Der wendige Testfall

TED 2015

Agile Softwareentwicklung erfordert agiles Testing.

Claudio Klingler

+49 1520 1000 988

ck@realtime-projects.com

www.realtime-projects.com

2015-06-01

Abstract

Wie können hohe Qualitätsansprüche an **Softwareprojekte** und deren Testing in einem **agilen Entwicklungsprozess** unter der Verwendung von **automatisierten Tests** erfüllt werden?

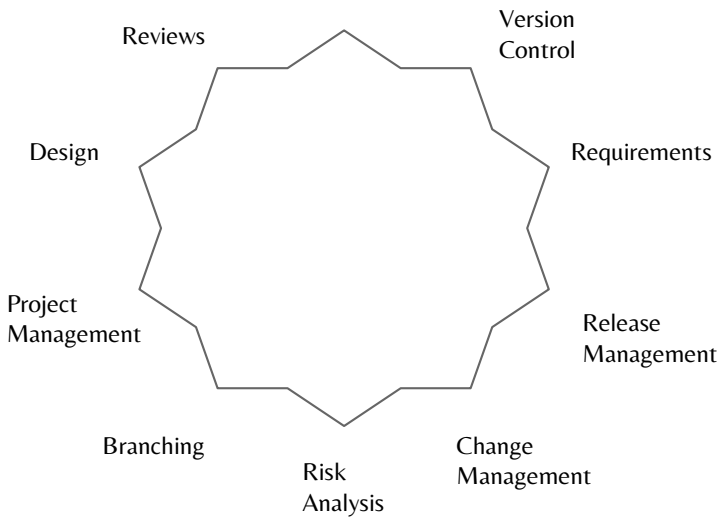
Überblick

- Warum agil?
- Die Grundlage agiler Softwareentwicklung
- Agil & Test = automatisiert?
- Die Umsetzung im agilen Projekt
- Erfahrungen

Warum agil und regulativ?

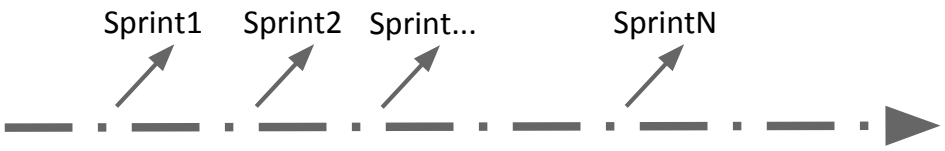
- Effizienz
- schneller Marktzugang
- unklare Ziele
- Steuerbarkeit des Projektes
- Komplexität und Beherrschbarkeit des Systems

