Automatisierung von Multi-Platform und Device GUI Tests

Reginald Stadlbauer

froglogic GmbH

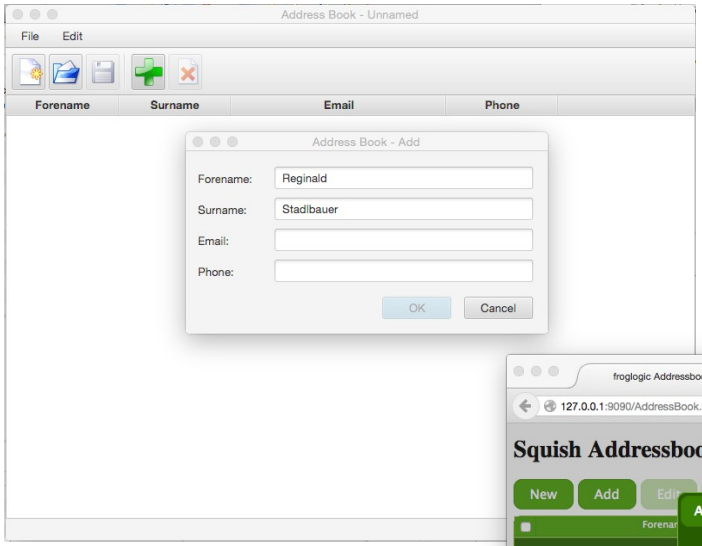
About me

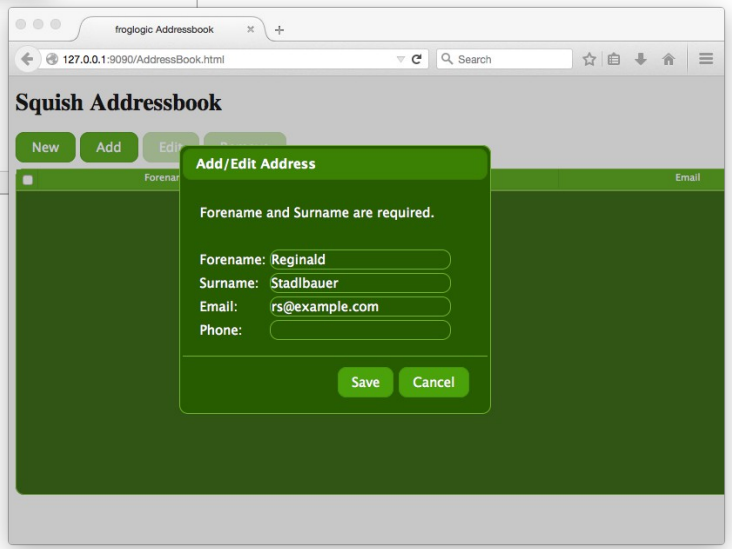
- Name: Reginald Stadlbauer
- Company: froglogic GmbH
- Position: co-founder and CEO
- Worked as Software Engineer at Trolltech and the KDE project

