

Title / Titel

Requirements Engineering und Testing in agilen Entwicklungsprojekten - ein Erfahrungsbericht

Speaker(s) / Referent(s)

Volz, Andreas ; Valtech GmbH, Deutschland

To whom is the presentation addressed? / An wen richtet sich der Beitrag?

Business Analysten, Entwickler, Test-Manager

Keywords / Stichwörter

Automation, FitNesse

Abstract / Zusammenfassung

Motivation

Projektbeschreibung

Die Produktpalette von BMW mit seinen technisch baubaren und vertrieblich angebotenen Fahrzeugen wird in einem zentralen Produktstammdatenmaster definiert und gepflegt. Anhand dieser Daten können bestehende Fahrzeugkonfigurationen auf ihre Gültigkeit hin überprüft werden bzw. gültige Fahrzeugkonfigurationen erstellt werden.

Die Komplexität der Regeln für gültige Fahrzeugkonfigurationen aus dem Derivat- und Variantenmanagement, Märkten und Produktionsstandorten hat in den letzten Jahren stark zugenommen. Die Gründe hierfür liegen vor allem im gestiegenen Mengengerüst (mehr Fahrzeugvarianten), der zunehmenden Produktkomplexität sowie den immer kürzer werdenden Lebenszyklen.

Hinzu kommt eine Vielzahl von Vorserienfahrzeugen, die gesonderte Regeln erfordern und störungsfrei in der Linie gefertigt werden müssen.

Vor diesem Hintergrund gilt es, Inkonsistenzen von Fahrzeugkonfigurationen gegenüber dem Produktangebot frühzeitig und nicht erst in der Produktion zu erkennen, wo diese Inkonsistenzen zu Fehlbauten führen.

Dazu müssen bestehende Fahrzeugkonfigurationen ständig gegen das aktuell definierte Produktangebot validiert werden.

Dabei wird ihre generelle Gültigkeit sicher gestellt. Dies kann, muss jedoch nicht, die Baubarkeit in einem bestimmten Werk umfassen.

Im Rahmen der schrittweisen Einführung des Produktstammdatenmasters sollte ein Prüf- und Konfigurationsdienst entwickelt werden, der zum einen die Validierung bestehender Fahrzeugkonfigurationen und zum anderen die Erstellung valider Fahrzeugkonfigurationen ermöglicht.

Implementierung

Berater der Valtech GmbH entwickelten gemeinsam mit BMW Mitarbeitern während der Projektlauf-

zeit von über 18 Monaten ein Konzept und die regelbasierte Lösung für den Prüf- und Konfigurationsdienst sowie dessen Einbindung in die BMW Systemlandschaft.

Die Kernkomponente wurde unter Verwendung der Drools Inferenzmaschine realisiert und in einer Java EE Applikation eingesetzt. Als Programmiersprache kam dabei Java zum Einsatz.

Die Software-Entwicklung wurde agil und iterativ durchgeführt, um möglichst kurze Feedback-Schleifen zu den Fachanwendern zu erreichen. Dies war aufgrund der hohen fachlichen Komplexität des Projekts und der damit verbundenen ebenfalls hohen Änderungsfrequenz unabdingbar.

Die Realisierung wurde in mehreren Phasen getätigt, um den Aufbau BMW eigener Kapazitäten in agiler Entwicklung von komplexen Lösungen zu fördern.

In diesem Umfeld war der Aufbau und die Verwendung eines geeigneten Testsystems von höchster Wichtigkeit; zum einen um die Anforderungen an den Prüf- und Konfigurationsdienst zu präzisieren und zum anderen um die agile Vorgehensweise zu unterstützen.

Requirements Engineering

Das Requirements Engineering befindet sich momentan im Umbruch. Die klassische Anforderungsanalyse und -dokumentation wie sie im Wasserfallmodell oder anderen schwergewichtigen Vorgehensmodellen zum Einsatz kommt, ist nicht mit dem Vorgehen in modernen agil und iterativ durchgeführten Entwicklungsprojekten vereinbar.

Somit mussten agile Prinzipien in das Requirements Engineering für den Prüf- und Konfigurationsdienst einfließen.

