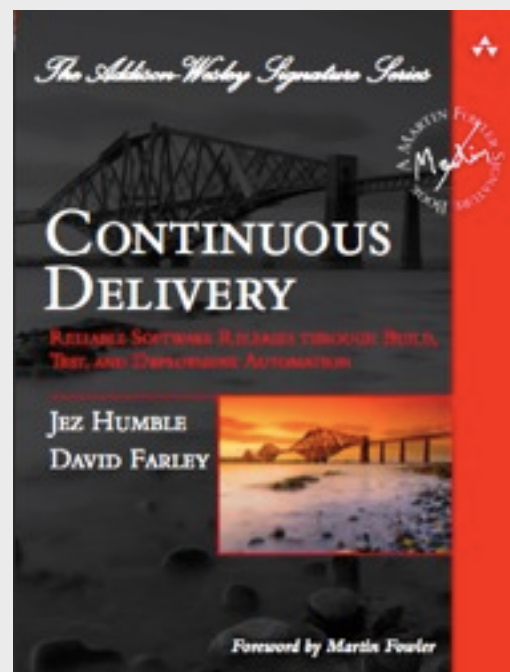


Maintainable Acceptance Tests



Badrinath Janakiraman

Jez Humble

Agile 2012 Dallas

@badrij | badri@thoughtworks.com

@jezhumble | jez@thoughtworks.com

<http://thoughtworks-studios.com/>

what to expect

- Creating high quality acceptance tests
- How to structure a maintainable acceptance test suite
- Patterns for effective teamwork
- Managing test data

what to take away

- Quality is everybody's responsibility
- High quality test suites are continuously curated
 - by testers and developers working together
- Test code needs to receive the same care as production code
- Exhaustive story-level testing is not a good basis for maintainable acceptance suites

