Certified Professional for Requirements Engineering

Foundation Level

Lehrplan

Stan Bühne
Martin Glinz
Hans van Loenhoud
Stefan Staal

Nutzungsbedingungen

1. Einzelpersonen und Seminaranbieter dürfen den Lehrplan als Grundlage für Seminare verwenden, sofern die Inhaber der Urheberrechte als Quelle und Besitzer des Urheberrechts anerkannt und benannt werden. Des Weiteren darf der Lehrplan zu Werbezwecken nur mit Einwilligung des IREB e.V. verwendet werden.
2. Jede Einzelperson oder Gruppe von Einzelpersonen darf den Lehrplan als Grundlage für Artikel, Bücher oder andere abgeleitete Veröffentlichungen verwenden, sofern die Autoren und der IREB e.V. als Quelle und Besitzer des Urheberrechts genannt werden.

Danksagung

Die ursprüngliche Version des Lehrplans wurde 2007 von Karol Frühauf, Emmerich Fuchs, Martin Glinz, Rainer Grau, Colin Hood, Frank Houdek, Peter Hruschka, Barbara Paech, Klaus Pohl und Chris Rupp erstellt. Sie wurden dabei von den IREB Mitgliedern Ian Alexander, Joy Beatty, Joseph Bruder, Samuel Fricker, Günter Halmans, Peter Jaeschke, Sven Krause, Steffen Lentz, Urte Pautz, Suzanne Robertson, Dirk Schüpferling, Johannes Staub und Thorsten Weyer unterstützt.

Die Version 3 ist eine umfassende Überarbeitung, die von Stan Bühne, Martin Glinz, Hans van Loenhoud und Stefan Staal erstellt wurde. Sie wurden von Karol Frühauf, Rainer Grau, Kim Lauenroth, Chris Rupp und Camille Salinesi unterstützt.

Während dieser Überarbeitung gaben Xavier Franch, Karol Frühauf, Rainer Grau, Frank Houdek und Thorsten Weyer Anregungen zum Inhalt. Ergänzende Anmerkungen kamen von Wim Decoutere und Hans-Jörg Steffe.

Reviews wurden von Christoph Ebert, Barbara Paech und Chris Rupp durchgeführt.

Die Freigabe erfolgte durch das IREB Council, auf Empfehlung von Xavier Franch und Frank Houdek, am 22. Juli 2020.
Übersetzt aus dem Englischen durch Sibylle Becker, Stan Bühne, Ruth Rossi und Stefan Sturm.

Allen sei für ihr Engagement gedankt.

Präambel

Im Sommer 2017 haben wir eine Umfrage durchgeführt, um die Relevanz der bestehenden Certified Professional for Requirements Engineering (CPRE) Foundation Level-Zertifizierung (Version 2.2) zu untersuchen. Ziel der Umfrage war es, Feedback über die praktische Marktrelevanz dieser Zertifizierung aus Perspektive der Schulungsanbieter sowie von zertifizierten CPRE-Praktikern, die als Requirements Engineers arbeiten, zu erhalten [MFeA2019]. Die Umfrage ergab, dass der bestehende Lehrplan für den CPRE Foundation Level v2.2 im Allgemeinen noch immer die wichtigsten Bedürfnisse des Marktes erfüllt und dem Kandidaten das relevante RE-Wissen vermittelt. Dennoch erhielten wir die Rückmeldung, dass einige Techniken in der Praxis nicht mehr verwendet werden, während andere fehlen, wenn man eine iterative und adaptive Entwicklung anstrebt. Dieses Feedback entsprach IREBs eigener Wahrnehmung der Veränderungen im Bereich des Requirements Engineering (RE). Wir haben daher den CPRE Foundation Level Lehrplan grundlegend überarbeiten, veraltete Inhalte entfernt und neue Inhalte hinzugefügt. Version 3 des Lehrplan spiegelt den Stand des heutigen RE wider und umfasst sowohl planbasierte als auch agile Ansätze zur Spezifizierung und Verwaltung von Anforderungen.

Von Kandidaten, die eine CPRE-Zertifizierung gemäß diesem Lehrplan anstreben, wird erwartet, dass sie Grundkenntnisse in der Systementwicklung mit planbasierten und agilen Ansätzen haben.

Zweck des Dokuments

Dieser Lehrplan definiert die Basisstufe (Foundation Level) der Certified Professional for Requirements Engineering Zertifizierung des International Requirements Engineering Board (IREB). Der Lehrplan dient den Schulungsanbietern als Grundlage für die Erstellung ihrer Kursunterlagen. Die Lernenden können sich anhand des Lehrplans auf die Prüfung vorbereiten.

Inhalt des Lehrplans

Das Foundation Level richtet sich an alle Personen, die in das Thema Requirements Engineering involviert sind. Dazu gehören Personen in Rollen wie Requirements Engineer, Business Analyst, Systemanalytiker, Product Owner oder Produktmanager, Entwickler, Projekt- oder IT-Manager oder Domänenexperte.

Im Lehrplan sowie im dazugehörigen Handbuch wird die Abkürzung RE für Requirements Engineering verwendet.

Inhaltsabgrenzung

Das CPRE Foundation Level vermittelt Grundlagen, die für alle Arten von Systemen (z.B. mobile Anwendungen, Informationssysteme oder cyber-physische Systeme) gleichermaßen gelten. Das CPRE Foundation Level setzt weder einen spezifischen Entwicklungsprozess voraus, noch ist es auf einen bestimmten Anwendungsbereich ausgerichtet. Schulungsanbieter können Schulungen anbieten, die sich auf bestimmte Arten von Systemen, Prozessen oder Anwendungsbereichen konzentrieren, solange die Lernziele dieses Lehrplans vollständig abgedeckt werden.

Detaillierungsgrad

Der Detaillierungsgrad dieses Lehrplans erlaubt international konsistentes Lehren und Prüfen. Um dieses Ziel zu erreichen, beinhaltet dieser Lehrplan Folgendes:

- Allgemeine Lernziele
- Inhalte mit einer Beschreibung der Lernziele
- Referenzen zu weiterführender Literatur (falls notwendig)

Geschlechtsspezifische Formulierungen

Wir haben in diesem Dokument bewusst auf geschlechtsspezifische Formulierungen verzichtet.

Selbstverständlich unterstützen wir als IREB geschlechtssensible Formulierungen. Wir sehen aber auch die Notwendigkeit, komplexe Sachverhalte so zu formulieren, dass sie leicht verstanden werden können.

Texte, die eigentlich eine männliche und weibliche Form fordern, wären weniger gut lesbar und damit schwieriger zu verstehen. Ziel dieses Dokuments ist es jedoch, Inhalte präzise und klar darzustellen und zu vermitteln. Da wir dem Leser helfen wollen, den Fokus auf den Inhalt zu richten, verwenden wir in diesem Dokument bewusst nur die männliche Form von Personen.

Dies soll nicht als Ausdruck von mangelndem Respekt verstanden werden.

Lernziele/Kognitive Stufen des Wissens

Allen Modulen und Lernzielen in diesem Lehrplan ist eine kognitive Stufe zugeordnet. Die folgenden Stufen werden verwendet:

- **K1: Kennen** (beschreiben, aufzählen, charakterisieren, erkennen, benennen, erinnern, ...) — Sich an zuvor gelernten Stoff erinnern oder ihn abrufen.
- **K2: Verstehen** (erklären, interpretieren, vervollständigen, zusammenfassen, begründen, klassifizieren, vergleichen, ...) — Bedeutung anhand von gegebenem Inhalt oder Situationen begreifen/herstellen.
- **K3: Anwenden** (spezifizieren, schreiben, entwerfen, entwickeln, implementieren, ...) — Wissen und Fähigkeiten in gegebenen Situationen anwenden.

Die höheren Stufen schließen die niedrigeren ein. Beachten Sie, dass alle Begriffe des Glossars, die als Grundbegriffe bezeichnet werden, bekannt sein müssen (K1), auch wenn sie in den Lernzielen nicht explizit erwähnt werden. Das Glossar steht auf der IREB Homepage zum Download zur Verfügung: https://www.ireb.org/downloads/#cpre-glossary

Lehrplanaufbau

Der Lehrplan besteht aus 7 Hauptkapiteln. Ein Kapitel umfasst eine Lerneinheit (LE). Die Titel der Hauptkapitel enthalten die kognitive Stufe ihres Kapitels, die die höchste Stufe ihrer Unterkapitel darstellt. Die Dauer gibt die Zeit an, welche in einer Schulung mindestens für dieses Kapitel aufgewendet werden sollte. Den Schulungsanbietern steht es frei, mehr Zeit zu investieren, sie sollten jedoch sicherstellen, dass das Verhältnis zwischen den LEs beibehalten wird. Wichtige Begriffe, die in einem Kapitel verwendet werden, sind am Anfang des Kapitels aufgeführt.

Beispiel

LE 4 Praktiken für die Erarbeitung von Anforderungen (K3)

Dauer: 4 Stunden 30 Minuten

Begriffe: Anforderungsquelle, Systemgrenze, Systemkontext, Anforderungsermittlung, Anforderungsvalidierung, Stakeholder, Kano-Modell, Konflikt, Konfliktlösung

Dieses Beispiel zeigt, dass Kapitel 4 Lernziele auf der Stufe K3 enthält und viereinhalb Stunden für das Lehren der Inhalte dieses Kapitel vorgesehen sind.

Jedes Kapitel enthält Unterkapitel. In deren Titel findet sich ebenfalls die kognitive Stufe der betroffenen Teilinhalte.

Vor dem eigentlichen Text sind die Lernziele (LZ) gelistet. Die Nummerierung zeigt die Zugehörigkeit zu Unterkapiteln an. Das Lernziel LZ 4.2.1 wird zum Beispiel im Unterkapitel 4.2 beschrieben.

Reihenfolge der Themen im Lehrplan

Die Reihenfolge der Kapitel in diesem Lehrplan stellt eine logische Reihenfolge der Themen dar. Sie ist nicht so aufzufassen, dass die Themen in genau dieser Reihenfolge zu unterrichten sind. Es steht den Schulungsanbietern frei, die Inhalte in jeder beliebigen Reihenfolge (einschließlich der Vermischung von Inhalten aus mehreren LEs) zu unterrichten, die sie im Rahmen ihrer Schulung für angemessen halten und die in ihr didaktisches Konzept passt.

Die Prüfung

Dieser Lehrplan ist die Grundlage für die Prüfung für das CPRE Foundation Level Zertifikat.

Eine Prüfungsfrage kann Stoff aus mehreren Kapiteln des Lehrplans abdecken. Alle Abschnitte (LE 1 bis LE 7) dieses Lehrplans können geprüft werden.

Das Format der Prüfung ist Multiple Choice.

Prüfungen können unmittelbar im Anschluss an einen Kurs aber auch unabhängig davon (z. B. in einem Prüfzentrum) abgelegt werden. Eine Liste der von IREB lizenzierten Zertifizierungsstellen finden Sie auf der Website https://www.ireb.org.

Versions-Historie

Version	Datum	Kommentar
1.0	2007	Erste Version
2.1	November 2010	Erstes großes Update
2.2	1. März 2015	Kleines Update (Inkonsistenzen beseitigt, Rechtschreib- und Grammatikfehler behoben)
3.0.0	1. Oktober 2020	Zweites großes Update, das den Stand des heutigen RE widerspiegelt und sowohl planbasierte als auch agile Ansätze zur Spezifizierung und Verwaltung von Anforderungen umfasst.
3.0.1	7. Oktober 2020	Rechtschreibfehler behoben und LE 4.3 umbenannt.
3.1.0	1. Sept. 2022	Tippfehler und Verweise im gesamten Dokument korrigiert, um die Lesbarkeit zu verbessern. LE 1 ▪ Lernziel LZ 1.3.2 nach LZ 1.1.2. verschoben ▪ Lernziele LZ 1.2.2 und LZ 1.3.1 aktualisiert LE 3 ▪ Lernziel LZ 3.1.3 angepasst ▪ LE 3.1.2 Abschnitt zu Abstraktionsebenen wurde überarbeitet ▪ LE 3.4: Alle Modelltypen, die auf Foundation Level nicht angewendet werden müssen, wurden in einen neuen Unterabschnitt 3.4.6 verschoben ▪ LE 3.6: Überschrift geändert in "Anforderungsdokumente und Dokumentations-strukturen". LE 4 ▪ LE 4.1: Die Beschreibung von Stakeholder und Stakeholder-Rollen wurde präzisiert. ▪ LE 4.2: Die Einleitung und die Begründung für Design- und Ideenfindungstechniken wurden aktualisiert, um präziser zu sein. ▪ LE 4.3: Überschrift geändert in "Auflösen von Konflikten bezüglich der Anforderungen" angepasst. LE 6 ▪ LZ 6.3.1 und LZ 6.5.2 wurden leicht modifiziert

Version	Datum	Kommentar
3.1.1	1. März 2023	Übersetzungsfehler in LE 4.2 behoben (statt Design- und Ideengenerierungstechniken in Entwurfs- und Ideenfindungstechniken umbenannt)
3.1.2	1. Januar 2024	Neues Corporate Design umgesetzt

Inhalt

1 Einführung und Überblick zum Requirements Engineering (K2)

Ziel: Wissen, worum es bei RE geht und den Wert des RE verstehen

Dauer: 1 Stunde

Begriffe: Anforderung, Anforderungsspezifikation, Requirements Engineering (RE), Stakeholder, System, Requirements Engineer

Lernziele

LZ 1.1.1 Sich an die grundlegende Terminologie erinnern (K1)
LZ 1.1.2 Die verschiedenen Arten von Anforderungen verstehen (K2)
LZ 1.2.1 Erklären des Wertes von RE (K2)
LZ 1.2.2 Aufzählung der Symptome für unzureichendes RE (K1)
LZ 1.3.1 Wissen, wo RE angewendet werden kann und wo Anforderungen auftreten (K1)
LZ 1.4.1 Die Hauptaufgaben des RE kennen und wissen, dass ein RE-Prozess auf die Durchführung dieser Aufgaben zugeschnitten sein muss (K1)
LZ 1.5.1 Charakterisieren der Rolle und Aufgaben eines Requirements Engineers (K1)
LZ 1.6.1 Sich daran erinnern, was ein Requirements Engineer lernen muss (K1)

1.1 Requirements Engineering: Was (K1)

Personen und Organisationen haben Wünsche und Bedürfnisse nach neuen Dingen, die erstellt, oder aus bestehenden Dingen weiterentwickelt werden. Wir nennen solche Bedürfnisse *Anforderungen*.

