

Software Testing

@Steve_Upton

Who am I?

- Steve Upton
- English / Welsh / British / Irish(?)
- BSc. Computer Science (Cardiff University)
- Physics & Astronomy (Open University)
- IBM (6 years)
 - Messaging
 - OASIS MQTT TC member
 - Working with clients on Microservice systems
 - London μService user group
- HERE Berlin (9 months)
 - Microservices
 - Robots!

Who are you?



Exclusive use of white box testing in a test-phase will:

- a) Ensure the test item is adequately tested
- b) Make the need for black-box testing redundant
- c) Run the risk that the requirements are not satisfied
- d) Suffice for the unit testing phase

Which is not in sequence in Step 11
of the Software Testing process?

- a) Assess development plan and status
- b) Develop the test plan
- c) Test software design
- d) Test software requirement

Software Test Documentation is described in which standard?

- a) IEEE 730-2014
- b) IEEE 829-2008
- c) IEEE 830-1984
- d) IEEE 1012-2012





A note on Quality

Understand importance testing

Understand how to test well

Be ready for a testing role

Be ready for a test interview!

Why do we test?













CHINA'S CHANGZHENG 3C ROCKET IS SEEN DURING PREPARATION FOR LAUNCH

```
L_M_BV_32 := TDB.T_ENTIER_32S ((1.0/C_M_LSB_BV) *  
                                G_M_INFO_DERIVE(T_ALG.E_BV));  
if L_M_BV_32 > 32767 then  
  P_M_DERIVE(T_ALG.E_BV) := 16#7FFF#;  
elsif L_M_BV_32 < -32768 then  
  P_M_DERIVE(T_ALG.E_BV) := 16#8000#;  
else  
  P_M_DERIVE(T_ALG.E_BV) := UC_16S_EN_16NS(TDB.T_ENTIER_16S(L_M_BV_32));  
end if;  
501 P_M_DERIVE(T_ALG.E_BH) := UC_16S_EN_16NS ((TDB.T_ENTIER_16S  
                                              ((1.0/C_M_LSB_BH) *  
                                               G_M_INFO_DERIVE(T_ALG.E_BH)))  
end LIRE_DERIVE;
```

```
L_M_BV_32 := TDB.T_ENTIER_32S ((1.0/C_M_LSB_BV) *  
                                G_M_INFO_DERIVE(T_ALG.E_BV));  
if L_M_BV_32 > 32767 then  
    P_M_DERIVE(T_ALG.E_BV) := 16#7FFF#;  
elsif L_M_BV_32 < -32768 then  
    P_M_DERIVE(T_ALG.E_BV) := 16#8000#;  
else  
    P_M_DERIVE(T_ALG.E_BV) := UC_16S_EN_16NS(TDB.T_ENTIER_16S(L_M_BV_32));  
end if;  
501 P_M_DERIVE(T_ALG.E_BH) := UC_16S_EN_16NS ((TDB.T_ENTIER_16S  
                                              ((1.0/C_M_LSB_BH) *  
                                               G_M_INFO_DERIVE(T_ALG.E_BH)))  
end LIRE_DERIVE;
```

\$370 million!

A80501-66
SX837

intel®
pentium™
PROCESSOR

L4142580
INTEL® ©1992

4,195,835
3,145,727



$$\frac{4,195,835}{3,145,727} = 1.333820449136241002$$



$$\frac{4,195,835}{3,145,727} = 1.333820449136241002$$

$$\frac{4,195,835}{3,145,727} = 1.333\textcolor{red}{739068902037589}$$



$$\frac{4,195,835}{3,145,727} = 1.333820449136241002$$

$$\frac{4,195,835}{3,145,727} = 1.333\textcolor{red}{739068902037589}$$

\$475 million!



So, why do we test?

What is the purpose of a tester?

To write lots of tests?

To write lots of tests?

No

To run lots of tests?

To run lots of tests?

Noooooooooooooooooooo

To run lots of tests?

Noooooooooooooooooooo



Find all the bugs?

Find all the bugs?



Find all the bugs?



Errata



Errata

SKL001	Reported Memory Type May Not Be Used to Access the VMCS and Referenced Data Structures
Problem	Bits 53:50 of the IA32_VMX_BASIC MSR report the memory type that the processor uses to access the VMCS and data structures referenced by pointers in the VMCS. Due to this erratum, a VMX access to the VMCS or referenced data structures will instead use the memory type that the MTRRs (memory-type range registers) specify for the physical address of the access.
Implication	Bits 53:50 of the IA32_VMX_BASIC MSR report that the WB (write-back) memory type will be used but the processor may use a different memory type.
Workaround	Software should ensure that the VMCS and referenced data structures are located at physical addresses that are mapped to WB memory type by the MTRRs.
Status	For the steppings affected, see the Summary Table of Changes.

SKL002	Instruction Fetch May Cause Machine Check if Page Size and Memory Type Was Changed Without Invalidation
Problem	This erratum may cause a machine-check error (IA32_MCI_STATUS.MCACOD=0150H) on the fetch of an instruction that crosses a 4-KByte address boundary. It applies only if (1) the 4-KByte linear region on which the instruction begins is originally translated using a 4-KByte page with the WB memory type; (2) the paging structures are later modified so that linear region is translated using a large page (2-MByte, 4-Mbyte, or 1-GByte) with the UC memory type; and (3) the instruction fetch occurs after the paging-structure modification but before software invalidates any TLB entries for the linear region.
Implication	Due to this erratum an unexpected machine check with error code 0150H may occur, possibly resulting in a shutdown. Intel has not observed this erratum with any commercially available software.
Workaround	Software should not write to a paging-structure entry in a way that would change, for any linear address, both the page size and the memory type. It can instead use the following algorithm: first clear the P flag in the relevant paging-structure entry (e.g., PDE); then invalidate any translations for the affected linear addresses; and then modify the relevant paging-structure entry to set the P flag and establish the new page size and memory type.
Status	For the steppings affected, see the Summary Table of Changes.

SKL003	Execution of VAESIMC or VAESKEYGENASSIST With An Illegal Value for VEX.vvvv May Produce a #NM Exception
Problem	The VAESIMC and VAESKEYGENASSIST instructions should produce a #UD (Invalid-Opcode) exception if the value of the vvvv field in the VEX prefix is not 111b. Due to this erratum, if CR0.TS is "1", the processor may instead produce a #NM (Device-Not-Available) exception.
Implication	Due to this erratum, some undefined instruction encodings may produce a #NM instead of a #UD exception.

Find bugs?

Find (lots of) bugs?

Find (lots of) bugs?

Impact of the bug matters!

Improve quality?

Improve quality?

Better... but a bit vague

Implications?

Implications?

Write lots of tests?

Run lots of tests?

Find all the bugs?

Find bugs?

Improve quality?

Implications?

Write lots of tests?

You are a code monkey

Run lots of tests?

Find all the bugs?

Find bugs?

Improve quality?

Implications?

Write lots of tests?

You are a code monkey

Run lots of tests?

You are a code monkey

Find all the bugs?

Find bugs?

Improve quality?

Implications?

Write lots of tests?

You are a code monkey

Run lots of tests?

You are a code monkey

Find all the bugs?

Your job is futile

Find bugs?

Improve quality?

Implications?

Write lots of tests?

You are a code monkey

Run lots of tests?

You are a code monkey

Find all the bugs?

Your job is futile

Find bugs?

Your job never ends

Improve quality?

Implications?

Write lots of tests?

You are a code monkey

Run lots of tests?

You are a code monkey

Find all the bugs?

Your job is futile

Find bugs?

Your job never ends

Improve quality?

Everyone has that job

What is the role of a tester?

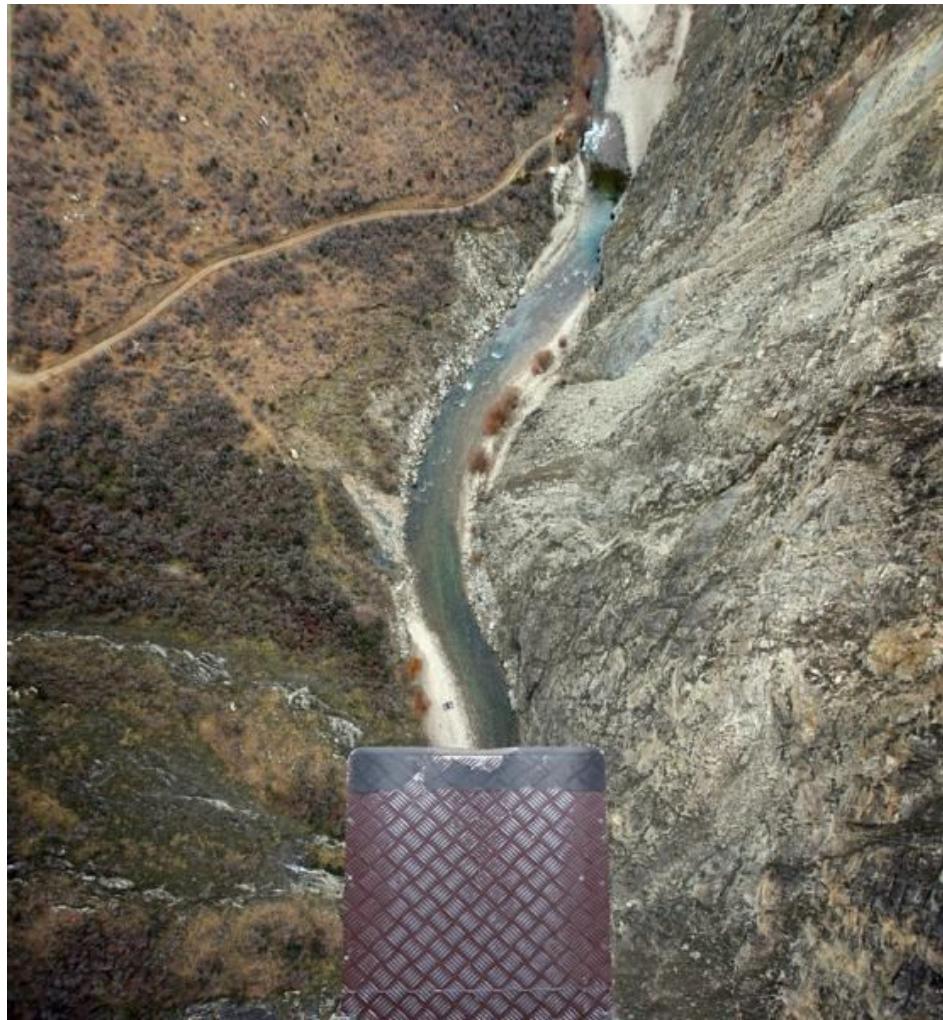
Richard Coppen
@getnextid

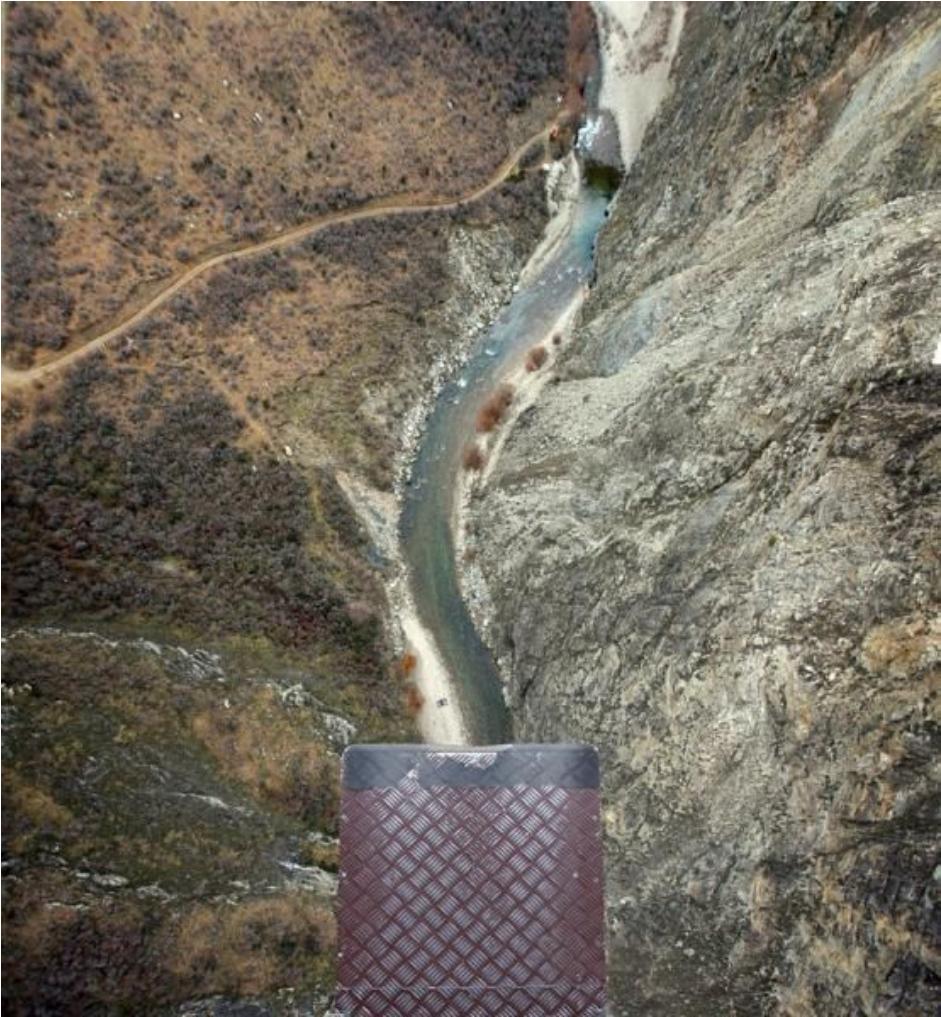












The **role** of a **tester** is to **quantify risk**

The **role** of a **tester** is to **quantify risk**

Then **use that information to make decisions**
that improve confidence and quality



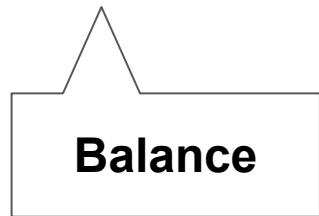
A tester...

increases confidence for stakeholders through
evidence



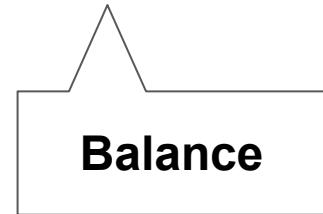
A tester...

increases confidence for stakeholders through
evidence

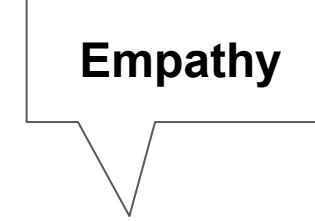


A tester...

increases confidence for **stakeholders** through
evidence



Balance

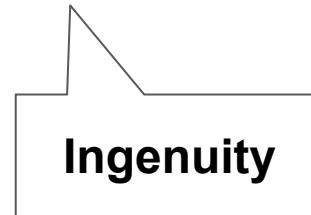
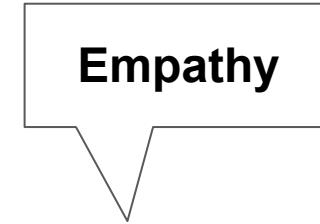
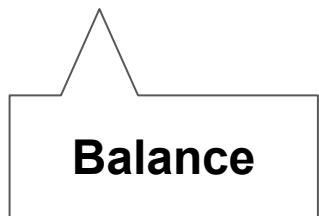


Empathy



A tester...

increases confidence for **stakeholders** through
evidence



How do we test?