About froglogic

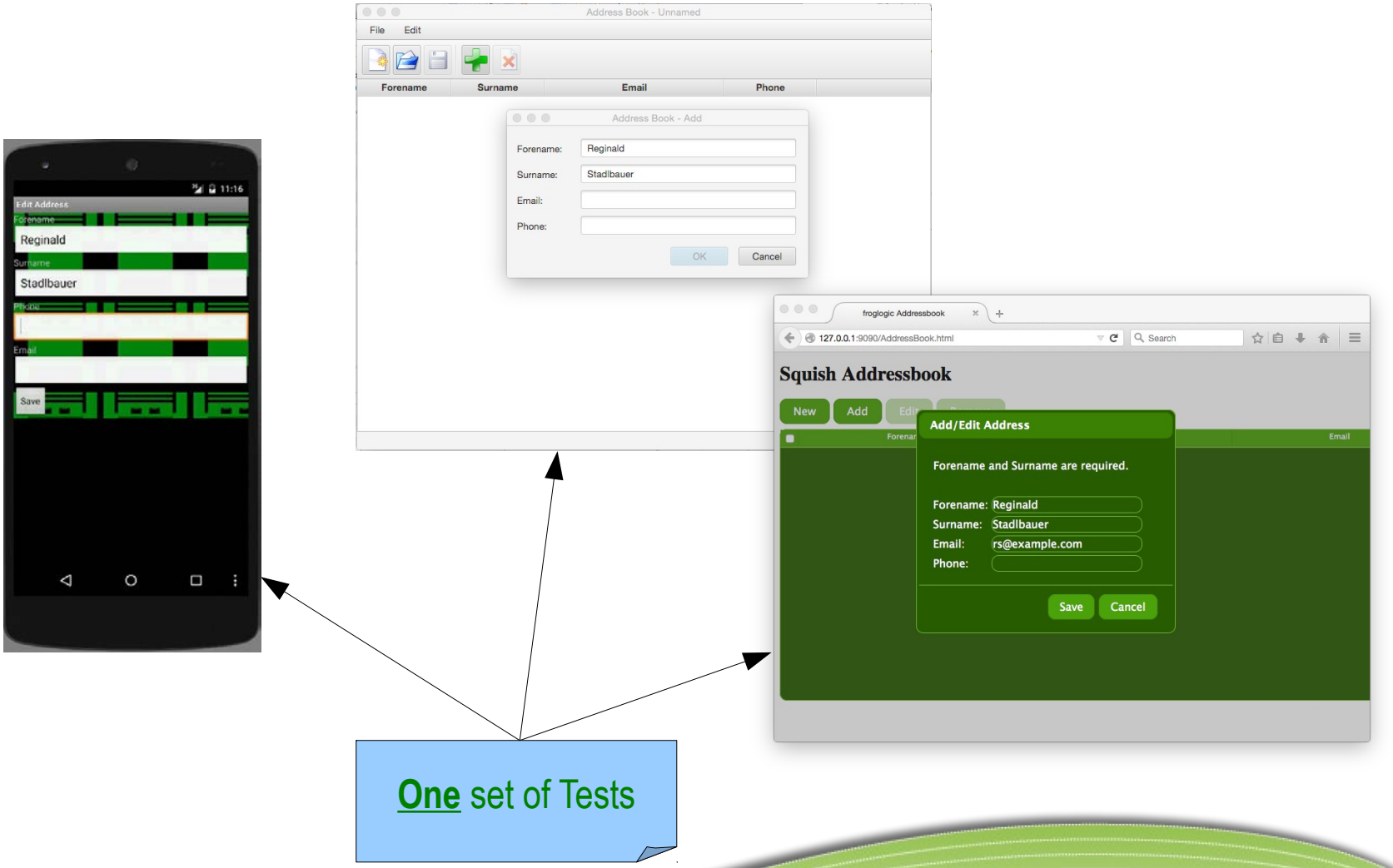
- HQ: Hamburg
- Founded: 2003
- US presence since 2008
- Product focus on Squish
  - Squish GUI Tester (Cross-Platform/Cross-Technology GUI Test Automation)
  - Squish Coco (C, C++ and C# Code Coverage)
- More than 3.000 customers world-wide

The Requirement: One Application – Many Front-Ends





The Requirement: One Test for All



What is functional GUI testing

- Outside-In
- Assume user's point of view

Why Automate?

- Faster
  - Get results quicker
  - Run more tests in the same time
- Trivial to replay in different configurations
- Reliable, reproducible and repeatable
- Relieve testers from monotonous tasks
- Quickly spot regressions
- Combine with profiling, coverage and other analysis and monitoring tools
- ...

But...

- Automating GUI tests is not trivial
- Typical reason for test effort failures: wrong test approach

Platform Challenge

Application needs to run on a range of platforms:

- Desktop
- Mobile
- Web
- Embedded

Toolkit Challenge

Different UI frameworks used for different platforms:

- Desktop: JavaFx, RCP/SWT, Swing, Qt, QML, WinForms, WPF, MFC, Cocoa, ...
- Mobile: iOS, Android, PhoneGap, Xamarin, ...
- Web: Wide range of JS / HTML5 frameworks
- Embedded: Qt, QML, ...

And even combinations of the above.

Platform Solution for Testing

Provided Abstractions by Test Tool

- Resolution independence via dedicated UI control support
- UI abstractions
- Synchronization methods
- Distributed environment

Capture and replay

- Produces massive test scripts
- Not readable
- Not maintainable
- No code re-use possible
- Brittle against changes in the UI

- Solution: Scripting, Refactoring and Abstractions

Proposed Solution

- Abstraction layer to separate test logic and implementation
- Implementation: Specialized UI test driver per Front-End
  - Scriptable
  - Supports used UI controls
  - Supports remote testing
- Allow switching between test implementations

==> **B**ehavior **D**riven **D**evelopment & Testing (BDD)

Overview

- What is BDD and TDD?
- Automating a Behavior-Driven Test
- Live Demo

What is BDD / BDT?

froglogic

“BDD is a second-generation, outside-in, pull-based, multiple-stakeholder, multiple-scale, high-automation, agile methodology. It describes a cycle of interactions with well-defined outputs, resulting in the delivery of working, tested software that matters.” - Dan North

http://en.wikipedia.org/wiki/Behavior-driven_development

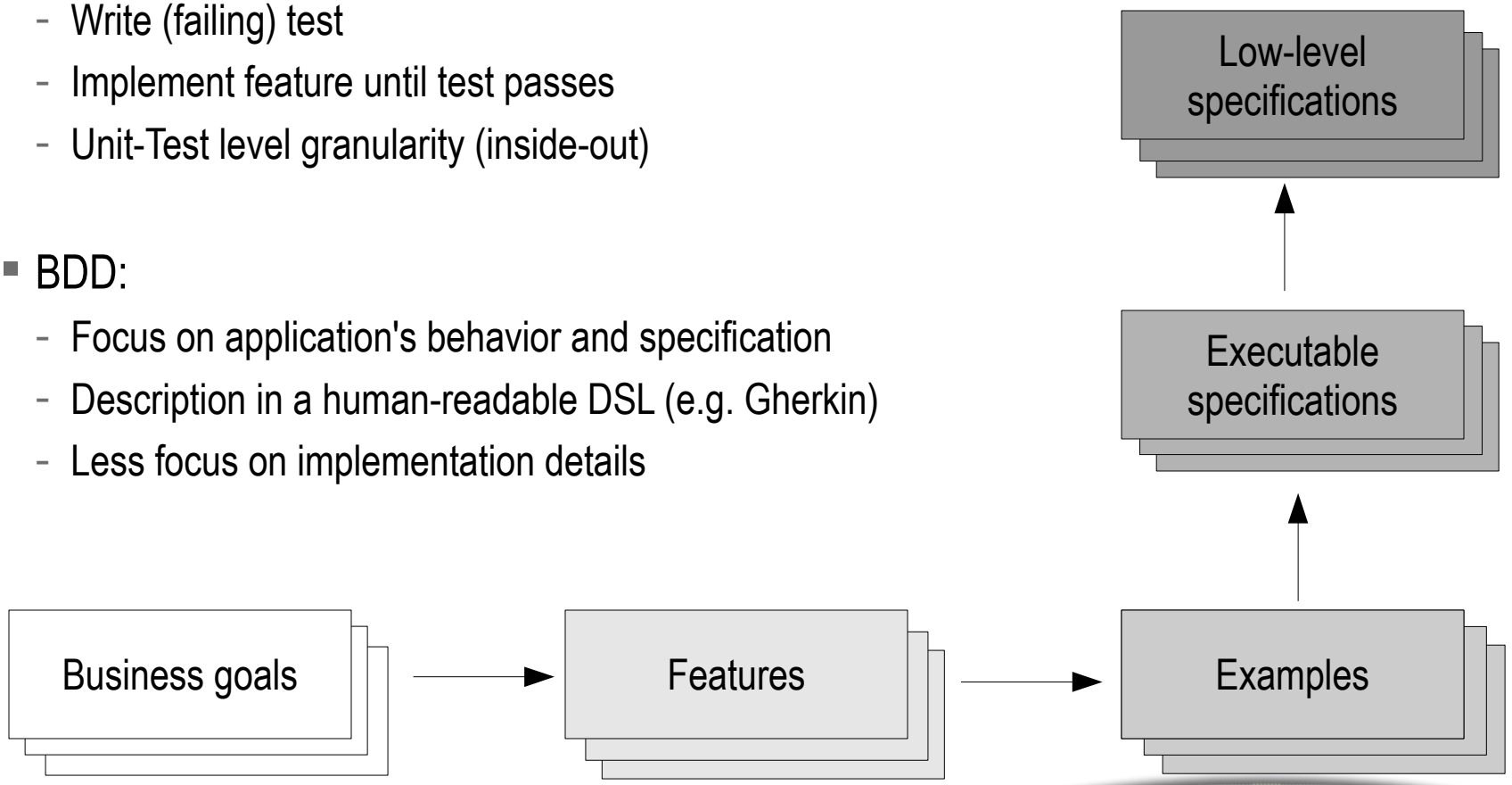
OR...

What is BDD / BDT?