Dinge, die erstellt oder weiterentwickelt werden sollen, können sein:

- Dem Kunden zur Verfügung gestellte *Produkte*
- Dem Kunden zur Verfügung gestellte *Dienstleistungen*
- Alle anderen *Ergebnisse* (Deliverables)wie Geräte (Devices), Verfahren oder Werkzeuge, die Personen und Organisationen helfen, ein bestimmtes Ziel zu erreichen
- *Zusammensetzungen* oder *Bestandteile* von Produkten, Dienstleistungen oder anderen Ergebnissen

Alle diese Dinge können als *Systeme* betrachtet werden. In diesem Lehrplan verwenden wir den Begriff *System*, um alle Dinge zu bezeichnen, an die *Stakeholder* Anforderungen stellen. *Stakeholder* sind Personen oder Organisationen, die die Anforderungen an ein System beeinflussen oder die von diesem System betroffen sind.

Das Ziel von RE ist es, Anforderungen an Systeme so zu spezifizieren und zu verwalten, dass die implementierten und bereitgestellten Systeme die Wünsche und Bedürfnisse der Stakeholder erfüllen.

Im RE unterscheiden wir zwischen drei Arten von Anforderungen [Glin2020]:

- *Funktionale Anforderungen* betreffen ein Ergebnis oder Verhalten, das durch eine Funktion eines Systems bereitgestellt werden soll. Dazu gehören Anforderungen an Daten oder die Interaktion eines Systems mit seiner Umgebung.

- *Qualitätsanforderungen* beziehen sich auf Qualitätsaspekte, die nicht durch funktionale Anforderungen abgedeckt sind, wie z.B. Leistung (Performance), Verfügbarkeit, Sicherheit oder Zuverlässigkeit.
- *Constraints (Randbedingungen)* sind Anforderungen, die den Lösungsraum über das hinaus begrenzen, was zur Erfüllung der gegebenen funktionalen Anforderungen und Qualitätsanforderungen notwendig ist.

1.2 Requirements Engineering: Warum (K2)

Angemessenes RE bietet einen *Mehrwert* im Prozess der Entwicklung und Weiterentwicklung eines Systems:

- Verringerung des Risikos, ein falsches System zu entwickeln
- Besseres Verständnis des Problems
- Grundlage für die Schätzung von Entwicklungsaufwand und Kosten
- Voraussetzung für das Testen des Systems

Typische Symptome für mangelhaftes RE sind fehlende, unklare oder falsche Anforderungen. Dies ist insbesondere zurückzuführen auf:

- Direkt mit der Entwicklung des Systems zu beginnen
- Kommunikationsprobleme zwischen den beteiligten Parteien
- Die Annahme, dass die Anforderungen selbstverständlich sind
- Unzureichende Ausbildung und Fähigkeiten im RE

1.3 Requirements Engineering: Wo (K2)

RE kann auf Anforderungen an jede Art von System angewendet werden. Der dominierende Anwendungsfall für RE sind heutzutage jedoch Systeme, in denen Software eine wesentliche Rolle spielt. Solche Systeme bestehen in der Regel aus Softwarekomponenten, physischen Elementen und organisatorischen Elementen.

Anforderungen können auftreten als:

- *Systemanforderungen* – was ein System tun soll
- *Stakeholderanforderungen* – was Stakeholder aus ihrer Perspektive wollen
- *Benutzeranforderungen* – was Benutzer aus ihrer Perspektive wollen
- *Domänenanforderungen* – erforderliche Domäneneigenschaften
- *Geschäftsanforderungen* – Geschäftsziele, Zielsetzungen und Bedürfnisse einer Organisation

1.4 Requirements Engineering: Wie (K1)

Die Hauptaufgaben im RE sind die Ermittlung (4.2), Dokumentation (3), Validierung (4.4) und Verwaltung (6) von Anforderungen. Werkzeugunterstützung (7) kann bei der Erfüllung dieser Aufgaben helfen. Die Anforderungsanalyse und die Lösung von Anforderungskonflikten (4.3) werden als Teil der Ermittlung betrachtet. Um RE-Aktivitäten richtig durchführen zu können,

muss ein geeigneter RE-Prozess aus einem breiten Spektrum von Möglichkeiten maßgeschneidert werden (5).

1.5 Die Rolle und Aufgaben eines Requirements Engineers (K1)

Requirements Engineer ist typischerweise keine Berufsbezeichnung, sondern eine *Rolle*, die Personen einnehmen, die:

- Als Teil ihrer Aufgaben Anforderungen ermitteln, dokumentieren, validieren und/oder verwalten.
- Über fundierte Kenntnisse im RE verfügen.
- Die Lücke zwischen dem Problem und möglichen Lösungen überbrücken können.

In der Praxis agieren Business-Analysten, Anwendungsspezialisten, Product Owner, Systemingenieure und sogar Entwickler in der Rolle eines Requirements Engineers.

1.6 Was über Requirements Engineering zu lernen ist (K1)

Dieser Lehrplan deckt die grundlegenden Fähigkeiten ab, die ein Requirements Engineer erlernen muss. Er umfasst die grundlegenden Prinzipien des RE (2), wie Anforderungen in verschiedenen Formen zu dokumentieren sind (3), wie Anforderungen mit verschiedenen Praktiken zu erarbeiten sind (4), wie geeignete RE-Prozesse zu definieren sind und mit ihnen zu arbeiten ist (5), wie bestehende Anforderungen zu verwalten sind (6) und wie Werkzeugunterstützung eingesetzt wird (7).

2 Grundlegende Prinzipien des Requirements Engineering (K2)

Ziel: Die Prinzipien des RE kennen und verstehen

Dauer: 1 Stunde 30 Minuten

Begriffe: Kontext, Anforderung, Requirements Engineering (RE), Stakeholder, gemeinsames Verständnis, Validierung

Lernziele

LZ 2.1.1 Aufzählen der Prinzipien des RE (K1)
LZ 2.2.1 Sich an die mit den Prinzipien verbundenen Begriffe erinnern (K1)
LZ 2.2.2 Erklären der Prinzipien und der Gründe, warum sie wichtig sind (K2)

2.1 Überblick über die Grundsätze (K1)

RE unterliegt einer Reihe von grundlegenden Prinzipien, die für alle Aufgaben, Aktivitäten und Praktiken im RE gelten. Die folgenden neun Prinzipien bilden die Grundlage für die in den folgenden Kapiteln dieses Lehrplans vorgestellten Praktiken.

1. Wertorientierung: Anforderungen sind Mittel zum Zweck, kein Selbstzweck
2. Stakeholder: Im RE geht es darum, die Wünsche und Bedürfnisse der Stakeholder zu befriedigen
3. Gemeinsames Verständnis: Erfolgreiche Systementwicklung ist ohne eine gemeinsame Basis nicht möglich
4. Kontext: Systeme können nicht isoliert verstanden werden
5. Problem – Anforderung – Lösung: Ein unausweichlich ineinandergreifendes Tripel
6. Validierung: Nicht-validierte Anforderungen sind nutzlos
7. Evolution: Sich ändernde Anforderungen sind kein Unfall, sondern der Normalfall
8. Innovation: Mehr vom Gleichen ist nicht genug
9. Systematische und disziplinierte Arbeit: Wir können im RE nicht darauf verzichten

2.2 Die Erläuterung der Prinzipien (K2)

Prinzip 1 - Wertorientierung: Anforderungen sind ein Mittel zum Zweck, kein Selbstzweck

Der Wert einer Anforderung ist gleich ihrem Nutzen abzüglich der Kosten für das Ermitteln, Dokumentieren, Validieren und Verwalten der Anforderung. Der Nutzen einer Anforderung ist der Grad, den sie dazu beiträgt:

- Ein System zu bauen, das die Wünsche und Bedürfnisse der Stakeholder erfüllt.

- Das Risiko von Fehlschlägen und kostspieligen Nacharbeiten bei der Entwicklung des Systems zu verringern.

Prinzip 2 – Stakeholder: Im RE geht es darum, die Wünsche und Bedürfnisse der Stakeholder zu befriedigen

Da es beim RE darum geht, die Wünsche und Bedürfnisse der Stakeholder zu verstehen, ist der richtige Umgang mit Stakeholdern eine Kernaufgabe des RE. Jeder Stakeholder hat eine Rolle im Kontext des zu errichtenden Systems, z.B. Benutzer, Kunde, Auftraggeber, Betreiber oder Regulierungsbehörde. Ein Stakeholder kann auch mehrere Rollen gleichzeitig ausüben. Für Stakeholder-Rollen mit zu vielen Einzelpersonen oder wenn Einzelpersonen unbekannt sind, können fiktive archetypische Beschreibungen, sogenannte *Personas,* als Ersatz definiert werden. Es reicht nicht aus, nur die Anforderungen der Endbenutzer oder Kunden zu berücksichtigen. Dies würde bedeuten, dass wir kritische Anforderungen anderer Stakeholder übersehen könnten. Benutzer, die Feedback zu einem in Gebrauch befindlichen System geben, sollten ebenfalls als Stakeholder betrachtet werden.

Stakeholder können unterschiedliche Bedürfnisse und Standpunkte haben, was zu konträren Anforderungen führen kann. Es ist eine Aufgabe des RE, solche Konflikte zu identifizieren und zu lösen.

Die Einbeziehung der richtigen Personen in die entsprechenden Stakeholder-Rollen ist für erfolgreiches RE von entscheidender Bedeutung. Praktiken zur Identifizierung, Priorisierung und zur Zusammenarbeit mit Stakeholdern werden in 4 beschrieben.

Prinzip 3 – Gemeinsames Verständnis: Erfolgreiche Systementwicklung ist ohne eine gemeinsame Basis nicht möglich

RE schafft, fördert und sichert ein gemeinsames Verständnis zwischen und innerhalb der beteiligten Parteien: Stakeholder, Requirements Engineers und Entwickler. Es gibt zwei Formen des gemeinsamen Verständnisses:

- *Explizites gemeinsames Verständnis* (erreicht durch dokumentierte und vereinbarte Anforderungen)
- *Implizites gemeinsames Verständnis* (basierend auf gemeinsamem Wissen über Bedürfnisse, Visionen, Kontext usw.)

Domänenwissen, frühere erfolgreiche Zusammenarbeit, gemeinsame Kultur und Werte sowie gegenseitiges Vertrauen sind die Voraussetzungen für ein gemeinsames Verständnis, während geografische Distanz, Outsourcing oder große Teams mit hoher Fluktuation Hindernisse darstellen.

Bewährte Praktiken zur Erzielung eines gemeinsamen Verständnisses umfassen die Erstellung von Glossaren (3.5), das Erstellen von Prototypen (3.7) oder die Verwendung eines

bestehenden Systems als Bezugspunkt. Zu den Praktiken zur Bewertung des gemeinsamen Verständnisses gehören unter anderem Beispiele für erwartete Ergebnisse, die Untersuchung von Prototypen oder die Schätzung der Kosten für die Umsetzung einer Anforderung. Die wichtigste Praxis zur Verringerung der Auswirkungen von Missverständnissen ist die Anwendung eines Verfahrens mit kurzen Rückkopplungsschleifen (5).

Prinzip 4 – Kontext: Systeme können nicht isoliert verstanden werden

Systeme sind in einen *Kontext* eingebettet. Ohne diesen Kontext zu verstehen, ist es unmöglich, ein System richtig zu spezifizieren. Im RE ist der Kontext eines Systems der Teil der Systemumgebung, der für das Verständnis des Systems und seiner Anforderungen relevant ist. Die *Systemgrenze* ist die Grenze zwischen einem System und seinem umgebenden Kontext. Anfangs ist die Systemgrenze oft nicht klar, und sie kann sich im Laufe der Zeit sogar ändern.

Die Klärung der Systemgrenze und die Definition der externen Schnittstellen zwischen dem System und den Kontextelementen, mit denen es interagiert, sind echte RE-Aufgaben. Gleichzeitig muss der *Umfang* des Systems – d.h. die Reichweite der gestaltbaren Dinge bei der Entwicklung des Systems – bestimmt werden. Wir müssen auch die sogenannte *Kontextgrenze* berücksichtigen, die den RE-relevanten Teil der Umgebung eines Systems vom Rest der Welt trennt.

Bei der Spezifizierung eines Systems reicht es nicht aus, nur die Anforderungen innerhalb der Systemgrenze zu berücksichtigen. Im RE muss man ebenfalls Folgendes berücksichtigen:

- Änderungen im Kontext, die sich auf die Systemanforderungen auswirken können.
- Anforderungen der realen Welt, die für das System relevant sind (und wie sie den Systemanforderungen zugeordnet werden können).
- Annahmen über den Kontext, die für das Funktionieren des Systems und die Erfüllung der Anforderungen der realen Welt gelten müssen.

Prinzip 5 – Problem - Anforderung - Lösung: Ein unausweichlich ineinandergreifendes Tripel

Ein *Problem* tritt auf, wenn die Stakeholder mit der Situation, wie sie ist, nicht zufrieden sind. Die *Anforderungen* erfassen, was die Stakeholder brauchen, um das Problem zu lösen oder zu vereinfachen. Ein sozio-technisches System, das diesen Anforderungen gerecht wird, stellt eine *Lösung* dar.

Probleme, Anforderungen und Lösungen treten nicht unbedingt in dieser Reihenfolge auf. Lösungsideen können Nutzerbedürfnisse erzeugen, die als Anforderungen ausgearbeitet

und in eine konkrete Lösung umgesetzt werden müssen. Dies ist typischerweise bei Innovationen der Fall.

- Probleme, Anforderungen und Lösungen sind eng miteinander verflochten: Sie können nicht isoliert angegangen werden.
- Dennoch sind Requirements Engineers bestrebt, Probleme, Anforderungen und Lösungen beim Denken, Kommunizieren und Dokumentieren so weit wie möglich voneinander zu trennen. Diese Trennung der Belange erleichtert die Abwicklung der RE-Aufgaben.

Prinzip 6 – Validierung: Nicht-validierte Anforderungen sind nutzlos

Schließlich müssen wir überprüfen, ob das eingesetzte System die Wünsche und Bedürfnisse der Stakeholder erfüllt. Um das Risiko unzufriedener Stakeholder von Anfang an zu kontrollieren, muss die Validierung bereits im RE beginnen. Wir müssen prüfen, ob:

- Unter den Stakeholdern eine Einigung über die Anforderungen erreicht wurde,
- Die Wünsche und Bedürfnisse der Stakeholder durch die Anforderungen ausreichend abgedeckt werden,
- Die Kontextannahmen (siehe Prinzip 4 oben) vernünftig sind.

Praktiken für die Validierung von Anforderungen werden in 4.4 diskutiert.

