Automatisiertes End-to-end Testen von Enterprise Applikationen

Bernhard Trummer
bernhard.trummer@gmx.at

9. April 2011

Überblick

Über mich

- TU / Telematik (1994 – 2001)
- Linux User Group Graz
- Angestellt bei (BearingPoint) Infonova (seit 2000)
- Technology Architekt, Bereich Infrastruktur und Testing

Linux User Group Graz

- Wann: Jeder erste Mittwoch im Monat, ab 20:00
- Wo: Pizzeria Cosa Nostra (Hans-Sachs Gasse)
- Wir beißen nicht. ;-)

Inhalt

- Enterprise-Applikationen
- Automatisierte Integrationstests
- Selenium
- Mythen und Fakten
- Live Demo

Inhalt

- Enterprise-Applikationen
- Automatisierte Integrationstests
- Selenium
- Mythen und Fakten
- Live Demo

Inhalt

- Enterprise-Applikationen
- Automatisierte Integrationstests
- Selenium
- Mythen und Fakten
- Live Demo

Inhalt

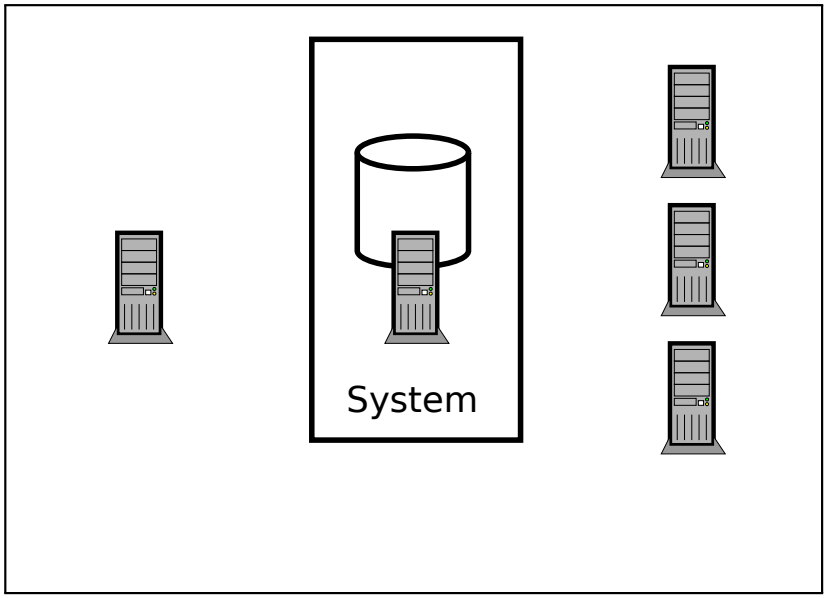
- Enterprise-Applikationen
- Automatisierte Integrationstests
- Selenium
- Mythen und Fakten
- Live Demo

Inhalt

- Enterprise-Applikationen
- Automatisierte Integrationstests
- Selenium
- Mythen und Fakten
- Live Demo

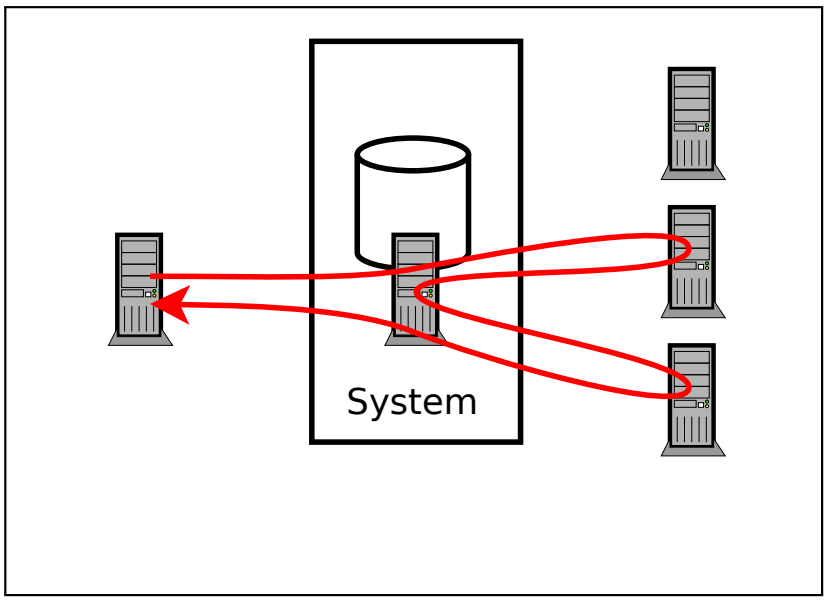
Enterprise Applikationen

Überblick



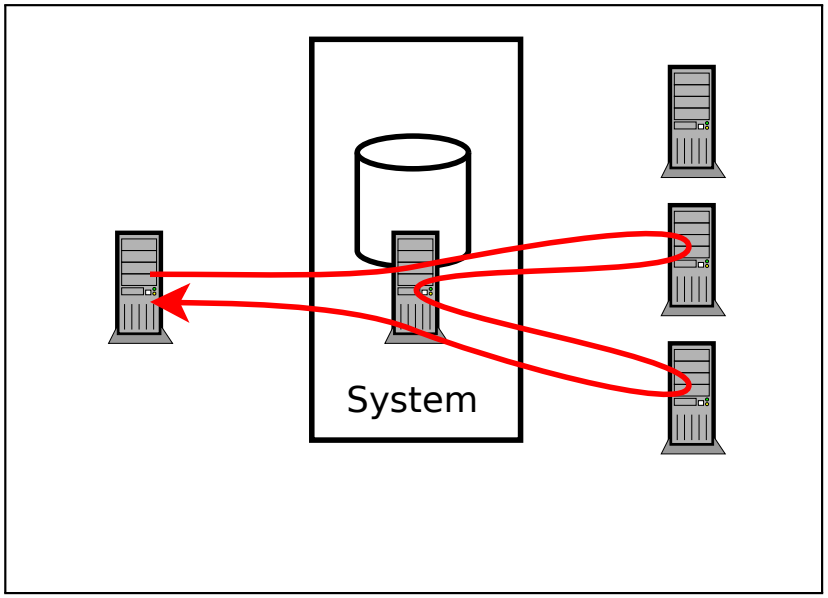
- Client
- System (und DB)
- Environment

