Ausgabe 2012 / 2

Erscheinungsart: ca. 4 x jährlich in elektronischer Form

Requirements-Spezifikation im agilen Umfeld

In dieser Ausgabe:

Braucht es im agilen Umfeld überhaupt Requirements-Spezifikation?

Ja, auch im agilen Umfeld ist Requirements-Spezifikation notwendig und entweder explizit oder jedenfalls implizit vorhanden.

Beispiele, wo Requirements-Spezifikation im agilen Bereich vorkommt:

- Erstellen von User-Stories,
- Ausarbeiten von Feature- oder Use-Case-Beschreibungen,
- User-Interface-Prototypen (Mock-Ups),
- Klären von unklaren Anforderungen zwischen Auftraggeber (Product-Owner) und Entwicklern, Testern, etc.,
- Diskussion und Feinspezifikation von Features z.B. im Rahmen des Sprint-Plannings,
- Erstellen von Testspezifikation und Test-Cases im Rahmen von testgetriebener Entwicklung,

Die Themen und Bereiche, in denen Requirements-Spezifikation im agilen Bereich durchgeführt bzw. wirksam wird, sind vielfältig. Man sollte dabei durchaus beachten, dass auch im agilen Umfeld mehr Spezifikation nicht automatisch schlecht ist!

Eine oft praktizierte Möglichkeit ist es, das Thema Requirements-Spezifikation im agilen Bereich intuitiv umzusetzen. Dies führt aber meist zu einer ineffizienten Arbeitsweise. Dieser Knowledge-Letter fasst daher einige derzeit gebräuchliche Requirements-Spezifikationstechniken im agilen Umfeld zusammen.

Folgende Spezifikationstechniken werden behandelt:

- **User-Stories**
- **Use-Case Description**
- **Test-getriebene Entwicklung**
- **Specification by Example**
- **Behaviour-driven Development**

Der Knowledge-Letter soll damit als Anstoß und Anregung dienen, sich mit diesem Thema auch strukturiert auseinander zu setzen und Requirements-Spezifikation als Thema auch in den agilen Prozessen explizit zu definieren und zu verankern.

Wie agil können und sollen Requirements sein?

Haben Sie schon einmal ein Projekt erlebt, in dem die Anforderungen von Anfang bis zum Ende stabil geblieben sind?

Bis auf wenige Ausnahmen ist es eine Tatsache (vor allem in der SW-Entwicklung), dass sich Anforderungen ändern. Stabile Anforderungen festzuschreiben würde daher bedeuten, die Realität zu ignorieren und dadurch Zusatzaufwände zu verursachen.

Ein bekanntes Zitat lautet: „Das einzig Beständige ist die ständige Veränderung!“ Daher kann es nicht das Ziel sein, die Requirements selbst einzufrieren oder deren Veränderung zu verhindern. Es kann auch nicht das Ziel sein, gar keine Requirements mehr zu spezifizieren, um dadurch zu verhindern, dass sie sich ändern. Requirements sind immer vorhanden, auch wenn sie nicht explizit niedergeschrieben werden.

Vielmehr muss es darum gehen, die passenden Rahmenbedingungen zu schaffen, damit sich Requirements verändern können, ohne dadurch das Projekt oder die Entwicklung zu gefährden oder — über den gesamten Lebenszyklus gesehen — unnötige Zusatzkosten zu verursachen.

Die Lösung liegt daher im Management der Anforderungsänderungen. Hier müssen flexible Tools und Techniken eingesetzt werden, um einerseits Anforderungen leicht zu erstellen und anderseits auch Veränderungen zu ermöglichen und strukturiert nachzuverfolgen.

Dipl.-Ing. Johannes Bergsmann
Staatl. befugter und beeideter Ingenieurkonsulent für Informatik

Der Quality-Knowledgeletter ist ein periodisches Informationsmedium von Software Quality Lab und dessen Partnern mit den Schwerpunkten IT-Qualitätsmanagement, Projekt- und Prozess-Management.

Inhalt: fachliche Beiträge und Schwerpunktthemen, Vorstellung neuer Produkte und Leistungen, neue wissenschaftliche Erkenntnisse und andere Fachbeiträge aus unseren Themenbereichen.

Aktuelle Fach- und Forschungsbeiträge sind willkommen. Einsendungen an info@software-quality-lab.com.

Weitere Infos zu diesem und anderen Themen finden Sie auf www.software-quality-lab.com.

Requirements-Spezifikation im agilen Umfeld

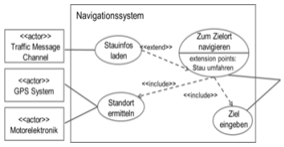
DI Johannes Bergsmann - Software Quality Lab

In Agilen Methoden wird Requirements-Engineering oft intuitiv betrieben – diesem Thema wird etwas zu wenig Beachtung geschenkt. Mittlerweile haben sich schon verschiedene Spezifikationstechniken im agilen Umfeld etabliert. Dieser Beitrag gibt einen Überblick über die derzeit am weitesten verbreitete Techniken der Requirements-Spezifikation und beleuchtet jeweils die Einsatzmöglichkeiten aber auch die Fallstricke der einzelnen Techniken.

