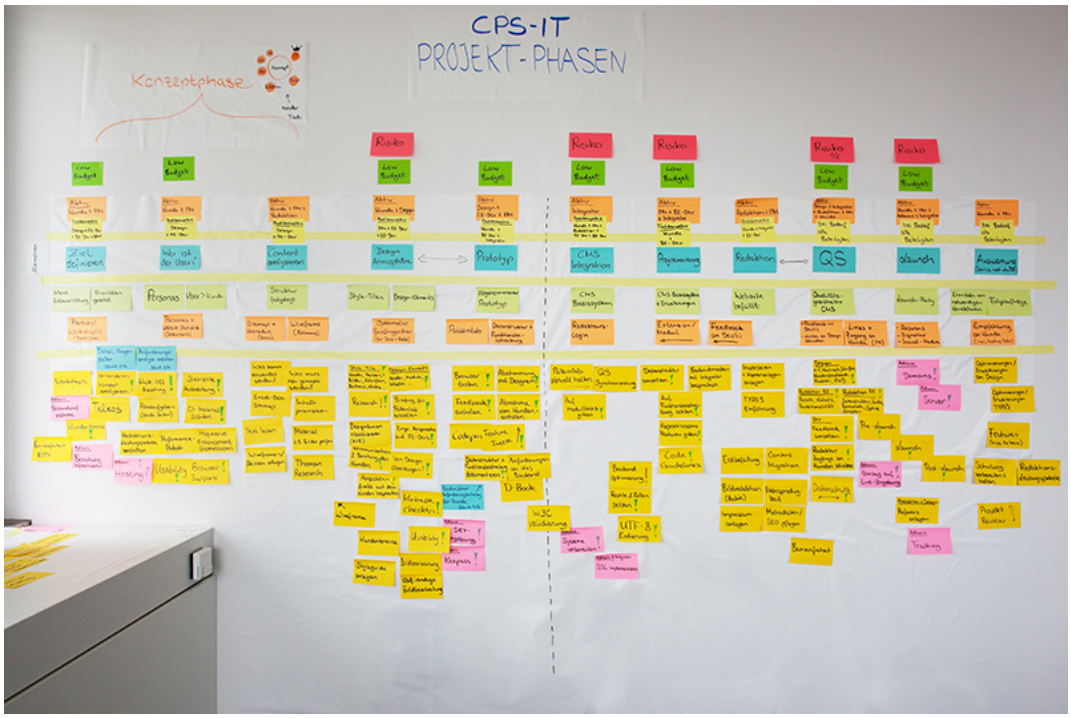

Abbildung 3.1: Projektphasenwand

9. QS
10. Launch
11. Auswertung

Außerdem lässt sich eine Einteilung in 2 Phasen ableiten, eine Phase für Konzeption und Design von Schritt 1 bis 5, sowie eine Phase für die Umsetzung von Schritt 6 bis 11. Aus den Interviews geht hervor, dass diese Einteilung weiterhin aktuell ist. Zwischen den beiden Phasen soll es zudem ein „Feature Freeze“ geben. In der zweiten Phase ist der Ablauf anschließend weniger agil. Konkret lässt sich dabei kein Vorgehensmodell aus der Softwareentwicklung erkennen, es handelt sich eher um Mischung verschiedener Ansätze. Bei der Entwicklung von Extensions hängt der Umfang in dem die einzelnen Schritte durchgeführt werden von der Größe der Extension ab. Eine Extension die beispielsweise nur als Backend Modul dient, wird dabei wenig bis gar nicht vom Design betrachtet.

Bei den Beteiligten am Entwicklungsprozess auf Seiten des Unternehmens gliedern sich die Rollen in Projektmanagement, Technischer Projektleitung, Design, Frontend-Entwicklung, Backend-Entwicklung, Integration und Redaktion.

Der Prozess bezogen auf das Anforderungsmanagement beginnt mit dem ersten Kundenkontakt vor Ort, im Rahmen eines Kick-Off Meetings mit dem Kunden. Dabei

spielen Entfernung, aber auch Projektgröße eine Rolle bei der Entscheidung über die Art des Meetings, bspw. vor Ort oder per Telefonkonferenz. Das Ziel ist die Einführung in das Projekt, sowie erste Erkenntnisse zu Anforderungen. Teilnehmer sind dabei weiterhin Projektmanagement, Design, teilweise Technische Projektleitung und Frontend-Entwicklung, sowie selten auch Backend-Entwicklung. Dies ist vor allem davon abhängig, wie konkret das Projekt bereits im Vorfeld beschrieben ist. Dabei ließ sich im Interview außerdem feststellen, dass aus Sicht des Projektmanagements die Backend-Entwicklung eingebunden wird, diese sich jedoch nicht eingebunden fühlt. Die Projektmanager bereiten sich auf das Interview durch die Sichtung von Material aus dem Vorabbriefing, sofern dieses vorhanden ist, sowie mit einem firmeninternen Fragenkatalog vor. Zum Teil wurden auch Fragebögen verschickt, dabei jedoch unterschiedliche Erfahrungen gemacht. Es fehlte dabei konkret die Beratungsleistung, die vom Kunden erwartet wird, da dieser nur selten das technische Verständnis mitbringt. Die Durchführung erfolgt anschließend mit Basisfragen aus Erfahrungswerten und dem Fragenkatalog. Dabei wird zunächst mit einfachen Fragen begonnen. Die Antworten werden teilweise als Gesprächsprotokoll erfasst, häufiger werden jedoch nur die Ergebnisse formlos zusammengefasst. Die Abbildung 3.2 stellt den Prozess der Anforderungsermittlung dar. Am Anfang ist es weiterhin empfehlenswert nur wenige Zahlen und Zeiten zu nennen, da sich der Kunde im späteren Verlauf auf diese ersten Schätzungen als feste Termine berufen könnte. Im Kick-Off Meeting findet auch die Heranführung an die Arbeit als Agentur statt.

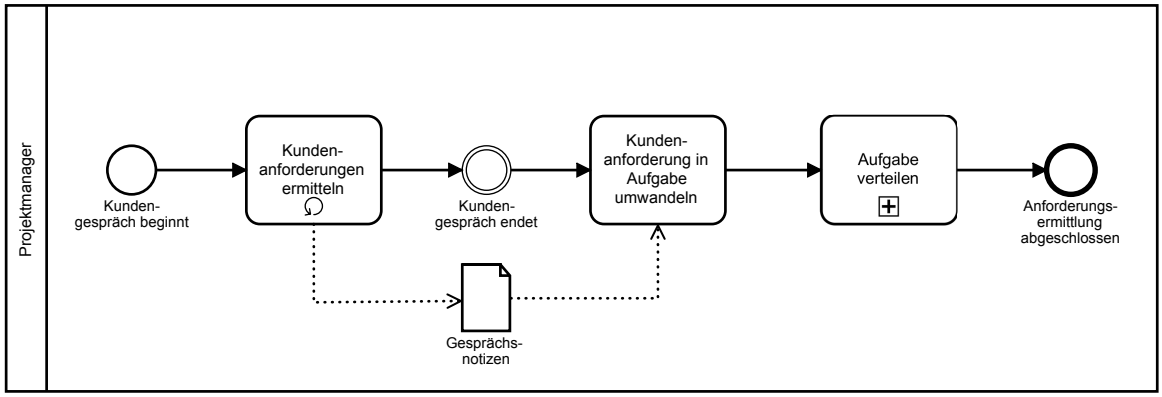

Abbildung 3.2: Prozess: Anforderungsermittlung

Die Heranführung an die Arbeit als Agentur ist sinnvoll, um den Kunden eine Übersicht über die Vorgänge zu geben und ein gemeinsames Projektverständnis zu erlangen. Das Vorgehen wird derzeitig im Angebot festgehalten. Wichtig ist dabei auch dem Kunden klar zu machen, dass er der Herr der Fachdomäne ist, während CPS-IT die Umsetzung durchführt. Dieser Punkt scheint nach den Aussagen der Interviewpartner nicht immer klar zu sein. Weiterhin werden die Agentur und das Portfolio vorgestellt. Unter Umständen werden auch Einblicke in Confluence gegeben, sowie die Vorteile der Arbeit mit Confluence erklärt. Dadurch soll auch eine Transparenz

gegenüber dem Kunden geschaffen werden. Das erweist sich jedoch schwieriger bei Konzernen und es ist von Seiten der Projektleitung Feingefühl erforderlich. Aus Sicht der Backend-Entwickler findet diese Heranführung noch nicht optimal statt. Man lässt sich dabei zu sehr von den Kundenwünschen treiben, anstatt diese zu lenken. Das führt zu späten Änderungen oder zu früher Umsetzung von Features, die dann Extrarunden verursachen. Die Einigung mit dem Kunden auf eine Arbeitsweise erweist sich zudem der Erfahrung nach als schwierig. Es sollen Formalien sein, aber die Flexibilität erhalten bleiben.

Die Kreativ-Workshops dienen der Entwicklung von Ideen und Zielen für die Gestaltung der Seite und werden in mehreren Iterationen bis zum fertigen Design durchgeführt. Daran beteiligt sind vor allem Design und Frontend-Entwicklung, aber auch Redakteure. Seltener vertreten sind Backend-Entwickler. Teilweise ist auch der Kunde direkt in den Kreativ-Workshop eingebunden, jedoch eher ab der zweiten Iterationen. Als Vorbereitung dienen Materialien vom Kunden, aber auch aktuelle Trends. In der ersten Phase des ersten Kreativ-Workshops wird das Ziel festgelegt und durch verschiedene Methoden, wie etwa einem Brainstorming, Ideen gesammelt. Das geschieht eher grob und noch nicht im Detail. In der zweiten Phase werden Ideen für den Workshop mit dem Kunden gesammelt, wie bspw. als Slogan für die Seite, welches durch das ganze Projekt mitgenommen wird. In weiteren Iterationen wird mit dem Kunden das Design entwickelt und verbessert, dazu werden von Designern und Frontend-Entwicklern Zwischenstände vorbereitet, wie etwa Prototypen und verschiedene Design-Vorschläge aus denen der Kunde wählen kann.

Neben den Meetings, die auch im Beisein des Kunden stattfinden, gibt es auch rein interne Meetings. Diese dienen vor allem der Besprechung von Projektfortschritt und Aufgabenverteilungen mit den internen Projektbeteiligten. Teilnehmer sind dabei abhängig vom Inhalt des Meetings, aber vor allem Projektmanagement, Technische Projektleitung, Design, Frontend- und Backend-Entwicklung, sowie Redaktion. Die Inhalte der Besprechung werden i.d.R. in Tickets überführt, wenn noch nicht geschehen, die anschließend von den Projektbeteiligten abgearbeitet werden.

Mit dem Fortschreiten der Designphase, beginnt auch die Arbeit der Frontend Entwickler. Diese erhalten vom Design ein Briefing-Protokoll zur Erstbefüllung des Pattern Lab. Anschließend erfolgt die konkrete Umsetzung des Designs. Die dafür benötigten Informationen werden aus Confluence, sowie den Redmine Tickets und teilweise dem Fileserver bezogen. Die Aufgaben, die in Form von Tickets kommen stellen jedoch nach Aussagen der Entwickler nicht immer klar was gemeint ist oder es fehlen notwendige Informationen. Die meisten dieser Probleme lassen sich jedoch durch Rückfrage klären. Die Kommentare in den Tickets, die sich unter der eigentlichen Aufgabenbeschreibung befinden, werden i.d.R. aufgearbeitet und Erkenntnisse zur Aktualisierung der Aufgabe verwendet. Ein Verweis auf Dokumente ist selten. Die Nachvollziehbarkeit der Anforderungen geht hierbei meistens verloren, weshalb es nach Aussage gut ist, in den Meetings zu sitzen. Eine Verifikation der Anforderungen findet bei der Umsetzung nur für den selbst geleisteten Teil statt, bei der Verantwortlichkeit wird hier

auf das Projektmanagement verwiesen. Dieser Punkt entfällt bei Zeitdruck meistens als erstes.

Die Umsetzung im Backend beginnt i.d.R. durch die Zuteilung von Tickets auf die jeweiligen Backend-Entwickler. Dabei gab es im Rahmen der Interviews unterschiedliche Aussagen bezüglich des Vorlaufes. Aus Sicht des Projektmanagements, bzw. der Technischen Projektleitung wird vor jedem Ticket das Gespräch mit dem Entwickler gesucht, um die Einzelheiten zu besprechen. Aus Sicht der Backend-Entwickler dagegen, findet dies eher selten statt. Es ist nicht ersichtlich inwiefern die Aufgaben eines Software/Web-Engineers durchgeführt werden, um Datenmodelle gemäß dem DDD zu entwickeln und mit dem Kunden abzustimmen. Die hierbei entstehenden Probleme aus mangelnder konzeptioneller Vorarbeit für die Aufgaben im Backend, sowie die unterschiedliche Wahrnehmung der Kommunikation und damit verbundener Informationsdefizite führt dazu, dass Teilaufgaben bei den Entwicklern landen, die nicht ohne weitere Nachfrage gelöst werden können. Um die Informationsdefizite auszugleichen, werden teilweise Dokumente bereit gestellt. Diese sind jedoch selten aufgearbeitet und enthalten wenig bis keine konkreten Informationen für die Backend Entwicklung. Die Abbildung 3.3 stellt den Prozess der Aufgabenverteilung dar.

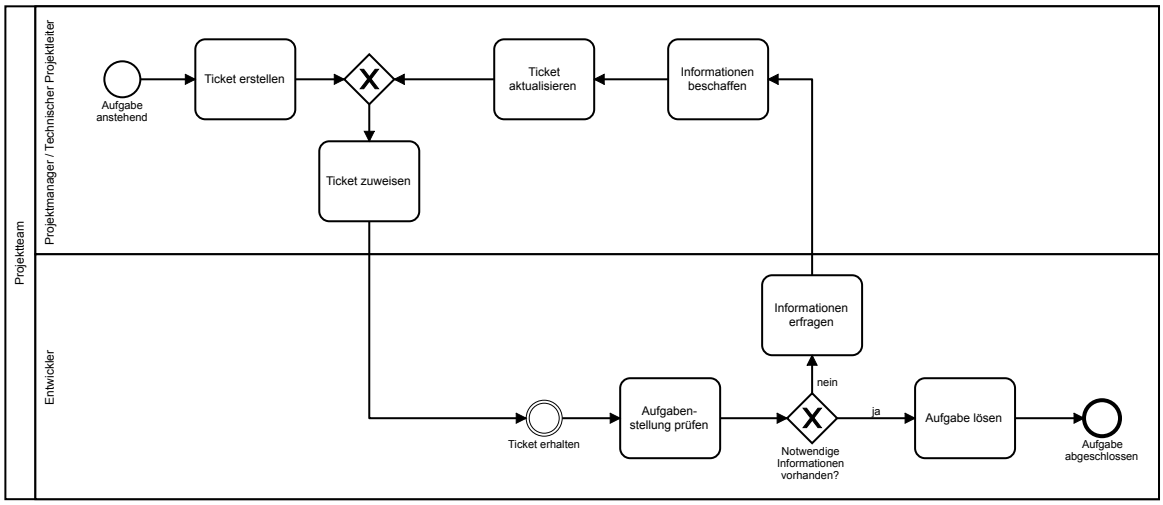

Abbildung 3.3: Prozess: Aufgabenverteilung

Eine der Schwierigkeiten in Softwareentwicklungsprozessen stellen Änderungsanfragen dar. Je nach Entwicklungsstatium können diese entscheidenden Einfluss auf den finanziellen Erfolg eines Projektes haben. Eine Arbeit von Leffingwell und Widrig hat dazu 2000 eine Aufstellung veröffentlicht, welche relativen Kosten für die Beseitigung von Softwarefehlern in den jeweiligen Phasen der Entwicklung zu erwarten sind (vgl. [Grande 2014, S. 8]). Dies lässt sich zu gewissen Teilen auch auf Änderungsanfragen beziehen, da sich die Prozesse zur Änderung einer Funktionaliät und der Behebung eines Fehlers sehr ähneln. Die Änderungsanfrage geht i.d.R. vom Kunden aus und

kann per E-Mail, Telefon oder im Rahmen eines Meetings eintreffen. Je nach Aufwand kann die Änderung sofort durch den Projektmanager eingepflegt werden, bspw. bei Änderungsanfragen während der Konzeptionsphase. Generell sollten Änderungsanfragen erst beim Projektmanager landen, es kommt jedoch auch vor, dass diese direkt dem Entwickler übermittelt werden. Sollte die Aufgabe nicht durch den Projektmanager gelöst werden können, übergibt dieser an den Technischen Projektleiter. Sollte auch dieser die Aufgabe nicht selbst lösen können, so wird sie an einen Entwickler übergeben. Wenn die Aufgabe gelöst ist, wird sie auf dem gleichen Weg wieder zum PM zurück gegeben, dabei liegt es in der Verantwortung der jeweiligen Person die Aufgabe auf Vollständigkeit und Richtigkeit zu überprüfen. Teilweise werden auch Folgen und Auswirkungen von Änderungen genauer betrachtet, dieser Teil des Prozesses scheint jedoch nicht immer und nicht durch alle Beteiligten durchgeführt zu werden. Änderungen in der Designphase werden durch das Design selbst bearbeitet. Bei bestehenden Projekten wird dazu ein Ticket erstellt, die Änderung durchgeführt und anschließend eine Rückmeldung an den Kunden gegeben. Bei neuen Projekten geschieht die Einarbeitung der Änderungen dagegen iterativ. Gelegentlich werden potenzielle Änderungsanfragen erst durch die Entwickler erkannt und signalisiert, da diese einen besseren technischen Blick auf die Umsetzung haben. Die Entwickler beklagten dabei generell zu wenig Einblick zu haben, Rücksprachen noch zu wenig erfolgen und sie stattdessen vordefinierte Tickets erhalten, die aus ihrer Sicht nicht realistisch sind. Die Frage nach der generellen Machbarkeit sollte dabei selbstverständlich sein. Aus Sicht der Entwickler werden Änderungsanfragen nur unzureichend durch das Projektmanagement gesteuert. Einzelne Anforderungen werden selten isoliert, sondern im Klärungsprozess um mehr Features und Anforderungen erweitert. Die ursprüngliche Aufgabenbeschreibung wird dabei selten angepasst, sodass sich die Entwickler aus dem Verlauf die aktuelle Aufgabe erlesen müssen. Problematisch sind Änderungsanfragen, die nicht als diese erkannt werden, da diese weder als Änderungsanfrage kommuniziert, noch kalkuliert werden. Der bei den Änderungsanfragen entstehende Mehraufwand hat auch Einfluss auf die Gefühlslage, da folgende Änderungen nach viel Arbeit mit Frustration verbunden sein können. Nach dem Abschluss der Änderung erfolgt selten bis nie eine Anpassung der Anforderungs-/Dokumente. Die Abbildung 3.4 stellt den Prozess der Änderungsanfrage dar.

In direktem Zusammenhang zur Änderungsanfrage steht auch die Einflussanalyse. Je nach Projektmanager ist die Durchführung der Einflussanalyse unterschiedlich stark ausgeprägt. Aus Sicht des Projektmanagements wird dabei nach Fall entschieden, ob eine eigene Einschätzung der Situation ausreichend ist oder ein Entwickler, bzw. der technische Projektleiter dazu befragt wird. Dabei werden mehrere Möglichkeiten mit Zeiteinschätzung evaluiert. Dies geschieht bevorzugt mit projektnahen Entwicklern. Aus Sicht der Backend-Entwickler finden nur kleine Einschätzungen statt, jedoch nichts was dem Charakter einer Einflussanalyse entspricht. Dabei wird nicht nach Wartbarkeit gefragt, sondern vielmehr nach Zeiteinschätzungen und unmittelbarer Umsetzung. Konkret kommen Einflussanalysen wie hier dargestellt aus dem Anforderungsmanagement und werden dort über die Beziehungen der Anforderungen

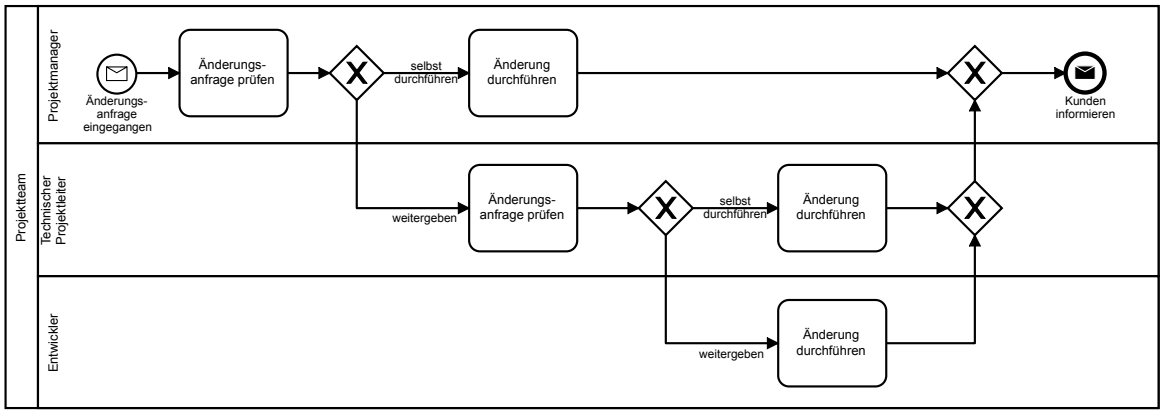

Abbildung 3.4: Prozess: Änderungsanfrage

zueinander durchgeführt, die fehlende Dokumentation der Beziehung von Anforderungen dürfte diesen Prozess jedoch erschweren. Die Abbildung 3.5 stellt den Prozess der Einflussanalyse dar.