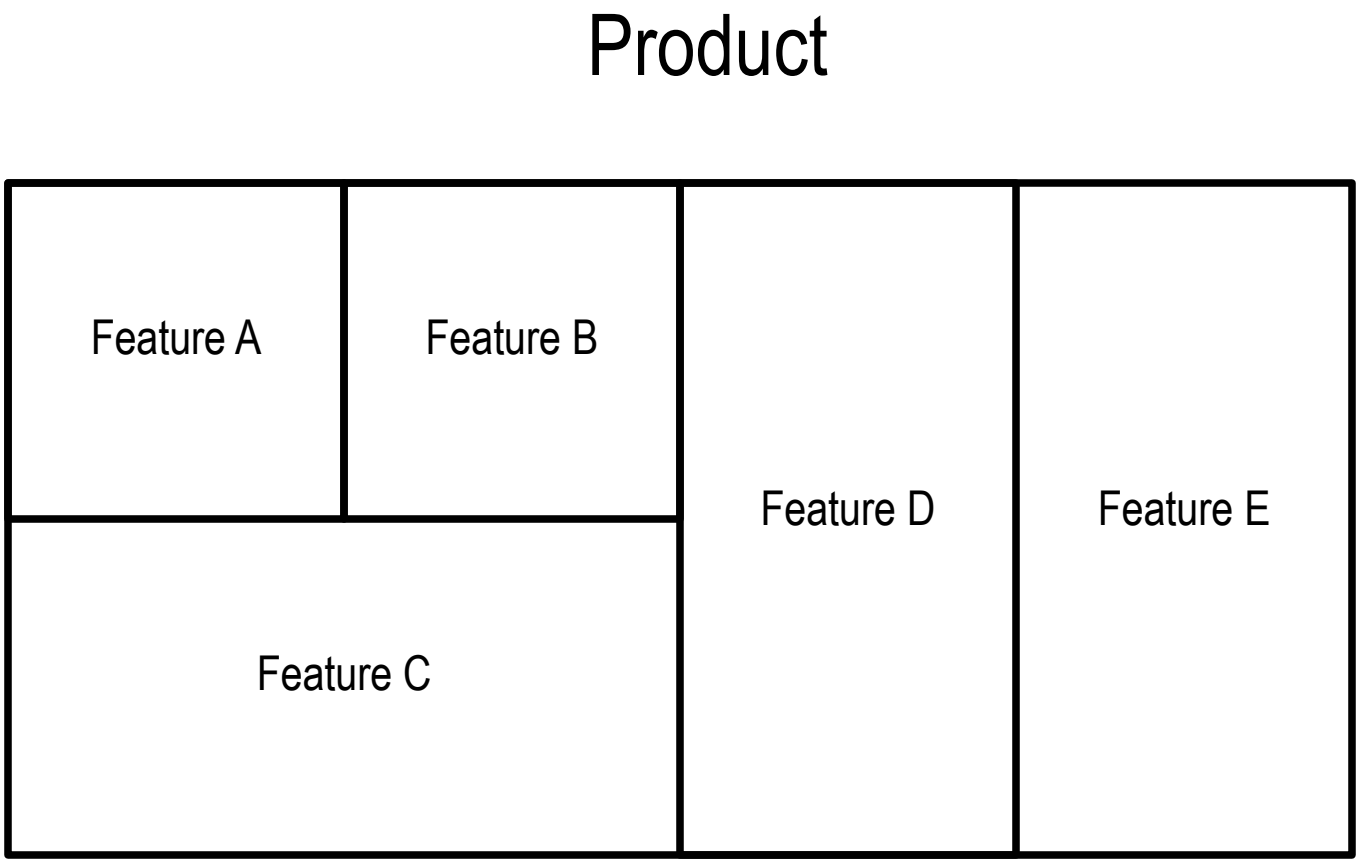
froglogic

- Test Driven Development (TDD)
  - Write (failing) test
  - Implement feature until test passes
  - Unit-Test level granularity (inside-out)

- BDD:
  - Focus on application's behavior and specification
  - Description in a human-readable DSL (e.g. Gherkin)
  - Less focus on implementation details