Prinzip 7 – Evolution: Sich ändernde Anforderungen sind kein Unfall, sondern der Normalfall

Systeme und ihre Anforderungen unterliegen der *Evolution*. Das bedeutet, dass sie sich ständig ändern. Anfragen zur Änderung einer Anforderung oder einer Menge von Anforderungen an ein System können z.B. verursacht werden durch:

- Geänderte Geschäftsprozesse
- Wettbewerber, die neue Produkte oder Dienstleistungen einführen
- Kunden, die ihre Prioritäten oder Meinung ändern
- Veränderungen in der Technologie
- Änderungen von Gesetzen oder Vorschriften
- Feedback von Systembenutzern, die nach einer neuen oder geänderten Funktion fragen

Darüber hinaus können sich die Anforderungen aufgrund von Rückmeldungen der Stakeholder bei der Validierung von Anforderungen, durch die Entdeckung von Fehlern in zuvor erhobenen Anforderungen oder durch geänderte Bedürfnisse der Stakeholder ändern.

Folglich müssen Requirements Engineers zwei scheinbar widersprüchliche Ziele verfolgen:

- Zulassen, dass sich Anforderungen ändern
- Anforderungen stabil halten

Wie dies erreicht werden kann, wird in 6.7 beschrieben.

Prinzip 8 – Innovation: Mehr vom Gleichen ist nicht genug

Wenn man den Stakeholdern nur genau das gibt, was sie äußern, verpasst man die Gelegenheit, Systeme aufzubauen, die ihre Bedürfnisse besser erfüllen, als sie es erwarten. Gutes RE strebt nicht nur danach, die Stakeholder zufrieden zu stellen, sondern auch danach, dass sie glücklich und begeistert sind oder sich sicher fühlen. Darum geht es schließlich bei Innovation.

RE gestaltet innovative Systeme:

- Im Kleinen durch das Streben nach aufregenden neuen Funktionen und Benutzerfreundlichkeit.
- Im Großen durch das Streben nach verändernden (disruptiven) neuen Ideen.

In 4.2 werden verschiedene Techniken zur Förderung von Innovationen im RE erörtert.

Prinzip 9 – Systematische und disziplinierte Arbeit: Wir können im RE nicht darauf verzichten

Es ist notwendig, geeignete Prozesse und Praktiken zur systematischen Ermittlung, Dokumentation, Validierung und Verwaltung von Anforderungen anwenden, unabhängig vom tatsächlich genutzten Entwicklungsprozess. Selbst wenn ein System ad hoc entwickelt wird, verbessert ein systematischer und disziplinierter Ansatz für das RE die Qualität des daraus resultierenden Systems.

Es gibt nicht den einen Prozess oder das eine Verfahren im RE, welches in jeder gegebenen Situation oder zumindest in den meisten Situationen gut funktioniert. Systematisches und diszipliniertes Arbeiten bedeutet, dass Requirements Engineers:

- Die von ihnen angewendeten Prozesse und Praktiken an das jeweilige Problem, den Kontext und die Arbeitsumgebung anpassen.
- Nicht immer das gleiche Verfahren und die gleichen Praktiken verwenden.
- Prozesse und Praktiken aus früheren erfolgreichen RE-Tätigkeiten nicht unreflektiert wiederverwenden.

Für jedes RE-Vorhaben müssen Prozesse, Praktiken und Arbeitsprodukte gewählt werden, die der jeweiligen Situation am besten entsprechen. Die Einzelheiten werden in 3, 4, 5 und 6 ausgearbeitet.

3 Arbeitsprodukte und Dokumentationspraktiken (K3)

Ziel: Die grundlegende Rolle von Arbeitsprodukten im RE verstehen und in der Lage sein, Arbeitsprodukte erstellen können

Dauer: 7 Stunden

Begriffe: Arbeitsprodukt, natürlichsprachige Arbeitsprodukte, vorlagenbasierte Arbeitsprodukte, modellbasierte Arbeitsprodukte, Glossar, Qualitätskriterien, Anforderungsspezifikation

Lernziele

LZ 3.1.1 Merkmale von RE-Arbeitsprodukten kennen und häufig verwendete Arten von Arbeitsprodukten aufzählen können (K1)
LZ 3.1.2 Wissen, welche Arbeitsprodukte wofür verwendet werden können und ihre Lebensdauer kennen (K1)
LZ 3.1.3 Erklären der verschiedenen Abstraktionsebenen für Anforderungen, einschließlich der Wahl geeigneter Abstraktions- und Detaillierungsebenen (K2)
LZ 3.1.4 Aspekte kennen, die in den Arbeitsprodukten zu berücksichtigen sind, sowie die Zusammenhänge zwischen diesen Aspekten (K1)
LZ 3.1.5 Nennen von allgemeinen Dokumentationsrichtlinien (K1)
LZ 3.1.6 Beschreiben können, warum es sich lohnt, die zu verwendenden Arbeitsprodukte zu planen (K1)
LZ 3.2.1 Natürlichsprachige Arbeitsprodukte mit deren Vor- und Nachteilen kennen (K1)
LZ 3.2.2 Regeln für das Schreiben guter natürlichsprachiger Anforderungen erklären (K2)
LZ 3.3.1 Die Kategorien von vorlagenbasierten Arbeitsprodukten und ihre Vor- und Nachteile kennen (K1)
LZ 3.3.2 Spezifizieren einer individuellen Anforderung und einer User Story mit Hilfe einer Satzschablone (K3)
LZ 3.3.3 Einen Use Case mit Hilfe einer Formularvorlage spezifizieren (K3)
LZ 3.4.1 Die Rolle, den Zweck und die Verwendung von Modellen im RE verstehen (K2)
LZ 3.4.2 Die Vorteile und Grenzen der Modellierung im RE verstehen (K2)
LZ 3.4.3 Die Begriffe Modell, Modellierungssprache, Aktivitätsmodell, Aktivitätsdiagramm, Klassenmodell, Klassendiagramm, Kontextmodell, Kontextdiagramm, Domänenmodell, Zielmodell, Interaktionsmodell, Prozessmodell, Sequenzdiagramm, Statechart, Zustandsmaschine, Zustandsdiagramm, Use Case, Use Case Diagramm kennen (K1)
LZ 3.4.4 Verstehen, wie man einen geeigneten Modelltyp zur Spezifizierung von Anforderungen in einer bestimmten Situation auswählt (K2)
LZ 3.4.5 Verstehen und Interpretieren einfacher, gegebenenfalls in UML beschriebener Modelle der folgenden Typen: Kontextmodelle, Use Cases und Use Case Diagramme, Domänenmodelle, Klassenmodelle, Aktivitätsmodelle, Prozessmodelle und Statecharts (K2)
LZ 3.4.6 Ein einfaches Modell der Daten eines Systems oder der Objekte in einer Domäne mit einem UML-Klassendiagramm spezifizieren (K3)
LZ 3.4.7 Eine einfache Systemfunktion oder Geschäftsprozess mit Hilfe eines UML-Aktivitätsdiagramms spezifizieren (K3)
LZ 3.5.1 Den Zweck von Glossaren und wie man ein Glossar erstellt erklären (K2)
LZ 3.6.1 Häufig verwendete Anforderungsspezifikationsdokumente kennen (K1)
LZ 3.6.2 Erklären, welche Dokumentstrukturen welchem Zweck dienen und welche Kriterien für die Strukturierung gelten (K2)

LZ 3.7.1 Verschiedene Arten von Prototypen kennen und wissen, wozu sie dienen (K1)
LZ 3.8.1 Qualitätskriterien für einzelne Anforderungen kennen (K1)
LZ 3.8.2 Qualitätskriterien für Arbeitsprodukte kennen (K1)

3.1 Arbeitsprodukte im Requirements Engineering (K2)

Ein Arbeitsprodukt ist ein aufgezeichnetes, in einem Arbeitsprozess erzeugtes Zwischen- oder Endergebnis. Es gibt eine Vielzahl von Arbeitsprodukten im RE, diese reichen beispielsweise von kurzlebigen grafischen Skizzen über sich entwickelnde Sammlungen von User Storys bis hin zu formell freigegebenen vertraglichen Anforderungsspezifikationen mit Hunderten von Seiten.

3.1.1 Merkmale der Arbeitsprodukte (K1)

Arbeitsprodukte sind durch ihren Zweck, ihre Darstellung, ihren Umfang und ihre Lebensdauer charakterisiert. Die folgenden Arbeitsprodukte kommen in der Praxis für die angegebenen Zwecke häufig vor. Dabei ist zu beachten, dass ein Arbeitsprodukt andere Arbeitsprodukte enthalten kann.

- Arbeitsprodukte für eine einzelne Anforderung umfassen individuelle Anforderungen und User Storys.
- Arbeitsprodukte für eine kohärente Menge von Anforderungen umfassen Use Cases, grafische Modelle jeglicher Art (3.4), Aufgabenbeschreibungen, Beschreibungen externer Schnittstellen und Epics.
- Zu den Arbeitsprodukten, die umfassende Dokumente oder Dokumentationsstrukturen darstellen, gehören System-Anforderungsspezifikationen, Produkt- und Sprint Backlogs sowie Story Maps.
- Andere Arbeitsprodukte umfassen Glossare, Textnotizen, grafische Skizzen und Prototypen.

Arbeitsprodukte können in verschiedenen Formen *dargestellt* werden:

- Auf natürlicher Sprache basierend (3.2)
- Vorlagenbasiert (3.3)
- Modellbasiert (3.4)
- Andere Darstellungen, wie Zeichnungen oder Prototypen (3.7)

Die meisten Arbeitsprodukte werden elektronisch als Dateien, in Datenbanken oder in RE-Tools *gespeichert*. Informelle, kurzlebige Arbeitsprodukte können auch auf anderen Medien gespeichert werden, z.B. auf Papier oder als Post-It-Notizen auf einem Kanban-Board.

Bei der Betrachtung der Lebensdauer von Arbeitsprodukten unterscheiden wir zwischen drei Kategorien:

- *Kurzlebige Arbeitsprodukte:* Erstellt, um die Kommunikation zu unterstützen und ein gemeinsames Verständnis zu schaffen.
- *Sich weiterentwickelnde Arbeitsprodukte:* Entstehen in der Regel im Laufe der Zeit in mehreren Iterationen; benötigen einige Metadaten (6.5); Änderungskontrolle kann erforderlich sein.

- *Langlebige Arbeitsprodukte:* Wurden als *Baseline* erstellt oder *freigegeben*; benötigen einen vollständigen Satz an Metadaten (6.5); die Änderungskontrolle muss eingehalten werden (6.3, 6.4).

Ein kurzlebiges Arbeitsprodukt kann (durch Aufbewahrung und Hinzufügen von Metadaten) in ein sich weiterentwickelndes Arbeitsprodukt umgewandelt werden. Analog dazu kann ein temporäres oder sich weiterentwickelndes Arbeitsprodukt zu einem langlebigen Arbeitsprodukt werden, indem es als Baseline festgelegt oder freigegeben wird.

3.1.2 Abstraktionsebenen (K2)

Anforderungen existieren in der Regel auf vielen *verschiedenen Abstraktionsebenen*, von z.B. abstrakten Anforderungen an einen neuen Geschäftsprozess bis hin zu Anforderungen auf einer sehr detaillierten Ebene, wie z.B. die Reaktion einer Softwarekomponente auf ein außergewöhnliches Ereignis.

Die Wahl der richtigen Abstraktionsebene hängt von dem zu spezifizierenden Gegenstand und dem Zweck der Spezifikation ab. Es ist jedoch wichtig, Anforderungen, die auf verschiedenen Abstraktionsebenen existieren, nicht zu vermischen. In kleinen und mittelgroßen Arbeitsprodukten sollten die Anforderungen mehr oder weniger auf derselben Abstraktionsebene liegen.

In großen Arbeitsprodukten wie einer System-Anforderungsspezifikation sollten Anforderungen auf unterschiedlichen Abstraktionsebenen getrennt gehalten werden, indem das Spezifikationsdokument entsprechend strukturiert wird (3.6). Eine Anforderung auf einer hohen Abstraktionsebene kann in mehrere detaillierte Anforderungen auf niedrigeren, konkreteren Ebenen verfeinert werden.

Wenn übergeordnete Geschäfts- oder Stakeholder-Anforderungen in langlebigen Arbeitsprodukten beschrieben werden – wie z. B. Spezifikationen für Geschäftsanforderungen, Spezifikationen für Stakeholderanforderungen oder Visions-Dokumente – gehen sie der Spezifikation der Systemanforderungen voraus. In anderen Bereichen können sich Geschäfts-, Stakeholder- und Systemanforderungen auch gemeinsam entwickeln.

3.1.3 Detaillierungsgrad (K2)

Der *Detaillierungsgrad*, in dem die Anforderungen spezifiziert werden sollten, hängt insbesondere von folgenden Faktoren ab:

- Problem und der Entwicklungskontext
- Grad des gemeinsamen Verständnisses des Problems
- Freiheitsgrad, der den Designern und Programmierern überlassen wird
- Verfügbarkeit von schnellem Stakeholder-Feedback während der Konzeption und Implementierung
- Kosten vs. Nutzen einer detaillierten Spezifikation
- Auferlegte Standards und regulatorische Einschränkungen

Je detaillierter die Anforderungen spezifiziert sind, desto geringer ist das Risiko, dass am Ende etwas Unerwartetes oder nicht Spezifiziertes herauskommt. Die Kosten für die Spezifikation steigen jedoch mit zunehmender Detaillierung.

3.1.4 In Arbeitsprodukten zu berücksichtigende Aspekte (K1)

Bei der Spezifikation von Anforderungen in Arbeitsprodukten müssen verschiedene Aspekte berücksichtigt werden.

1. Die Anforderungen werden nach ihrer Art (1.1) eingeteilt in:
   a) Funktionale Anforderungen
   b) Qualitätsanforderungen
   c) Randbedingungen
2. Funktionale Anforderungen konzentrieren sich auf verschiedene Aspekte der Funktionalität eines Systems:
   a) Struktur und Daten
   b) Funktion und Ablauf
3. Zustand und Verhalten
   a) Schließlich können Anforderungen nur im Kontext (Prinzip 4 in 2) verstanden werden:
   b) Systemkontext, einschließlich externer Akteure
   c) *Systemgrenze* und externe Schnittstellen

Zwischen den genannten Aspekten gibt es viele Wechselbeziehungen und Abhängigkeiten. Beispielsweise kann eine von einem Benutzer (Kontext) ausgehende Anforderung einen Zustandsübergang (Zustand und Verhalten) auslösen, der eine Aktion auslöst, gefolgt von einer weiteren Aktion (Funktion und Ablauf), die Daten (Struktur und Daten) benötigt, um dem Benutzer (Kontext) innerhalb eines bestimmten Zeitintervalls (Qualität) ein Ergebnis zu liefern.

