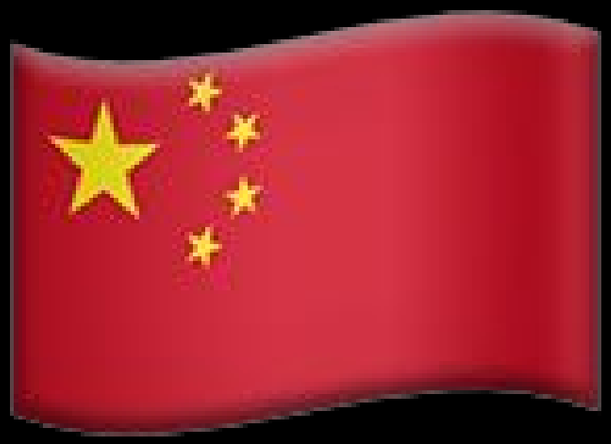
Agile Softwareentwicklung

Weil es mehr braucht,
als einen Kicker im Büro

18 Jahre
Agiles Manifest

warum eigentlich agil?



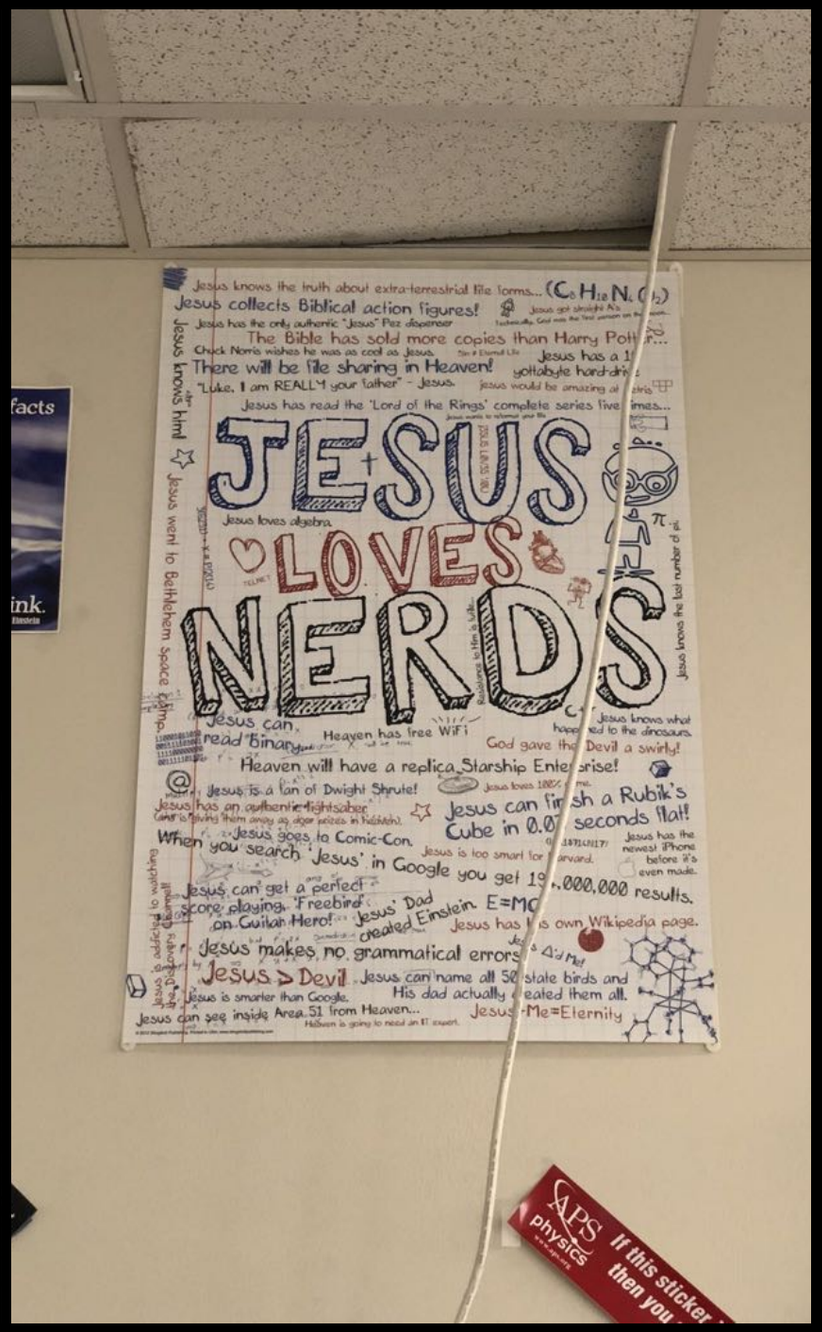
Don‘t make me…

…ask

…think

…wait

Komplettes Ökosystem



Fehlerkultur und Qualität

“Im Vergleich zu China ist Silicon Valley ein Kindergarten“

Ziel + Plan + System

Agile China

Schlüsseltechnologien

Innovation + Qualität

Erfolgsfaktoren

- 1 -

Grundlagen nutzen

- Agile = Qualität
- Testmethodiken
- Statische Analyse
- Clean Code & Reviews
- (Test)automatisierung

Teststufen

- Unittests
- Integrationstests
- Akzeptanztests
- E2E-Tests

- Unterschiedlich:
  - Fokus
  - Testziel
  - Gefundene Fehler

Unittests

- Feedback!
- Klasse/Methode/funktionale Einheit
- keine Abhängigkeiten (keine DB, Files, Systeme)
- schnell (fast feedback)
- Test one thing (single assumption)
- Qualität über Quantität (Testentwurfs-methoden, ~~viel hilft viel~~)
- Fehlerzustand schnell erreichbar
- Testbasis: Code & Struktur

Design for Testability!

