BDD + DDD = eine hybride Testkonzeption

Das agile Produktteam „KiM“, verwendet in Bezug auf die Qualitätssicherung seiner Produkte zwei wesentliche Ansätze, die wir im Rahmen dieses Blogs kurz skizzieren wollen. Dabei geben wir nicht nur einen konzeptionellen Einblick, sondern stellen auch aus unserer Sicht die Vorteile für die Anwendung eines hybriden Ansatzes heraus.

Behaviour Driven Development

Beim Behaviour Driven Development steht das erwünschte Verhalten der Software aus Sicht des Kunden im Vordergrund. In der Anforderungsanalyse werden die Ziele, Aufgaben und Ergebnisse der Software in Textform als Szenarien festgehalten. Die Szenarien werden unter Verwendung des „Given-When-Then“ – Konzepts erstellt (siehe auch: Akzeptanztests bei Thalia) und dienen als Basis für die Entwicklung automatisierter Tests. Somit wird bei den automatisierten Tests stark die Perspektive des Kunden eingenommen.

Dieser Ansatz bietet nicht nur den Vorteil der einfachen Lesbarkeit, sondern auch die Möglichkeit alle beteiligten Akteure (wie bspw. den Product Owner, die Fachseite, etc.) während der Testfallerstellung mit einzubeziehen und dadurch ein gemeinsames Verständnis vom System bzw. der Software aufzubauen.
Ein mögliches Beispiel für ein Szenario:

```
Szenario: Benutzer meldet sich über die Mobile-App an
Angenommen "Max Mustermann" benutzt die Mobile-App
Wenn "Max Mustermann" sich während des Bestellvorgangs erfolgreich anmeldet
Dann soll die Checkout-Seite angezeigt werden
Und "Max Mustermann" ist in der Mobile-App angemeldet
```

Ein weiteres Element, welches beim BDD-Konzept das gemeinsame Verständnis stärkt, ist der Einsatz einer „ubiquitous language“.
Die „ubiquitous language“ dient als gemeinsam definiertes und genutztes Vokabular, wodurch Fehlinterpretationen bei der Verwendung von Begriffen

minimiert werden. Beispielsweise definieren Fachbereiche wie das CRM oder die Softwareentwicklung Begriffe wie „Kunde“ oder „Auftrag“ unterschiedlich. Der Kunde könnte zum einen als Kunde interpretiert werden, sobald er sich im Webshop eingeloggt hat oder zum anderen wenn er mindestens einen Auftrag durchgeführt hat. Dieses heterogene Verständnis kann zu Problemen bei der Testfallerstellung führen, beispielsweise durch fehlende oder kontroverse Testfälle.

Die eingenommene Perspektive beim BDD-Ansatz offenbart allerdings auch einige Schwächen, die im folgenden beschrieben werden und weshalb sich das Team KiM zu einem Einsatz eines weiteren Konzepts entschieden hat: Dem Domain-Driven Design.

Domain-Driven-Design in der Praxis

Um den eingenommenen Blickwinkel des Endkunden durch den BDD Ansatz im Softwareentwicklungsprozess zu erweitern und mögliche Grenzen zwischen IT und Business aufzuweichen, kombiniert das Team KiM während der Anforderungsanalyse den Behaviour-Driven-Development mit dem Domain-Driven-Design (DDD) Ansatz. Ausgehend vom Business eines Unternehmens, bis hin zu einer fachlichen Domäne (wie bspw. Vertrieb, Buchhaltung, CRM, etc.) bietet das strategic Design des DDD Ansätze zur Strukturierung und Aufteilung der Domäne. Das strategic Design, als einer- von vier Bestandteilen des DDD-Konzeptes, steht bei Team KiM im besonderen Fokus, da sich hier sehr schnell ein Mehrwert generieren lässt. Aufbauend auf der ubiquitous language bietet das strategic Design u.a. folgende Techniken bzw. Konzepte:

- Event Storming
- bounded Context
- Context Map

Event Storming

Nach der Bestimmung der Domäne können anhand des Event Storming die jeweilig relevanten Events innerhalb einer Domäne identifiziert und zugeordnet werden.

Am Beispiel eines sog. „Newsletter“-Projektes“ hat das Team KiM in enger Zusammenarbeit mit Product-Owner und dem Fachbereich die Events in der Domäne ermittelt und das Ergebnis festgehalten:

Event Storming – Newsletter

bounded Context

Die im Event Storming ermittelten Events lassen sich in bounded Contexts (sog. eigenständige Modelle) clustern. Folgende bounded Contexts wurden festgehalten:

- Kontaktverwaltung
- Registrierung
- Newsletter (An/-Abmeldung)
- Gewinnspiel
- Präferenzen

Der bounded Context (s. Bild unten) bietet leider noch keinen brauchbaren Systemüberblick unter Berücksichtigung des Kommunikations-, Daten- und Modellflusses. Diesen Part übernimmt die Context map, die den Zusammenhang zwischen Model und dem Kontext skizziert.

Context Map

Die Context Map bietet einen guten Ausgangspunkt für die nachfolgende Softwareentwicklung und die Testfallerstellung. Am Beispiel des Newsletter-Projektes hat das Team KiM folgende context map erstellt:

Context Map – Newsletter

Die Domain Events aus dem Event-Storming werden den einzelnen bounded contexts (blaue Post-its) zugeordnet. Darüber hinaus ist der Daten- und Modellfluss anhand von Pfeilen (rot + schwarz) skizziert.
Die Abkürzungen CF, OHS, SUP und CUS stellen sogenannte context map patterns dar, mit der Bedeutung:

- CF = Conformist (Das Downstream Team übernimmt das Modell des Upstream Teams)
- OHS = Open / Host Service (der Host bietet einen vordefinierten Satz an Services an)
- SuP = Supplier (Supplier-Customer i.S.v. Kunde-Lieferantenbeziehung)
- CuS = Customer (Supplier-Customer i.S.v. Kunde-Lieferantenbeziehung)

Die erstellte Context Map bündelt nicht nur die Ergebnisse der hier vorgestellten Techniken, sondern unterstützt zugleich die Testfallentwicklung. Hierbei hilft nicht nur die Visualisierung des Systems, sondern auch die Gruppierung der Events und die Modellzuordnung. Dadurch lassen sich präzisere bzw. eindeutig definierte Testfälle erstellen.

Vorteile

Den hier skizzierten hybriden Ansatz hat das Team KiM bereits erfolgreich in den Softwareentwicklungsprozess integriert und aufgrund folgender Vorteile zu schätzen gelernt:

- Berücksichtigung mehrerer Perspektiven (Business und Endkunde)
- Integration der Testfallentwicklung

- Beteiligung aller relevanten Akteure am Entwicklungsprozess
- Reduzierung von Fehlinterpretationen bei Sachverhalten/Begrifflichkeiten (ubiquitous langauge)
- Reduzierung des Testaufwandes durch Testautomatisierung

Abschließend zu diesem Blog stellen sich dem Team die Fragen:

1. Wie ist eure Meinung zu dem o.g. Vorgehen ?
2. Welche Erfahrungen habt ihr mit dem BDD- bzw dem DDD-Ansatz gemacht?