Einige Arbeitsprodukte konzentrieren sich auf einen bestimmten Aspekt und abstrahieren von den anderen Aspekten. Dies gilt insbesondere für Anforderungsmodelle (3.4). Andere Arbeitsprodukte, wie z.B. eine System-Anforderungsspezifikation, decken all diese Aspekte ab. Wenn verschiedene Aspekte in getrennten Arbeitsprodukten oder in getrennten Kapiteln desselben Arbeitsprodukts dokumentiert sind, müssen diese Arbeitsprodukte oder Kapitel miteinander konsistent gehalten werden.

3.1.5 Allgemeine Richtlinien für die Dokumentation (K1)

Unabhängig von den verwendeten Techniken gelten bei der Erstellung von Arbeitsprodukten die folgenden Richtlinien:

- Wählen Sie einen Typ von Arbeitsprodukten aus, der den *beabsichtigten Zweck* erfüllt.
- *Vermeiden Sie Redundanz*, indem Sie auf Inhalte verweisen, anstatt denselben Inhalt noch einmal zu wiederholen.
- *Stellen Sie sicher, dass es keine Inkonsistenzen* zwischen den Arbeitsprodukten gibt, insbesondere wenn sie verschiedene Aspekte abdecken.

- *Verwenden Sie Begriffe konsistent*, so wie im Glossar definiert.
- *Strukturieren* Sie Arbeitsprodukte angemessen.

3.1.6 Planen Sie die zu verwendenden Arbeitsprodukte (K1)

Jeder Projektrahmen und jede Domäne ist anders, sodass für jedes Vorhaben die zu entwickelnden Arbeitsprodukte ausgewählt werden müssen. Folgende Punkte müssen folglich vereinbart werden:

- In welchen Arbeitsprodukten sollen die Anforderungen erfasst werden und zu welchem Zweck?
- Welche Abstraktionsebenen sind zu berücksichtigen?
- Bis zu welchem Detaillierungsgrad müssen Anforderungen auf jeder Abstraktionsebene dokumentiert werden?
- Wie sollen die Anforderungen in diesen Arbeitsprodukten dargestellt werden?

Die zu verwendenden Arbeitsprodukte sollten zu einem frühen Zeitpunkt im Projekt festgelegt werden. Dies hat mehrere Vorteile:

- Es hilft bei der Planung von Aufwand und Ressourcen.
- Es stellt sicher, dass geeignete Notationen verwendet werden.
- Es stellt sicher, dass alle Ergebnisse in den richtigen Arbeitsprodukten erfasst werden.
- Es stellt sicher, dass keine größere Umstrukturierung von Informationen und keine „Schlussredaktion" erforderlich ist.
- Es hilft, Redundanzen zu vermeiden, was zu weniger Arbeit und einfacherer Wartbarkeit führt.

3.2 Natürlichsprachige Arbeitsprodukte (K2)

Seit den Anfängen des systematischen RE sind natürlichsprachige Anforderungen ein zentrales Mittel zur Spezifikation von Anforderungen in der Praxis.

Natürlichsprachige Arbeitsprodukte haben große Vorteile:

- Uneingeschränkte natürliche Sprache ist extrem ausdrucksstark und flexibel.
- Fast jede denkbare Anforderung kann in all ihren Aspekten in natürlicher Sprache ausgedrückt werden.
- Die natürliche Sprache wird im täglichen Leben verwendet und in der Schule gelehrt, sodass für das Lesen und Verstehen von Texten in natürlicher Sprache keine spezielle Ausbildung erforderlich ist.

Diese Vorteile gehen zu Lasten der Tatsache, dass in natürlicher Sprache verfasste Texte häufig unterschiedlich interpretiert werden können, was bei der Spezifikation von Anforderungen ein Problem darstellt. Darüber hinaus ist die Aufdeckung von Mehrdeutigkeiten, Auslassungen und Unstimmigkeiten in solchen Texten schwierig und kostspielig.

Das Schreiben guter natürlichsprachiger Anforderungen kann unterstützt werden durch:

- Das Schreiben kurzer und gut strukturierter Sätze.
- Das Definieren und die konsequente Verwendung einer einheitlichen Terminologie (3.5).
- Das Vermeiden ungenauer oder mehrdeutiger Begriffe und Phrasen.
- Das Kennen nachfolgend aufgeführter Fallstricke des technischen Schreibens.

Beim Schreiben von technischen Dokumenten in natürlicher Sprache gibt es einige bekannte Fallstricke, die vermieden oder mit Vorsicht verwendet werden sollten [GoRu2003].

Dinge, die es zu vermeiden gilt:

- Unvollständige Beschreibungen
- Unspezifische Substantive
- Unvollständige Bedingungen
- Unvollständige Vergleiche

Dinge, die mit Vorsicht zu verwenden sind:

- Passive Formulierung
- Universalquantoren (wie „alle" oder „nie")
- Nominalisierungen (d.h. von einem Verb abgeleitete Substantive, z.B. „Authentifizierung")

3.3 Vorlagenbasierte Arbeitsprodukte (K3)

Vorlagenbasierte Arbeitsprodukte werden verwendet, um einige der Unzulänglichkeiten natürlichsprachiger Arbeitsprodukte zu überwinden, indem vordefinierte Strukturen bereitgestellt werden.

- *Satzschablonen* bieten eine vordefinierte syntaktische Struktur für einen Satz, der eine Anforderung ausdrückt, insbesondere eine individuelle Anforderung oder eine User Story.
- *Formularvorlagen* stellen eine Reihe vordefinierter Felder in einem Formular zur Verfügung, die ausgefüllt werden können, z.B. zum Schreiben eines Use Cases oder einer messbaren Qualitätsanforderung.
- *Dokumentvorlagen* bieten eine vordefinierte Struktur für ein Anforderungsdokument.

In der Literatur wurden verschiedene Vorlagen beschrieben. [ISO29148], [MWHN2009] und [Rupp2014] stellen Satzschablonen für individuelle Anforderungen zur Verfügung. [Cohn2004] definiert eine weit verbreitete Satzschablone für User Storys und [Cock2001] beschreibt Formularvorlagen für Use Cases. [Laue2002] hat eine Vorlage für Aufgabenbeschreibungen vorgeschlagen. [ISO29148] und [RoRo2012] stellen Dokumentvorlagen für ganze Spezifikationen zur Verfügung. Darüber hinaus kann ein Kunde die Verwendung von kundenspezifischen Vorlagen in einem Projekt vorschreiben.

Vorteile von vorlagenbasierten Arbeitsprodukten:

- Schaffen einer klaren, wiederverwendbaren Struktur

- Hilfe bei der Erfassung der wichtigsten Informationen
- Anforderungen und Anforderungsspezifikationen einheitlich aussehen lassen
- Verbesserung der Gesamtqualität von Anforderungen und Anforderungsspezifikationen

Nachteile und Tücken von vorlagenbasierten Arbeitsprodukten:

- Es wird oft mehr Aufmerksamkeit auf das formale Ausfüllen der Vorlage als auf den Inhalt gelegt.
- Aspekte, die nicht in der Vorlage enthalten sind, werden eher weggelassen.

3.4 Modellbasierte Arbeitsprodukte (K3)

In natürlicher Sprache dargestellte Anforderungen haben Grenzen [Davi1993], insbesondere im Hinblick auf den Überblick über eine Menge von Anforderungen und das Verständnis der Beziehungen zwischen Anforderungen. Die Modellierung von Anforderungen befasst sich mit diesen Einschränkungen.

3.4.1 Die Rolle von Modellen im Requirements Engineering (K2)

Ein *Modell* ist eine abstrakte Darstellung eines bestehenden oder zu schaffenden Teils der Realität. Der Begriff Realität umfasst jede denkbare Menge von Elementen, Erscheinungsformen oder Konzepten, einschließlich anderer Modelle. In Bezug auf ein Modell wird der modellierte Teil der Realität als das *Original* bezeichnet. Beispiele für Modelle außerhalb der Domäne der Softwareentwicklung sind Gebäudeinformationsmodelle (Building Information Models – BIM) [ISO19650], die Elemente modellieren, die für die Planung, den Bau und die Verwaltung von Gebäuden und anderen Bauelementen erforderlich sind.

Im RE helfen Modelle, die Beziehungen und Zusammenhänge zwischen Anforderungen zu verstehen und den Überblick über eine Menge von Anforderungen zu behalten. Dies wird in erster Linie dadurch erreicht, dass man sich auf einzelne Aspekte — z.B. das Verhalten — konzentriert, während man von allen anderen Aspekten abstrahiert. Die Verwendung einer grafischen Notation für ein Modell hilft auch dabei, einen Überblick zu gewinnen. Modelle können aber auch auf nicht-grafische Weise dargestellt werden, zum Beispiel mit Tabellen.

Anforderungsmodelle haben Vorteile im Vergleich zu den in natürlicher Sprache dargestellten Anforderungen:

- Beziehungen und Zusammenhänge zwischen Anforderungen sind mit grafischen Modellen leichter zu verstehen als in natürlicher Sprache.
- Die Konzentration auf einen einzigen Aspekt reduziert die damit verbundene kognitive Anstrengung zum Verstehen der modellierten Anforderungen.
- Modellierungssprachen für Anforderungen haben eine eingeschränkte Syntax, die mögliche Mehrdeutigkeiten und Auslassungen reduziert.

Modelle haben auch ihre Grenzen:

- Modelle, die sich auf einzelne Aspekte konzentrieren, konsistent zu halten, ist eine Herausforderung.
- Informationen aus verschiedenen Modellen müssen zu einem besseren kausalen Verständnis integriert werden.
- Modelle fokussieren hauptsächlich auf funktionale Anforderungen. Die meisten Qualitätsanforderungen und Randbedingungen können nicht mit vertretbarem Aufwand in Modellen ausgedrückt werden.
- Die eingeschränkte Syntax einer grafischen Modellierungssprache impliziert, dass nicht jede relevante Information in einem Modell ausgedrückt werden kann.

Daher werden häufig Anforderungsmodelle und natürlichsprachige Anforderungen kombiniert.

Im RE können Modelle verwendet werden zum:

- *Spezifizieren* von (primär funktionalen) Anforderungen, um textuell erfasste Anforderungen teilweise oder sogar vollständig zu ersetzen.
- *Zerlegen* einer komplexen Realität in klar definierte und sich ergänzende Aspekte; jeder Aspekt wird durch ein spezifisches Modell dargestellt.
- *Umschreiben* von textuell beschriebenen Anforderungen, um ihre Verständlichkeit zu verbessern, insbesondere im Hinblick auf die Beziehungen zwischen ihnen.
- *Validieren* von textuell beschriebenen Anforderungen mit dem Ziel, Auslassungen, Mehrdeutigkeiten und Inkonsistenzen aufzudecken.

Modellierungssprachen werden verwendet, um Modelle auszudrücken. Einige Modellierungssprachen, zum Beispiel UML [OMG2017] oder BPMN [OMG2013], wurden standardisiert. Bei der Spezifikation von Anforderungen in einer nicht standardisierten Modellierungssprache ist eine Legende erforderlich, welche die Syntax und Semantik der verwendeten Modellierungssprache erklärt.

Es gibt viele Modelltypen, die im RE verwendet werden können. Ein Requirements Engineer muss verstehen, welcher Modelltyp am besten geeignet ist, die Anforderungen in einer gegebenen Situation zu spezifizieren.

In einer frühen Phase beginnt der Requirements Engineer oft mit der Modellierung des Kontextes (3.4.2) oder der Ziele des geplanten Systems.

3.4.2 Kontextmodellierung (K2)

Modelle, die sich auf den Kontextaspekt konzentrieren, spezifizieren die strukturelle Einbettung eines Systems in seine Umgebung und die Interaktion zwischen einem System und den Akteuren im Systemkontext.

Kontextmodelle spezifizieren ein System und die Akteure im Systemkontext, die mit dem System interagieren. Ein Kontextmodell skizziert auch die Schnittstellen zwischen einem System und seinem Kontext (z.B. im Hinblick darauf, welche Informationen ausgetauscht werden).

Kontextdiagramme werden als grafische Modellierungssprache zum Ausdruck von Kontextmodellen verwendet. Es gibt keine standardisierte Notation für Kontextdiagramme. Kontextdiagramme aus der strukturierten Analyse [DeMa1978] oder maßgeschneiderte Box-and-Line-Diagramme [Glin2019] können verwendet werden, um Kontextmodelle darzustellen.

Die Modellierungssprache UML [OMG2017] bietet mit Hilfe von *Use Case Diagrammen* die Möglichkeit, ein System und seinen Kontext hinsichtlich der Use Cases des Systems und der Akteure im Systemkontext zu modellieren, die über diese Use Cases mit dem System interagieren.

Use Cases modellieren die dynamische Interaktion zwischen einem Akteur im Systemkontext und einem System aus der Perspektive des Akteurs. Use Cases werden meist mit Formularvorlagen in natürlicher Sprache (3.3) oder unter Verwendung von UML-Aktivitätsdiagrammen (3.4.4) beschrieben.

3.4.3 Modellierung von Struktur und Daten (K3)

Modelle, die sich auf Struktur- und Datenaspekte konzentrieren, spezifizieren Anforderungen an die statischen Struktureigenschaften eines Systems oder einer Domäne.

Statische Domänenmodelle spezifizieren die (Geschäfts-)Objekte und ihre Beziehungen in einem Interessenbereich. Sie können mit UML-Klassendiagrammen [OMG2017] ausgedrückt werden.

Klassenmodelle spezifizieren in erster Linie die Klassen eines Systems und ihre Attribute und Beziehungen. Klassen repräsentieren materielle und immaterielle Einheiten in der realen Welt, die das System zur Erfüllung seiner Aufgaben kennen muss. UML-Klassendiagramme werden in der Regel als Modellierungssprache für Klassenmodelle verwendet.

3.4.4 Modellierung von Funktion und Ablauf (K3)

Modelle, die sich auf Funktions- und Ablaufaspekte konzentrieren, spezifizieren Anforderungen an die Abfolge von Aktionen, die erforderlich sind, um aus gegebenen Eingaben die gewünschten Ergebnisse zu erzielen, oder an die zur Ausführung eines (Geschäfts-)Prozesses erforderlichen Aktionen, einschließlich des Kontroll- und Datenflusses zwischen den Aktionen und stellen dar, wer für welche Aktion verantwortlich ist.

Aktivitätsmodelle werden verwendet, um Systemfunktionen zu spezifizieren. In der Modellierungssprache UML [OMG2017] werden *Aktivitätsdiagramme* verwendet, um Aktivitätsmodelle auszudrücken. Sie stellen Elemente für die Modellierung von Aktionen und den Kontrollfluss zwischen Aktionen zur Verfügung. Aktivitätsdiagramme können auch ausdrücken, wer für welche Aktion verantwortlich ist. Fortgeschrittene Modellierungselemente (nicht im CPRE Foundation Level abgedeckt) bieten Mittel zur Modellierung des Datenflusses.