Integrationstests

- Verschiedene Ebenen:
  - Integration der Units
  - Integration der Komponenten
  - Integration der Systeme
- Schnittstellentests
- Testbasis: Architektur/Design

Akzeptanztests

- Prüfung der Akzeptanzkriterien
- + links, rechts, negativ (Testentwurfsmethodik)
- Verknüpfung Fachlichkeit & Code
- Behavior-Driven Development
- Acceptance Test-Driven Development

E2E-Tests

- Test auf der Oberfläche
- (oder knapp darunter -> Framework)
- Testbasis: Geschäftsprozesse, User-Stories, Epics
- Stark (geschäfts)-risikobasiert

Testarten

	Funktionalität	Effizienz	Sicherheit	…
Unittest	x			x
Integrations-test	x	x		
Akzeptanz-tests	x	x	x	
…				

Testquadranten

Fachlich

E2E-Tests

Explorative Tests

Akzeptanztests

Team-unterstützend
(expected)

Produkt-kritisierend
(unexpected)

Integrationstests

Last- und
Performancetests

Unittests

Sicherheitstests

Technisch

- Äquivalenzklassen
- Grenzwerte
- Entscheidungstabellen
- Zustände
- Abläufe
- Pairwise

Statische Analysen

- Frühzeitig Fehlerzustände/Mängel finden
- Robustheit & Wartbarkeit
- Einhaltung Programmierregeln
- Prüfung gegen State-of-the-Art-Pattern
- Metriken für die Zeit
  - Quantität
  - Qualität
  - Komplexität

Clean Code & Code Reviews

- Technische Schulden vermeiden
- Ständiges Optimieren

(Test)automatisierung

- Automatisieren, was automatisiert werden kann
- „If you automate a mess, you get automated mess.“

E2E-Tests

expl. Tests

Akzeptanztests

Integrationstests

Unittests

- 2 -

Mindset

- Qualität ganzheitlich
- Kundennutzen
- hohe Selbstverantwortung
- Be your own IT-Manager
- Lösungsorientiert
- Disziplin

- 3 -

Reflexion

- hohe Selbstreflexion
- Hinterfragen
- kritisch
- Anpassen
- eigener Weg

- 4 -

Integrales Team

- Durchlässige Rollen
- Interdisziplinär
- Wertschätzung

- 5 -

Think Out of the Box

- Nichtwissen
- Problem- vs. Lösungsebene
- Denkfallen (Bias)

„Probleme kann man niemals mit der Denkweise lösen, durch die sie entstanden sind“

- Albert Einstein -

- 6 -

Ganzheitlich

Vision

Zugehörigkeit

Identität

Werte/Glaubenssätze

Fähigkeiten

Verhalten

Umgebung

Aber wie kommt man da hin?

Zeit +

Energie +

Geduld

Coaching

+

Vorbild

Selbst-
verantwortung

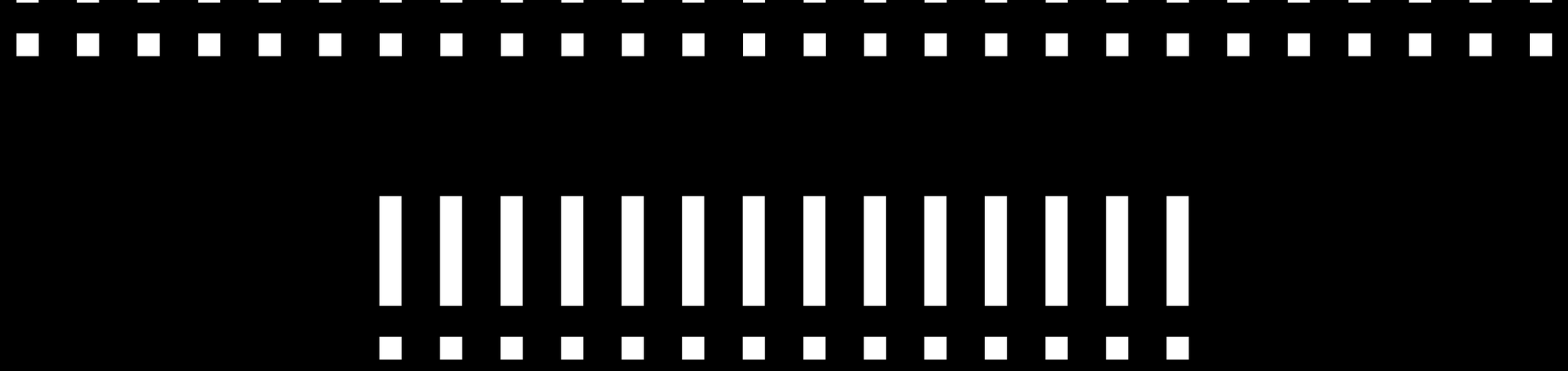
Agile Ideen statt Dogmen

Experimente

Retrospektiven

Retrospektiven!!!!!!

Retrospektiven



No Bullshit

„Machen ist wie Wollen

- nur krasser“

www.richard-seidl.com

mail@richard-seidl.com

Follow me: