Den Anforderungsstall ausmisten

Prinzipien der Einfachheit beim Requirements Engineering

Matthias Moll, Helbling Technik GmbH

Viele bedeutende Geister haben Einfachheit als Erfolgsprinzip erkannt, und auch heute liegt das Thema im Trend. Gerade im komplexen Gebiet der Anforderungsentwicklung mit ihren vielen Akteuren, Einflussfaktoren und komplexen Systemeigenschaften bietet Einfachheit großes Potential für Effizienzgewinne in der Systementwicklung. Einfachheit ist jedoch nicht einfach zu erreichen. Im Kontext der Anforderungsentwicklung lassen sich jedoch Grundprinzipien identifizieren, mit deren Hilfe qualifizierte Anforderungen entwickelt beziehungsweise gegebene Anforderungssammlungen vereinfacht werden können.

Einfachheit in der Anforderungsentwicklung

Entwickler von Produkten und Systemen sind häufig mit Komplexität konfrontiert. Aus Komplexität entstehen jedoch Risiken, denn das menschliche Gehirn kann das dynamische Zusammenspiel von Strukturen und Wirkzusammenhängen nur in begrenztem Umfang überschauen. Standardisierte Prozessmodelle und Entwicklungsmethoden adressieren dieses Problem und versuchen, die Komplexität von Entwicklungsprojekten und dynamischen Systemen in kleinere, einfachere und damit beherrschbare Einheiten aufzuteilen.

Der Anforderungsentwicklung kommt dabei eine besondere Bedeutung zu. Anforderungen stehen nicht nur am Beginn der Systementwicklung und determinieren damit in hohem Maße die Komplexität des zu entwickelnden Produkts, sondern bilden insbesondere auch die zentrale Schnittstelle zum Auftraggeber. In dieser Funktion entscheiden Anforderungen ganz wesentlich über den Erfolg oder Misserfolg der Systementwicklung, denn qualifizierte Anforderungen sind die „Definition of Done“ des Projekts.

Einfachheit in der Anforderungsentwicklung bietet in diesem Zusammenhang das Potential, die Komplexität der Systementwicklung und damit Budget- und Terminrisiken gleich in der frühen Projektphase zu reduzieren. Die Praxis zeigt jedoch, dass Anforderungen diesem Paradigma der Einfachheit oft nicht genügen. Dies führt zu Missverständnissen und langwierigen Abstimmungsprozessen, sowie einem erhöhten Aufwand für Änderungsmanagement, Dokumentation und Systemtest. Hingegen verbessert eine qualifizierte Anforderungsentwicklung die Qualität des Produkts, ermöglicht größere Freiheiten bei der Lösungsfindung, führt zu mehr Harmonie in der Kundenbeziehung und fördert ganz allgemein den Spaß an der Entwicklungszusammenarbeit.

Konzentration auf das Wesentliche

Im ersten Schritt lassen sich Anforderungen vereinfachen, indem Vorlagen zurechtgestutzt, die Dokumentenstruktur vereinfacht und nicht benötigte Inhalte weggelassen werden. Vorlagen für Anforderungen beinhalten in ihrer vorgegebenen Kapitelstruktur oft eine Übermenge aller Aspekte, die möglicherweise eine Rolle spielen könnten, die aber größtenteils für das konkrete Projekt irrelevant sind. Außerdem beobachtet man oft eine Überfrachtung mit Projektvorgaben und -informationen, die

keinen Einfluss auf das Produkt haben. An dieser Stelle lassen sich erste Maßnahmen zur Vereinfachung ergreifen:

- Verzichten Sie auf eine Änderungshistorie im Anforderungsdokument, wenn dieses in einem Konfigurationsmanagement-Tool verwaltet wird. Diese Tools beinhalten eine Versionsverwaltung inklusive Änderungshistorie. Wird diese genutzt, müssen Änderungen nicht zusätzlich im Dokument verwaltet werden
- Verzichten Sie auf Projektinformationen und -anforderungen, betriebswirtschaftlichen Aspekte sowie Anforderungen außerhalb des betrachteten Systemumfangs
- Interpretieren Sie Normen und Standards, anstatt pauschal auf die entsprechende Norm zu verweisen. Extrahieren Sie diejenigen Aspekte, die für das vorliegende System relevant sind, und formulieren Sie die Normvorgabe als konkrete, funktionale Systemanforderung
- Strukturieren Sie nach Schnittstellen und verzichten Sie auf eine übergeordnete Trennung zwischen funktionalen und nicht-funktionalen Anforderungen. Operationalisieren Sie nicht-funktionale Anforderungen, d.h. interpretieren Sie die geforderte Systemeigenschaft als konkrete, funktionale Systemanforderung
- Löschen Sie Ausfüllhilfen, Vorlagentexte und nicht benötigte Abschnitte, anstatt sie beispielsweise mit „not applicable“ zu markieren

Nur die Schnittstellen zählen

Überflüssige Komplexität in den Anforderungen ergibt sich häufig aus der fehlenden Trennung zwischen dem geforderten Verhalten an den Schnittstellen des betrachteten Systems und Festlegungen in Bezug auf die Lösung. Um dies zu vermeiden, sollte konsequent ein Black-Box-Ansatz verfolgt werden.