User-Stories

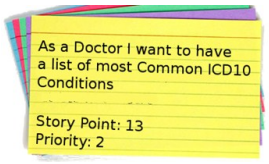
Die Charakterisierung von Use-Cases nach der Agile Alliance (www.agilealliance.org) ist wie folgt definiert:

- **Use Case/Feature:**
  A fuzzy, client valued capability of the system. A feature typically contains many stories… in fact, there is almost always "another one".



  Hinweis: Es muss klar unterschieden werden zwischen Use-Case und Use-Case-Description (diese Technik wird im nächsten Kapitel beschrieben)

- **User Story:**
  Represents something that provides externally visible benefit to the project (as in XP) but may be too big to build until broken down further.



- **Developer Story:**
  The fundamental unit of work, represents something that provides externally visible benefit to the project

- **Task:**
  The actual work that individuals do in order to make the story "come true" – there are usually many tasks per story

User Stories sollen im Kontext der agilen Vorgehensweisen so formuliert werden, dass Sie folgende Eigenschaften erfüllen (= ein INVEST sind):

- **I**ndependent – unabhängig von einander
- **N**egotiable – verhandelbar (kein fixer Vertrag) *)
- **V**aluable – erkennbarer Wert für den Kunden
- **E**stimatable – Aufwand soll geschätzt werden können
- **S**mall – klein und schnell umsetzbar (in 1 Sprint)
- **T**estable – definierte Test-/Abnahme-Kriterien

**) Bis zum Commitment des Teams mit dem Product-Owner. Danach ist es fix, bis die Definition of Done erfüllt ist.*

Eine User Story ist dann abgeschlossen (fertig), wenn die vorab definierten „Definitions of Done" erfüllt wurden (in Scrum).

Damit User-Stories einfach und systematisch erstellt werden können, hat sich eine sogenannte **Satz-Schablone** etabliert:

Als <**Benutzerrolle/Systemrolle**>
will ich <**das Ziel / Story**>,
sodass <**Grund für das Ziel / Nutzen**>

Hinweis: Die Anwendung ist nicht unbedingt genau in dieser Syntax erforderlich!

Beispiele für einfache User-Story Formulierungen:

- Als Projektmitglied will ich monatlich meine geleisteten Projektstunden abfragen.
- Das System darf bei 50 gleichzeitig aktiven Usern eine Reaktionszeit von max. 3 Sec. aufweisen, da das System auch bei einer Expansion bis zu dieser Größe geeignet sein muss.
- Ein User kann die in seinem Konto gespeicherten Kreditkartendaten bearbeiten.

Schlechtes Beispiel (keine typische User-Anforderung):

- Das System wird mit der Datenbank über einen Konnektor-Pool verbunden.

Beachtet werden muss, dass User-Stories nur eine bestimmte Ebene in der Spezifikationshierarchie Stories repräsentieren. User-Stories sind meist gute Hilfsmittel für die Grobspezifikation und sind gut als Strukturierungs- und Planungsinstrument geeignet.

Achtung: Die Gefahr, wenn man sich vorwiegend auf diese Technik konzentriert ist, dass die Detailspezifikation ev. vernachlässigt wird! Auf der Story Card ist typischerweise zu wenig Platz, um die Detail-Informationen im benötigten Umfang niederzuschreiben. Ebenso ist es ein Problem, wenn die übergeordnete Spezifikation nicht erstellt wird. Vor allem Business-Prozesse werden oft zu unsystematisch und zu ungenau beschrieben, weil dies oft ein Mehraufwand ist und man aufgrund der User-Story-Schablone verleitet ist, zu schnell in funktionale Spezifikationen abzugleiten!

Durch User-Stories werden auch nicht alle Aspekte einer guten Spezifikation abgedeckt. So fehlen meist Architektur, Kontext, grafische Prozessmodellierung, Visualisierung, etc.

User-Stories alleine stellen daher zumeist keine vollständige / ausreichende Spezifikation dar!

Vollständiger Knowledge Letter Zugang

Wir freuen uns, dass Sie an diesem Thema Interesse haben und den Knowledge Letter von Software Quality Lab bis hierher gelesen haben.

Dieser Knowledge Letter ist eine Vorschau (gekürzte Version des gesamten Artikels).

Wenn Sie den ungekürzten Knowledge Letter lesen möchten, registrieren Sie sich bitte unter
http://www.software-quality-lab.com/download/knowledge-letter/anfrage-knowledge-letter/

Sie erhalten nach der Registrierung vollen Zugang zu allen bisherigen Knowledge Lettern von Software Quality Lab und erhalten automatisch künftige Knowledge Letter per E-Mail.