Testing Lifecycle

Requirements analysis

Test planning

Test development

Test execution

Test reporting

Test result analysis

Defect Retesting

Regression testing

Test Closure

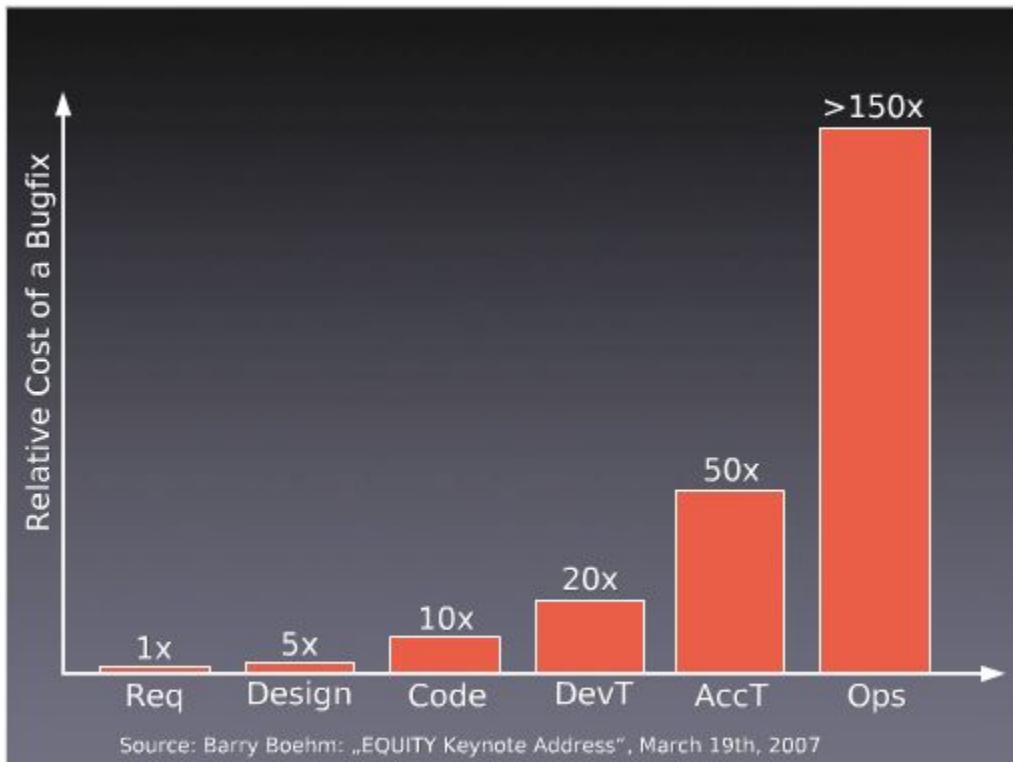
Test strategy

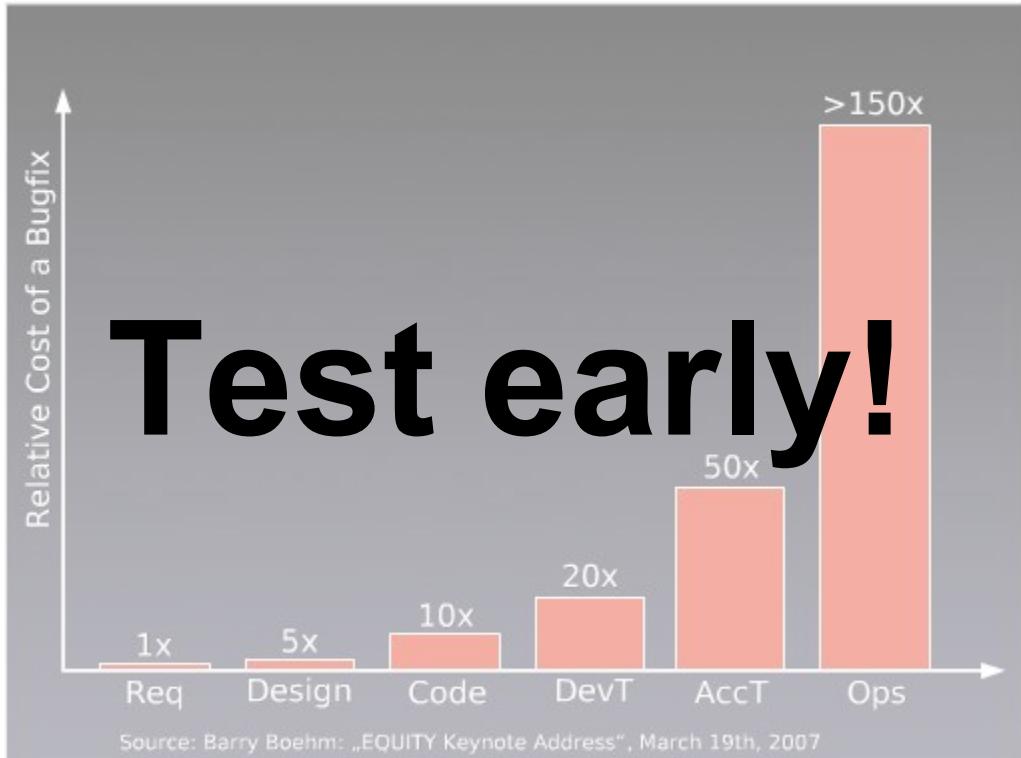
Test techniques

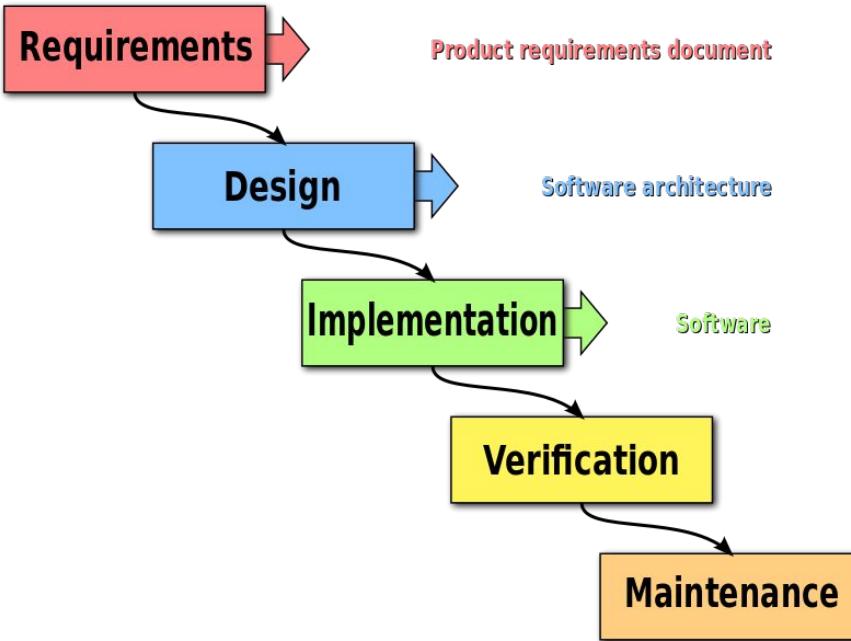
Defect management

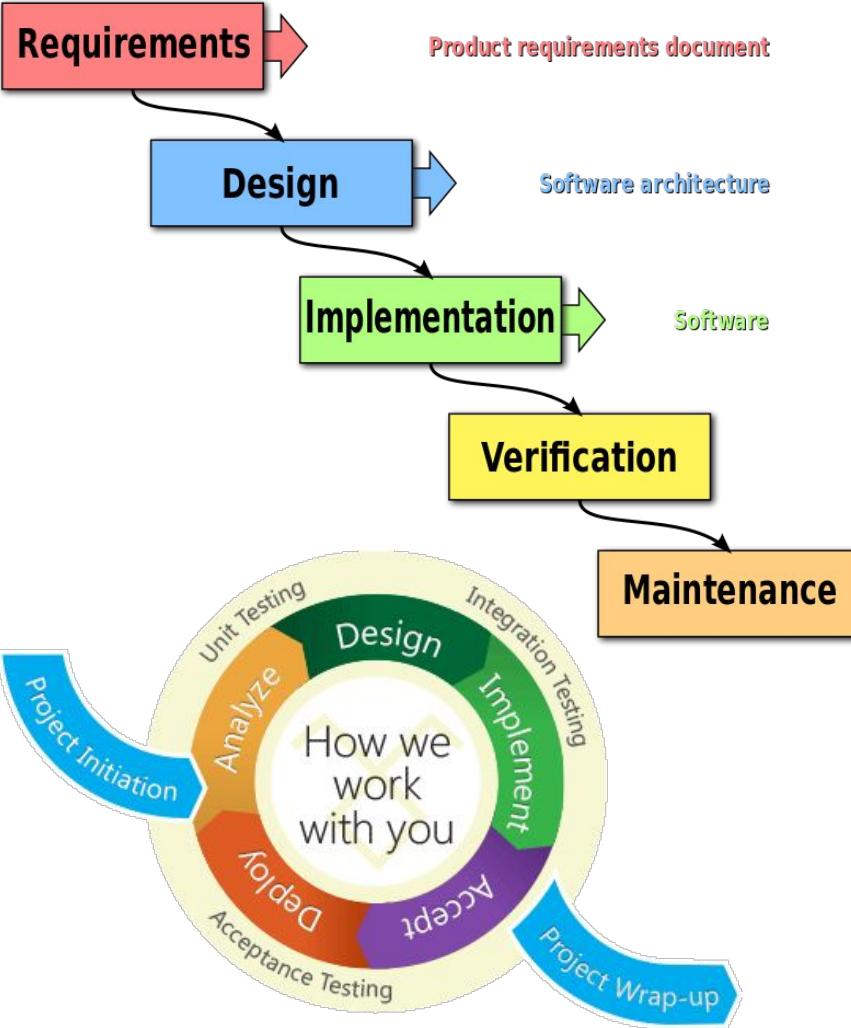
Test metrics

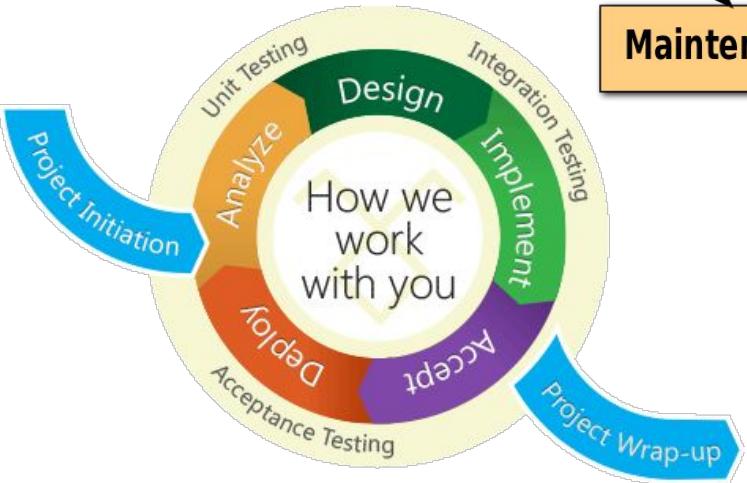
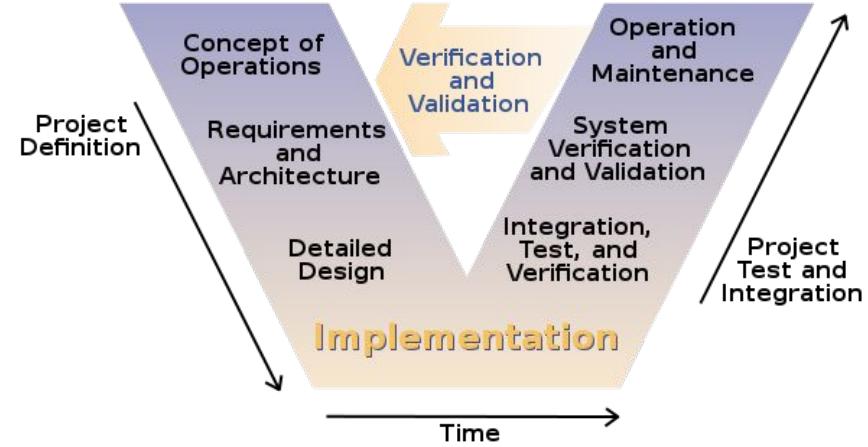
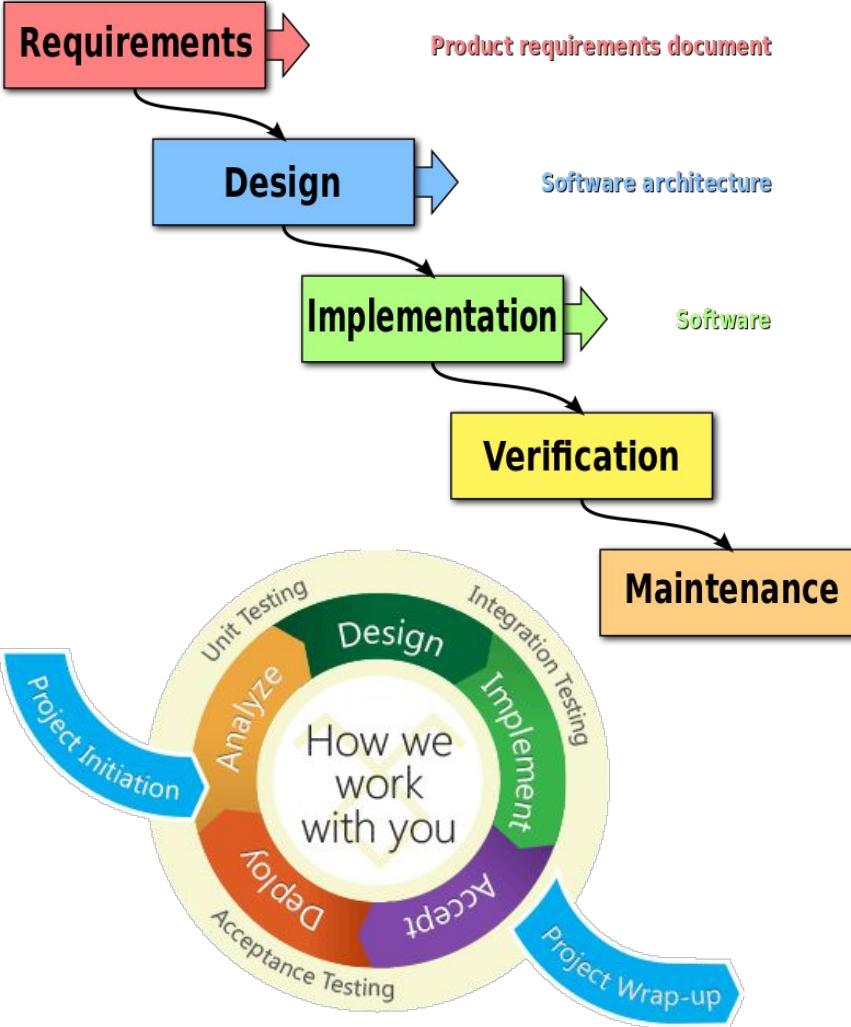
Test Strategy

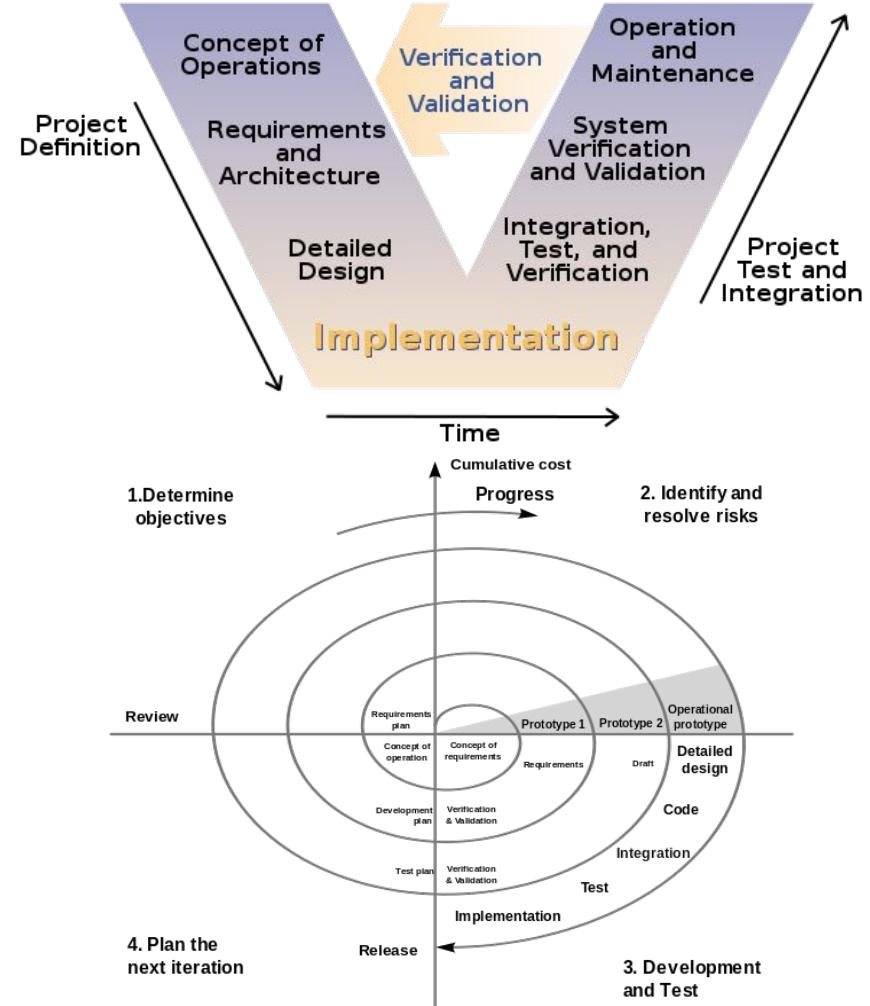
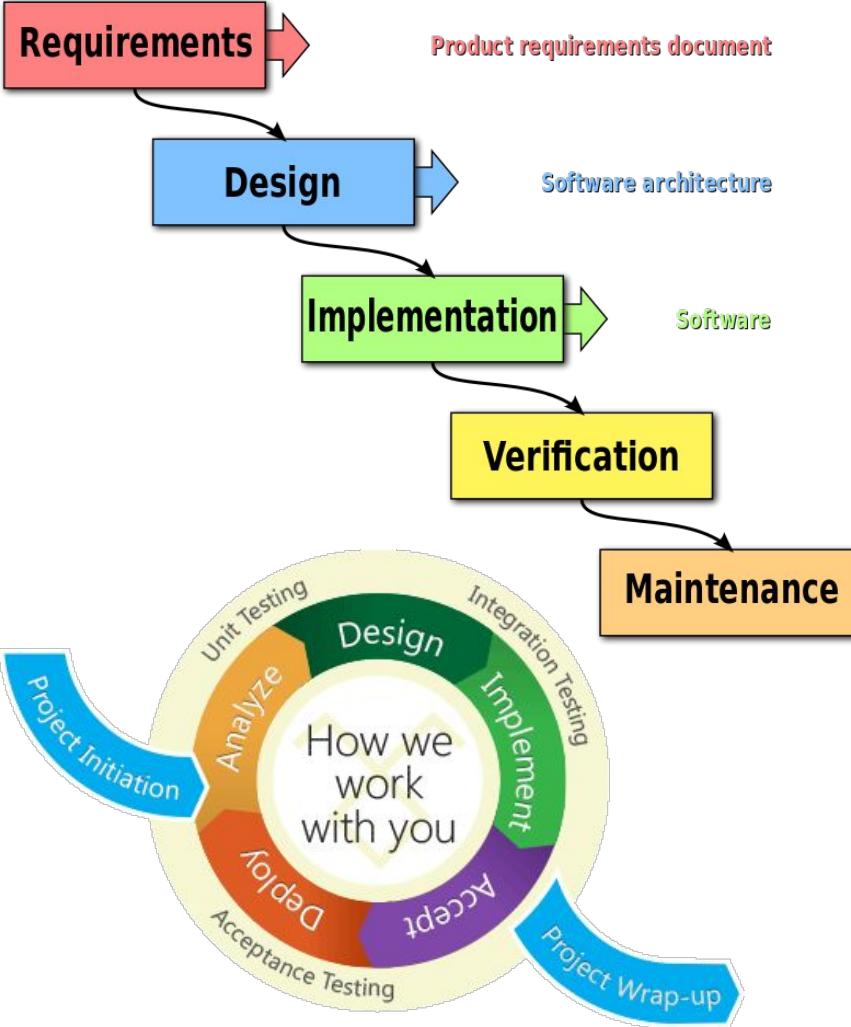












Requirements

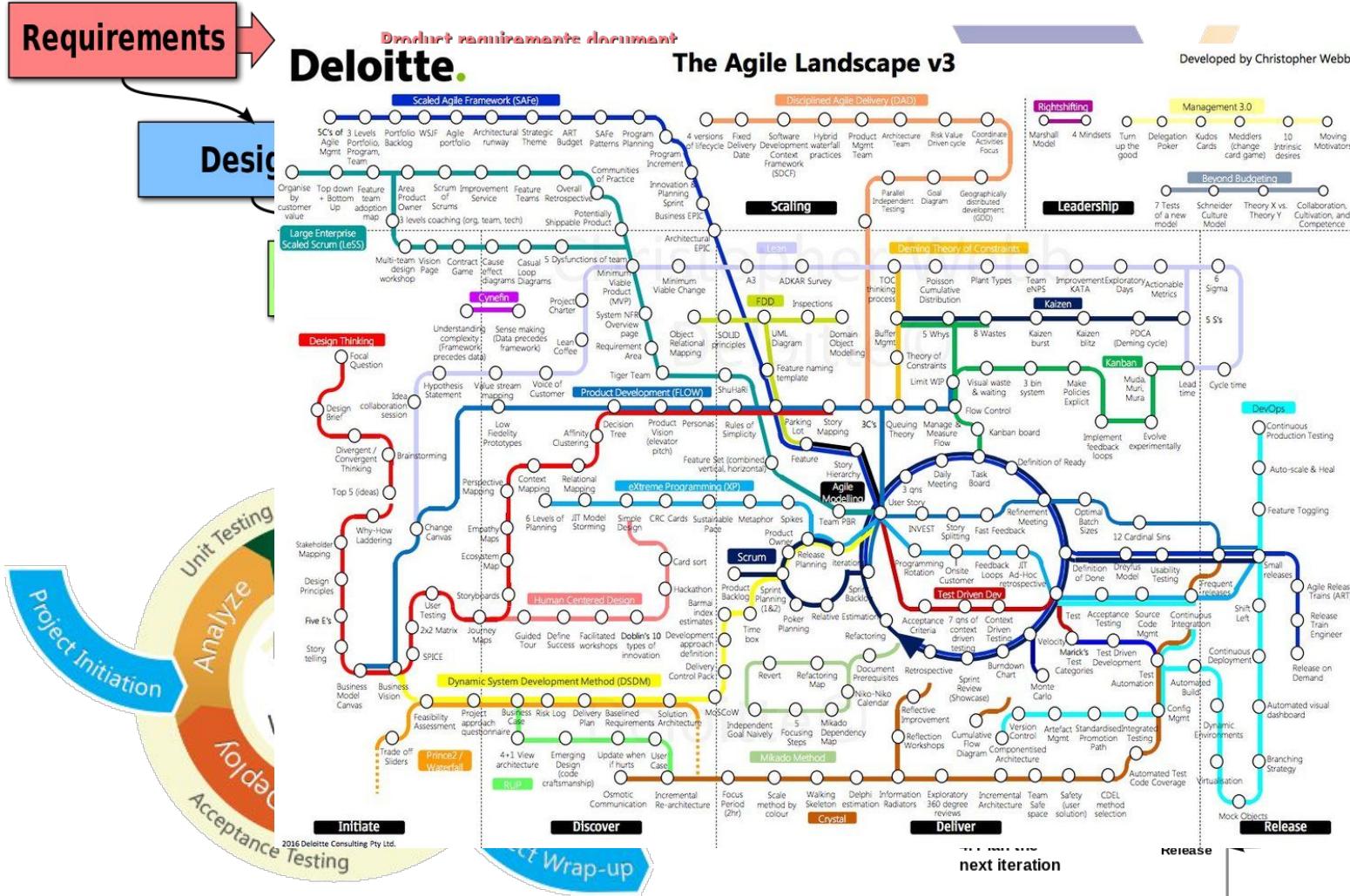
Deloitte.

