Requirements Reengineering

WIE SICH DIE QUALITÄT VON ANFORDERUNGSDOKUMENTEN NACHTRÄGLICH ERHÖHEN LÄSST

Dieser Beitrag beschreibt ein werkzeuggestütztes Verfahren, um die Qualität der Anforderungsdokumente nachträglich zu steigern und damit eine fundierte Kostenkalkulation zu ermöglichen. So kann die Anforderungsdokumentation zum Testorakel im „Requirement-based Testing“ werden.

Die meisten Probleme in der Softwareentwicklung sind auf die mangelhafte Anfor-derungsdokumentation zurückzuführen. Das haben zahlreiche Studien belegt. Die Anforderungen sind zumeist oberflächlich, unvollständig und inkonsistent. Es sind darin längst nicht alle Fragen geklärt. Die Dokumente sind schlicht und ergreifend laienhaft. Dennoch werden sie als Grundlage für die Ausschreibung wichtiger Projekte verwendet. Hinzu kommt, dass die Bewerber für die ausgeschriebenen Projekte verbindliche Angebote aufgrund dieser unzulänglichen Dokumente abgeben müssen. Damit sind die Erfolgschancen für die geplanten Projekte von Anfang an stark eingeschränkt. Um dies zu verhindern, muss die Qualität der Anforderungsdokumente – soge-nannte „Lasten-“ und „Pflichtenhefte“ – erhöht und stärker kontrolliert werden. Es muss möglich sein, anhand der Dokumente den Aufwand und die Kosten des geplanten Vorhabens annähernd genau zu schätzen. Am besten wäre hierzu ein werkzeuggestütztes Verfahren.

„Refaktorierung“ des Lastenhefts

Anforderungsdokumente werden im deutschsprachigen Raum in der Regel als deutschsprachige Texte von den zuständigen Fachabteilungen verfasst. Die Verfasser der Dokumente sind Sachgebietsexperten und haben keine formale Ausbildung in „Requirements Engineering“. Sie sind überfordert, eine fachliche Lösung zu beschreiben und gleichzeitig so zu formulieren, dass sie den Qualitätsansprüchen des „Requirements Engineering“ genügt. Wir schlagen daher ein Zwei-Schritt-Verfahren vor:

- Im ersten Schritt werden die Anforderungen in einem Lastenheft erfasst und eine fachliche Lösung in einem Pflichtenheft ausgearbeitet.
- Im zweiten Schritt werden die beiden Dokumente überarbeitet und in eine Form versetzt, die sich weiterverarbeiten lässt.

Die Anforderungsdokumente werden im Sinne der Programmentwicklung „refaktoriert“ und zwar durch professionelle Requirements-Ingenieure, auch Ghost-Writer genannt. Dieses Sanieren beinhaltet die Restrukturierung und Markierung der Texte mittels einer Auszeichnungssprache (Mark-up), damit sie von einem Textanalyse-Werkzeug verarbeitet werden können. Der zweite Schritt wird als „Requirements Reengineering“ bezeichnet [Bria17].
Falls die Qualität der Anforderungsdokumentation immer noch zu niedrig ist oder die Kosten des Projektes zu hoch sind, kann die zuständige Fachabteilung auf den ingenieurmäßig aufbereiteten Dokumenten aufsetzen und sie überarbeiten, entweder um die Qualität zu steigern, indem Mängel beseitigt werden, oder die Kosten zu senken, indem Anforderungen gestrichen werden. Dieser Reengineering-Prozess wird solange wiederholt, bis die Anforderungsdokumentation eine ausreichende Qualität und das damit beschriebene System einen vertretbaren Umfang hat [Berg15].