Abgeleitet von den zentralen Aspekten des Agilen Manifestes

(<http://agilemanifesto.org/principles.html>) muss das Requirements Engineering in agilen Projekten unter anderem folgende Punkte adressieren:

- **early delivery**
Die Entwicklung und Auslieferung der Software beginnt bereits bevor sämtliche Anforderungen detailliert spezifiziert sind.
- **changing requirements**
Die Anforderungen dürfen sich auch in späten Phasen der Entwicklung noch ändern. Das bedeutet, dass das Requirements Engineering über die gesamte Projektlaufzeit hinweg durchgeführt wird.
- **deliver frequently**
Aus diesem Punkt leitet sich eine iterative Vorgehensweise bei der Software-Entwicklung ab, was wiederum Auswirkungen auf das Requirements Engineering hat: spätestens zu Beginn einer jeden Iteration müssen sämtliche Anforderungen, die innerhalb dieser Iteration implementiert werden sollen, detailliert spezifiziert und mit dem Kunden abgeklärt sein.
- **face-to-face conversation**
Unmittelbarer persönlicher Austausch zwischen Analyst, Entwickler und Kunde.
- **working software is the primary measure of progress**
Nicht die Spezifikation einer Anforderung ist entscheidend, sondern deren Implementierung! Es sollte also nur so viel wie nötig dokumentiert werden.

Diesen Punkten wurde mit verschiedensten Maßnahmen Rechnung getragen. Aufbauend auf einer frühen Version des Fachkonzeptes wurden gemeinsam mit dem Kunden grobe Anforderungen definiert, die in einem browsergestützten Issue Tracking System dokumentiert und verwaltet wurden. Dadurch konnten die Anforderungen sowie deren Umsetzungsgrad von jedem Projektmitarbeiter eingesehen und somit transparent gemacht werden.

Nach Einplanung in die einzelnen Entwicklungs-Iterationen und Zuweisung an die verschiedenen

Entwickler wurden die Anforderungen mittels Testfällen in dem Test-Framework FitNesse detailliert spezifiziert. Die Testfälle wurden dabei direkt von Fachanwendern spezifiziert.

Die Definition von Testfällen hat sich insbesondere aufgrund der hohen fachlichen Komplexität bei der Entwicklung des Prüf- und Konfigurationsdienstes als unerlässliches Hilfsmittel erwiesen und trug hierdurch maßgeblich zum Erfolg des Projektes bei.

Die Fachanwender wurden bei der Erstellung eines Testfalls durch die Spezifizierung der Eingabe sowie der erwarteten Ausgabe des Software-Systems dazu gezwungen, die Implikation einer Anforderung genau zu überdenken. Dadurch konnten Inkonsistenzen innerhalb der Anforderungen in einem frühen Stadium erkannt und behoben werden.

Zusätzlich sorgte die Formalisierung der Anforderungen bei der Testfalldefinition dafür, dass die zumeist "schwammig" in Prosa formulierten Anforderungen strukturiert und geschärft wurden. Somit entstanden detaillierte Anforderungen mit klar zugeordneten Testfällen, gegen die entwickelt werden konnte.

Die fertigen Testfälle dienten neben ihrer Funktion als Leitlinie und psychologischer Anreiz für die Entwicklung am Ende einer Iteration zusätzlich als Abnahmekriterium. Eine Anforderung galt dabei erst als umgesetzt, wenn alle zugeordneten Testfälle bestanden wurden.

Darüber hinaus wurden bestandene Testfälle als Regressionstests eingesetzt, um die implementierte Funktionalität gegenüber späteren Änderungen abzusichern.

Agile Testing

Auch an das Testen werden in einem agil durchgeführten Software-Entwicklungsprojekt besondere Anforderungen gestellt.

Insbesondere weisen diese Projekte eine höhere Testfrequenz auf als Projekte, die mittels eines klassischen Entwicklungsmodells umgesetzt werden. Dies erfordert einen hohen Grad an Automatisierung innerhalb des Testens.

Bei einem testgetriebenen Vorgehen, wie es in den meisten agilen Projekten - und so auch in unserem - verfolgt wird, müssen vor jeder Iteration für alle zu implementierenden Anforderungen zusammen mit dem Kunden Testszenarien erstellt werden. Die Testszenarien werden wiederum in Testfälle zerlegt, um eine adäquate Testabdeckung für die geplante Funktionalität zu gewährleisten.

Während der einzelnen Iterationen testen die Entwickler ihren Code ständig gegen die erstellten Testfälle und sorgen dafür, dass die Testfälle bestanden werden.

Die erstellten Testfälle müssen jedoch auch nach dem erfolgreichen Abschluss einer Iteration weiterhin als Regressionstests durchlaufen werden, um die Korrektheit des Software-Systems abzusichern. Aufgrund der hohen Änderungshäufigkeit in agil durchgeführten Entwicklungsprojekten ist dieser Punkt besonders wichtig: das Erreichte muss effizient abgesichert werden!