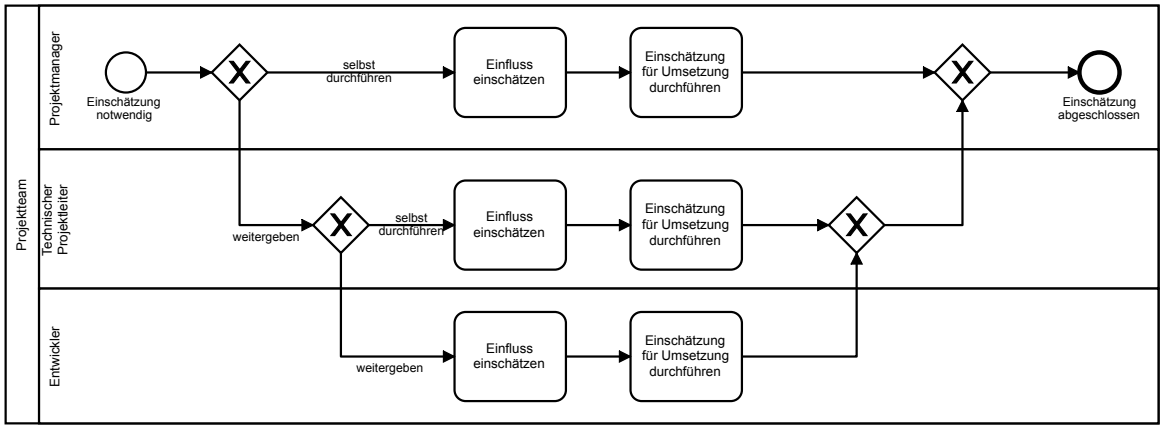

Abbildung 3.5: Prozess: Einflussanalyse

3.2 Übergabeformen

Mit Übergabeformen sind alle Formen von Informationen gemeint, die zwischen den Projektbeteiligten ausgetauscht werden. Angefangen beim Kunden, mit Wünschen und Vorstellungen, die in Protokollen festgehalten und in teilformalisierte Dokumente überführt werden, bis hin zu Tickets mit Aufgaben der Entwickler. Weiterhin sind bei der Übergabe auch die Medien und Speicherorte, sowie Status und Aktualität entscheidend.

Die Formate für die Ermittlung von Anforderung sind abhängig von Projektgröße und Ort. Bei wenigen Informationen vorab sind diese anders als bei großen Ausschreibungen mit vielen Informationen im Vorfeld. Übliche Formate sind Kick-Off Meeting, Schulterblicke, Telefongespräche und E-Mails.

Bei der Durchführung eines Projektes entstehen diverse Dokumente in den verschiedenen Phasen und Gewerken. Auf der Ebene des Gesamtprojektes zählen dazu Angebot, Grob- und Feinkonzept, Briefings und Protokolle. Die digitale Kommunikation erfolgt per E-Mail, wobei einzelne E-Mails ebenfalls eine Art von Dokument darstellen. Darüber hinaus werden jedoch auch XMPP und Mattermost als Textchat verwendet. Im Bereich Design gibt es Design-Briefings, Wireframes, Mockups, Post-its, Styletiles, Mindmaps und Grafiken. Eine Besonderheit stellt der Projektfahrplan dar, da dieser weniger ein konkretes Dokument, sondern eine Anwendung zur Darstellung des Projektstatus ist. Weiterhin werden übergreifend auch verschiedene Tabellenkalkulations- und Textdokumente verwendet. Mittlerweile wird auch viel mit Atlassian Confluence gearbeitet. Aus den Interviews ließ sich dabei jedoch entnehmen, dass dies im Prozess nur bis zur Frontend-Entwicklung, weniger bis gar nicht jedoch bei den Backend-Entwicklern eingesetzt wird. Generell hat die Backend-Entwicklung weniger konkrete Vorstellungen zu verwendeten und erstellten Dokumenten. Die Dokumentation der Anwendungen erfolgt direkt im Quellcode und es werden dazu keine weiteren Dokumente erstellt.

Als Ablageorte werden der Fileserver, aber auch Confluence und SparkleShare verwendet. Dabei gibt es außerdem unterschiedliche Ablageorte und Ordnerstrukturen auf dem Fileserver von Projektmanagement und Design. Ausgehend vom Projektmanagement wird dabei auch ein vollständiger Übergang zu Confluence angestrebt. Auch bei den Ablageorten ging aus den Interviews hervor, dass bei den Backend-Entwicklern nicht immer bekannt ist, welche Dokumente überhaupt existieren und wo sie abgelegt sind. Oftmals ist außerdem die Dateiablage fatal, mit weit verteilten Dokumenten und unklarer Gültigkeit und Status. Eine Struktur und die Verantwortlichkeiten für die Dokumente sind nicht immer erkennbar. Eine erzwungene Aktualität, bspw. über festgelegte Prozesse oder Ablaufdaten, gibt es nicht.

Bei den Informationen vom Kunden ist tendenziell das Erstbriefing für ein Projekt nicht ausreichend um alle nötigen Informationen zu erhalten. Meistens erfordert es viele Nachfragen. Die Fragen ergeben sich jedoch auch oft erst in der Konzeptionsphase. Problematisch ist auch, dass selten technische Expertise beim Kunden vorhanden ist. Für die Backend-Entwicklung sind die Informationen vom Kunden selten nützlich. Häufig ist dem Kunden die Komplexität der in Auftrag gegebenen Anwendung nicht bewusst und dieses Bewusstsein wird durch fehlende Beratungsleistung auch nicht geschaffen. Dies resultiert nicht nur aus der fehlenden Expertise und Beratungsleistung, sondern auch aus der Unkenntnis des Kunden über die Komplexität der eigenen Prozesse. Oftmals erfolgt die Kommunikation mit dem Kunden per E-Mail, was von vielen Entwicklern, aber auch Designern eher abgelehnt wird, da der Kunde die Wünsche nicht entsprechend formulieren kann und schnell Unklarheit darüber besteht, was

genau gemeint ist. Weiterhin gibt es keine Anstrebungen zur Schaffung einer Ubiquitious Language, der gemeinsamen Sprache, wie sie im Domain-Driven Design gefordert wird.

Bei den Dokumenten und Informationen für die Entwickler gibt es verschiedene Ansichten. Von Seiten des Projektmanagements befinden sich mittlerweile alle Dokumente im Confluence, jedoch werden diese nicht von den Entwicklern gelesen. Im wöchentlichen Kick-Off erfolgt die Aufgabenverteilung und Bereitstellung der nötigen Informationen. Anschließend erfolgt die weitere Kommunikation über Office-Dokumente und Redmine Tickets. Aus der Sicht Backend-Entwickler werden jedoch nur wenige bis keine relevanten Dokumente bereit gestellt, die nötigen Informationen kommen vor allem durch die Tickets im Redmine. Dabei sind die Tickets und Dokumente selten oder gar nicht gefiltert und werden nur flüchtig gelesen.

Bezüglich Aktualität von Dokumenten ist vor allem der Projektmanager verantwortlich. Jedoch sind die einzelnen Entwickler für die Doumentation ihrer Anwendungsteile verantwortlich, die jeweils im Quellcode erfolgen. Teilweise werden die Aufgaben zur Sicherstellung der Aktualität und Dokumentation auch vom technischen Projektleiter übernommen. Für die Häufigkeit der Aktualisierung gibt es keinen vereinheitlichten Prozess. Oftmals sind die Dokumente bereits bei der Ablage veraltet und werden nicht mehr aktualisiert. Es gibt regelmäßige Abstimmung über den Status der Dokumente, jedoch ist die Synchronisierung dabei schwierig, da bspw. lokal vorgehaltene Dokumente nicht auf den Fileserver zurück gespielt werden und damit den anderen Projektbeteiligten nicht zur Verfügung stehen. Auch ist die Projektphase entscheidend, welche Dokumente aktualisiert werden und welche nicht. Für die Entwickler ist es nur wichtig, dass relevante Dokumente immer aktuell sind. Außerdem ist, laut Aussage eines Interviewpartners, der Projektmanager der „Single-Point-of-Contact“, welcher sich fehlendes Wissen holen und die Übersicht behalten muss.

Bei den Dokumenten wird die Nutzung der Industrienormen zu Lasten- und Pflichtenheft abgelehnt, teilweise insgesamt, oftmals jedoch nur die Begrifflichkeit, um daraus resultierende Verpflichtungen zu vermeiden. Für die aktuellen Kunden ist eine Festlegung auf Industrienormen nicht relevant und mit den derzeitigen eigenen Prozessen nicht umsetzbar. Außerdem müssten entsprechende Kompetenzen geschaffen werde, da diese nicht vorhanden sind. Die Verwendung standardisierter Prozesse wird als positiv angesehen.

3.3 Projektmanagement

Dieser Bereich umfasst die Eigenheiten des Projektmanagements im Unternehmen, bezogen auf den Rahmen des Themas.

Die generelle Methodik ist ein agiles Vorgehen ohne genaue Definition, enthält jedoch auch Elemente des Wasserfall-Modells, wie die umfassende Konzeptionsphase mit Konzeptionsdokumenten und festem Projektbudget. Dabei ließ sich aus den Interviews entnehmen, dass die Steuerung teilweise unklar und viel zusätzliche Kommunikation notwendig ist. Die Wahrnehmung im Bereich Design und Frontend-Entwicklung sieht die Methodik agil bis zum Feature-Freeze und der eigentlichen Umsetzung, da mit dem Kunden über mehrere Runden hinweg das Design entwickelt wird. Anschließend verläuft die Umsetzung eher starr bis zum Projektende.

Das Projektcontrolling erfolgt dabei weniger im großen Umfang, sondern eher nach Zeit und Kosten. Teilweise werden dabei in der technischen Umsetzung nur die Stunden erfasst. Durch die verpflichtende Nutzung von Revolver als Zeiterfassungssystem für alle Mitarbeiter könnte jedoch in Zukunft ein präziseres Projektcontrolling erfolgen. Aus Sicht einiger Projektmanager verursacht Konzeption und Entwicklung gleich viel Aufwand wie die Entwicklung, Letztere wird jedoch wesentlich höher budgetiert. Vor allem bei kleinen Projekten wird in der Konzeptionsphase gespart und eine direkte Umsetzung angestrebt.

Die Verantwortlichkeit für das Anforderungsmanagement liegt für die überwiegende Mehrheit beim Projektmanagement, jedoch auch beim technischen Projektleiter für die technischen Anforderungen. Hier wurde jedoch auch die Bestrebung geäußert, diese Verantwortung an Konzepter zu übergeben.

Die nachträgliche Validierung der unpräzisen Anforderungen dient der Beseitigung von Unklarheiten während der Konzeptionsphase. Nach Aussagen des Projektmanagements ist dies abhängig von Budget, Aufwand und der jeweiligen Anforderung. Die Entscheidung obliegt dabei beim Projektmanager. Die Validierung ist nicht kalkuliert und wird daher auch nicht bezahlt. Bei Unklarheiten wird, nach Aussage der Projektmanager, eigentlich immer gefragt. Es gibt eine Grenze der Eigengestaltung für die Entwickler, dabei ist jedoch die Größe des Projekts entscheidend. Außerdem ist ein entscheidender Punkt, ob der Kunde schon eine konkrete Vorstellung hat oder freie Hand lässt.

Im Unterschied zur Validierung, die sich mit dem Abgleich von Anforderung und Kundenwunsch befasst, dient die Verifikation der Anforderungen dazu, die Anforderung mit der Umsetzung abzugleichen. Aus den Interviews ließ sich entnehmen, dass dies als Teil des Qualitätsmanagement zwar teilweise durchgeführt, jedoch eher stiefmütterlich behandelt und nicht kalkuliert wird. Eine Prüfung auf Basis des Konzepts wird, nach anderer Aussage, gar nicht durchgeführt. Meistens wird die Qualitätssicherung vom Kunden oder den jeweiligen Mitarbeitern der Entwicklung selbst übernommen. Mängel müssen nachträglich beseitigt werden, wenn sie vom Kunden festgestellt werden. Die Backend-Entwickler erhalten keine Anforderungsdokumente, auf dessen Basis sie eine Verifikation durchführen könnten. Die Informationen in den Tickets sind dazu nach ihren Aussagen nicht geeignet, da sie redundante oder widersprüchliche Informationen enthalten und die Aufgabenstellung zu unpräzise ist, um daraus eine zu verifizierende Anforderung abzuleiten. Die Backend-Entwickler sehen sich weierhin

nicht dafür zuständig, wie es vom Kunden gewollt ist, sondern was laut Aufgabenbeschreibung umgesetzt werden soll. Für die konkrete Prüfung ist eine Testabteilung notwendig. Oftmals werden Anforderungen im laufenden Prozess verifiziert, was den Unterschied zwischen den im Confluence hinterlegten Anforderungen und den Anforderungen des Kunden darstellt. Weiterhin gab es auch die Aussage, dass ein vom Kunden abgenommenes Projekt nicht mehr verifiziert werden muss.

Ein Punkt der aus dem professionellen Anforderungsmanagement kommt, ist die Nachvollziehbarkeit von Anforderungen. Nach den Aussagen der Interviewpartner ist diese nur bedingt gegeben, aber nicht in dem Umfang, der für eine Nachvollziehbarkeit von Beginn der ersten Kundengespräche, bis zur Implementierung notwendig wäre. Problematisch ist bereits, dass es nicht für alle Meetings ein Protokoll gibt, auf das man sich beziehen könnte. Durch die fehlende direkte Verbindung zwischen den vorhandenen Protokollen, welche in einfachen Textdokumenten oder im Confluence erstellt werden und den Tickets im Redmine, ist die Nachvollziehbarkeit auch nur indirekt gegeben. Bei Absprachen am Telefon werden die Tickets meist direkt erstellt, ohne weitere Protokollierung vorzunehmen. Auch durch den Änderungsmodus von Office Dokumenten und die Historie von Tickets und Confluence lassen sich die Anforderungen nicht vollständig nachvollziehen. Auch an diesem Punkt wird von der Backend-Entwicklung bemängelt, dass Anforderungen selten bis gar nicht nachvollziehbar sind.

In direkter Verbindung zur Nachvollziehbarkeit steht auch die Änderungshistorie von Anforderungen. Dies geschieht laut Projektmanagement nur auf Dokumentenebene, nicht auf der Ebene einzelner Anforderungen und auch nicht im weiteren Detaillierungsgrad. Auf technischer Ebene werden Änderungen über das Versionskontrollsystem aufgenommen. Während der Umsetzung werden selten Änderungen im Feinkonzept vorgenommen. Im Bereich Design wird die Versionierung durch die Benennung der Ordnerstruktur vorgenommen. Auch in der Frontend-Entwicklung lassen sich Änderungen nur durch Kombination von Confluence und Versionierung im Git nachvollziehen.

Um den Stand von Anforderungen zu einem bestimmten Zeitpunkt festzusetzen, werden Baselines erzeugt. Als einzige Baseline in derzeitigen Projekten kann das Feinkonzept angesehen werden, da es eine Auflistung von Anforderungen mit zum Zeitpunkt der Erstellung des Feinkonzepts schriftlich festhält. Im späteren Verlauf werden durch Releaseplanung nur weitere Funktionen und Fehlerbeseitigung geplant und terminiert, jedoch keine konkreten Anforderungen in Baselines festgehalten. Ohne konkrete Baseline ist auch die Festlegung, was einen Change Request darstellt nicht immer eindeutig. Daher sollen neben dem Feinkonzept auch die Funktions- und Bedienungsanleitung als Grundlage dienen, um Change Requests zu erkennen. Zu den daraus resultierenden Nachverhandlungen gibt es unterschiedliche Aussagen. Generell betrifft es mehr Faktoren als offensichtliche Änderungen der Anforderung, wie Projektgröße, Kunde und Einfluss der Änderungen.

3.4 Reifegrad

In diesem Abschnitt soll der aktuelle Zustand mit den in CMMI-DEV beschriebenen Praktiken und Ergebnissen abgeglichen werden. Für den weiteren Verlauf lassen sich daraus Empfehlungen ableiten, die zu besseren Prozessen führen sollen.

Die Anforderungsentwicklung beginnt mit dem spezifischen Ziel der Entwicklung von Kundenanforderungen. Die dazu geforderten Schritte der Ermittlung von Bedürfnissen und anschließender Überführung in relevante Kundenanforderungen werden durchgeführt, jedoch lässt sich aus den Interviews nicht erkennen, in welchem Umfang dies geschieht. Auch CMMI-DEV gibt an dieser Stelle noch keine konkrete Antwort darauf. Das folgende spezifische Ziel ist die Entwicklung der Produktanforderungen. Die spezifischen Praktiken werden bedingt erfüllt, da weder eine Erfassung kritischer Qualitätsattribute, noch Qualitätskennzahlen entwickelt oder Beziehungen zwischen den Anforderungen etabliert und dokumentiert werden. Auch die Identifikation von Schnittstellenanforderungen findet größtenteils erst bei der Umsetzung durch den jeweiligen Entwickler statt. Die nachfolgende Analyse und Validierung von Anforderungen stellt ein weiteres spezifisches Ziel dar, das ebenfalls nur als teilweise erfüllt angesehen werden kann. An dieser Stelle werden keine Anwendungsszenarien etabliert, wie sie über Aktivitätsdiagramme, Anwendungsfälle oder User Stories dargestellt werden könnten. Die anschließende Analyse der Anforderungen soll feststellen, ob Anforderungen den übergeordneten Zielen der Stakeholder genügen und in sich vollständig, durchführbar, umsetzbar und verifizierbar sind. Dies ließ sich aus dem aktuellen Prozess heraus nicht feststellen. Durch das Fehlen definierter Qualitätsanforderungen mit entsprechenden Messkriterien ist auch eine Bewertung der Kosten und Risiken, welche durch den Einfluss der Qualitätsanforderungen entstehen, nicht durchführbar. Eine Validierung der Anforderungen unter Abstimmung mit den Stakeholdern findet nur teilweise statt, da vollständige Modelle und Anwendungsszenarien zum Zeitpunkt der Konzeption nicht existieren.

Das Anforderungsmanagement hat lediglich die Verwaltung von Anforderungen als spezifisches Ziel. Als spezifische Praxis sollen zunächst die Anforderungen verstanden werden. Die dazu erforderlichen Unterscheidung zugelassener Anforderungsgeber, sowie die Bewertung und Annahme von Anforderungen und dessen Kriterien werden nur bedingt erhoben. Auch die Zusage für Anforderungen ist nicht vollständig nach CMMI-DEV erfüllt. So kann eine Einflussanalyse ohne vollständige Nachvollziehbarkeit nur eingeschränkt oder mit viel Mehraufwand zur Rekonstruktion der Verbindungen zwischen den Anforderungen durchgeführt werden. Auch eine Dokumentation sämtlicher Zusagen zu Anforderungen findet nicht immer statt, was aus den Interviews hervor ging. Dies gilt auch für die Verwaltung von Anforderungsänderungen, da Aufgaben ohne Anpassung der Anforderungen direkt in die Umsetzung gegeben werden. Eine Anforderungsdatenbank, wie sie CMMI-DEV als Arbeitsergebnis für die Dokumentation von Anforderungen, Anforderungsänderungen, deren Begründung und Anforderungsstatus vorschlägt, existiert nicht. Die Aufrechterhaltung bidirektionaler

Nachverfolgbarkeit von Anforderungen wird nicht durchgeführt. Auch der Abgleich von Anforderungen und der Projektarbeit ist nur bedingt erfüllt, wird jedoch nicht proaktiv als dieser im Prozess des Anforderungsmanagements umgesetzt, sondern wird im Projektmanagement mitgeführt.

Bei der Einordnung in den Fähigkeitsgrad nach CMMI würde durch die unvollständige Erfüllung der spezifischen Ziele in beiden Prozessgebieten nur der Fähigkeitsgrad „Unvollständig“ erreicht werden.

4 Empfehlungen für zukünftiges Vorgehen

Dieses Kapitel beschäftigt sich mit den Empfehlungen für das zukünftige Vorgehen. Die Optimierung von Prozessen und Organisationsstrukturen steht dabei im Vordergrund. Die Empfehlungen basieren auf dem im vorherigen Kapitel beschriebenen Zustand, welcher auf Basis von Interviews und Erfahrungen aus dem Unternehmen erhoben wurde.

4.1 Prozess

Zu Beginn der Projektarbeit muss eine Heranführung des Kunden an die Arbeit innerhalb der Agentur, aber auch an die Arbeit mit Anforderungen stattfinden. Der Kunde muss im Rahmen der Beratungsleistungen durch den eigenen Prozess geführt werden. Dabei muss dem Kunden auch klar gemacht werden, dass es sich nicht um die simple Umsetzung einer Website, sondern um ein komplexes Softwareprodukt handelt.

Die Anforderungen für die zu entwickelnde Extension müssen bereits vor der Angebotserstellung für die endgültige Umsetzung aufgenommen werden. Dazu muss neben der derzeitigen Feinkonzeption ein Anforderungsdokument entstehen, welches die Anforderungen atomar darstellt und als erste Baseline zu sehen ist. Wichtiges Kriterium sollte hierbei in Zukunft auch der späteste Projektbeginn sein, um die Umsetzung fristgerecht und mit voller Kundenzufriedenheit fertigstellen zu können. Die Abbildung 4.1 stellt hierbei den Prozess der Anforderungsermittlung dar, welcher durch die neuen Erkenntnisse erweitert wurde.