different kinds of tests

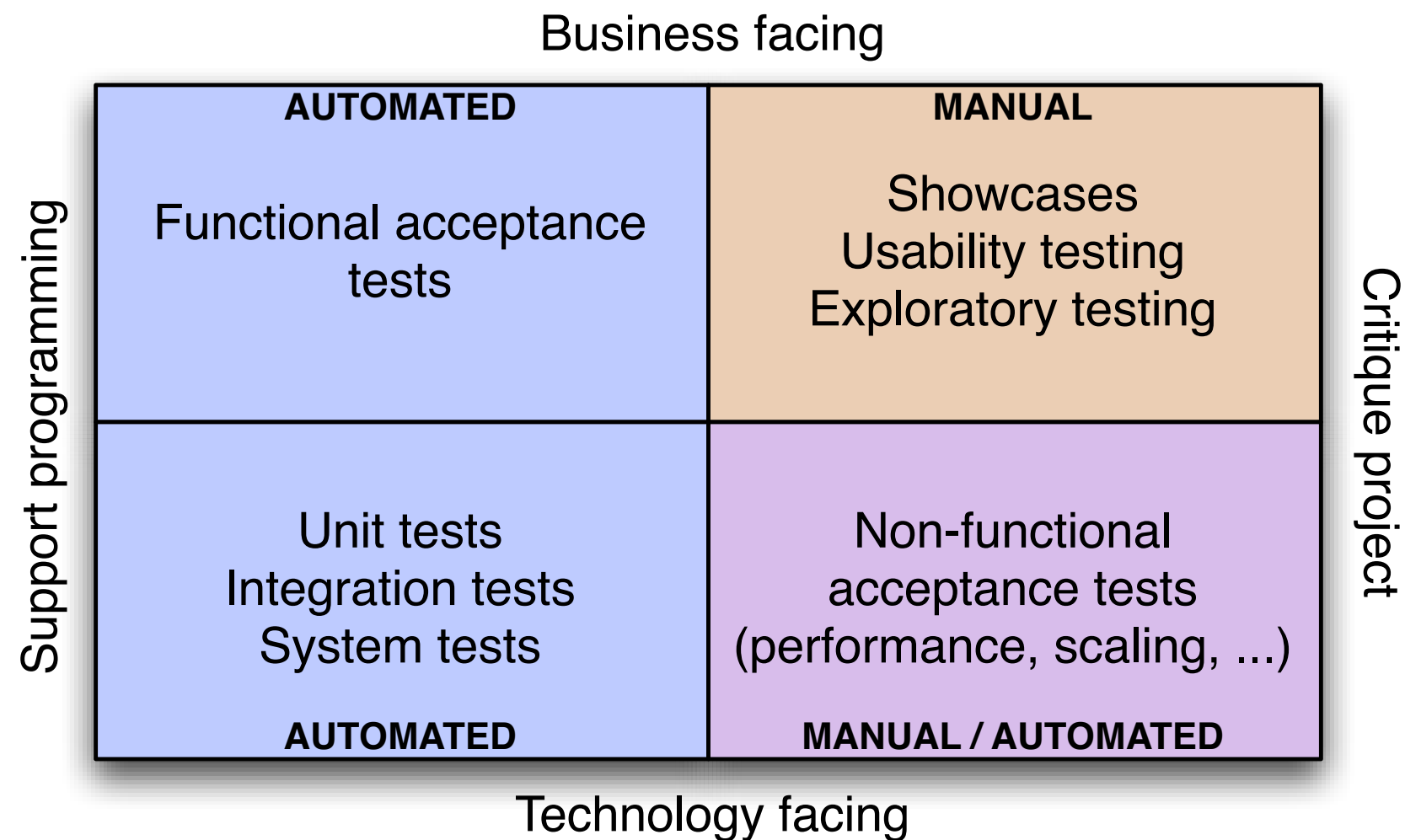