Analog zur agilen Entwicklung steht beim agilen Testen der Kunde im Fokus; genauer gesagt die Personen, die später das fertige System benutzen werden. Die Tests sollten somit in einer Sprache definiert werden, die der Kunde versteht.

Aufgrund sich ändernder Anforderungen des Kunden kann es dazu kommen, dass die Erwartungshaltungen von Tests angepasst werden. Unter Umständen kann dies sogar bedeuten, dass Tests komplett neu definiert werden müssen.

Mit FitNesse wurde ein Test-Framework gefunden, das es uns ermöglichte, die oben genannten Anforderungen zu erfüllen.

Insbesondere die Erstellung neuer bzw. die Anpassung bestehender Testfälle konnte unproblematisch und vor allem schnell durchgeführt werden. Dasselbe galt für die Fixtures; also die Java-Klassen, die die Logik zur Definition von FitNesse-Testfällen beinhalten.

Des Weiteren ließen sich die FitNesse-Tests einfach in unsere Build-Umgebung integrieren. Durch diese Integration konnte ein hoher Automatisierungsgrad beim Testen erzielt werden.

Unsere Build-Umgebung bestand aus einem Continuous Integration-Tool, das primär die Stabilität des Builds sichergestellt hat. Außerdem sorgte es dafür, dass nach jedem erfolgreichen Build die komplette Regressionstestsuite durchlaufen wurde und Informationen zu evtl. fehlgeschlagenen Testfällen an die Entwickler weitergeleitet wurden.

Des Weiteren enthielt der Build eine QS-Suite aus Open-Source-Werkzeugen, die QS-Metriken und die Testabdeckung geprüft hat. Dadurch konnte die Code-Qualität zu jedem Zeitpunkt bestimmt werden.

Ergänzt wurde das automatische Testen durch separat durchgeführte Performance- und Integrations-Tests sowie eine Phase, in der die Ergebnisse des neu entwickelten Prüf- und Konfigurationsdienstes mit denen des abzulösenden Legacy-Systems verglichen wurden.

Fazit

Die größten Herausforderungen bei der Entwicklung des Prüf- und Konfigurationsdienstes waren die hohe fachliche Komplexität und die damit einhergehenden unklar spezifizierten Anforderungen. Nur die agile, auf Änderungen ausgerichtete Vorgehensweise konnte vor diesem Hintergrund zum Erfolg führen. Dies wird auch von der Tatsache belegt, dass mehrere nicht agil durchgeführte Projekte vor uns an der Aufgabe (Entwicklung eines Prüf- und Konfigurationsdienstes) gescheitert sind.

Aus unseren Erfahrungen in diesem Projekt lässt sich ableiten, dass Testfälle so früh wie möglich, am besten bereits bei der Spezifizierung der Anforderungen, definiert werden sollten. Dies führt zu einer Schärfung und klaren Strukturierung der Anforderungen. Zudem stellen Testfälle, die Anforderungen zugeordnet sind, einen zuverlässigen Gradmesser des gegenwärtigen Entwicklungsstands dar und sorgen natürlich für eine hohe Testabdeckung.

Biography / Biografie

Andreas Volz ist Dipl. Informatiker und als Consultant bei der Valtech GmbH angestellt. Er verfügt sowohl in der Systemintegration mittels EAI als auch in der Entwicklung von regelbasierten Systemen im automotive Bereich über mehrjährige Projekterfahrung. Sein besonderes Interesse gilt der Anwendung agiler Methoden in der Softwareentwicklung; insbesondere im Requirements Engineering.

Olaf Gottschalk ist Dipl. Ingenieur der Elektrotechnik und seit 2000 bei der BMW AG als IT Spezialist angestellt. Er arbeitete mehrere Jahre im Bereich der Produktionssteuerung in in- und ausländischen Standorten der BMW Group. Seit 2007 ist er im Bereich der IT Architektur und im Build Management tätig.

Aus der Zeit der Produktionssteuerung verfügt er über vielfältige Erfahrungen sowohl im Bereich von unternehmensweiten Rolloutprojekten, der Standardisierung und dem Betrieb von produktionskritischen Systemen. Die enge Zusammenarbeit mit (internen) Kunden und das Verständnis der fachlichen Zusammenhänge, gemeinsam mit den Kunden sowohl die richtige Definition zu finden zählt zu seinen täglichen Aufgaben.

Contact information / Kontaktinformationen

Valtech GmbH
Andreas Volz
Zweigstr. 10
80336 München
Deutschland