Softwarequalität umfasst

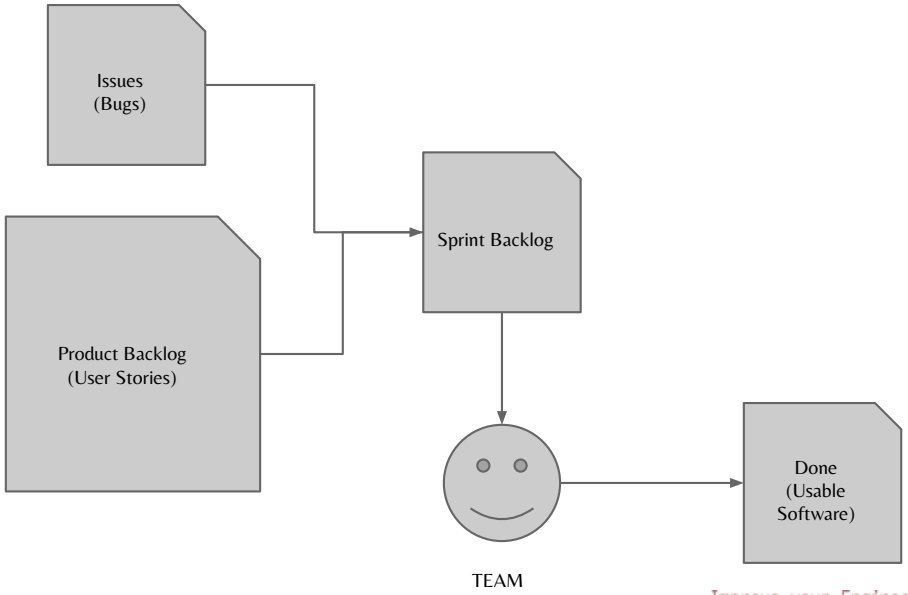


Die Grundlagen agiler Softwareentwicklung

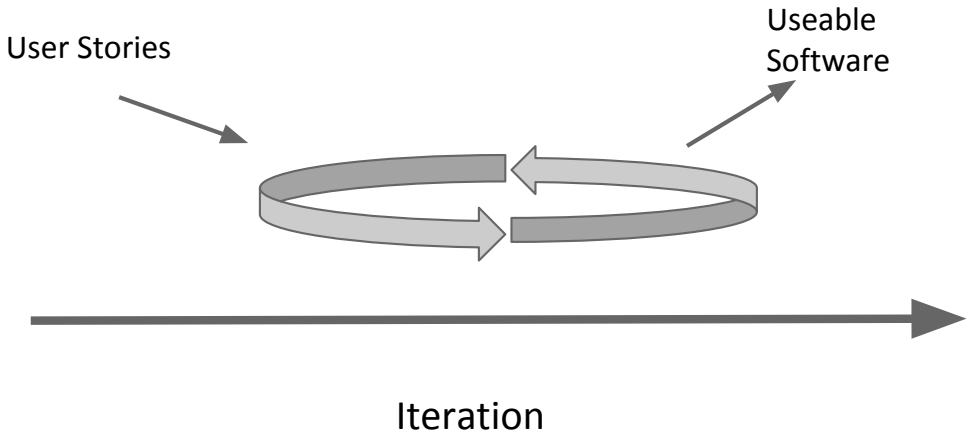
Sprints produzieren “usable Software”



agile Planung



Ein agiler Sprint



Chance: Wiederholung

Was ich wiederhole, kann ich optimieren.

- Fehlerquellen beseitigen
- Effizienz steigern

Was bedeutet agile Softwareentwicklung?

- Kurze Releasezyklen (1-2 Wochen)
- Continuous Integration
- Zeit zwischen 'neuer Anforderung', bzw. Änderungsanfrage und Release optimal 2 Wochen
- Häufiges Refactoring (auch von Schnittstellen)
- Planungshorizont 2-3 Monate?
- Anpassungen an bestehenden Tests notwendig

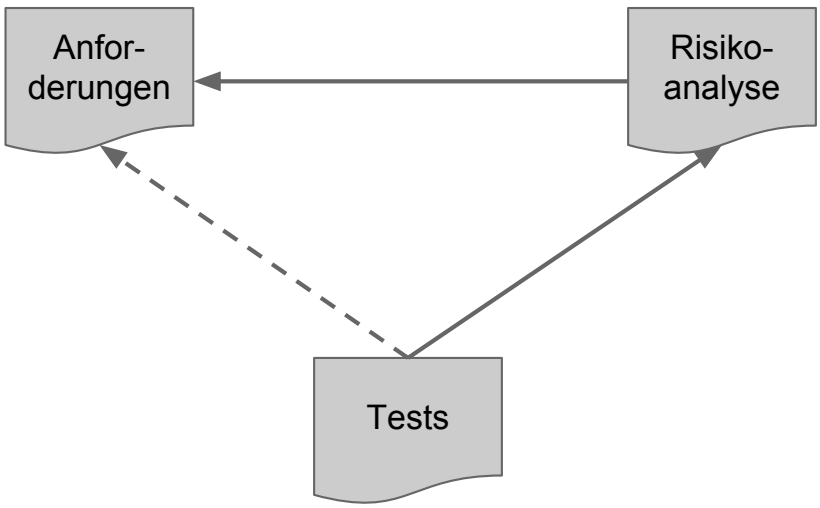
Agil + Testing = automatisiert?

- Die Entwickler profitieren von schnellem und direktem Feedback durch das agile Test-Team für Themen, die sie gerade bearbeiten
- Häufiges Refactoring ist nur möglich, wenn automatisierte Tests die Funktion des Systems sicher stellen
- ***Agil und Testing macht erst mit Testautomatisierung richtig Sinn***

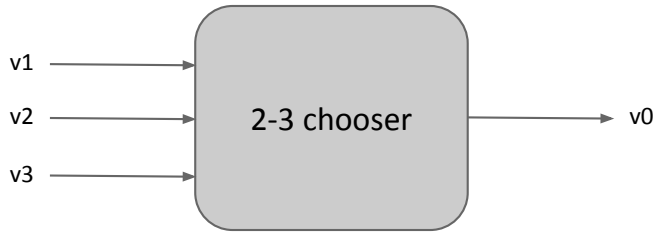
qualitätsgetriebene Projekte

- *document* what you *do*
  and *do* what you *document*
- auf Ebene des *Systems*, der *Software*, der *Organisation*
  und der *Vorgehensweisen*
- *test* what you *do (bottom-up)*
  and *test* what you *document (top-down)*