Prozessmodelle werden zur Beschreibung von Geschäftsprozessen oder technischen Prozessen verwendet. Sie können mit UML-Aktivitätsdiagrammen oder mit spezifischen Sprachen zur Prozessmodellierung, wie BPMN [OMG2013], ausgedrückt werden. Im CPRE

Foundation Level verwenden wir für die Prozessmodellierung ausschließlich UML-Aktivitätsdiagramme.

3.4.5 Zustands- und Verhaltensmodellierung (K2)

Modelle, die sich auf Zustand und Verhalten konzentrieren, spezifizieren Anforderungen an das Verhalten eines Systems oder einen Teil einer Domäne im Hinblick auf zustandsabhängige Reaktionen auf Ereignisse oder die Dynamik der Komponenten-Interaktion.

Zustandsmaschinen modellieren Ereignisse, die den Übergang von einem Zustand in einen anderen auslösen, sowie die Aktionen, die bei einem Zustandsübergang ausgeführt werden müssen. *Statecharts* [Hare1988] sind Zustandsmaschinen mit Zuständen, die hierarchisch und/oder orthogonal zerlegt sind. Zustandsmaschinen, einschließlich Statecharts, können in der Modellierungssprache UML [OMG2017] mit *Zustandsdiagrammen* dargestellt werden.

3.4.6 Weitere Modelltypen im Requirements Engneering (L1)

Im CPRE Foundation Level ist das Verständnis und die Anwendung von Modellen auf ausgewählte, wichtige Modelltypen beschränkt. Es gibt weitere Modelltypen, die im Requirements Engineering verwendet werden. Für den CPRE Foundation Level reicht es aus, sie zu kennen und zu wissen, wofür sie verwendet werden.

Ziel-Modelle stellen eine Reihe von Zielen, Unterzielen und die Beziehungen zwischen ihnen dar. Zielmodelle können auch Aufgaben und Ressourcen enthalten, die zur Erreichung eines Ziels erforderlich sind, sowie Akteure, die ein Ziel erreichen wollen, und Hindernisse, die die Erreichung eines Ziels behindern.

In SysML [OMG2019] können *Blockdefinitionsdiagramme* (Block Definition Diagrams) angepasst werden, um Kontextdiagramme abzubilden, indem stereotypisierte Blöcke für das System und die Akteure verwendet werden. Blockdefinitionsdiagramme können auch verwendet werden, um die Struktur eines Systems in Bezug auf die konzeptionellen Entitäten und der Beziehungen zwischen ihnen zu modellieren.

Domain-Story-Modelle können zur Modellierung von Funktionen und Abläufen verwendet werden, indem visuelle Geschichten darüber spezifiziert werden, wie Akteure mit Geräten, Artefakten und anderen Elementen in einer Domäne interagieren, typischerweise unter Verwendung domänenspezifischer Symbole [HoSch2020]. Sie dienen zum Verständnis des Anwendungsbereichs, in dem das System betrieben wird.

Interaktionsmodelle modellieren die dynamischen Interaktionen zwischen Objekten oder Akteuren. UML-*Sequenzdiagramme* sind ein beliebtes Mittel, um die Interaktion zwischen Objekten zu spezifizieren.

3.5 Glossare (K2)

Bei jedem RE-Vorhaben, an dem mehr als eine Person beteiligt ist, besteht die Gefahr, dass es an einem gemeinsamen Verständnis der Terminologie mangelt, d.h. dass manche

Personen dieselben Begriffe unterschiedlich interpretieren. Um dieses Problem zu entschärfen, wird das gemeinsame Verständnis der relevanten Begriffe in einem Glossar festgehalten.

Ein *Glossar* ist eine zentrale Sammlung von Definitionen für: Kontextspezifische Begriffe, Alltagsbegriffe mit besonderer Bedeutung im gegebenen Kontext, Abkürzungen und Akronyme. Synonyme (verschiedene Begriffe, die die gleiche Sache bezeichnen) sollten als solche gekennzeichnet werden. Homonyme (Verwendung desselben Begriffs für verschiedene Dinge) sollten vermieden oder als solche gekennzeichnet werden.

Die folgenden Regeln gelten für Glossare:

- Verwalten Sie das Glossar zentral.
- Pflegen Sie das Glossar über den gesamten Verlauf der Systementwicklung.
- Definieren Sie eine Person oder eine kleine Gruppe, die für das Glossar verantwortlich ist.
- Verwenden Sie einen einheitlichen Stil und eine einheitliche Struktur für das Glossar.
- Beziehen Sie die Stakeholder ein und streben Sie eine Einigung über die Terminologie an.
- Machen Sie das Glossar für alle Beteiligten zugänglich.
- Machen Sie die Verwendung des Glossars zur Pflicht.
- Überprüfen Sie Arbeitsprodukte auf die korrekte Verwendung des Glossars.

3.6 Anforderungsdokumente und Dokumentationsstrukturen (K2)

Anforderungsspezifikationen umfassen mehrere RE-Arbeitsprodukte (3.1.1). Daher ist es wichtig, solche Dokumente mit einer klar definierten Struktur zu organisieren, um eine konsistente und verwaltbare Sammlung von Anforderungen zu schaffen. Neben den Anforderungen kann ein Anforderungsdokument auch zusätzliche Informationen und Erklärungen enthalten, z.B. ein Glossar, Abnahmebedingungen, Projektinformationen oder Merkmale der technischen Umsetzung.

Anforderungen können auch in anderen Dokumentationsstrukturen als den klassischen Dokumenten organisiert sein.

Häufig verwendete Dokumente sind:

- Stakeholder-Anforderungsspezifikation
- Benutzer-Anforderungsspezifikation (eine Teilmenge einer Stakeholder-Anforderungsspezifikation, die nur die Anforderungen von Benutzern abdeckt)
- System-Anforderungsspezifikation
- Geschäfts-Anforderungsspezifikation
- Visions-Dokument

Häufig verwendete alternative Dokumentationsstrukturen sind:

- Produkt-Backlog
- Sprint-Backlog
- Story Map

Sowohl die Auswahl einer Dokumentationsstruktur als auch die interne Organisation der gewählten Struktur hängen von unterschiedlichen Faktoren ab:

- Gewählter Entwicklungsprozess (5)
- Entwicklungsart und Domäne
- Dem Vertrag (ein Kunde kann die Verwendung einer bestimmten Dokumentationsstruktur vorschreiben)
- Größe des Dokuments

Dokumentvorlagen können bei der Strukturierung einer Anforderungsspezifikation helfen. Vorlagen sind in der Literatur [Vole2020], [RoRo2012] und in Normen [ISO29148] verfügbar. Vorlagen können auch aus früheren, ähnlichen Projekten, wiederverwendet werden oder von einem Kunden auferlegt werden. Eine Organisation kann auch beschließen, eine Vorlage als internen Standard zu erstellen.

3.7 Prototypen im Requirements Engineering (K1)

Im RE sind *Prototypen* ein Mittel zur Spezifikation von Anforderungen durch Beispiele und zur Validierung von Anforderungen. Insbesondere, wenn die beteiligten Stakeholder keine natürlichsprachige, vorlagen- oder modellbasierten Arbeitsprodukte schreiben und überprüfen wollen, können Prototypen verwendet werden.

Explorative Prototypen [LiSZ1994] werden verwendet, um ein gemeinsames Verständnis zu schaffen, Anforderungen zu klären oder Anforderungen auf verschiedenen Ebenen der Detaillierung (fidelity) zu validieren. Sie werden nach Gebrauch verworfen.

- *Wireframes* sind mit einfachen Materialien oder Skizzier-Werkzeugen gebaute Prototypen mit geringer Genauigkeit (low-fidelity), die in erster Linie zur Diskussion und Validierung von Designideen und Konzepten zur Benutzeroberfläche dienen.
- *Mock-Ups* sind Prototypen mittlerer Genauigkeit (medium-fidelity). Bei der Spezifikation digitaler Systeme verwenden sie reale Bildschirme und Klickfolgen (click flows), aber ohne echte Funktionalität. Sie dienen in erster Linie der Spezifikation und Validierung von Benutzerschnittstellen.
- *Native Prototypen* sind high-fidelity Prototypen, die kritische Teile eines Systems so weit implementieren, dass die Stakeholder anhand des Prototyps erkennen können, ob der prototypische Teil des Systems sich wie erwartet verhält und funktioniert.

Abhängig vom Grad der Genauigkeit können explorative Prototypen ein teures Arbeitsprodukt sein, sodass immer ein Kompromiss zwischen Kosten und erzieltem Nutzen besteht.

Evolutionäre Prototypen [LiSZ1994] sind Pilotsysteme, die den Kern eines zu entwickelnden Systems bilden. Das endgültige System entsteht durch die schrittweise Erweiterung und Verbesserung des Pilotsystems in mehreren Iterationen.

3.8 Qualitätskriterien für Arbeitsprodukte und Anforderungen (K1)

Eine Anforderung muss bestimmte Qualitätskriterien erfüllen, um als gute Anforderung zu gelten. Im modernen RE, mit wertorientierten Ansätzen (Prinzip 1 in 2), muss der Erfüllungsgrad eines Qualitätskriteriums mit dem durch die Anforderung geschaffenen Wert in Einklang stehen. Das heißt, dass Anforderungen nicht alle Qualitätskriterien vollständig erfüllen müssen – aber je höher der Wert einer Anforderung ist, desto wichtiger sind die Qualitätskriterien, um das Risiko eines Misserfolgs zu reduzieren.

Adäquatheit und Verständlichkeit sind die wichtigsten Qualitätskriterien für einzelne Anforderungen. Ohne sie ist eine Anforderung nutzlos oder sogar schädlich, unabhängig von der Erfüllung aller anderen Kriterien.

Qualitätskriterien für *einzelne Anforderungen*:

- Adäquat (beschreibt echte und abgestimmte Bedürfnisse der Stakeholder)
- Notwendig
- Eindeutig
- Vollständig (in sich abgeschlossen)
- Verständlich
- Prüfbar

Wie in 3.1.1 beschrieben, werden Anforderungen in der Regel in verschiedenen Arbeitsprodukten festgehalten, die einzelne oder mehrere Anforderungen abdecken. Die oben genannten Qualitätskriterien dienen dazu, qualitativ hochwertige Einzelanforderungen in einem Arbeitsprodukt zu erstellen. Bei Arbeitsprodukten, die mehr als eine Anforderung abdecken, sind zusätzlich die folgenden Qualitätskriterien zu berücksichtigen.

Qualitätskriterien für Arbeitsprodukte, die *mehrere Anforderungen* abdecken:

- Konsistent
- Nicht redundant
- Vollständig (keine bekannten und relevanten Anforderungen versäumt)
- Modifizierbar
- Verfolgbar
- Konform

4 Praktiken für die Erarbeitung von Anforderungen (K3)

Ziel: Verstehen der Anwendung von Praktiken zur Identifizierung von Anforderungsquellen, zur Ermittlung von Anforderungen, zur Identifizierung und Lösung von Konflikten und zur Validierung von Anforderungen.

Dauer: 4 Stunden 30 Minuten

Begriffe: Anforderungsquelle, Systemgrenze, Systemkontext, Anforderungsermittlung, Anforderungsverhandlung, Anforderungsvalidierung, Stakeholder, Kano-Modell, Konfliktlösung

Lernziele

LZ 4.1.1 Die Grenzen des Systems festlegen um auf die relevanten Anforderungen zu fokussieren (K3)
LZ 4.1.2 Sich an die relevanten Quellen für das zu erstellende System erinnern (K1)
LZ 4.1.3 Identifizieren der Stakeholder und schreiben einer Stakeholderliste (K3)
LZ 4.1.4 Die Vorteile eines Stakeholder-Managements verstehen (K2)
LZ 4.2.1 Verstehen, wie das Kano-Modell helfen kann, die richtigen Anforderungen zu ermitteln (K2)
LZ 4.2.2 Den Unterschied zwischen Erhebungs-, Entwurfs- und Ideenfindungstechniken verstehen (K2)
LZ 4.2.3 Verstehen, wie eine geeignete Ermittlungstechnik für eine bestimmte Situation gewählt wird (K2)
LZ 4.3.1 Sich an die verschiedenen Arten von Konflikten erinnern (K1)
LZ 4.3.2 Verstehen, welche Aktivitäten notwendig sind, um Konflikte zu lösen (K2)
LZ 4.3.3 Verstehen, wie geeignete Konfliktlösungstechniken anwendet werden (K2)
LZ 4.4.1 Verstehen, warum Anforderungsdokumente validiert werden müssen (K2)
LZ 4.4.2 Sich an die vier wichtigen Aspekte für die Anforderungsvalidierung erinnern (K1)
LZ 4.4.3 Verstehen, wie geeignete Techniken zur Anforderungsvalidierung angewendet werden (K2)

4.1 Quellen für Anforderungen (K3)

Die Qualität und Vollständigkeit von Anforderungen hängen erheblich von den einbezogenen Anforderungsquellen ab. Eine relevante Quelle außer Acht zu lassen, führt zu einem unvollständigen Verständnis der Anforderungen oder zu unvollständigen Anforderungen. Die Identifizierung von Anforderungsquellen ist ein iterativer und rekursiver Prozess, der stets neu überdacht werden muss.

Ein gemeinsames Verständnis (Prinzip 3 in 2) über den Kontext des zu entwickelnden Systems ist eine Voraussetzung dafür, dass die relevanten Anforderungsquellen identifiziert werden können. Der Bereich zwischen der Systemgrenze und der Kontextgrenze wird als (System-) Kontext bezeichnet (Prinzip 4in 2). Der (System-)Kontext wird benötigt, um die Art der zu entwickelnden Anforderungen zu verstehen und so die ursprünglichen Quellen für Anforderungen zu identifizieren.

Anforderungsquellen werden in drei Typen klassifiziert:

- Stakeholder
- Dokumente
- Systeme

Die Stakeholder eines Systems (Definition siehe [Glin2020], siehe auch Prinzip 2 in 2) sind die Hauptquelle für Anforderungen. Typische Stakeholder-Rollen sind [BiSp2003]:

- Nutzer (auch Endbenutzer genannt)
- Sponsoren
- Manager
- Entwickler
- Behörden
- Kunden

Darüber hinaus sollten Personen oder Organisationen, die von einem System *betroffen* sind, als (indirekte) Stakeholder betrachtet werden.

Die systematische Identifizierung von Stakeholdern sollte zu Beginn einer Entwicklung stattfinden, und die Ergebnisse sollten im Rahmen einer kontinuierlichen Aktivität verwaltet werden. Dazu gehört die Identifizierung sowohl der Stakeholder-Rollen als auch der Personen, die diese Rollen einnehmen.