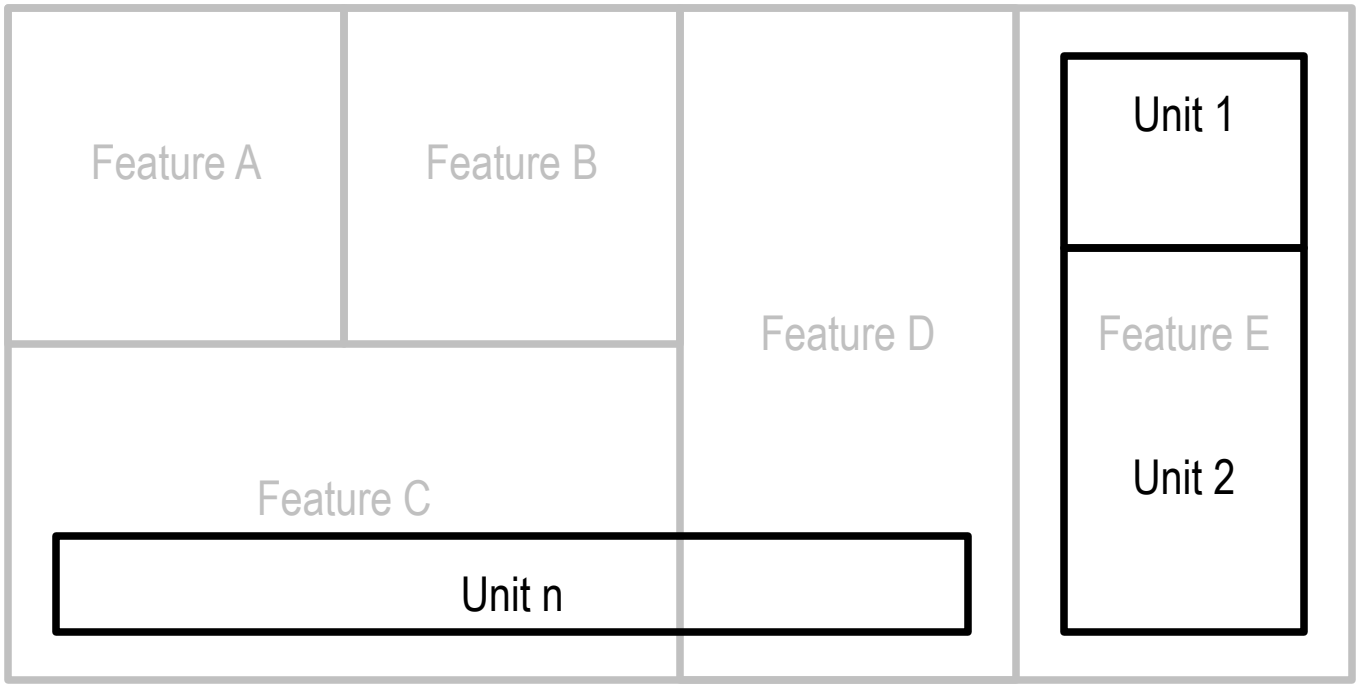


Feature Partitioning

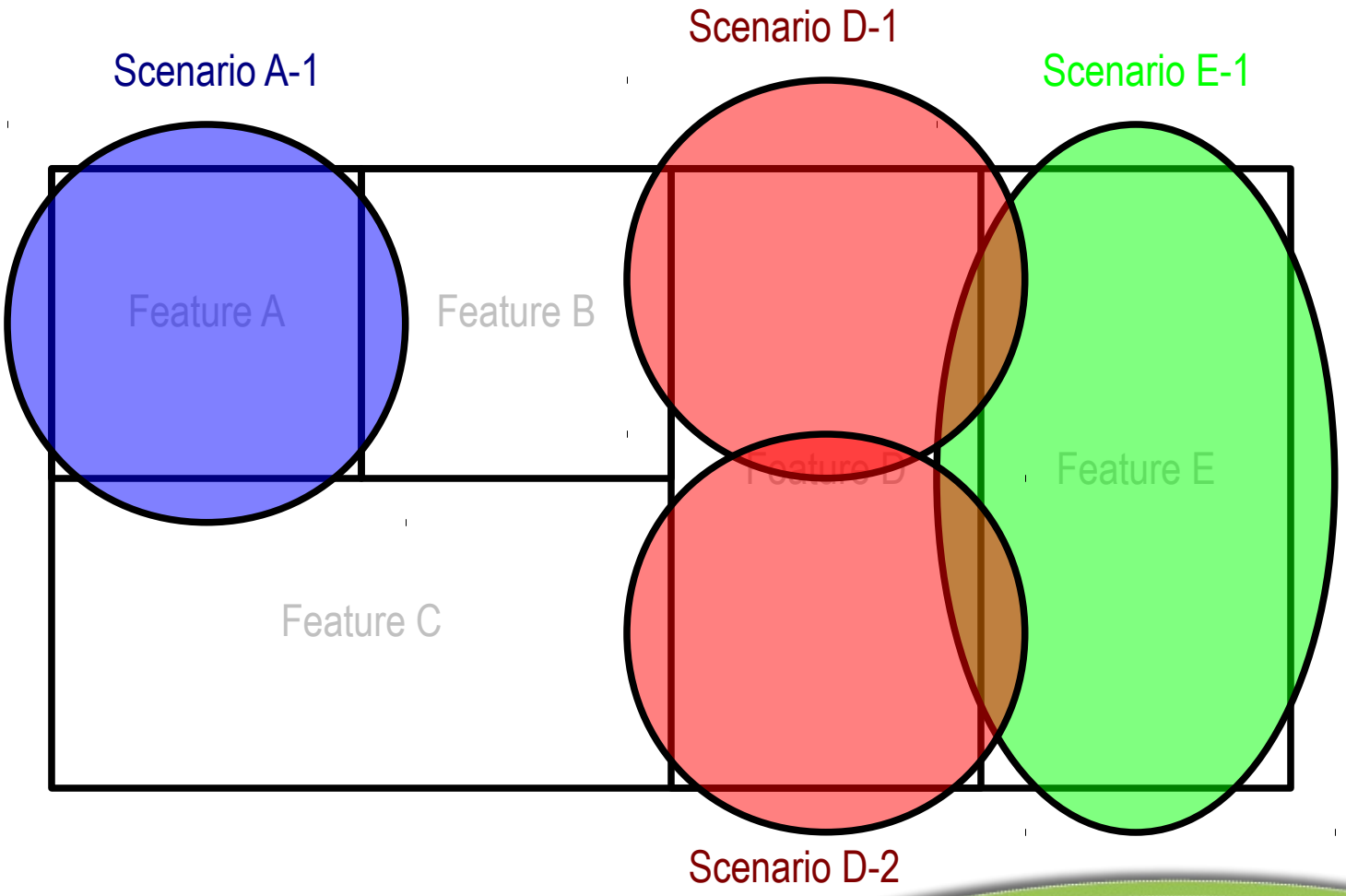


froglogic

Features ≠ Components

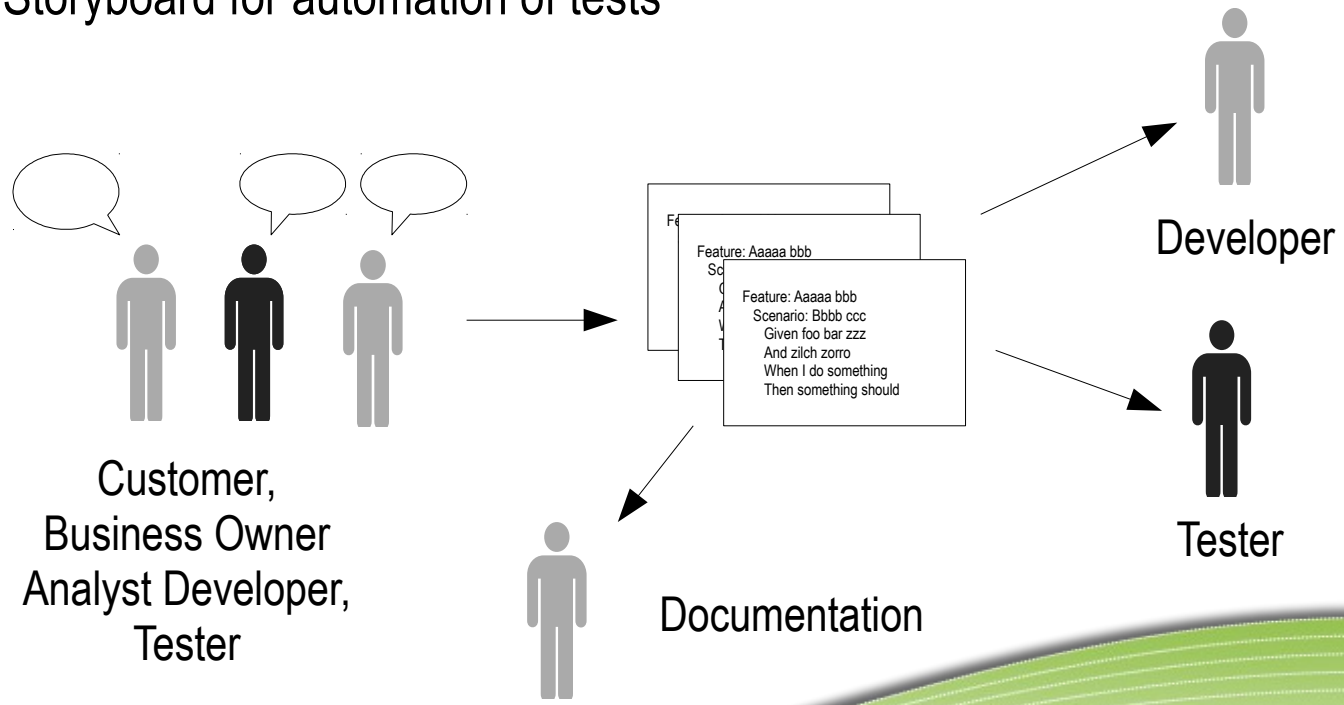


Scenarios


Scenario D-1
Scenario A-1
Scenario E-1
Feature A
Feature B
Feature D
Feature E
Feature C
