Hajo Hoffmann

STABLE - Ein Ansatz zur systematischen Strukturierung von Anforderungen

Motivation

In vielen Branchen der Systementwicklung mit einer ausgeprägten Auftraggeber- und Auftragnehmer-Trennung haben in natürlicher Sprache verfasste Spezifikationen eine zentrale Position zur Festlegung und Abstimmung der zu entwickelnden Umfänge.

Bei großen Projekten entwickelt sich allein aufgrund der schieren Menge an beteiligten Teams, Abteilungen und zu berücksichtigenden Normen eine heterogene Spezifikationslandschaft. Zeitdruck, persönlicher Stil, variierende fachliche Rahmenbedingungen und zu ungenaue oder sich widersprechende Standards tun ihr übriges, dass die für einzelne oft unüberschaubare Anzahl entstehender Dokumenten an Inkonsistenz, schwankender Qualität und insbesondere verwirrend verschiedener Strukturierung leidet.

STABLE soll genau dort Abhilfe schaffen, wo bisherige Spezifikationsvorlagen konzeptuelle Lücken aufweisen: bei der systematischen Strukturierung der funktionalen Anforderungen. Anhand des STABLERegelwerkes können Spezifikationslandschaften sauberer strukturiert – und damit besser, lesbarer und effizienter werden.

Stand der Technik

Viele Standardgliederungen für Spezifikationen (z.B. [1][2][3]) geben einen detailliert ausgearbeiteten, möglichen Rahmen für viele relevante Details einer Systemspezifikation vor. Da sie aber so allgemeingültig wie möglich sein sollen, wird stets darauf verzichtet, eine Anleitung zur feineren Strukturierung der funktionalen Anforderungen zu machen. Diese ist prinzipiell zu abhängig von den fachlichen Gegebe heiten, um diesbezüglich Vorgaben machen zu können, die inhaltsunabhängig gelten. Man könnte dieses Dilemma als den gordischen Knoten des Informationszeitalters bezeichnen: die bislang ungelöste Aufgabe, ein allgemeingültiges Prinzip zur Strukturierung beliebiger Inhalte zu finden. Zwar werden häufig präzise, Unternehmens- oder sogar projektspezifische Regeln zur Ordnung der Spezifikationslandschaft oder zur groben Gliederung von Spezifikationsartefakten erstellt. Aber bezüglich der einzelnen Anforderungen, beschränken sich die Vorgaben zumeist auf ein lapidar ausgewiesenes Kapitel für alle funktionalen Anforderungen. Die Gliederung dieser meist das Herzstück der Spezifikation darstellenden Inhalte wird daher allein den Autoren überlassen (siehe Abbildung 1).

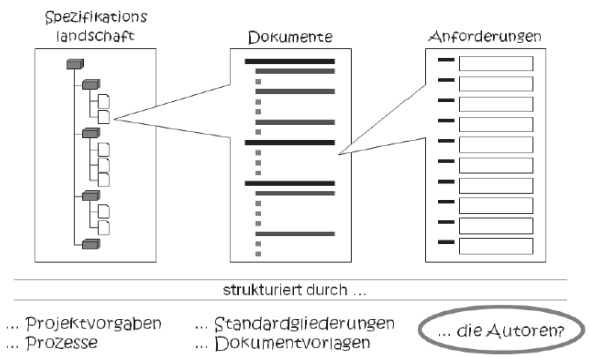

Abbildung 1: Der menschliche Faktor als Ursache für heterogene Spezifikationslandschaften