Bei allen Systemen mit einer Benutzerschnittstelle stellen die *Endbenutzer* des Systems eine Stakeholder-Gruppe dar, die für den Requirements Engineer von besonderem Interesse ist. Die Endbenutzer sollten in Gruppen zusammengefasst werden (z. B. nach ähnlichen Rollen, Aufgaben oder Zuständigkeiten).

Wenn die Endbenutzer individuell identifiziert werden können, sollten Vertreter aus jeder Gruppe ausgewählt werden. Andernfalls können Personas zur Repräsentation der relevanten Endbenutzergruppen definiert werden [Coop2004].

Potentielle Quellen für die Identifizierung relevanter Stakeholder und Stakeholder-Rollen sind:

- Checklisten mit typischen Stakeholder-Gruppen und -Rollen
- Organisationsstrukturen
- Geschäftsprozessdokumentation
- Marktberichte
- Anfängliche Stakeholder zur Identifizierung *zusätzlicher* Stakeholder

Stakeholder sollten in einer aktuellen Stakeholderliste mit (mindestens) den folgenden Informationen dokumentiert werden:

- Name
- Funktion (Rolle)
- Zusätzliche persönliche und Kontaktdaten
- Zeitliche und räumliche Verfügbarkeit während der Projektlaufzeit

- Relevanz
- Fachgebiet und Umfang des Fachwissens
- Ziele und Interessen bezogen auf das Projekt

Schwierigkeiten mit Stakeholdern entstehen meist dann, wenn die Rechte und Pflichten eines Stakeholders unklar sind oder wenn die Bedürfnisse des Stakeholders unzureichend beachtet werden. Stakeholder-Relationship-Management [Bour2009] ist ein effektiver Weg, um Schwierigkeiten mit Stakeholdern entgegen zu wirken.

In den meisten Systemkontexten stehen weitere Quellen. Diese müssen für ein erfolgreiches neues System ebenfalls in Betracht gezogen werden, da die meisten Stakeholder nicht über das Offensichtliche sprechen: ihre „unterbewussten" Anforderungen (4.2).

Zusätzliche Quellen für Anforderungen sind unter anderem:

- Bestehende und Altsysteme
- Prozessdokumente
- Rechtliche oder regulatorische Dokumente
- Unternehmensspezifische Vorschriften
- (Marketing-)Informationen über potenzielle zukünftige Benutzer

Eine weitere Anforderungsquelle kann durch die Betrachtung ähnlicher Situationen in völlig anderen Bereichen gefunden werden.

4.2 Ermittlung von Anforderungen (K2)

Im Rahmen der Anforderungsermittlung hat der Requirements Engineer die Aufgabe, die Wünsche und Bedürfnisse der Stakeholder zu verstehen und gleichzeitig sicherzustellen, dass die Anforderungen aller relevanten Anforderungsquellen ermittelt werden, indem er angemessene Techniken zu deren Ermittlung einsetzt. Ein wesentlicher Teil der Anforderungsermittlung ist es, implizite Forderungen, Wünsche und Erwartungen in explizite Anforderungen zu überführen.

Für die Ermittlung von Anforderungen ist es entscheidend, das Wesen und die Notwendigkeit jeder Anforderung zu kennen. Diese können sich von Projekt zu Projekt und auch im Laufe der Zeit ändern. Das Kano-Modell [KaeA1984] klassifiziert Anforderungen in drei relevante Kategorien:

- Begeisterungsfaktoren (Synonyme: Delighters, unbewusste Anforderungen)
- Leistungsfaktoren (Synonyme: Satisfiers, bewusste Anforderungen)
- Basisfaktoren (Synonyme: Dissatisfiers, unterbewusste Anforderungen)

Es gibt viele verschiedene Techniken für die Ermittlung dieser Anforderungskategorien. Wir unterscheiden zwischen:

- Erhebungstechniken
- Entwurfs- und Ideenfindungstechniken

Erhebungstechniken sind etablierte Techniken zur Anforderungsermittlung [BaCC2015], die durch das Untersuchen unterschiedlicher Quellen helfen, Leistungsfaktoren und Basisfaktoren zu ermitteln.

Es lassen sich vier Hauptkategorien unterscheiden:

- Fragetechniken
- Kollaborationstechniken
- Beobachtungstechniken
- Artefaktbasierte Techniken

Design- und Ideengenerierungstechniken sollen die Kreativität bei der Anforderungsermittlung anregen. Sie zielen darauf ab, Ideen zur Lösung eines Problems zu schaffen und Gestaltungsideen zu erforschen [Kuma2013]. Dies kann zu neuen oder innovativen Anforderungen führen, die oft Begeisterungsfaktoren sind. Beliebte Beispiele für solche Techniken sind Brainstorming [Osbo1979], Analogien, Prototyping (z. B. Mock-Ups), Szenarien und Storyboards.

Ein weiter gefasster Begriff im Zusammenhang mit Entwurfs- und Ideenfindung ist das *Design Thinking*. Es gibt verschiedene Ansätze, wie z.B. *d.school* [Sdsc2012] und *Designing for Growth* [LiOg2011], die eine große Anzahl von Techniken anbieten, die zur Anforderungsermittlung verwendet werden können.

Ermittlungstechniken sollten in der Lage sein, alle Arten von Anforderungen, funktionale und Qualitätsanforderungen sowie Randbedingungen gleichermaßen aufzudecken. In der Praxis wird den Qualitätsanforderungen und Randbedingungen oft weniger Aufmerksamkeit geschenkt.

Für die Ermittlung von *Qualitätsanforderungen* [ISO25010] sollte ein Qualitätsmodell wie die Norm ISO 25010 als Checkliste verwendet werden. Dieses Modell kann auch hilfreich sein, um Anforderungen zu quantifizieren.

Randbedingungen können durch die Berücksichtigung möglicher Einschränkungen des potenziellen Lösungsraumes aufgedeckt werden, z.B. technische, rechtliche, organisatorische, kulturelle oder umweltbezogene Probleme.

Die Auswahl der richtigen Ermittlungstechniken ist eine entscheidende Schlüsselkompetenz, die von vielen unterschiedlichen Faktoren abhängt, z.B.:

- Art des Systems
- Software-Entwicklungs-Lebenszyklusmodell
- Beteiligte Personen
- Organisatorischer Aufbau

Die besten Ergebnisse werden in der Regel durch die Kombination verschiedener Ermittlungstechniken erzielt. [CaDJ2014] stellen einen systematischen Ansatz zur Auswahl der Techniken dar.

4.3 Lösung von Konflikten bezüglich Anforderungen (K2)

Ermittlungstechniken allein stellen nicht sicher, dass der daraus resultierende Satz von Anforderungen als Ganzes konsistent, vollständig, konform usw. ist (3.8). Für den abschließenden Satz von Anforderungen müssen alle Stakeholder alle für sie relevanten Anforderungen verstehen und ihnen zustimmen. Stimmen einige Stakeholder nicht zu, so muss dies als Konflikt betrachtet werden, der entsprechend gelöst werden sollte. Geeignete Konfliktlösungstechniken sollten abhängig von der Art des Konflikts und der Kontextinformationen ausgewählt werden. Dies erfordert ein tiefes Verständnis des Wesens des Anforderungskonflikts und der Haltung der beteiligten Stakeholder.

Aufgaben zur Identifizierung und Lösung von Konflikten sind:

- Konfliktidentifizierung
- Konfliktanalyse
- Konfliktlösung
- Dokumentation der Konfliktlösung (getroffene Entscheidungen)

Es ist sinnvoll, zwischen verschiedenen Konflikttypen zu unterscheiden [Moor2014]. Folgende Arten von Konflikten erfordern oft die Aufmerksamkeit des Requirements Engineers:

- Sachkonflikt
- Datenkonflikt
- Interessenkonflikt
- Wertekonflikt
- Beziehungskonflikt
- Strukturkonflikt

Für eine erfolgreiche Konfliktlösung können gängige Techniken angewendet werden:

- Einigung
- Kompromiss
- Abstimmung
- Ober-sticht-Unter
- Variantenbildung

Zusätzlich gibt es verschiedene unterstützende Techniken, zum Beispiel:

- Consider-All-Facts
- Plus-Minus-Interesting
- Entscheidungsmatrix

4.4 Validieren von Anforderungen (K2)

Validieren von Anforderungen ist ein wichtiger Schritt zu einem erfolgreichen System (Prinzip 6 in 2). Das Sicherstellen der Anforderungsqualität im Vorfeld reduziert unnützen Aufwand im Nachhinein. Die Validierung von Anforderungen bedeutet die Prüfung der

vorliegenden Arbeitsprodukte sowie der einzelnen Anforderungen darin bezüglich ihrer Qualität (siehe 3.8 für Details).

Wichtige Aspekte, die bei der Anforderungsvalidierung berücksichtigt werden müssen, sind:

- Beteiligung der richtigen Stakeholder
- Trennung von Fehlererkennung und Fehlerkorrektur
- Validierung aus unterschiedlichen Sichten
- Wiederholte Validierung

Es gibt mehrere Techniken zur Validierung (z.B. [GiGr1993], [OleA2018]). Diese Validierungstechniken werden oft eingeteilt in:

- *Review-Techniken*, einschließlich:
  - Walkthroughs
  - Inspektionen
- *Explorationstechniken*, zum Beispiel:
  - Prototyping
  - Alpha- und Betatests
  - A/B-Tests [KoTh2017]
  - Erstellen eines Minimum Viable Product (MVP)
- *Probe-Entwicklung*

Diese Techniken unterscheiden sich in Formalität und Aufwand. Welche Technik ausgewählt werden soll, hängt von Faktoren wie dem Software-Entwicklungs-Lebenszyklusmodell, dem Reifegrad des Entwicklungsprozesses, der Komplexität und dem Risikoniveau des Systems, etwaigen gesetzlichen oder behördlichen Anforderungen und/oder der Notwendigkeit eines Prüfpfads ab.

5 Prozess und Arbeitsstruktur (K3)

Ziel: Die Konzepte eines RE-Prozesses erklären und geeignete Prozesskonfigurationen anwenden

Dauer: 1 Stunde 15 Minuten

Begriffe: Prozess, RE-Prozess

Lernziele

LZ 5.1.1 Kennen der wichtigen Faktoren, die einen RE-Prozess beeinflussen (K1)
LZ 5.1.2 Verstehen, wie und warum diese Faktoren Einfluss haben (K2)
LZ 5.2.1 Verstehen der Facetten, die bei der Gestaltung eines RE-Prozesses zu berücksichtigen sind (K2)
LZ 5.3.1 Kennen typischer RE-Prozess-Konfigurationen (K1)
LZ 5.3.2 Verstehen der Schritte zur Konfiguration eines RE-Prozesses (K2)
LZ 5.3.3 Wählen und anwenden einer geeigneten Konfiguration des RE-Prozesses für einfache System- und Entwicklungsgegebenheiten (K3)

Zur Gestaltung und Strukturierung der in einem gegebenen Kontext zu leistenden RE-Arbeit ist ein Prozess erforderlich. Da es keinen „one-size-fits-all" RE-Prozess gibt (vgl. 1.4), muss ein maßgeschneiderter RE-Prozess konfiguriert werden, der zum gegebenen Entwicklungs- und Systemkontext passt.

Der RE-Prozess gestaltet den Informationsfluss und das Kommunikationsmodell zwischen verschiedenen Beteiligten (z.B. Kunden, Anwender, Requirements Engineers, Entwickler, Tester) und definiert auch die zu verwendenden oder zu erstellenden Arbeitsprodukte. Somit bietet der RE-Prozess den Rahmen für das Erheben, Dokumentieren, Validieren und Verwalten von Anforderungen.

5.1 Einflussfaktoren (K2)

Viele Faktoren beeinflussen die Konfiguration eines RE-Prozesses. Die wichtigsten Faktoren sind:

- Eignung im Gesamtprozess: Der RE-Prozess muss zum gesamten Systementwicklungsprozess passen.
- Entwicklungskontext
- Fähigkeit und Verfügbarkeit von Stakeholdern
- Gemeinsames Verständnis
- Komplexität und Kritikalität des zu entwickelnden Systems
- Randbedingungen
- Verfügbare Zeit und Budget
- Volatilität von Anforderungen
- Erfahrung der Requirements Engineers

Eine Analyse der Einflussfaktoren gibt Aufschluss darüber, wie der RE-Prozess zu gestalten ist. Die Einflussfaktoren schränken auch den Raum der möglichen Prozesskonfigurationen

ein. Wenn z.B. die Stakeholder nur zu Beginn des Projekts zur Verfügung stehen, kann kein Prozess gewählt werden, der auf kontinuierlichem Feedback der Stakeholder aufbaut.

5.2 Facetten von Requirements Engineering Prozessen (K2)

Es gibt drei entscheidende Facetten, die bei der Konfiguration eines RE-Prozesses berücksichtigt werden müssen [Glin2019].

Zeitfacette: Linear versus Iterativ

In einem linearen Prozess werden Anforderungen im Vorfeld in einer einzelnen Phase des Prozesses spezifiziert. In einem iterativen Prozess werden Anforderungen schrittweise spezifiziert, wobei mit allgemeinen Zielen und einigen anfänglichen Anforderungen begonnen wird und dann in jeder Iteration Anforderungen hinzugefügt oder geändert werden.

Kriterien für die Wahl eines *linearen* RE-Prozesses:

- Der Entwicklungsprozess für das System ist planbasiert und weitgehend linear.
- Die Stakeholder kennen ihre Anforderungen und können diese im Voraus spezifizieren.
- Eine umfassende Anforderungsspezifikation ist als vertragliche Grundlage für die Fremdvergabe des Designs und der Implementierung des Systems erforderlich.
- Die Regulierungsbehörden verlangen eine umfassende, formell freigegebene Anforderungsspezifikation in einem frühen Stadium der Entwicklung.

Kriterien für die Wahl eines *iterativen* RE-Prozesses:

- Der Entwicklungsprozess für das System ist iterativ und agil.
- Viele Anforderungen sind nicht im Voraus bekannt, sondern werden erst im Laufe der Entwicklung des Systems entstehen und sich entwickeln.
- Stakeholder stehen zur Verfügung, sodass kurze Rückkopplungsschleifen eingerichtet werden können, um das Risiko der Entwicklung eines falschen Systems zu mindern.
- Die Dauer der Entwicklung lässt mehr als nur ein oder zwei Iterationen zu.
- Die Möglichkeit zur leichten Änderung von Anforderungen ist wichtig.

Zweckfacette: Präskriptiv versus Explorativ

In einem präskriptiven RE-Prozess stellt die Anforderungsspezifikation einen Vertrag dar: Alle Anforderungen sind verbindlich und müssen umgesetzt werden. In einem explorativen RE-Prozess sind nur die Ziele im Vorfeld bekannt, während die konkreten Anforderungen ermittelt werden müssen.