Zweck des Requirements Reengineering

Es gibt keinen verbindlichen Standard für die Gestaltung von Anforderungsdokumenten. Bisher hat es nur Empfehlungen und Richtlinien gegeben. Die erste Empfehlung war der IEEE Guide für „Software Requirements Specifications“ [IEEE83]. Diese allgemeine Richtlinie wurde mehrfach revidiert bis er 2011 vom ISO/IEC/IEEE-Standard 29148 [ISO11] abgelöst wurde. Dieser Standard regelt in erster Linie die Struktur und den formalen Inhalt eines Anforderungsdokuments.
Im deutschsprachigen Raum kommt das V-Modell-XT des bundesdeutschen Innenministeriums mit zusätzlichen Detailvorschriften für Lasten- und Pflichtenhefte hinzu [Bund15]. Allerdings sind diese Richtlinien recht allgemein formuliert. Jede Organisation ist aufgefordert, sie durch eigene Richtlinien zu ergänzen. In vielen Fällen ist es den einzelnen Fachbereichs-Analytikern überlassen, ihre Anforderungen nach Belieben zu verfassen. Wenn sie die Anforderungen nach dem Volere-Modell von Robertson und Rupp [RoRo99] formulieren, steigt zwar die Qualität des Lastenhefts, aber das gibt wenig her in Richtung Aufwandsschätzung. Um Aufwand zu schätzen, braucht der Schätzer zumindest ein fachliches Lösungskonzept beziehungsweise ein Pflichtenheft. Ohne dies kann von einer systematischen Aufwandsschätzung keine Rede sein.
Um ein Fachkonzept ordentlich zu prüfen und zu vermessen, braucht es auch eine normierte Struktur auf Basis eines semiformalen Modells [Eber12].
Das Fachkonzept muss das zugrunde liegende Modell widerspiegeln. Falls es das in der ursprünglichen Form nicht tut, muss es nachträglich eingearbeitet werden. Es geht darum, dem informalen Text zumindest ein semiformales Modell aufzuerlegen, ein Modell, das es erlaubt, die Anforderungen im Sinne des Modells zu interpretieren. Das originale Anforderungsdokument muss so weit nachbearbeitet werden, bis gewisse Konzeptelemente darin erkennbar sind, die eine Qualitätsbewertung und eine algorithmische Aufwandsschätzung zulassen. Dazu gehören solche Konzeptelemente wie gewünschte Funktionen, erforderliche Stammdaten, Eingaben, Ausgaben und angestrebte Qualitätskriterien.

Diese minimalen Modellelemente zu identifizieren und kenntlich zu machen ist der Zweck des Requirements Reengineering.

Voraussetzungen für die werkzeugunterstützte Anforderungsanalyse

Damit die Ziele erreicht werden können, müssen zunächst gewisse Vorbedingungen erfüllt werden. Es ist selten möglich, Anforderungsdokumente, die direkt vom Anwender stammen, automatisch zu analysieren. Technisch ist es machbar. Der Anforderungsanalytiker muss nur die PDF- (.pdf) oder DOC-Dateien (.docx) in ein Text-Format (.txt) umwandeln. Dabei gehen einige grafische Komponenten, wie Baum- und Flussdiagramme, verloren, aber solche Darstellungen sind meistens ohnehin im Text enthalten. Die Abbildungen in Anforderungsdokumenten sind zusätzlich zum geschriebenen Wort. Verwertbare Ergebnisse erhält man nur, wenn das Anforderungsdokument vorstrukturiert ist und die Schlüsselentitäten markiert sind. Also ist eine „ingenieurmäßige Sanierung der Anforderungen“, neudeutsch Requirement Reengineering, erforderlich. Wie aufwendig diese Sanierungsarbeit ist, hängt davon ab, wie groß und wie verständlich das Originaldokument ist. Die größte bisher von den Autoren verarbeitete Anforderungsdokumentation bestand aus den in Tabelle 1 genannten Teilen und konnte innerhalb von zehn Tagen von einem Anforderungsanalytiker mit einer Hilfskraft so saniert werden, dass sämtliche Anforderungen, Anwendungsfälle und Datenobjekte von einem Textanalysewerkzeug erkannt werden konnten. Die Messung ergab unter anderem die in Tabelle 2 zusammengefassten Ergebnisse.

Lastenheft	89 Seiten
Pflichtenheft	286 Seiten
Datenmodell	70 Seiten
Gesamt	445 Seiten

Tabelle 1: Umfang einer Anforderungsdokumentation

Anzahl der erkannten Zustände	1.396
Anzahl der erkannten Aktionen	1.964
Anzahl der erkannten Regel	1.454
Anzahl spezifizierter Systemakteure	8
Anzahl spezifizierter Anwendungsfälle	267
Anzahl der Anwendungsfallauslöser	5
Anzahl der Anwendungsfallpfade	199
Anzahl der Anwendungsfallschritte	737
Anzahl der Anwendungsfallvorbedingungen	24
Anzahl der Anwendungsfallnachbedingungen	80
Anzahl der Anwendungsfallausnahmen	0
Anzahl der Anwendungsfallbeziehungen	932
Anzahl der fachlogischen Testfälle	3.871