Use Cases



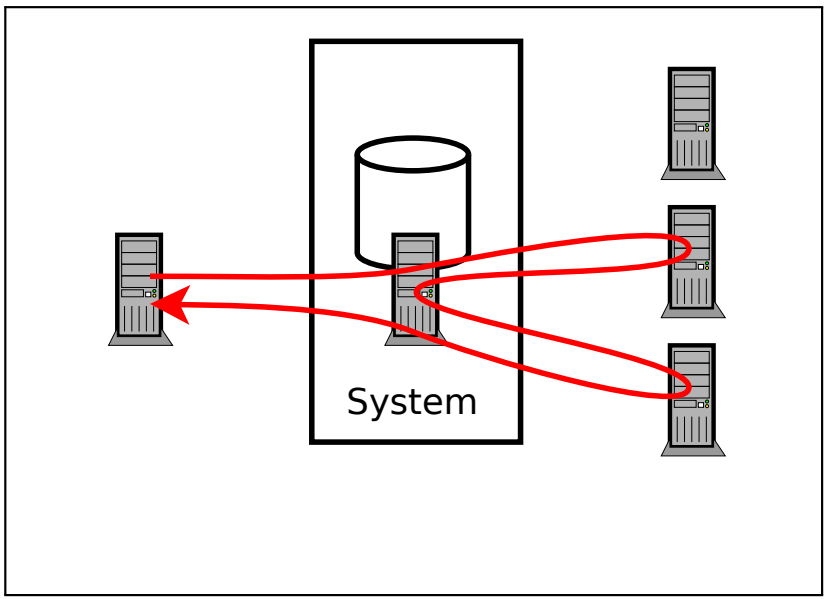
- Ausgehend vom Client
- Quer durchs System und zurück
- Diese Abläufe will man testen
- . . . und zwar automatisiert

Use Cases



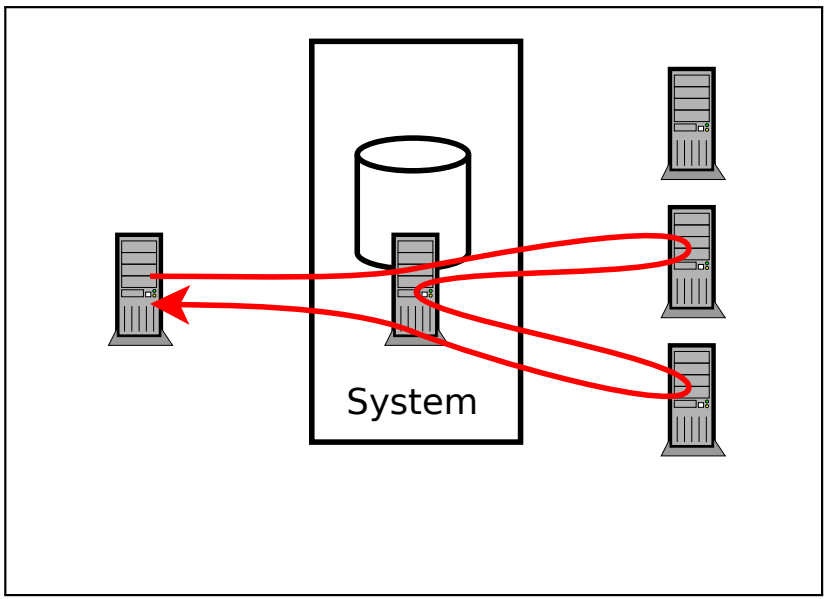
- Ausgehend vom Client
- Quer durchs System und zurück
- Diese Abläufe will man testen
- . . . und zwar automatisiert

Use Cases



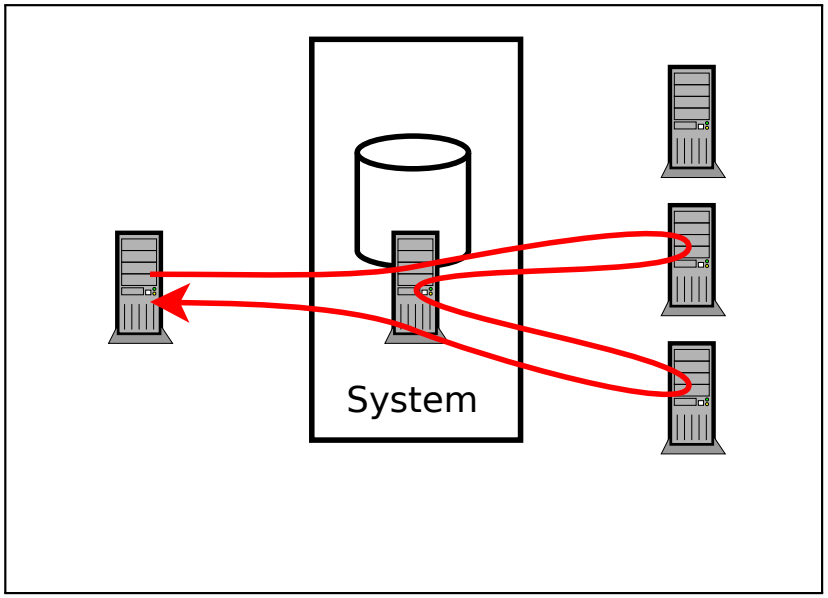
- Ausgehend vom Client
- Quer durchs System und zurück
- Diese Abläufe will man testen
- . . . und zwar automatisiert

Use Cases



- Ausgehend vom Client
- Quer durchs System und zurück
- Diese Abläufe will man testen
- . . . und zwar automatisiert

Use Cases



- Ausgehend vom Client
- Quer durchs System und zurück
- Diese Abläufe will man testen
- . . . und zwar automatisiert

Automatisierte Integrationstests

Clean Code Developer

- `http://www.clean-code-developer.de/`
- Oranger Grad

Warum?

Integrationstests stellen sicher dass der Code tut was er soll. Diese wiederkehrende Tätigkeit nicht zu automatisieren wäre Zeitverschwendung.

Clean Code Developer

- `http://www.clean-code-developer.de/`
- Oranger Grad

Warum?

Integrationstests stellen sicher dass der Code tut was er soll. Diese wiederkehrende Tätigkeit nicht zu automatisieren wäre Zeitverschwendung.

Brownfield: Wo anfangen?

- Was sind die wichtigsten Use Cases?
- Was sind die wichtigsten manuellen Regression-Tests?
- In welchem Bereich gibt es die meisten Bugs?

Brownfield: Wo anfangen?

- Was sind die wichtigsten Use Cases?
- Was sind die wichtigsten manuellen Regression-Tests?
- In welchem Bereich gibt es die meisten Bugs?

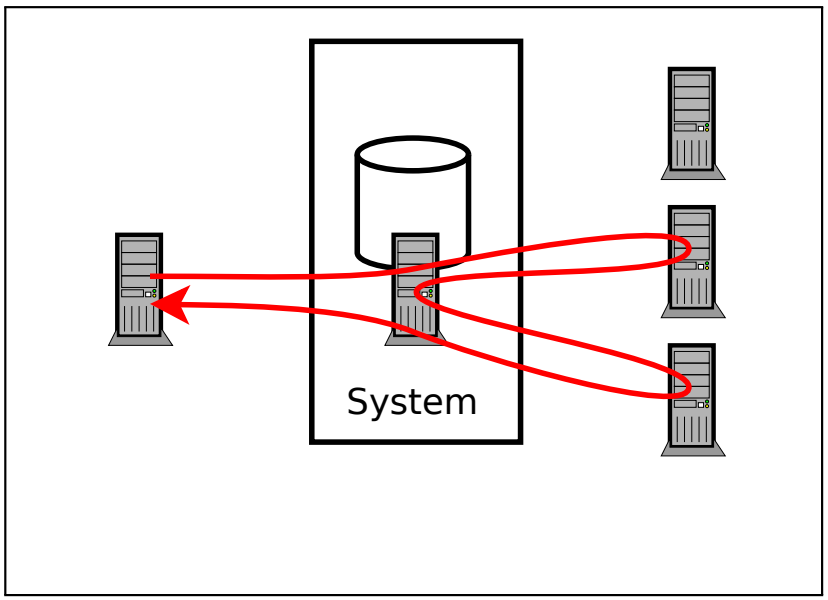
Brownfield: Wo anfangen?

- Was sind die wichtigsten Use Cases?
- Was sind die wichtigsten manuellen Regression-Tests?
- In welchem Bereich gibt es die meisten Bugs?

Brownfield: Wo anfangen?

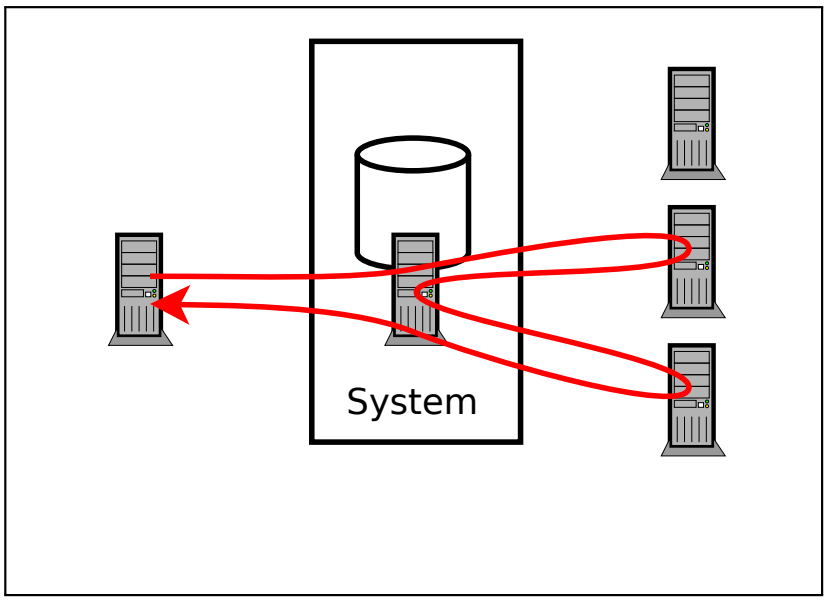
- Was sind die wichtigsten Use Cases?
- Was sind die wichtigsten manuellen Regression-Tests?
- In welchem Bereich gibt es die meisten Bugs?

Test-Framework



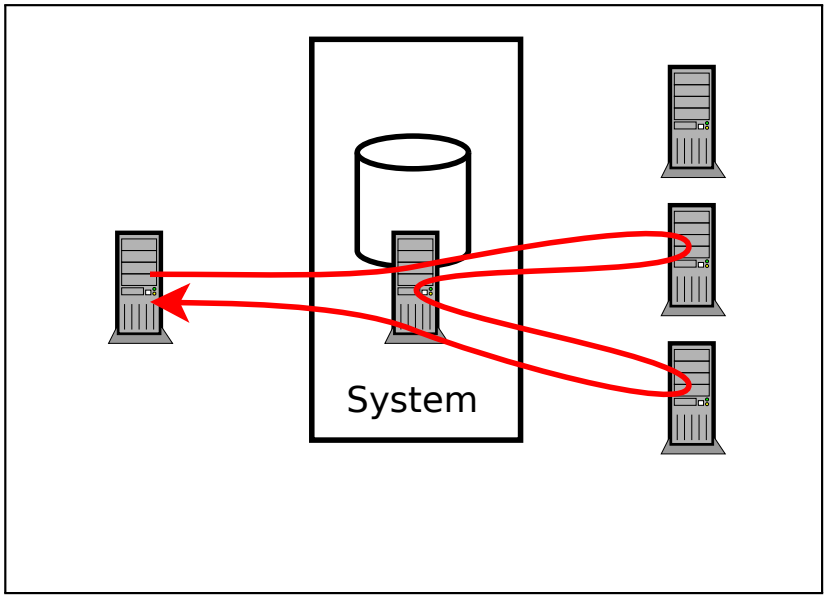
- Aktionen anstoßen
- Environment mocken
  - wenn notwendig
  - bzw. wenn sinnvoll
- Endergebnis prüfen

Test-Framework



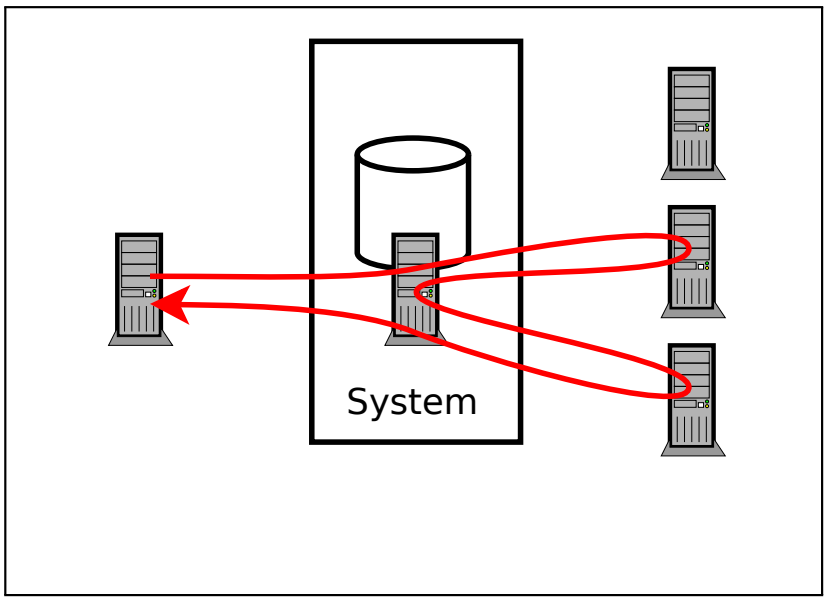
- Aktionen anstoßen
- Environment mocken
  - wenn notwendig
  - bzw. wenn sinnvoll
- Endergebnis prüfen

Test-Framework



- Aktionen anstoßen
- Environment mocken
  - wenn notwendig
  - bzw. wenn sinnvoll
- Endergebnis prüfen

Test-Framework



- Aktionen anstoßen
- Environment mocken
  - wenn notwendig
  - bzw. wenn sinnvoll
- Endergebnis prüfen

Test-Framework

- Es gibt keine „fertigen“ Frameworks.
- Was es gibt sind viele fertige „Bausteine“ mit denen man sich ein Framework basteln kann.
- Das Framework ist abhängig von der Applikation.
- . . . und abhängig davon, was man testen will.
- Es muß nicht „perfekt“ sein.