Product requirements document

The Agile Landscape v3

Developed by Christopher Webb

Design



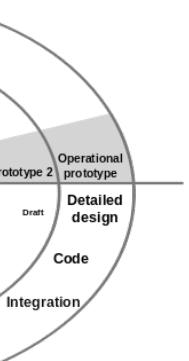
Operation and Maintenance

System Verification, Validation

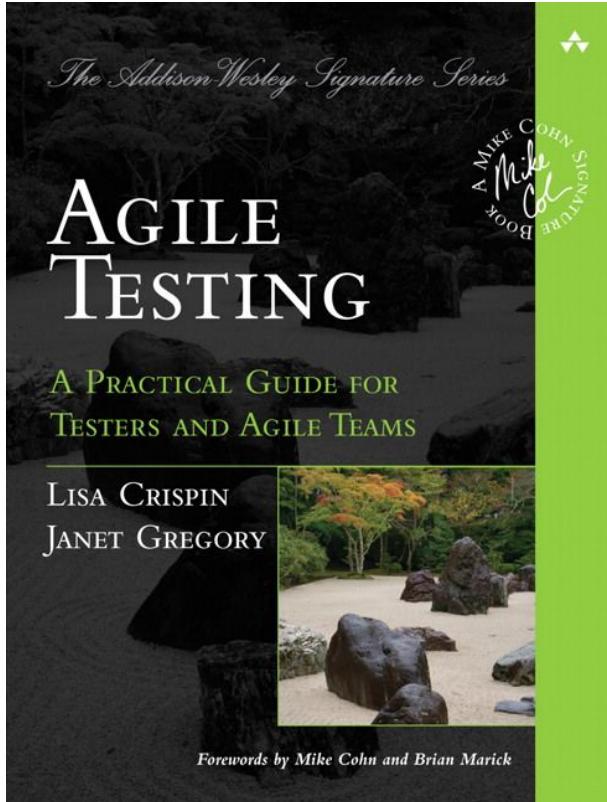
Configuration, and Optimization

→

2. Identify and resolve risks

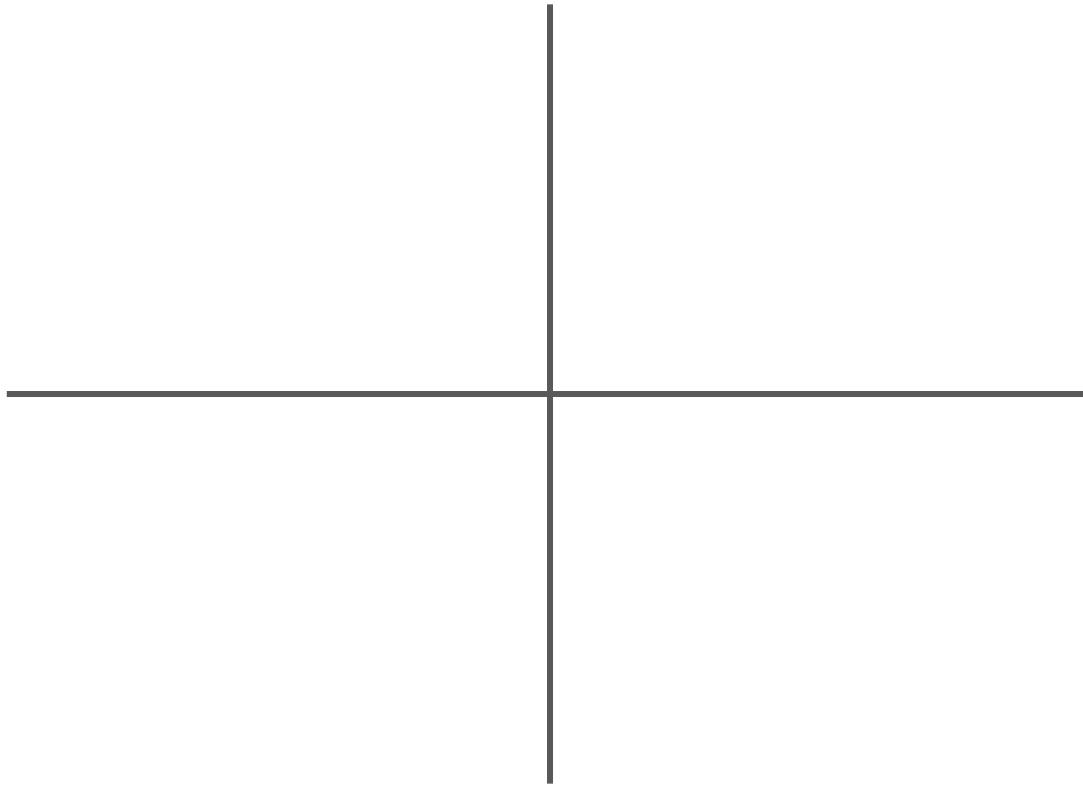


3. Development and Test

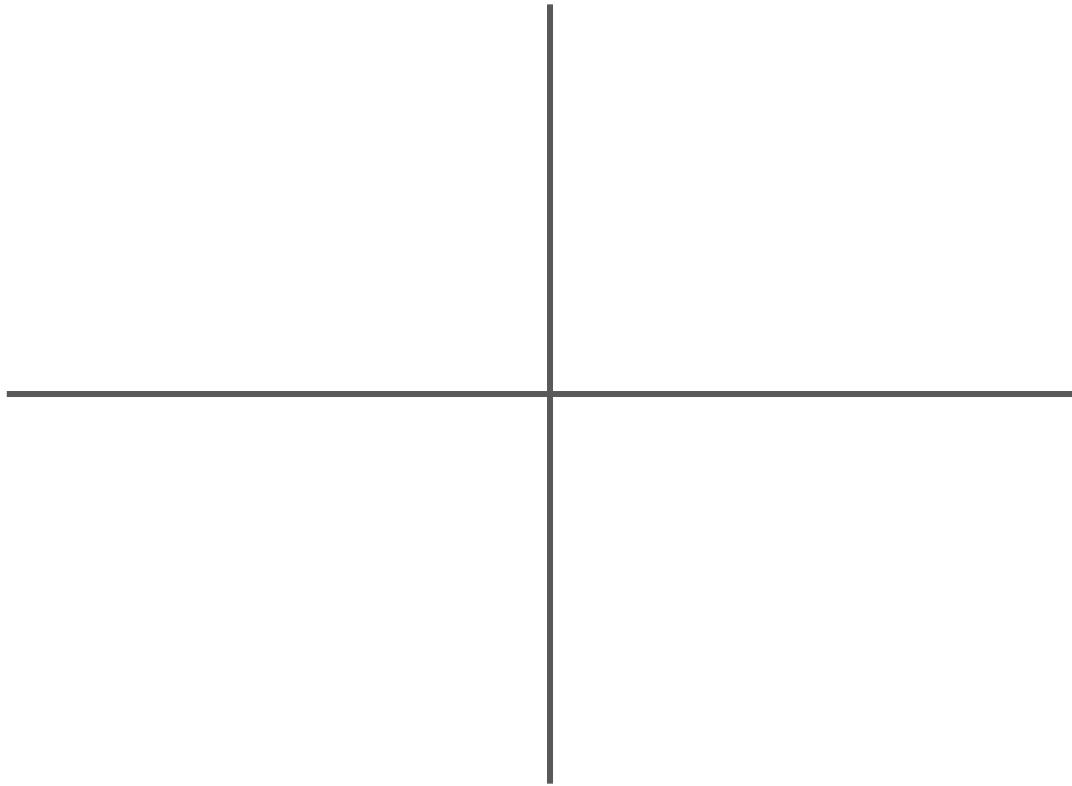


Lisa Crispin
@lisacrispin





Business Facing



Technology Facing

Business Facing

Supporting the team

Critique the product

Technology Facing

Business Facing

Supporting the team

Critique the product

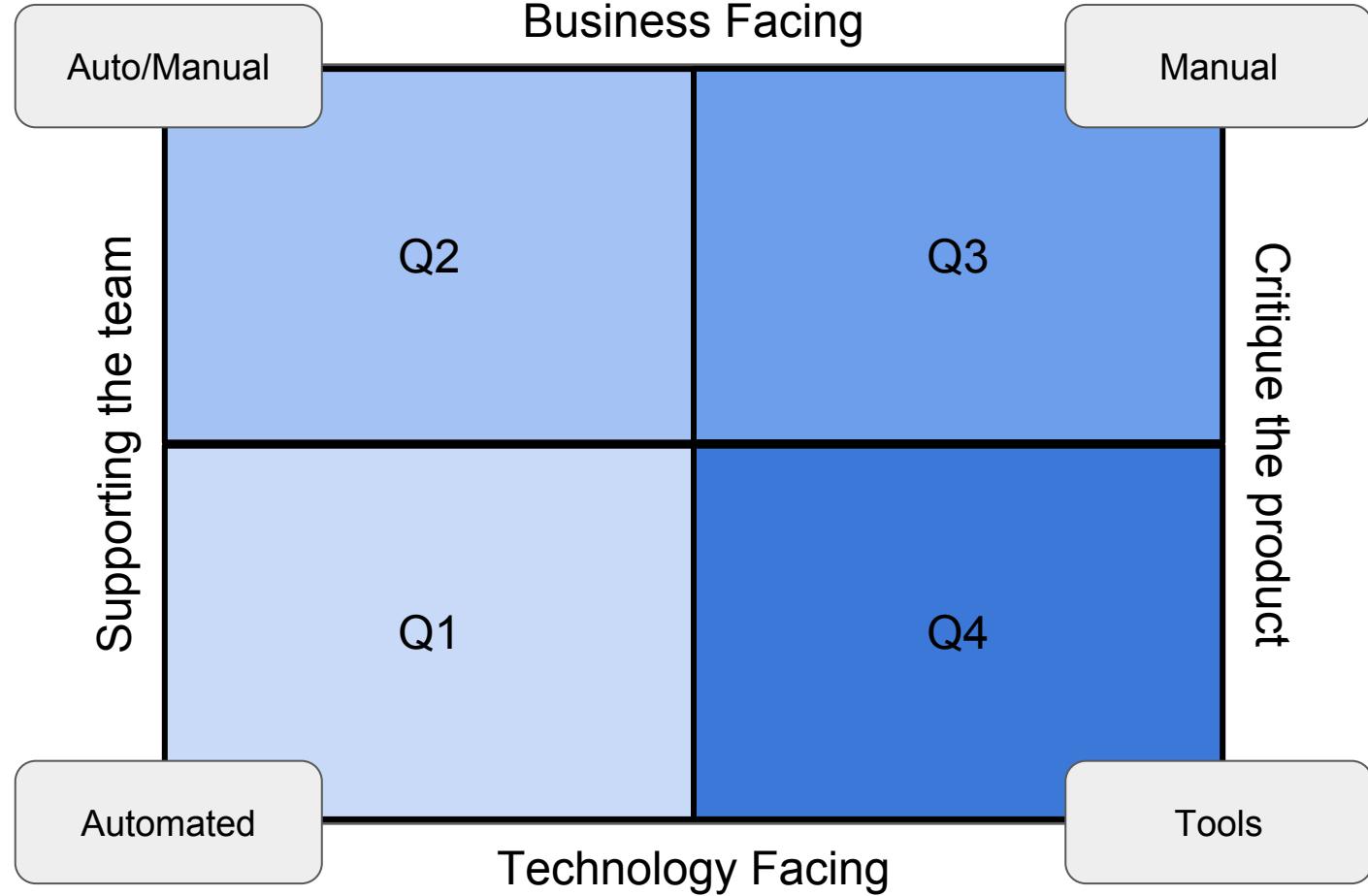
Q2

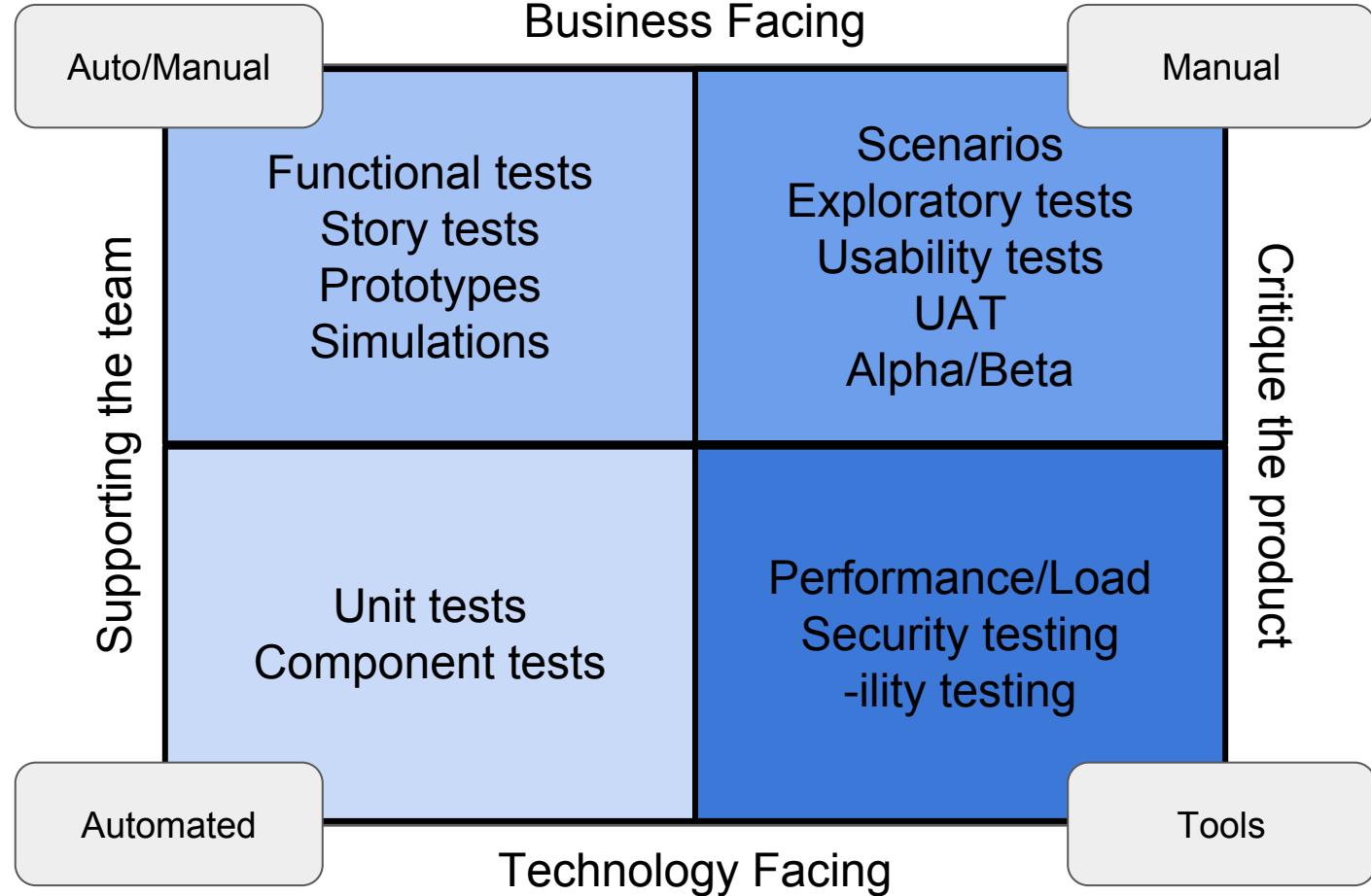
Q3

Q1

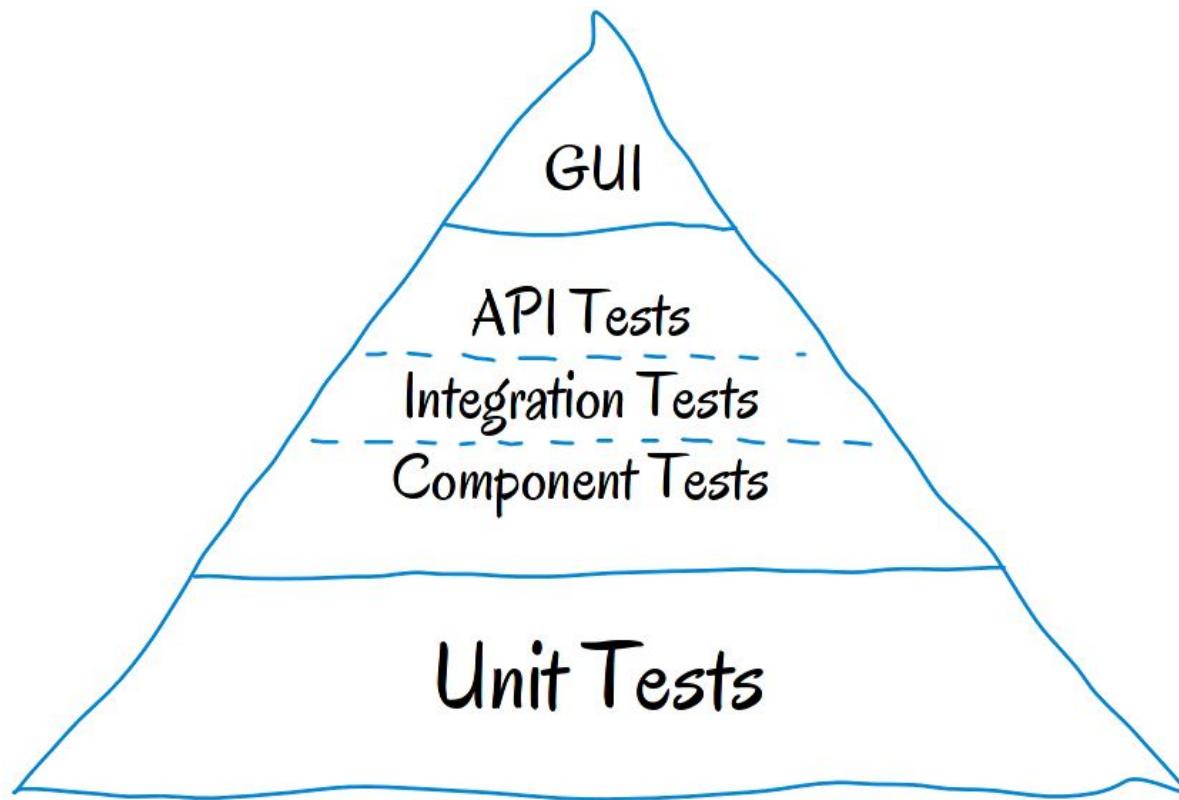
Q4

Technology Facing

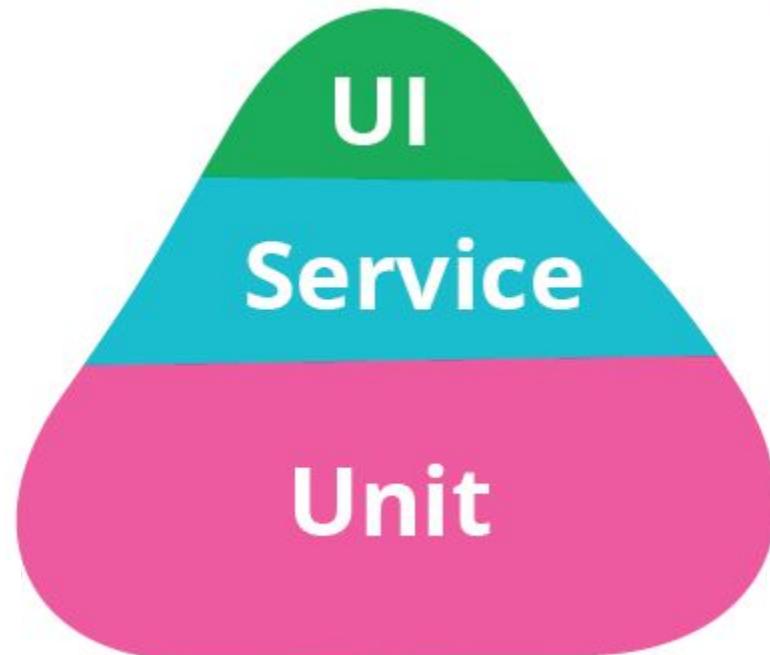




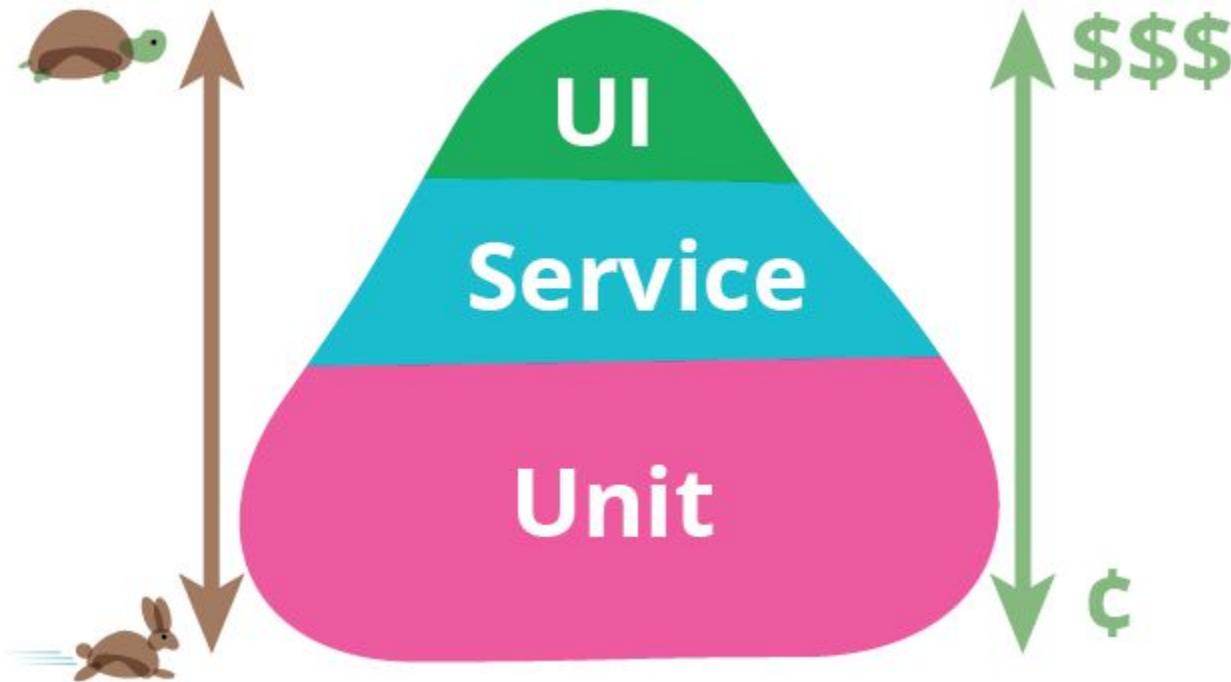
Testing Pyramid



Testing Pyramid



Testing Pyramid



Unit testing

Functional testing

Unit testing

Testing individual units of code

Functions, methods, classes etc.

Generally done by devs

Tests that the code is **doing things right**

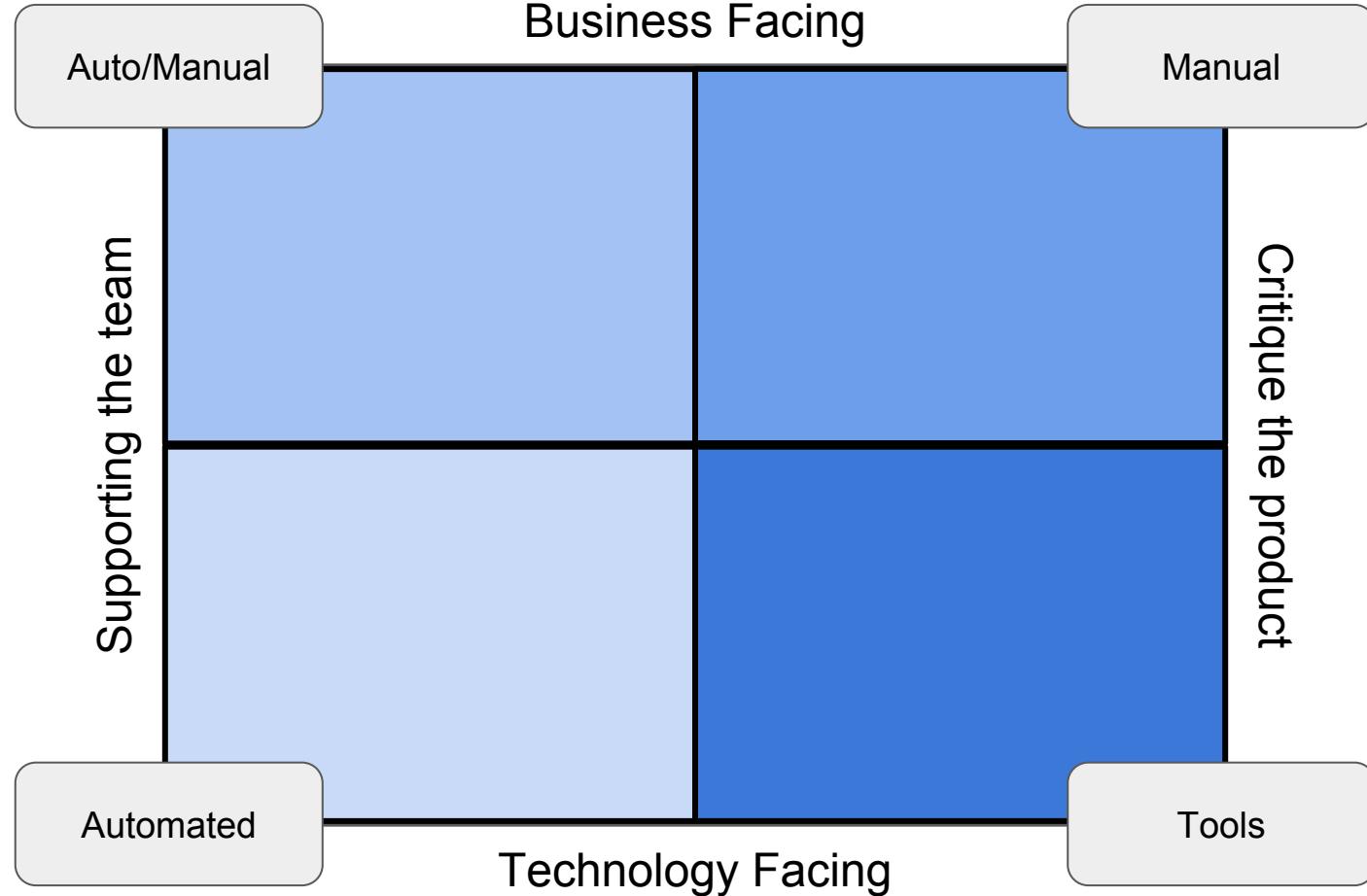
Functional testing

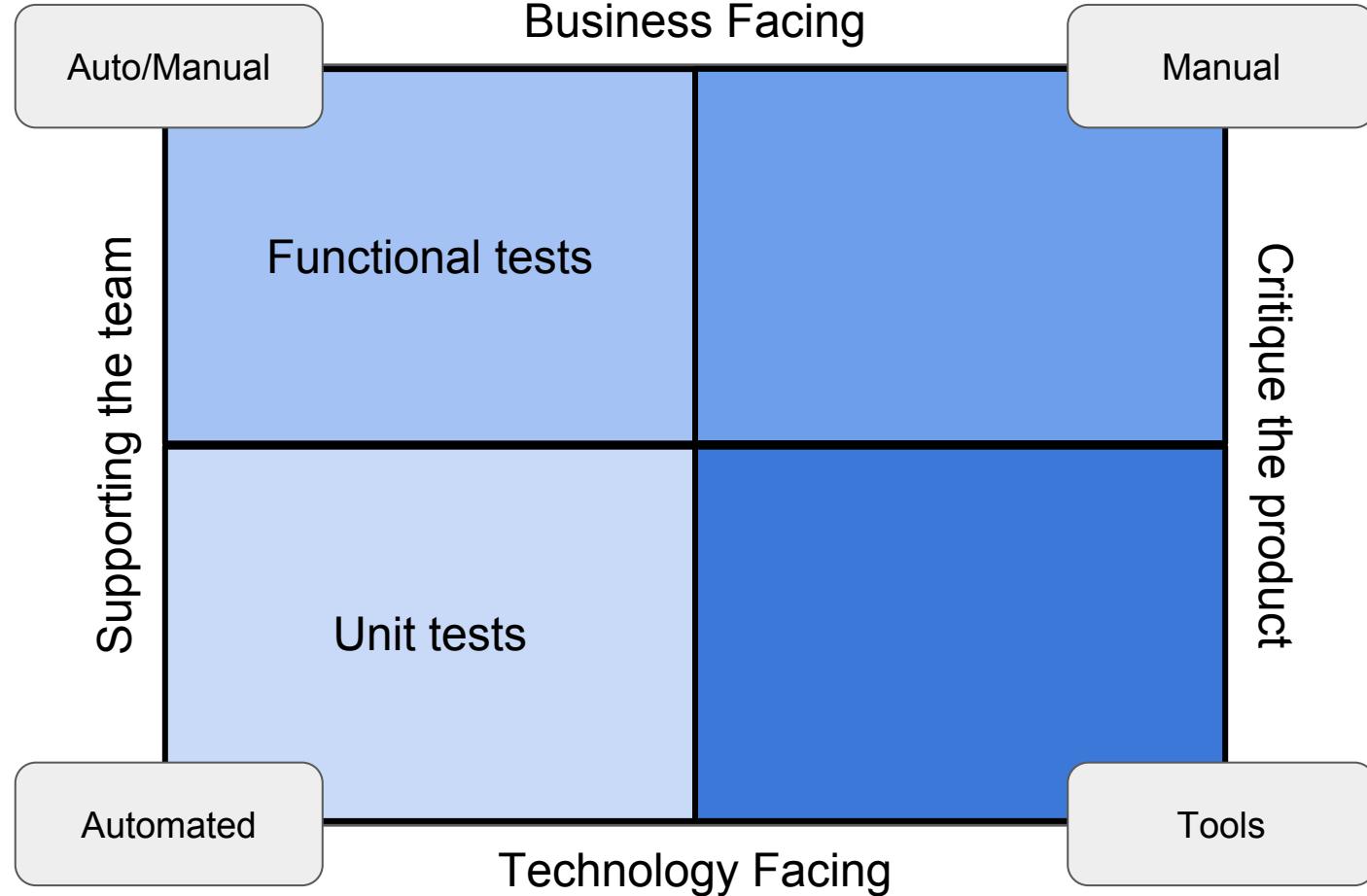
Testing functionality (not a **function!**)

User interfaces

Generally done by testers

Tests that the code is **doing the right thing**





Unit Testing

Let's test... `string.index`

```
string.index(s, sub[, start[, end]])
```

Like `find()` but raise `ValueError` when the substring is not found.

Let's test... `string.index`

`string.index(s, sub[, start[, end]])`

Like `find()` but raise `ValueError` when the substring is not found.

`string.find(s, sub[, start[, end]])`

Return the lowest index in `s` where the substring `sub` is found such that `sub` is wholly contained in `s[start:end]`. Return `-1` on failure. Defaults for `start` and `end` and interpretation of negative values is the same as for slices.

Manual testing

Open a code prompt...

Quick to get going...

... but slow in the long run

```
>>> "abcd".index("b") ;
```

1

Automated testing

Same principle, but **repeatable**

Useful for regression testing

Requires tooling

```
#!/bin/bash

out=`python script.py`

if [ $out = 1 ]; then
    echo "test passed"
else
    echo "test failed"
fi
```

`unittest`

Standard library

xUnit based

Supported by other test runners

`py.test`

`pip install pytest`

xUnit based

No boilerplate, simpler syntax

xUnit

Collection of unit testing frameworks that share structure and functionality

Started with SUnit for Smalltalk

JUnit (Java), RUnit (R), NUnit (.NET) etc.

Object Oriented

xUnit components

Test Case

Test Suite

Test Runner

Test Fixtures

xUnit components - Test Case

xUnit components - Test Case

“A set of test inputs, execution conditions, and expected results developed for a particular objective, such as to exercise a particular program path or to verify compliance with a specific requirement.”

IEEE Standard 610 (1990)

xUnit components - Test Case

“A ~~set of test inputs, execution conditions, and expected results developed for a particular objective, such as to exercise a particular program path or to verify compliance with a specific requirement.~~”

~~IEEE Standard 610 (1990)~~

xUnit components - Test Case

Inputs and expected outputs

Don't get expected results = fail

"A ~~set of test inputs, execution conditions, and expected results developed for a particular objective, such as to exercise a particular program path or to verify compliance with a specific requirement.~~"

~~IEEE Standard 610 (1990)~~

xUnit components - Test Case

Inputs and expected outputs

Don't get expected results = fail

All tests inherit from TestCase class

"A ~~set of test inputs, execution conditions, and expected results developed for a particular objective, such as to exercise a particular program path or to verify compliance with a specific requirement.~~"

~~IEEE Standard 610 (1990)~~

xUnit components - Test Case

Inputs and expected outputs

Don't get expected results = fail

All tests inherit from TestCase class

Test methods start with test

Has at least one assert statement

"A ~~set of test inputs, execution conditions, and expected results developed for a particular objective, such as to exercise a particular program path or to verify compliance with a specific requirement.~~"

~~IEEE Standard 610 (1990)~~

xUnit components - Test Suite

Collection of test cases

All test cases should be independent

Allows us to run specific sets of tests

xUnit components - Test Fixture

Pre-requisites and cleanup need by one or more tests

Generate fake data, temporary databases etc. (**context**)

Includes cleanup!

`unittest` provides `setUp()` and `tearDown()` methods

xUnit components - Test Runner

Runs test cases or test suites and returns results

`unittest` provides `TextTestRunner` but many others available

`HTMLTestRunner` outputs a HTML report

`XMLTestRunner` outputs XML

```
class TestIndex(unittest.TestCase):  
  
    # Tests go here  
  
if __name__ == '__main__':  
    unittest.main()
```

```
def test_my_first_test(self):
```

```
def test_my_first_test(self):  
    alphabet = "abcdefghijklmnopqrstuvwxyz"  
    str1 = "ab"  
    self.assertEqual(alphabet.index(str1), 0)
```

```
$ python main.py
```

```
.
```

```
Ran 1 test in 0.000s
```

```
OK
```

Let's test... `string.index`

`string.index(s, sub[, start[, end]])`

Like `find()` but raise `ValueError` when the substring is not found.

`string.find(s, sub[, start[, end]])`

Return the lowest index in `s` where the substring `sub` is found such that `sub` is wholly contained in `s[start:end]`. Return `-1` on failure. Defaults for `start` and `end` and interpretation of negative values is the same as for slices.

```
def test_value_error(self):  
    with self.assertRaises(ValueError):  
        alphabet = "abcdefghijklmnopqrstuvwxyz"  
        str2 = "not_in_the_alphabet"  
        alphabet.index(str2)
```

```
# Run before each test

def setUp(self):
    self.alphabet = "abcdefghijklmnopqrstuvwxyz"

# Run after each test

def tearDown(self):
    return
```

```
def test_value_error(self):  
    with self.assertRaises(ValueError):  
        self.alphabet.index('not_in_the_alphabet')
```

Let's test... `string.index`

`string.index(s, sub[, start[, end]])`

Like `find()` but raise `ValueError` when the substring is not found.

`string.find(s, sub[, start[, end]])`

Return the lowest index in `s` where the substring `sub` is found such that `sub` is wholly contained in `s[start:end]`. Return `-1` on failure. Defaults for `start` and `end` and interpretation of negative values is the same as for slices.

Golden Path Testing

Error Path Testing

Boundary Value Testing

Equivalence Partitioning

Defect clustering

Black Box/White Box

Golden Path Testing

Golden Path Testing

Test the ideal data

Don't do anything to induce errors

Don't rock the boat

Golden Path Testing

Test the ideal data

Don't do anything to induce errors

Don't rock the boat

Danger of writing 'easy' tests



Error Path Testing

Extreme values ($+\infty$, $-\infty$ etc.)

Invalid values (string in place of int etc.)

Out of date data (expired password etc.)

Unusual values, characters (⌚️ IO 💀 屁)

Null values ("", "", 0, null, undefined)

Let's test...

```
math.factorial(x)
```

Return x factorial. Raises `ValueError` if x is not integral or is negative.

Let's test...

Exhaustive testing is impossible

Equivalence Partitioning

Identify partitions of inputs eg.

- Age 0-18, 18-26, 26+
- 2^n (32-64, 64-128, 128-256...)
- ≤ 1 , 0, ≥ 1

Equivalence Partitioning

Identify partitions of inputs eg.

- Age 0-18, 18-26, 26+
- 2^n (32-64, 64-128, 128-256...)
- ≤ 1 , 0, ≥ 1

Consider each group to be the equivalent

Only need to test 1 value per partition

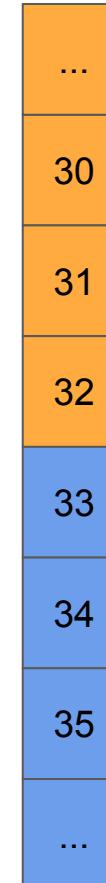
Equivalence Partitioning

Identify partitions of inputs eg.

- Age 0-18, 18-26, 26+
- 2^n (32-64, 64-128, 128-256...)
- ≤ 1 , 0 , ≥ 1

Consider each group to be the equivalent

Only need to test 1 value per partition



Boundary value testing

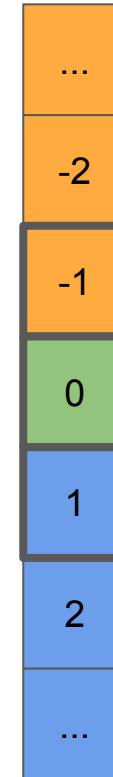
Identify partitions of inputs again

Test values on the edges of the partitions

Boundary value testing

Identify partitions of inputs again

Test values on the edges of the partitions



Defect clustering

Defects tend to be clustered together

80% of bugs in 20% of code (Pareto principle)

Defect clustering

Defects tend to be clustered together

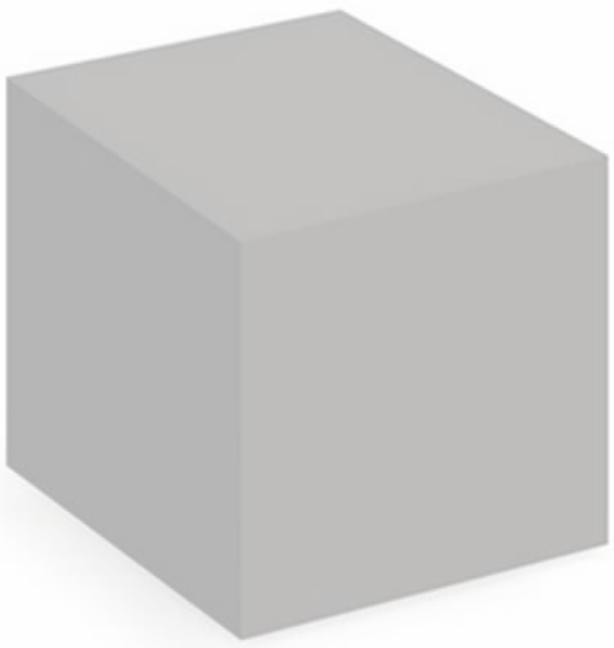
80% of bugs in 20% of code (Pareto principle)

Adapt testing plans based on results!

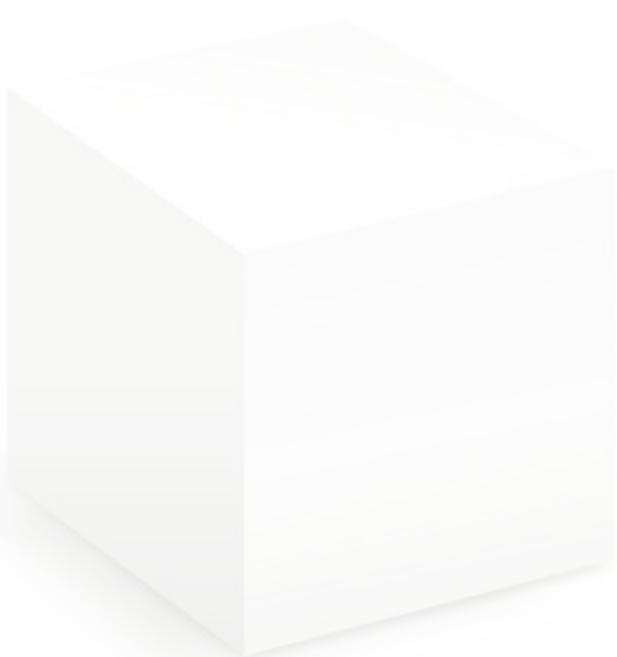
White Box testing



Black Box testing



White Box testing



Black Box testing

Tests based on external interfaces

No knowledge of internals

Intuit error paths (test data selection)

Generally done by testers

White Box testing

Writing tests based on code

Uses knowledge of internals

Identify and test error paths in code

Generally done by developers

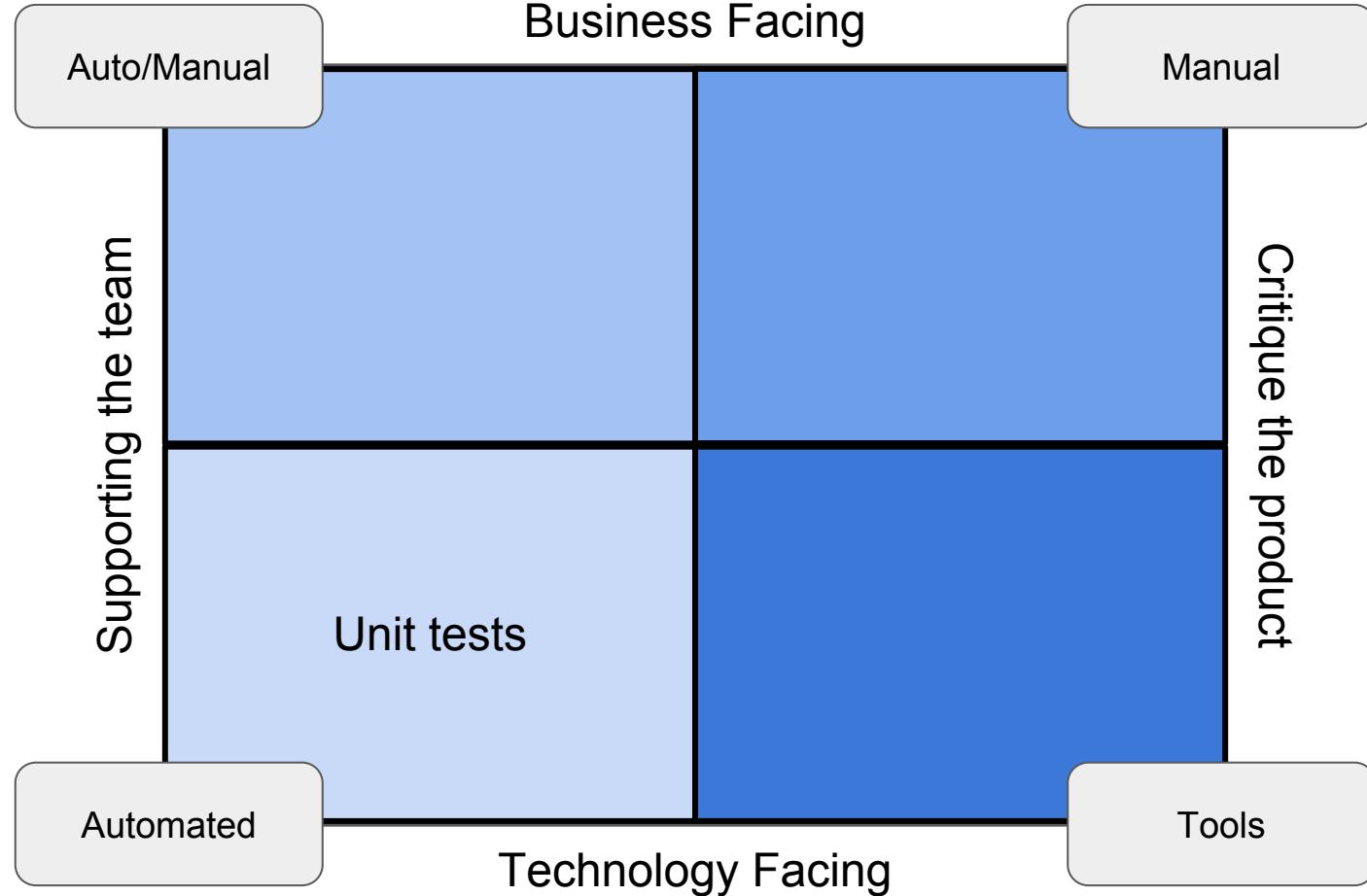
Black Box testing

Tests based on external interfaces

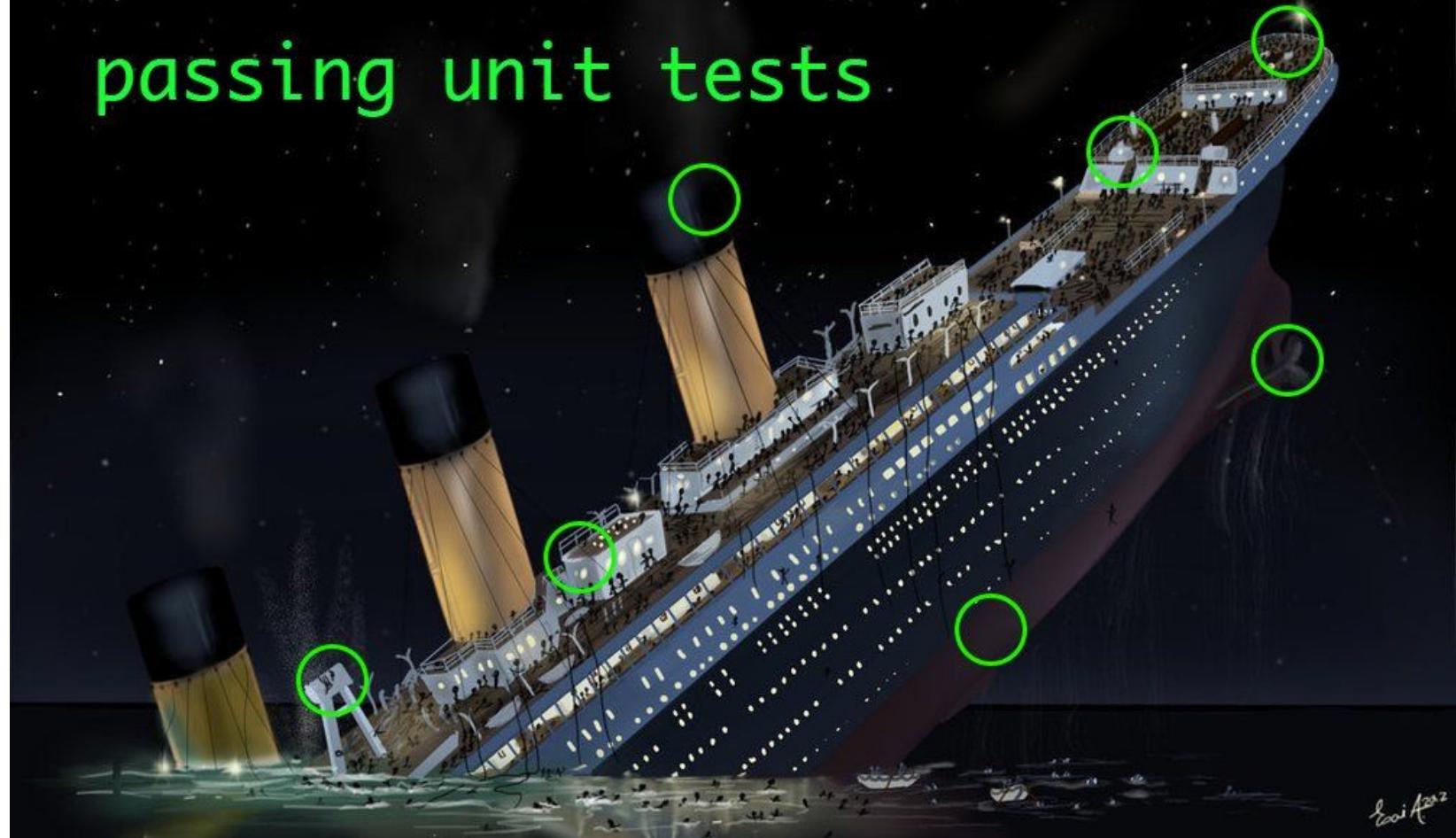
No knowledge of internals

Intuit error paths (test data selection)

Generally done by testers



passing unit tests



Eric A. 2022

Let's test... the Fibonacci sequence

$$F_n = F_{n-1} + F_{n-2},$$

$$F_0 = 0, \quad F_1 = 1.$$

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, ...

Let's test... the Fibonacci sequence

```
import unittest

# Returns the nth Fibonacci number

def fib(n):
    return;
```

Let's test... the Fibonacci sequence

```
class TestFibonacci(unittest.TestCase):  
  
    def test_fibonacci(self):  
        self.assertEqual(fibonacci(0), 0)  
        self.assertEqual(fibonacci(1), 1)  
        self.assertEqual(fibonacci(2), 1)  
        self.assertEqual(fibonacci(3), 2)  
        self.assertEqual(fibonacci(4), 3)  
        self.assertEqual(fibonacci(5), 5)  
        self.assertEqual(fibonacci(6), 8)  
        self.assertEqual(fibonacci(7), 13)  
        self.assertEqual(fibonacci(8), 21)  
        self.assertEqual(fibonacci(9), 34)  
        self.assertEqual(fibonacci(10), 55)  
  
if __name__ == '__main__':  
    unittest.main()
```

Let's test... the Fibonacci sequence

```
class TestFibonacci(unittest.TestCase):  
  
    def test_fib_1(self):  
        self.assertEqual(fib(1), 1)  
  
if __name__ == '__main__':  
    unittest.main()
```

Let's test... the Fibonacci sequence

```
$ python main.py
```

```
FFFFF
```

```
=====
FAIL: test_fib_0 (__main__.TestFibonacci)
```

```
-----
Traceback (most recent call last):
```

```
  File "main.py", line 15, in test_fib_0
```

```
    self.assertEqual(fib(0), 0)
```

```
AssertionError: None != 0
```

Let's test... the Fibonacci sequence

```
def fib(n):  
  
    if n == 0: return 0  
  
    elif n == 1: return 1  
  
    else: return fib(n-1)+fib(n-2)
```

Let's test... the Fibonacci sequence

```
$ python main.py
```

```
.....
```

```
Ran 5 tests in 0.000s
```

```
OK
```

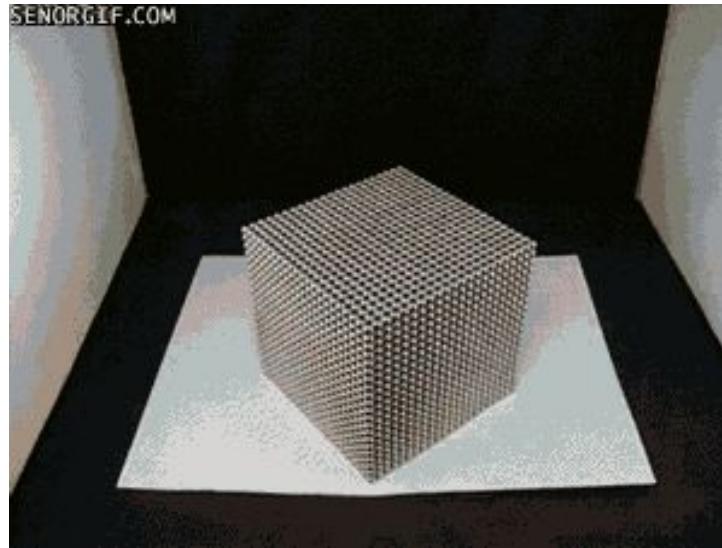
Let's test... the Fibonacci sequence

```
def fib(n):  
    a,b = 0,1  
  
    for i in range(n):  
        a,b = b,a+b  
  
    return a
```

Refactoring

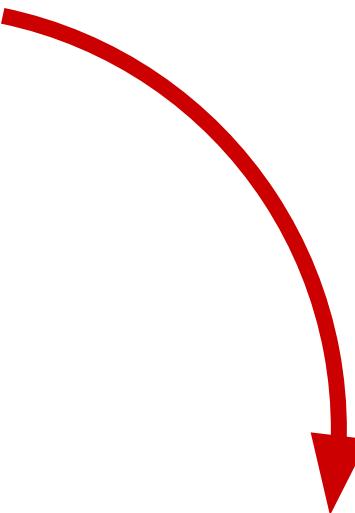


Refactoring



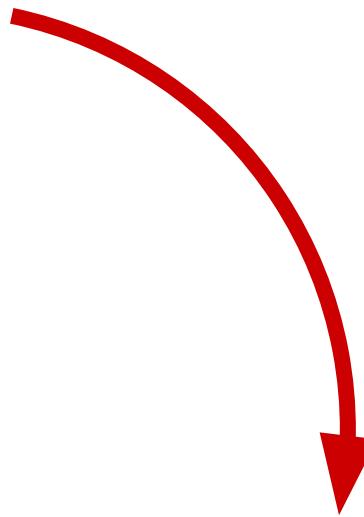
Red

Red



Green

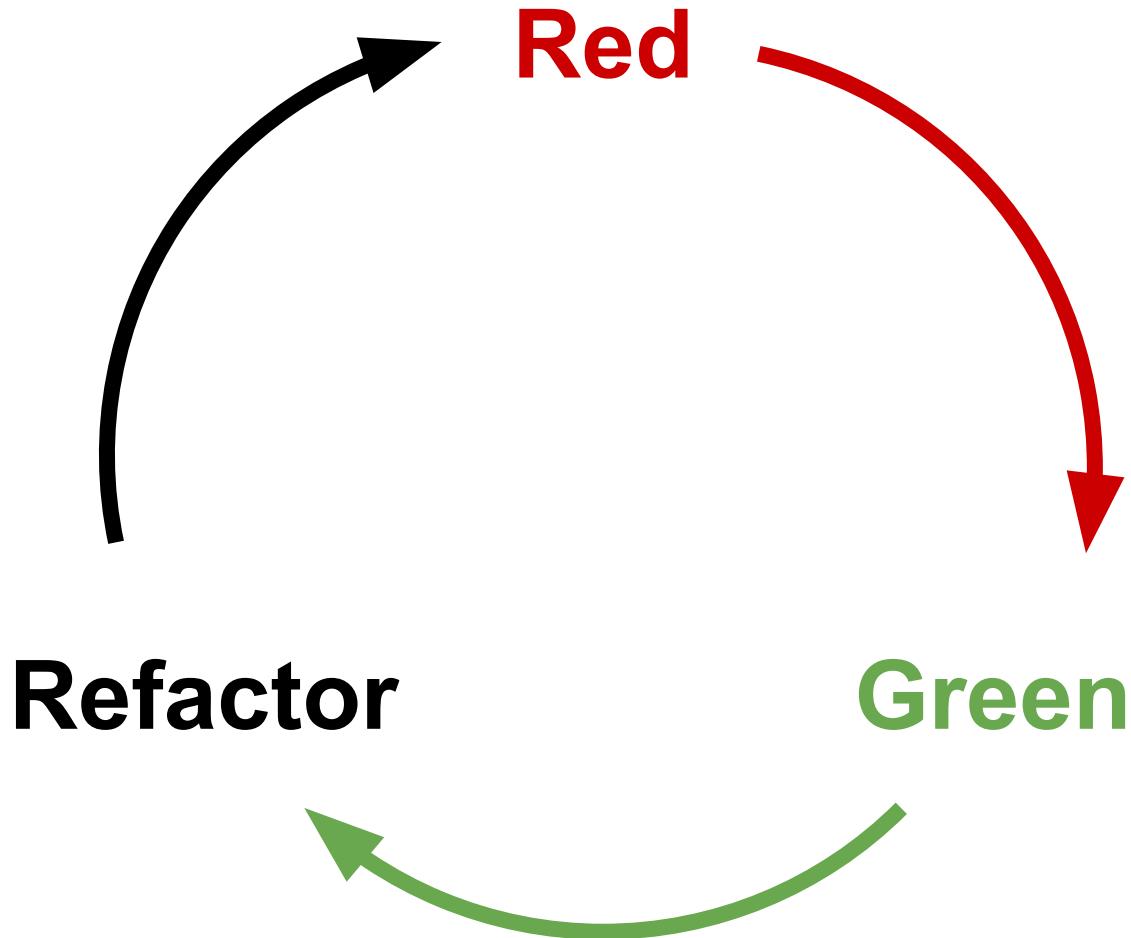
Red



Refactor

Green



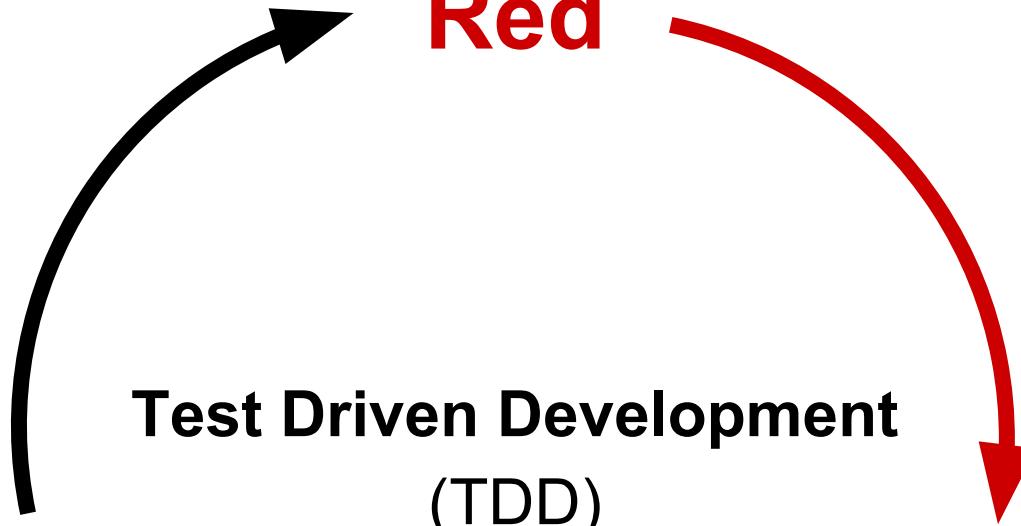


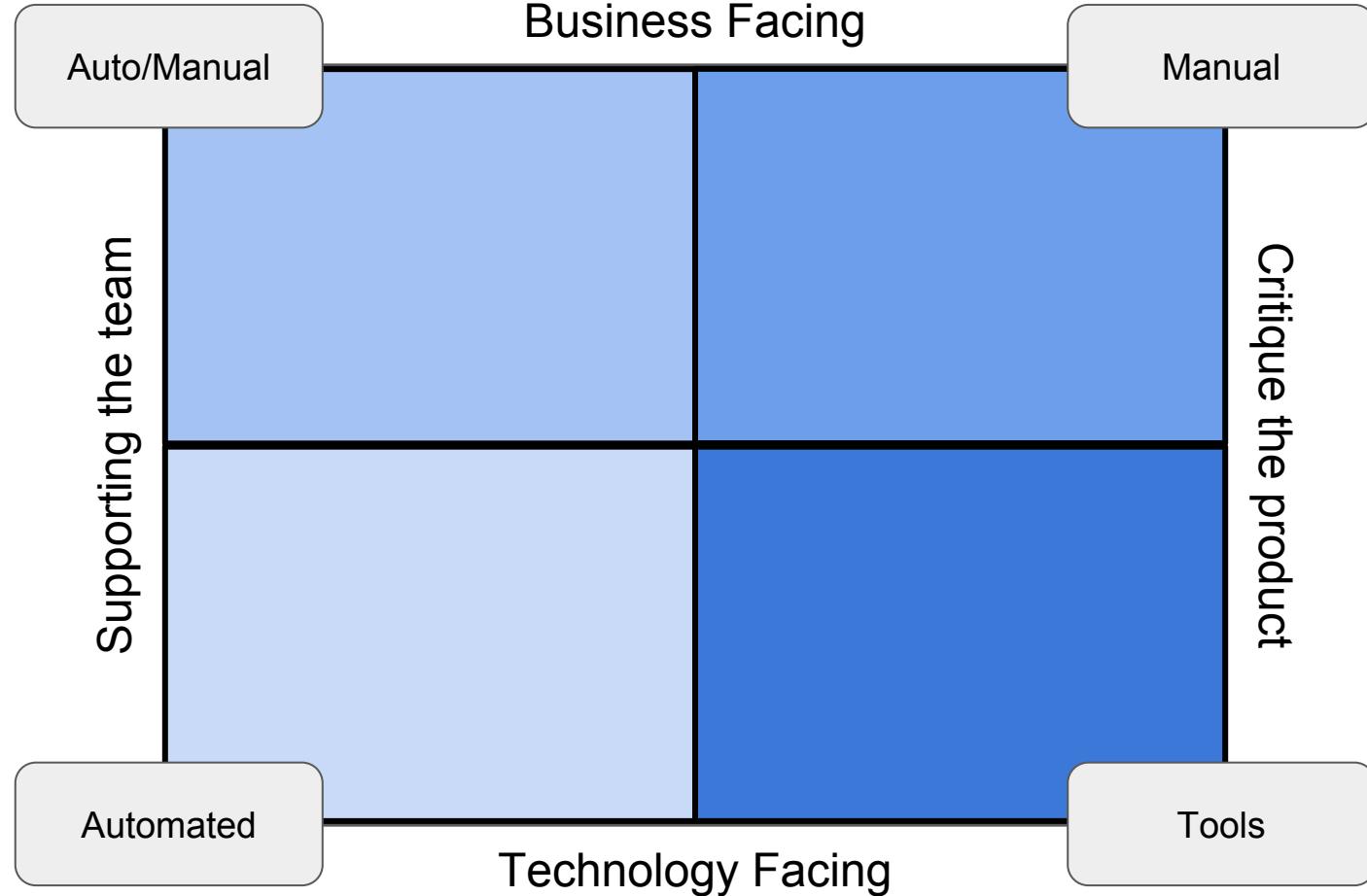
Red

**Test Driven Development
(TDD)**

Green

Refactor





Dan North
@tastapod



Where do we start testing?

Where do we start testing?

What do we test?

Where do we start testing?

What do we test?
(and what don't we test?)

Where do we start testing?

What do we test?
(and what don't we test?)

How much do we test in one go?

Where do we start testing?

What do we test?
(and what don't we test?)

How much do we test in one go?

How do we name our tests?

Where do we start testing?

What do we test?
(and what don't we test?)

How do we understand why a test fails?

How much do we test in one go?

How do we name our tests?

Behaviour Driven Development (BDD)

As a [user]

I want [feature]

so that [benefit]

As a customer

**I want to get coffee from the machine
so that I don't have to make my own**

Given [some initial context]

When [an event occurs]

Then [ensure some outcomes]

Given the coffee machine is installed

And the coffee machine has coffee

And I have deposited €1

When I press the coffee button

Then I should be served a coffee

Given the coffee machine is installed

And the coffee machine has no coffee

And I have deposited €1

When I press the coffee button

Then I should be shown an error

And I should be have my €1 returned

Feature: Some terse yet descriptive text of what is desired

Textual description of the business value of this feature

Business rules that govern the scope of the feature

Any additional information that will make the feature easier to understand

Scenario: Some determinable business situation

Given some precondition

And some other precondition

When some action by the actor

And some other action

And yet another action

Then some testable outcome is achieved

And something else we can check happens too

Scenario: A different situation

...

Feature: Some terse yet descriptive text of what is desired

Textual description of the business value of this feature

Business rules that govern the scope of the feature

Any additional information that will make the feature easier to understand

Scenario: Some determinable business situation

Given some precondition

And some other precondition

When some action by the actor

And some other action

And yet another action

Then some testable outcome is achieved

And something else we can check happens too

Scenario: A different situation

...



Feature: Some terse yet descriptive text of what is desired

Textual description of the business value of this feature

Business rules that govern the scope of the feature

Any additional information that will make the feature easier to understand

Scenario: Some determinable business situation

Given some precondition

And some other precondition

When some action by the actor

And some other action

And yet another action

Then some testable outcome is achieved

And something else we can check happens too

Scenario: A different situation



BDD tools



freshen

lettuce

behave

Renamed the whole thing 'Cucumber'. Patty decided on the bus.

⌚ master ⚒ v3.0.0.pre.1 ... 0.3.11.200906161540



aslakhellesoy committed on Apr 11, 2008

1 parent 7853cc9

Given

Put the system in a known state

Given the coffee machine is installed

Put the system in a known state

Given

Given the coffee machine is installed



```
@given('the coffee machine is installed')  
  
def step_impl(context):  
    context.coffee_machine = CoffeeMachine()
```

When I press the coffee button

When

Describe the key action

When

Describe the key action

When I press the coffee button



```
@when('I press the coffee button')  
  
def step_impl(context):  
  
    context.coffee_machine.press_coffee_button()
```

Then I should be served a coffee

Then

Observe outcomes

Then
Observe outcomes

Then I should be served a coffee



```
@then('I should be served a coffee')

def step_impl(context):
    assert context.coffee_machine.served_drink is "coffee"
```

```
$ behave
```

```
Feature: A coffee machine that dispenses coffee # coffee_machine.feature:1
```

```
Scenario: run a simple test          # coffee_machine.feature:3
```

```
  Given the coffee machine is installed # steps/steps.py:4 0.000s
```

```
  And the coffee machine has coffee    # steps/steps.py:8 0.000s
```

```
  And I have deposited 1 euro        # steps/steps.py:12 0.000s
```

```
  When I press the coffee button     # steps/steps.py:16 0.000s
```

```
  Then I should be served a coffee   # steps/steps.py:20 0.000s
```

```
1 feature passed, 0 failed, 0 skipped
```

```
1 scenario passed, 0 failed, 0 skipped
```

```
5 steps passed, 0 failed, 0 skipped, 0 undefined
```

```
Took 0m0.000s
```

behave

`pip install behave`

Features go in a `.feature` file

Steps (Givens, Whens and Thens) go in `steps/`

Exercise

Should you use BDD?