Für die Abdeckung der konzeptionellen Aufgaben bei der Entwicklung von Extensions ist die Erweiterung der Rollendefinition notwendig. Dazu wäre die Position des Software Engineers mit der Spezialisierung auf Web Anwendungen oder gleichbedeutend des Web Engineers in den Entwicklungsprozess zu implementieren. Das derzeitige Rollenbild der Technischen Projektleiter ist eine Vermengung der Aufgaben von Projektmanagement, Product Owner, Software Engineer und Entwickler, ohne klar definierte Trennlinien. Hier muss zukünftig auf einer Ebene differenziert werden, die nicht im Speziellen die Entwicklung von Extensions betrifft, sondern den generellen Projektablauf.

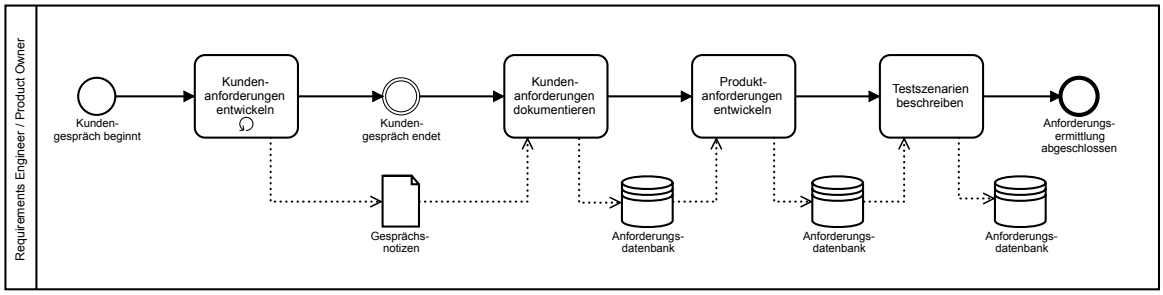

Abbildung 4.1: Prozess: Anforderungsermittlung mit neuen Erkenntnissen

Weiterhin ließ sich erkennen, dass die konzeptionelle Vorarbeit für die optischen Bestandteile und die Bedienung durch den Nutzer wesentlich ausgereifter durchgeführt wird, als die Ausarbeitung von Abläufen oder Modellen. Ähnlich wie die Workshops des Design zu den optischen Bestandteilen des Produkts, müssen auch Workshops zur Geschäftslogik mit dem Kunden durchgeführt werden. Dazu muss das Domain-Model gemeinsam mit den Experten der Fachdomäne entwickelt werden, so wie es Evans für das Domain-Driven Design fordert. Eine Zusammenarbeit fordert nicht nur die intensive Auseinandersetzung mit der Fachdomäne, sondern auch die Entwicklung einer gemeinsamen Sprache. Des Weiteren wird bei der Entwicklung in Zusammenarbeit die Verständlichkeit beim Kunden erhöht, da dieser am Entstehungsprozess beteiligt war.

Mittelfristig muss ein geeignetes Tool für das Anforderungsmanagement evaluiert werden, welches nicht nur im Rahmen der Extension Entwicklung, sondern auch in anders gearteten Projekten zum Einsatz kommt. Als Grundlage kann die „Vergleichsstudie zu Werkzeugen für das Anforderungsmanagement“ von Studenten der Wirtschaftsinformatik an der DHBW-Stuttgart dienen. [Hahn 2014] Durch das vorhandene Confluence wäre dabei Jira, welches ebenfalls von Atlassian stammt und eine gute Integration mit Confluence bietet, eine mögliche Lösung.

Die Nutzung von speziellen Anwendungen für die Durchführung des Anforderungsmanagements hilft auch bei der Sicherstellung von Nachvollziehbarkeit. Diese ist essenziell um zukünftig Einflussanalysen vor der Umsetzung von Änderungen durchzuführen. Ohne vorherige Sicherstellung der Nachvollziehbarkeit ist dies jedoch nur mit viel Mehraufwand möglich. Die Abbildung 4.2 stellt den Prozess der Einflussanalyse dar, welcher um die neuen Erkenntnisse erweitert wurde.

Bei Änderungsanfragen muss neben der Einflussanalyse auch die generelle Frage nach der Machbarkeit bei den Entwicklern als selbstverständlich gelten. Dabei muss ggf. über einen Kompromiss die Machbarkeit mit dem Wunsch des Kunden in Einklang gebracht werden. Hierbei ist auch das Releasemanagement zu beachten. Dabei müssen Änderungen die auf ein späteres Release verschoben werden, wenn sie derzeitig nicht

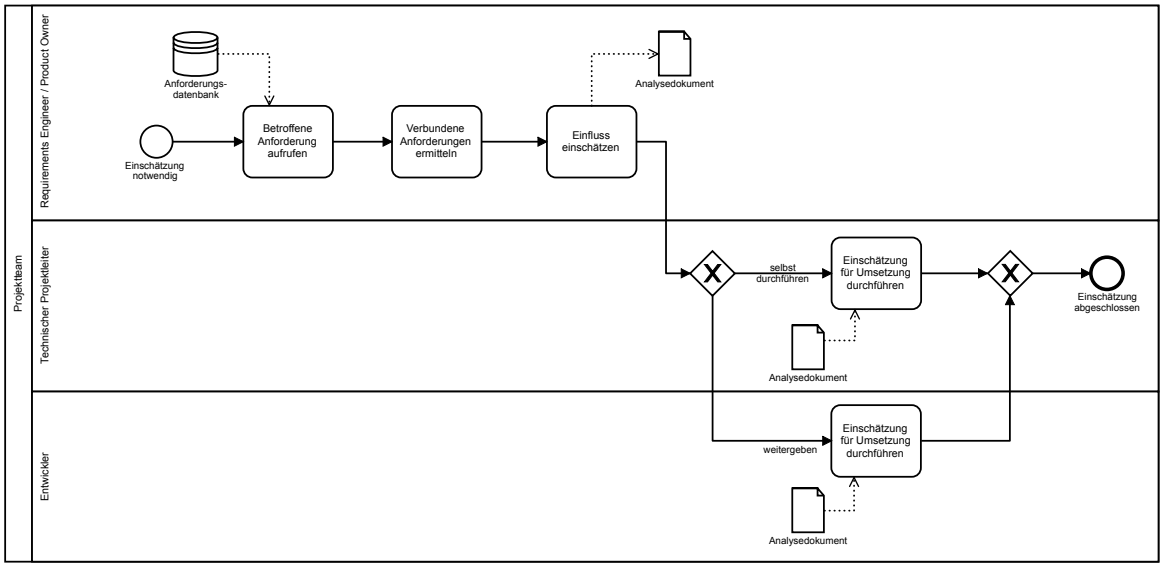

Abbildung 4.2: Prozess: Einflussanalyse mit neuen Erkenntnissen

umsetzbar sind. Die Abbildung 4.3 stellt den Prozess der Änderungsanfrage dar, welcher um die neuen Erkenntnisse erweitert wurde.

Die Beschreibung von Testfällen sollte zukünftig bereits zusammen mit der Beschreibung der Anforderungen erfolgen. Eine Anforderungsbeschreibung mittels User Stories erfordert bspw. die Definition von Akzeptanzkriterien, auf dessen Basis die Entwickler testen und die Qualitätssicherung prüfen können.

Vor der Umsetzung einer Aufgabe durch den Entwickler muss das Ticket im Ticketsystem aufgearbeitet werden. Dabei muss das Ticket alle zur Lösung des Problems notwendigen Informationen in einer übersichtlichen Darstellungsweise enthalten. Dies kann neben einer textuellen Beschreibung auch die grafische Darstellung von Domain-Models in UML Klassendiagrammen, ER-Diagramme oder Prozessmodelle enthalten. Auf zusätzlich notwendige Dokumente muss verwiesen werden, besser wäre jedoch, wenn die darin enthaltenen Informationen ebenfalls bereits zusammengefasst wurden. Auch die Diskussionsteile müssen aufgearbeitet werden, um alle wichtigen Zusatzinformationen zu extrahieren und in die ursprüngliche Aufgebenstellung einzuarbeiten. Insbesondere bei der Übergabe an bis dahin noch nicht am Projekt beteiligte Entwickler vermeidet eine Aufarbeitung evtl. Missverständnisse und daraus resultierende Mehrarbeit. Weiterhin müssen Tickets einen Verweis auf ihren Ursprung enthalten, mit Angaben zu Medium, Beteiligten und Zeitpunkt.

Neben dem Vorgehen bei der Entwicklung von Extensions sollten auch andere Prozesse in Zukunft festgehalten werden. Dies ermöglicht eine bessere Übersicht zu bestehenden Prozessen und ein Verständnis über das Gesamtsystem. Des Weiteren wird dadurch

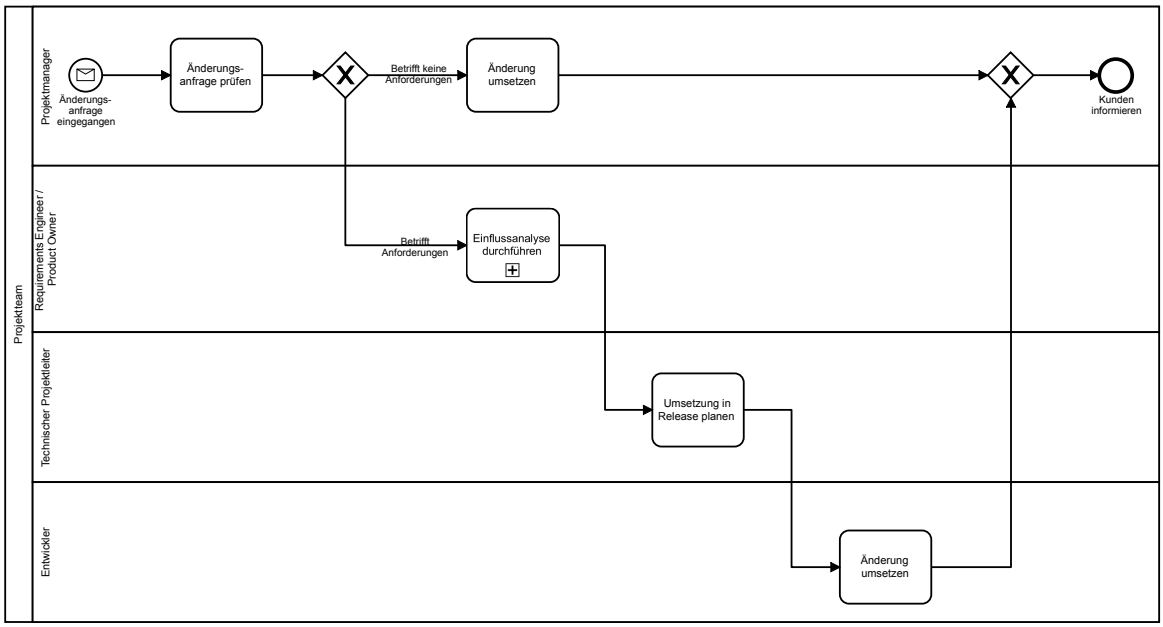

Abbildung 4.3: Prozess: Änderungsanfrage mit neuen Erkenntnissen

das Onboarding neuer Mitarbeiter erleichtert, da sich diese bereits vor Beginn ihrer Arbeit einen Überblick zu ihren Tätigkeiten verschaffen können.

4.2 Übergabeformen

Die Feinkonzeption ist derzeitig auf den Kunden ausgerichtet und nicht auf die umsetzende Abteilung. Daher ist es empfehlenswert die Anforderungsdokumentation als eigenständiges Dokument von der Feinkonzeption zu lösen. Eine Anforderungsdokumentation benötigt keinen auf den Kunden ausgerichteten Marketing-Charakter. Sie sollte zukünftig als rechtsverbindliche Grundlage für die Umsetzung dienen. Dabei können spätere Änderungen der Anforderungen durch den Kunden wesentlich leichter neu fakturiert werden. Dies unterstützt auch beim Projektcontrolling, da alle Abweichungen von den usprünglich definierten Anforderungen atomar sichtbar sind. Mit der notwendigen Erhaltung der Nachverfolgbarkeit lassen sich somit sämtliche Zeiten und Kosten für jede Anforderungsänderung nachvollziehen.

Die Ablageorte für einzelne Projekte müssen vereinheitlicht werden. Es gibt derzeit eine Vielzahl von Ablageorten und Dateiformaten, ohne einheitlich verbindliche Vorlagen oder Vorgaben. Auch der letzte Bearbeitungsstand und Status des Dokumentes müssen immer ersichtlich sein. Dazu bieten sich zu Beginn des Dokuments die Einführung von Änderungshistorien und Referenzangaben an. Zuständig für die Pflege

und Aktualisierung von Projektdokumenten ist das Projektmanagement. Die Verantwortlichkeit für einzelne Dokumente kann zwar abgegeben, muss jedoch trotzdem durch das Projektmanagement begleitet werden. Wenn Confluence als führendes System eingesetzt werden soll, muss dies ausreichend bei allen Mitarbeitern kommuniziert werden, um Missverständnisse bei den Ablageorten zu vermeiden.

Bei der Zusammenarbeit mit externen Unternehmen müssen Schnittstellen in einem annehmbaren Zustand dokumentiert sein. Auch dies muss in Zukunft einen Teil der Anforderungsbeschreibung darstellen.

Zwar gibt es eine generelle Ablehnung gegenüber Industrienormen für die Arbeit nach Lasten- und Pflichtenheft, jedoch ist eine sinnvolle Struktur von Dokumenten erforderlich. Durch die Erarbeitung eines Hausstandards als Vorlage für die Erstellung verschiedener Dokumente, wie etwa Grob- und Feinkonzept, Protokolle, Softwaredokumentation und Anforderungsdokumente wird ein schneller Einstieg neuer Mitarbeiter gewährleistet und ein klares Gesamtbild geschaffen. Außerdem unterstützen die Strukturen bei Erstellung und verringern die Wahrscheinlichkeit, dass wichtige Informationen vergessen werden, da sie von der Vorlage eingefordert werden.

4.3 Projektmanagement

Für das derzeitig agile Vorgehen ohne genaue Definition sollte langfristig eine Annäherung an bestehende Vorgehensmodelle stattfinden. Ansätze dazu finden sich bereits in der Einführung des Redmine Agile Plugin im Redmine Ticketsystem, welches eine Ausrichtung nach Scrum erlauben würde. Dabei wäre es auch möglich zunächst durch die Einführung von Kanban eine stetige Verbesserung der Entwicklungsprozesse zu erreichen, während das finale Ziel eine Ausrichtung nach Scrum ist.

Aus den Interviews ging hervor, dass bereits Projektcontrolling durchgeführt wird. Die unterschiedliche Tiefe des Projektcontrollings durch das Projektmanagement muss zukünftig abgeglichen werden. Dazu sollte ein Austausch zwischen den Projektmanagern stattfinden. Die genauerer Erfassung und spätere Auswertung kann zu einer Verbesserung der eigenen Prozesse führen.

Das Anforderungsmanagement muss zukünftig an einen ausgewiesenen Mitarbeiter übertragen werden, damit sich der Projektmanager auf die eigenen Aufgaben konzentrieren kann. Die Ausrichtung dieser Rolle muss jedoch übergeordnet zur Entwicklung von Extensions stattfinden und bezieht sich dabei auch auf andere Projekte. Möglich wären dabei die Rollen des Product Owners oder des Requirements Engineer.

Auch eine Verifikation der Anforderungen muss zukünftig durchgeführt werden. Dies ist als Aufgabe der neuen Rolle in Zusammenarbeit mit dem Qualitätsmanagement

durchzuführen. Weiterhin lässt sich dies bereits in der Umsetzung durch den Entwickler testen, wenn bspw. bei User Stories die entsprechenden Akzeptanzkriterien angegeben sind.

Die Protokollierung aller Meetings und Gespräche muss sich zu einem Standard etablieren. Für die Sicherstellung der Nachvollziehbarkeit müssen auch von Telefonaten schriftliche Nachweise geführt und abgelegt werden, auf die sich später Anforderungen, Anforderungsänderungen und die daraus resultierenden Aufgaben beziehen. Die Ablage muss dabei, zusammen mit allen anderen Dokumenten, im Confluence erfolgen.

Die Releaseplanung ist über das Konfigurationsmanagement direkt mit den Baselines verbunden, daher sollte die Festlegung von Baselines auch in Absprache mit der Releaseplanung geschehen. Wenn durch Änderungsanfragen eine gewisse Anzahl von Anforderungsänderungen stattgefunden haben, wird daraus ein Release erzeugt, welches im Anforderungsmanagement eine neue Baseline darstellt. Die anschließende Entscheidung über eine nachträgliche Fakturierung von Änderungen obliegen dem Projektmanagement und ggf. der Geschäftsleitung. Mit diesem Vorgehen sind jedoch Mehrkosten für die Änderungen besser ersichtlich, da sie von der Anfrage zur Anforderungsänderung bis zur Umsetzung nachvollzogen werden können.

4.4 Reifegrad

Die Empfehlungen zum Reifegrad wurden nach kurz- und mittelfristigen Anpassungen gegliedert. Dabei soll langfristig jedoch der höchste Fähigkeitsgrad „Definiert“ nach CMMI in den Prozessbreichen Anforderungsentwicklung und Anforderungsverwaltung erreicht werden.

Zunächst wurden die kurzfristigen Änderungen bei der Anforderungsentwicklung aufgestellt. Dazu zählt die Aufnahme von Qualitätsattributen bei der Entwicklung von Produktanforderungen. Dies wird derzeit nur unzureichend umgesetzt, was schon ersichtlich wird wenn man bestehende Dokumente nur mit den Qualitätsmerkmalen nach ISO/IEC 25010 abgleicht. [ISO 25010:2011 E] Die Schnittstellenanforderungen müssen bereits bei der Entwicklung der Produktanforderungen identifiziert werden. Die Herstellung von Beziehungen zwischen den Anforderungen sollte ein kurzfristiges Ziel sein, steht aber in Verbindung mit dem mittelfristigen Ziel zur Anschaffung eines geeigneten Tools für das Anforderungsmanagement. Daher sollte als kurzfristiges Ziel die verpflichtende Einführung von Querverweisen für den Ursprung von Tickets eingeführt werden und erst auf mittelfristige Sicht die vollständige Nachvollziehbarkeit hergestellt werden. Daran schließen sich die mittelfristigen Anpassungen an, welche eine striktere Trennung von Kunden- und Produktanforderungen fordert. Dazu werden zunächst Bedürfnisse aufgenommen, welche dann detailiert in Kundenanforderungen überführt und dokumentiert werden. Dafür muss auch das Verständnis

für unterschiedliche Darstellungsweisen, wie Aktivitäts- und Klassendiagramme, aber auch User Stories erhöht werden, die bei der Beschreibung von Kundenanforderungen zum Einsatz kommen sollen. Zuletzt muss auch die Validierung der dann atomaren Anforderungen im Anforderungsmanagement Tool mit den Stakeholdern validiert werden.

Bei der Anforderungsverwaltung sollte zunächst auf kurzfristige Sicht eine ausführlichere Darstellung der Stakeholder erfolgen. Diese Darstellung kann durch die bereits vorhandenen Personas ergänzt werden. Neben der erforderlichen generellen Protokollierung von Gesprächen ist auch die Protokollierung von Zusagen zu Anforderungen noch einmal besonders hervor zu heben. Auf mittelfristige Sicht soll eine Bewertung und Annahme von Anforderungen, sowie deren Zusage durch den Anforderungsgeber durchgeführt werden. Die Einführung einer Anforderungsdatenbank steht ebenfalls im Zusammenhang mit der Nutzung eines Anforderungsmanagement Tools. Damit verbunden ist anschließend auch die bidirektionale Nachverfolgung möglich, sowie die detailierte Verwaltung von Anforderungsänderungen.

5 Fazit

Die vorliegende Arbeit befasste sich mit der Frage, wie anhand von Requirements Engineering der Prozess der CMS Individualentwicklung am Beispiel von TYPO3 Extensions verbessert werden kann.

Dazu wurde zunächst der Prozess durch Interviews mit Mitarbeitern, bestehende Dokumente und Erfahrungen aus dem Projektalltag aufgenommen und analysiert. Dazu wurden die Bereiche Prozess, Übergabeformen, Projektmanagement und Reifegrad betrachtet. Dabei wurde festgestellt, dass es bisher keine ausführliche Dokumentation der bestehenden Prozese gibt. Das Vorgehen erfolgt auf eine nicht näher bestimmte agile Weise und nicht nach einem bekannten Vorgehensmodell aus der Softwareentwicklung. Es gibt viele implizite Annahmen, statt klar definierter Anforderungen und dementsprechend auch keine atomare Anforderungsdokumentation. Für das Anforderungsmanagement ist der Projektmanager, neben seiner eigentlichen Tätigkeit, zuständig. Auch eine Validierung und Verifikation der Anforderungen findet nicht in geordneter Weise statt. Die verschiedenen Dokumente und Ablageorte in den Projekten, sowie unklarer Status und Verantwortlichkeiten führen zu Unsicherheit bei den Projektbeteiligten.

Anschließend wurden Lösungen und mögliche Lösungsansätze für die bestehenden Probleme ermittelt. Dazu gehören die Erweiterung des Rollenkonzeptes um einen Product Owner oder Requirements Engineer, welcher die Aufgaben des Anforderungsmanagements vom Projektmanager übernimmt und die Erweiterung des Aufgabenspektrums des Technischen Projektleiters an den Aufgaben des Software Engineers, für die konzeptionelle Vorarbeit für die Umsetzung der Geschäftslogik. Weiterhin soll die Übergabe von Informationen und Ablage von Dateien zukünftig weiter an Confluence ausgerichtet werden. Zuletzt erfolgte die Definition von kurz-, mittel- und langfristigen notwendigen Anpassungen der Prozesse zur Erreichung der Forderungen von CMMI-DEV in der Anforderungsentwicklung und Anforderungsverwaltung.