Test-Framework

- Es gibt keine „fertigen“ Frameworks.
- Was es gibt sind viele fertige „Bausteine“ mit denen man sich ein Framework basteln kann.
- Das Framework ist abhängig von der Applikation.
- . . . und abhängig davon, was man testen will.
- Es muß nicht „perfekt“ sein.

Test-Framework

- Es gibt keine „fertigen“ Frameworks.
- Was es gibt sind viele fertige „Bausteine“ mit denen man sich ein Framework basteln kann.
- Das Framework ist abhängig von der Applikation.
- . . . und abhängig davon, was man testen will.
- Es muß nicht „perfekt“ sein.

Test-Framework

- Es gibt keine „fertigen“ Frameworks.
- Was es gibt sind viele fertige „Bausteine“ mit denen man sich ein Framework basteln kann.
- Das Framework ist abhängig von der Applikation.
- . . . und abhängig davon, was man testen will.
- Es muß nicht „perfekt“ sein.

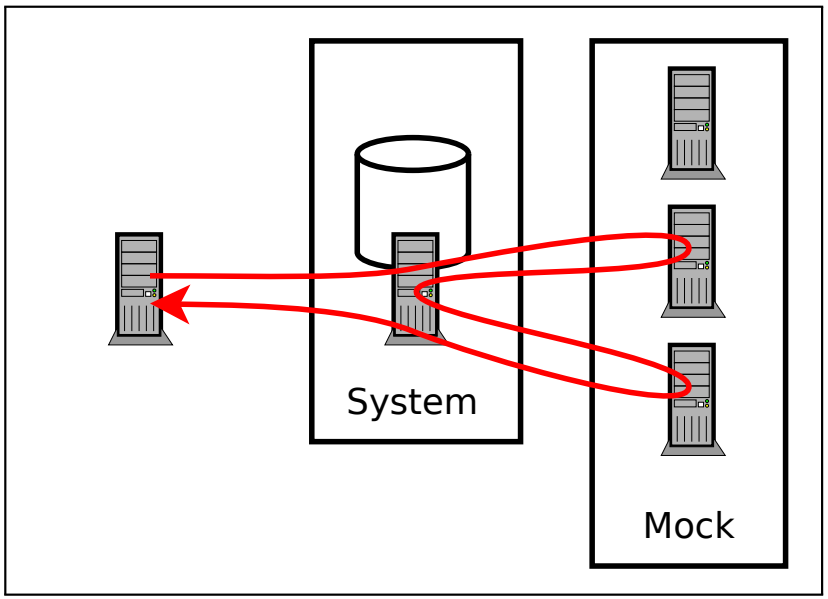
Test-Framework

- Es gibt keine „fertigen“ Frameworks.
- Was es gibt sind viele fertige „Bausteine“ mit denen man sich ein Framework basteln kann.
- Das Framework ist abhängig von der Applikation.
- . . . und abhängig davon, was man testen will.
- Es muß nicht „perfekt“ sein.

Test-Framework

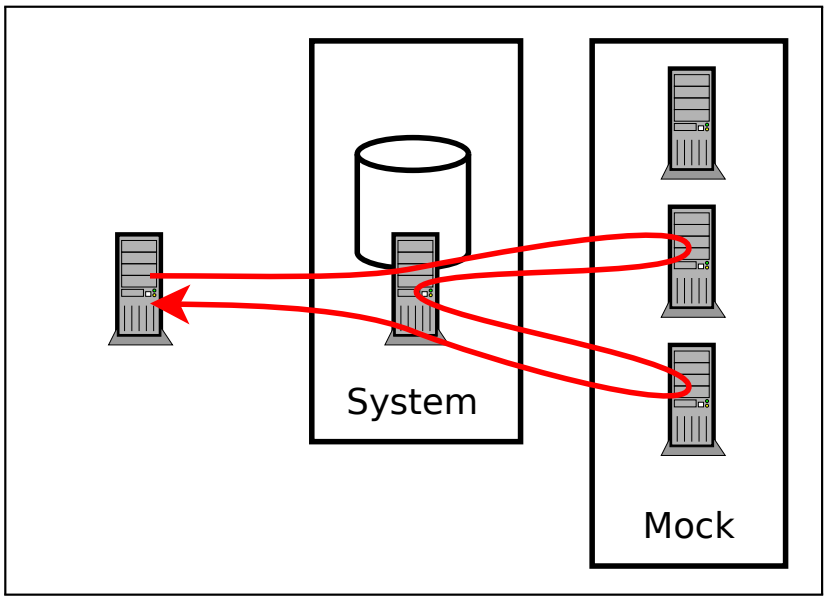
- Es gibt keine „fertigen“ Frameworks.
- Was es gibt sind viele fertige „Bausteine“ mit denen man sich ein Framework basteln kann.
- Das Framework ist abhängig von der Applikation.
- . . . und abhängig davon, was man testen will.
- Es muß nicht „perfekt“ sein.

Beispiel: Standalone-Mocks



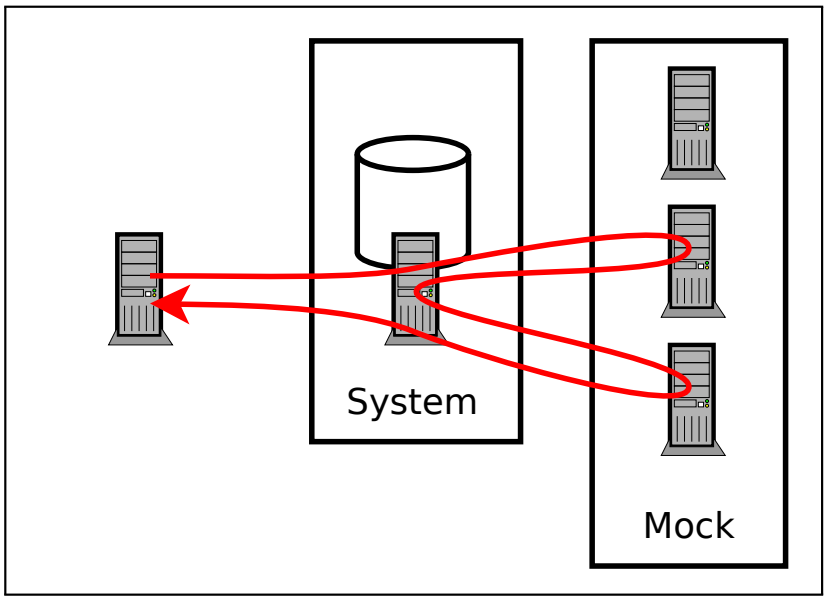
- Auch für manuelle Tests.
- SoapUI (WebServices), jes (E-Mail), etc.
- Infonova: WWB
  - Konfigurierbares und scriptbares HTTP Servlet.
  - Wird als WAR mit der Applikation mitdeployt.