Kriterien für die Auswahl eines *präskriptiven* RE-Prozesses:

- Der Kunde benötigt einen festen Vertrag für die Systementwicklung.
- Funktionalität und Umfang haben Vorrang vor Kosten und Terminen.
- Die Entwicklung des spezifizierten Systems kann ausgeschrieben oder ausgelagert werden.

Kriterien für die Auswahl eines *explorativen* RE-Prozesses:

- Die Stakeholder haben zunächst nur eine vage Vorstellung ihrer Anforderungen.
- Die Stakeholder sind stark involviert und liefern kontinuierliches Feedback.
- Termine und Kosten haben Vorrang vor Funktionalität und Umfang.
- Es ist nicht von vornherein klar, welche Anforderungen tatsächlich umgesetzt werden sollen und in welcher Reihenfolge sie umgesetzt werden.

Zielfacette: Kundenspezifisch versus Marktorientiert

In einem kundenspezifischen RE-Prozess wird das System von einem Kunden bestellt und von einem Lieferanten entwickelt. In einem marktorientierten RE-Prozess wird das System als Produkt oder Service für einen Markt entwickelt, ausgerichtet auf bestimmte Nutzersegmente.

Kriterien für die Wahl eines *kundenspezifischen* RE-Prozesses:

- Das System wird hauptsächlich von der Organisation genutzt, die das System bestellt hat und für seine Entwicklung bezahlt.
- Die wichtigen Stakeholder sind hauptsächlich mit der Organisation des Kunden verbunden.
- Für die Stakeholder-Rollen können konkrete Personen identifiziert werden.
- Der Kunde wünscht eine Anforderungsspezifikation, die als Vertrag dienen kann.

Kriterien für die Auswahl eines *marktorientierten* RE-Prozesses:

- Die entwickelnde Organisation beabsichtigt, das System in einem bestimmten Marktsegment als Produkt oder Dienstleistung zu verkaufen.
- Künftige Benutzer sind nicht eindeutig identifizierbar.
- Die Requirements Engineers müssen die Anforderungen so gestalten, dass sie den erwarteten Bedürfnissen der Zielnutzer entsprechen.
- Product Owner, Marketingpersonen, Digital Designer und Systemarchitekten sind die primären Stakeholder.

Hinweise und Warnungen

- Die oben vorgestellten Kriterien sind eher *Heuristiken* als feste Regeln. Beispielsweise erfolgt die Auslagerung der Entwicklung eines Systems eher mit einem präskriptiven als mit einem explorativen RE-Prozess, da der Vertrag zwischen Kunde und Lieferant in der Regel auf einer umfassenden Anforderungsspezifikation basiert. Es ist jedoch auch möglich, einen Outsourcing-Vertrag auf der Grundlage eines explorativen RE-Prozesses auszuhandeln.
- Lineare RE-Prozesse funktionieren nur, wenn ein durchdachtes Verfahren für sich ändernde Anforderungen vorhanden ist.
- Lineare RE-Prozesse implizieren lange Rückkopplungsschleifen: Um das Risiko der Entwicklung eines falschen Systems zu mindern, müssen die Anforderungen intensiv validiert werden.
- Bei der Definition eines RE-Prozesses werden häufig *linear* und *präskriptiv* gemeinsam gewählt.

- Explorative RE-Prozesse sind typischerweise auch iterativ (und umgekehrt).
- In einem marktorientierten Prozess ist das Feedback der Benutzer das einzige Mittel, um zu validieren, ob das Produkt tatsächlich die Bedürfnisse des angestrebten Benutzersegments erfüllt.
- Die marktorientierte Facette lässt sich nicht gut mit den linearen und präskriptiven Facetten kombinieren.

5.3 Konfigurieren eines Requirements Engineering Prozesses (K3)

In einem konkreten Systementwicklungskontext müssen die für das RE verantwortlichen Personen den anzuwendenden RE-Prozess konfigurieren. Basierend auf einer Analyse der Einflussfaktoren in 5.1 kann eine geeignete Kombination der in 5.2 beschriebenen Prozessfacetten eingesetzt werden [Glin2019]. Im Folgenden werden drei typische Kombinationen beschrieben.

Partizipativer RE-Prozess: Iterativ & explorativ & kundenspezifisch

Hauptanwendungsfall:	Lieferant und Kunde arbeiten eng zusammen. Die Stakeholder sind sowohl in den RE- als auch in den Entwicklungsprozess stark eingebunden.
Typische Arbeitsprodukte:	Produkt-Backlog mit User Storys und/oder Aufgabenbeschreibungen, Prototypen
Typischer Informationsfluss:	Kontinuierliche Interaktion zwischen Stakeholdern, Product Ownern, Requirements Engineers und Entwicklern; kann Rückmeldungen von Anwendern beinhalten

Vertraglicher RE-Prozess: Typischerweise linear (manchmal iterativ) & präskriptiv & kundenspezifisch

Hauptanwendungsfall:	Die Anforderungsspezifikation stellt die vertragliche Grundlage für die Entwicklung eines Systems durch Personen dar, die nicht an der Spezifikation beteiligt sind und mit nur wenig Interaktion mit den Stakeholdern nach der Anforderungsphase.
Typische Arbeitsprodukte:	Klassische System-Anforderungsspezifikation, bestehend aus natürlichsprachigen Anforderungen und Modellen
Typischer Informationsfluss:	Hauptsächlich von den Stakeholdern zu den Requirements Engineers

Produktorientierter RE-Prozess: Iterativ & explorativ & marktorientiert

Hauptanwendungsfall:	Eine Organisation spezifiziert und entwickelt Software, um sie als Produkt oder Service zu verkaufen oder zu vertreiben.
Typische Arbeitsprodukte:	Product Backlog, Prototypen
Typischer Informationsfluss:	Interaktion zwischen Product Ownern, Marketing, Requirements Engineers, Digital Designern, Entwicklern und (vielleicht) schnellem Feedback von Kunden/Benutzern

Beachten Sie, dass es System- und Entwicklungstkontexte geben kann, bei denen keine der oben genannten Konfigurationen passt. Zum Beispiel können regulatorische Vorgaben die Verwendung eines Verfahrens vorschreiben, das mit bestimmten Normen wie ISO/IEC/IEEE 29148 [ISO29148] konform ist.

Bei der Konfiguration eines RE-Prozesses empfehlen wir ein fünfstufiges Vorgehen:

1. Analysieren der Einflussfaktoren (5.1)
2. Beurteilen der Facettenkriterien (5.2)
3. Konfigurieren des Prozesses (5.3)
4. Arbeitsprodukte bestimmen (3)
5. Geeignete Praktiken auswählen

6 Praktiken für das Requirements Management (K2)

Goal: Verstehen der Notwendigkeit und des Nutzens von Requirements Management

Dauer: 2 Stunden

Begriffe: Requirements Management, Änderungsmanagement, Verfolgbarkeit (Traceability), Anforderungsattribute, Anforderungslebenszyklus, Priorisierung

Lernziele

LZ 6.1.1 Wissen, worum es beim Requirements Management geht und warum es notwendig ist (K1)
LZ 6.2.1 Erklären, warum Arbeitsprodukte ein Status-/Lebenszyklusmodell benötigen (K2)
LZ 6.3.1 Erklären, wie ein Versionierungskonzept für Anforderungen in einer bestimmten Projektsituation aussieht (K2)
LZ 6.4.1 Die Verwendung von Anforderungskonfigurationen und Basislinien kennen (K1)
LZ 6.5.1 Den Zweck von Attributen für Anforderungen kennen (K1)
LZ 6.5.2 Erklären, wie ein geeigneter Satz von Attributen für Anforderungen in einer bestimmten Projektsituation aussieht (K2)
LZ 6.5.3 Den Zweck von Sichten auf Anforderungen erklären und die unterschiedlichen Sichten auf Anforderungen benennen (K2)
LZ 6.6.1 Gründe für die Verfolgbarkeit von Anforderungen nennen (K1)
LZ 6.6.2 Unterschiede zwischen impliziter und expliziter Verfolgbarkeit zusammenfassen (K1)
LZ 6.6.3 Wissen, wie explizite Verfolgbarkeit dokumentiert werden kann (K1)
LZ 6.7.1 Wissen, wie man mit Änderungen in linearen (planbasierten) und agilen Ansätzen umgeht (K1)
LZ 6.8.1 Den Grund für die Priorisierung von Anforderungen und sinnvolle Bewertungskriterien kennen (K1)
LZ 6.8.2 Schritte zur Priorisierung von Anforderungen benennen (K1)
LZ 6.8.3 Verschiedene Kategorien von Priorisierungstechniken benennen (K1)

6.1 Was ist Requirements Management? (K1)

Requirements Management ist der Prozess der Verwaltung bestehender Anforderungen, die in verschiedenen Arbeitsprodukten aufgezeichnet sind. Dazu gehören insbesondere das Speichern, Ändern und Verfolgen von Anforderungen [Glin2020]. Requirements Management kann je nach gewähltem Entwicklungsprozess und -kontext auf unterschiedliche Arten und auf unterschiedlichen Ebenen erfolgen, z.B. [Leff2011], [Rupp2014], [WiBe2013]. Unabhängig von den Gegebenheiten, besteht die Aufgabe des Requirements Management darin, Anforderungen so zu verwalten, dass alle Rollen in einem Projekt effektiv und effizient arbeiten können.

6.2 Verwaltung des Lebenszyklus (K2)

Verwaltung des Lebenszyklus bedeutet, alle Arbeitsprodukte in Bezug auf ihren Status im Lebenszyklus zu überwachen. Jede dokumentierte Anforderung und jedes Arbeitsprodukt

hat seinen eigenen Lebenszyklus: Es wird erstellt, dann bewertet und verfeinert, bevor es überprüft, überarbeitet, konsolidiert, vereinbart etc., wird. Zur Identifizierung, welche Arbeitsprodukte sich in welchem Status befinden, ist ein Lebenszyklusmodell notwendig, das alle zulässigen Lebenszyklus-Zustände und Zustandsübergänge definiert. Der aktuelle Status eines Arbeitsprodukts sollte immer klar erkennbar sein, einschließlich (üblicherweise) der Historie seiner Veränderungen.

6.3 Versionskontrolle (K2)

Versionskontrolle von Anforderungen bezeichnet den Prozess der Überwachung aller Arbeitsprodukte während ihrer Evolution. Jede Änderung in einem Arbeitsprodukt sollte sich in einer neuen Version widerspiegeln. Versionierung ermöglicht es, die Historie eines Arbeitsprodukts bis zu seinem Ursprung zurückzuverfolgen und jede frühere Version dieses Arbeitsprodukts wieder herzustellen. Zu diesem Zweck sind zur Versionskontrolle drei Maßnahmen erforderlich:

- Einer Versionsnummer, um die Version eines Arbeitsprodukts eindeutig zu identifizieren.
- Eine Historie dessen, was geändert wurde.
- Einem Konzept für die Speicherung von Arbeitsprodukten.

Versionierung muss bei allen Arbeitsprodukten berücksichtigt werden [WiBe2013]. Eine Versionsnummer besteht typischerweise aus mindestens zwei Teilen: Der Version und dem Inkrement.

6.4 Konfigurationen und Basislinien (K1)

Eine *Anforderungskonfiguration* ist eine konsistente Zusammenstellung von Arbeitsprodukten, die Anforderungen enthalten. Jede Konfiguration ist für einen bestimmten Zweck definiert und enthält höchstens eine Version jedes Arbeitsprodukts [Glin2020]. Der Zweck von Konfigurationen besteht beispielsweise darin, ein Review auf eine Zusammenstellung von Arbeitsprodukten durchzuführen oder um den Entwicklungsaufwand abzuschätzen.

Eine *Basislinie* ist eine stabile, änderungskontrollierte *Konfiguration* von Arbeitsprodukten und dient der Release-Planung oder anderen Liefermeilensteinen in einem Projekt[Glin2020].

Konfigurationen haben die folgenden Eigenschaften:

1. Logische Verbindung
2. Konsistenz
3. Einzigartigkeit
4. Unveränderlichkeit
5. Grundlage für das Zurücksetzen

6.5 Attribute und Sichten (K2)

Attribute sind erforderlich, um wichtige Metadaten für ein Arbeitsprodukt zu dokumentieren und werden typischerweise verwendet, um eine Reihe wichtiger Fragen während des Projekt- oder Produktlebenszyklus zu beantworten.

Das Ziel der Verwendung von Attributen zur Charakterisierung von Anforderungen besteht darin, es den Teammitglieder und anderen Stakeholdern zu ermöglichen, zu jedem Zeitpunkt während des Projekts die Informationen über die Anforderungen zu erhalten, die sie benötigen.

Die Festlegung der relevanten Attribute hängt vom Informationsbedarf der verschiedenen Stakeholder im Projekt ab. Bestehende Normen, z.B. [ISO29148], geben einen Überblick über die wichtigsten Attribute.

Sichten sind ein Auszug aus dem Gesamtpaket der Anforderungen, die nur den aktuell interessanten Inhalt enthalten. Aus technischer Betrachtung ist eine Sicht eine Kombination von Filter- und Sortier-Einstellungen, die durch Speicherung der ausgewählten Kombination anderen Teilnehmern zur Verfügung gestellt oder wiederverwendet werden kann.

Wir unterscheiden zwischen drei Arten von Sichten:

- *Selektive Sichten*
- *Projektive Sichten*
- *Verdichtende Sichten*

In den meisten Fällen sind Anforderungssichten Kombinationen aus selektiven, projektiven und verdichtenden Sichten für die Erstellung von Berichten.

6.6 Verfolgbarkeit (K1)

Verfolgbarkeit [GoFi1994] ist die Fähigkeit, eine Anforderung zurück zu ihrem Ursprung (d. h. zu Stakeholdern, Dokumenten, Begründungen usw.) und weiter zu nachfolgenden Arbeitsprodukten (z. B. Testfällen) sowie zu anderen Anforderungen, von denen diese Anforderung abhängt, zu verfolgen.

Die Verfolgbarkeit ist eine Voraussetzung für das Verwalten von Anforderungen und wird in Normen, Gesetzen und Vorschriften oft explizit gefordert. Die Umsetzung von Verfolgbarkeit bedeutet im Wesentlichen die Aufrechterhaltung von Beziehungen zwischen verschiedenen Arbeitsprodukten (3.1) auf verschiedenen Abstraktionsebenen (3.1.2), Detaillierungsgraden (3.1.3) und zu allen relevanten Vorgängern und Nachfolgern aus Gründen der Analyse, Konformität und Information.