Der Geltungsbereich dieser Arbeit bezieht sich lediglich auf die im Unternehmen angewandten Prozesse und Vorgehensweisen. Erkenntnisse die über den aktuellen Wissensstand in den Bereichen Requirements oder Software Engineering hinaus gehen und für die Wissenschaft relevant sind, wurden dabei nicht erlangt.

Zusammenfassend lässt sich erkennen, dass es diverse Verbesserungspotenziale in der CMS Individualentwicklung innerhalb des Unternehmens gibt, welche sich durch Requirements Engineering lösen lassen. Jedoch spielen auch Faktoren aus anderen Bereichen, wie dem Software Engineering, eine Rolle. Daher ist eine isolierte Betrachtung

zwar möglich, aber erst im Gesamtzusammenhang mit angrenzenden Bereichen sinnvoll.

Die bisherigen Erkenntnisse dieser Arbeit können genutzt werden, um die Prozesse des Unternehmens zu verbessern, jedoch steht diese Entwicklung erst am Anfang und Bedarf noch weiterer Optimierung um langfristig erfolgreich zu sein.

Literaturverzeichnis

[BuiltWith Pty Ltd 2017] BuiltWith Pty Ltd: *TYPO3 Usage Statistics*. 2017. – https://trends.builtwith.com/cms/TYPO3 (Abruf am 13.01.2017)

[CMMI Product Team 2011] CMMI Product Team: CMMI®für Entwicklung, Version 1.3 / Carnegie Mellon University. https://www.sei.cmu.edu/library/assets/whitepapers/10tr033de_v11.pdf, November 2011. – Forschungsbericht

[Evans 2015] Evans, Eric: *Domain-Driven Design Reference - Definitions and Pattern Summaries*. Dog Ear Publishing (Indianapolis), 2015.

[Foster 2014] Foster, Elvis C.: *Software Engineering - A Methodical Approach*. 1. Aufl. Apress (New York), 2014.

[Grande 2014] Grande, Marcus: *100 Minuten für Anforderungsmanagement - Kompaktes Wissen nicht nur für Projektleiter und Entwickler*. 2., aktualisierte Auflage. Springer Vieweg (Berlin Heidelberg New York), 2014.

[Hahn 2014] Hahn, Kevin ; Hajdu, Matthias ; Kocbinar, Yasemin ; Schacht, Julika: Vergleichsstudie zu Werkzeugen für das Anforderungsmanagement - Open Source und kommerzielle Softwarelösungen im Vergleich / Kompetenzzentrum Open Source der DHBW-Stuttgart. 2014. – Forschungsbericht

[Hruschka 2014] Hruschka, Peter: *Business Analysis und Requirements Engineering - Produkte und Prozesse nachhaltig verbessern*. Carl Hanser Verlag (München), 2014.

[IEEE 1993] IEEE: IEEE Recommended Practice for Software Requirements Specifications / IEEE. https://www.utdallas.edu/~chung/RE/IEEE830-1993.pdf, DEC 1993. – Forschungsbericht

[ISO 25010:2011(E) 2011] ISO: Systems and software engineering – Systems and software Quality Requirements and Evaluation (SQuaRE) – System and software quality models / International Organization for Standardization. , 2011. – Standard

[Koch 2004] Koch, Nora ; Escalona, M. J.: Requirements Engineering for Web Applications – A Comparative Study. In: *Journal of Web Engineering, Vol. 2, No.3 (2004) 193-212* (2004). – http://nexus.hs-bremerhaven.de/Library.nsf/a249ddae15ac617ac12573460029d00b/62f5dd6375f75795c12573620056beef/$FILE/KochEscalonaRE_forWeb_comparativeStudy.pdf

[Krause 2016] Krause, Marcus: *T3census - TYPO3 CMS census*. 2016. – http://t3census.info/ (Abruf am 17.12.2016)

[Kurfürst 2016] Kurfürst, Sebastian ; Rau, Jochen: *Developing TYPO3 Extensions with Extbase and Fluid*. 2016. – https://docs.typo3.org/typo3cms/ExtbaseFluidBook/latest/2-BasicPrinciples/2-Domain-Driven-Design.html (Abruf am 14.12.2016)

[Leffingwell 2000] Leffingwell, Dean ; Widrig, Don: *Managing Software Requirements - A Unified Approach*. 1. Edition. Addison-Wesley Professional (Boston), 2000.

[Mendes 2006] Mendes, Emilia ; Mosley, Nile: *Web Engineering*. Springer, 2006

[Murugesan 1999] Murugesan, San ; Deshpande, Yogesh ; Hansen, Steve ; Ginige, Athula: Web engineering: A New Discipline for Development of Webbased systems. In: *Web Engineering - Managing Diversity and Complexity of Web Application Development* (1999), S. 1–9

[Overmyer 2000] Overmyer, S.P.: What's Different about Requirements Engineering for Web Sites? In: *Requirements Engineeering* (2000)

[Rupp 2014] Rupp, Chris ; SOPHISTen, die: *Requirements-Engineering und -Management - Aus der Praxis von klassisch bis agil*. 6., aktualisierte und erweiterte Auflage. Carl Hanser Verlag (München), 2014.

[Schatten 2010] Schatten, Alexander ; Demolsky, Markus ; Winkler, Dietmar ; Biffl, Stefan ; Gostischa-Franta, Erik ; Östreicher, Thomas: *Best Practice Software-Engineering - Eine praxiserprobte Zusammenstellung von komponentenorientierten Konzepten, Methoden und Werkzeugen*. 1. Auflage. Spektrum Akademischer Verlag (Heidelberg), 2010.

[TYPO3 Association 2015] TYPO3 Association: *The History of TYPO3*. 2015. – https://typo3.org/about/the-history-of-typo3/ (Abruf am 13.01.2017)

[TYPO3 Association 2017] TYPO3 Association: *Key Features*. 2017. – https://typo3.org/typo3-cms/key-features/ (Abruf am 13.01.2017)

[TYPO3 Documentation Team 2015] TYPO3 Documentation Team: *TYPO3 Core APIs*. 2015. – https://docs.typo3.org/typo3cms/CoreApiReference/latest/_pdf/PROJECT.pdf (Abruf am 23.12.2016)

[Wilde 2006] Wilde, Thomas ; Hess, Thomas: Methodenspektrum der Wirtschaftsinformatik: Überblick und Portfoliobildung / Institut für Wirtschaftsinformatik und Neue Medien der Ludwig-Maximilians-Universität München. 2006. – Forschungsbericht

Anhang

- Gesprächsprotokoll - Dan Bechard
- Fragenkatalog für das Interview
- Interviews 1 - 10

Gesprächsprotokoll, vom 08.12.2016

Dan Bechard, BSc. Computer Science
Software Engineer

[11:03 AM] Julian Fastnacht: Someone an idea why Web Engineering was a term about 10 years ago which almost completely vanished today?
[8:54 PM] Dan Bechard: @Julian Fastnacht Because the web has become ubiquitous at this point and re-engineering it would would be a massive undertaking and break everything. Most of the low-level web stuff is pretty standardized and rigid for obvious compatibility reasons.(edited)
[9:04 PM] Julian Fastnacht: @Dan Bechard I'm talking more about the term. Web Engineering was meant as the Software Engineering for topics of the web, but it wasn't adopted very well. While web development is a slightly bigger topic than software development when comparing Google Trends, Web Engineering is a lot less used when compared to Software Engineering.
[9:06 PM] Dan Bechard: @Julian Fastnacht Yeah I know, but the reason it's not used it because it sounds like it described people who engineer the web, not people who engineer web sites using web technologies
[9:06 PM] Dan Bechard: there is very little engineering involved in modern web development unless you're working on a rather large software system
[9:06 PM] Dan Bechard: it's much more development than engineering
[9:06 PM] Dan Bechard: hence the popularity of "Web Development"
[9:07 PM] Dan Bechard: and in the case of "large software systems" again.. "Software Engineering" is more accurate
[...]
[9:09 PM] Dan Bechard: From Wikipedia:
Engineering is the application of mathematics and scientific, economic, social, and practical knowledge in order to invent, innovate, design, build, maintain, research, and improve

I think it's pretty easy to argue that very few people are "inventing, innovating, designing, or building" the web these days. The actual web itself, not the software and content that exists on top of it.(edited)
[9:09 PM] Dan Bechard: However, there are definitely people "maintaining, researching, and improving" the web, so those people could be considered "Web Engineers" for sure.
[9:10 PM] Dan Bechard: Hence the use of the word "engineer" in IETF, "Internet Engineering Task Force", https://www.ietf.org/
[9:12 PM] Julian Fastnacht: The only regular source for articles about Web Engineering is the International Conference on Web Engineering (ICWE), but that seemed to be like an isolated group without many other publications.

Interview - Fragen

Länge: 30min (+30min Puffer)
Personen: 10 - 12

Informationen

- kurze Vorstellung
- Zielsetzung: Ermittlung des momentanen Standes zum Anforderungsmanagement bei der Entwicklung von TYPO3-Extensions
    - insbesondere Abbildung der einzelnen Tätigkeiten
    - Wichtig: IST-Zustand, nicht SOLL-Zustand
- das Interview dient der Verbesserung des Prozesses und nicht der Überprüfung der Mitarbeiter

Einstiegsfragen

- 1. Seit wann in der Web-/Software-Entwicklung beschäftigt?
- 2. In welchem Gewerk tätig?
- 3. Erfahrungen mit:
    - klassischen Methoden in der Web-Entwicklung (Wasserfall, V-Modell)
    - agilen Methoden in der Web Entwicklung (Scrum, Kanban)
- 4. [PM, TPL] Einsatz von Industrienormen (DIN, ISO, VDE, VDI)
    - Erfahrungen
    - Haltungen
- 5. [PM, TPL] Gibt es eine Dokumentation der Verteilung der IST-Kosten für Projekte?
- 6. Berührungspunkte mit Anforderungsmanagement? (Vorerfahrung)
    - Einordnen der eigenen Tätigkeit in die Tätigkeiten des Anforderungsmanagement - wenn möglich
- 7. [PM, TPL] Aktuelle Verantwortlichkeit beim Anforderungsmanagement? (evtl. weiter untergliedern)
- 8. Gab es bereits Versuche zur Einführung von Methoden des AM?

Ermitteln

- 9. Vorbereitung für Kundentermine? (Erstellung von Fragebögen? Übliche Fragen? Cheatsheet? Frei nach Gefühl/Erfahrung?)
- 10. Übliche Formate für die Ermittlung von Anforderungen? (Meeting, Screensharing, Telefonat?)
- 11. Wird der Kunde/Werden die Projektbeteiligten an die Arbeit mit Anforderungen herangeführt?

- 12. Werden Entwickler an der Kommunikation mit dem Kunden beteiligt? (z.B. Abstimmung von techn. Machbarkeit, Aufwandsschätzung) / Welche Personen sind in welchem Kontext beteiligt?
- 13. Welche Informationen kommen vom Kunden? (direkt / auf Nachfrage, Vollständigkeit)

Dokumentieren

- 14. Welche Dokumente werden innerhalb des Projektes erstellt?
- 15. Wie findet die Ablage von Dokumenten statt?
    - Art der Dokumente
    - Ablageort
    - Zugriffsmöglichkeiten
- 16. (Protokollieren/Ablegen von Kundengesprächen?)
- 17. Dokumentieren von Fehlern/Problemen?
- 18. Welche Dokumente/Informationen bekommen die Entwickler?
- 19. Enthalten Tickets im Redmine alle nötigen Informationen zur Lösung der Aufgabe? (Wenn nein, wie oft kommt es vor, dass Informationen fehlen/angefragt werden? Sind es vorwiegend Probleme die intern gelöst werden oder mit dem Kunden abgestimmt werden müssen?)
- 20. Sollten Entwickler die Dokumente des Projektes lesen?

Abstimmen, Prüfen & Validieren

- 21. [PM, TPL] Findet eine nachträgliche Validierung der unpräzisen Anforderungen unter Rücksprache mit dem Auftraggeber statt?
- 22. Findet eine Verifikation der Anforderungen statt? Wer ist für die Verifikation (Testen) der Anforderungen zuständig?
- 23. Sind Anforderungen jederzeit nachverfolgbar? (Traceability)

Verwalten

- 24. Wer ist zuständig für die Aktualität im Projekt erstellter Dokumente?
- 25. Wann/Wie oft werden diese Dokumente aktualisiert?
- 26. Gibt es eine Änderungshistorie für Anforderungen?
- 27. Wie laufen Änderungsanfragen des Kunden für gewöhnlich ab? (Änderungsanfrage, Prüfung der Machbarkeit, Annahme?, Durchführung der Änderung, Änderung der Dokumente)
- 28. [PM, TPL] Werden vor Änderungen von Anforderungen Einflussanalysen durchgeführt?

Abschluss

- 29. Lassen sich wiederkehrende Probleme im Entwicklungsprozess ableiten und wie sind diese zu Priorisieren? (evtl. nachreichen)
- 30. [PM, TPL] Kein Projektende durch fehlende Baselines?
- 31. [PM, TPL] Sind Nachverhandlung durch fehlende Anforderungen üblich/selten/kein Thema?
- 32. Was läuft gut / nicht gut (muss verbessert werden) im Bereich AM?

Interview 1

Einstiegsfragen

2. In welchem Gewerk tätig?

- PM / Account Mgmt./Kundenbetreuung
    - Anfragen evaluieren (Anf.mgmt.)
    - die Extension wird bezahlt, nicht die Innovation
    - schwierig Innovationen abzurechnen

3. Erfahrungen mit klassischen/agilen Methoden?

- Verlauf ist hier weniger strukturiert
- bereits Erfahrungen mit agilem Produktmanagement gesammelt
- Scrum in anderem Unternehmen
    - gute Erfahrungen
    - es wurde in kurzer Zeit viel umgesetzt
- PM gute Theorie, aber Praxis schwierig
    - zu viele Produkte, keine festen Teams und Arbeitsumgebungen, viele Zwischenrufe
    - "Aufholen ist nicht Scrum"
    - fokussiert über längeren Zeitraum arbeiten
    - benötigt den Verkauf von Sprints, nicht fertiges Produkt
- Scrum gerne am Beispielprojekt testen
    - im Kontext durch Spezialisten schwierig
- eher "normales" PM
    - "jonglieren" mit Aufgaben
    - viel Kommunikation notwendig
    - Entwickler werden verplant, obwohl sie woanders eingesetzt werden müssten

4. Einsatz von Industrienormen?

- ist sehr aufwendig
- es wären Schulungen notwendig
    - Was ist erlaubt, was nicht?
- Probleme, wenn nicht mit der Norm übereinstimmt
- generell "gilt das geschriebene Wort"
- Feinkonzept als Begriff und vom Inhalt her ist okay
- wichtig ist ein gutes Gefühl beim Kunden
- Normen wären eher wichtig für große Kunden
    - im Moment ist das jedoch nicht zu leisten

5. Gibt es eine Dokumentation zur Verteilung der IST-Kosten für Projekte?

- idealerweise sollte das gemacht werden
- man versucht in der angesetzten Projektzeit zu bleiben
    - beruht auf Zeitschätzungen und Prognosen, alles weitere sollten Zusatzaufträge sein und neu evaluiert werden
    - ansonsten verursacht es einen Kostenberg
- Zeiten werden falsch eingeschätzt, Entwickler werden nicht fertig/gestört/überfordert
    - Entwickler von anderen Projekten werden dazu geholt, um "Brände" zu löschen - jedoch wird nicht der "Brandherd" gelöscht
    - verursacht wiederum Probleme in anderen Projekten
- Konzept und Entwicklung verursachen etwa gleich viel Aufwand
    - aber bei Angebotserstellung stellt Entwicklung meist 80% und Konzept nur 20% der Kosten dar, Erfahrung zeigt: eher unrealistisch
- viel/mehr Zeit in Konzeption wäre besser

    - muss einzeln verrechnet werden
    - Anforderungskatalog als ein Ergebnis/Milestone/wichtigen Zwischenschritt und Bildung einer Grundlage für die techn. Entwicklung

6. Berührungspunkte zum Anforderungsmanagement?

-

7. Aktuelle Verantwortlichkeit beim Anforderungsmanagement?

- momentan PM (IST)
- Wunsch: Konzepter
    - bei kleinen Projekten würde dabei auch ein kurzer Austausch reichen, um Konzepter nicht zusätzlich zu belasten
    - beim PM sollte der Aufwand des Anforderungsmanagements extra betitelt werden

8. Gab es bereits Versuche zur Einführung von Methoden des Anforderungsmanagements?

- es wurden schon Projekte durchgeführt, in denen erst eine Anforderungsanalsyse/Feinkonzept verkauft wurde
    - ohne Beteiligung eines Konzepters
- es gab auch schon Kunden, die eigenen Konzepter hatten
- Wann ist On-the-Fly Umsetzung zu viel?
    - Ab wann muss man zurück zum Konzept?

Ermitteln

9. Vorbereitung für Kundentermine?

- es beginnt mit einem ersten Gespräch
- oftmals wird keine Konzeption für One-Pager eingeplant (auch hier bildet es die Grundlage für alles weitere im Projektverlauf)
- Komplettangebot oder Konzeption als Angebot (Teilung)
- viel Austausch zwischen Konzeption und Design verursacht Extra-Kosten
- es gibt auch die Möglichkeit, dass nach erfolgter Konzeption keine technische Umsetzung erfolgt
    - Gründe können sein: zu großes Risiko
- verschiedene Dokumente dienen der Vorbereitung
- Basic Fragen aus Erfahrungswerten
    - Ansprechpartner-Definition
    - Hosting
    - ...
- Dokumente für die Anfangsphase (Fragenkatalog)
    - Fragen nur zu 50-60% auf Kunden passend
    - müssen trivial sein
    - vieles ergibt sich aus dem Gespräch
- Immer hinterfragen, ob Wunsch oder konkrete Anforderung?
- theoretisch sollte schon beim Kick-Off Meeting ein TPL (je nach Größe auch mehr Instanzen!) eingebunden sein
    - dieser kann bessere Bewertung (Machbarkeit und Kostenschätzung) abgeben
- die Anforderungen müssen auch an das Budget angepasst sein
    - Sind alle Anforderungen mit dem gegebenen Budget umsetzbar?
- Kick-Off wird teilweise nicht durchgeführt und nicht eingepreist

10. Übliche Formate für die Ermittlung von Anforderungen?

- Kick-Off Meeting mit Kunden
    - manchmal auch ohne Kunde, wenn nur auf das Projekt gebrieft werden soll
    - dem Kunden die Projektbeteiligten vorstellen und den Ablauf erklären
- Kick-Off ist Teil des Konzepts

11. Wird der Kunde/Werden die Projektbeteiligten an die Arbeit mit Anforderungen herangeführt?

siehe 10.

12. Werden Entwickler an der Kommunikation mit dem Kunden beteiligt?

- ja
- im Kick-Off Meeting
- sonst eher nicht
- Ausnahme bilden wichtige Termine

13. Welche Informationen kommen vom Kunden?

- meistens viel Nachfrage
- kommt auf den Kunden an
    - Website ist toll wie sie ist, es soll jedoch etwas neu gemacht werden
    - Website wird im aktuellen Zustand als unzureichend empfunden, soll komplett neu gemacht werden

Dokumentieren

14. Welche Dokumente werden innerhalb des Projektes erstellt?

- Confluence ist die Grundlage der Dokumentation
- Design-Briefing
    - vereinfacht die Übergabe von Design an die Technik
    - wird weiter ausgebaut
- verschiedene Dokumente, von Mockups bis Konzept

15. Wie findet die Ablage von Dokumenten statt?

- das Ziel ist, alles auf Confluence zu haben
    - stellt eine Umgewöhnung dar
    - einheitlich gesammelt, alle Informationen verfügbar
- es sollte keine Abweichung davon geben

16. Protokollieren/Ablegen von Kundengesprächen?

- Protokoll
    - Kick-Off Gesprächsprotokoll (benötigt jedoch immer einen Protokollanten)
- die Konzeption ist das "Protokoll" was vom Kunden abgenommen wird

17. Dokumentieren von Fehlern/Problemen?

- stellt Erfahrungswerte dar, die effektiv an unterschiedliche Projektbeteiligte, auch über das Projektteam hinaus (direkte Kollegen) weiter gegeben und für die Zukunft genutzt werden können
- man muss von allen Beteiligten Transparent fordern
- es gibt derzeit keine Grundlage für proaktives Wissensmanagement
    - Idee: die Umsetzung eines allgemeinen Ablageortes gestalten, anbieten würde sich Confluence
- neuen PMs sind Vorgehensweisen ist nicht klar, was gemacht werden muss, welche Vorgehensweisen/Workflows in der Firma herrschen

18. Welche Dokumente/Informationen bekommen die Entwickler?

- es gibt ein wöchentliches Projekt Kick-Off mit allen nötigen Informationen
- je nach Projekt liegen alle Dokumente zentral gesammelt, meist im Confluence (sollte sich als "Standard" implementieren)
    - handelt sich eher um ein internes Kommunikationsproblem, weil Entwickler teilweise die Dokumente nicht lesen, es strukturiell nicht richtig aufbereitet ist oder die Zeit für die Einarbeitung zu knapp bemessen ist.