Beispiel: Standalone-Mocks



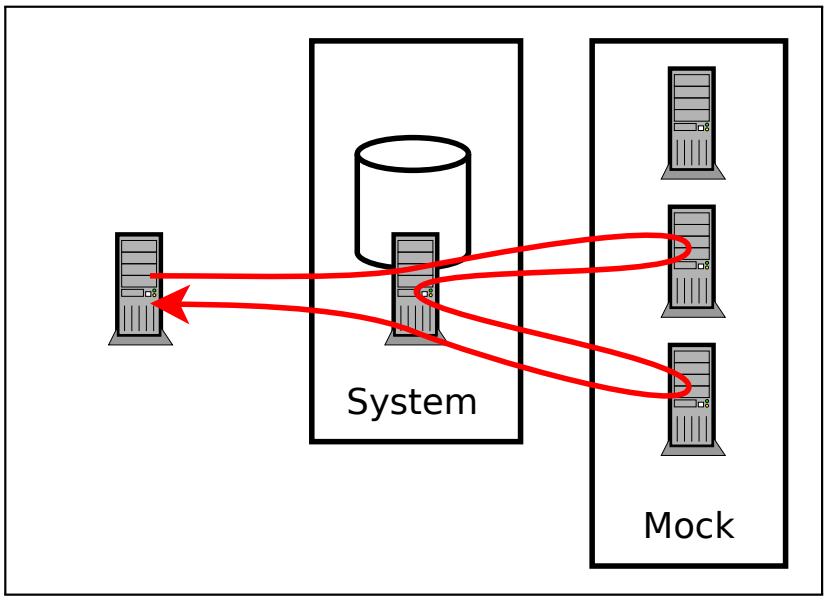
- Auch für manuelle Tests.
- SoapUI (WebServices), jes (E-Mail), etc.
- Infonova: WWB
  - Konfigurierbares und scriptbares HTTP Servlet.
  - Wird als WAR mit der Applikation mitdeployt.

Beispiel: Standalone-Mocks



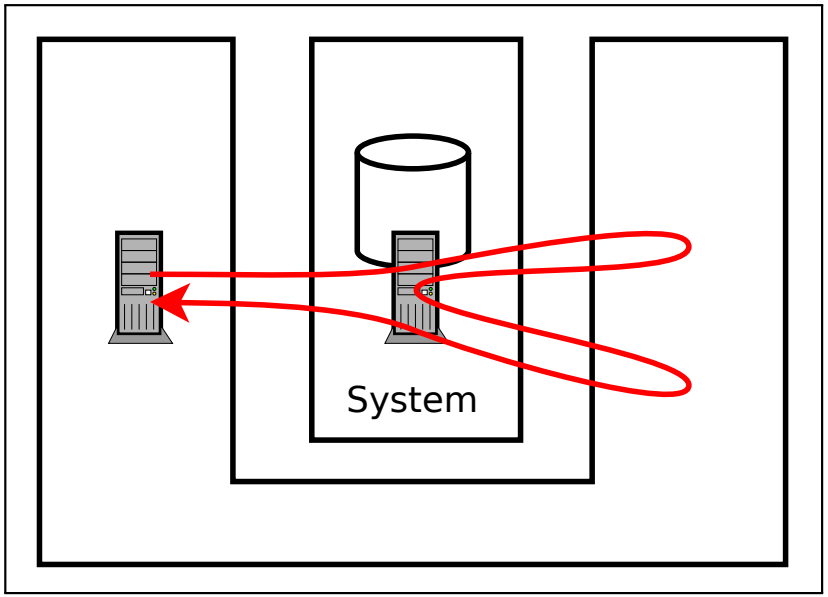
- Auch für manuelle Tests.
- SoapUI (WebServices), jes (E-Mail), etc.
- Infonova: WWB
  - Konfigurierbares und scriptbares HTTP Servlet.
  - Wird als WAR mit der Applikation mitdeployt.

Beispiel: Standalone-Mocks



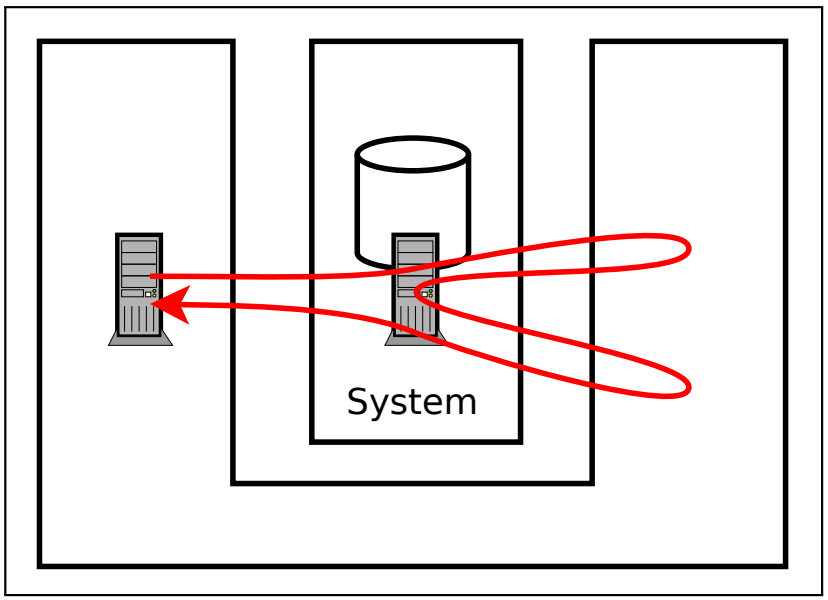
- Auch für manuelle Tests.
- SoapUI (WebServices), jes (E-Mail), etc.
- Infonova: WWB
  - Konfigurierbares und scriptbares HTTP Servlet.
  - Wird als WAR mit der Applikation mitdeployt.

Beispiel: Integrierte Mocks



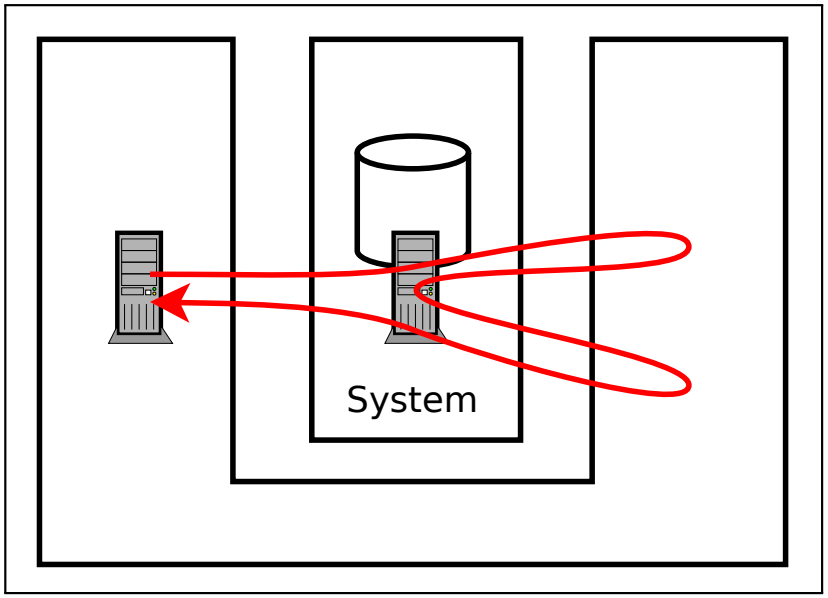
- Für Template-driven Testing.
- Umsetzung z.B. mit HttpServer (Java 6) oder embedded Jetty.
- Voraussetzung: Applikation zur Laufzeit konfigurierbar.

Beispiel: Integrierte Mocks



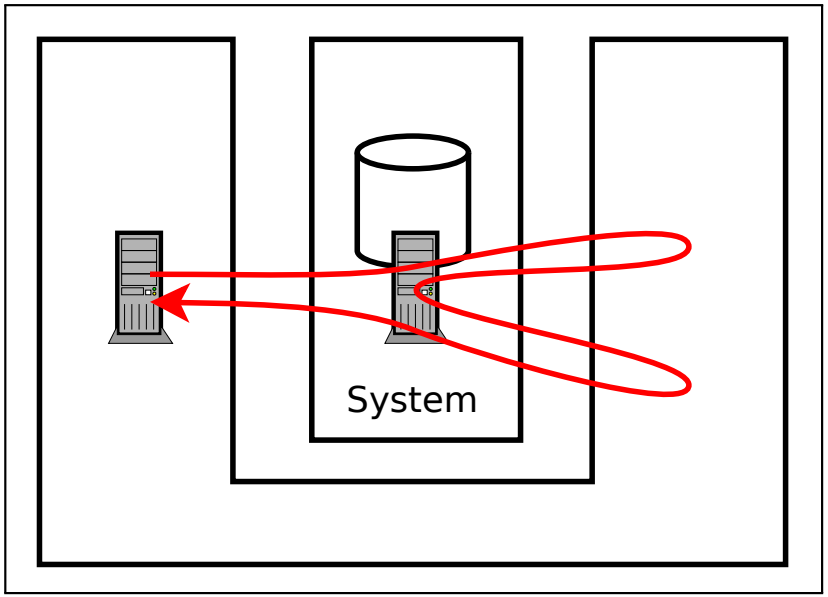
- Für Template-driven Testing.
- Umsetzung z.B. mit HttpServer (Java 6) oder embedded Jetty.
- Voraussetzung: Applikation zur Laufzeit konfigurierbar.

Beispiel: Integrierte Mocks



- Für Template-driven Testing.
- Umsetzung z.B. mit HttpServer (Java 6) oder embedded Jetty.
- Voraussetzung: Applikation zur Laufzeit konfigurierbar.

Beispiel: Integrierte Mocks



- Für Template-driven Testing.
- Umsetzung z.B. mit HttpServer (Java 6) oder embedded Jetty.
- Voraussetzung: Applikation zur Laufzeit konfigurierbar.

Selenium

Selenium: Überblick

- `http://seleniumhq.org/`

Selenium is a suite of tools to automate web app testing across many platforms.

- Grob beschrieben:
  - Fernsteuerung eines Browsers (IE, Firefox, Chrome).
  - Zugriff auf Seitenelemente programmatisch möglich.

Selenium: Überblick

- `http://seleniumhq.org/`

Selenium is a suite of tools to automate web app testing across many platforms.

- Grob beschrieben:
  - Fernsteuerung eines Browsers (IE, Firefox, Chrome).
  - Zugriff auf Seitenelemente programmatisch möglich.

Selenium: Überblick

- `http://seleniumhq.org/`

> Selenium is a suite of tools to automate web app testing across many platforms.

- Grob beschrieben:
  - Fernsteuerung eines Browsers (IE, Firefox, Chrome).
  - Zugriff auf Seitenelemente programmatisch möglich.

SeleniumIDE

- Firefox-Plugin zur Aufnahme von Aktionen.
- Den Output davon aber bitte nicht 1:1 als Test verwenden!
- Sondern das Page Object Pattern verwenden:
  - http://code.google.com/p/selenium/wiki/PageObjects
  - Trennung von Page-Internas und Tests
  - Ergebnis: wartbare(re) Tests

SeleniumIDE

- Firefox-Plugin zur Aufnahme von Aktionen.
- Den Output davon aber bitte nicht 1:1 als Test verwenden!
- Sondern das Page Object Pattern verwenden:
  - http://code.google.com/p/selenium/wiki/PageObjects
  - Trennung von Page-Internas und Tests
  - Ergebnis: wartbare(re) Tests

SeleniumIDE

- Firefox-Plugin zur Aufnahme von Aktionen.
- Den Output davon aber bitte nicht 1:1 als Test verwenden!
- Sondern das Page Object Pattern verwenden:
  - `http://code.google.com/p/selenium/wiki/PageObjects`
  - Trennung von Page-Internas und Tests
  - Ergebnis: wartbare(re) Tests

SeleniumIDE

- Firefox-Plugin zur Aufnahme von Aktionen.
- Den Output davon aber bitte nicht 1:1 als Test verwenden!
- Sondern das Page Object Pattern verwenden:
  - `http://code.google.com/p/selenium/wiki/PageObjects`
  - Trennung von Page-Internas und Tests
  - Ergebnis: wartbare(re) Tests

SeleniumIDE

- Firefox-Plugin zur Aufnahme von Aktionen.
- Den Output davon aber bitte nicht 1:1 als Test verwenden!
- Sondern das Page Object Pattern verwenden:
  - `http://code.google.com/p/selenium/wiki/PageObjects`
  - Trennung von Page-Internas und Tests
  - Ergebnis: wartbare(re) Tests

SeleniumIDE

- Firefox-Plugin zur Aufnahme von Aktionen.
- Den Output davon aber bitte nicht 1:1 als Test verwenden!
- Sondern das Page Object Pattern verwenden:
  - `http://code.google.com/p/selenium/wiki/PageObjects`
  - Trennung von Page-Internas und Tests
  - Ergebnis: wartbare(re) Tests

Beispiel: schlechter Test

```
public void testLoginLogout() {
    webDriver.findElement(By.id("username")).sendKeys(USERNAME);
    webDriver.findElement(By.id("passwd")).sendKeys(PASSWORD);
    webDriver.findElement(By.id("login")).click();

    webDriver.findElement(By.id("logout")).click();
}
```

Beispiel: besserer Test mit Page Objects

```
public void testLoginLogout() {
    LoginPage loginPage = new LoginPage(webDriver);
    HomePage homePage = loginPage.login(USERNAME, PASSWORD);
    homePage.logout();
}
```

Mythen und Fakten

Wir haben eh Unit-Tests

- Applikation wird nicht **am Zielsystem** getestet.
- Oracle verhält sich nicht immer so wie hsqldb.
- EJB3-Annotationen, Spring-Konfiguration, Datasources, . . .

Wir haben eh Unit-Tests

- Applikation wird nicht **am Zielsystem** getestet.
- Oracle verhält sich nicht immer so wie hsqldb.
- EJB3-Annotationen, Spring-Konfiguration, Datasources, . . .

Wir haben eh Unit-Tests

- Applikation wird nicht **am Zielsystem** getestet.
- Oracle verhält sich nicht immer so wie hsqldb.
- EJB3-Annotationen, Spring-Konfiguration, Datasources, . . .

Wir haben eh Unit-Tests

- Applikation wird nicht **am Zielsystem** getestet.
- Oracle verhält sich nicht immer so wie hsqldb.
- EJB3-Annotationen, Spring-Konfiguration, Datasources, ...

Testen tun eh die Tester

- Gefahr der Silo-Bildung.
- Testen sollte ein Teil der Entwicklung sein.
- Automatisierte Tests können ein Sicherheitsnetz sein.
- Entwickler „trauen“ sich (wieder), Dinge zu verändern.

Testen tun eh die Tester

- Gefahr der Silo-Bildung.
- Testen sollte ein Teil der Entwicklung sein.
- Automatisierte Tests können ein Sicherheitsnetz sein.
- Entwickler „trauen“ sich (wieder), Dinge zu verändern.

Testen tun eh die Tester

- Gefahr der Silo-Bildung.
- Testen sollte ein Teil der Entwicklung sein.
- Automatisierte Tests können ein Sicherheitsnetz sein.
- Entwickler „trauen“ sich (wieder), Dinge zu verändern.

Testen tun eh die Tester

- Gefahr der Silo-Bildung.
- Testen sollte ein Teil der Entwicklung sein.
- Automatisierte Tests können ein Sicherheitsnetz sein.
- Entwickler „trauen“ sich (wieder), Dinge zu verändern.

Testen tun eh die Tester

- Gefahr der Silo-Bildung.
- Testen sollte ein Teil der Entwicklung sein.
- Automatisierte Tests können ein Sicherheitsnetz sein.
- Entwickler „trauen“ sich (wieder), Dinge zu verändern.

Testen wir lieber manuell, das geht schneller

- Ein manueller Test ist Voraussetzung für Automatisierung.
- Ein automatisierter Test bzw. eine Erweiterung des Test-Frameworks kostet Zeit.
- Langfristig gesehen zahlt sich diese Investition jedoch aus.
- Zahlenbeispiel:
  - 100 Tests automatisiert in der Nacht.
  - oder manuell in 5 Stunden (Annahme: 3 Minuten / Test)
- Automatisierung bringt gratis Regressions-Tests.

Testen wir lieber manuell, das geht schneller

- Ein manueller Test ist Voraussetzung für Automatisierung.
- Ein automatisierter Test bzw. eine Erweiterung des Test-Frameworks kostet Zeit.
- Langfristig gesehen zahlt sich diese Investition jedoch aus.
- Zahlenbeispiel:
  - 100 Tests automatisiert in der Nacht.
  - oder manuell in 5 Stunden (Annahme: 3 Minuten / Test)
- Automatisierung bringt gratis Regressions-Tests.

Testen wir lieber manuell, das geht schneller

- Ein manueller Test ist Voraussetzung für Automatisierung.
- Ein automatisierter Test bzw. eine Erweiterung des Test-Frameworks kostet Zeit.
- Langfristig gesehen zahlt sich diese Investition jedoch aus.
- Zahlenbeispiel:
  - 100 Tests automatisiert in der Nacht.
  - oder manuell in 5 Stunden (Annahme: 3 Minuten / Test)
- Automatisierung bringt gratis Regressions-Tests.

Testen wir lieber manuell, das geht schneller

- Ein manueller Test ist Voraussetzung für Automatisierung.
- Ein automatisierter Test bzw. eine Erweiterung des Test-Frameworks kostet Zeit.
- Langfristig gesehen zahlt sich diese Investition jedoch aus.
- Zahlenbeispiel:
  - 100 Tests automatisiert in der Nacht.
  - oder manuell in 5 Stunden (Annahme: 3 Minuten / Test)
- Automatisierung bringt gratis Regressions-Tests.

Testen wir lieber manuell, das geht schneller

- Ein manueller Test ist Voraussetzung für Automatisierung.
- Ein automatisierter Test bzw. eine Erweiterung des Test-Frameworks kostet Zeit.
- Langfristig gesehen zahlt sich diese Investition jedoch aus.
- Zahlenbeispiel:
  - 100 Tests automatisiert in der Nacht.
  - oder manuell in 5 Stunden (Annahme: 3 Minuten / Test)
- Automatisierung bringt gratis Regressions-Tests.

Testen wir lieber manuell, das geht schneller

- Ein manueller Test ist Voraussetzung für Automatisierung.
- Ein automatisierter Test bzw. eine Erweiterung des Test-Frameworks kostet Zeit.
- Langfristig gesehen zahlt sich diese Investition jedoch aus.
- Zahlenbeispiel:
  - 100 Tests automatisiert in der Nacht.
  - oder manuell in 5 Stunden (Annahme: 3 Minuten / Test)
- Automatisierung bringt gratis Regressions-Tests.

Es ist aufwendig, die Tests immer nachzuziehen

- Tests leben, wachsen und ändern sich mit der Applikation.
- Evolvierbarkeit der Tests ist genauso wichtig wie die der Applikation.
- CCD-Prinzipien auch für Tests befolgen (z.B. DRY).
- Selenium: Page Object Pattern.

Es ist aufwendig, die Tests immer nachzuziehen

- Tests leben, wachsen und ändern sich mit der Applikation.
- Evolvierbarkeit der Tests ist genauso wichtig wie die der Applikation.
- CCD-Prinzipien auch für Tests befolgen (z.B. DRY).
- Selenium: Page Object Pattern.