Tracebility



2-3 chooser



Ein Lösungsansatz

- Alle für die Erfüllung der Qualitätsvorgaben notwendigen Tests werden *vollständig automatisiert* ausgeführt.
- Die gesamte notwendige Dokumentation wird automatisch generiert.
- Das ermöglicht, das *jeder Change* (Commit) am System Under Test auf “Knopfdruck” *getestet und validiert werden kann.*

Dies umfasst

- Testplan
- Testprotokoll
- Traceability Matrix
- Schnittstellen-Dokumentation
- Software-Dokumentation
- Release-Notes
- Anforderungsdokument
- Designdokument
- Review-Protokolle

Warum?

- **“It’s done, when it’s done”**
  - *Agil im regulativen Umfeld macht erst dann richtig Sinn, wenn alle Aufgaben, auch die formalen erledigt sind.*
- Frühes Feedback über Auswirkungen von Changes auf die Validierung für den Entwickler (KnowHow-Halbwertszeit)
- Frühzeitiges Einbinden der Entwicklung in regulative Themen
- Grenzen zwischen Testing und Entwicklung verschwimmen

Beispiel: Live-View ins Projekt

Id	Anforderung		Risiko	Test	Klassifizierung	Massnahme
R100	Wenn alle Signalquellen (v1, v2, v3) den gleichen Eingangswert liefern, soll das System den Eingangswert als Ausgangswert v0 auf der Anzeige darstellen		Das System zeigt einen falschen Messwert an	T105, PASSED	1A	Testfall
R101	Wenn 2 Signalquellen den gleichen Eingangswert liefern und eine Eingangsquelle einen abweichenden Wert liefert, so soll der Ausgangswert v0 den Eingangswert der gleichen Signalquellen darstellen und gleichzeitig die gelbe Warnlampe aktivieren.		Das System zeigt nicht den Wert der 2 gleichen Signalquellen	T101, PASSED *Stelle sicher, dass das System den gleichen Wert der 2 verbliebenen Eingangsquellen darstellt, wenn der Wert der dritten Eingangsquelle abweicht.*	1A	Testfall
R102			Das System aktiviert nicht die gelbe Warnlampe	T102, FAILED	1C	kein Qualitätsrisiko
R103	Wenn alle Singalquellen einen abweichenden Wert liefern, so soll das System als Ausgangswert v0 den Wert "0" darstellen und gleichzeitig die rote Warnlampe aktivieren.		...	R101, NOTIMPLEMENTED		

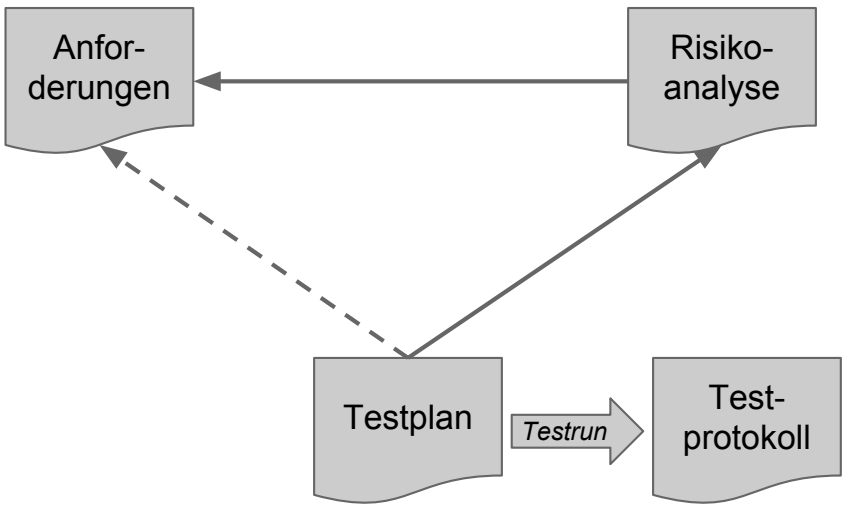
Fragen an das (Live-) Reporting

- Welche Anforderungen / Risiken sind nicht mit Tests verlinkt?
- Welche Anforderungen / Risiken haben noch keinen implementierten Test?
- Welche Anforderungen / Risiken sind mit fehlschlagenden Tests verknüpft?