Tabelle 2: Ausschnitt aus einem Anforderungsmetrikbericht

Das Modell hinter dem Anforderungsdokument

Jedes strukturierte Anforderungsdokument verbirgt ein Modell des geplanten Systems. Auch unstrukturierte Dokumente beinhalten ein Modell, es ist nur schwer zu erkennen. Die Anforderungsanalyse hat die Aufgabe, dieses Modell herauszuarbeiten. Ein Objekt-Modell besteht aus Entitäten und Beziehungen zwischen den Entitäten:

- In einem Modell eines IT-Systems sind die Entitäten beziehungsweise die Datengruppen, die Einzeldaten, die Vorgänge beziehungsweise Anwendungsfälle, die Aktionen, die Benutzer beziehungsweise Akteure, die Regeln und die Anforderungen.
- Die Beziehungen sind die Abhängigkeiten und Interaktionen zwischen diesen Entitäten, zum Beispiel die Objekt-zu-Objekt-, die Vorgang-zu-Vorgang- und die Vorgang-zu-Objekt-Beziehungen [Chen79].

Es ist der Zweck des Anforderungs-Re-Engineerings, das dahinterliegende Modell zu enthüllen. Denn nur wenn das Modell erkennbar ist, lässt sich der Aufwand, das Modell zu implementieren, kalkulieren. Alle algorithmischen Schätzmethoden basieren auf einem Modell des geplanten Systems – Function-Points auf dem Funktionsmodell, Data-Points auf dem Datenmodell, Object-Points auf dem Objektmodell und so weiter. Die Größe eines IT-Systems wird durch a.) die Anzahl der Entitäten und b.) die Anzahl der Beziehungen bestimmt. Es muss also möglich sein, die Entitäten und Beziehungen zu zählen. Die Anforderungssanierung zielt darauf ab, die Modellelemente in den Textdokumenten zu erkennen und zu markieren. Danach können sie eindeutig gezählt werden.

Soll-Ist-Vergleich der Anforderungsdokumente

Anforderungsdokumente, die von den Fachbereichsanalytikern verfasst werden, sind nicht gänzlich ohne Struktur und normierten Inhalt. Es gibt eine Vielzahl von Büchern darüber, wie IT-Vorhaben zu definieren sind [PoRu15]. Viele Fachbereichsanalytiker haben Kurse besucht, in denen sie gelernt haben, IT-Systeme semiformal zu beschreiben. Sie wissen zwischen funktionalen und nicht-funktionalen Anforderungen zu unterscheiden. Sie wissen, wie ein Datenmodell darzustellen ist und wie Anwendungsfälle abzubilden sind. Insofern lehnt sich jedes Anforderungsdokument zu einem gewissen Grad an ein Referenzmodell an. Was die Fachbereichsanalytiker schon erreicht haben, sollte hier möglichst erhalten bleiben. Die zusätzlichen Modellentitäten und Beziehungen, die der Anforderungssanierer einbaut, sollten auf den bestehenden aufsetzen. Sie sind sozusagen eine konstruktive Ergänzung. Es geht darum, möglichst viele Dokumentationselemente so zu belassen, wie man sie vorfindet, nur ergänzt durch Auszeichnungselemente (Mark-ups) und Schlüsselwörter, damit die Dokumente automatisch analysiert werden können. Falls die Auszeichnungselemente aber fehlen, sollten sie eingepflegt werden [FeHW16].
Der erste Schritt bei der Anforderungs-Re-engineerings-Analyse ist herauszufinden, zu welchem Grad das vorliegen-

de originale Anforderungsdokument dem Referenz-Anforderungsdokument entspricht. Der Anforderungsanalytiker vergleicht dabei die in die Konzepttexte eingewickelten Modellelemente mit den Modellelementen, die hier propagiert werden. Falls sie fehlen, ist es seine Aufgabe, sie einzubauen. Falls sie präsent, aber unzulänglich beschrieben sind, ist es seine Aufgabe, die Beschreibung zu verschärfen. Durch das Editieren des ursprünglichen Dokumentes entsteht ein neues und formal verbessertes Anforderungsdokument im Sinne des Requirements Engineerings [Pohl07].

