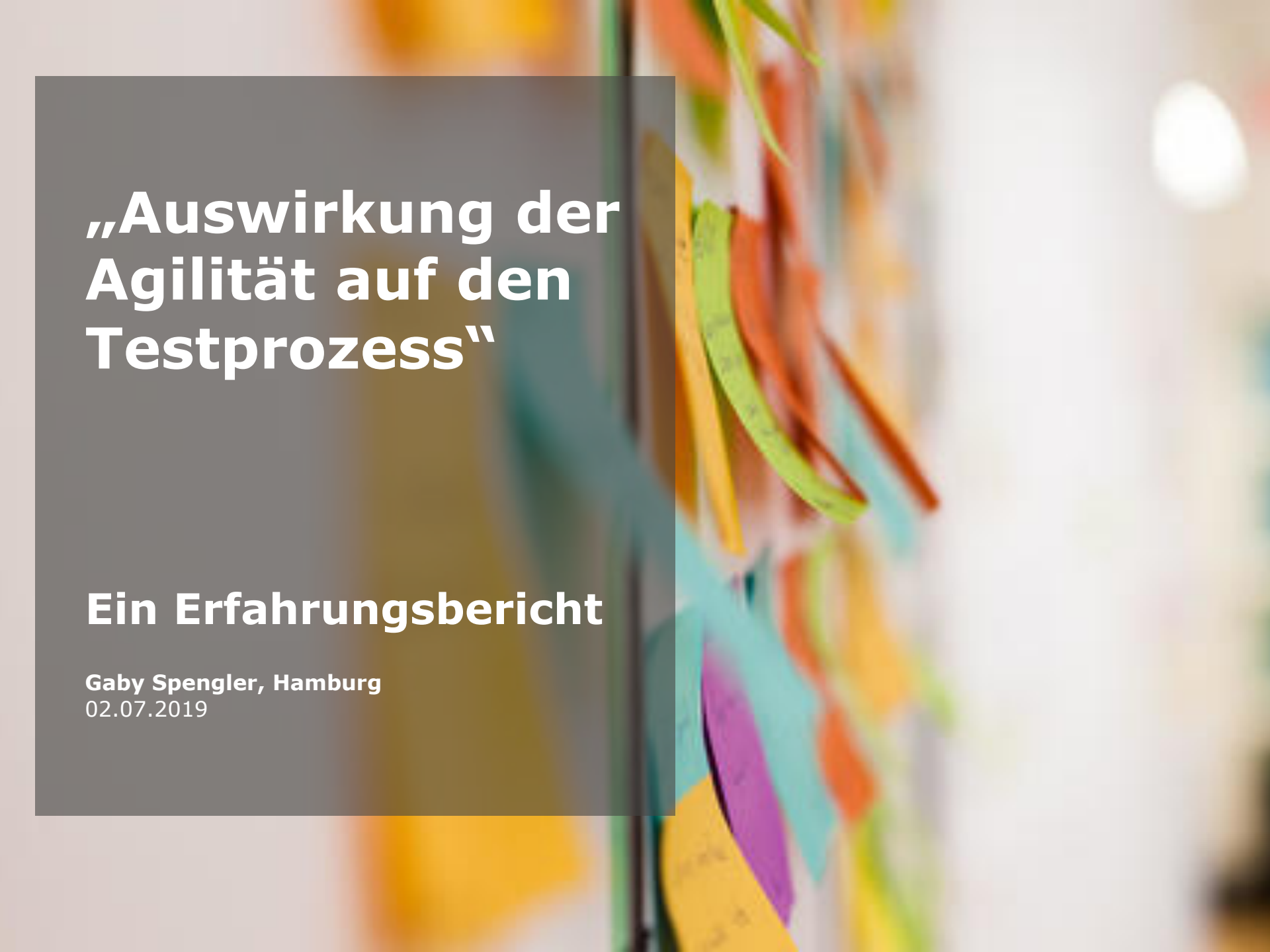


„Auswirkung der Agilität auf den Testprozess“

Ein Erfahrungsbericht

Gaby Spengler, Hamburg
02.07.2019

Die Ausgangssituation

Umfeld

- Bestandsführungssystem für Sach-Versicherungen (Standard)
- kundenindividuelle Anpassung durch Customizing und Programmierung
- ca. 23 Bestandskunden, Zahl steigend
- ca. 4 parallele Projekte (Neuentwicklung und/oder Update)
- Updates für Bestandskunden alle 1-2 Jahre
- 2-Schicht-Architektur (GUI - DB)
- kaum automatisierte Entwicklertests (Unit-/Integrationstests)
- Systemtest
  - Teilbereiche automatisiert
  - es wurden auch Funktionalitäten getestet, die eigentlich in die unteren Teststufen gehören
  - automatisiert wurde, wenn Zeit war
- manueller Abnahmetest

Projektablauf

- iteratives Vorgehen
- jeweils eigener Projektplan
  - Entwicklungsphasen
  - Testphasen
  - Auslieferungstermine
- Projekte ändern/erweitern bei Bedarf die Standardfunktionalitäten

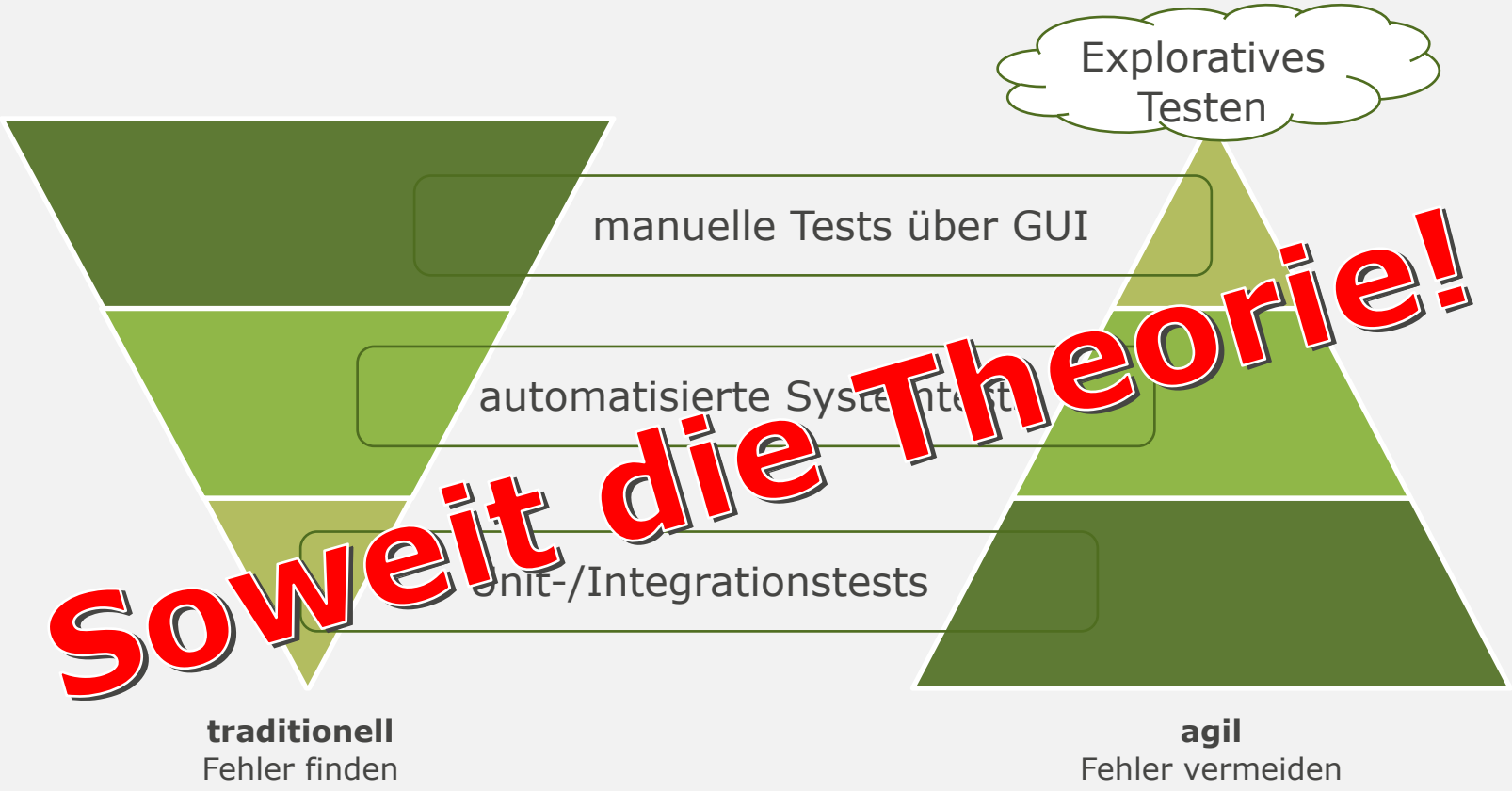
Die Hausforderung

Vorhaben

- Wechsel der GUI-Technologie
- Umstellung auf 3-Schicht-Architektur (GUI - **WS** - DB)
- Wechsel der Entwicklungsmethode (iterativ -> agil (Scrum/Kanban))
- 4-wöchige Sprints
- Auslieferung alle 2 Monate
- jedes 2. Release auch als Update für die Bestandskunden.
  - -> 3 Updates im Jahr (mit wenig Aufwand) anstatt alle 1-2 Jahre (Aufwand ähnlich Neuprojekt)
- Etablierung eines Kernelteams für die Standardentwicklung für alle Projekte (ca. 10 MA)
  - PO - Produktmanager/-architekt
  - AEs - GUI-/WS-/DBA
  - Tester - Fachberater, Testmanager

Die Ereignisse

Teststrategie/Testkonzept anpassen



Kürzere Entwicklungszyklen bedingen mehr Automatisierung!!!

-> Teststrategie ändert sich mit dem Vorgehensmodell.

Umgebungskonzept

- 3 Umgebungen je Kundenvariante und für den Standard
- EN - Entwicklung
  - Unittests (auch im Rahmen des Buildprozesses / Nightly Build)
  - evtl. einzelne man. Vorabtests bei Entwicklung komplexer Anforderungen
  - sonst keine Testaktivitäten
- FT - Fachlicher Systemtest
  - Gesamttest vor Auslieferung
  - Test der Tasks/User-Stories (Bestandteil der DoD)
  - Regressionstest
- AT - Abnahmetest
  - Test des Auslieferungsprozesses an die Kunden
  - Smoke-Test der Funktionalität

Test neuer Anforderungen

- **testbereite Tasks** (Neu, Change) werden in FT-Umgebung getestet
  - ständig und nicht in definierten Testphasen -> DoD
  - „PairTesting“ (AE und Tester) hat sich in einigen Fällen als sehr effizient herausgestellt
  - Problem: Nicht immer war klar, ob die Entwicklung schon in FT verfügbar ist. Probieren, kommunizieren -> Zeitverlust