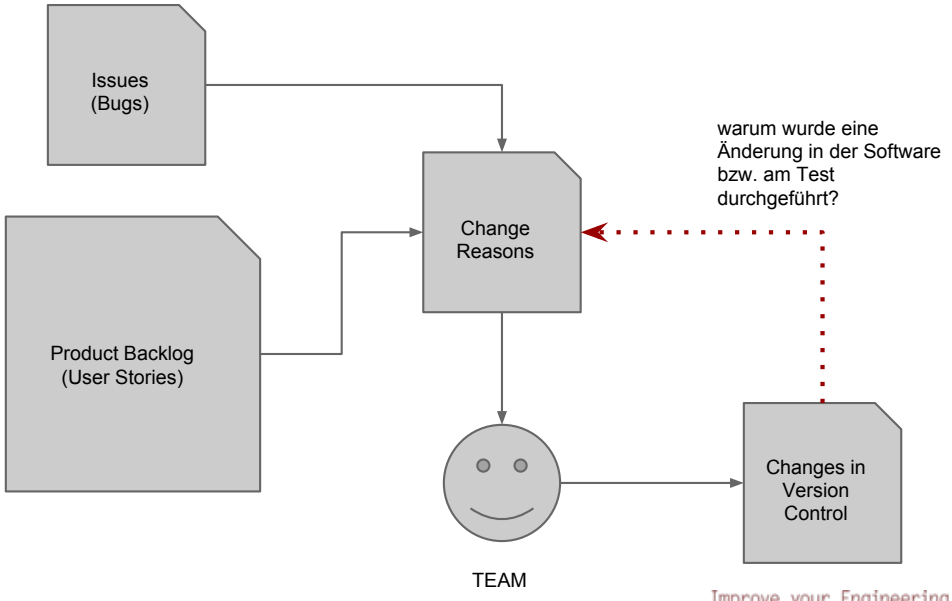
Ein Feature-Test

```
@R101, @REVIEWED
Feature: T101 One_Input_Deviation
    Stelle sicher, dass das System den gleichen Wert der 2
    verbliebenen Eingangsquellen darstellt, wenn der Wert der
    dritten Eingangsquelle abweicht.
    Scenario: T101 V1_Input_Deviation
        Given I supply 50 as v1
        And I supply 35 as v2
        And I supply 35 as v3
        Then I want to see 35 in the display
        And I want to see the yellow warning light
    Scenario: T101 V2_Input_Deviation
        ...
```

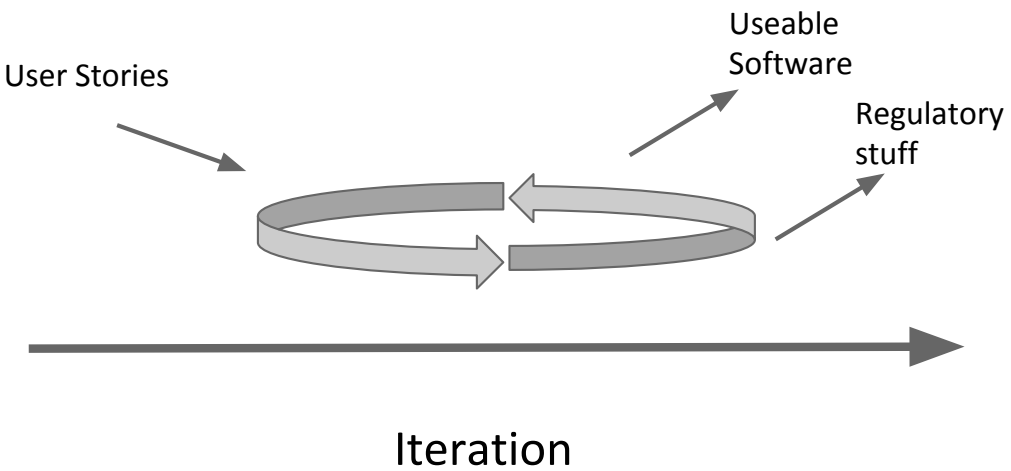

Artefakte



managing changes



Ein agiler Sprint im regulativen Umfeld



Aufwand für “Changes”

Funktionaler Aufwand:

- Analyse
- Architektur
- Implementierung
- Refactoring
- Test

Formaler Aufwand:

- **Review**
- Change Management
- Anforderungen
- Testplanung
- Tracing
- Dokumentation
- Release Notes, ...