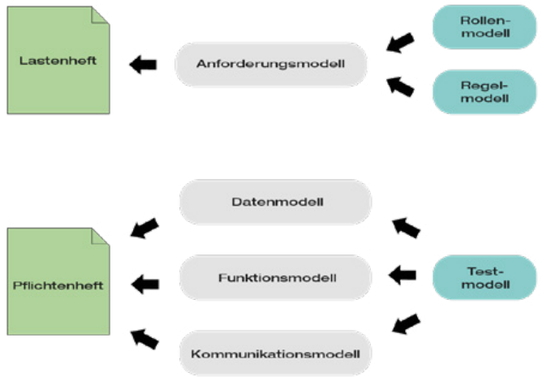

Abb. 1: Lasten- und Pflichtenheft

Struktur der Anforderungsdokumentation

Anforderungsdokumente im deutschsprachigen Raum bestehen nach dem geltenden V-Modell aus Lasten- und Pflichtenheften (siehe Abbildung 1). Das V-Modell-XT ist verbindlich für die meisten deutschen Behörden und gilt auch in Österreich als Richtlinie für Entwicklungsprojekte im öffentlichen Bereich [HöHö08]. Lastenhefte beschreiben „Was“ und Pflichtenhefte beschreiben „Wie“ etwas zu erreichen ist. In dem Lastenheft werden die Kundenanforderungen spezifiziert. In dem Pflichtenheft wird eine technische Lösung zur Erfüllung der Kundenanforderungen dargestellt. Diese Unterscheidung gibt es in der englischsprachigen Welt nicht. Im Englischen gibt es nur „The Requirement Specification“. In beiden Sprachen gibt es „das Design“, das eine Vorstufe der Implementierung ist. Heutzutage ist das Design in der Regel ein UML-Modell.

Das Lastenheft

Das Lastenheft beziehungsweise die Anforderungsspezifikation besteht aus Prosatexten ergänzt durch frei gestaltbare UML-Diagramme und Tabellen. Es enthält nur die Anforderungen beziehungsweise die Wünsche des Anwenders. Das Lastenheft entspricht im Grunde einer Weihnachtswunschliste. Die Anforderungen werden in funktionale und nicht-funktionale Anforderungen geteilt. Manche Anforderungsspezifikationen enthalten auch Datenanforderungen. Es gibt eine Regel dafür, wie man Anforderungen formulieren sollte, zum Beispiel im aktiven Modus und in kurzen Sätzen mit Subjekt, Objekt und Prädikat. Die Stilregeln für die Anforderungsbeschreibung stammen aus dem Volere-Modell von Chris und Susanne Robertson. Chris Rupp hat diese Regeln übernommen und ins Deutsche übersetzt [Rupp07]. Es gibt inzwischen Werkzeuge, mit denen die Anforderungen in Tabellen erfasst und verwaltet werden, zum Beispiel Doors, RequisitePro und HPs ALM. Diese Werkzeuge haben den Vorteil, dass die Anforderungen leichter zu ändern und fortzuschreiben sind [DaHB15]. Das endgültige Anforderungsdokument kann aus der Datenbank jederzeit generiert werden.

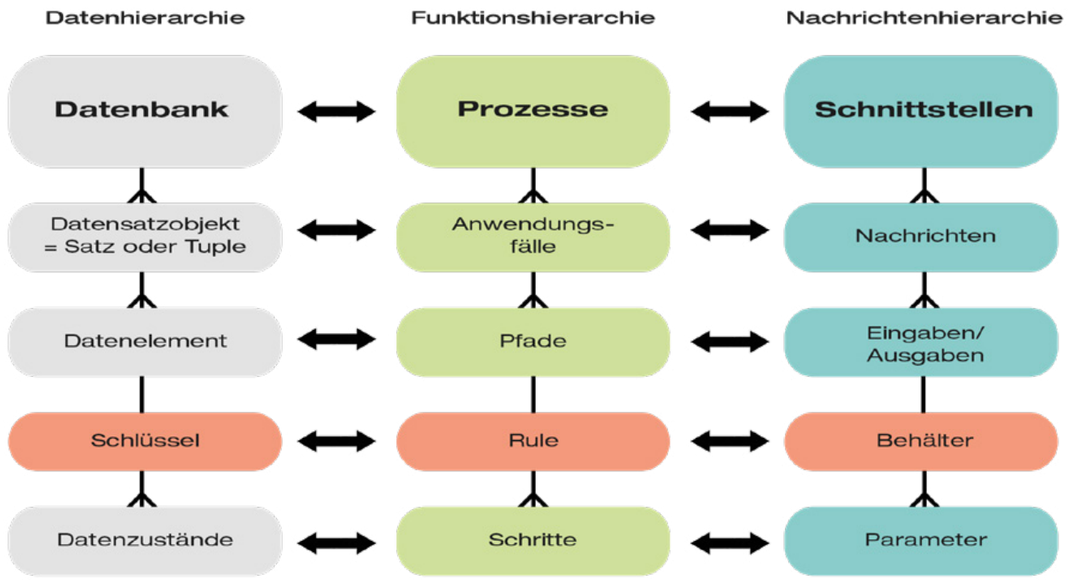

Abb. 2: Kommunikationsmodell

Das Pflichtenheft

Pflichtenhefte beschreiben fachliche Lösungskonzepte. Sie beschreiben die Struktur der Daten beziehungsweise das Datenmodell und die Verarbeitung der Daten beziehungsweise das Funktionsmodell sowie die Interaktion des Systems mit seiner Umgebung beziehungsweise das Kommunikationsmodell. In einem Pflichtenheft sollten alle drei Modelltypen vorkommen, Daten-, Funktions- und Kommunikationsmodell (siehe Abbildung 2). Das sind die Grundmodelle, die erforderlich sind, um den Projektaufwand zu schätzen. Außerdem können drei weitere Modelltypen wahlweise hinzukommen, um das Anforderungsdokument zu ergänzen:

- Rollenmodell,
- Regelmodell und
- Testmodell.

Das Datenmodell

Die Daten selbst werden in einem Datenmodell (Objekt- und/oder Klassendiagramm) beschrieben. Hier sind die Datenentitäten samt Datenattribute und Datenbeziehungen im Sinne des Entity-Relationship-Modells aufzuführen. In der Regel wird das Datenmodell grafisch abgebildet. Zusätzlich sollten die Datenentitäten und deren Schlüssel und Attribute in Tabellen erfasst werden. Die Tabellen können von inem Textanalysewerkzeug verarbeitet werden, die grafischen Darstellungen jedoch nicht. Je vollständiger das Datenmodell ist, desto zuverlässiger wird die datenbezogene Aufwandsschätzung. Manche IT-Systeme lassen sich ausschließlich anhand ihres Datenmodells kalkulieren (siehe dazu die Data-Point-Methode). Das Datenmodell schildert die logischen Datenobjekte und deren Schlüssel und Attribute sowie deren Wertbereiche und Beziehungen zueinander.

Das Funktionsmodell

Die Verarbeitung der Daten wird nach Zeit und Raum geordnet. Einzelne Ereignisse, die zur gleichen Zeit an einem Ort stattfinden und ein geschlossenes Ergebnis liefern, werden als Anwendungsfälle beziehungsweise Use Cases bezeichnet [JaSK16]. Ein Anwendungsfall entspricht dem, was früher als Vorgang beziehungsweise Geschäftsvorfall bezeichnet wurde. Er wird durch ein bestimmtes Ereignis oder einen bestimmten Zustand ausgelöst. Er hat einen vordefinierten Anfang – seinen Vorzustand (Precondition) – und ein vordefinierbares Ende – seinen Nachzustand (Postcondition). Dazwischen verwandelt er bestehende Datenentitäten und erzeugt neue nach vorgegebenen Regeln (Rules). Ein Anwendungsfall hat einen oder mehrere Pfade beziehungsweise Wege durch die Anwendungslogik. Ein Pfad ist der Hauptpfad, der zum gewünschten Ergebnis führt. Die anderen Pfade sind Alternativpfade. Es kann vorkommen, dass etwas schief geht und ein Pfad nicht weiter begehbar ist. In dem Fall tritt eine Ausnahmebedingung (Exception Condition) auf. Je vollständiger das Funktionsmodell ist, desto zuverlässiger wird die funktionsbezogene Aufwandsschätzung (siehe Function-Point- und UseCase-Point-Methoden).