  - Lösung:
    - Versionsnummern für WS und Client in Statuszeile der Anwendung
    - Eintrag in der Task, ab welcher Version (WS, Client) testbar

- Testfälle werden geprüft und in Regressionstestbestand aufgenommen

Fachlicher Systemtest (anfangs)

fachl. Systemtest vor Auslieferung (**anfangs**)

- Neuerungen/Änderungen manuell
- Regressionstest auch manuell durch das gesamte Team
  - Erweiterung fachl. Wissen aller (Funktionalität und Zusammenhänge)
  - Sensibilisierung aller für die Testbarkeit
  - Unterstützungsbereitschaft für die Testautomatisierung wächst mit dem „Leidensdruck“
  - **!!! Nicht anders machbar !!!**

Testautomatisierung (TA) - Setup

Testautomatisierung des Regressionstests

- Konzeption der Automatisierung
  - Bereitstellungsprozesse
  - Umgebung
  - Struktur
  - Wiederverwendbarkeit für Kundenvarianten, etc.
- Priorisierung der Reihenfolge
  - Risikoorientierung: wichtigste Funktionen zuerst
  - Aufwandsorientierung: aufwändige/komplexe Tests
- Anpassung der SW für bessere Testbarkeit
  (z. B. object properties für stabile Identifizierung)
- autom. Übernahme der Excel-Testfälle als Struktur der TestSuite (Testfall-ID und Beschreibung (Voraussetzung, Testschritte, erwartetes Ergebnis))

Testautomatisierungsumgebung/-ablauf

- mehrere virtuelle Testserver stehen zur Verfügung
- beim Start der TA wird autom. ein freier Testserver gesucht
- Vollautomatische Bereitstellung auf Basis variabler Quelle
  - Erstellung der DB als Struktur-Kopie einer Quell-DB
  - Client und WS aus Quell-Umgebung
  - Synchronisation der Automatisierungssoftware aus dem Repository
- Durchlauf aller Tests (Acception-Handling)
- E-Mail mit Testergebnis an ausgewählte Personen
- exkl. Testdaten je Testfall sind besser als gemeinsamer Testdatenbestand

Testautomatisierung - Hürden

- Anfangs viele Fehler in der TA selber -> TA instabil
  - Problem: hoher Analyseaufwand für Testergebnisse
  - Lösung: TA immer wieder durchführen und Probleme korrigieren

- Beim autom. Test vor Auslieferung wurden sehr viele Fehler gefunden
  - Problem: zu wenig Zeit um alle Fehler zu fixen
  - Lösung: wöchentl. Bereitstellung der Testumgebung
    -> Fehler werden früher gefunden

Testautomatisierung - Hürden II

- Für alle Abweichungen wurden Fehler-Tasks angelegt
  - Problem: Fehler aus TA wurden nicht sofort gefixt -> hoher Analyseaufwand bei nachfolgenden Testläufen. (bekannter/gemeldeter oder neuer Fehler)
  - Lösung: gemeinsamer Beschluss, dass Fehler aus TA mit höchster Prio zu fixen sind (auch „Schönheitsfehler"!)

- Idee: Einbau einiger autom. Smoketests in den Buildprozess der Entwicklungsumgebung; bei Fehler -> Abbruch
  - Problem: Entwickler weigerten sich: „Das ist eine Entwicklungsumgebung!"
  - Lösung: Smoketests nächtl. gegen die Entwicklungsumgebung gefahren und das Ergebnis den AEs "nur zur Info" mitgeteilt -> Fehler wurden immer zeitnaher gefixt

Fachlicher Systemtest (heute)

fachl. Systemtest vor Auslieferung (**heute**)

- Neuerungen/Änderungen manuell (wie bisher)
- Regressionstest fast vollständig automatisiert (für Standard)
  - Ergebnisse innerhalb von 2 Stunden
  - verlässliche Ergebnisse: „rot = Fehler in SuT" zu annähernd 100% korrekt
- Regressionstest für Kundenvarianten wird auf-/ausgebaut
  - Focus auf Nicht-Standardfunktionalität
- Planung
  - Kritisch: zeitgleiche Testphase aller Projekte
  - Kernelteam legt Testumfang für Standard (Neuentwicklung, Regressionstest) fest und führt die Tests durch
  - andere Teams ergänzen individuelle Tests und führen primär nur diese durch