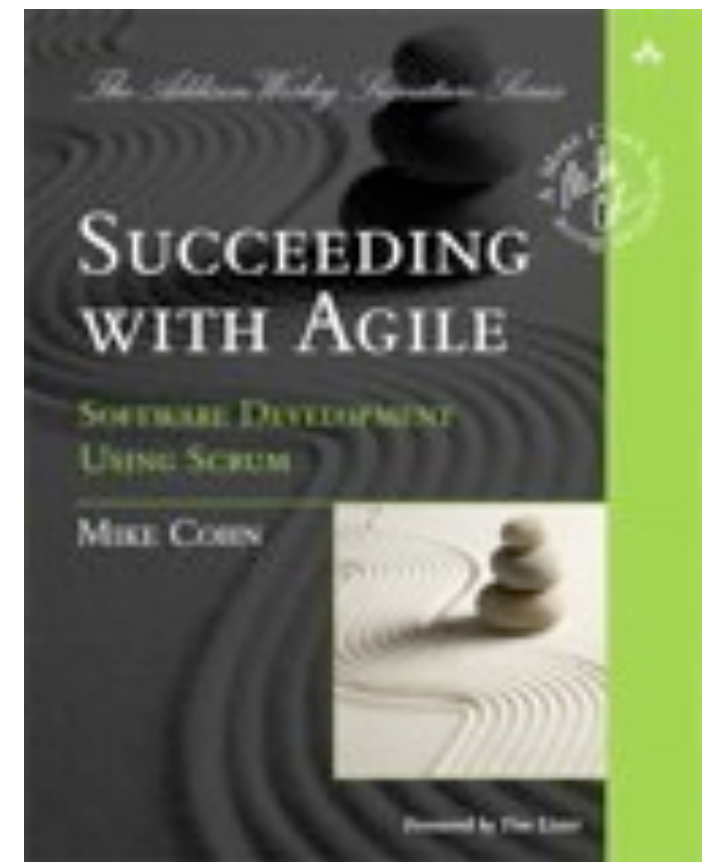