Mögliche Folgen heterogener Spezifikationen und Spezifikationslandschaften

Es gibt eine Reihe von möglichen Nachteilen und Problemen, die sich aus dem Fehlen einer Systematik zur Strukturierung von Anforderungen ergeben. Zum Beispiel fängt jeder Autor eines neuen Dokumentes bezüglich der Anforderungsgliederung immer bei null an – oder er orientiert sich an anderen Spezifikationen, deren abweichende fachliche Ausrichtung keine sinnvoll nutzbaren Hinweise bezüglicher Struktur geben kann. Weiterhin hat jeder Autor aufgrund seiner persönlichen Erfahrungen, Kultur- und Bildungshintergründe eine eigene Vorstellung davon, wie sich ein spezifischer fachlicher Sachverhalt am besten gliedern lässt. Da Menschen vergesslich sind, kommt es oft vor, dass wenn der selbe Autor in der selben Spezifikation zu verschiedenen Zeitpunkten Anforderungen gliedert, dass sich sein Prinzip der Gliederung verändert – die Spezifikation demnach sogar in sich selbst heterogen strukturiert ist. Wenn mehrere Autoren die selbe Spezifikation bearbeiten, kommt es noch häufiger vor, dass verschiedene Vorgehensweisen zur Gliederung angewandt werden. Ohne zentrale Verantwortung für die Struktur ist eine inkonsistente Gliederung quasi garantiert. Eine Folge inkonsistenter Gliederungen ist die erschwerte Lesbarkeit eines Dokumentes. Die Leser können sich nicht in ein systematisches Gliederungskonzept eindenken und fragen sich ständig wo sie denn die gesuchten Inhalte finden können. Dies erzeugt potenziell Frust, Missverständnisse und Irrtümer. Aber nicht nur inkonsistente sondern auch einfach ungewohnte Gliederungen können die Lesbarkeit erschweren. Menschen denken nun einmal verschieden – und was dem einen als sinnvoll und eindeutig strukturiert vorkommt kann den anderen heillos verwirren. Erschwert werden die Mängel inkonsistenter oder verwirrender Gliederungen, wenn sie sich über verschiedene Dokumente hinziehen. Leser, die sich in einem Dokument zurechtfinden, müssen in anderen Dokumenten wieder umdenken. Aber nicht nur für Leser sind inkonsistente oder verwirrende Gliederungen von Nachteil. Falls Autoren, die neue Inhalte in eine Spezifikation einfügen möchten, sich angesichts der mangelhaften Struktur nicht zurechtfinden, kann dies zu „Geschwürbildung“ führen. Damit ist der selbstverstärkende Effekt gemeint, der dadurch entsteht, dass neue Inhalte an eine ungeeignete Position in der Gliederung eingefügt werden, die Gliederung dadurch an dieser Stelle mehrdeutig wird und für weitere Neuinhalte geeigneter erscheint – nach dem Motto „wenn jener Sachverhalt dort steht, dann kann dieser neue Sachverhalt auch dort beschrieben werden“. Die durch mangelnde Systematik hervorgerufene, ausufernde Schwammigkeit solcher Gliederungen kann sogar anfänglich sauber strukturierte Strukturen nachhaltig zerrütten. Diese und die sich daraus ergebenden Nachteile, sorgen dafür, dass heterogene Spezifikationsstrukturen sich negativ auf die Effizienz im Umgang mit Spezifikationsdokumenten auswirken. Denn die in umfangreichen Spezifikationslandschaften verborgenen Informationen sind desto schwerer zu erfassen, je mehr Steine den Lesern durch verwirrende Strukturen in den Weg gelegt werden.

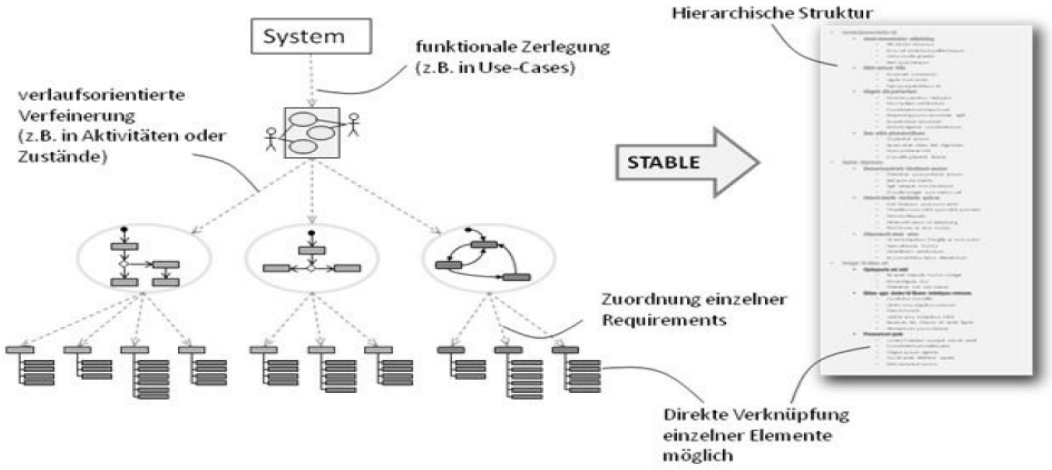

Abbildung 2: Der STABLE-Ansatz zur Erzeugung einer Gliederungsstruktur aus den Ergebnissen der Use-Case-Analyse

Lösung

Um der fehlenden Systematik zu begegnen haben wir das STABLE-Regelwerk entwickelt. Dieses basiert auf der Use-Case-Analyse [4]. Bei diesem Vorgehen zur Systemanalyse wird das betrachtete (Teil)System von grob nach fein in verhaltensorientierte Elemente zerlegt. Begonnen wird dabei mit einer Sammlung von Use-Cases, die die Systemfunktionalität aus Benutzersicht abstrakt beschreiben. Diese werden in Zuge der weiteren Analyse immer weiter verfeinert, in feinere Use-Cases, Aktivitäten oder Zustände. Häufig werden die Ergebnisse dieser Analyse in Form von UML-Diagrammen dokumentiert. Dieses Vorgehen ist auf der linken Seite von Abbildung 2 schematisch dargestellt. Das dem STABLE-Regelwerk zugrunde liegende Prinzip wurde von uns und unseren Kollegen zwar schon häufig in Projekten angewandt und in Schulungen vermittelt, aber es fehlte bislang eine dokumentierte Form der konkreten Vorgehensweise zur Erzeugung einer hierarchischen Gliederung auf Basis der verhaltensorientierten Zerlegung. Einfach formuliert, werden entsprechend des STABLE-Regelwerkes die im Rahmen der Use-Case- Analyse erzeigten Modelle und Artefakte fast 1:1 in eine entsprechende Gliederung überführt, welche dann die Kapitelstruktur für die natürlichsprachigen Anforderungen darstellt. Wichtig war uns, dass das STABLE-Regelwerk deterministisch und iterativ aufgebaut ist. Alle Entscheidungen, wo welches Kapitel oder Unterkapitel angelegt werden muss, lassen sich eindeutig anhand der einzelnen Regeln fällen. Da bei der Use-Case- Analyse viele verschiedene Kombinationen von Diagrammen und Notationselementen zur Anwendung kommen können, sind die meisten Regeln durch vorangestellte Bedingungen eingeschränkt. Zum Beispiel gelten, je nachdem, ob ein Use-Case- Diagramm einen anderen Use-Case verfeinert, oder ein System auf oberster Ebene gliedert, andere Regeln bezüglich der Einordnung der Use-Cases in die Kapitelstruktur.