Auswirkungen auf Testaktivitäten der Kunden

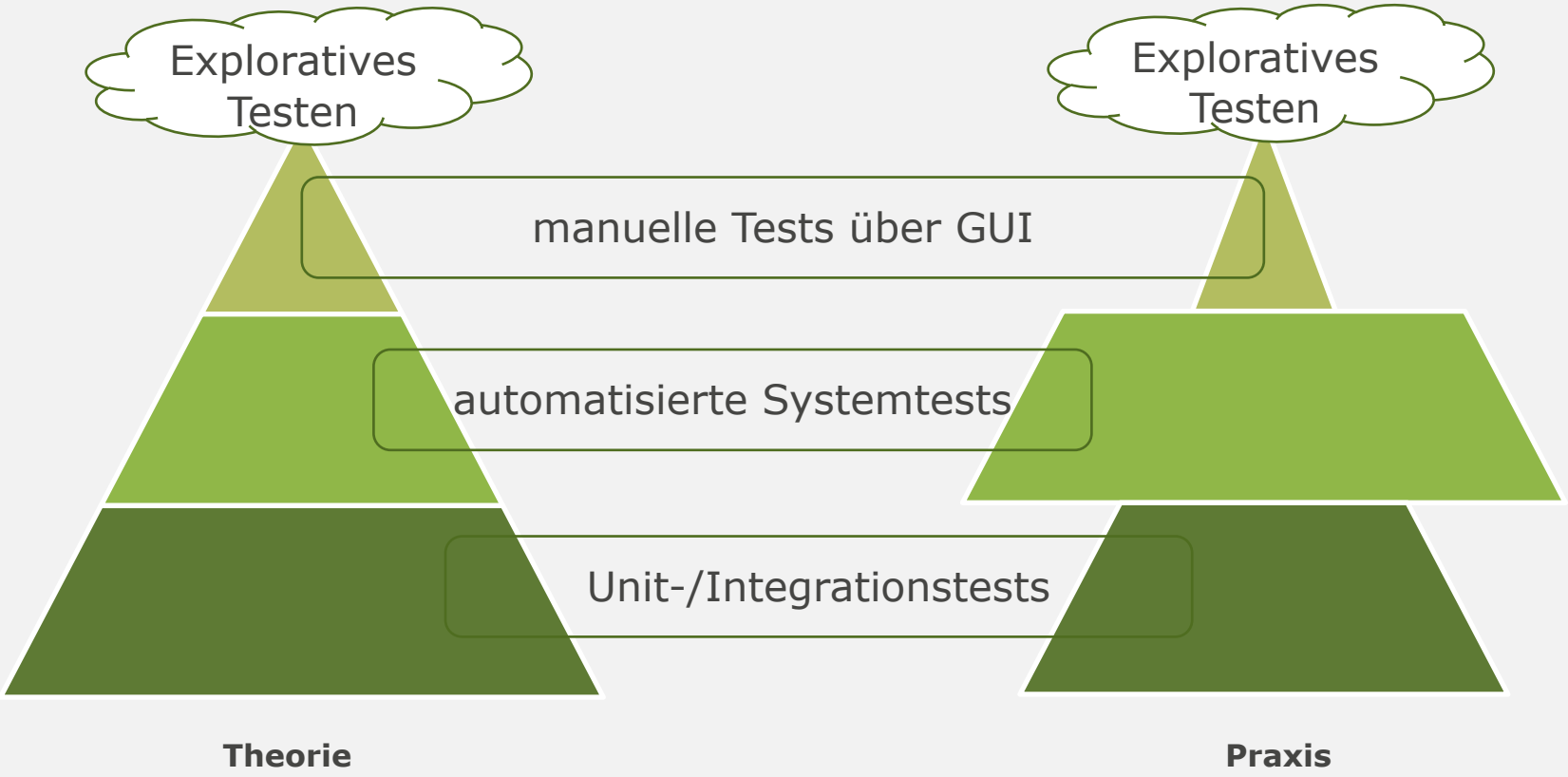
Hohe Dynamik im Versicherungsumfeld

- digitale Geschäftsmodelle
- Anbindung neuer Vertriebskanäle (Vergleichsportale, Vermittler, Makler (im 2 Tagesrhythmus))

schafft Akzeptanz für

- Aufbau der Testautomatisierung
- TA-Aufwand mittlerw. fester Bestandteil des Projektbudgets (ca. 10 - 15 %)
- neue autom. Testfälle entstehen auf beiden Seiten (intern, Kunde)
- Konzept für gegenseitigen Austausch neuer und angepasster Testfälle
- gemeinsame Planung/Abstimmung

Testkonzept - Theorie und Praxis



- **Rückstände bei autom. Unit-/Integrationstests aufgrund großer Anforderungsmenge**
- **überplanmäßiger Ausbau der autom. Systemtests aufgrund häufiger Seiteneffekte (z.B. größeres Fehlerpotential durch weitere Schnittstelle)**

- **!!! Fast vollständige Automatisierung des Regressionstests !!!**

Das Fazit

Automatisierung ist existentiell

- Ohne Automatisierung geht es nicht!
- Automatisierung erfordert Testbarkeit
  -> Testbarkeit erhöht Qualität!
- Automatisierung erfordert bessere Datenqualität
  -> Nutzt auch der Entwicklung

All diese Punkte **KANN** man auch mit anderen Entwicklungsmethoden umsetzen, mit agilen Methoden **MUSS** man es.

Erfolgsfaktoren

+ Konsens im/Unterstützung durch das ganze Team 
+ sehr gute Vollzeit-Testingenieure 
+ Risikoorientierung 
+ Pragmatismus -> „Die ~~perfekte~~ beste Lösung.“ * 
+ Flexibilität 
+ Engagement / Wille 
+ Ausdauer 

* brand eins Verlag, App-Titel 09/2015

Was wir gelernt haben

- Wenn das Fehleraufkommen zu hoch ist…
  - erhöhe die Anzahl der TA-Durchläufe.
  - fixe **ALLE** Fehler **sofort.**

- Sorge für ausreichende Hardwarekapazität, denn langsame TA ist nur bedingt brauchbar.

- **Halte durch, es lohnt sich !!!**

**„Nicht weil es schwer ist,
wagen wir es nicht,
sondern weil wir es nicht wagen,
erscheint es schwer."**

Lucius Annaeus Seneca (röm. Philosoph und Staatsmann)

Gaby Spengler

Die Techniker
Tel. 040 - 6909 - 2848
gaby.spengler@tk.de

Falls Sie noch Fragen haben ...

... stehe ich Ihnen gerne zur Verfügung.