



Testen? Wird überschätzt!
Schneller zu Testfällen in APEX

Andreas Fend
Consultant


Trivadis
makes IT
easier.

BASEL • BERN • BRUGG • DÜSSELDORF • FRANKFURT A.M. • FREIBURG I.B.R. • GENÈVE
HAMBURG • KOPENHAGEN • LAUSANNE • MÜNCHEN • STÜTTGART • WIEN • ZÜRICH

trivadis
makes IT easier.

■ **Unser Unternehmen.**

Trivadis ist **führend bei der IT-Beratung, der Systemintegration, dem Solution Engineering** und der Erbringung von **IT-Services** mit Fokussierung auf **ORACLE** - und **Microsoft** -Technologien in der Schweiz, Deutschland, Österreich und Dänemark. Trivadis erbringt ihre Leistungen aus den strategischen Geschäftsfeldern:




Trivadis Services übernimmt den korrespondierenden Betrieb Ihrer IT Systeme.

2 29.04.2016 © Trivadis – Das Unternehmen (Kurzpräsentation)

trivadis
makes IT easier.

Der Fokus als IT-Beratung und Systemintegrator liegt auf den Geschäftsfeldern Business Intelligence, Application Development, Infrastructure Engineering und Training. In einer eigenen Einheit – der Trivadis Services – übernehmen wir den Betrieb und die Wartung/Weiterentwicklung einzelner Systeme wie z.B. Datenbanken, einzelner Applikationen oder übernehmen im Outsourcing die Verantwortung für komplexere Umgebungen. Unsere Leistungen erbringen wir Deutschland, Österreich, der Schweiz und Dänemark, mit Fokus auf Oracle und Microsoft Technologien.

■ Mit über 600 IT- und Fachexperten bei Ihnen vor Ort.



- 14 Trivadis Niederlassungen mit über 600 Mitarbeitenden.
- Über 200 Service Level Agreements.
- Mehr als 4'000 Trainingsteilnehmer.
- Forschungs- und Entwicklungsbudget: CHF 5.0 Mio.
- Finanziell unabhängig und nachhaltig profitabel.
- Erfahrung aus mehr als 1'900 Projekten pro Jahr bei über 800 Kunden.

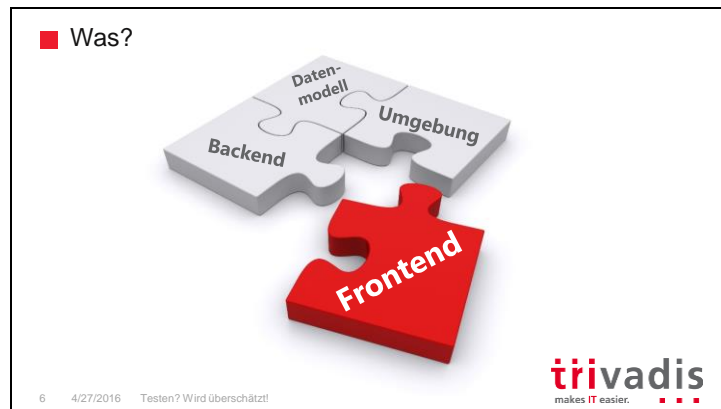
3 29.04.2016 © Trivadis – Das Unternehmen (Kurzpräsentation)

trivadis
makes IT easier. ■ ■ ■

■ Agenda

1. Testen
2. Zuviel Arbeit?
3. Vereinfachen
4. „FAAT“ - Framework for Automated APEX-Testing
5. Das Tool => Demo
6. Ausblick
7. Fragen?

Testen



Testen in Apex besteht aus vielen Bereichen, die jeweils problemlos einen ganztägigen oder mehrtägigen Kurs füllen können. In diesem Vortrag betrachten wir primär, wie man die Erstellung der Oberflächentest vereinfachen kann.

■ Wie?

Wie sieht Testen oft aus?