Scenario D-2

Versatile usage of Feature Files

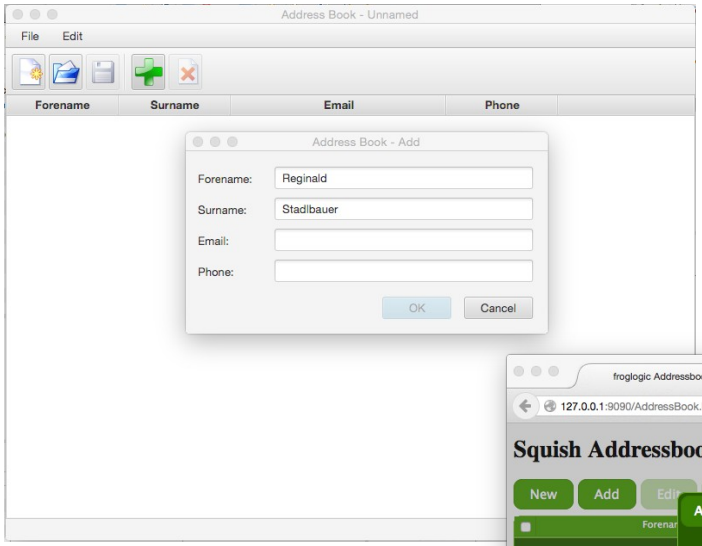
- User story / feature specification
- Communicate with customer / users
- Documentation of acceptance test
- Sequence to walk through for manual tests
- Storyboard for automation of tests