19. Enthalten Tickets im Redmine alle nötigen Informationen zur Lösung der Aufgabe?

- Ein Ticket enthält keine großen Briefingdokumente, jedoch den Verweis auf Dokumente innerhalb des Projektbereiches
- Tickets enthalten meistens nur den Verweis auf die Umsetzung einer Extension, technische Details sind unzureichend angegeben

    - Entwickler muss sich dennoch meist die Details für ein vernünftiges Set Up / reinkommen in das Problem selbst suchen

20. Sollten Entwickler die Dokumente des Projektes lesen?

- ja
- Entwickler sollten Dokumente lesen, tun es teilweise nicht
    - führt zu Kommunikationsproblemen
    - es könenn nicht alle Microdetails in ein Ticket verpackt werden
- es ist nicht die Aufgabe des PM jeden Entwickler individuell zu behandeln
    - Entwickler müssen mitdenken und bei Bedarf eigenständig agieren, um unnötiges "Ticketschupsen" zu vermeiden

Abstimmen, Prüfen & Validieren

21. Findet eine nachträgliche Validierung der unpräzisen Anforderungen unter Rücksprache mit dem Auftraggeber statt?

- beides, abhängig von Frage, Budget und Aufwand **(--> das verstehe ich nicht)**
- man muss die Anforderung anschauen und dann entscheidet der PM die weitere Vorgehensweise
- Validierung wird nicht bezahlt
    - zusätzliche Anforderungen werden wahrscheinlich nicht nachkalkuliert
    - lohnt sich vor allem bei langfristigen Kunden, bei denen mehrere Aufträge vorhersagbar sind, auch im Interesse der Erarbeitung eines Formates für eine ideale künftigen Zusammenarbeit

22. Findet eine Verifikation der Anforderungen statt? Wer ist für die Verifikation der Anforderungen zuständig?

- QM findet im Projekt statt, hoher Organisationsbedarf für den PM
- wird jedoch stiefmütterlich behandelt
- Projekte ohne kalkulierte QS sind schwierig zu validieren (Der Aufwand von Nacharbeiten steigt zudem immens in die Höhe, bis hin zu: wirtschaftlich untragbar)
- meistens findet dann eine Abstimmung mit dem Kunden statt, dass es durch ihn übernommen wird

23. Sind Anforderungen jederzeit nachverfolgbar? (Traceability)

- idealerweise ja
- unterliegt der Konzeption
    - auch abhängig von der Reaktion/Stellungswert der Kunden

Verwalten

24. Wer ist zuständig für die Aktualität im Projekt erstellter Dokumente?

- PM, aber
    - wenn TPL vorhanden, dann ist dieser für die Aktualität technischer Aufgaben zuständig

25. Wann/Wie oft werden Dokumente aktualisiert?

- je nach Projektphase
- die Konzeption und das Angebot werden idealerweise nicht mehr geändert
    - Ausnahme: Kundenwunsch und -abstimmung (liegt einem Zusatzangebot zu Grunde)

26. Gibt es eine Änderungshistorie für Anforderungen?

- nur auf Dokumentenebene
    - Confluence
    - Email
    - sonstige Ablagesysteme

27. Wie laufen Änderungsanfragen des Kunden gewöhnlich ab?

- bei Eingang der Anfrage zunächst mit dem TPL austauschen
- Welche Folgen/Auswirkungen hat es?
    - abhängig davon ob Flexibilität für Änderungen in der UMsetzugn bedacht wurden
- Mehraufwand hat auch Einfluss auf die Gefühlslage
    - Änderungwünsche nach viel Arbeit durch den Entwickler führen zu Frustration (auf Kunden- und Agenturebene)

28. Werden vor Änderungen von Anforderungen Einflussanalysen durchgeführt?

siehe 27.

Abschluss

29. Lassen sich wiederkehrende Probleme im Entwicklungsprozess ableiten und wie sind diese zu Priorisieren?

- das Learning wird nicht weitergegeben (Review Meetings)
- QS ist nicht im Projekt eingeplant
- bei der Kalkulation des Angebots werden PM und Entwickler kaum eingebunden
    - Entwickler für die Kalkulation meist ungleich der Umsetzung, führt zu planerischen Differenzen
- der Posten QS/Konzepter ist nicht eingeplant, wird nicht kalkuliert, jedoch zwingend für jedes Projekt benötigt. Es bildet die Grundlage.

30. Kein Projektende durch fehlende Baselines?

- es gibt im Konzept nur Angaben dazu, was umgesetzt wird, keine Abgrenzung dazu, was nicht umgesetzt wird
    - dadurch kein Ende, selbstverschuldet.

31. Sind Nachverhandlungen durch fehlende Anforderungen üblich/selten/kein Thema?

-

32. Was läuft gut / nicht gut (muss verbessert werden) im Bereich AM?

- bei Einordnung der Größe eines Projektes hinterfragen ob es zur momentanen Auslastung passt
    - wird ohne Beachtung der Auslastung angenommen "nur ein kleines Projekt"
    - Kunden für einen späteren Termin zusagen
- bei der Kalkulation sollte mehr als nur Sales einbezogen werden
    - nicht nach "Preisliste" kalkulieren
- Transparenz schaffen
    - muss proaktiv gelebt werden
    - Austausch untereinander fördern
- verhindern, dass einzelne Mitarbeiter sich hinter ihren Aufgaben/ihrem Aufgabenbereich verstecken
- regelmäßige Reviews, nicht nur um die Zeit, sondern auch den Fortschritt und Aufgabenbereich der Entwickler zu tracken. PM sollte vollumfängliche Kontrolle über sein Projekt haben.

Interview 2

Einstiegsfragen

2. In welchem Gewerk tätig?

- aktuell PM
- vorher bereits in versch. Bereichen:
    - Musik, Grafik, Konzeption, Producing (Browsergames)

3. Erfahrungen mit klassischen/agilen Methoden?

- agile Methoden eignen sich auf Grund von schlechter Planbarkeit besser
- dabei wir jedoch kein reines Scrum eingesetzt

    - es handelt sich um einen relativ agilen Workflow
- Bsp.: Kunde ändert Anforderungen während der Umsetzung, die nicht Teil vom Angebot sind
    - schwierig bei starrer Projektierung
- darum lieber kleine Schritte
    - keine großen Blöcke
- die Angebotslage muss immer klar sein

4. Einsatz von Industrienormen?

- Nein!
- gefährlich die Begrifflichkeiten zu verwenden, weil sie Verpflichtungen mit sich bringen
- inhaltliche Nutzbarkeit schwer zu beurteilen, ohne einen konkreten Überblick zu haben
- nicht für Projekte jeder Größe geeignet
- evtl. für Großprojekte
    - treibt jedoch die Kosten in die Höhe

5. Gibt es eine Dokumentation zur Verteilung der IST-Kosten für Projekte?

- Stunden werden erfasst, jedoch hauptsächlich bei der technischen Umsetzung

6. Berührungspunkte zum Anforderungsmanagement?

- Anforderungsmanagement ist Definitionssache
- wird eher aus der Erfahrung heraus betrieben
    - keine feste Schule
- es ist ein Fragenkatalog als Querschnitt vorhanden

7. Aktuelle Verantwortlichkeit beim Anforderungsmanagement?

- bei PM
- Einschränkung bei einigen Kunden
    - externe Konzeption/Verantwortung für Anforderungsmanagement

8. Gab es bereits Versuche zur Einführung von Methoden des Anforderungsmanagements?

- der Fragenkatalog
- Grundstruktur der Projektdurchführung im Redmine

Ermitteln

9. Vorbereitung für Kundentermine?

- mit dem Fragebogen und Material vom Kunden
    - Abgleich von Material und Fragebogen
- teilweise wurden auch Fragebögen an den Kunden geschickt
    - unterschiedliche Erfahrungen (positiv/negativ)
- erst einfache Fragen stellen, statt in einem großen Block
    - nicht gleich "Highlevel"

10. Übliche Formate für die Ermittlung von Anforderungen?

- alle mgl. Formate
- Ablaufrahmen
    - Kick-Offs
    - Schulterblicke
- bei kleinen Projekten lockerer als bei großen Projekten
- räumliche Aspekte und Fahrtkosten spielen eine Rolle, ob vor Ort oder nicht
- bei wenigen Infos anders als bei umfangreichen Ausschrebungen mit vielen Informationen

11. Wird der Kunde/Werden die Projektbeteiligten an die Arbeit mit Anforderungen herangeführt?

- ja

- Kunde ist der Herr der Fachdomäne, während CPS Experten für CMS ind
- Kunden klar machen, dass Domänenwissen vom ihm erbracht werden muss
    - anschließend Übersetzung in das System

12. Werden Entwickler an der Kommunikation mit dem Kunden beteiligt?

- je unkonkreter das Projekt ist, desto kleiner die Gruppe für den Einstieg
    - evtl. ist schon ein Design von extern vorhanden
- wenn komplexe Prozesse oder Technik nötig, dann wird TPL einbezogen
- TPL und Teamleiter Entwicklung qualifizieren die geeigneten Entwickler für die jeweiligen Aufgaben
- TPL übernimmt die techn. Orchestrierung
    - hat den Gesamtüberblick
- TPL muss früh dabei sein

13. Welche Informationen kommen vom Kunden?

- kommt darauf an
    - unterschiedlich

Dokumentieren

14. Welche Dokumente werden innerhalb des Projektes erstellt?

- Dokumentation bleibt meist in kleinen Projekten auf der Strecke
    - v.a. Betriebshandbuch
- Technische Konzept
    - "hoffentlich" vor Angebot zur Umsetzung
    - nicht immer der Fall
- Prototypen (kein Dokument)
- es gab bereits Kunden bei denen externe Dienstleister beteiligt war und Dokumentation auf Seiten von CPS durchgeführt werden sollte
- Protokolle
    - Gesprächsprotokoll
    - formlose Zusammenfassung
- Styletiles, Mockups und Sitemap in Excel
    - daraus werden HTML-Prototypen und Ansichten erstellt
- Ticketsystem (für Dokumentation, nicht als Dokument)
    - Screenshots und Kommentare
    - wird statt Mail und Telefon eingesetzt

15. Wie findet die Ablage von Dokumenten statt?

- werden in Ordner des Projekts abgelegt (Fileserver)
- Redmine
- Confluence
- OwnCloud
- Git

- Grafik und Entwicklung haben unterschiedliche Ablageorte/-weisen

16. Protokollieren/Ablegen von Kundengesprächen?

siehe 14.

17. Dokumentieren von Fehlern/Problemen?

- nicht viel
- "Golden Master" Repositories im Git
    - eher technisch
    - Wissen einfließen lassen
    - Schnellstart für TYPO3 Projekte

18. Welche Dokumente/Informationen bekommen die Entwickler?

- Entwickler haben Zugriff auf die Dokumente

19. Enthalten Tickets im Redmine alle nötigen Informationen zur Lösung der Aufgabe?

- Tickets sollten alle Informationen zur Lösung der Aufgabe erhalten

20. Sollten Entwickler die Dokumente des Projektes lesen?

- Entwickler sollten das technische Konzept lesen
    - bei Unklarheiten muss TPL eingebunden werden

Abstimmen, Prüfen & Validieren

21. Findet eine nachträgliche Validierung der unpräzisen Anforderungen unter Rücksprache mit dem Auftraggeber statt?

- früher einfach tun, mittlerweile wird Kunde noch einmal gefragt

22. Findet eine Verifikation der Anforderungen statt? Wer ist für die Verifikation der Anforderungen zuständig?

- kommt drauf an
- mal durch Entwickler, mal durch internen Prozess
- optimalerweise sind die notwendigen Tests im Ticket abgelegt
- kein internes QM

23. Sind Anforderungen jederzeit nachverfolgbar? (Traceability)

- kommt auf das Werkzeug an
- Confluence hat hohe Traceability
- wenn jedoch Absprachen über Telefon geschehen, aus denen nur ein Ticket, aber keine Notizen entstehen, dann schwierig
- Office Dokumente im Änderungsmodus

Verwalten

24. Wer ist zuständig für die Aktualität im Projekt erstellter Dokumente?

- kommt darauf an
- i.d.R. PM und TPL

25. Wann/Wie oft werden Dokumente aktualisiert?

- im Confluence kontinuierlich während des Projekts
    - erhält Historie
- Office Dokumente haben dagegen oft Aktualitätsproblem
    - es werden Office Dokumente mit falschen Ständen genutzt

26. Gibt es eine Änderungshistorie für Anforderungen?

- keine Änderungshistorie einzelner Anforderungen
- Confluence und Redmine führen Änderungen
- eher Versionierung auf Dokumentenebene

27. Wie laufen Änderungsanfragen des Kunden gewöhnlich ab?

- Lässt sich Anfrage als Change Request (CR) qualifizieren?
    - nach Aufwand werden Änderungswünsche sofort geändert
- besser wenn dies bereits auf Dokumentenebene und nicht erst während der Umsetzung passiert
- wenn CR aktuell nicht passt, wird dieser in andere Version verschoben
    - Redmine Feature Liste

    - Ausgrenzen der zusätzlichen Änderungen
- individueller Umgang mit Dokumenten
- Abnahmekriterien basieren auf Dokument

28. Werden vor Änderungen von Anforderungen Einflussanalysen durchgeführt?

- wird durchgeführt
- Zusammensetzen mit entspr. Entwicklern und schauen, welche Einflüsse die Änderung hat
- auch ein Prototyp kann der Dokumentation dienen, wenn er vom Kunden abgenommen wird

Abschluss

29. Lassen sich wiederkehrende Probleme im Entwicklungsprozess ableiten und wie sind diese zu Priorisieren?

- Kunde hält die Fachdomäne und muss mitmachen
    - wird nicht immer ernst genommen

30. Kein Projektende durch fehlende Baselines?

- ganzheitlich selten
- je größer das Projekt, desto akribischer das Vorgehen

31. Sind Nachverhandlungen durch fehlende Anforderungen üblich/selten/kein Thema?

- kommt darauf an

32. Was läuft gut / nicht gut (muss verbessert werden) im Bereich AM?

- einige Kunden haben techn. Ansprechpartner im Haus
    - Probleme entstehen bei Entscheidungsträgern ohne technischen Hintergrund

Interview 3

Einstiegsfragen

2. In welchem Gewerk tätig?

- Projektmanager / Account-Manager

3. Erfahrungen mit klassischen/agilen Methoden?

- Arbeitsumfeld waren hauptsächlich Agenturen
- es stellt ein Problem dar mit Kunden agil zu arbeiten
    - Ansprechpartner ist das Marketing
    - der Produktzyklus läuft bei den Kunden nach Wasserfall
- agile Vogehensweisen sind bekannt, es werden jedoch nur agile Elemente genutzt
    - bspw. Selbstorganisation
- dabei muss immer die Kosteneffizienz im Auge behalten werden
    - vor allem den Zeitfaktor beachten

4. Einsatz von Industrienormen?

- Verwendung das Vokabular unbedarft
    - intern kein Problem, auch bei manchen Kunden nicht
- Problem entsteht, wenn der Kunde sich auf diesen Begriff stützt
    - es gibt klare Vorgaben, was bspw. in Lasten-/Pflichtenheft enthalten ist, auf die sich der Kunde berufen könnte
- daher eher Verwendung "schwammiger" Begriffe wie Anforderungsanalyse und Feinkonzept

5. Gibt es eine Dokumentation zur Verteilung der IST-Kosten für Projekte?

- Projektcontrolling wird durchgeführt
    - Zeiten werden berichtet
- die Einteilung erfolgt eher nach Aufgaben, nicht nach Phasen (Konzeption, Design, Entwicklung, ...)
    - statt z.B. einer Phase für das Design wird Design als Aufgabe verstanden
- die Durchführung der Aufgaben wird beurteilt
    - Einschätzung der Zeit/Kosten im geplanten Rahmen?
    - Gibt es Ausgleich zwischen verschiedenen Aufgaben oder Ausschläge?
    - Wo ist ggf. das Problem ggü. der Kalkulation?
- am Ende muss die Rentabilität stimmen
- Problem sind auch fehlende Faktoren (Budget) im Projektmanagement Tool (Planio?)
    - keine Darstellung der "Cost-Rates" (Kostensatz? Arbeitskosten/h?)
- es wird kein Zusammenhang zwischen Umfang der Konzeptionsphase und Umfang der Entwicklungsphase gebildet
    - Fragestellung war: Bedeutet eine längere Konzeptionsphase eine Verringerung der Entwicklungsphase und wie verhält es sich mit den Kosten?

6. Berührungspunkte zum Anforderungsmanagement?

- bereits zu Beginn geklärt

7. Aktuelle Verantwortlichkeit beim Anforderungsmanagement?

- Hauptverantwortlichkeit liegt im Bereich Kundenberatung
- dabei werden auch immer Technische Projektleiter und Entwickler zu den Anforderungen befragt
    - Machbar, Plausibel, ...

8. Gab es bereits Versuche zur Einführung von Methoden des Anforderungsmanagements?

- das zu Anfang bereits erwähnte Paper
- grobe Abfrage von Anforderungen, die Details kommen dann im Laufe des Prozesses

Ermitteln

9. Vorbereitung für Kundentermine?

- in der Regel gibt es ein Briefing durch den Kunden
- entweder ist eine Seite/Extension bereits vorhanden, die man sich anschauen kann
    - Vergleich der vorhandenen Seite/Extension mit ?
    - Ermittlung von Aufgabe, Geschäft und/oder Außendarstellung
- oder es muss weiter nachgefragt werden
    - Anforderungen sind im Briefing eigentlich nie vollständig
- meistens werden die Briefings durch die Marketing-Abteilung/Personal aus dem Marketing gegeben, nicht durch technisch versiertes Personal

10. Übliche Formate für die Ermittlung von Anforderungen?

- Briefing

11. Wird der Kunde/Werden die Projektbeteiligten an die Arbeit mit Anforderungen herangeführt?

- Einführung in Arbeitsprinzipien findet statt
- meistens ist die einzige Information des Kunden "Will eine Website bis zum XX.XX.XXXX ."
    - der Kunde will Layouts für alles, oftmals in gedruckter Form
    - das eigene Vorgehen ist dabei jedoch Styletiles und Einzelelemente zu verwenden
- die Einführung findet nicht im Erstgespräch statt
    - sondern eher in den Folgeterminen, mit Details
- bei Projektstart gibt es dann eine Einführung in die Arbeitsweise, jedoch hat der Kunde ein hohes Mitspracherecht

12. Werden Entwickler an der Kommunikation mit dem Kunden beteiligt?

- alles geschieht unter Rückfrage beim Entwickler

    - dabei werden speziell die Entwickler gewählt, die für die Lösung besonders gut geeignet sind
- Ansonsten werden auch Konzepter, Designer und andere Entwickler mit eingebunden

13. Welche Informationen kommen vom Kunden?

- das Spektrum der Informationen ist groß
- nicht alle Informationen werden beim Erstkontakt vermittelt
- meistens sind mehrere Nachfragen notwendig

Dokumentieren

14. Welche Dokumente werden innerhalb des Projektes erstellt?

- Briefing Dokument in MS Word
- dann Konzept-Paper und daraus die Technische Konzeption

15. Wie findet die Ablage von Dokumenten statt?

- Dokumente werden in Confluence abgelegt
    - auch Entwickler greifen darauf zu

18. Welche Dokumente/Informationen bekommen die Entwickler?

- alle Beteiligten sind zu Beginn in einem Kick-Off Meeting
    - dort sollten alle mit dem Projekt vertraut werden
- Tickets haben einen Verweis auf das Confluence, in dem Entwickler nachlesen können
- wenn trotz Ticket und Confluence die Aufgabe nicht gelöst werden kann, aber die Anforderungen des Kunden in Ordnung waren, handelt es sich vermutlich um interne Kommunikationsprobleme
    - Strukturprobleme

16. Protokollieren/Ablegen von Kundengesprächen?

- vor allem Gesprächsprotokolle oder Notizen
    - diese werden auf dem Fileserver abgelegt, aber nicht im Confluence
    - Aufgaben werden auf Basis dieser Dokumente erstellt und verweisen auf diese

(Wunsch: Wiki für die Softwaredokumentation)

17. Dokumentieren von Fehlern/Problemen?

- wichtig ist Gesamtbasis vs. In-Time
- am Ende von Projekten werden dafür Projektreviews durchgeführt
    - alle Projekte über 10.000,00€ (Vertragssumme oder interne Kosten?)
- aus Problemen lernen
    - wird nur im Team geteilt, aber nicht zentral abgelegt
- teilweise gibt es Diskussion über Neuentwicklungen, die schon einmal umgesetzt wurden oder verschiedene Lösungen für gleiche Probleme
    - resultiert aus fehlendem Knowledge Management
    - Sebastian oder Kjeld wären dafür die Ansprechpartner

19. Enthalten Tickets im Redmine alle nötigen Informationen zur Lösung der Aufgabe?

- sollte sich aus Ticket und Verweis ergeben (*s. 18.*)

20. Sollten Entwickler die Dokumente des Projektes lesen?

- sollte sich aus Ticket und Verweis ergeben (*s. 18.*)

Abstimmen, Prüfen & Validieren

21. Findet eine nachträgliche Validierung der unpräzisen Anforderungen unter Rücksprache mit dem Auftraggeber statt?

Fragenteil scheinbar unter gegangen, bei Beantwortung zusammen mit 22.