Die Verfolgbarkeit kann *implizit* dokumentiert werden, indem die Arbeitsprodukte strukturiert und standardisiert werden, oder *explizit,* indem die Arbeitsprodukte auf der Grundlage ihrer eindeutigen Identifikationsmerkmale in verschiedenen Formen zueinander in Beziehung gesetzt werden [HuJD2011]. Gängige Darstellungsformen sind Hyperlinks, Referenzen, Matrizen, Tabellen oder Graphen.

6.7 Umgang mit Änderungen (K1)

Anforderungen sind nicht statisch. Änderungen in Anforderungen haben viele verschiedene Gründe und müssen richtig gehandhabt werden (Prinzip 7 in 2), z.B. durch die Erstellung eines formellen *Änderungsantrags* oder durch das Hinzufügen eines neuen Eintrags im *Produkt-Backlog*.

Die Entscheidungsfindung, Planung und Kontrolle der Umsetzung einer Änderung hängt vom Entwicklungsansatz und dem Zeitpunkt ab, zu dem die Änderung auftritt.

In einem *linearen* Ansatz wird die Entscheidung über eine Änderung oft von einem Change Control Board (bei Projekten) oder einem Change Advisory Board (im Betrieb) getroffen. In einem eher *iterativen* Ansatz nimmt der Product Owner die Änderung in das Produkt-Backlog auf und priorisiert den neuen Eintrag entsprechend.

6.8 Priorisierung (K1)

Nicht alle Anforderungen sind gleich wichtig [Davi2005]. Bewertung und Priorisierung werden verwendet, um die wichtigsten Anforderungen für das nächste Produkt-Release oder Inkrement zu bestimmen.

Die *Bewertung* der Anforderungen ist die Grundlage für ihre Priorisierung, die oft durch die Verwendung mehrerer Bewertungskriterien wie Geschäftswert, Dringlichkeit, Aufwand, Abhängigkeiten und andere bestimmt wird.

Die *Priorität* einer Anforderung beschreibt die Bedeutung einer einzelnen Anforderung im Vergleich zu anderen Anforderungen anhand bestimmter Kriterien [Glin2020]. Die *Priorisierung* selbst wird auf der Grundlage eines einzigen oder mehrerer Kriterien durchgeführt, dies hängt hauptsächlich von der gewählten Priorisierungstechnik ab.

Schritte zur Priorisierung:

- Festlegen der wichtigsten Ziele und Randbedingungen für die Priorisierung
- Definieren der gewünschten Beurteilungskriterien
- Festlegen, welche Stakeholder einbezogen werden müssen
- Festlegen, welche Anforderungen priorisiert werden müssen
- Auswahl der Priorisierungstechnik
- Durchführung der Priorisierung

Priorisierungstechniken können klassifiziert werden in:

- *Ad-Hoc* Priorisierungstechniken
- *Analytische* Priorisierungstechniken

7 Werkzeugunterstützung (K2)

Ziel: Einen Überblick über die Rolle von RE-Werkzeugen und Aspekte für die Umsetzung geben

Dauer: 30 Minuten

Begriffe: Werkzeug, RE-Werkzeug

Lernziele

LZ 7.1.1 Die verschiedenen Arten von RE-Werkzeugen kennen (K1)
LZ 7.2.1 Erklären, was bei der Einführung von RE-Werkzeugen zu beachten ist (K2)

7.1 Werkzeuge im Requirements Engineering (K1)

Der RE-Prozess kann durch Werkzeuge unterstützt werden, die spezielle Aufgaben und Aktivitäten erleichtern. Da der RE-Prozess recht individuell ist (5), konzentrieren sich die bestehenden RE-Werkzeuge oft nur auf bestimmte Aspekte im RE und unterstützen selten alle Aktivitäten. Requirements Engineers sollten vor der Auswahl eines Werkzeugs entscheiden, welche Aufgaben und Aktivitäten während des RE-Prozesses wie unterstützt werden sollen. Wir unterschieden zwischen Werkzeugen, die folgendes unterstützen:

- Verwaltung von Anforderungen:
  - Definieren und Speichern von Anforderungsattributen
  - Priorisierung von Anforderungen
  - Verwalten von Versionen und Konfigurationen
  - Verfolgung von Anforderungen
- Verwaltung von Anforderungsänderungen
  - Steuerung des RE-Prozesses:
  - Messung und Berichterstattung über den RE-Prozess
  - Messung und Berichterstattung über die Produktqualität
- Verwaltung des RE-Workflows
  - Dokumentation des Kenntnisstandes über die Anforderungen:
  - Gemeinsame Nutzung der Anforderungen
  - Schaffung eines gemeinsamen Verständnisses der Anforderungen
- Modellierung von Anforderungen
- Zusammenarbeit im RE
- Testen/Simulieren von Anforderungen

Werkzeuge bieten häufig eine Kombination der oben genannten Merkmale. Um sicherzustellen, dass alle RE-Aufgaben angemessen abgedeckt werden, können verschiedene Werkzeuge kombiniert werden.

Manchmal werden auch andere Arten von Werkzeugen, z.B. Office- oder Issue-Tracking-Tools, verwendet, um Anforderungen zu dokumentieren oder zu verwalten. Diese Werkzeuge haben ihre Grenzen und sollten nur verwendet werden, wenn der RE-Prozess beherrscht wird und die Anforderungen abgestimmt und hinreichend stabil sind.

7.2 Werkzeugeinführung (K2)

Die Auswahl eines RE-Werkzeugs unterscheidet sich nicht von der Auswahl eines Werkzeugs für einen anderen Zweck. Das Ziel, der Kontext und die Anforderungen müssen für eine erfolgreiche Auswahl im Vorfeld beschrieben werden [Fugg1993].

Ein geeignetes Werkzeug kann erst dann ausgewählt werden, wenn die richtigen RE-Vorgehensweisen und Techniken eingeführt worden sind. Die Werkzeugeinführung setzt klare Verantwortlichkeiten und Vorgehensweisen im RE voraus. Bei der Einführung eines RE-Tools sind folgende Aspekte relevant:

- Berücksichtigung sämtlicher Lebenszykluskosten zusätzlich zu den Lizenzkosten
- Benötigte Ressourcen einplanen
- Pilotprojekte nutzen, um Risiken zu vermeiden
- Bewertung des Werkzeugs nach definierten Kriterien
- Mitarbeiter in der Anwendung des Tools unterweisen

Literaturverzeichnis

[AnPC1994] Annie I. Antón, W. Michael McCracken, Colin Potts: Goal Decomposition and Scenario Analysis in Business Process Reengineering. CAiSE (Conference on Advanced Information Systems Engineering), 1994, 94–104.

[BaCC2015] K. Baxter, C. Courage, K. Caine: Understanding Your Users: A Practical Guide to User Research Methods, 2. Auflage. Morgan Kaufmann, Burlington, 2015.

[BiSp2003] K. Bittner, I. Spence: Use Case Modelling. Pearson Education, Boston, 2003.

[Bour2009] L. Bourne: Stakeholder Relationship Management: A Maturity Model for Organisational Implementation. Gower Publishing Ltd, Birlington, 2009.

[CaDJ2014] D. Carrizo, O. Dieste, N. Juristo: Systematisierung der Auswahl der Erhebungstechnik für die Anforderungen. Information and Software Technology 2014, 56(6): 644–669.

[Cock2001] A. Cockburn: Use Cases effektiv erstellen. Addison-Wesley, Boston 2001.

[Cohn2004] M. Cohn: User Stories – für die agile Software-Entwicklung mit Scrum, XP u.a. mitp, Heidelberg, 2010.

[Coop2004] A. Cooper: The Inmates are Running the Asylum: Why High Tech Products Drive Us Crazy and How to Restore the Sanity. Que, Indianapolis, 2004.

[Davi2005] A. M. Davis: Just Enough Requirements Management – Where Software Development Meets Marketing. Dorset House Publishing, New York, 2005.

[Davi1993] A. M. Davis: Software Requirements – Objects, Functions, & States, 2nd edition, Prentice Hall, New Jersey, 1993.

[DeMa1978] T. DeMarco: Strukturierte Analyse und Systemspezifikation. Yourdon Press, New York, 1978.

[Fugg1993] A. Fuggetta: Eine Klassifizierung der CASE-Technologie. IEEE Computer 1993, 26 (12): 25–38.

[GiGr1993] T. Gilb, D. Graham: Software Inspektion. Addison Wesley, Boston, 1993.

[Glin2019] M. Glinz: Requirements Engineering I. Kursnotizen, Universität Zürich, 2019. https://www.ifi.uzh.ch/en/rerg/courses/archives/hs19/re-i.html#resources. Zuletzt besucht im Juli 2020.

[Glin2020] M. Glinz: Wörterbuch der Requirements Engineering Terminologie. Version 2.0. https://www.ireb.org/de/downloads/#cpre-glossary. Zuletzt besucht im Juli 2020.

[GoFi1994] O. Gotel, A.Gotel, A. Finkelstein: An Analysis of the Requirements Traceability Problem. 1. Internationale Konferenz über Anforderungsmanagement, Colorado Springs, 1994. 94–101.

[GoRu2003] R. Goetz, C. Rupp: Psychotherapie für Systemvoraussetzungen. 2. Internationale Konferenz des IEEE über kognitive Informatik (ICCI'03), London, 2003. 75–80.

[GRL2020] Goal oriented Requirement Language. University of Toronto, Canada https://www.cs.toronto.edu/km/GRL. Zuletzt besucht im Mai 2020.

[Hare1988] D. Harel: On Visual Formalisms. Communications of the ACM 1988, 31 (5): 514–530.

[HoSch2020] S. Hofer, H. Schwentner: Domain Storytelling – A Collaborative Modeling Method. Verfügbar auf Leanpub, http://leanpub.com/domainstorytelling. Zuletzt besucht im Juli 2020.

[HuJD2011] E. Hull, K. Jackson, und J. Dick: Requirements Engineering. Springer, 3rd Ed, 2011.

[ISO29148] ISO/IEC/IEEE 29148: Systems and Software Engineering – Life Cycle Processes – Requirements Engineering, International Organization for Standardization, 2018.

[ISO19650] ISO 19650: Organization and Digitization of Information about Buildings and Civil Engineering Works, including Building Information Modelling (BIM)– Information Management Using Building Information Modelling – Part 1 and 2, International Organization for Standardization, 2018.

[ISO25010] ISO/IEC/IEEE25010:2011: Systems and software Quality Requirements and Evaluation (SQuaRE) –– System and software quality models. International Organization for Standardization, Genf, 2011.

[Jack1995] M. A. Jackson: Software Requirements and Specifications: A Lexicon of Practice, Principles and Prejudices. Addison-Wesley, New York, 1995.

[Jack1995b] M. Jackson: Die Welt und die Maschine. 17th International Conference on Software Engineering 1995 (ICSE 1995). 287–292.

[KaeA1984] N. Kano et al: Attraktive Qualität und Must-be-Quote. Journal of the Japanese Society for Quality Control 1984, 14(2): 39–48. (auf Japanisch)

[KoTh2017] R. Kohavi, S. Thomke: Die überraschende Kraft von Online-Experimenten – Das Beste aus A/B- und anderen kontrollierten Tests herausholen. Harvard Business Review, September-Oktober 2017: 74–82.

[Kuma2013] V. Kumar: 101 Design Methods – A Structured Approach for Driving Innovation in Your Organization. John Wiley & Sons, Hoboken, 2013.

[Laue2002] S. Lauesen: Software-Anforderungen. Styles und Techniken Addison-Wesley, Harlow, 2002.

[Leff2011] D. Leffingwell: Agile Software Requirements, Lean Requirements Practices for Teams, Programs, and the Enterprise. Addison-Wesley, Boston, 2011.

[LiOg2011] J. Liedtka, T.Liedtka, T. Ogilvie: Designing for Growth: A Design Thinking Tool Kit For Managers. Columbia University Press, 2011.

[LiSZ1994] H. Lichter, M. Schneider-Hufschmidt, H. Zullighoven: Prototyping in industriellen Softwareprojekten – Überbrückung der Kluft zwischen Theorie und Praxis. IEEE Transactions on Software Engineering 1994, 20 (11): 825–832.

[MFeA2019] D. Méndez Fernández, X. Franch, N. Seyff, M. Felderer, M. Glinz, M. Kalinowski, A. Volgelsang, S. Wagner, S. Bühne, K. Lauenroth: Predigen wir, was wir praktizieren? Untersuchung der praktischen Relevanz von Lehrplänen des Requirements Engineering – Der Fall des IREB. CIbSE 2019: 476–487.

[Moor2014] C. W. Moore: The Mediation Process – Practical Strategies for Resolving Conflicts, 4th edition. John Wiley & Sons, Hoboken, 2014.

[MWHN2009] A. Mavin, P. Wilkinson, A. Harwood, and M. Novak: Leichte Annäherung an die Anforderungssyntax (EARS). 17. Internationale Konferenz des IEEE zur Anforderungserhebung (RE'09), Atlanta, Georgia, 2009. 317–322.

[OleA2018] K. Olsen et al.: Certified Tester, Foundation Level Syllabus – Version 2018. International Software Testing Qualifications Board, 2018.

[OMG2013] Object Management Group: Business Process Model and Notation (BPMN), version 2.0.2. OMG document formal/2013-12-09 http://www.omg.org/spec/BPMN. Zuletzt besucht im Juli 2020.

[OMG2017] Object Management Group: OMG Unified Modeling Language (OMG UML), version 2.5.1. OMG document formal/2017-12-05. https://www.omg.org/spec/UML/About-UML/. Zuletzt besucht im Juli 2020.

[OMG2019] Object Management Group: OMG Systems Modeling Language (OMG SysML™), Version 1.6. OMG Document formal/19-11-01. https://www.omg.org/spec/SysML/. Zuletzt besucht im Januar 2022.

[Osbo1979] A. F. Osborn: Applied Imagination, 3rd revised edition. Charles Scribner´s Sons, New York, 1979.

[RoRo2012] S. Robertson and J. Robertson: Beherrschung des Anforderungsprozesses. Addison-Wesley, Boston, 2012.

[Rupp2014] C. Rupp: Requirements-Engineering und Management, 6. Auflage. Hanser, München, 2014. (auf Deutsch).

[Sdsc2012] Stanford d.school: Eine Einführung in das Design Thinking. Hasso Plattner Institute of Design, Stanford, 2012. https://dschool-old.stanford.edu/groups/designresources/wiki/36873. Zuletzt besucht im Juli 2020.

[vLam2009] Axel van Lamsweerde: Requirements Engineering: From System Goals to UML Models to Software Specifications. Chichester: John Wiley & Sons, 2009.

[Vole2020] Volere: Requirements Resources. https://www.volere.org. Zuletzt besucht im Juli 2020.

[WiBe2013] K. Wiegers and J. Beatty: Software Requirements, 3rd edition. Microsoft Press, Redmond, 2013.