Das Kommunikationsmodell

Anwendungsfälle kommunizieren miteinander und mit anderen Systemen über Schnittstellen. Schnittstellen zwischen Anwendungsfällen im gleichen Prozess gelten als interne Schnittstellen. Schnittstellen zwischen Anwendungsfällen in fremden Prozessen gelten als externe Schnittstellen. Eine Schnittstelle entspricht einer Menge gleichartiger Nachrichten. Nachrichten beinhalten einzelne Daten und Datengruppen. Sie werden wie ein Brief von einer Entität zur anderen gesendet. Der Sender packt die Daten in eine Nachricht ein. Der Empfänger packt die Daten aus. In dem Kommunikationsmodell werden die Nachrichtenströme beziehungsweise die Schnittstellen abgebildet. Das Kommunikationsmodell verbindet das Datenmodell mit dem Funktionsmodell. IT-Systeme, die kommunikationsbetont sind, lassen sich am besten über das Kommunikationsmodell Object-Points schätzen [SnBa11].

Function-Points	4.656
Data-Points	30.392
Object-Points	16.450
UseCase-Points	2.686
Test-Points	4.933

FALL für Anwendungsfall	REGEL für Rule	TRIG für Trigger/Auslöser
OBJEKT für Datenobjekt	MASK für Oberfläche	POST für Nachbedingung
ATTR für Datenattribut	PARM für Parameter	PRE für Vorbedingung
NACH für Nachricht	ACT für Akteuer	PATH für Pfad

Tabelle 3: Umfang einer Anforderungsdokumentation

Modellbasierte Schätzverfahren

Für die Aufwandsschätzung eines Neuentwicklungsprojektes ist das Anforderungs-Dokument (Lastenheft) zu wenig. Damit wird nur das Problem, aber nicht die Lösung geschätzt. Zu jedem Problem gibt es zahlreiche mögliche Lösungen. Es gilt, den Aufwand für die jeweilige Lösung zu schätzen. Deshalb muss auf Basis des Pflichtenheftes geschätzt werden, es sei denn, man setzt eine Standardlösung voraus. Mit der Analyse des Pflichtenheftes werden die rohen Größen gezählt. Aus der Anforderungsdokumentation mit den 445 Seiten ergaben sich die Zahlen in Tabelle 3 . Über die Zuverlässigkeit solcher Zahlen lässt sich diskutieren (Unsicherheitsraum 1 bis 4 nach oben und unten [Cohn03]). Sie bieten jedoch einen Ausgangspunkt, der immerhin besser als gar nichts ist. Die Alternative wäre der Vergleich mit einem bereits implementierten System, das dem jetzigen gleichkommt. Dem zuständigen Schätzer bleibt nur die Möglichkeit, zu zählen oder zu vergleichen. Was hier in zehn Tagen gezählt wird, würde durch eine manuelle Zählung mehrere Perso-nenmonate kosten. Und ob die Qualität der Zählung dadurch viel besser ist, bleibt auch zu bezweifeln. Tom Gilb attestiert: Jeder quantitative Messwert ist besser als gar kein Messwert [Gilb88].

Auszeichnungsschlüsselwörter

Im Mittelpunkt der Requirements Reengineering steht das Einfügen von Schlüsselwörtern in den Requirementstext. Die Schlüsselwörter = Keywords in Context entsprechen den Tags in einem XML-Dokument [Parn78]. Sie dienen als Beacons, um den Text mit dem dahinterliegenden Modell zu verbinden. Sie werden jedoch nicht mit Sonderzeichen wie < > gekennzeichnet. Sie stehen als erstes Wort in einer Zeile. Wenn der Textparser ein solches Wort findet, das mit einem Leerzeichen beendet wird, erkennt er, dass der darauffolgende Satz bis zum nächsten Satzendezeichen, zum Beispiel: „.“, „!“, „?“ oder „:“, dazugehört. Dadurch bleibt der Text in seiner ursprünglichen Form möglichst unverändert. Dennoch sind die Satzarten anhand der Schlüsselwörter zu erkennen. Der Benutzer kann diese Wörter selbst vergeben. Er muss sie nur in der Schlüsselworttabelle den festen Schlüsseln zuweisen. Beispiele von Schlüsselwörtern zeigt Tabelle 4 . Diese Schlüsselwörter sind frei wählbar, sie müssen nur den internen Schlüsseln in einer Key-Word-Tabelle zugeordnet werden. Sie sind mit Tags in XML vergleichbar. Die internen Schlüssel sind im Textparser fest verdrahtet und können nicht verändert werden. Sie sind nach außen unsichtbar. Die Benutzer der Anforderungsdokumentation bekommen nur die benutzerdefinierten Namen beziehungsweise die Tags zu sehen.