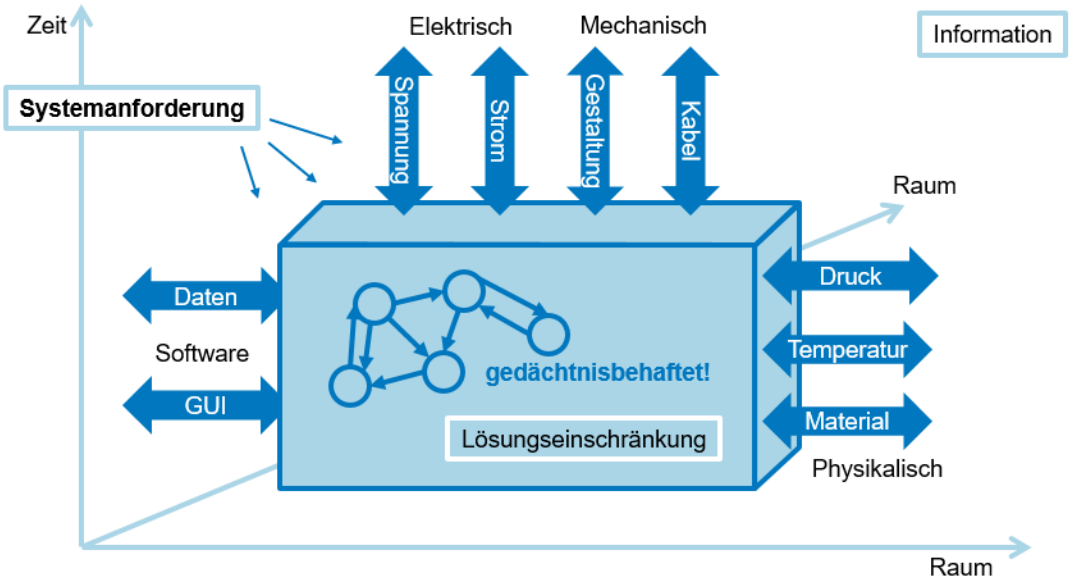

Abb. 1: Black-Box-Ansatz: Systemgrenzen, Schnittstellen und interne Zustände

In Abb. *1* ist der Black-Box-Ansatz bildlich dargestellt. Die wesentlichen Merkmale des Black-Box-Ansatzes sind Systemgrenzen, Schnittstellen und interne Zustände. Für eine qualifizierte Anforderungsentwicklung ist es wichtig, diese Merkmale in Bezug auf das betrachtete System vollständig und eindeutig zu klären, bevor die eigentlichen Anforderungen formuliert werden. Dabei ist zu beachten:

Systemgrenzen und Schnittstellen sollten sich auf die betrachtete Systemebene beziehen, und zwar explizit ohne die interne Systemarchitektur zu berücksichtigen. Eine interne Schnittstelle zwischen Komponenten der Systemarchitektur (Teilsysteme) ist aus Sicht des Gesamtsystems bereits Teil der Lösung, wie Abb. *2* veranschaulicht:

