

BDD - Brauchst Du Das?

@KatrInRabow

BDD - Behaviour Driven Development

@KatrinRabow

@KatrinRabow

- Reiseverkehrskauffrau
- Buchhalterin
- Beraterin
- Softwareentwicklerin



BDD in a nutshell



Auch ATDD (Acceptance Test Driven Development)

Entwicklung wird getrieben durch die Frage, wie sich die Software für die Nutzenden verhalten soll



Stakeholder werden sehr früh in den Entwicklungsprozess miteinbezogen

Anhand konkreter Beispiele wird das gewünschte Verhalten der Software beschrieben



Aus der Spezifikation dieser Beispiele ergeben sich die Akzeptanzkriterien für die Umsetzung.

Software Development



Product Owner:in oder Business Analyst:in
Welches Problem versuchen wir zu lösen?

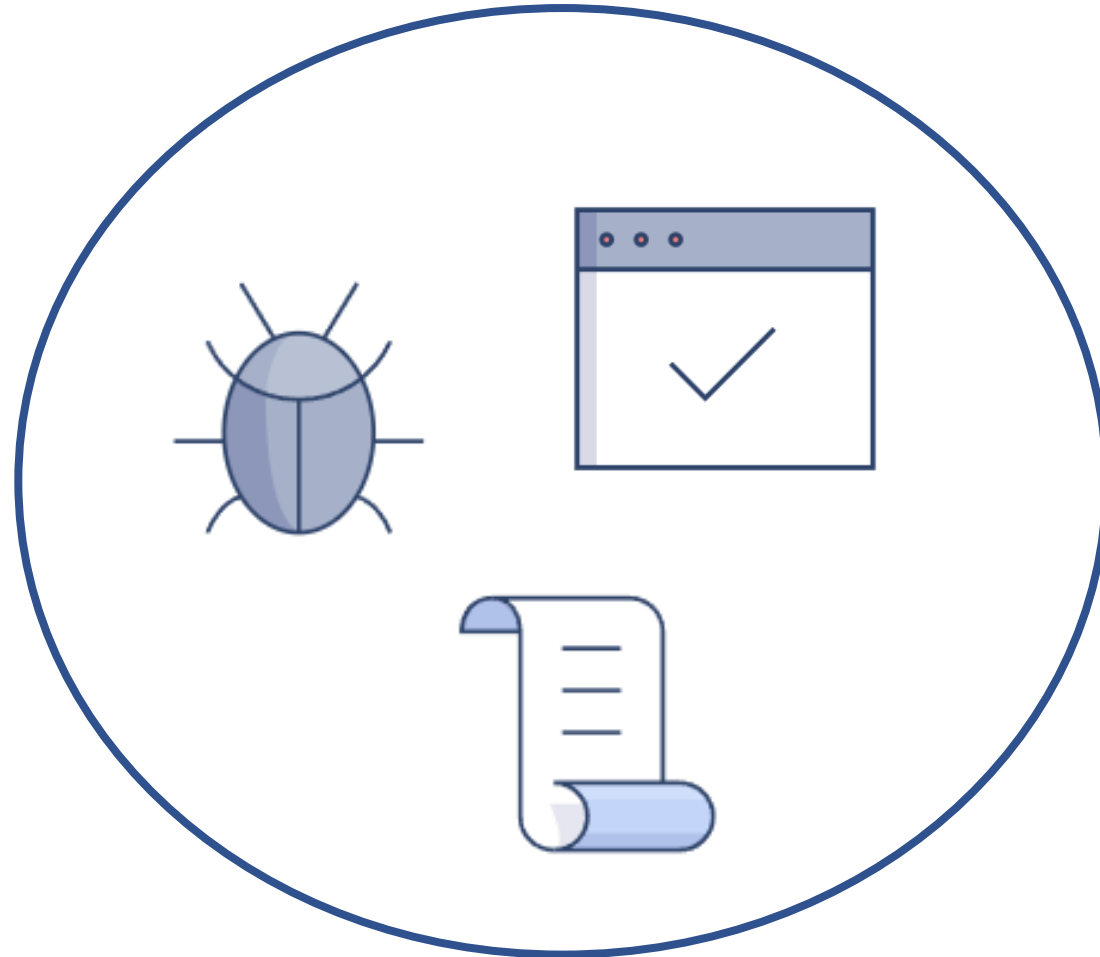


Entwickler:in
Wie könnte eine Lösung zu diesem Problem aussehen?

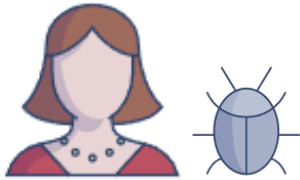


Tester:in
Was würde passieren, wenn...?

