Freie Universität Berlin

Agiles Testen

Oder Qualität ohne QA

Holger Schmeisky

17.01.2013

Inhalt

- Traditionelles Testen
- Agiles Testen

- Fallstudien
- Erfahrungsberichte

- Zusammenfassung
- Offene Fragen

Traditionelles Testen

- Testen Nach Entwicklung
- Unabhängigkeit
- Blickwinkel
- Testen schwierig
- Orakelproblem
- Metriken & Modelle

Agiles Testen

- Agile Entwicklung (z.B. XP, Scrum)
- Kurze Iterationen
- Changing Requirements
- Face-to-Face
- Working Software
- Unit Tests
- Keine Metriken

Agiles Testen

- Aber:
- Analyse XP, (etc.)
- Beinhalten konstruktive QA Praktiken
- Früher, öfter und integrierter
- Konstruktiv statt destruktiv
- Keine Integrationsphase am Ende
- Wenig umfassendere QA Praktiken

Zusammenfassung

- Testphase, Unabhängige Tester, Spezifikation
- Agiles Testen ganz anders
- Funktioniert agil überhaupt?
- Funktioniert traditionell überhaupt?

Empirische Studien

- Wenig Quellen zu agilem Testen
- Agile adoption Geschichten indirekt

- Fallstudien mit wissenschaftlicher Methodik
- Erfahrungsberichte

- 10 Quellen
- Gucken wo Gemeinsamkeiten sind

Fallstudie: Talby

Titel	Agile software testing in a large-scale project
Kontext	Israeli Air Force, Information System, kleines Team mit 1 Tester
Frage	Verlauf erstes XP Projekt
Methodik	Quantitative Projektdaten, Reflektionsmeetings, Fragebögen, Interviews, Dokumentenanalyse

Fallstudie: Talby

Beobachtungen

- Unabhängiger Tester unwichtige Fehler
- Entwicklertests sehr gut
- Ungetestete Features zählen als unvollständig zwingt Entwickler
- Kurze Feedback loop verringert Fix-Zeit