Es ist aufwendig, die Tests immer nachzuziehen

- Tests leben, wachsen und ändern sich mit der Applikation.
- Evolvierbarkeit der Tests ist genauso wichtig wie die der Applikation.
- CCD-Prinzipien auch für Tests befolgen (z.B. DRY).
- Selenium: Page Object Pattern.

Es ist aufwendig, die Tests immer nachzuziehen

- Tests leben, wachsen und ändern sich mit der Applikation.
- Evolvierbarkeit der Tests ist genauso wichtig wie die der Applikation.
- CCD-Prinzipien auch für Tests befolgen (z.B. DRY).
- Selenium: Page Object Pattern.

Es ist aufwendig, die Tests immer nachzuziehen

- Tests leben, wachsen und ändern sich mit der Applikation.
- Evolvierbarkeit der Tests ist genauso wichtig wie die der Applikation.
- CCD-Prinzipien auch für Tests befolgen (z.B. DRY).
- Selenium: Page Object Pattern.

Live Demo

Überblick

- Produkt WebAC: Place Order
- Produkt: Mediation und Rating
- SFM: XDSL-Provisionierung

Produkt WebAC: Place Order

- DB Cleanup bzw. Setup über PL/SQL Prozeduren.
- DB Setup stellt Produkt-Konfiguration her.
- Triggern von „Place Order“ über WebAC.
- Polling bis Order den Status COMPLETED erreicht hat.
- Überprüfung der Kundendaten im WebAC.

Produkt WebAC: Place Order

- DB Cleanup bzw. Setup über PL/SQL Prozeduren.
- DB Setup stellt Produkt-Konfiguration her.
- Triggern von „Place Order“ über WebAC.
- Polling bis Order den Status COMPLETED erreicht hat.
- Überprüfung der Kundendaten im WebAC.

Produkt WebAC: Place Order

- DB Cleanup bzw. Setup über PL/SQL Prozeduren.
- DB Setup stellt Produkt-Konfiguration her.
- Triggern von „Place Order“ über WebAC.
- Polling bis Order den Status COMPLETED erreicht hat.
- Überprüfung der Kundendaten im WebAC.

Produkt WebAC: Place Order

- DB Cleanup bzw. Setup über PL/SQL Prozeduren.
- DB Setup stellt Produkt-Konfiguration her.
- Triggern von „Place Order“ über WebAC.
- Polling bis Order den Status COMPLETED erreicht hat.
- Überprüfung der Kundendaten im WebAC.

Produkt WebAC: Place Order

- DB Cleanup bzw. Setup über PL/SQL Prozeduren.
- DB Setup stellt Produkt-Konfiguration her.
- Triggern von „Place Order“ über WebAC.
- Polling bis Order den Status COMPLETED erreicht hat.
- Überprüfung der Kundendaten im WebAC.

Produkt WebAC: Place Order

- DB Cleanup bzw. Setup über PL/SQL Prozeduren.
- DB Setup stellt Produkt-Konfiguration her.
- Triggern von „Place Order“ über WebAC.
- Polling bis Order den Status COMPLETED erreicht hat.
- Überprüfung der Kundendaten im WebAC.

Produkt: Mediation und Rating

- DB Cleanup und Setup wie gehabt.
- Triggern von Usage-Records über File-Schnittstelle.
- Anstoßen der Bearbeitungsprozesse.
- Überprüfung in der DB.
- Ist daher nicht so schön anzusehen wie ein Selenium-Test.
- Ist aber mindestens genauso wichtig.

Produkt: Mediation und Rating

- DB Cleanup und Setup wie gehabt.
- Triggern von Usage-Records über File-Schnittstelle.
- Anstoßen der Bearbeitungsprozesse.
- Überprüfung in der DB.
- Ist daher nicht so schön anzusehen wie ein Selenium-Test.
- Ist aber mindestens genauso wichtig.

Produkt: Mediation und Rating

- DB Cleanup und Setup wie gehabt.
- Triggern von Usage-Records über File-Schnittstelle.
- Anstoßen der Bearbeitungsprozesse.
- Überprüfung in der DB.
- Ist daher nicht so schön anzusehen wie ein Selenium-Test.
- Ist aber mindestens genauso wichtig.

Produkt: Mediation und Rating

- DB Cleanup und Setup wie gehabt.
- Triggern von Usage-Records über File-Schnittstelle.
- Anstoßen der Bearbeitungsprozesse.
- Überprüfung in der DB.
- Ist daher nicht so schön anzusehen wie ein Selenium-Test.
- Ist aber mindestens genauso wichtig.

Produkt: Mediation und Rating

- DB Cleanup und Setup wie gehabt.
- Triggern von Usage-Records über File-Schnittstelle.
- Anstoßen der Bearbeitungsprozesse.
- Überprüfung in der DB.
- Ist daher nicht so schön anzusehen wie ein Selenium-Test.
- Ist aber mindestens genauso wichtig.

Produkt: Mediation und Rating

- DB Cleanup und Setup wie gehabt.
- Triggern von Usage-Records über File-Schnittstelle.
- Anstoßen der Bearbeitungsprozesse.
- Überprüfung in der DB.
- Ist daher nicht so schön anzusehen wie ein Selenium-Test.
- Ist aber mindestens genauso wichtig.

Produkt: Mediation und Rating

- DB Cleanup und Setup wie gehabt.
- Triggern von Usage-Records über File-Schnittstelle.
- Anstoßen der Bearbeitungsprozesse.
- Überprüfung in der DB.
- Ist daher nicht so schön anzusehen wie ein Selenium-Test.
- Ist aber mindestens genauso wichtig.

SFM: XDSL-Provisionierung

- DB Setup besteht nur aus “Verbiegen“ der konfigurierten WebService-URLs.
- Triggern des XML-Requests über DB-Schnittstelle.
- Mocking des Environments ist im Test integriert.
- Template-driven Testing.

SFM: XDSL-Provisionierung

- DB Setup besteht nur aus “Verbiegen“ der konfigurierten WebService-URLs.
- Triggern des XML-Requests über DB-Schnittstelle.
- Mocking des Environments ist im Test integriert.
- Template-driven Testing.

SFM: XDSL-Provisionierung

- DB Setup besteht nur aus “Verbiegen“ der konfigurierten WebService-URLs.
- Triggern des XML-Requests über DB-Schnittstelle.
- Mocking des Environments ist im Test integriert.
- Template-driven Testing.

SFM: XDSL-Provisionierung

- DB Setup besteht nur aus “Verbiegen“ der konfigurierten WebService-URLs.
- Triggern des XML-Requests über DB-Schnittstelle.
- Mocking des Environments ist im Test integriert.
- Template-driven Testing.

SFM: XDSL-Provisionierung

- DB Setup besteht nur aus “Verbiegen“ der konfigurierten WebService-URLs.
- Triggern des XML-Requests über DB-Schnittstelle.
- Mocking des Environments ist im Test integriert.
- Template-driven Testing.

Ende

Ende

- Questions & Answers