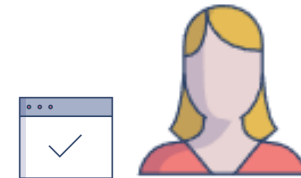
Three Amigo Prinzip



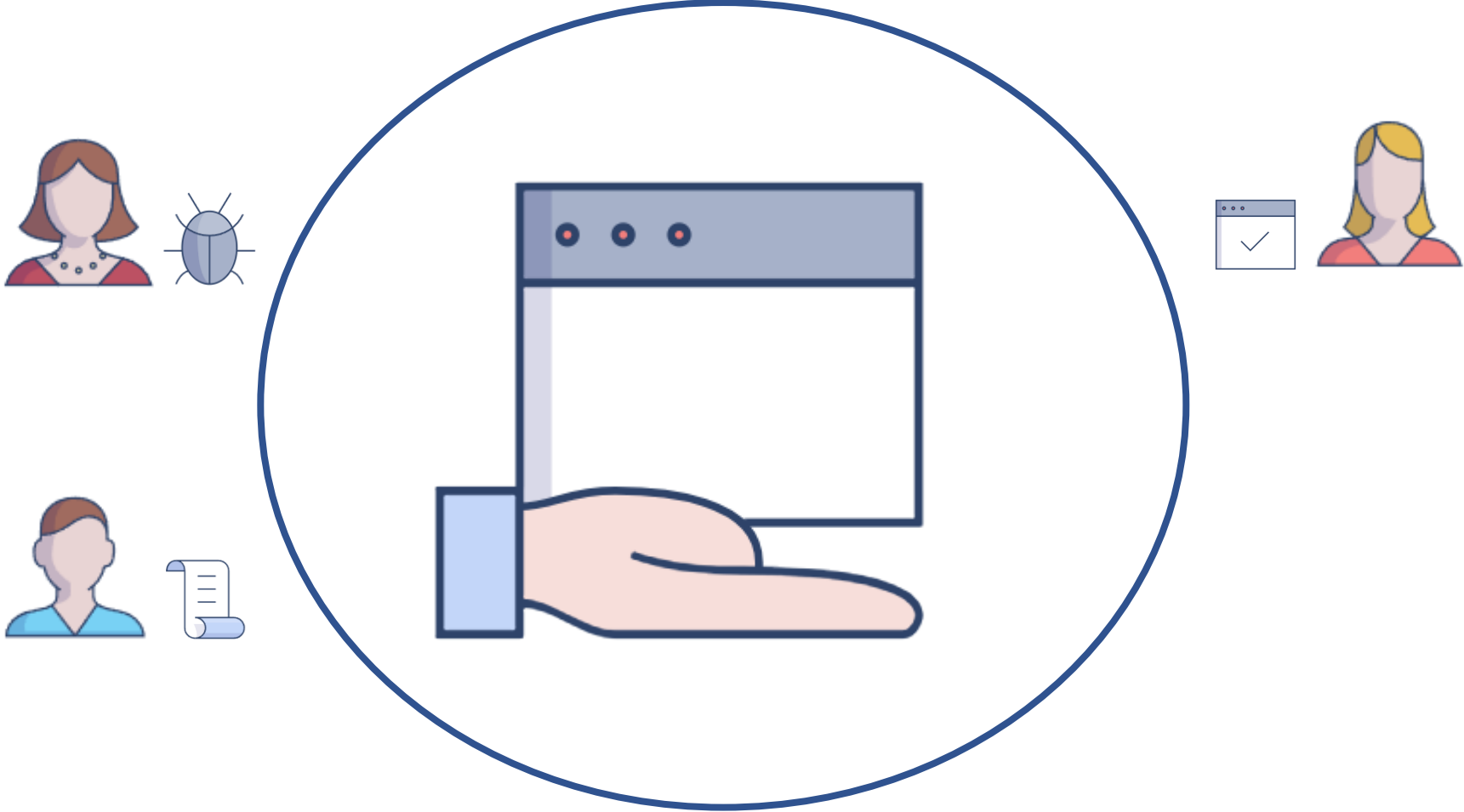
Three Amigo Prinzip



Feature: Formular ausfüllen
Scenario: Ausfüllen
Given Person hat Formular ausgefüllt
When Person das Formular einreicht
Then erhält die Person eine Mail
Scenario: Keine Telefonnummer
Given Person erfasst ihre Daten
When die Person die Daten speichert
Then kommt ein Hinweis
...



Three Amigo Prinzip



Test Driven Development



DEMO

Die Beschreibung des gewünschten Verhaltens (Funktionalität und Szenarien) erfolgt in natürlicher Sprache unter Anwendung der jeweiligen Fachsprache (der sogenannten ‚ubiquitären Sprache‘). Dabei liegt der Fokus darauf zu formulieren, WAS erreicht werden soll und nicht, WIE es erreicht werden soll. Die Szenarien sollten daher relativ abstrakt beschrieben werden - so präzise wie nötig, nicht so präzise wie möglich.