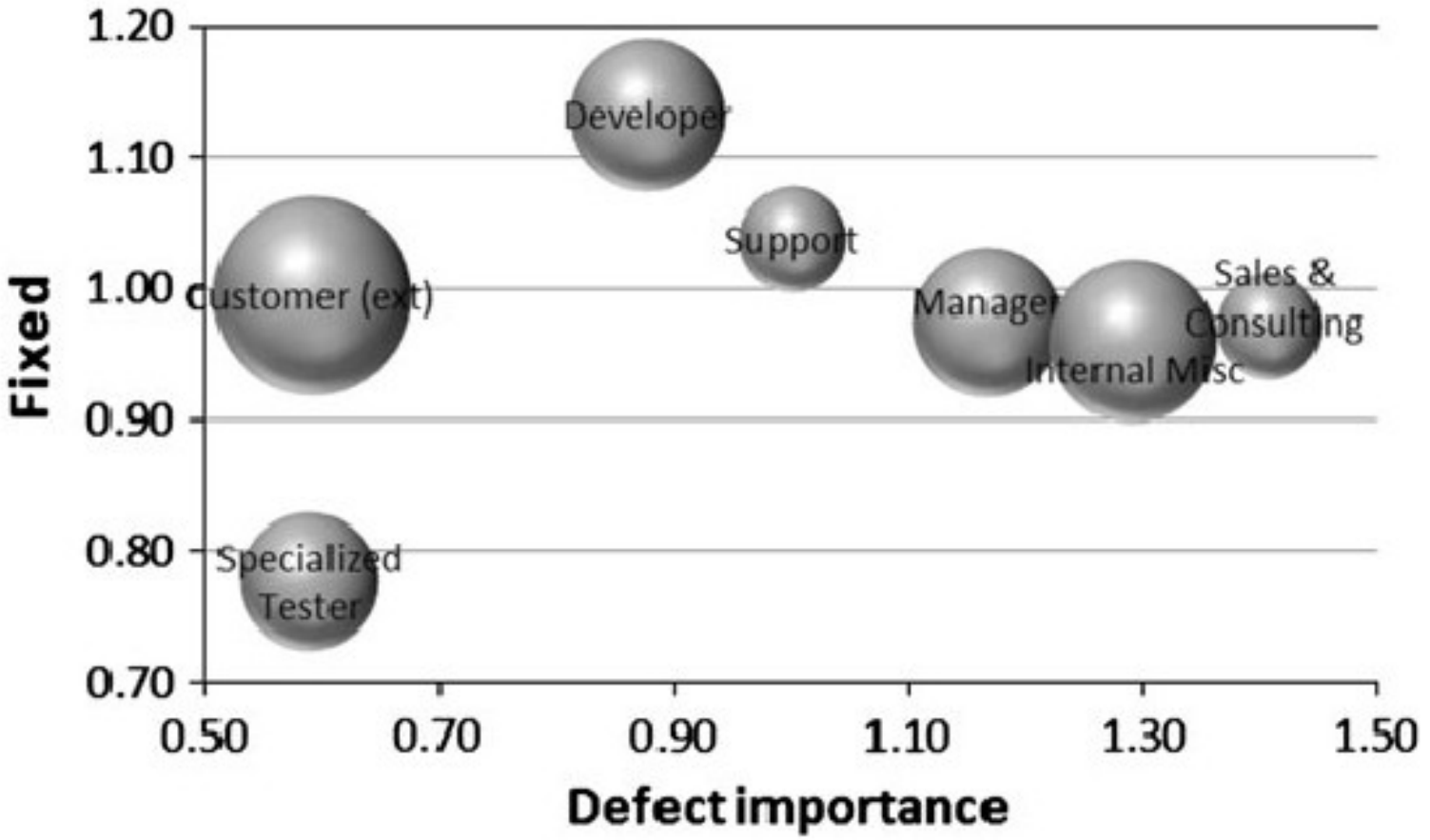
Ergebnisse

- Agiles Testen funktioniert gut
- Design for testability
- Traditionelle QA nötig bei Agil?

Fallstudie: Mäntilä

Titel	Who tested my software? Testing as an organizationally cross-cutting activity
Kontext	Explorative Fallstudie von 3 Softwarefirmen, Komplexe SW
Frage	Welche Angestelltengruppen testen Software?
Methodik	Dokumentenanalyse, Defektdatenbank, Interviews

Fallstudie: Mäntilä



"Specialized testers are more likely to find and report defects that are unlikely to occur in real use or those that have only minor effects." [Mäntilä]

Fallstudie: Guo

Titel	Characterizing and predicting which bugs get fixed: An empirical study of Microsoft Windows
Kontext	Microsoft, Bugdatenbank von Windows Vista
Frage	Einfluss von Eröffner (etc.) auf Wahrscheinlichkeit, dass Bug gefixt wird
Methodik	Defektdatenbank zusammen mit Organisationsstruktur, Umfrage

Fallstudie: Guo

Ergebnisse Je größer Distanz (geographisch, organisatorisch), desto geringer WK auf Fix

Fallstudie: Kettunen

Titel	A study on agility and testing processes in software organizations
Kontext	Fallstudie in 12 agilen Softwarefirmen
Frage	Welche Auswirkungen hat agile Entwicklung auf Testprozess- und Aktivitäten?
Methodik	Grounded Theory Research, Interviews mit Designern, Managern, Testern

Fallstudie: Kettunen

Beobach-tungen	Org's wo Entwickler testen Org's mit Testern in Teams Org's mit seperater Test-Einheit
Ergebnisse	• Alle haben zuwenig Testresourcen • Tests weggelassen wenn kein Tester • Entwickeln wichtiger gesehen als Testen • Tests in agilen Methoden früher → frühes Feedback wertvoller

Bericht: Marschall

Titel	Transforming a Six month release cycle to continuous flow
Kontext	Conject.com Immobilienportal Agile Adoption
Geschichte	• 6 monatige Releasezyklen mit Testphase am Ende • Nie alle Bugs gefixt, Notreleases • Entwickler kein Gefühl für Verantwortung • Schrittweise kleinere Releases

Bericht: Marschall

Beobachtungen

- Schneller Release deckte Fehler schnell auf
- Entwickler mussten selber testen
- Mussten Tests automatisieren
- ”Done” = Released
- Entwickler haben Verantwortung für Qualität übernommen