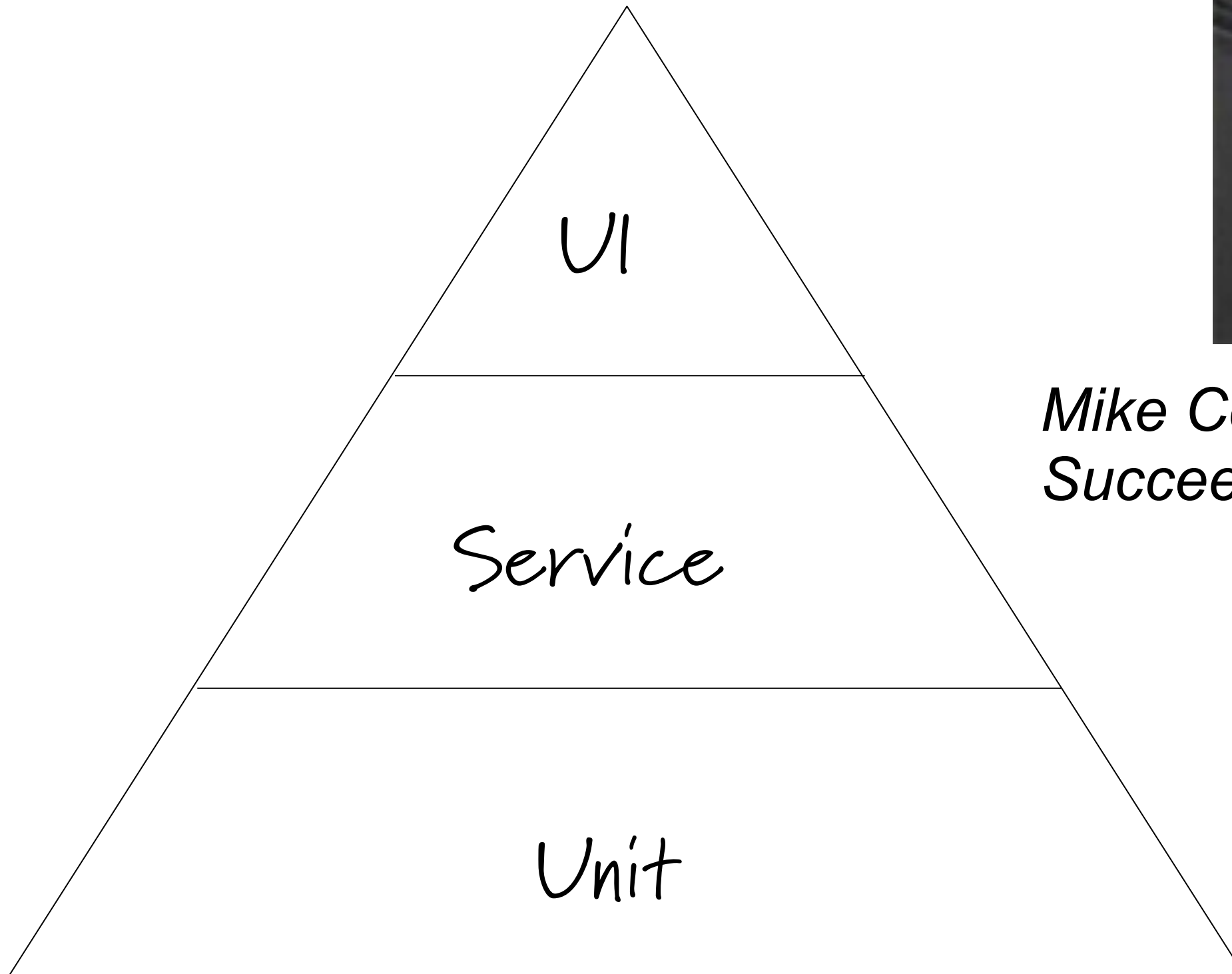
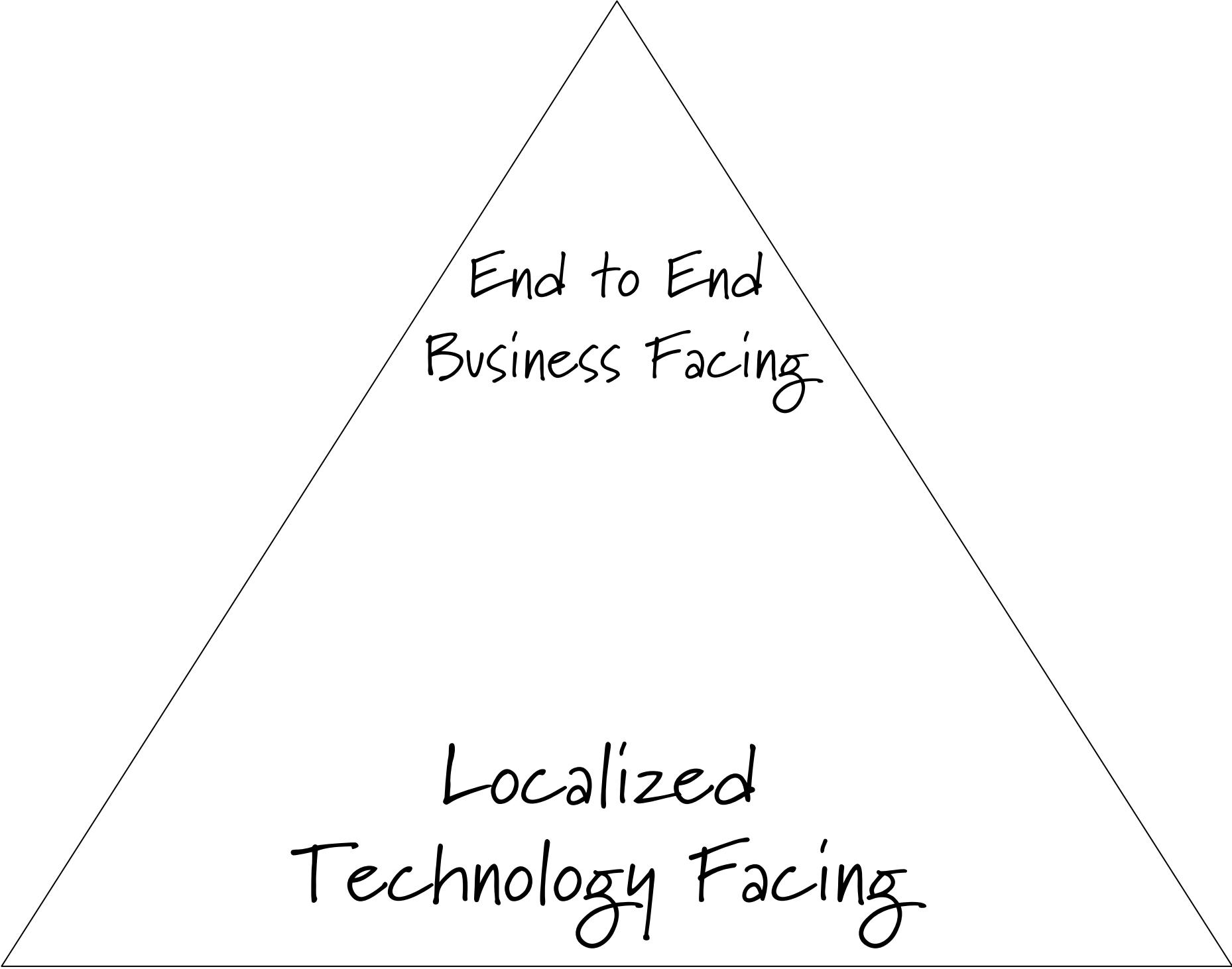


Diagram invented by Brian Marick



*Mike Cohn:
Succeeding with Agile*



End to End
Business Facing

Localized
Technology Facing

principle 0

Writing good acceptance tests is hard.

(good: when the tests are **green**, we know the software works)

mingle

2006 20 tests 500 LOC 2 minutes	2012 3000 tests 50k LOC 12 hours
---	--

- actual running time: 55 minutes
- 7-10 times a day
- for 6 years, across 4 offices now

why do test suites decay?

for the same reasons code does

we don't pay enough attention to expressing intent

only testers care about maintaining tests

principles



principle 1

Tests are first-class citizens of
your project

preventing decay in test code

treat test code as production code

refactor relentlessly

don't repeat yourself

don't repeat yourself

~~use record playback tools to build your suite~~

preventing decay of intention

given-when-then is insufficient

separate intention from mechanics

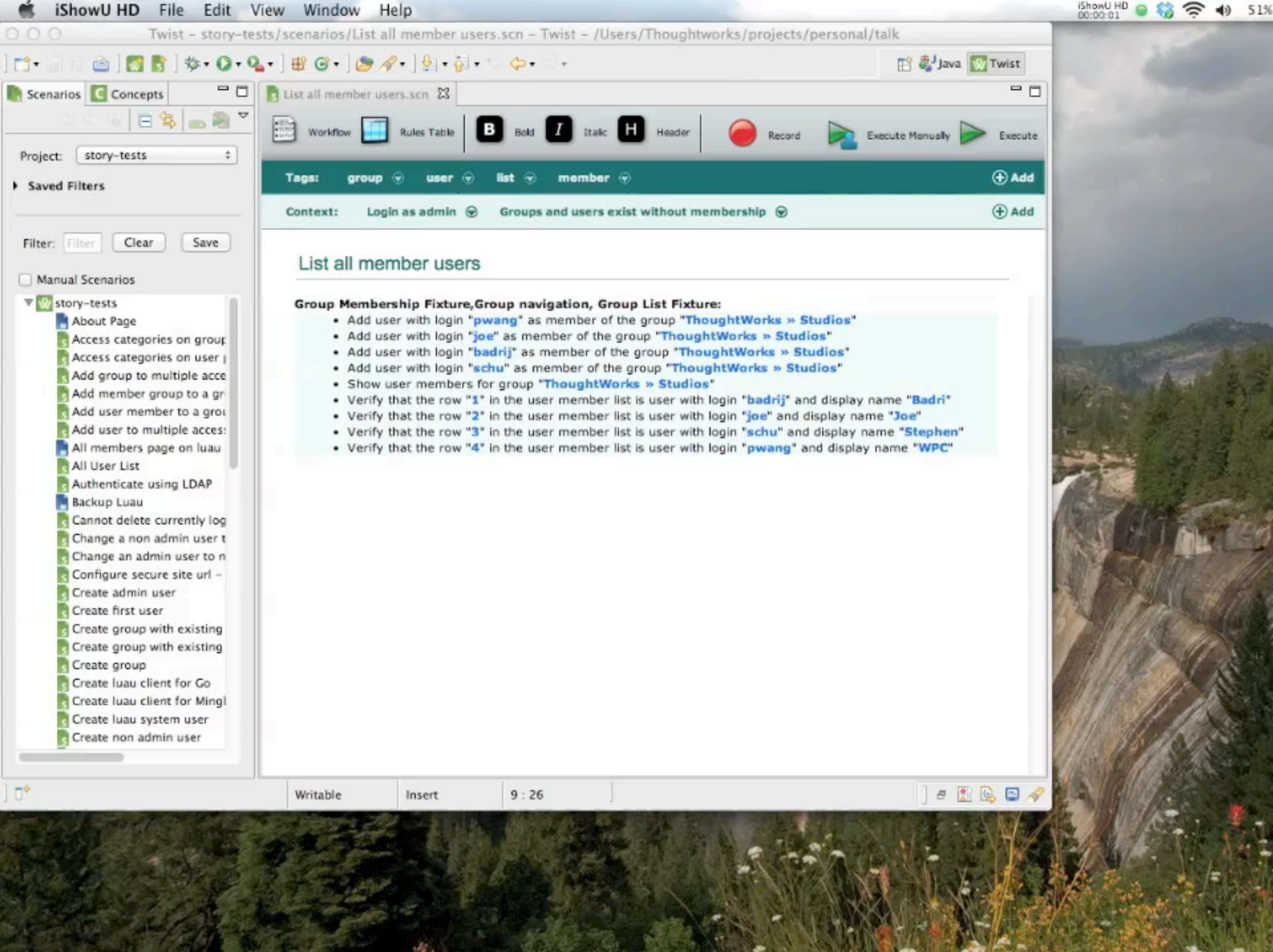
express the test as steps of a user's journey

a solution

use **natural language** to express intentions

use a **general purpose programming language** to express **test mechanics**

use a **tool** that allows you to **operate in either domain** transparently



page object

github gist <https://gist.github.com/3345556>

```
public class LoginPage {

    private final SeleniumSession browser;

    public LoginPage(Selenium browser){
        this.browser = browser;
    }

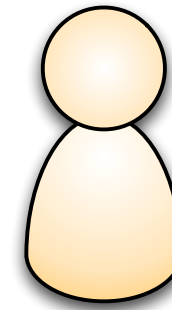
    public HomePage loginAs(String user, String password){
        browser.type('#login', login);
        browser.type('#password', password);
        browser.submit('#login-form');
        return new HomePage(this.browser);
    }

    public HomePage loginExpectingError(String user, String password){
        browser.type('#login', login);
        browser.type('#password', password);
        browser.submit('#login-form');
        return new LoginPage(this.browser);
    }
}
```

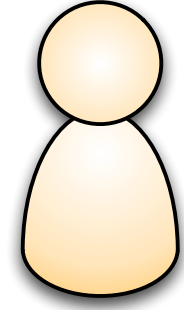

Acceptance Criteria



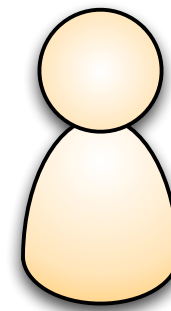
Test implementation



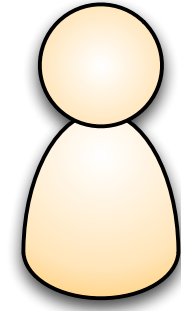
Customer



Tester



Developer



Tester

tester / quality analyst

...is a role, not a person

...is not a failed developer

...advocates for the user and makes the quality of the system transparent

...should not be primarily working on manual regression testing

...should be focussed on exploratory testing & maintaining automated acceptance tests

remember

passing acceptance tests are necessary (but insufficient) for “done”

encapsulate!

the acceptance tests are owned by - and the responsibility of - the team

principle 2

always interact with the system
under test the same way a user
would

browser based tests are unreliable

"the test fails in CI, but when I run the app,
everything seems fine"

usually an indication that test mechanics and user
interaction patterns differ

ajax based tests?

JS heavy applications, which need non-zero
processing time to modify the UI

some solutions

Test your application the way a user might use it.

Understand when behavior is asynchronous and account for it explicitly

Don't use bare sleeps: poll

If it's hard to write the test, you need to have a conversation with the team

some solutions

wait-utils (<https://github.com/itspanzi/WaitUtils>)

for ajax tests, if your JS framework provides a pre- and post-call hook, intercept those to count the number of active calls before proceeding

```
var AjaxTracker = {

  PENDING_REQUESTS: $A([]),

  onCreate: function(request) {
    if (!request.url.match(/gadgets\/js\/\//)) {
      this.PENDING_REQUESTS.push(request.url);
    }
  },

  onComplete: function(request) {
    this.PENDING_REQUESTS = this.PENDING_REQUESTS.without(request.url);
  },

  onException: function(request, exception) {
    try {
      this.onComplete(request);
    } catch (e) {
      if (Prototype.isFireDebugEnabled) {
        console.log("Got Exception on request: " + request.url);
        console.log(e);
        throw(e);
      }
    }
  },

  allAjaxComplete: function(includeCardSummary) {
    var requests = this.PENDING_REQUESTS.reject(function(url) {
      return url.match(/cards\/card_summary/) || url.match(/also_viewing/);
    });
    return requests.size() == 0;
  }
};

Ajax.Responders.register(AjaxTracker);
```


remember

make time to **go back and refactor** your tests

use **layers and encapsulation**: separate high level intent and low level mechanics

use **page object** to interact with SUT; run against service layer where possible

principle 3

continuously curate the structure
of your test suites

#1301

As a... I want... So that...

Given...
When...
Then...

Given...
When...
Then...

#1306

As a... I want... So that...

Given...
When...
Then...

Given...
When...
Then...

#1310

As a... I want... So that...

Given...
When...
Then...

Given...
When...
Then...

#1312

As a... I want... So that...

Given...
When...
Then...

Given...
When...
Then...

#1315

As a... I want... So that...

Given...
When...
Then...

Given...
When...
Then...

#1308

As a... I want... So that...

Given...
When...
Then...

Given...
When...
Then...

that...

Given...
When...
Then...

#1307

As a... I want... So that...

Given...
When...
Then...

Given...
When...
Then...

#1317

As a... I want... So that...

Given...
When...
Then...

Given...
When...
Then...

#1311

As a... I want... So that...

Given...
When...
Then...

Given...
When...
Then...

#1313

As a... I want... So that...

Given...
When...
Then...

Given...
When...
Then...

#1309

As a... I want... So that...

Given...
When...
Then...

Given...
When...
Then...

#1305

As a... I want... So that...

Given...
When...
Then...

Given...
When...
Then...

#1302

As a... I want... So that...

Given...
When...
Then...

Given...
When...
Then...

#1316

As a... I want... So that...

Given...
When...
Then...

Given...
When...
Then...

#1318

As a... I want... So that...

Given...
When...
Then...

Given...
When...
Then...

#1314

As a... I want... So that...

Given...
When...
Then...

Given...
When...
Then...

journey tests

Buy Product

Search product catalogue
Add product to cart
Check out
Create new account
Provide address details
Provide credit card details
Complete order
Verify order created
Verify credit card debited
Verify email sent

#1612

As a customer
I want a gift wrapping option
So that I don't have to wrap
them and post them myself

Buy Product

Search product catalogue
Add product to cart
Check out
Create new account
Provide address details
Provide credit card details
Select gift wrapping option
Complete order
Verify order created
Verify gift wrapping option
Verify credit card debited
Verify email sent

some solutions

identify user journeys

(journey: the path a persona takes through the application to achieve an end goal)

most applications have very **few distinct personas**

most stories in iterative development are **enhancements** to existing journeys

features

Basic shopping cart functionality

Searching for a product

- searching for a word should bring up all products which have that word in their name
- searching for a phrase should bring up all products which have any of the words in their name
- searching for a quoted phrase should bring up all products which have all words in the the quoted phrase in their name

Paginating search results

- return 25 results per page by default
- if there are fewer than 25 results, do not show pagination links
- provide a "previous" link on every page other than the first page of results
- provide a "next" link on every page other than the last page of results
- if user supplies a page number which is less than 1 in the URL, stay on the first page
- if the user supplies a page number greater than the last page of results, stay on the last page

Gift-wrap option

story tests

Story tests for search

- test that searching for "friends" brings back 782 results
- results should include how to win friends and influence people

- test that searching for dead friends brings back 8900 results
- results should include <how to win friends and influence people>
- results should include <The Zombie Survival Guide: Complete Protection from the Living Dead>

- test that searching for "dead friends" brings back 57 results
- results should include <all my friends are dead>

Story tests for pagination

- with a catalog of 3 products, I should see no pagination links
- with a catalog of 25 products, I should see no pagination links
- with a catalog of 26 products, I should see 1 link to page two, along with a next link but no previous link
- with a catalog of 26 products, on page 2, I should see one product, with a link to page one, a previous link but no next link

Story tests for gift wrapping

journey tests

Journey of user buying a book

- Login as user "bob"
- Search for <"my friends" dead>
- Make sure that 3 pages of results show
- Verify that "All My Friends Are Dead" by "Avery Monson" is on the first page
- Add two copies of the book to the shopping cart
- Gift wrap one of them
- Proceed to checkout

more solutions

extract journeys from your acceptance tests

make them **fast** and run them first

do **test the most likely path** that the team, business and UX folk agree upon

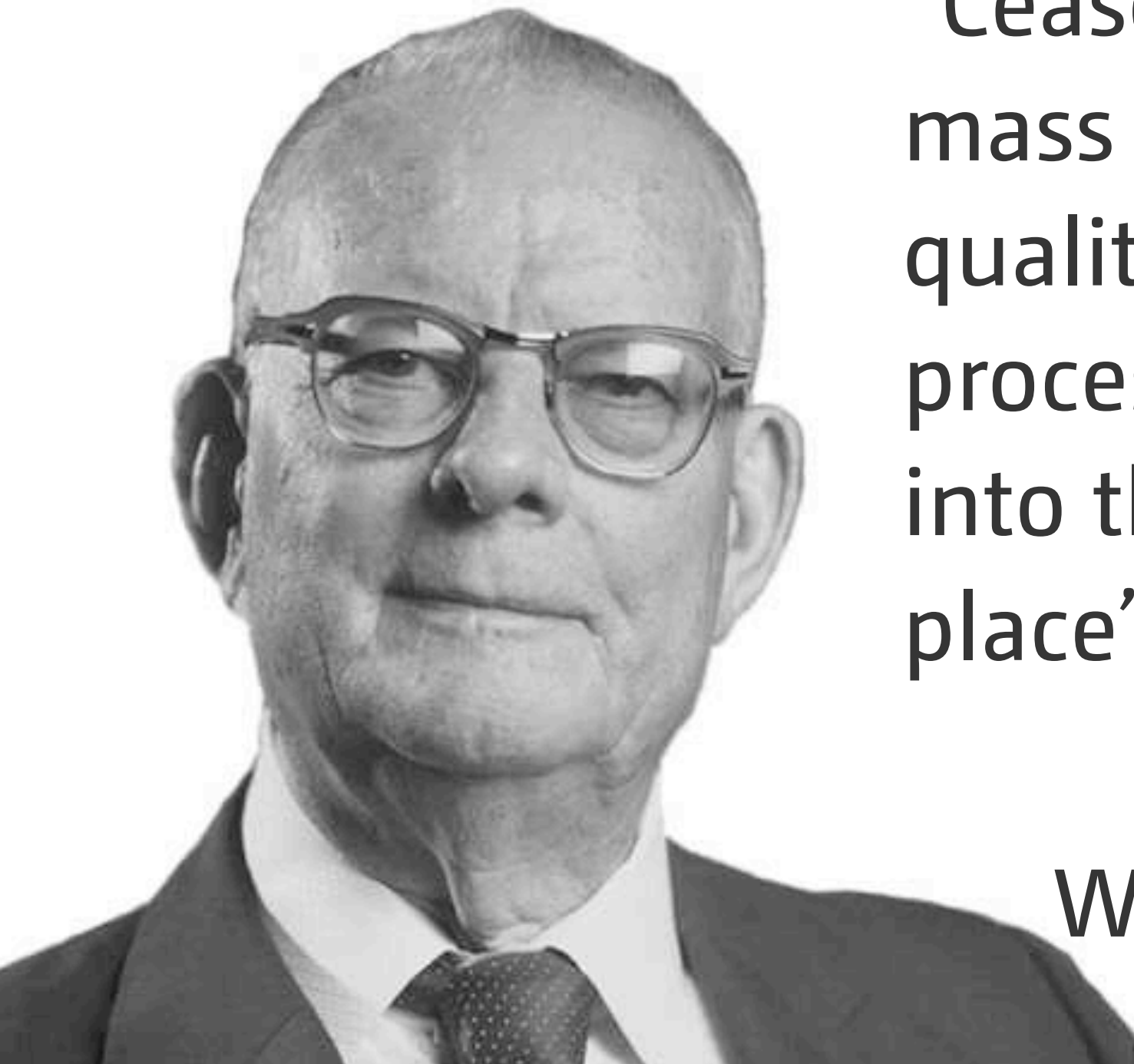
do **not test every possible path** through the system

extract negative tests and edge cases into a **regression suite** which runs after your journey tests

build quality in

“Cease dependence on mass inspection to achieve quality. Improve the process and build quality into the product in the first place”

W. Edwards Deming



why cross-functional teams?

output of testing is not just bug reports

feedback from testing to product design

feedback from test framework to system architecture

developers and testers share knowledge and skills

principle 4

everyone owns acceptance tests

when acceptance tests break

Triage to find root cause

1. There was an environmental problem
2. There is a bug with the test
3. An assumption changed
4. The test actually caught a bug

Fix the problem

Add a guard to prevent it happening again

Optimise your test suite: detect failures fast

Optimise your process for time to fix tests

intermittent failures

flaky tests are *worse than* useless

quarantine flaky tests - but not forever

[http://martinfowler.com/articles/
nonDeterminism.html](http://martinfowler.com/articles/nonDeterminism.html)

external systems

not all tests should call the external system

parameterize connections to external systems

Run integration smoke tests before full acceptance suite

impersonator pattern

create a proxy from SUT to external system

cache results from integration smoke tests

run integration smoke tests before acceptance suite

periodically expire cache

only run acceptance suite if integration smoke tests
pass!

principle 5

acceptance tests are responsible
for managing their own test data

test data

Test-specific data

Test reference data

Application reference data

Ensure tests are independent

Don't use production data dumps (except for performance testing and staging)

recap

1. treat acceptance tests like production code
2. always interact with the SUT like a user would
3. continuously curate your user journeys
4. collective ownership of acceptance tests
5. acceptance tests own their data

take-aways

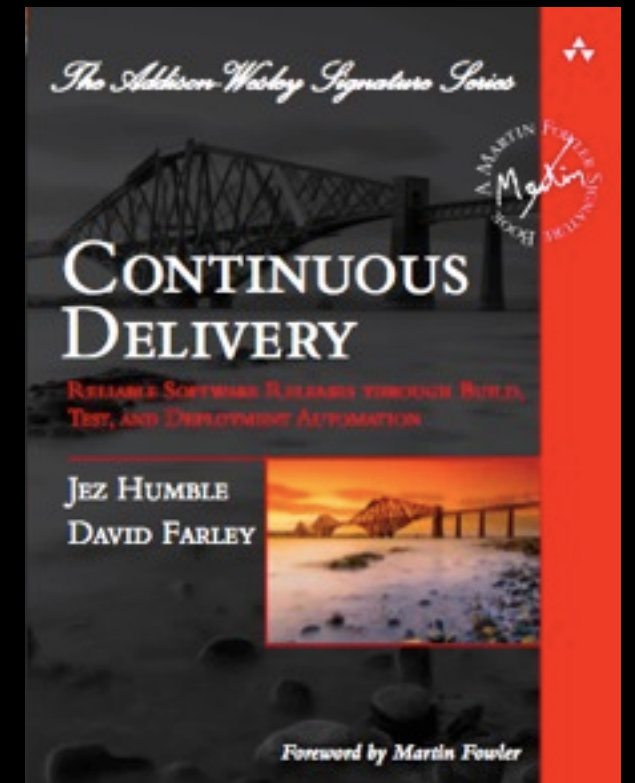
- quality is everybody's responsibility
- high quality test suites are continuously curated
 - by testers and developers working together
- test code needs to receive the same care as production code
- exhaustive story-level testing is not a good basis for maintainable acceptance suites

questions

@badrij | badri@thoughtworks.com

@jezhumble | jez@thoughtworks.com

<http://continuousdelivery.com/>



© 2011 ThoughtWorks, Inc.

ThoughtWorks
STUDIOS

<http://thoughtworks-studios.com/>