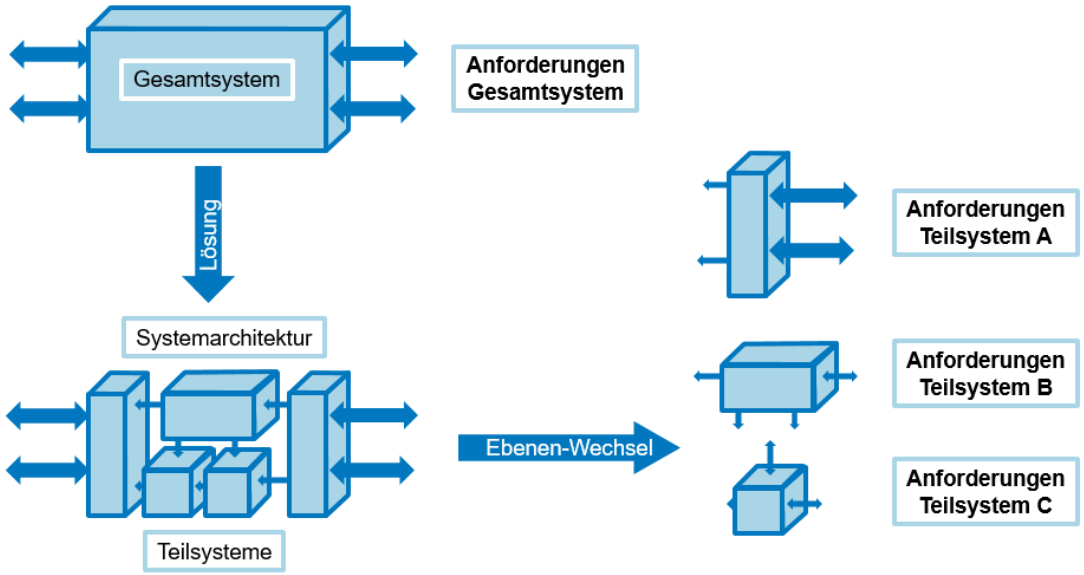

Abb. 2: Anforderungsebenen auf Systemarchitektur abbilden

Anforderungen an Teilsysteme könnten Teil der initialen Anforderungsdokumentation sein, wenn die Systemarchitektur zumindest teilweise vorbestimmt ist und eine klare Trennung zwischen den jeweils betrachteten Ebenen gewährleistet ist.

Anforderungen sind abhängig von internen Systemzuständen. Interne Systemzustände sind im Sinne der Systemtheorie das Ergebnis eines Ausgangszustands und einer Abfolge von Signalen an den Eingangsschnittstellen. Damit sind in dynamischen Systemen die Anforderungen auch immer abhängig von Ereignissen in der Vergangenheit. Das System wird damit gedächtnisbehaftet. Für eine klare Anforderungsentwicklung ist es wichtig, sich dessen bewusst zu sein und Betriebsmodi, Systemzustände, Daten und ähnliche, zeitlich veränderbare Systemeigenschaften mit einzubeziehen.

Ursache und Wirkung statt Zustand und Struktur

Systemgrenzen, Schnittstellen und interne Zustände sind nicht die Anforderungen selbst, sondern bilden die Basis für die einfache Formulierung von passenden Anforderungen. Die DNA jeder Anforderung ist dabei die Beschreibung einer gewünschten Auswirkung, die sich als Folge eines Ereignisses unter einer bestimmten Bedingung einstellen soll. Sind alle Systemschnittstellen und internen Zustände vollständig bestimmt (s.o.), lässt sich auch das gewünschte Systemverhalten prinzipiell vollständig definieren:

	Zustand A	Zustand B	Zustand C
Ereignis X	Auswirkung Neuer Zustand	…	…
Ereignis Y	…	…	…
Ereignis Z	…	…	…

Abb. 3: Übergangstabelle als Beschreibung des Systemverhaltens

Das System wird in diesem Sinne als Zustandsautomat aufgefasst, dessen Zustandsübergänge in ihrer Gesamtheit das Systemverhalten spezifizieren. Jede Zelle der Tabelle aus Abb. *3* kann als Anforderung formuliert oder im Sinne einer modellbasierten Anforderungsentwicklung dargestellt werden. Für zeitkontinuierliche oder zeitdiskrete Signalverarbeitung (z.B. in digitalen Reglern) werden die Anforderungen analytisch, d.h. über Differenzial- bzw. Differenzengleichungssysteme definiert. Auch hier entspricht die Anforderungsdefinition der Beschreibung eines Ursache-Wirkungs-Zusammenhangs, basierend auf Eingangssignalen und internen Zuständen, die sich abhängig vom zeitlichen Verlauf der Eingangssignale ändern.

Fazit

Einfachheit in der Anforderungsentwicklung bedeutet Vollständigkeit bei minimalem Umfang. Dies erreicht man durch die Konzentration auf die wesentlichen, das Systemverhalten bestimmenden Aspekte. Dabei zählt ausschließlich das Verhalten an den Schnittstellen, d.h. die gewünschte Auswirkung an den Ausgangsschnittstellen basierend auf den aktuellen und vergangenen Ereignissen an den Eingangsschnittstellen. Anforderungen sollten dabei immer als Zusammenhang zwischen Ursache und Wirkung beschrieben werden, anstatt lediglich den strukturellen Aufbau des Systems oder statische Systemzustände zu beschreiben.

Eine in diesem Sinne sorgfältige und gründliche Anforderungsanalyse in einer frühen Projektphase führt zu verbesserter Qualität bei verminderten Zeit- und Kostenrisiken.

Autor

Matthias Moll ist Leiter der Embedded Software Entwicklung bei der Helbling Technik GmbH in München. Nach seinem Studium der Informationssystemtechnik in Braunschweig arbeitete er 11 Jahre lang bei der Robert Bosch GmbH als Softwareentwickler und Software-Projektleiter, zunächst im Bereich Getriebesteuergeräte und später Brandmeldezentralen. Im Jahr 2016 wechselte er in die Geschäftsführung am Münchner Standort der Helbling Unternehmensgruppe. Sein Interessensschwerpunkt liegt auf der effizienten Entwicklung, Verwaltung und Implementierung von Anforderungen in wechselnden Technologie-, Branchen- und Projektlandschaften. Kontakt: matthias.moll@helbling.de