Bericht: Sumrell

Titel	From Waterfall to Agile - How does a QA Team Transition?
Kontext	Großes Team bei IBM, Scrum Adoption
Geschichte	• 6-monatige Release Zyklen • Großer Teil davon Testen • Nur QA verantwortlich für Qualität • Hauptsächlich Unit Testing

Bericht: Sumrell

Beobachtungen

- Entwickler mussten Unit Tests schreiben
- Haben damit Verantwortung für Qualität übernommen
- Sinnvoll Tester und Entwickler parallel arbeiten

Bericht: Google

Titel	How Google Tests Software
Kontext	Google Testing 2004 - heute
Geschichte	• Viele, schnelle Releases • 1000 Entw. - 50 Tester • Hauptsächlich UI Tests • Fing an zu schmerzen

Bericht: Google

Beobacht-ungen

- Keine Tester ”Software Engineer in Test”
- Projekte konkurrieren um SWTs
- Entwickler müssen Tests schreiben – klappt gut
- Wenige SWTs nur für Testability Features
- ”Testing Services” umbenannt in ”Engineering Productivity” brachte Kulturwandel

Bericht: Amazon

Titel	Accidental Adoption: The Story of Scrum at Amazon.com
Kontext	1 Entwicklerteam bei Amazon
Beobach-tungen	• Entwickler testen selbst, weil seperate Tester selten • Funktioniert gut • Dadurch Scrum sehr einfach

Bericht: Shaye

Titel	Transitioning a team to agile test methods
Kontext	>100, verteiltes Team bei IBM, >1 Million Zeilen Legacy code
Beobach-tungen	• Entwickler und Tester parallel • Design for Testability jetzt möglich

Bericht: Stolberg

Titel	Enabling Agile Testing through Continuous Integration
Kontext	3er Team mit 1 Tester
Beobach-tungen	• Scheitert erst mit trad. Testen • (Selber Tester) findet agiles Testen viel besser • Würde nie wieder zurückgehen

Auflistung

- Fallstudien:
  - Talby, Mäntilä, Guo, Kettunen
- Erfahrungsberichte:
  - Marschall, Sumrell, Google, Amazon, Shaye, Stolberg

Ergebnisse

- Traditionelles Testen oft ineffektiv [Talby, Mäntilä, Marschall]
- Entwicklertests allein ausreichend [Talby, Google, Amazon]
- Design for Testability [Talby, Shaye, Google]
- Schnelles Feedback gut [Talby, Marschall]
- Agiles Testen Schwierig [Kettunen, Marschall, Sumrell, Google]

Erfolgsfaktoren

- Tester selten machen [Google, Amazon]
- Schnelles Feedback [Google, Marschall, Talby]
- Entwicklertests [Talby, Sumrell, Google]
- Tester nur Testability [Google]
- Definition of Done [Talby, Marschall]

Offene Fragen

- Warum geistert trad. Testen noch herum?
- Wie wichtig sind automatisierte Tests bei Agil?
- Kann man immer ein bischen davon anwenden?
- Wie verändert Continuous Delivery Entwickler?
- Welche Faktoren wichtig für gute Umsetzung?
- ...

Freie Universität Berlin

Agiles Testen

Oder Qualität ohne QA

Holger Schmeisky
17.01.2013

”Heute geht es um Agile Testen, oder wie man fehlerarme Software ohne klassische Qualitätssicherung baut.”

- keine Prozesse, CMMI, Agilität
- Studien und Erfahrungsberichte aus der Praxis

”Umfrage: - wer hat entwickelt? - gabs da tester/QA?”
Üblicherweise in Firmen: Entwickler und Tester
Persönliche Erfahrung: Einstellung: gebe Verantwortung ab, mache mehr Fehler
Agile Methoden entfernen Tester, geht aber auch ohne Agil
Literatur abgegrast und zeigt in dieselbe Richtung:
- traditionelles Testen ist ineffektiv
- kann besser gemacht werden mit mehr Verantwortung beim Entwickler

Freie Universität Berlin

Inhalt

- Traditionelles Testen
- Agiles Testen

- Fallstudien
- Erfahrungsberichte

- Zusammenfassung
- Offene Fragen

”Dazu erzähl ich erst was zu tradit...”