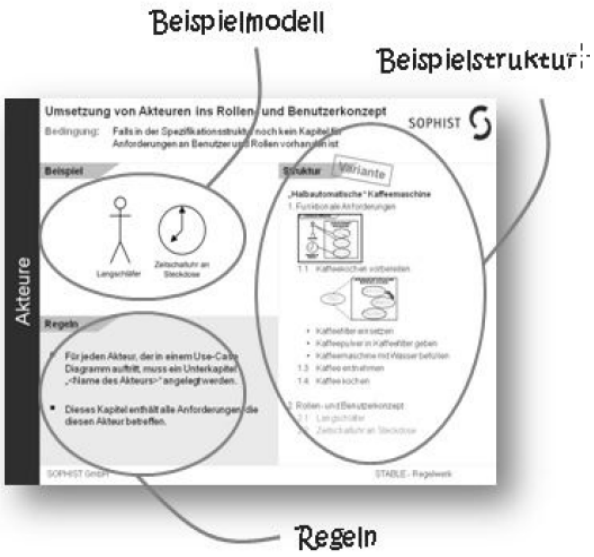

Abbildung 3: Die Dokumentation des STABLERegelwerkes

Da Verfeinerungen prinzipiell beliebig tief verschachtelt sein können, sind einige Regeln der STABLE-Systematik rekursiv aufgebaut – sie verweisen also auf andere Regeln, die letztendlich wieder zu auf sie selbst verweisen. Zur Dokumentation des Regelwerkes haben wir Präsentationsfolien entworfen, die neben den eigentlichen Regeln stets ein UML-Model oder einen Ausschnitt eines UML-Models nebst einer sich kontinuierlich vervollständigenden Beispielgliederung enthalten (siehe Abbildung 3). In der ersten Version des Regelwerkes wurden nur die für die Use-Case-Analyse relevanten UMLDiagramme betrachtet (sprich: Use-Case-Diagramm, Aktivitätsdiagramm, Zustandsautomat). Auch haben wir uns bei den betrachteten Notationselementen auf die in der Systemanalyse gebräuchlichsten beschränkt. Die Einsatzmöglichkeiten des STABLE-Regelwerkes sind vielfältig. Es kann beispielsweise als Handlungsanweisung für Anforderungsanalytiker fungieren, die anhand der Regeln systematisch Gliederungen erzeugen möchten. an die Hand gegeben werden, um systematisch Gliederungen für Anforderungen zu erzeugen. Ein weiterer Zweck von STABLE ist die sich daraus ergebende Möglichkeit zur sinnvollen Integration von UML-Modellen in die natürlichsprachigen Spezifikationen. Denn die aus den Diagrammen erzeugte Gliederung stellt eine enge Verzahnung der Modelle und der natürlichen Sprache dar.

Fazit

Das STABLE-Regelwerk ist ein probates Mittel, die Homogenität von Spezifikationslandschaften zu födern. Es eignet sich besonders für Projekte, in denen die Use-Case-Analyse bereits angewandt wird. In Projekten, in denen bislang rein natürlichsprachig spezifiziert wird, die Einführung der modellbasierten Systemanalyse aber angedacht wird, kann STABLE einen sinnvollen Ansatz zur Integration beider Dokumentationsformen bieten und damit weitere Argumente für die Einführung der formalen, grafischen Modellierung in die Analyse liefern. Es lohnt sich, denn mit STABLE **ST**rukturieren Sie **A**nforderungen **B**esser, **L**esbarer und **E**ffizienter.

Quellen

[1] VOLERE - Robertson S., Robertson J.: Mastering the Requirements Process. 2nd Edition. Pearson 2006

[2] V-Modell XT - Beispielprojekt WiBe: Bundesrepublik Deutschland. KBSt 2004, http:/kbst.bund.de

[3] IEEE Standards Board: IEEE Std 830-1998. IEEE Recommended Practice for Software Requirements Sepcifications. IEEE Press, 1998.

[4] Cockburn A.: Agile Software-Entwicklung. Die Prinzipien der agilen Software-Entwicklung dargestellt und erläutert. The MIT Press, 2003