22. Findet eine Verifikation der Anforderungen statt? Wer ist für die Verifikation der Anforderungen zuständig?

- die Verifikation erfolgt durch die Qualitätssicherung (Andreas)
    - sowohl Funktionalität, als auch Optik
    - dieser gleicht auch das Produkt mit den definierten Anforderungen ab
- bei Fehlern wird die Aufgabe wieder zum Entwickler zurück geleitet, nicht zum PM
    - wenn die Anforderungen nicht erfüllt wurden
- Testszenarien werden teilweise auch vorher durch einen Entwickler oder bereits zusammen mit den Anforderungen definiert
- erst nachdem dies abgeschlossen ist, erfolgt der Test durch den Kunden

23. Sind Anforderungen jederzeit nachverfolgbar? (Traceability)

- es erfolgt keine Versionierung in Confluence (im Sinne einer Versionstabelle)
    - in Tickethistorie sind Änderungen nachvollziehbar
- im Ticket erfolgt auch der Verweis auf Dokumente
    - Ausnahme sind dabei kleine Tickets/Aufgaben

Verwalten

24. Wer ist zuständig für die Aktualität im Projekt erstellter Dokumente?

- jede Abteilung ist für die eigenen Dokumente verantwortlich
    - TPL für den technischen Teil der Dokumentation und bereitstellen des Wiki
    - Konzepter erarbeitet das Konzept
- die Prüfung erfolgt nicht durch den PM
    - optimalerweise prüft jede Abteilung selbst
    - es könnte auch der TPL machen

26. Gibt es eine Änderungshistorie für Anforderungen?

- Revisionswechsel durch Änderungen in Konzeption
    - eine Änderungshistorie gibt es dabei im Code
- Change Request (CR) an Feinkonzept
    - mit Beschreibung der Änderungen

27. Wie laufen Änderungsanfragen des Kunden gewöhnlich ab?

- im schlimmsten Fall werden diese nicht als Change Request erkannt
- wenn es neue Anforderungen sind, dann werden diese so kommuniziert und kalkuliert
    - Letzteres ist abhängig vom Umfang (es wird eine Abgrenzung vorgenommen)
    - oftmals treten diese während des Umsetzungsprozesses auf und sollen sofort umgesetzt werden
- potenzielle Change Requests werden ggf. auch vom Entwickler signalisiert, weil ihnen zugewiesene Aufgabe eigentlich Change Request ist
    - technische Änderungen werden vom Entwickler besser realisiert

28. Werden vor Änderungen von Anforderungen Einflussanalysen durchgeführt?

- Technischer Projektleiter führt die (Einfluss-?)Analyse durch
    - Zeiteinschätzung erfolgt durch den Entwickler
    - dabei werden die Möglichkeiten und deren Zeiteinschätzung angegeben und dann durch PM/TPL? entschieden
    - die Anfrage erfolgt bei Entwicklern, die im Projekt aktiv sind (sich auskennen)

Abschluss

29. Lassen sich wiederkehrende Probleme im Entwicklungsprozess ableiten und wie sind diese zu Priorisieren?

- Flaschenhals ist die Beratung
    - vor allem unklare Briefings
- fehlende Technische Projektleiter
    - PM ohne technischen Hintergrund sollte keine Entwickler briefen

30. Kein Projektende durch fehlende Baselines?

- Funktionsbeschreibung und Bedienungsbeschreibung als Grundlage
    - alles was dort nicht auftaucht ist Change Request
- es müssen immer Abstimmungen mit dem Kunden erfolgen
- es sollte mit Baselines gearbeitet werden

31. Sind Nachverhandlungen durch fehlende Anforderungen üblich/selten/kein Thema?

Fehlt?

32. Was läuft gut / nicht gut (muss verbessert werden) im Bereich AM?

- Briefings sollten verbessert werden
    - es gibt kein genormtes Briefing
    - teilweise sind es nur PDF Dateien mit ein paar Anmerkungen vom Kunden
- Wie sehen die Dokumente aus? (Dokumentvorlagen)

Interview 4

Einstiegsfragen

2. In welchem Gewerk tätig?

- 6 Jahre Entwicklung
- seit ~2 Jahren PL

3. Erfahrungen mit klassischen/agilen Methoden?

- "Mischmasch"
- Arbeit in Agenturen und Start-Ups
    - dort von klassisch bis agil alles genutzt
- bisher kein Projekt was wirklich einem klassischen Modell gefolgt ist

4. Einsatz von Industrienormen?

- in einer Agentur (FR) dieser Größe sollten bestimmte Prozesse umgesetzt werden
    - nicht zwangsläufig basierend auf Normen (keine Pflicht)

5. Gibt es eine Dokumentation zur Verteilung der IST-Kosten für Projekte?

- abhängig vom Projekt
- bei Großprojekten wird dies durchgeführt
- bei kleinen Projekten weniger
- Projektcontrolling ist immer wichtig
    - Budget ist wichtig für die Umsetzung
    - bei kleinen Projekten wird weniger Geld in technische Konzeptionsphase investiert
    - bei großen Projekten ist der Fokus anders

6. Berührungspunkte zum Anforderungsmanagement?

- ja, im Sinne von: "herausfinden was der Kunde will"

7. Aktuelle Verantwortlichkeit beim Anforderungsmanagement?

- TPL verantwortlich was die speziellen technischen Anforderungen des Kunden sind
    - projektspezifische Anforderungen (z.B. Nutzen für den Kunden) nicht

8. Gab es bereits Versuche zur Einführung von Methoden des Anforderungsmanagements?

- nein, nicht "lehrbuchmäßig"
- es wird auch zu wenig dokumentiert
    - wenig Vereinheitlichung
    - zu viel über E-Mails und mündliche Absprachen

Ermitteln

9. Vorbereitung für Kundentermine?

- Sichtung des vorhandenen Materials vom Kunden (aus dem Vorabbriefing)
- Absprache mit Kollegen
- meistens geschieht es mehr aus der Erfahrung heraus
- Argumente für bestimmte Umsetzungen aus den Dokumentation suchen

10. Übliche Formate für die Ermittlung von Anforderungen?

- Bestandskunden
    - eher einfach, dann kein Meeting
- Neukunden
    - Meeting empfohlen
- Meetings vor allem bei Personaländerung, um an das Projekt heran zu führen

11. Wird der Kunde/Werden die Projektbeteiligten an die Arbeit mit Anforderungen herangeführt?

- das ist abhängig von der Komplexität
- dem Kunden sollten in den ersten Meetings möglichst wenig Zahlen und Zeiten genannt werden
- die Agentur wird vorgestellt (mit Portfolio)
- das Vorgehen ist ein Teil des Angebots und dort festgehalten

12. Werden Entwickler an der Kommunikation mit dem Kunden beteiligt?

- technische Meinung wird abgefragt von eingebundenen Entwicklern
    - nur bei technischen Fragestellungen, sonst nicht

13. Welche Informationen kommen vom Kunden?

- Erstbriefings meistens nicht ausreichend
- Fragen ergeben sich vor allem in der Konzeptionsphase
- technische Expertise ist beim Kunden selten vorhanden
    - es ist Aufgabe des Fachpersonals nachzufragen
    - Kunde ist sich seiner Rolle als Product Owner nicht bewusst

Dokumentieren

14. Welche Dokumente werden innerhalb des Projektes erstellt?

- Word/Confluence
- Grobkonzept mit Grobangebot
    - für die Konzeptphase
- dann wird das Feinkonzept erstellt
- Ideal wären keine Änderungen nach Fertigstellung des Feinkonzepts
    - meistens gibt es jedoch späte Änderungen
    - je nach Größe führt das zu Änderungen im Feinkonzept oder nur über ein einfaches Ticket

15. Wie findet die Ablage von Dokumenten statt?

- SparkleShare
    - Git-Verwaltungstool
    - Versionierung auf Ebene des Dokuments
- ansonsten Confluence und Fileserver

16. Protokollieren/Ablegen von Kundengesprächen?

-

17. Dokumentieren von Fehlern/Problemen?

- kein Wissensmanagement

18. Welche Dokumente/Informationen bekommen die Entwickler?

- Projekt startet mit einem Kick-Off Meeting
    - dort werden Informationen bereit gestellt, wo welche Dokumente zu finden sind
    - Art der Aufgabe für den einzelnen Entwickler wird bekannt gegeben
- bei Drittanbietern gibt es eine API Dokumentation
- weiteres wird über Office Dokumente oder im Ticket kommuniziert

19. Enthalten Tickets im Redmine alle nötigen Informationen zur Lösung der Aufgabe?

- im Ticket sollten alle Informationen zur Umsetzung stehen
- aus der Praxis heraus gibt es jedoch auch Tickets, die in den Kommentaren wichtige Informationen enthalten
    - das ist schlecht
    - in den Kommentaren sollten keine neuen Anforderungen stehen
    - Änderungen müssen in der Beschreibung des Tickets ergänzt werden

20. Sollten Entwickler die Dokumente des Projektes lesen?

- eine gewisse Selbstständigkeit der Entwickler wird erwartet
    - z.B. API Dokumentation anschauen

Abstimmen, Prüfen & Validieren

21. Findet eine nachträgliche Validierung der unpräzisen Anforderungen unter Rücksprache mit dem Auftraggeber statt?

- "eigentlich" wird immer der Kunde gefragt
- es gibt jedoch eine Grenze der Eigengestaltung
    - die Größe der Anforderung ist hierbei entscheidend
- z.B. direkte 1:1 Übersetzung oder individuelle anderssprachige Inhalte
    - Änderung wäre gravierend
- bei kleinen Änderungen wird selbst entschieden
    - Kunde erwartet das von einer Agentur

22. Findet eine Verifikation der Anforderungen statt? Wer ist für die Verifikation der Anforderungen zuständig?

- es gibt eine interne QA
    - Prüfung ob Konzept umgesetzt wurde
- Abnahme durch Kunden erfolgt extra
    - wenn dann Änderungen der Anforderungen bestehen, handelt es sich um ein Kommunikationsproblem im Vorfeld

23. Sind Anforderungen jederzeit nachverfolgbar? (Traceability)

- genaue Nachvollziehbarkeit nur zum Teil
- die Freigabe des Konzepts ist wichtiger
- in der Theorie müssten alle Protokolle vom Kunden freigegeben werden
- in der Konzeptionsphase gibt es jedoch nicht für alle Meetings auch ein Protokoll
- danach soll jedoch alles dokumentiert werden

Verwalten

24. Wer ist zuständig für die Aktualität im Projekt erstellter Dokumente?

- PL muss den Überblick haben
- einzelner Entwickler verantwortlich für seinen Teil
- aber, für das Redaktionshandbuch auch PL verantwortlich
- bei technischer Dokumentation ist TPL oder Entwickler zuständig

25. Wann/Wie oft werden Dokumente aktualisiert?

- in der Theorie
    - bevor ein Release gemacht wird, sollten Dokumente aktualisiert werden
- in der Praxis
    - wird oft zuerst an der Dokumentation gespart

26. Gibt es eine Änderungshistorie für Anforderungen?

- nicht in dem Detailierungsgrad
- auf technischer Ebene jede Änderung
- bei großen Projekten jedes Feature
- Aufgabe liegt bei PL
    - Änderungen verwalten, die vom Kunden kommen

27. Wie laufen Änderungsanfragen des Kunden gewöhnlich ab?

- der Kunde wendet sich zunächst an den PL
    - entscheidet ob fachlicher Rat notwendig ist oder es selbst gelöst werden kann
    - wenn nicht, dann Übergabe an den PL
    - dieser löst es entweder selbst oder übergibt es dem Fachpersonal
    - nach Lösung zunächst zurück an TPL und von dort zurück zu PL

28. Werden vor Änderungen von Anforderungen Einflussanalysen durchgeführt?

- die technische Basis wird geprüft
    - Auswirkungen auf den Betrieb des Projektes prüfen
- auch unter Einbeziehung des Projektumfelds
    - Timing
    - Mehraufwand
- je nach persönlicher Einschätzung wird das Fachpersonal zur Einschätzung befragt

Abschluss

29. Lassen sich wiederkehrende Probleme im Entwicklungsprozess ableiten und wie sind diese zu Priorisieren?

- unklare Anforderungen
    - Ursache ist eine Mischung aus Versäumnissen auf Kunden und unserer Seite
    - Kunde kann evtl. keine klaren Anforderungen liefern
    - Nachfragen durch den TPL bleiben jedoch aus
    - teilweise werden auch große Dokumente erstellt, die wenig Inhalt haben
- Fehleinschätzungen von Tickets
    - dabei etwa 60% der Fehleinschätzung durch unklare Anforderungen
    - etwa 40% ist ein menschliches Problem
    - Entwickler neigen dazu wesentlich optimistischer einzuschätzen

30. Kein Projektende durch fehlende Baselines?

- prinzipiell haben Projekte immer ein Ende
- Gefühl kommt daher, dass Großprojekte über einen Zeitraum von 4-5 Jahren laufen
    - der PL sollte den Fokus auf eine Funktion oder ein Release legen, um Stückweise das Gefühl zu vermitteln, dass das Projekt voran kommt
- Kunde lehnt teilweise Empfehlungen ab, möchte jedoch später die empfohlene Änderung umgesetzt haben
    - beim Entwickler entsteht dadurch Frustration, weil evtl. viel Arbeitszeit in diese Umsetzung investiert wurde

31. Sind Nachverhandlungen durch fehlende Anforderungen üblich/selten/kein Thema?

- jedes großes Projekt unterliegt Änderungen
- bei kleinen Projekten sind es meistens kleine Änderungen
- entsteht oftmals durch unzureichende Beratungsleistungen
    - unsinnige Anforderungen werden ohne Rückfrage aufgenommen
    - z.B. ein Feature soll sofort umgesetzt werden, obwohl einige Wochen später ein gesammter Umbau angesetzt ist

32. Was läuft gut / nicht gut (muss verbessert werden) im Bereich AM?

siehe 31.

- es gibt diverse Punkte, die Probleme bereiten
- den Entwicklern ist unklar Was und Wie dokumentiert werden soll
- unklar: "Welche Anforderungen müssen vom Kunden eingefordert werden?"
- Junior Berater bringen Neuerungen, denen man Beachtung schenken sollte

Nachtrag

- Emi verwaltet ihre Anforderungen im Ticket
- Konzeptionsphasen sollten immer +30% sein (Zeitlich, finanziell?)
- fehlende Nachweise vor allem bei Telefonaten
    - Kunde: "Hab ich doch gesagt!"
    - niemand kann sich daran erinnern, aber es verursacht Mehraufwand
- es sollten immer Puffer eingebaut werden
- es gibt eine interne Verrechnung von Projekten unterschiedlicher finanzieller Höhe, um Unterschiede auszugleichen
- bei schwierigen Projekten oder Kunden, die man bereits kennt, muss mehr Puffer eingebaut werden
- es muss eine gemeinsame Basis geben, an der man sich orientiert, die jedoch für jedes Projekt angepasst wird
- der Umfang der Dokumentation ist auch abhängig vom Kunden
    - wird eine umfangreiche Dokumentation vom Kunden gefordert, dann muss sie auch geliefert werden

Interview 5

Einstiegsfragen

2. In welchem Gewerk tätig?

- TPL / Entwicklung (alles)

3. Erfahrungen mit klassischen/agilen Methoden?

- es werden keine agilen Methoden eingesetzt
- divergierende Methoden
    - keine feste Struktur
    - Arbeitsweise hängt von eigenem Können ab
    - ist nicht Projekt, sondern von Personen und Interessen getrieben

4. Einsatz von Industrienormen?

- kein praktischer Einsatzzweck
- es gibt selten Kunden, bei denen das Know-How nötig ist
- 80-90% der Projekte stellen keinen praktikablen Einsatzbereich dar
- die eigenen Prozesse sind nicht dazu geeignet sich an Industrienormen zu halten
- üblicherweise werden Seiten für einen Einsatz auf 3-4 Jahre erstellt
- es gibt im Projekt zu viele Stellschrauben

5. Gibt es eine Dokumentation zur Verteilung der IST-Kosten für Projekte?

- nicht direkt an der Erstellung beteiligt
    - nur Ermahnung/Hinweis an den PM dies durchzuführen
- letztendlich ist die Zeit entscheidend
- die Aufgabe sollte eher bei PM liegen, nicht bei Entwicklern
    - Praxis sieht anders aus

6. Berührungspunkte zum Anforderungsmanagement?

- permanent
- findet nicht statt oder ist zu diffus
- unterschiedliche Auffassung zu Aufgaben des PM
    - sollte in einer Agentur die auf Web ausgelegt ist wenigstens literarisches Wissen zur Informatik besitzen
    - sollte Prozesse nachverfolgen und modellieren können
- ohne technisches Verständnis ist es nicht möglich den Überblick zu behalten

7. Aktuelle Verantwortlichkeit beim Anforderungsmanagement?

- das Anforderungsmanagement liegt zur Zeit bei PM
- es sollte in PM und Konzeption aufgeteilt werden
    - derzeit ist "PM gefühlt bessere Sekretärin"

8. Gab es bereits Versuche zur Einführung von Methoden des Anforderungsmanagements?

- vereinzelt
- "Lehrbücher und Standards" nicht zwingend nötig
    - wichtiger ist als Grundvoraussetzung ein allgemeines Verständnis, was fehlt
- wichtige Fragestellungen beachten
    - Wer macht es?
    - Wofür ist es wichtig?
- danach kann man über Tools reden

Ermitteln

9. Vorbereitung für Kundentermine?

- Entwickler werden teilweise einbezogen, ist jedoch abhängig vom Kunden/Projekt
- wenn es nur um ein spezielles Thema geht, dann direkt Fragen stellen
    - Fragen und Stichpunkte vorab notieren
- ggf. bereits ein (Daten-)Modell vorbereiten
- aus der Theorie: Kunde hat einen Wunsch
    - daraus die Absicht, Ziel und Zielgruppe ableiten
    - mit Experten absprechen
    - gemeinsame Sprache finden
- meistens jedoch anders
    - Kunde sendet nur eine "Excel-Liste" mit Informationen
- auf Nachfragen wird schlecht reagiert
- es gibt selten die Bereitschaft vom Kunden das Thema vollständig zu durchdringen
- selbst wenn alles protokolliert ist, sind Kunden teilweise unzufrieden mit der Umsetzung nach ihren Vorgaben
    - Änderungswünsche werden dann trotz Protokoll durchgeführt
    - verschiedene Gründe dafür: Politik, Finanzen, Gewichtung des Kunden, ...

10. Übliche Formate für die Ermittlung von Anforderungen?

siehe 9.

11. Wird der Kunde/Werden die Projektbeteiligten an die Arbeit mit Anforderungen herangeführt?

- es ist schwierig sich mit dem Kunden auf eine Arbeitsweise zu einigen
    - nicht immer möglich den Kunden auf Arbeitsweise "fest zu zurren"
- man muss flexibel genug bleiben und nur eine Ausrichtung vorgeben

    - Was wollen wir?
    - In-Time, In-Budget und Kunde muss zufrieden sein
    - Welches Vorgehen ist in welcher Situation geeignet?
- Formalismen sollten sein, aber trotzdem muss die Flexibilität erhalten bleiben
- Verantwortlichkeiten sind wichtiger
- historisch war Anforderungsmanagement immer auf Tisch des Entwicklers

12. Werden Entwickler an der Kommunikation mit dem Kunden beteiligt?

- als TPL öfter
- als Entwickler weniger
- das hängt von den Personen und der Situation ab
- fehlende Einbindung von Entwicklern verursacht jedoch später oft Probleme

13. Welche Informationen kommen vom Kunden?

- man muss immer noch einmal nachfragen
    - Kunde hat kein technisches Verständnis und kann seine Anforderungen nicht definieren
- Kunde sieht nur seinen eigenen Bereich (Anwendungsdomäne)

Dokumentieren

14. Welche Dokumente werden innerhalb des Projektes erstellt?

- abhängig davon ob Teilangebot oder Großprojekt
- bei Großprojekten
    - unterschiedliche Kanäle und Senderichtungen (viele Beteiligte)
    - Strukturieren durch Confluence, mit Zugang durch Kunden
    - trotzdem Probleme bei Abnahme der Projekte
- es gibt keine 100% Lösung
- Tasks werden in Redmine mit Verweis auf Confluence angelegt
- der Kunde liest nicht die komplette Dokumentation
- mehr Dokumente verbessern nicht den Workflow, sondern erhöhen den Aufwand und überfordern den Kunden

15. Wie findet die Ablage von Dokumenten statt?

-

16. Protokollieren/Ablegen von Kundengesprächen?

-

17. Dokumentieren von Fehlern/Problemen?

- solche Issues selten
- im Redmine Wiki oder Confluence ablegen
    - Informationen wie "Extension nutzt Webservices" müssen nicht dokumentiert werden
- Problemlösungen sind zu schnelllebig
    - es gibt evtl. bereits bessere Lösungen, wenn das Problem erneut auftritt
- Besonderheiten sind zu selten
- "keine Dokumentation, deswegen Fehler" ist eine Ausrede

18. Welche Dokumente/Informationen bekommen die Entwickler?

- sollten auf so viele Informationen wie möglich Zugriff haben
- Junior oder Senior Entwickler haben unterschiedliche Anforderungen an die Menge der benötigten Informationen
- am besten die Tasks zusammen mit einem Entwickler definieren, dann ist der Detailgrad ausreichend
- Task/Feature sollte nicht direkt zu einem Entwickler gegeben werden
    - erst mit erfahrenem Entwickler/TPL absprechen
    - daraus einen Plan entwickeln und dann modellieren
    - realistisch kalkulieren

- bei Umsetzung einer ganzen Extension in eigenem Dokument beschreiben
- bei Umsetzung einer einzelnen funktionalen Erweiterung reicht ein Ticket
- "druckzeitlich begrenzte Ressourcenverteilung"
    - halbgare Übergabe von Tickets an den Entwickler
- durch fehlende Rücksprache mit ursprünglichen Entwicklern bei gewünschten Änderungen entstehen Fehler
    - individuelles Wissen wird nicht abgefragt
- oftmals mangelt es an der internen Kommunikation
    - kann nicht durch Dokumentation ersetzt werden
- situationsabhängige Entscheidung
- Ticket unterscheidet sich nach Können des Entwicklers
    - es sollte die Kreativität der Entwickler nicht einschränken
    - aber unerfahrene Entwickler weit genug zur Umsetzung anleiten
- Pareto-Prinzip beachten

19. Enthalten Tickets im Redmine alle nötigen Informationen zur Lösung der Aufgabe?

siehe 18.