Freie Universität Berlin

Traditionelles Testen

- Testen Nach Entwicklung
- Unabhängigkeit
- Blickwinkel
- Testen schwierig
- Orakelproblem
- Metriken & Modelle

”Für trad. Testen gibt es einige Prinzipien, die allgemein anerkannt sind:”

- Testphase nach abgeschlossener Entwicklung [wasserfall]
- ”Programmierer sollten vermeiden zu versuchen ich ...” (2004)
- Destruktiv, nicht konstruktiv
- Tester spezielles Skillset
- Was sind richtige Ergebnisse?
- Metriken um Qualität zu bewerten?

Freie Universität Berlin

Agiles Testen

- Agile Entwicklung (z.B. XP, Scrum)
- Kurze Iterationen
- Changing Requirements
- Face-to-Face
- Working Software
- Unit Tests
- Keine Metriken

”Es gibt aber auch noch eine andere Gruppe von Entwicklungsprozessen, wo es aus sicht trad probleme gibt”

- Endphase testen schwer bei kurzer Iteration (aber auch bei Trad.)
- Trad. Ist testen nach Spezifikation
- nur in Köpfen von Entwickler & Busi
- sw muss von anfang an getestet sein
- statt oracle, difficult, independency
- Agile Methoden; Qualität durch Produkt

”Aus Sicht Trad. Furchtbare Idee”

Freie Universität Berlin

Agiles Testen

- Aber:
- Analyse XP, (etc.)
- Beinhalten konstruktive QA Praktiken
- Früher, öfter und integrierter
- Konstruktiv statt destruktiv
- Keine Integrationsphase am Ende
- Wenig umfassendere QA Praktiken

”Dann habe ich ein Paper gefunden, wo geguckt wird, ob agile vielleicht doch QA enthalten, nur anders”

- verschiedene Methoden, XP wichtig
- Agil (insb. XP) enthalten sehr viele konstruktive Qualität techniken
- zB unit tests, daily meetings, design Durchsichten, regular builds, integration,
- die Rollen sind sehr stark verzahnt und kommunizieren sehr oft
- fehlt langlauftests, test infra & tools, reliability,

Freie Universität Berlin

Zusammenfassung

- Testphase, Unabhängige Tester, Spezifikation
- Agiles Testen ganz anders
- Funktioniert agil überhaupt?
- Funktioniert traditionell überhaupt?

”Also, nochmal zusammengefasst: Trad. Testen heißt:...”

Agiles pfeift darauf, mehr konstruktive Praktiken, mehr Verantwortung beim Entwickler

”Die Frage die sich stellt:?”

Freie Universität Berlin

Empirische Studien

- Wenig Quellen zu agilem Testen
- Agile adoption Geschichten indirekt

- Fallstudien mit wissenschaftlicher Methodik
- Erfahrungsberichte

- 10 Quellen
- Gucken wo Gemeinsamkeiten sind

”Ich habe nur wenige Quellen zu agilem Testen gefunden”

- erzähl jetzt einzeln was zu studien, am Ende wird gesammelt
- so gedacht, zwischendurch abzuschweifen, dann ganz schnell wieder rein
- 10 Quellen
- am Ende wird alles wiederholt
- Tabellen sehen immer gleich aus

Freie Universität Berlin

Fallstudie: Talby

Titel	Agile software testing in a large-scale project
Kontext	Israeli Air Force, Information System, kleines Team mit 1 Tester
Frage	Verlauf erstes XP Projekt
Methodik	Quantitative Projektdaten, Reflektionsmeetings, Fragebögen, Interviews, Dokumentenanalyse

”Als erstes eine Fallstudie von einem XP Projekt bei der Israeli Air Force ...”

Programm ist ”kritisch”, also wichtig
- Forscher von Uni

Hohe Qualität, da quantitative Daten direkt aus Projekt, viele verschiedenartige Quellen

Freie Universität Berlin

Fallstudie: Talby

Beobacht-ungen	• Unabhängiger Tester unwichtige Fehler • Entwicklertests sehr gut • Ungetestete Features zählen als unvollständig zwingt Entwickler • Kurze Feedback loop verringert Fix-Zeit
Ergebnisse	• Agiles Testen funktioniert gut • Design for testability • Traditionelle QA nötig bei Agil?