Beispiel (noch nicht formalisiert)

„Damit die nutzende Person unseren Vorgang ausführen kann, werden ihre persönlichen Daten benötigt.“

Daraus können sich unterschiedliche Szenarien ergeben (Daten liegen bereits vor, Daten müssen noch erfasst werden, ...). Um den Übergang zu automatisierten Tests zu erleichtern, werden die Beispiele nun in eine bestimmte Form gebracht:

Beispiel (formalisiert)

Funktionalität: Die persönlichen Daten der nutzenden Person werden benötigt. Diese können bereits in einem Konto gespeichert sein oder müssen erst noch erfasst werden.

Als nutzende Person möchte ich meinen Namen und meine Anschrift in einem Konto speichern, um sie immer wieder verwenden zu können.

Szenario: Name und Anschrift hinzufügen

Angenommen die Antragstellung wurde gestartet

Wenn die nutzende Person Name und Adresse eingibt

Dann soll diese dauerhaft gespeichert werden

Szenario: Name und Anschrift aus Konto abrufen

...

Die natürlichsprachigen Beispiele, die auf diese Weise gemeinsam erarbeitet wurden, können als Textdateien im Projekt anschließend mit Hilfe von Cucumber in den Code übernommen werden.

Cucumber ist ein spezielles Adapter-Framework, welches die einzelnen Szenario-Schritte der Gherkin Feature Files mit der zugehörigen Implementierung der Testautomatisierung verknüpft und ausführt.

Dabei basiert der erste Code auf Klassen und Methoden, die es noch gar nicht gibt, und die z.B. mit Mocks oder Stubs imitiert werden. Im Laufe der Implementierung wird jedes „Angenommen“, jedes Ereignis aus einem „Wenn“ und jedes Ergebnis aus „Dann“ durch eine Klasse repräsentiert, so dass letztlich ein Ende-zu-Ende-Test daraus resultiert.

Die BDD Beschreibungen können direkt verarbeitet werden, indem jedem Schritt wie „Wenn die nutzende Person Name und Adresse eingibt“ konkrete Benutzeraktionen und Verifikationen zugewiesen werden. Jede Step-Definition-Methode hat einen regulären Ausdruck, der in der zugehörigen Annotation spezifiziert ist. Das Pattern wird benutzt, um die Step-Definition mit allen passenden Schritten zu verknüpfen. Auf diese Weise werden die gemeinsam verfassten Beschreibungen zu einer ‚ausführbaren Spezifikation‘.

Cucumber-Tests lassen sich mit einem speziellen TestRunner für JUnit ausführen. Hierfür legt man eine leere Testklasse an und versieht diese mit der Annotation `@RunWith(Cucumber.class)`. Jedes Szenario kann bei Bedarf auch einzeln getestet werden. Natürlich lassen sich Cucumber-Tests auch im Buildprozess mit Ant oder Maven anstoßen.

Three Amigo Prinzip



Takeaways

Scenarios müssen lesbar und verständlich sein

Richtige Granularität finden

Scenarios gemeinsam schreiben oder mindestens ein Review durchführen

Nicht alle Scenarios müssen (automatisierte) Tests werden

WICHTIG: Es muss eine gemeinsame Sprache (ubiquitous language) vorhanden sein / gefunden werden

Welche Projekte profitieren besonders?

Projekte mit sehr fachlichem Hintergrund

Hoher Reifegrad im Team

Gemeinsame Verantwortung

Wunsch nach gemeinsamem Verständnis

Bereitschaft sich in andere und deren Problemstellungen
hineinzudenken

Und welche eher nicht so?

Stark technikgetrieben

Große komplexe Strukturen

Product Owner:in steht nicht ausreichend zur Verfügung

Testende stehen nicht von Anfang an zur Verfügung

Fazit

Fokus liegt auf dem Verhalten der Software im Zusammenspiel zwischen Business und Technik

Wenn von Anfang an der Testgedanke zu stark im Vordergrund steht, fällt es viel schwerer die Beispiele zu formulieren. Gherkin soll die Beispiele unterstützen, nicht einschränken!

BDD ist eine Kollaborationsmethode – KEINE Testmethode!

Das Ziel von Behavior Driven Development ist die Zusammenarbeit und Kommunikation zwischen allen Stakeholdern. Als wertvolles Beiproduct entsteht eine Testautomatisierung und lebende Dokumentation, die immer den aktuellen Stand der Erwartungen der nutzenden Personen abbildet.

@KatrinRabow
follow me on 

Vielen Dank!
Questions welcome 😊

[Tools für Acceptance Test-Driven Development \(ATDD\) | Informatik Aktuell \(informatik-aktuell.de\)](#)

[BDD: Behavior-Driven-Development - Beispiele, Expertenwissen, Best Practices \(testing-board.com\)](#)

[Gurken zum Kaffee \(entwickler.de\)](#)

[GitHub - TNG/JGiven: Behavior-Driven Development in plain Java](#)

[Start Testing With BDD and Spock Framework | Pluralsight | Pluralsight](#)

[An Introduction to Behavior-Driven Development \(BDD\) with Cucumber for Java – YouTube](#)