Aufwand für “Changes”

Funktionaler Aufwand:

- Analyse
- Architektur
- Implementierung
- Refactoring
- Test

Formaler Aufwand:

- **Review**
- Change Management
- Anforderungen
- Testplanung
- Tracing
- Dokumentation
- Review
- Release Notes, ...

DEFINITION OF DONE

Die Phasen eines "Changes"

Analyse	Anforderung	Entwurf	Implementierung	Test	Freigabe
	Story1				
Story3					
		Story4			
		Story5			
			Story6		
				StoryA	
				StoryB	
					StoryC

Organisation in Iterations/Sprints

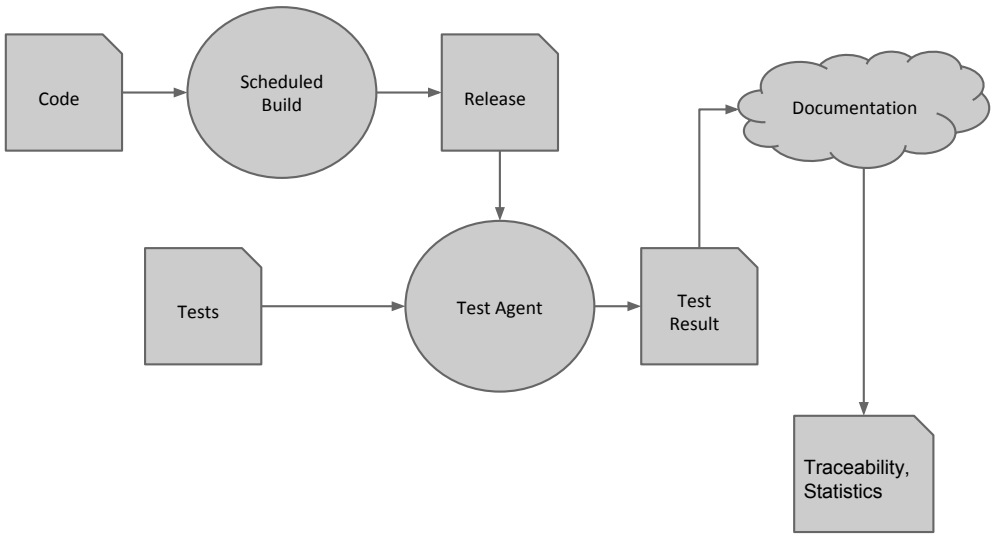
	Anforderung	Entwurf	Implementierung	Test	Freigabe
	Story1				
Story3					
		Story4			
		Story5			
			Story6		
				StoryA	

Was packe ich in einen Sprint ( )?

Der Ablauf aus der Sicht des Tests

1. Analyse der Anforderung
2. Risikobewertung
3. Testplanung (welche Testfälle benötige ich?)
4. Verlinkung der Test mit den Risiken / Anforderungen
5. Testimplementierung
6. Testrun
7. Auswertung der Ergebnisse
8. Tracebility Analyse
9. Review

Daily Statistics



Erfahrungen

Erfahrungen

- Der Aufwand ist abhängig von der Erfahrung des Teams mit agilen und regulativen Projekten.
- Ein eingespieltes Team hat effiziente Abläufe.
- Nicht alle im Team müssen Vorkenntnisse mitbringen, 2 oder 3 Zugpferde mobilisieren das Team
- Qualtät & Agil ist eine TEAM-Fähigkeit, keine PROJEKT-Fähigkeit
- DO CRASH INVESTIGATIONS

Anforderungen an das Tooling

- Unterstützung von eindeutigen Ids für Anforderungen und Test Cases
- versionierte Verlinkung zwischen Anforderungen und TestCases
- Einfacher export / import der Daten ohne Informationsverlust
- Änderungsverfolgung für Anforderungen und Test Cases mit leistungsfähigem Diff auf Dokumenten- und Anforderungsebene
- einfache Report-Generierung

Speicher

- Performance / Skalierbarkeit
- Definition of Done - Mit oder ohne Tests?
- Agile Code Reviews
- Flexible Teammitglieder können jede Task erledigen.
- Version Control: Test und Entwicklung in einem Projekt?

Links

- Behaviour Driven Development (BDD)
- Cucumber (a BDD implementation for Ruby)
- Bridging the Communication Gap