- Lokaler Entwicklertest
- Manuell erstellte Testdaten und Skripte
- Kundentest auf Integrationssystem
- Evtl. Testskript mit „Use-Cases“
- „umfassende Test“ in Produktion

7 4/27/2016 Testen? Wird überschätzt!

trivadis
makes IT easier.

- Entwicklertest
Das lokale Testen ist die Regel und wird von jedem Entwickler mehr oder weniger gründlich durchgeführt. Selten werden hierfür explizite Testskripte verwendet oder gar auf ein Test-Driven-Development zurückgegriffen.
- Testdaten
Testdaten werden üblicherweise «Nach Bedarf» während des Entwickelns erstellt. Einige Entwickler erstellen Skripte, mit denen sie einen spezifizierten Datenstand (ansatzweise) wieder herstellen können
- Integrationstest
Der «Integrationstest» soll eigentlich, dem Namen nach, die Integration der neuen Entwicklung in die Umgebung simulieren und prüfen. Oft wird er zusätzlich als Acceptance-Test verwendet. Außerdem erfolgt oft auch die Abnahme der Entwicklung durch den Kunden auf diesem System, gelegentlich auch parallel.
- Testscript
Wenn ein Testscript existiert, deckt es meist nur die wichtigsten Funktionalitäten ab. Insbesondere Negativfälle werden selten in solchen Skripten abgeprüft.
- Produktion
In Produktion treten in Folge dessen oft zu viele Fehler auf oder Fehler werden zu spät (manchmal erst nach weiteren Releases) entdeckt.

■ ...wie es sein sollte

I have a dream...

- Entwicklertest
 - umfangreicher Satz Testdaten, angemessene Hardware, ...
- Testdaten
 - Realitätsnah, Rollback-Konzept, ...
- Integrationstest
- Akzeptanztest durch Endanwender
- Produktionsfehler : „Kann man die Buttonfarbe anpassen?“

8 4/27/2016 Testen? Wird überschätzt!

trivadis
makes IT easier.

Dies stellt die Idealwelt dar. In der Praxis ist dies oft nicht realisierbar, geschweige denn anzutreffen. Um eine qualitativ hochwertige Software zu liefern, wäre dieser Aufwand aber notwendig.

Da in der Softwarebranche diese Art des Testens leider nicht weit verbreitet ist, werden stattdessen sog. «Bananen-Produkte» ausgeliefert, die erst beim Kunden «reifen».

Zuviel Arbeit?

■ Zuviel Arbeit/Zeit/Geld

Was schätzen Sie?

- Testen
 - 10-50% des Entwicklungsaufwandes on Top für Testdatenerstellung
 - 5-10% des Testdatenaufwandes zur Erstellung der Daten-Aufbauskripte
 - 10-50% des Entwicklungsaufwandes on Top für Testausführung

Und das macht dann?

Entwicklungsaufwand 120h + 60h Testdaten + 6h Scripting + 60h Ausführung

Hoppla.... 246h?!?

*Quellen in den Speakernotes

10 4/27/2016 Testen? Wird überschätzt!


makes IT easier. ■ ■ ■

Viele setzen den Aufwand für das Testen viel zu niedrig an. Je nach Quelle geht man von 50% bis über 100% des Entwicklungsaufwandes für Testing (inkl. Datenerstellung usw.) aus, der auf die Entwicklungsaufwände aufgeschlagen werden muss. Für ausführliche und gründliche Tests halte ich (im Schnitt) persönlich ca. 75% der Entwicklungsaufwände für durchaus angemessen. Das Beispiel geht von 110% aus, was gerade bei Neuentwicklungen durchaus realistisch ist.

Diese Aufwände verdoppeln damit den Preis der Software in etwa. Da sich ein Risiko bzw. die Kosten ungenügend getesteter Software nur schlecht beziffern lassen und man aus der Softwareentwicklung, bei Treibern zuhause, Betriebssystemen und ähnlichen bereits fehlerhafte Software gewohnt ist, wird dieser «Normalzustand» zugunsten geringerer Projektkosten jedoch oft in Kauf genommen.

Quellen:

- Wikipedia beziffert den Aufwand je nach Quelle zwischen 20%-70% des Gesamtaufwandes für ein Softwareprojekt. <https://de.wikipedia.org/wiki/Testaufwand>
- Ein Blog zum Thema Zertifizierung durch das International Software Testing Qualifications Board geht davon aus, dass man, wenn man keine Referenzwerte hat, 50% des Gesamtaufwandes als Testaufwand betrachten muss. <http://test.silke-wingens.de/2011/06/02/testmanagement-%E2%80%93-planung-testkosten-einschätzung-des-testaufwandes/>
- Prof. Dr. Andreas Spillner für den Fachbereich Informatik an der HS Bremen geht von 50% des Gesamtaufwandes aus (sprich Entwicklung, Analyse usw.), welcher als Testaufwand veranschlagt werden muss. <https://smartwebapps.de/frage/wenn-keine-erfahrungswerte-f%C3%BCr-den-testaufwand>

Vereinfachen

■ Kann man die Tests nicht vereinfachen?

Also z.B.

...weniger Tests

- Nur die Hauptbereiche der Applikation testen... die sog. „kritischen“ Bereiche
- Nur mit Realdaten testen „Das sind ja schließlich auch die, die dann vorkommen“
- Oder einfach einmal durchgeführte Tests im nächsten Durchgang überspringen ?
- ...



12 4/27/2016 Testen? Wird überschätzt!

trivadis
makes IT easier. ■ ■ ■

Bei Beibehaltung der Test-Vorgehensweise und Weglassen von Tests muss klar sein, dass immer das Risiko für unentdeckte Fehler steigt.

■ was kann man risikoarm vereinfachen / reduzieren?

Die Aufwände bei Erstellung der Tests und der Durchführung !

- **Testdaten Erstellung**
 - Durch Tools die aus Realdaten Testdaten erstellen (für das Volumen)
 - Durch Testdatengeneratoren die ggf. auch „unsinnige“ Daten erstellen
- **Testscript Erstellung** (technisch)
 - Recorder zum Erstellen der Skripte (zur Wiederverwendung) (z.B. Selenium)
 - Generische Testskripte (wiederverwendbar)
- **Automatisierte** Abarbeitung der erzeugten technischen Testskripte

13 4/27/2016 Testen? Wird überschätzt!

trivadis
makes IT easier.

Zum Glück bietet das Testen umfangreiche Möglichkeiten, Dinge zu automatisieren.

- Testdaten

Viele Testdatenverwaltungstools wie z.B. das Oracle Test Data Management Pack bieten Möglichkeiten, Echtdaten zu verfremden oder anhand von Datenmodellen Testdaten zu erstellen. Hiermit können schnell große Mengen Daten für Last- oder Performance-Tests erstellt werden. Zur Erstellung von Testdaten, die Fachlichkeit prüfen, muss oft nach wie vor der manuelle Weg gewählt werden. Allerdings sind hier oft bereits im Entwicklungsprozess Daten entstanden, die als Ausgangsbasis genutzt werden können, sofern diese ausreichend dokumentiert und reproduzierbar abgelegt wurden.

- Technische Test-Skripte

Betreibt man die Ausführung der Tests sowie das Erstellen der Testdaten soweit wie möglich automatisiert, so ist die Erstellung der technischen Testskripte der aufwendigste Teil.

Als technisches Testskript verstehe ich hier eine Reihe von Anweisungen für einen «Test-Roboter», der die beschriebenen Schritte ausführt.

- Ausführung

Moderne Testsuiten und Software bietet Oberflächentest-Automatisierung auch für APEX und Webanwendungen an. Hier werden die erstellten Skripte ausgeführt und die Ergebnisse ausgewertet und aufbereitet.

„FAAT“

(Framework for Automated APEX-Testing)

■ APEX

Eckpunkte

- APEX ist deklarativ und prinzipiell einfach aufgebaut!
- Alle Deklarationen liegen in Tabellen in der Datenbank
- Mit entsprechenden Berechtigungen ist der Zugriff kein Problem
- Die Struktur einer APEX-Applikation lässt sich also analysieren



15 4/27/2016 Testen? Wird überschätzt!

trivadis
makes IT easier.

Die Trivadis verfolgt derzeit einen Ansatz, der eine automatische Erstellung von technischen Testskripten, anhand der für die Ausführung der Applikation notwendigen Repository-Einträge in der Datenbank verfolgt. Glücklicherweise unterstützt die Struktur von APEX ein solches Vorgehen weitestgehend.

■ **APEX**

Schwierigkeiten

- Die Variationen bei der Deklaration sind ziemlich unübersichtlich
- Die zu analysierende Datenmenge ist je nach Applikation sehr groß
- Konventionen bei der Anwendung der Möglichkeiten von APEX oft unzureichend
- Die Entwicklung von APEX bleibt nicht stehen

trivadis
makes IT easier. ■ ■ ■

16 4/27/2016 Testen? Wird überschätzt!

Allerdings gibt es auch einige Stolpersteine. Zu nennen wären hierbei die folgenden Punkte:

Viele Möglichkeiten ein und dasselbe Ziel zu erreichen

So kann ein Link zu einer anderen Seite z.B. auf viele Art und Weisen erfolgen:

- Link in einem Report
- Button mit Redirect
- Link in HTML Quelltext
- Navigationslisten
- Branche als Ergebnis eines Submits
- Dynamic Actions
- JavaScript Funktion im Seitenquelltext
-

An diesem einfachen Beispiel ist gut zu erkennen, wie komplex die Abbildung eines einzelnen Vorgangs sein kann.

Dasselbe gilt natürlich auch für die Verarbeitung von Validierungen, Befüllen von LOV's etc....

Die Datenmenge wächst rasant. Mit jeder Seite in der Applikation steigt sie. Zum Glück ist die Struktur relativ klar, so dass man bei einem schrittweisen Aufbau des Testgenerators (so wie gerade im Gange) bereits früh mit Ergebnissen arbeiten kann, ohne dass die Möglichkeiten bereits voll ausgereizt wurden. Der Generator kann so Stück für Stück um Funktionalität erweitert werden.

Konventionen. Hiermit sind Entwicklungskonventionen gemeint. So sind z.B., um das Beispiel mit dem Link aufzugreifen, in der Beispielapplikation P-Track in einigen Links Synonyme und Seitennamen statt der Seitennummern verwendet. Ein einheitliches Vorgehen kann die automatisierte Erstellung von Testskripten und Analyse von Applikationen deutlich vereinfachen.

Zu guter Letzt wird APEX stetig weiter entwickelt, was ständige Erweiterungen und Anpassungen des Generators und der durch ihn erstellten Skripte nötig macht.

■ Wie können wir das Nutzen?

Anhand des Repositorys lassen sich automatisch Skripte zum Testen der fertigen Oberfläche erstellen!

■ Was gibt es zu tun?

Die Hauptaufgabe

- Automatische Analyse der Applikation
 - Seiten auslesen
 - Elemente ermitteln (Buttons, Select Listen, Text-Boxen, Popup-LOV's....)
 - Links analysieren (Navigationslisten, Branches, Buttonlinks....)
 - Validations analysieren
 - Datentypen hinter Elementen checken (Datenbankfelder, sofern keine Info am Element hinterlegt)
 - usw....

trivadis
makes IT easier. ■■■

18 4/27/2016 Testen? Wird überschätzt!

Um die Applikation zu erfassen und zu analysieren, muss primär erstmal die Anzahl der Seiten feststehen.

Diese Information lässt sich relativ leicht aus dem Repository extrahieren. Für jede Seite muss dann eine Liste der Items erstellt werden, welche zu testen sind. Dabei muss feststehen, welche Elemente getestet werden sollen.

Labels machen z.B. nur sekundär Sinn.

Images hingegen lassen sich mit den meisten Frameworks gar nicht vernünftig abgleichen.

Hier muss ggf. der Test um eine manuelle Komponente erweitert werden. So gibt es bei verschiedenen Frameworks z.B. die Möglichkeit Screenshots zu erstellen, welche dann genutzt werden können, um z.B. dynamische Grafiken wie Charts im Nachgang manuell zu verifizieren.

Links auf der Seite müssen analysiert werden und ggf. weiterverfolgt werden, um z.B. einen Wizard testen zu können.

Falls es Validierungen auf einer Seite gibt, sind auch diese zu analysieren, um korrekte bzw. fehlerhafte Daten zuordnen zu können und so deren Funktion ebenfalls bewerten zu können

Datentypen, die mit einem Element verknüpft sind, sind ebenso zu analysieren, um z.B. sicherstellen zu können, dass evtl. automatisch erstellte Testdaten korrekt erstellt werden. So muss ein Number-Feld Zahlenwerte enthalten und ein Text- oder Datums-Feld entsprechend mit Text oder einem Datum befüllt werden.

Unzählige weitere Punkte können analysiert und zur Verbesserung der Tests herangezogen werden.

■ Selenium

- Test- / Browser- automatisierungs-Tool
- Apache 2.0 License

<http://www.seleniumhq.org/>

- bestehend aus
 - IDE (Plugin für Firefox)
 - Server (Standalone)
 - Webdriver (Plugin für verschieden Testinglösungen)

Unsere Wahl fiel, der Einfachheit halber, vorerst auf die IDE zur Ausführung der Testscripts

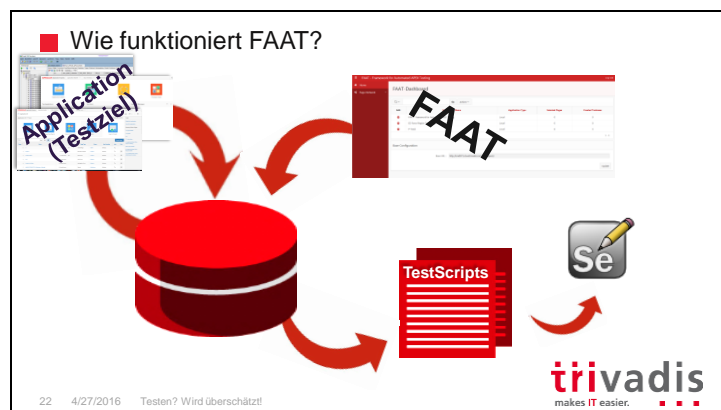
Das Tool „FAAT“

■ „FAAT“

- Framework for Automated APEX-Testing
APEX-Applikation und Framework aus PL/SQL Packages

Was leistet FAAT?

- Analysieren von Anwendungen
- Erstellen von Testskripten
- Erstellen eines Seitennetzwerks (Grafische Übersicht über die Seiten einer Applikation sowie deren Verknüpfungen zueinander)
- Weiterentwicklung App → Anpassen der Tests
- Verwalten der Testdaten



Da wir nicht davon ausgehen, dass überall Testdaten vorhanden sind, ist in dieser ersten Beta-Version ein Testdatengenerator integriert, der die Eingabefelder auf den Masken mit Zufallsdaten befüllt. Naturgemäß kann hier KEINE fachliche Sinnhaftigkeit gewährt werden. Der Generator analysiert lediglich den Datentyp des Feldes sowie die maximale Größe der aufnehmbaren Daten. Dabei bedient er sich der Einstellungen der APEX-Items. Sind hier keine Werte gepflegt, so wird, falls eine Verknüpfung zu einer Tabellenspalte besteht, deren Datentyp herangezogen. Ein Eingabefeld mit 200 Zeichen wird vom Generator mit genau 200 Buchstaben befüllt. Ein Number(5,2) Feld bekommt eine 5 Stellige Zahl inkl. 2 Nachkommastellen usw.

Für die Zukunft ist eine Schnittstelle geplant, die es ermöglicht, Testdaten im CSV-Format oder als Daten aus einer Tabelle einzuspielen und zu verwenden. Das ermöglicht dann auch fachlich vorbereitete Tests zu erstellen.

Leider wird am Ende niemals ein fertiger Test stehen, der die volle Funktionalität der Applikation abdeckt. Individuelle Lösungsansätze bei einzelnen Problemstellungen sowie die reine Fachlichkeit einer Applikation wird ein automatisiertes Testsystem nicht erfassen können. Daher kann unsere Lösung lediglich einen möglichst umfangreichen Ausgangspunkt zur Optimierung der Tests erstellen. Diese «Fleißaufgabe» kann ein solches System jedoch gut übernehmen und damit den Aufwand deutlich reduzieren.

■ Wie funktioniert FAAT?

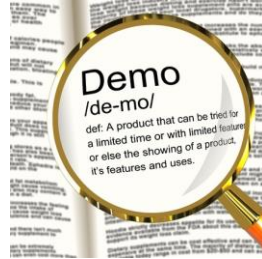
The diagram illustrates the FAAT (Functional Acceptance Acceptance Test) process. It starts with a stack of red documents labeled 'TestScripts'. A red arrow points from these scripts to a central icon representing Selenium (Se), which is a grey square with a pencil and the letters 'Se'. Another red arrow points from the Selenium icon to a screenshot of a web browser window. The browser window shows a test execution log with various status messages and a table of test results.

Test Case	Result
Test case 1: Login successful	Pass
Test case 2: Login failed	Fail
Test case 3: Logout successful	Pass

23 4/27/2016 Testen? Wird überschätzt!

trivadis
makes IT easier.

■ Demo



Ausblick

■ Das war's?

Die Testdaten

- Im ersten Schritt automatisiert erstellte Dummydaten / Manuelle Testdaten
- Demnächst:
 - Einlesen von Testdaten zur Befüllung der Skripte beim Erstellen
 - Vorbereitung von Testdaten-Templates anhand der Analyse der Applikation

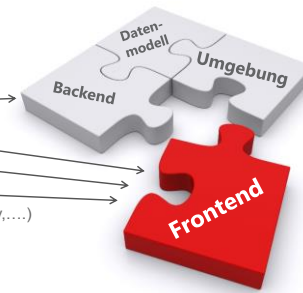
Auswertung (Testbericht/Log)

- Am Ende eines Test müssen die Ergebnisse ausgewertet und visualisiert werden

■ Was ist noch geplant?

Ideen in der Pipeline

- Backend-Tests anbinden
- Dynamic Actions
- Interactive Reports
- Java Script
- Weitere Tools anbinden (z.B. BadBoy,....)
- ...



The diagram illustrates a puzzle metaphor for software development. It features four puzzle pieces: a grey piece labeled 'Backend', a red piece labeled 'Frontend', a grey piece labeled 'Datenmodell', and a grey piece labeled 'Umgebung'. The 'Frontend' piece is positioned below the 'Backend' piece. Arrows from the list items point to the 'Backend' and 'Frontend' pieces, indicating planned integrations.

trivadis
makes IT easier.

27 4/27/2016 Testen? Wird überschätzt!

Fachliche Tests, neuronale Netze zum «intelligenten» Erkennen von Mustern in Applikationen..... Die Erweiterungsmöglichkeiten sind unbegrenzt.

■ Fragen ? ... oder Ideen ?



Vielen Dank!

Andreas Fend
Consultant

Tel. +49 89 99 27 59 32 5
Andreas.Fend@trivadis.com



trivadis
makes IT easier.

29 4/27/2016 Testen? Wird überschätzt!