”Was haben die rausgefunden? Der unabh. Tester, als er unabh. Gearbeitet hat, hat nur ...”

- ganz krass: in 3 Wochen nur 2 kleine Spezi fehler
- Das war so, weil unit tests alles gefangen
- Das lag vielleicht daran, dass ungetestet...

Freie Universität Berlin

Fallstudie: Mäntilä

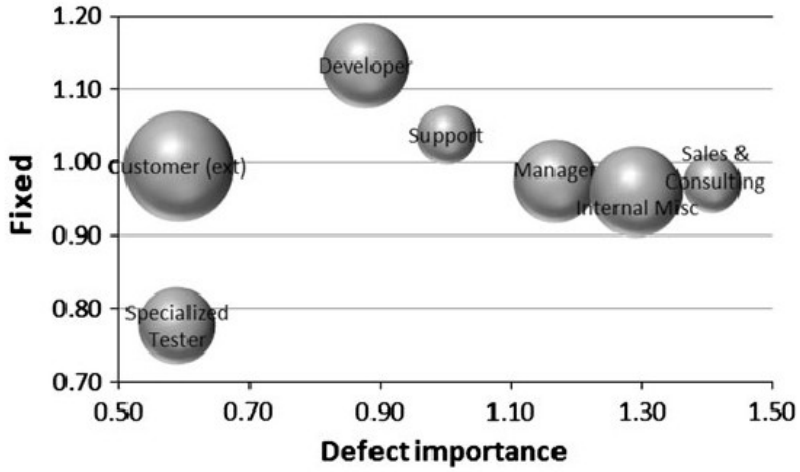
Titel	Who tested my software? Testing as an organizationally cross-cutting activity
Kontext	Explorative Fallstudie von 3 Softwarefirmen, Komplexe SW
Frage	Welche Angestelltengruppen testen Software?
Methodik	Dokumentenanalyse, Defektdatenbank, Interviews

”Als nächstes eine Fallstudie, die sich mit einer anderen Frage beschäftigt, nämlich wer […] und inwiefern stimmt das mit der trad. Sicht überein?”

- Ingenieursentwurfs Sw und gEschäftssoftware – echte Männerarbeit, keine FB Spiele
- wie wichtig sind die Defekte (nach Gruppe) und wie oft werden sie gefixt
- hohe Validität, da mehrere Firmen, verschiedenartige Datenquellen (zB Hypothesen gemacht und dann Manager gefragt, ob stimmen)

Freie Universität Berlin

Fallstudie: Mäntilä



"Specialized testers are more likely to find and report defects that are unlikely to occur in real use or those that have only minor effects." [Mäntilä]

”Dabei fand ich dieses Bild am interessantesten:”

Ist nur von 2 Firmen, eine hatte keine Tester

Entwickler fixen am ehesten eigene Fehler
Interne Anwender finden jede Menge
Tester finden unwichtigste

Manager gefragt, ob das stimmt, stimmten zu

Freie Universität Berlin

Fallstudie: Guo

Titel	Characterizing and predicting which bugs get fixed: An empirical study of Microsoft Windows
Kontext	Microsoft, Bugdatenbank von Windows Vista
Frage	Einfluss von Eröffner (etc.) auf Wahrscheinlichkeit, dass Bug gefixt wird
Methodik	Defektdatenbank zusammen mit Organisationsstruktur, Umfrage

”Eine Studie auf ganz besonderen Daten: Der Bugdatenbank von Windows Vista”

- relativ hohe Validität, da erstklassige Daten

Freie Universität Berlin

Fallstudie: Guo

Ergebnisse Je größer Distanz (geographisch, organisatorisch), desto geringer WK auf Fix

Spricht dafür Tester nah bei Entwicklern zu halten

Beispiel: QA Abteilung in Indien von vornherein verurteilt

Freie Universität Berlin

Fallstudie: Kettunen

Titel	A study on agility and testing processes in software organizations
Kontext	Fallstudie in 12 agilen Softwarefirmen
Frage	Welche Auswirkungen hat agile Entwicklung auf Testprozess- und Aktivitäten?
Methodik	Grounded Theory Research, Interviews mit Designern, Managern, Testern