20. Sollten Entwickler die Dokumente des Projektes lesen?

siehe 18.

Abstimmen, Prüfen & Validieren

21. Findet eine nachträgliche Validierung der unpräzisen Anforderungen unter Rücksprache mit dem Auftraggeber statt?

- unterschiedlich
- Fälle in denen Entwickler direkt umsetzt, ohne zu fragen
    - kann dann zu Problemen führen
- ist abhängig vom Kunden
    - Kunde mit konkreten Vorstellungen
    - oder Kunde lässt freie Hand
- Validierung findet statt

22. Findet eine Verifikation der Anforderungen statt? Wer ist für die Verifikation der Anforderungen zuständig?

- teilweise durch den Entwickler
- Anforderung und das was im Confluence hinterlegt ist unterscheiden sich
- Anforderungen sind umfangreicher als die Umsetzung
- PM sollte die Verifikation durchführen
- oftmals Verifikation On-the-Fly, dadurch Unterschiede zwischen Anforderungen des Kunden und dem, was in Confluence hinterlegt ist
- Kunde hat das Produkt an der Stelle abgenommen, was ist die Konsequenz?
    - Einplanung von Personen, die Qualität überprüft, obwohl bereits durch Kunden abgenommen
- insgesamt schwer zu beantworten
    - funktioniert nicht in allen Projekten
    - sorgt evtl. für Verzögerung des Projekts
    - in manchen Situationen funktioniert es gar nicht

23. Sind Anforderungen jederzeit nachverfolgbar? (Traceability)

- "jain"
- "Was ist die Konsequenz daraus?"
- selten nachvollziehbar
- man muss sich auf individuelles Wissen und Erfahrung verlassen

Verwalten

24. Wer ist zuständig für die Aktualität im Projekt erstellter Dokumente?

- PM

25. Wann/Wie oft werden Dokumente aktualisiert?

- bei der Änderung von Anforderungen
    - "Dokument im Fluss"
- PM ist "Single-Point-of-Contact"
    - muss sich fehlendes Wissen holen
    - muss die Übersicht behalten

26. Gibt es eine Änderungshistorie für Anforderungen?

- schlecht fest zu machen
- keine stringenten Anforderungen
- teilweise zu grobe Anforderungen
    - Bsp.: Umsetzung "Glossar als Akkordeon"
    - dann nachfragen
- dies in Dokumenten nieder zu schreiben ist schwierig
- keine dedizierte Änderungshistorie
- Confluence und Word bieten es auf Dokumentenebene

27. Wie laufen Änderungsanfragen des Kunden gewöhnlich ab?

- unterschiedlich, nicht genau fest zu machen

28. Werden vor Änderungen von Anforderungen Einflussanalysen durchgeführt?

- nein, nur kleine Einschätzungen

Abschluss

29. Lassen sich wiederkehrende Probleme im Entwicklungsprozess ableiten und wie sind diese zu Priorisieren?

- falsche Modellierung
    - führt im späteren Verlauf zu Fehlern
    - sind schwer irreversibel

30. Kein Projektende durch fehlende Baselines?

-

31. Sind Nachverhandlungen durch fehlende Anforderungen üblich/selten/kein Thema?

- häufiger

32. Was läuft gut / nicht gut (muss verbessert werden) im Bereich AM?

- Informationsdefizit
- implizite Annahmen
- Ressourcendefizit

Interview 6

Einstiegsfragen

2. In welchem Gewerk tätig?

- Backend-Entwickler

3. Erfahrungen mit klassischen/agilen Methoden?

- Frage zu offen gehalten

4. Einsatz von Industrienormen?

-

5. Gibt es eine Dokumentation zur Verteilung der IST-Kosten für Projekte?

-

6. Berührungspunkte zum Anforderungsmanagement?

- zu Anforderungen befragt nicht
- jedoch bereits Anforderungen kritisiert
- hat sich im Workflow ergeben

7. Aktuelle Verantwortlichkeit beim Anforderungsmanagement?

-

8. Gab es bereits Versuche zur Einführung von Methoden des Anforderungsmanagements?

- wurde versucht, ist jedoch gescheitert
- Workflow wurde von einigen Mitarbeitern nicht anerkannt
- PM und Kreativabteilung haben Entwickler wenig einbezogen

Ermitteln

9. Vorbereitung für Kundentermine?

- keine Einbeziehung in Kundentermine

10. Übliche Formate für die Ermittlung von Anforderungen?

siehe 9.

11. Wird der Kunde/Werden die Projektbeteiligten an die Arbeit mit Anforderungen herangeführt?

- Beteiligung am Kick-Off ist selten
    - aber im entferntesten Sinne: ja

12. Werden Entwickler an der Kommunikation mit dem Kunden beteiligt?

- keine Beteiligung, was jedoch gut ist
    - Verantwortung für die Kundenkommunikation liegt beim PM

13. Welche Informationen kommen vom Kunden?

- Informationen direkt vom Kunden sind meistens nicht zu gebrauchen

Dokumentieren

14. Welche Dokumente werden innerhalb des Projektes erstellt?

- keine Beteiligung, daher kein Wissen über Dokumente

15. Wie findet die Ablage von Dokumenten statt?

- viele Unklarheiten
    - Was existiert?
    - Wo liegen die Dokumente?
    - Wie kann ich darauf zugreifen?

16. Protokollieren/Ablegen von Kundengesprächen?

-

17. Dokumentieren von Fehlern/Problemen?

- kein Wissensmanagement

18. Welche Dokumente/Informationen bekommen die Entwickler?

- keine Dokumente

19. Enthalten Tickets im Redmine alle nötigen Informationen zur Lösung der Aufgabe?

- ja, aber
    - zu viele Informationen
    - müssen durchsucht werden
    - dort werden teilweise (ungefiltert) ganze E-Mail Dumps abgelegt
    - Zustand nicht zumutbar für den Entwickler

20. Sollten Entwickler die Dokumente des Projektes lesen?

- Kerninformationen sollten im Ticket sein, mit Mehrinformationen im Dokument

Abstimmen, Prüfen & Validieren

21. Findet eine nachträgliche Validierung der unpräzisen Anforderungen unter Rücksprache mit dem Auftraggeber statt?

-

22. Findet eine Verifikation der Anforderungen statt? Wer ist für die Verifikation der Anforderungen zuständig?

- Prüfung findet nur anhand des Tickets statt
    - dort jedoch redundante Informationen, die sich teilweise widersprechen
- Verantwortlichkeiten
    - Entwickler ist nicht zuständig dafür, wie es vom Kunden gewollt ist
    - Entwickler nur zuständig wie es auf Basis des Tickets umgesetzt wird (wie es gewollt sein könnte)
    - kann keine Garantie geben
- Testabteilung dafür nötig

23. Sind Anforderungen jederzeit nachverfolgbar? (Traceability)

- gar nicht nachvollziehbar
    - basierend auf den letzten Projekten

Verwalten

24. Wer ist zuständig für die Aktualität im Projekt erstellter Dokumente?

- PM
- eigentlich besser zwischen PM und Entwickler eine Schicht für die Kommunikation zu etablieren
    - PM ←→ ? ←→ Entw.

25. Wann/Wie oft werden Dokumente aktualisiert?

- keine Einsicht

26. Gibt es eine Änderungshistorie für Anforderungen?

- nein
- Änderungen werden auch nicht kommuniziert
- Entwickler ist die letzte Person, die es erfährt

27. Wie laufen Änderungsanfragen des Kunden gewöhnlich ab?

- keine Einsicht in ganzen Ablauf
- nur Ticket von PM, ohne vorherige Rückfrage

28. Werden vor Änderungen von Anforderungen Einflussanalysen durchgeführt?

siehe 27. (keine Rückfrage)

Abschluss

29. Lassen sich wiederkehrende Probleme im Entwicklungsprozess ableiten und wie sind diese zu Priorisieren?

- Informationsmanagement
    - nicht oder falsch weitergeleitete Informationen
    - Bindeglied fehlt
    - es lassen sich davon weitere Probleme ableiten

30. Kein Projektende durch fehlende Baselines?

-

31. Sind Nachverhandlungen durch fehlende Anforderungen üblich/selten/kein Thema?

-

32. Was läuft gut / nicht gut (muss verbessert werden) im Bereich AM?

- was gut ist, ist schwer zu sagen
    - fehlende Referenzen
- verschiedene Revisionen von Informationen vorhanden
    - falsche, widersprüchliche Informationen
    - bspw. wird etwas im Pattern Lab anders umgesetzt, als in der Anforderung beschrieben oder im Endprodukt umgesetzt

Interview 7

Einstiegsfragen

2. In welchem Gewerk tätig?

- Backend-Entwickler

3. Erfahrungen mit klassischen/agilen Methoden?

- kein Vorgehen nach Wasserfall
- ohne genaue Definition
    - Art von agiler Methode
    - nicht definierter, flexibler Prozess
- auch schon in anderen Agenturen innerhalb von Projekten mit agiler Methode gearbeitet
- "Kunden und Art der Projekt erlauben nichts anderes, als agil zu reagieren"
    - "Wie kann man das steuern?"
    - innerhalb der Iteration müssen Informationen in aktuelle Dokumente einfließen
- Wasserfall würde nicht funktionieren

4. Einsatz von Industrienormen?

-

5. Gibt es eine Dokumentation zur Verteilung der IST-Kosten für Projekte?

-

6. Berührungspunkte zum Anforderungsmanagement?

- bereits Auseinandersetzung mit Anforderungsmanagement
- Tätigkeit als PM
    - Entwicklung einer Beratungsmethode für Anforderungen

- vor der Angebotserstellung die Anfoderungen erheben
- Ausschreibungen mit klaren Anforderungen

7. Aktuelle Verantwortlichkeit beim Anforderungsmanagement?

-

8. Gab es bereits Versuche zur Einführung von Methoden des Anforderungsmanagements?

- Versuch zur Integration von Methoden
    - wurde nicht konsequent umgesetzt
    - nach erster Phase nicht weiter fortgeführt
    - das lag sowohl am Kunden, als auch an "uns"
- Anforderungsdokument für ein Projekt geschrieben
    - guter Start
    - Redmine-Referenz in Tickets auf das Dokument
    - positive Erfahrung

Ermitteln

9. Vorbereitung für Kundentermine?

- Einbindung in Kundentermine ist unterschiedlich
- i.d.R. werden Entwickler nicht eingebunden
- Teilaufgaben landen beim BE-Entwickler
    - dann beginnt ein Frage-Antwort-Zyklus
- Ticket muss erst in reale Aufgabe übersetzt werden
    - Rückfragen sind nötig
    - BE-Entwickler muss Annahmen treffen, wie es umgesetzt werden muss
    - anschließend Rücksprache mit dem TPL
- Fragen wie Dinge zu migrieren sind
    - Vorbereitung → Lösung für bessere Architektur skizzieren
    - stellt Umsetzungsmöglichkeit dar
    - mit dem Kunden wird eher über den Prozess gesprochen
- in anderer Agentur "grüne Wiese"-Erfahrungen
    - Mindmaps mit dem Kunden strukturieren und daraus nächsten Workshop vorbereiten
    - den Kunden fragen, was wichtig ist (Priorisierung)
    - nicht overengineeren
    - Unterteilung nach Arbeitspaketen

10. Übliche Formate für die Ermittlung von Anforderungen?

-

11. Wird der Kunde/Werden die Projektbeteiligten an die Arbeit mit Anforderungen herangeführt?

- sollte so sein
    - der Kunde muss geführt werden
- keine Einsicht, aber vom Gefühl her wird dies nicht gemacht
- man lässt sich von Kundenwünschen "hin und her treiben"
- Aufgaben werden nicht in der richtigen Reihenfolge umsetzt
- es werden viele Extrarunden gedreht
- späte Änderungen oder zu frühe Umsetzungen

12. Werden Entwickler an der Kommunikation mit dem Kunden beteiligt?

- es wird gefragt
    - oft zu einem Zeitpunkt, an dem man Probleme schlecht einschätzen kann
- Rahmenbedingungen ändern sich und werden nicht hinterfragt
- hängt vom PM ab
    - durchreichen ohne zu fragen (fehlende Beratung)

    - PM denkt zu kurz
    - Entwicklung braucht konzeptionellen Input
- Entwickler versucht den Code stabil zu halten
    - schwer bei häufigen Änderungen, die sofort umgesetzt werden sollen

13. Welche Informationen kommen vom Kunden?

- bei Webseiten liefert der Kunde wahrscheinlich genügend Informationen
- dem Kunden ist nicht klar, dass er ein komplexes Softwareprodukt in Auftrag gegeben hat
- teilweise sind die Prozesse dem Kunden selbst nicht klar
    - zu wenig Wissen über Prozesse und Modelle
    - man spricht nicht in der Sprache des Kunden
    - der Kunde muss an dieser Stelle besser abgeholt werden
- Kunde muss in der Lage sein, auf Detaillevel schneller zu reagieren
    - Prozesse sind nicht mit Use Cases oder User Stories abgebildet

Dokumentieren

14. Welche Dokumente werden innerhalb des Projektes erstellt?

- E-Mails
    - als negatives Beispiel
- zunehmend Tickets im Redmine
- formale Dokumente
    - selten, zumindest nach eigenen Anforderungen für Dokumente
- einzelne Dokumente
    - bspw. Excel-Tabellen
- Mockups
    - aber, es fehlen Angaben für die Funktionalität

15. Wie findet die Ablage von Dokumenten statt?

- Laufwerke mit Dokumenten
    - teilweise wurde bekannt gegeben, wo Dokumente liegen
    - sind nicht umfassend oder nicht mehr aktuell (Gültigkeit und Status unklar)
    - Nachteile der Ablage im Dateisystem (bspw.: gleichzeitiger Zugriff)
- kein DMS
    - Dateiablage fatal
    - Strukturen und Verantwortlichkeiten unklar
    - Aktualität wird nicht "erzwungen" (bspw.: DMS "Dokument ist abgelaufen")

16. Protokollieren/Ablegen von Kundengesprächen?

-

17. Dokumentieren von Fehlern/Problemen?

- nein
- eher durch Kommunikation
    - individuelles Wissen

18. Welche Dokumente/Informationen bekommen die Entwickler?

- Tickets
- Entwickler bekommt im Verlauf Dokumente
    - Wahrscheinlichkeit groß, dass diese nur flüchtig gelesen werden
- es braucht einen Einstiegspunkt mit allen nötigen Informationen
    - bspw.: Redmine Wiki
    - alle Beteiligten müssen diese aktuell halten
- ein Dokument am Anfang zu erstellen, an das sich über den Projektverlauf alle halten ist illusorisch
    - alle müssen an der Arbeit am Dokument beteiligt werden

- Dokumentation findet zu wenig statt

19. Enthalten Tickets im Redmine alle nötigen Informationen zur Lösung der Aufgabe?

- Aufgaben die beim Entwickler landen, sind teilweise schon qualifiziert worden
    - Zeit-/Aufwandsschätzung
    - ohne Rücksprache
- Referenzen im Ticket sind nicht unbedingt nötig
    - wenn Dokumente sich immer an der gleichen Stelle befinden

20. Sollten Entwickler die Dokumente des Projektes lesen?

- Mitarbeit der Entwickler an Dokumentation empfohlen
    - aus Erfahrung wissen Entwickler, was in die Dokumentation gehört

Abstimmen, Prüfen & Validieren

21. Findet eine nachträgliche Validierung der unpräzisen Anforderungen unter Rücksprache mit dem Auftraggeber statt?

-

22. Findet eine Verifikation der Anforderungen statt? Wer ist für die Verifikation der Anforderungen zuständig?

- Entwickler erhalten kein Anforderungsdokument
    - können nicht verifizieren
- Aufgabenbeschreibungen sind zu ungenau, um darüber zu verifizieren
- Umsetzung von Features, die nicht geplant waren
    - Kunde sagt ob Anforderung erfüllt ist oder nicht
    - kein dazugehöriges Dokument

23. Sind Anforderungen jederzeit nachverfolgbar? (Traceability)

- nein
    - theoretisch wissen alle Entwickler, dass Änderungen eine Quelle brauchen
- Tickets werden wie E-Mails behandelt
    - Entwickler muss sich die Informationen selbst zusammen fassen
- Anforderungen werden ausschließlich "im Laufen" aufgenommen
- es muss eine Form gefunden werden, mit der man es nachvollziehbar macht
- Kundenwünsche, die keine Anforderungen sind
    - Kunde will es evtl. gar nicht (als Wunsch, nicht als Anforderung formuliert)

Verwalten

24. Wer ist zuständig für die Aktualität im Projekt erstellter Dokumente?

-

25. Wann/Wie oft werden Dokumente aktualisiert?

-

26. Gibt es eine Änderungshistorie für Anforderungen?

-

27. Wie laufen Änderungsanfragen des Kunden gewöhnlich ab?

- keine Annahme von E-Mails durch die Entwickler
- Klärungsprozess läuft im Ticket ab
- teilweise werden in Tickets Anforderungen und Features beschrieben, die nichts mit dem ursprünglichen Ticket zu tun haben
- Änderungen werden auf einen "Haufen" geworfen

    - "muss noch gemacht werden"
    - werden nicht isoliert abgelegt
- die Entwickler müssen die PM bremsen
    - Feature in anderes Release verschieben
    - Topic für Release festlegen (Feature Release, Bugfix Release, ...)
- Änderungsanfragen werden nicht von PM gesteuert

28. Werden vor Änderungen von Anforderungen Einflussanalysen durchgeführt?

- es wird nur episodisch durchgeführt
    - ohne etablierten Prozess
- i.d.R. wird nicht über den Tellerrand geschaut
    - der Kunde erwartet Qualität
    - nach Wartbarkeit wird nicht gefragt
- es wird nicht systematisch iteriert
    - daher kann auch nicht systematisch refakturiert werden

Abschluss

29. Lassen sich wiederkehrende Probleme im Entwicklungsprozess ableiten und wie sind diese zu Priorisieren?

- Entwicklungen werden nicht ausreichend an Zielsystem ausgerichtet
- Onboarding ist aufwendig
    - nicht "mit einem Click"
- lokal unterschiedliche Abbildungen des Live-Systems
- Deployment nicht automatisiert
    - keine großartigen Fortschritte in dem Bereich
    - ausliefern ist fehlerträchtiger, aufwendiger Prozess
    - stellt einen fundamentalen Fehler dar
    - "Wie kann lokal so gearbeitet werden, wie auf dem Live-System?"
- zu späte und zu viele manuelle Tests
- späte Änderungen
    - erzeugt zu viele Fehler bei Live-Gang
    - betrifft vor allem unbekannte Fehler
    - während Fehler behoben werden, kommen neue Features dazu
- viele Projekte produzieren "One-Track-Scripts"
    - geringe Rolle von Widerverwendbarkeit und Wartbarkeit
    - keine Verallgemeinerung von Problemlösungen
    - Zeit ist vielen wichtiger

30. Kein Projektende durch fehlende Baselines?

-

31. Sind Nachverhandlungen durch fehlende Anforderungen üblich/selten/kein Thema?

-

32. Was läuft gut / nicht gut (muss verbessert werden) im Bereich AM?

- keine technische Konzeption
    - Konzeption wird ohne technische Kompetenz erstellt
    - es werden nicht die richtigen Leute gefragt / Fragen gestellt
    - es wird eher nach optischen Merkmalen gefragt, nicht die Rahmenbedingungen oder technischen Fragestellungen betrachtet
- konzeptionelle Vorarbeit
    - Fragestellung: "Was muss es können?"

Interview 8

Einstiegsfragen

2. In welchem Gewerk tätig?

- Einstieg Integration, dann FE-Entwicklung

3. Erfahrungen mit klassischen/agilen Methoden?

- nur klassischer Wasserfall
- wirklich agil gar nicht
    - ein bisschen mit bestimmten Kunden

4. Einsatz von Industrienormen?

- Erfahrungen mit DIN-Normen
- mit der Begrifflichkeit vorsichtig sein
    - man könnte sie sich evtl. anschauen
- Projekte an und für sich betrachten
    - keine starren Projektstrukturen

5. Gibt es eine Dokumentation zur Verteilung der IST-Kosten für Projekte?

-

6. Berührungspunkte zum Anforderungsmanagement?

- in der Konzeption
    - Anforderungen erfragen
- was der Kunde erzählt, muss nicht das sein, was er will

7. Aktuelle Verantwortlichkeit beim Anforderungsmanagement?

-

8. Gab es bereits Versuche zur Einführung von Methoden des Anforderungsmanagements?

- die "Wall" als Prozessmodell
    - erstellt aus eigenem Learning und Erfahrungen aus anderen Agenturen
    - deckt von Akquise des Kunden bis Umsetzung alles ab

Ermitteln

9. Vorbereitung für Kundentermine?

- Einbeziehung von Entwicklern "wird besser"
    - wurde bereits länger gefordert
- eher bei größeren Projekten
- FE-Entwickler sollten immer mit dabei sein, auch bei kleinen Terminen
    - Umsetzung diskutieren (Einschränkungen, Empfehlungen)
    - auch BE-Entwickler
- In welche Richtung will der Kunde?
    - stellt Gesprächsgrundlage dar
- früher wurden 4-5 Designs erstellt, heute wird iterativ ein Design erstellt
- zu jedem Meeting soll etwas Neues mitgebracht werden
    - Mockups, Entwurf oder Prototyp
    - langsam herantasten

10. Übliche Formate für die Ermittlung von Anforderungen?

