

Ausführbare Spezifikationen mit xUML

Mit modellbasierten Ansätzen effizient und ökonomisch zu korrekter Software: Agile Methoden haben durchaus ihre Berechtigung – wer dabei jedoch die Mächtigkeit richtig genutzter Modelle ausser Acht lässt, verpasst ein zentrales Element eines modernen Softwareprozesses.



Ausführbare Spezifikation eines Stellwerksystems im Bahnbereich aus Benutzersicht.

Präzise und lückenlose Pflichtenhefte sind in vielen Softwareprojekten noch immer eine Wunschvorstellung. Oft wird sehr früh mit der Teilrealisierung des geplanten Systems begonnen. Statt die Anforderungen und das konzeptionelle Systemdesign zu präzisieren, steckt man wertvolle Kapazitäten in technische Lösungen. Der Werkzeugkasten agiler Methoden mag zwar helfen, das Risiko von Fehlentwicklungen zu reduzieren, doch ab einer gewissen Projektgrösse stossen diese Ansätze an ihre Grenzen. Wer Software effizient und ökonomisch entwickeln will, kommt deshalb nicht um modellbasierte Ansätze herum.

Modellbasierte Ansätze

Modelle sind heute mehr als eine Sammlung grafischer Repräsentationen: Die dargestellten Informationen und Zusammenhänge lassen sich auch maschinell interpretieren und effizient im Entwicklungsprozess nutzen. Man trennt die Implementation von der Spezifikation und pflegt so kein grosses Modell, das schrittwei-

se bis zum Abbild des Codes erweitert wird. Im Software-Engineering haben sich zwei Modelle etabliert: Im plattform-unabhängigen Modell werden Anforderungen in Form einer lösungsneutralen Spezifikation formuliert. Das plattformspezifische Modell beschreibt die technische Umsetzung der Spezifikation auf der Zielplattform. Beide Modelle werden mit der Notation der UML (Unified Modeling Language) erstellt. Darauf basierend lassen sich viele Aufgaben im Entwicklungsprozess automatisieren – bis hin zur vollständigen Code-Generierung.

Lösungsneutrale Spezifikation

Ein wichtiges Bindeglied zwischen den Kundenanforderungen und der technischen Implementation ist die lösungsneutrale Spezifikation. Dabei handelt es sich um eine präzise und vollständige Beschreibung der Funktionalität des künftigen Systems. Klassendiagramme zeigen die fachlichen Objekte und ihre strukturellen Eigenschaften. Mittels Use-Case- und einfachen Sequenzdiagrammen

wird die Funktionalität definiert, die das System seinen Benutzern zur Verfügung stellt. Schliesslich lässt sich mit Zustandsdiagrammen das dynamische Verhalten der fachlichen Objekte spezifizieren. Es empfiehlt sich, die so genannte xUML (executable UML) zu verwenden – eine präzise Variante der UML, die nebst der grafischen Notation auch Gebrauch von den textuellen Erweiterungen OCL (Object Constraint Language) und den UML Action Semantics macht. Damit wird es möglich, komplexe Aktionen und Berechnungen so zu beschreiben, dass die Funktionalität gleich ausführbar wird.

Involvierung der Kundenseite

Doch was sollen noch präzisere UML-Modelle, wenn der Kunde jetzt schon keine UML-Diagramme lesen will? Genau das ist der Punkt: Der Kunde soll keine UML-Diagramme lesen, sondern direkt mit der ausführbaren Spezifikation arbeiten – und so testen, ob die beschriebene Funktionalität ihre Anforderungen erfüllt. Die Kommunikation mit dem Entwicklerteam wird auf diese Weise direkter und die Spezifikation in der Folge rasch stabilisiert. Da die Spezifikation ausführbar ist, entsteht bei allen Beteiligten schnell ein gutes Gefühl bezüglich Vollständigkeit.

Auf dieser Grundlage lässt sich das Testen systematisieren: Die auf der Simulation durchgespielten Testfälle und -daten lassen sich aufzeichnen und als Regressionstests beliebig wiederholen. Eine gut ausgestattete Simulationsumgebung erlaubt nebst Black-Box- und Glas-Box-Tests auch eine feingranulare Kontrolle der Simulationszeit. Dies ist eine grosse Hilfe bei der Weiterentwicklung der Sys-

temfunktionalität. Da die Tests direkt auf der Spezifikation erfasst und mit den Anforderungen verknüpft sind, vereinfacht sich das Test-Management. Zum einen lässt sich die Spezifikation validieren und zum anderen können daraus Tests für die Implementation generiert werden.

Gute Erfahrungen

Unabhängig davon, ob eine ausführbare Spezifikation zur externen Realisierung vergeben wird oder einem Code-Generator als Input dient, ist sie eine präzise Vorgabe, die von Mensch und Maschine eindeutig verstanden wird. Modellbasierte Ansätze skalieren sowohl für kleine als auch für grosse Systeme, welche durch mehrere Entwickler im Parallelbetrieb spezifiziert und getestet werden. Ausführbare Spezifikationen sind darum der effizienteste Weg, Anforderungen rasch zu stabilisieren und präzise zu formulieren. Unsere Erfahrung zeigt, dass sich ein xUML-Modell eines realen Systems innerhalb weniger Tage erstellen lässt. Darin sind jedoch die Aufwände für die Abklärung der geforderten Funktionalität nicht enthalten. Aber diese Aufwände sind sowieso zu leisten, was in einem konventionellen Entwicklungsprozess aber später – und damit wesentlich teurer – geschieht. ☺

Rolf Gubser, KnowGravity

Info

KnowGravity Inc.

Hohlstrasse 534
8048 Zürich
Tel. 044 43 42 000
info@knowgravity.com
www.knowgravity.com