”Als letztes eine Fallstudie die Probleme an agilem Testen aufzeigt: ”

- methodik: Fallstudie mit Hypothesen und Belegen dafür
- Firmen verschieden viel Agile Adoption (manche nur unit tests oder expl. testen, manche komplettes SCRUM)
- Studie vorallem beschäftigt welche Faktoren einfluss auf Test haben
- Validität hmm – richtiges Thema, aber Firmen nur wenig agil (zB nur unit tests, nur scrum); keine quant.

Freie Universität Berlin

Fallstudie: Kettunen

Beobach-tungen	Org's wo Entwickler testen Org's mit Testern in Teams Org's mit seperater Test-Einheit
Ergebnisse	• Alle haben zuwenig Testresourcen • Tests weggelassen wenn kein Tester • Entwickeln wichtiger gesehen als Testen • Tests in agilen Methoden früher → frühes Feedback wertvoller

”Was haben die herausgefunden?”

- diese org. Formen gefudnen
- Ursache der meisten Problem, Prioritäten
- keine Tester keine Lösung
- Tests früher, daher werden manche Probleme früher gefunden die sonst später schwierig wären

Freie Universität Berlin

Bericht: Marschall

Titel	Transforming a Six month release cycle to continuous flow
Kontext	Conject.com Immobilienportal Agile Adoption
Geschichte	• 6 monatige Releasezyklen mit Testphase am Ende • Nie alle Bugs gefixt, Notreleases • Entwickler kein Gefühl für Verantwortung • Schrittweise kleinere Releases

”Das wars mit wissenschaftlicher Arbeit, jetzt kommen Berichte vom Schlachtfeld. Auch aus Konferenzen, zB AGILE”

- Kunden als Tester
- das funktionierte ziemlich schlecht
- haben halt was entwickelt, ein halbes jahr später ausgeführt

Freie Universität Berlin

Bericht: Marschall

Beobacht-ungen	• Schneller Release deckte Fehler schnell auf • Entwickler mussten selber testen • Mussten Tests automatisieren • ”Done” = Released • Entwickler haben Verantwortung für Qualität übernommen

”Am Ende hat dann gut funktioniert, die haben alle 3 Tage released in hoher Qualität.”

- richtig gut, hat das erst mit done = released funktioniert, weil Entwickler sich richtig mühe gaben
- Erst langsamere Entwicklung wegen Angst, Qualität aber viel besser
- Schneller Release

Freie Universität Berlin

Bericht: Sumrell

Titel	From Waterfall to Agile - How does a QA Team Transition?
Kontext	Großes Team bei IBM, Scrum Adoption
Geschichte	• 6-monatige Release Zyklen • Großer Teil davon Testen • Nur QA verantwortlich für Qualität • Hauptsächlich Unit Testing

”Bei IBM gabs ein ähnliches Problem”

Freie Universität Berlin

Bericht: Sumrell

Beobacht-ungen	• Entwickler mussten Unit Tests schreiben • Haben damit Verantwortung für Qualität übernommen • Sinnvoll Tester und Entwickler parallel arbeiten

”Sie wollten vorallem weniger manuelle Tests. Dafür mussten ...”

- anscheinend Effekt von Scrum

Freie Universität Berlin

Bericht: Google

Titel	How Google Tests Software
Kontext	Google Testing 2004 - heute
Geschichte	• Viele, schnelle Releases • 1000 Entw. - 50 Tester • Hauptsächlich UI Tests • Fing an zu schmerzen

”Als nächstes eine Firma die wir alle lieben: Google. Die finden, sie können so toll testen dass jemand da ein Buch darüber geschrieben hat.”

2004 – das war so mit Dokumenten im Web und Hyperlinks dazwischen

Freie Universität Berlin

Bericht: Google

Beobacht-ungen	• Keine Tester ”Software Engineer in Test” • Projekte konkurrieren um SWTs • Entwickler müssen Tests schreiben – klappt gut • Wenige SWTs nur für Testability Features • ”Testing Services” umbenannt in ”Engineering Productivity” brachte Kulturwandel

”Sie hatten sowieso schon wenig Tester, die haben sie aber auch noch rausgeschmissen und dafür SWT ...”