Ausschnitt aus einem echten Lastenheft

Anforderung 1: RESTful Webservice

&ANF-1: RESTful Webservice	
Priorität	Hoch
Beschreibung	Der Auftraggeber benötigt ein RESTful Webservice mit zwei Funktionen je Filiale.

Anforderung 11: Verkettung

&ANF-11: Verkettung	
Priorität	Hoch
Beschreibung	Verkettung der Tages-DEPs von einer bis N Kassen zu einem Gesamt-DEP pro Filiale.

Anforderung 12: Prüfung

&ANF-12: Prüfung	
Priorität	Hoch
Beschreibung	Prüfung der zusammengeführten DEPs pro Filiale.

Ausschnitt aus einem echten Pflichtenheft

&AFall_1 Verarbeitung eines Standardbelegs

Eingabewerte

Eingabewerte für Felder laut Prozess 2.1, die aus Belegdaten aufbereitet werden:
&EING_1 Beleg_Satz_Normal
&EING_2 Beleg_Satz_Ermaessigt_1 &EING_3 Beleg_Satz_Ermaessigt_2 &EING_4 Beleg_Satz_Null
&EING_5 Beleg_Satz_Besonders

Eingabewerte, die aus der Kasse extrahiert werden:
Felder laut Prozess 2.1:
&EING_6 Kassen_ID
&EING_7 Belegnummer
&EING_8 Beleg_Datum_Uhrzeit
&EING_9 Zertifikat_Seriennummer Weitere Daten:
&EING_10 Umsatzzähler_der_Kasse &EING_11 AES_Schlüssel_der_Kasse

Prozessbeschreibung

Dieser Prozess ist für die Erstellung von Standardbelegen relevant. Werte für Felder laut Prozess 2.1, die berechnet werden müssen:
&AUSG_1 Stand_Umsatz_Zaehler_AES256_ICM
&AUSG_2 Sig_Voriger_Beleg

Es werden nun folgende Abläufe durchgeführt:
&PFAD_1 Aufbereiten/Aktualisieren/Verschlüsseln_des_Umsatzzählers:
siehe Ablauf 2.5.
Als Ergebnis der dort beschriebenen Detailprozesse erhält man den Wert des Felds
Stand_Umsatz_Zaehler_AES256_ICM
&PFAD_2 Berechnung_des_Verkettungswerts:
Für den Verkettungswert wird der vorhergehende Beleg herangezogen. Siehe Ablauf 2.4.2: Als Ergebnis erhält man den Wert des Felds Sig_Voriger_Beleg.

Literatur & Links

[Abra15] A. Abran, Software Project Estimation:
The Fundamentals for Providing High Quality Information to Decision Makers, John Wiley & Sons, IEEE Computer Society Press, 2015
[Berg15] J. Bergsmann, Requirements Engineering für agile Softwareentwicklung, dpunkt.verlag, Heidelberg, 2015
[Bria17] L. Briand, Analyzing Natural Language Requirements – The not-too-sexy and yet curiously difficult Research that Industry needs, in:
23rd Int. Working Conf. on Requirements Eng., Foundation for Software Quality (REFSQ 17), 2017
[Bund15] IT-Beauftragter der Bundesregierung, V-Modell XT, CIO Bund, 2015, siehe:
https://www.cio.bund.de/Web/DE/Architekturen-und-Standards/V-Modell-XT/vmodell_xt_node.html
[Chen79] P. Chen, Entity-Relationship Approach to Systems Analysis and Design, North-Holland Publishing Company, 1979
[Cohn03] M. Cohn, Agile Estimating and Planning, Robert Martin Series, Prentice-Hall, 2003
[DaHB15] M. Daneva, A. Herrmann, L. Buglione, Coping with Quality Requirements in Large, Contract-based Projects, in:
IEEE Software Magazine, Nov. 2015, s. 84
[Eber12] C. Ebert, Anforderungen ermitteln – Zehn Schritte für die Praxis, in: OBJEKTspektrum, 6/2012
[FeHW16] H. Femmer, B. Hauptmann, A. Widera, Requirements-Smells – Automatische Unterstützung bei der Qualitätssicherung von Anforderungs-dokumenten, in: OBJEKTspektrum, 2/2016
[Gilb88] T. Gilb, Principles of Software Engineering Management, Addison-Wesley, 1988, S. 59
[HöHö08] R. Höhn, S. Höppner, Das V-Modell XT, Springer Verlag, 2008
[IEEE83] ANSI/IEEE Standard 830 – Guide to Requirements Specifications, Computer Society Presse, 1983
[ISO11] ISO/IEC/IEEE 29148:2011, Systems and software engineering – Life Ccycle processes – Requirements engineering, siehe:
https://www.iso.org/standard/45171.html
[JaSK16] I. Jacobson, I. Spence, B. Kerr, Use-Case 2.0, in: Comm. of ACM, Vol. 59, No. 5, May 2015, S. 61
[Parn78] D. L. Parnas, K. Heninger, D. Shore, Software Requirements for the A-7E Aircraft, Naval Research Report 3876, Washington D.C.,
1978 [Pohl07] K. Pohl, Requirements Engineering, dpunkt.verlag, 2007
[PoRu15] K. Pohl, C. Rupp, Basiswissen Requirements Engineering, dpunkt.verlag, 2015
[RoRo99] S. Robertson, J. Robertson, Mastering the Requirements Process, Addison-Wesley, 1999
[Rupp07] C. Rupp u. Sophisten, Requirements-Engineering und Management, Hanser Verlag, 2007
[SnBa11] H. Sneed, M. Baumgartner, Software in Zahlen, Hanser Verlag, 2011