- vor allem Meeting
- Telefongespräche sind abhängig von Länge und Thema auch gut
- problematisch sind E-Mails
    - mögliche Missverständnisse

    - sprachlich lässt es sich leichter abstimmen

11. Wird der Kunde/Werden die Projektbeteiligten an die Arbeit mit Anforderungen herangeführt?

- ja, mit der "Wall"
- in der Akquise schon darlegen
- erklären: "Warum arbeiten wir so?"
    - Vorteile der Arbeitsweise
- Transparenz gegenüber dem Kunden schaffen

12. Werden Entwickler an der Kommunikation mit dem Kunden beteiligt?

- es wird Rücksprache gehalten
    - aber budgetabhängig
    - dann Entscheidung auf PM Ebene
- sichere Entscheidungen können nur gemeinsam getroffen werden
- je größer das Projekt, desto größer der Einfluss

13. Welche Informationen kommen vom Kunden?

- selten kommen die Informationen, die man benötigt
    - deswegen auch E-Mails schwierig
- Informationen sind oftmals auch "mit Vorsicht zu genießen"
- der Kunde kann seinen Wunsch nicht richtig formulieren
- selbst wenn der Kunde weiß, was er will, muss es nicht das Beste für ihn sein
    - Beratungsleistungen erbringen

Dokumentieren

14. Welche Dokumente werden innerhalb des Projektes erstellt?

- vor allem E-Mails
- Protokolle
    - "sollten" ausgefüllt werden
    - oftmals kommen die Informationen nicht an
- Meeting-Protokolle
    - werden nachträglich ins Confluence übernommen
- Projektfahrplan für kleinere Projekte
- Angebot
- Wireframes
- Post-its
    - werden abfotografiert und ins Confluence übertragen
- Dokumentation im Code
- Briefing-Protokolle
    - Protokoll für die Übergabe von Design an Frontend-Entwicklung

15. Wie findet die Ablage von Dokumenten statt?

- Redmine
- Confluence
- Fileserver
    - jedes Projekt hat eine eigene Ordnerstruktur

16. Protokollieren/Ablegen von Kundengesprächen?

siehe 14

17. Dokumentieren von Fehlern/Problemen?

- relativ wenig
- Fehler sind zu schnelllebig
    - Fehler sind im nächsten Projekt evtl. schon nicht mehr relevant

- Feedback-Meeting
    - weniger technisch, eher auf das Projekt bezogen
- Absprachen und Wissenstransfer passiert eher individuell
- "Golden Master"
    - für CMS und FE
    - Anfangsaufwand minimieren
    - übliche Fehler im Projektstart durch Vorarbeit vermeiden

18. Welche Dokumente/Informationen bekommen die Entwickler?

- Briefing-Protokoll als Vorlage für die Erstbefüllung
- Mix aus Ticket und Confluence
    - im Confluence mit Screenshot und Beschreibung
    - im Redmine die Tickets

19. Enthalten Tickets im Redmine alle nötigen Informationen zur Lösung der Aufgabe?

- Informationen werden in Tickets festgehalten
    - daraus werden Untertickets erstellt mit einzelnen Features erstellt
    - nicht den Blick für das Ganze verlieren
- viele Kommentare im Ticket
    - Entwickler möchten nicht alles lesen
    - muss aufgearbeitet sein
    - unterschiedliche Sprache (Kunde, PM, Design, FE und BE)
- Verweise auf Dokumente selten

20. Sollten Entwickler die Dokumente des Projektes lesen?

-

Abstimmen, Prüfen & Validieren

21. Findet eine nachträgliche Validierung der unpräzisen Anforderungen unter Rücksprache mit dem Auftraggeber statt?

-

22. Findet eine Verifikation der Anforderungen statt? Wer ist für die Verifikation der Anforderungen zuständig?

- muss vom PM im Auge behalten werden, nicht vom Entwickler
- QS fällt als erstes runter
    - bei Zeitmangel

23. Sind Anforderungen jederzeit nachverfolgbar? (Traceability)

- geht meistens verloren
- im Redmine keine Hinweise
    - deswegen gut in den Meetings zu sitzen

Verwalten

24. Wer ist zuständig für die Aktualität im Projekt erstellter Dokumente?

- PM muss die Dokumente im Auge behalten
    - Umsetzung liegt jedoch bei den Abteilungen
- Problem ist, dass Dokumente bereits veralten, sobald sie abgelegt werden
- Verständnis für solche Dokumente, aber langfristig skeptisch (bzgl. der Aktualität)
- im FE muss der Code gut und kommentiert sein
    - eine gesonderte Dokumentation ist schwierig umzusetzen

25. Wann/Wie oft werden Dokumente aktualisiert?

siehe 24.

26. Gibt es eine Änderungshistorie für Anforderungen?

- Versionierung mit Git
    - in Kombination mit Redmine lassen sich Änderungen an Features nachvollziehen
- sonst nicht

27. Wie laufen Änderungsanfragen des Kunden gewöhnlich ab?

- grundlegend so wie das Projekt zu Beginn abläuft
    - nur in kleineren Kreisen
- Anfrage → Abstimmung → Umsetzung → QS → Übergabe an den Kunden

28. Werden vor Änderungen von Anforderungen Einflussanalysen durchgeführt?

-

Abschluss

29. Lassen sich wiederkehrende Probleme im Entwicklungsprozess ableiten und wie sind diese zu Priorisieren?

- Zeitdruck
    - Deadlines passen nicht zur Kalkulation
- Änderungen der Anforderungen während der Umsetzung
- Informationsverlust an den Schnittstellen (Stille-Post-Prinzip)
    - Informationen werden durchgereicht und verändern sich dabei

30. Kein Projektende durch fehlende Baselines?

-

31. Sind Nachverhandlungen durch fehlende Anforderungen üblich/selten/kein Thema?

-

32. Was läuft gut / nicht gut (muss verbessert werden) im Bereich AM?

- Kunde will etwas anderes, als besprochen wurde
    - mehr Kommunikation mit dem Kunden
    - Fragestellung: "Wo will der Kunde wirklich hin?"

Interview 9

Einstiegsfragen

2. In welchem Gewerk tätig?

- erst Integration, dann FE-Entwicklung

3. Erfahrungen mit klassischen/agilen Methoden?

- erst Konzepter, aber FE-Entwickler früh mit einbeziehen
- Arbeit mit Pattern Lab
- durchläuft viele Schleifen zur Optimierung
- Integration findet erst am Ende statt
- Confluence
    - MVC-Einteilung
    - jeder hat Zugriff auf Confluence und kann eintragen/ändern
    - Informationen können von jedem entnommen werden

4. Einsatz von Industrienormen?

-

5. Gibt es eine Dokumentation zur Verteilung der IST-Kosten für Projekte?

-

6. Berührungspunkte zum Anforderungsmanagement?

- ständig Berührungspunkte durch Arbeit mit den Kunden
    - Rückversicherung mit Kollegen zu den Anforderungen

7. Aktuelle Verantwortlichkeit beim Anforderungsmanagement?

-

8. Gab es bereits Versuche zur Einführung von Methoden des Anforderungsmanagements?

- keine Aussage möglich

Ermitteln

9. Vorbereitung für Kundentermine?

- ja
    - bisher eher wenige Projekte
- Kick-Off Meeting Teilnahme
    - eher Informationen erhalten/mitnehmen
    - Schritt davor ist Brainstorming
    - "Was sind die Kunden?"
    - Flipchart mit ersten Eindrücken
    - nicht alle Bereiche integriert
- Fragestellung: "Wen holt man mit in welches Meeting?"

10. Übliche Formate für die Ermittlung von Anforderungen?

- keine direkt Aussage möglich
- Zuständigkeit liegt eher bei Konzept und Design
- höchstens Research für einzelne Anforderungen betreiben
    - bspw.: wie werden bestimmte Elemente aktuell umgesetzt

11. Wird der Kunde/Werden die Projektbeteiligten an die Arbeit mit Anforderungen herangeführt?

- i.d.R. ist das so

12. Werden Entwickler an der Kommunikation mit dem Kunden beteiligt?

- ja
- Machbarkeit und Aufwandsschätzung

13. Welche Informationen kommen vom Kunden?

- indirekte Informationsweitergabe von Kunden über PM
    - Informationen die ankommen sind in Ordnung

Dokumentieren

14. Welche Dokumente werden innerhalb des Projektes erstellt?

- Confluence und Projektfahrplan
    - stellt bis ins kleinste Detail dar, was gewünscht wird
- Briefing auf Papier
    - sobald ausgefüllt erfolgt die Umsetzung
- Confluence mit MVC

    - auch BE-Entwickler tragen einen Teil dazu bei
    - wird während des Projekts mit Informationen gefüttert
    - "funktioniert super"
    - Sammlung aller Informationen, wie die komplette Website aussehen soll
    - sorgt für transparente Dokumentation

15. Wie findet die Ablage von Dokumenten statt?

siehe 14.

16. Protokollieren/Ablegen von Kundengesprächen?

- durch Confluence Einblicke
    - Gespräche mit Kunden
    - enthalten Aufgaben mit Deadlines und die Antworten vom Kunden
    - stellt eine Art Protokoll dar

17. Dokumentieren von Fehlern/Problemen?

- noch nicht in der Form an einem Problem hängen geblieben
    - Input über interne Kommunikation/Kollegen hilft weiter

18. Welche Dokumente/Informationen bekommen die Entwickler?

- Confluence
- Redmine
- Briefing-Protokoll

19. Enthalten Tickets im Redmine alle nötigen Informationen zur Lösung der Aufgabe?

- wünschenswert, wenn im Redmine Ticket alles klar wäre
    - vermutlich nicht möglich
    - Rückversicherung über Confluence
    - kein Problem mit Redmine und Confluence in Kombination
- Rückfrage bei Unklarheiten
- PM sollte Überblick über das Große und Ganze haben

20. Sollten Entwickler die Dokumente des Projektes lesen?

siehe 19.

Abstimmen, Prüfen & Validieren

21. Findet eine nachträgliche Validierung der unpräzisen Anforderungen unter Rücksprache mit dem Auftraggeber statt?

-

22. Findet eine Verifikation der Anforderungen statt? Wer ist für die Verifikation der Anforderungen zuständig?

- für den eigenen Teil
- eine Art QS für jeden Teil sollte zumindest drauf schauen
    - Zeit ist nicht immer dafür vorhanden
    - QS sollte weiter oben stehen

23. Sind Anforderungen jederzeit nachverfolgbar? (Traceability)

- keine Aussage möglich

Verwalten

24. Wer ist zuständig für die Aktualität im Projekt erstellter Dokumente?

- für Confluence ist nicht nur eine Person zuständig

    - keine einheitliche Zuständigkeit
    - jeder hält Informationen aktuell
- es wäre auch möglich im Redmine Wiki zu arbeiten
- Dokumentation sollte nicht zu viel werden
    - alle müssen sich an der Dokumentation beteiligen

25. Wann/Wie oft werden Dokumente aktualisiert?

- zeitlich nicht möglich regelmäßig alle Dokumeten zu aktualisieren
- die wichtigsten Dokumente müssen aktuell sein

26. Gibt es eine Änderungshistorie für Anforderungen?

- über Pattern Lab/Git und Confluence abbildbar

27. Wie laufen Änderungsanfragen des Kunden gewöhnlich ab?

- Selbstverständlichkeit nach Machbarkeit zu fragen
    - Wunsch der Kunden und Machbarkeit in Einklang bringen
    - lässt sich meistens im Gespräch lösen, ob etwas machbar ist oder nicht
    - Bedenken äußern und Lösung finden
- i.d.R. betrifft das nicht direkt die FE-Entwicklung
    - keine Details zum genauen Prozess

28. Werden vor Änderungen von Anforderungen Einflussanalysen durchgeführt?

-

Abschluss

29. Lassen sich wiederkehrende Probleme im Entwicklungsprozess ableiten und wie sind diese zu Priorisieren?

- extremer Zeitdruck (zeitweise)
    - zu viele Aufgaben gleichzeitig

30. Kein Projektende durch fehlende Baselines?

-

31. Sind Nachverhandlungen durch fehlende Anforderungen üblich/selten/kein Thema?

-

32. Was läuft gut / nicht gut (muss verbessert werden) im Bereich AM?

- keine Aussage möglich

Interview 10

Einstiegsfragen

2. In welchem Gewerk tätig?

- Design/Konzeption

3. Erfahrungen mit klassischen/agilen Methoden?

- diverse Erfahrungen vorhanden
- Wasserfall "leider" bekannt
- optimiert gerne die vorhandenen Prozesse
- seit 2011 eine bessere Übergabeform mit den Entwicklern herausgebildet
    - agiler Workflow (mit Nils entwickelt)
- verschiedene Erfahrungswerte

    - mit dem Prozess soweit zufrieden, sofern diesern eingehalten wird
- von der Akquise geht es in eine aufgefächerte Konzeptionsphase
    -
        1. Phase besteht aus Konzeption und Design
    -
        1. Phase ist dann die Umsetzung
- auf Papier ohne FE-Entwickler lässt sich dabei nicht die Stimmung einfangen, die erzeugt werden soll
- erst die Gesamtzusammenarbeit ergibt eine "coole Seite"
- zwischen den zwei Phasen ist eine "gestrichelte Linie", bis dort gibt es noch viele Änderungen
    - dann erst "Feature-Freeze" und 2. Projektphase
    - in der 2. Phase weniger agil, weil dort die Umsetzung stattfindet
- das Konzept muss praktikabel sein
- dafür wird Confluence genutzt
    - MVC-Prinzip
    - alles an einer Stelle dokumentiert

4. Einsatz von Industrienormen?

-

5. Gibt es eine Dokumentation zur Verteilung der IST-Kosten für Projekte?

-

6. Berührungspunkte zum Anforderungsmanagement?

siehe 3.

- ja

7. Aktuelle Verantwortlichkeit beim Anforderungsmanagement?

-

8. Gab es bereits Versuche zur Einführung von Methoden des Anforderungsmanagements?

siehe 3.

- ja

Ermitteln

9. Vorbereitung für Kundentermine?

- wird mit in Kundentermine einbezogen
    - Kick-Off Termin
    - bei neuen Projekten "Kreativ-Workshop"
    - dieser wird auch zusammen (CPS+FR) abgehalten, um den Zusammenschluss zu fördern
- "Kreativ-Workshop"
    - Phase 1
        - Zielfestlegung (Brainstorming)
        - eher grob, teilweise Zoom
        - nicht nur Designer, auch Redakteure und Entwickler
    - Phase 2
        - Ideen für den Workshop mit dem Kunden sammeln
        - z.B. Slogan
    - kann man auch unter Einbeziehung des Kunden durchführen
- Wichtig: "User ist größer als Kunde"
    - in welche Richtung soll es gehen
- Fragebogen

    - priorisieren von Inhalten
    - reduzieren, clustern, kopieren
    - Wie sind die Seiten aufgebaut?
- Wireframes für den Kundentermin

10. Übliche Formate für die Ermittlung von Anforderungen?

siehe 9.

11. Wird der Kunde/Werden die Projektbeteiligten an die Arbeit mit Anforderungen herangeführt?

- ja
    - bestehende Projekte zeigen
    - Einblicke ins Confluence geben
- Vertreter des Kunden fragen nach "Startseite für Vorgesetzte"
    - Kunden überzeugen, dass Inhalte des CMS nicht statisch sind
- auf Designelemente beschränken
- Stückweise in Absprache entwickeln
- einzelne Elemente im Prototyp zeigen
- das Vorgehen ist schwierig bei Konzernen
    - benötigen meist große Präsentationen und wollen alles statisch auf Papier
    - Feingefühl gefordert

12. Werden Entwickler an der Kommunikation mit dem Kunden beteiligt?

- ja

13. Welche Informationen kommen vom Kunden?

- kommt darauf an
- übliche Aussagen
    - wenig Budget
    - schneller Abschluss
    - wenig Zeit für Beteiligung
- wichtig ist immer ein Ansprechpartner beim Kunden
- Ableiten, Analysieren als Aushilfe
- es gibt jedoch Probleme, wenn bestimmte Ressourcen nicht bereitgestellt werden (bspw. Logo)
- verschiedene Arten von Kunden
    - z.B.: große Dokumenten, aber wenig Aussagekraft
    - z.B.: keine Produkte, nur Thema
    - Konzept muss trotzdem entwicklet werden

Dokumentieren

14. Welche Dokumente werden innerhalb des Projektes erstellt?

- Wireframes
- Mindmaps
- Grafik (Illustrator, PS)
- Dokumentation ansonsten im Confluence

15. Wie findet die Ablage von Dokumenten statt?

- Confluence
- Fileserver
    - festgelegte Ordnerstruktur, anders als PM Struktur
    - Versionierung über Ordnerbezeichnungen

16. Protokollieren/Ablegen von Kundengesprächen?

-

17. Dokumentieren von Fehlern/Problemen?

- kein Wissensmanagement
- es ist bereits ein Thema, mit dem man sich beschäftigt
    - wird zur Zeit bespochen und entwickelt
- alle 14 Tage Meeting mit guten/schlechten Beispielen

18. Welche Dokumente/Informationen bekommen die Entwickler?

- kleine Aufgaben kommen per Ticket
- Projektdokumente sind verfügbar
    - versch. Dokumente als Word, Excel, bzw. in Confluence

19. Enthalten Tickets im Redmine alle nötigen Informationen zur Lösung der Aufgabe?

- Redmine als Schnittstelle zu anderen Kollegen
- schlecht geeignet für Kreativ/Design
- Tickets ergeben sich erst bei Übergabe an den Entwickler
- Darstellung der Anhänge im Redmine ist problematisch

20. Sollten Entwickler die Dokumente des Projektes lesen?

- Konzepter sollte alles lesen und auch recherchieren
    - Neuerungen kennen
    - der Anspruch ist: mehr als der Kunde erwartet
- Was muss visualisiert werden?
- Designer soll sich auf den kreativen Teil konzentrieren,, nicht auf das Vorgehen vom Kunden

Abstimmen, Prüfen & Validieren

21. Findet eine nachträgliche Validierung der unpräzisen Anforderungen unter Rücksprache mit dem Auftraggeber statt?

-

22. Findet eine Verifikation der Anforderungen statt? Wer ist für die Verifikation der Anforderungen zuständig?

- ja
- nicht genau im Konzept zu sagen
- Mischung aus konzeptioneller Inhalt und Design
- muss fachlich gut sein
- Vordenken für FE-Entwickler
- analoges Dokument mit den Fakten (Patternlab-Protokoll)
- Barrierefreiheit, Lesbarkeit, Struktur, ...

23. Sind Anforderungen jederzeit nachverfolgbar? (Traceability)

- über E-Mails, Historie nur durch Person bekannt
- es werden Rebriefings / Schulterblicke durchgeführt
- Tools werden erst getestet
- Projektfahrplan
    - Nachvollziehbarkeit für den Projektfortschritt
    - Kunde hat darauf Zugriff

Verwalten

24. Wer ist zuständig für die Aktualität im Projekt erstellter Dokumente?

- einzelne Projektverantwortliche

25. Wann/Wie oft werden Dokumente aktualisiert?

- regelmäßige Abstimmung über Status der Dokumente
- Server macht häufig Probleme

    - Dokumente werden daher lokal überspielt, aber es wird vergessen diese wieder auf den Server zurück zu führen
- teilweise wird dadurch auch die Verknüpfung von Dokumenten verloren

26. Gibt es eine Änderungshistorie für Anforderungen?

- Versionierung geschieht durch Ordner
    - Projektstände/-pakete
    - oder Kundenfeedback und folgende Änderung
- keine goldene Regel bei Versionierung, eher Intuition

27. Wie laufen Änderungsanfragen des Kunden gewöhnlich ab?

- bei bestehenden Projekten/Bestandskunden
    - Ticket → Änderung → Antwort
- bei neuen Projekten/Neukunden
    - Einarbeitung der Änderung
    - geschieht über mehrere Runden von Änderungen
    - Styletiles werden erzeugt
    - Designer konzentriert sich auf einzelne Elemente
    - mehrere Versionen von Elementen, Kunde entscheidet dann (gute Erfahrung)
    - "Kunde hat Gefühl zu bestimmen"
    - hohe Zufriedenheit, wenig Korrekturen notwendig

28. Werden vor Änderungen von Anforderungen Einflussanalysen durchgeführt?

-

Abschluss

29. Lassen sich wiederkehrende Probleme im Entwicklungsprozess ableiten und wie sind diese zu Priorisieren?

- viele Probleme wurden mit dem angesprochenen Prozess beseitigt
- es werden manchmal TYPO3 Funktionen vergessen
    - bspw. Rahmen von TYPO3, valid-Farbe, Tabellenstil
    - dafür gibt es ein Formular für Pattern Lab zur Erstbefüllung
- Pattern Lab Formular muss noch zu Mobile First umgestellt werden
    - prograssive Enhancemennt
    - gute Basisstruktur

30. Kein Projektende durch fehlende Baselines?

-

31. Sind Nachverhandlungen durch fehlende Anforderungen üblich/selten/kein Thema?

-

32. Was läuft gut / nicht gut (muss verbessert werden) im Bereich AM?

- es sollte das Bewusstsein dafür, dass AM wichtig ist geschärft werden
    - im Angebot muss eine Konzeptionsphase vorhanden sein, auch bei kleinen Projekten
    - PMs müssen diesen Prozess vereinheitlichen
- kleine Projekte ohne Konzeptionsphase sind meist wesentlich zeitaufwendiger als angesetzt

Nachtrag

- Confluence Projektphasen-Wand
    - bildet die Projektphasen mit jeweiligen Rollen ab
    - Ziel der Phase und Inhalt der Übergabe
    - Phasen

    1. Ziel definieren
    2. Wer ist der User?
    3. Content analysieren
    4. Design-Atmosphäre
    5. Prototyp
- Problem ist auch, dass Vorgehen/Prozesse/Dokumente von einigen Mitarbeitern ignoriert werden