Kulturwandel: Danach nicht mehr als Tester fürs Ende, sondern als supporter um von anfang an testabiltiy zu machen

Freie Universität Berlin

Bericht: Amazon

Titel	Accidental Adoption: The Story of Scrum at Amazon.com
Kontext	1 Entwicklerteam bei Amazon
Beobach-tungen	• Entwickler testen selbst, weil seperate Tester selten • Funktioniert gut • Dadurch Scrum sehr einfach

”Damit hier auch keine große Firma fehlt: Amazon. Da ist der Bericht aber nicht so ergiebig”

- Manager erzählt, wie er in seinem Team Scrum (völlig falsch) einführte und sich das nach und nach verbreitete und wie toll das alles ist
- nicht sehr spannend
- trotzdem hohe SW Qualität
- Amazon hostet das halbe Internet
- Scrum anscheinend führt mit zu agile testing

Freie Universität Berlin

Bericht: Shaye

Titel	Transitioning a team to agile test methods
Kontext	>100, verteiltes Team bei IBM, >1 Million Zeilen Legacy code
Beobach-tungen	• Entwickler und Tester parallel • Design for Testability jetzt möglich

”Jetzt nochmal IBM, aber nur ganz kurz”

- test findens gut, weil sie jetzt testabiltiy features einbauen lassen, was früher unmöglich war. Dadurch ihr testen viel einfacher und weniger UI lastig
- effizienzgewinn

Freie Universität Berlin

Bericht: Stolberg

Titel	Enabling Agile Testing through Continuous Integration
Kontext	3er Team mit 1 Tester
Beobach-tungen	• Scheitert erst mit trad. Testen • (Selber Tester) findet agiles Testen viel besser • Würde nie wieder zurückgehen

”Zuletzt habe ich noch eine Geschichte gefunden, wo einer erzählt wie er in seinem 3-Mann Team CI aufgesetzt hat”
- nicht so spannend

Freie Universität Berlin

Auflistung

- Fallstudien:
  - Talby, Mäntilä, Guo, Kettunen
- Erfahrungsberichte:
  - Marschall, Sumrell, Google, Amazon, Shaye, Stolberg

”So, geschafft! Jetzt müssen wir nur noch die Teile einsammeln und gucken, was für Punkte wie wichtig finden. Nochmal zur Erinnerung”

Israeli Air Force Projekt; Specialized Testers; Distanz; Nicht so Einfach

6-monate release transformation; Release Transformation; Engineering Productivity; Amazon hat auch keine Tester; Stolberg mag auch lieber parallel; IBM Team Tester parallel;

Freie Universität Berlin

Ergebnisse

- Traditionelles Testen oft ineffektiv [Talby, Mäntilä, Marschall]
- Entwicklertests allein ausreichend [Talby, Google, Amazon]
- Design for Testability [Talby, Shaye, Google]
- Schnelles Feedback gut [Talby, Marschall]
- Agiles Testen Schwierig [Kettunen, Marschall, Sumrell, Google]

”5 Ergebnisse, die ich mir rausgepackt habe, weil ich sie am wichtigsten finde. Wenn ihr welche vermisst, lasst uns drüber reden, das würde mich sehr interessieren”

Freie Universität Berlin

Erfolgsfaktoren

- Tester selten machen [Google, Amazon]
- Schnelles Feedback [Google, Marschall, Talby]
- Entwicklertests [Talby, Sumrell, Google]
- Tester nur Testability [Google]
- Definition of Done [Talby, Marschall]

”Was besonders interessant ist, ist wo der Unterschied zwischen ”hier macht mal” und agilem Testen hergestellt wird. Daher hier

Freie Universität Berlin

Offene Fragen

- Warum geistert trad. Testen noch herum?
- Wie wichtig sind automatisierte Tests bei Agil?
- Kann man immer ein bischen davon anwenden?
- Wie verändert Continuous Delivery Entwickler?
- Welche Faktoren wichtig für gute Umsetzung?
- ...