Fazit

Die Sanierung von Anforderungen und Anforderungsdokumenten, kurz Requirements Reengineering, durch Require-ment-Ingenieure ist eine schon seit Jahren erfolgreich praktizierte Dienstleistung, um Anforderungsdokumente vor der Ausschreibung beziehungsweise der Implementierung, aber auch immer öfter im Nachhinein bei Streitigkeiten zu prüfen, zu vermessen und zu bewerten. Dazu werden die Anforderungsdokumente mittels Markups in einem für die Benutzer wirtschaftlich vertretbaren Aufwand aufgezeichnet. Die Markierung findet über die Editierung der Textdokumente statt. Der Text wird umstrukturiert und mit Schlüsselwörtern markiert, um das dahinterliegende Modell kenntlich zu machen. Erst danach lassen sich die Texte automatisch analysieren und die Qualität des dahinterliegenden Modells formal prüfen. Die Anforderungsdokumentation soll schließlich als Testorakel in „Requirementbased Testing“ dienen. Dazu muss sie maschinell verarbeitungs- und fortschreibungsfähig sein. Dies ist auch die Voraussetzung für eine systematische Aufwandsschätzung. Die Modellelemente müssen sich erkennen und zählen lassen [Abra15]. Für die zuständigen Fachbereiche lohnt es sich auf jeden Fall, ihre Anforderungskonzepte auf diese Art und Weise zu bereinigen und zu strukturieren (Refactoring), ehe sie mit der Umsetzung des Projektes beginnen.

DIE AUTOREN

DI Dr.tech Wolfgang Prentner
IT-Ziviltechniker . Informatiker . CEO ZTP.digital
prentner@ztp.digital . www.ztp.digital

ist seit 1998 IT-Ziviltechniker im Fachbereich Informationstechnologie. Außerdem ist er Geschäftsführer der ZTP.digital, Gerichtssachverständiger und promovierter Informatiker an der TU Wien. Als unabhängige Prüf- und Überwachungsstelle für Informatik, CyberSecurity, Datenschutz und dem CYBERBELT unterstützt Dr. Prentner in ehrenamtlicher Funktion die Länderkammer, die Bundeskammer und das Bundeskomitee Die Freien Berufe Österreichs sowie das österreichische Bundeskanzleramt seit 2004.

Harry M. Sneed
Consultant
harry.sneed@t-online.de . www.ztp.digital

ist zurzeit Software-Tester, -Vermesser und -Reengineer bei ZTP.digital in Wien. Nebenbei lehrt er Software-Rengineering an der TU Dresden sowie Software-Evolution und Softwaremessung an den Fachhochschulen Hagenberg und Wien. Er erhielt den Stevens Award von der IEEE 2009 für seinen Beitrag zur Software-Maintenance, den ersten Preis für Qualitätssicherung von der ASQF 2011 und die Auszeichnung als internationaler Tester des Jahres von der ISTQB im Jahre 2013.