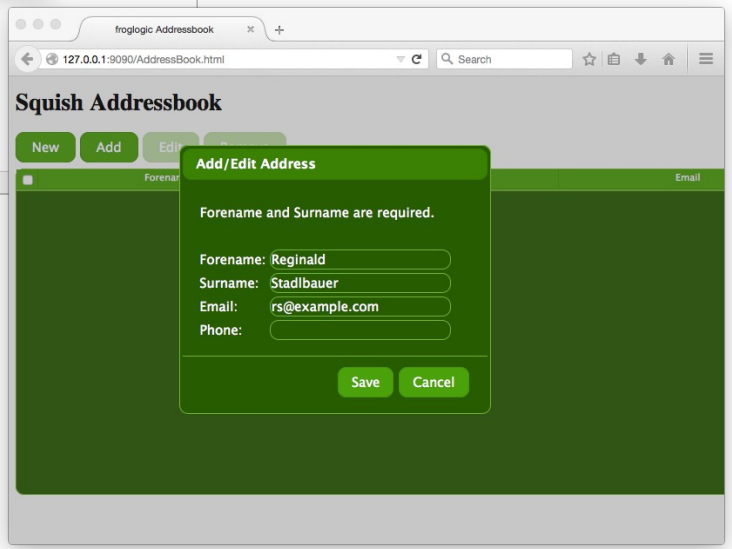


Why BDD/BDT?

- “Test first” development on a higher level
- Clearly separate test logic from implementation
- Allow non-programmers to define features & tests
- Have a common, single language two communicate

Example: Multi-Platform Addressbook





Example: Fill an empty addressbook

```
Feature: Filling Empty Addressbook

Scenario: Adding a single person
    Given the addressbook application is running
    When I create a new addressbook
    And I add a new person … to the address book
    Then 1 entries should be displayed in the address book
```

Feature File (Gherkin)

Step Types

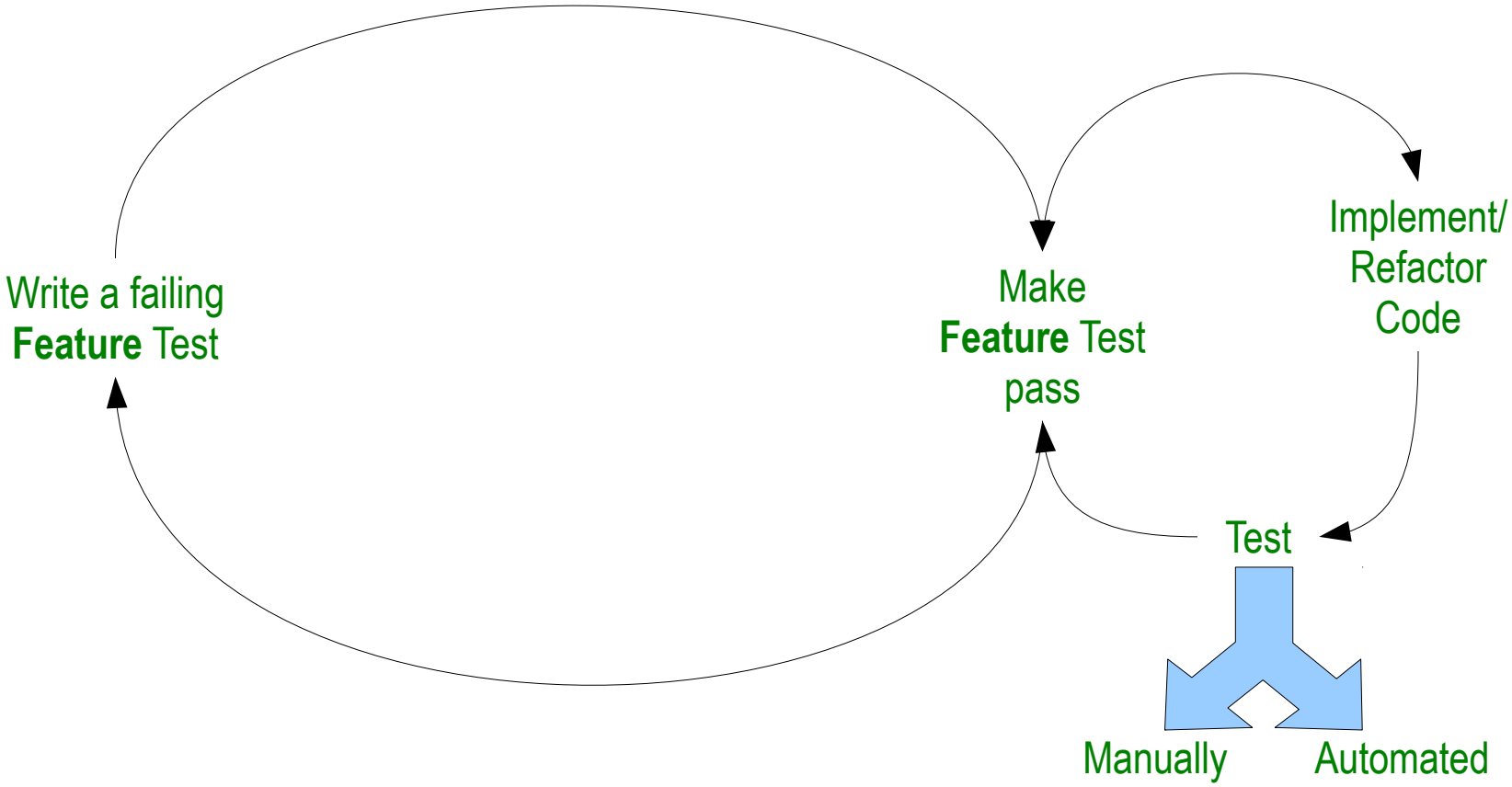
```
Scenario: Adding a single person
    Given the addressbook application is running
    When I create a new addressbook
    And I add a new person … to the address book
    Then 1 entries should be displayed in the address book
```

Precondition

Actions

Expectations

What is BDD / BDT – The Process



Automating a Behavior-Driven Test

froglogic

- Requirements
  - BDT framework
  - GUI Test Automation framework
  - Glue between both

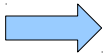
BDT Framework – Generate Skeletons

- Parse feature files
- Generate step definition skeletons (functions and annotations) in preferred language

Test.feature

```
Feature: Fill Addressbook

    Scenario: Add a person
        Given the Addressbook is running
        When I add a person
        ....
```



Test.py

```
@Step("Given the Addressbook is running")
def step(context):
    test.warning("Implement me")

@Step("When I add a person")
def step(context):
    test.warning("Implement me")
```

Test Automation framework – Test Implementation

- Support the specific (UI) technology of Application
- Support the scripting/programming language of the BDT framework
- Tooling for convenient test creation, maintenance and debugging

Test.py

```
@Step("Given the Addressbook is running")
def step(context):
    startApplication("Addressbook")

@Step("When I add a person")
def step(context):
    click("FirstName")
    typeText("Max")
```

BDT Framework + TA Framework – Run Feature Files & Reports

- Parse feature files
- Execute feature files by mapping to steps to step definitions (functions)
- Reporting

Test.py

```
@Step("Given the Addressbook is running")
def step(context):
    [...]

@Step("When I add a person")
def step(context):
    [...]
```

Test.feature

```
Feature: Fill Addressbook

    Scenario: Add a person
        Given the Addressbook is running
        When I add a person
        ....
```

Integrating BDT and Test Automation frameworks

- Need to “talk the same language”
- Reporting
- Debugging

BDT Frameworks

- Behave
- Cucumber
- JBehave
- Lettuce
- radish
- RSpec
- SpecFlow
- Squish GUI Tester
- ...

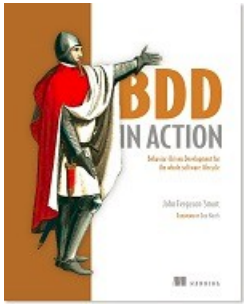
Test Automation tools

- CppUnit
- GoogleTest
- xUnit
- NUnit
- JUnit
- Squish GUI Tester
- HP QTP
- Rational Functional Tester
- Selenium
- ...

Live Demo

- BDD GUI test for cross-platform/device Addressbook
  - Create test for one frontend
  - Re-implement for other frontends
  - Extend framework to allow switching between implementations to run the same test on different frontents

Books



BDD in Action: Behavior-driven development for the whole software lifecycle (John Ferguson Smart)

The Cucumber Book: Behaviour-Driven Development for Testers and Developers (Matt Wynne)

About Squish GUI Tester

- Cross-Platform / Cross-GUI-Technology Test Automation
  - Windows, Linux, Mac OS X, Unix, QNX, VxWorks, iOS, Android, ...
  - Java (Swing/AWT, SWT/RCP, JavaFx), Qt/QML/QtQuick, Web, MFC, WinForms, WPF, iOS, Cocoa, Carbon, Android, Tk, Flex, …
- Object-based GUI object identification
- Record & replay
- Powerful scripting (JavaScript, Python, Ruby, Tcl, Perl)
- Eclipse-based IDE
- **Built-in BDD framework and support**

- Batch-testing via command-line tools
- Remote/distributed testing architecture
- Integrations: Microsoft ALM, HP QC/ALM, Rational RQM, Seapine TCM, SpiraTest, MKS, XStudio, Jenkins, Hudson, TeamCity, Bamboo, Robot Framework, JUnit, Maven, …

Q & A

froglogic

Questions? Visit our booth or email sales@froglogic.com

Free and supported trial of Squish at http://www.froglogic.com/evaluate