Should you use BDD?

More tooling, organisational requirements

Should you use BDD?

More tooling, organisational requirements

Might not be applicable in all cases

Should you use BDD?

More tooling, organisational requirements

Might not be applicable in all cases

Pick and choose!

```
describe('Array', function() {  
  
  describe('#indexOf()', function() {  
  
    it('should return -1 when value is not present', function() {  
  
      assert.equal(-1, [1,2,3].indexOf(4));  
  
    });  
  
  });  
  
});
```

Should you use BDD?

More tooling, organisational requirements

```
$ mocha
```

Might not be applicable in all cases

```
Array
```

Pick and choose!

```
#indexOf()
```

```
✓ should return -1 when value is not present
```

```
1 passing (9ms)
```

Should you use BDD?

More tooling, organisational requirements

```
$ mocha
```

Might not be applicable in all cases

```
Array
```

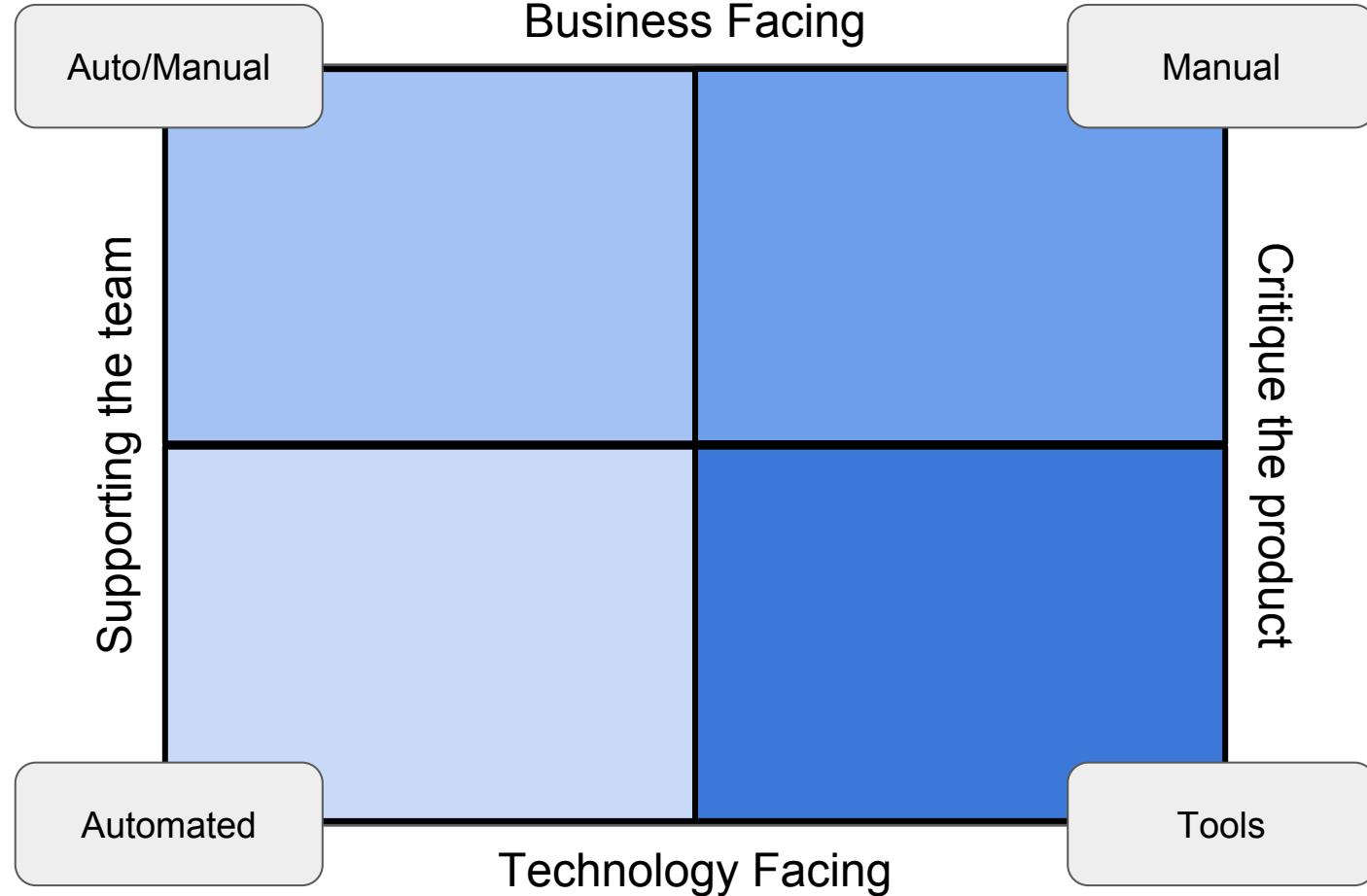
Pick and choose!

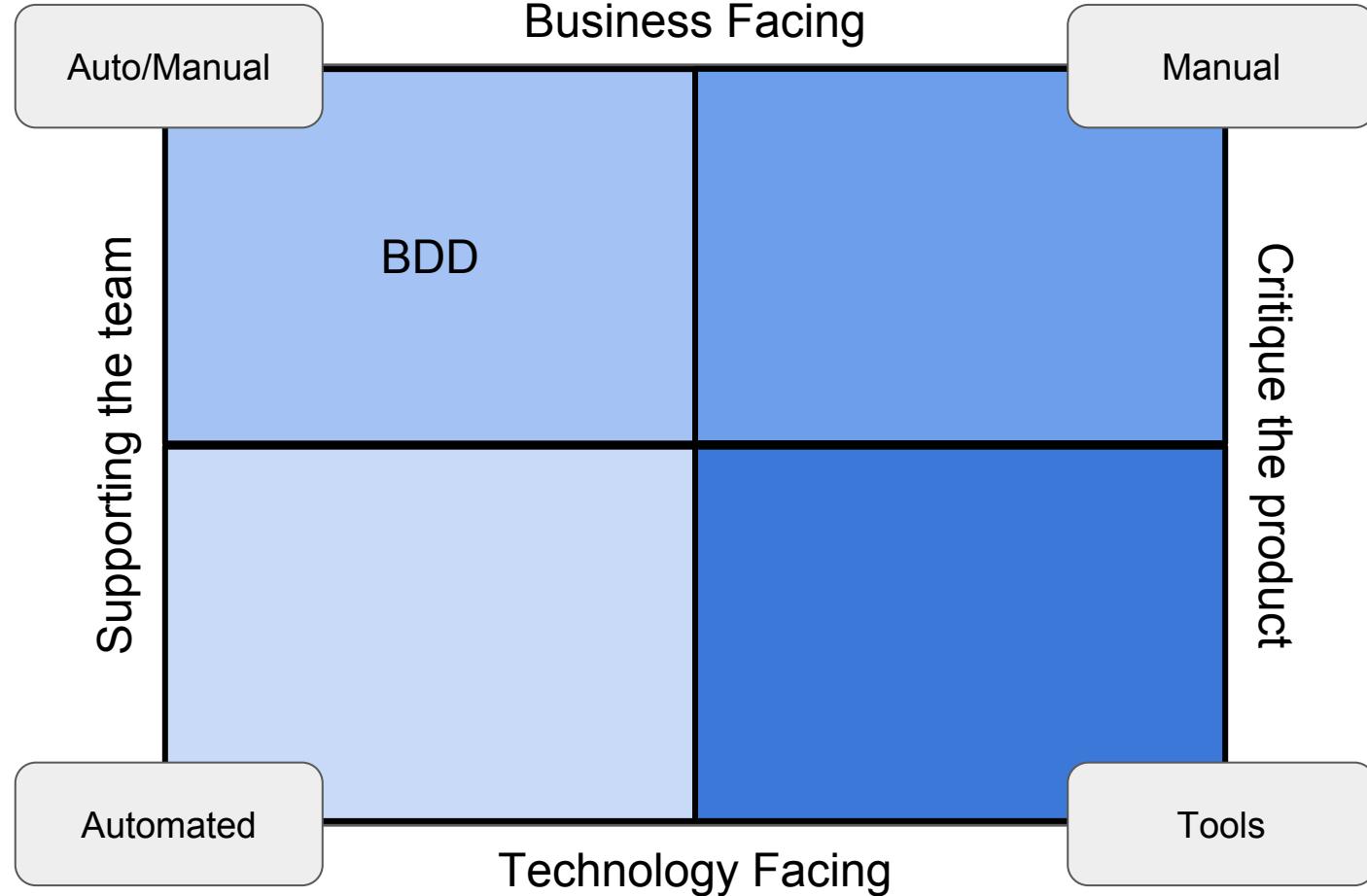
```
#indexOf()
```

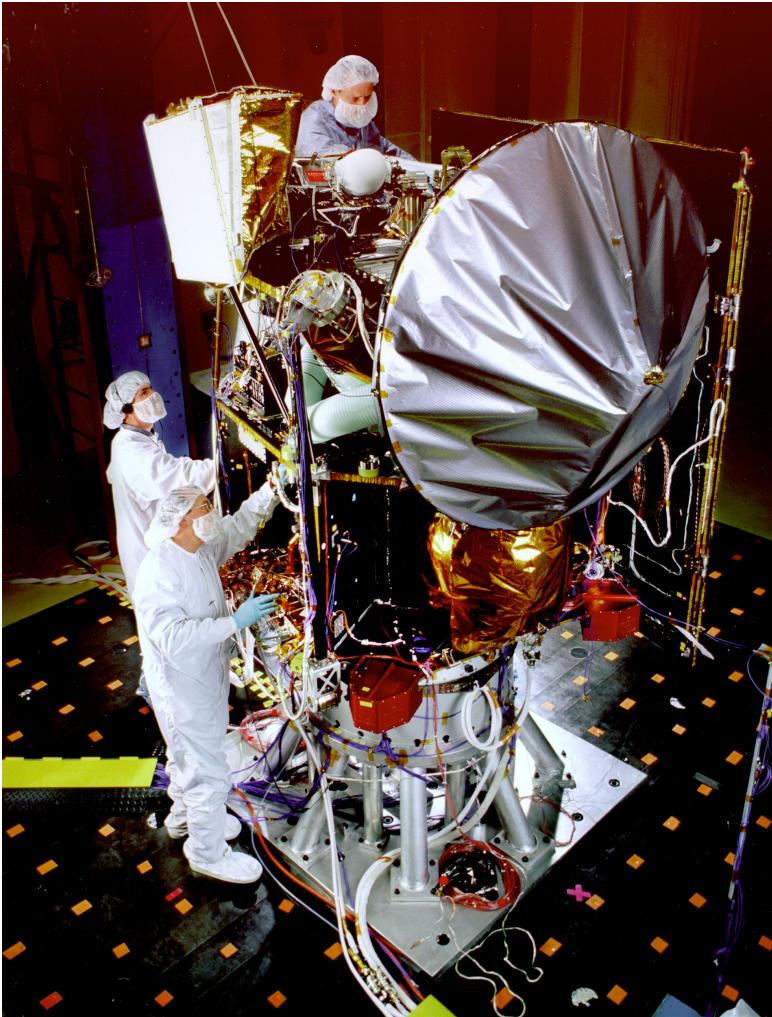
Collaborate!

```
✓ should return -1 when value is not present
```

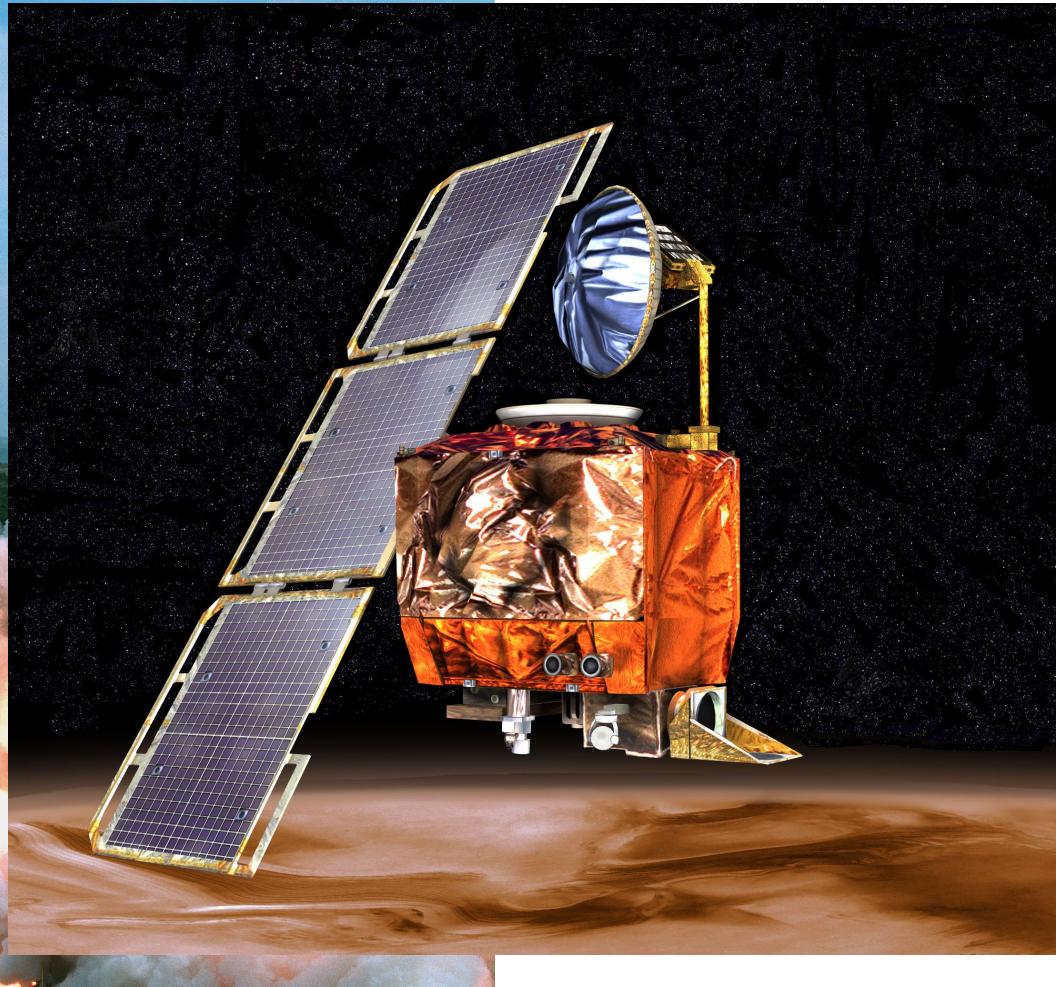
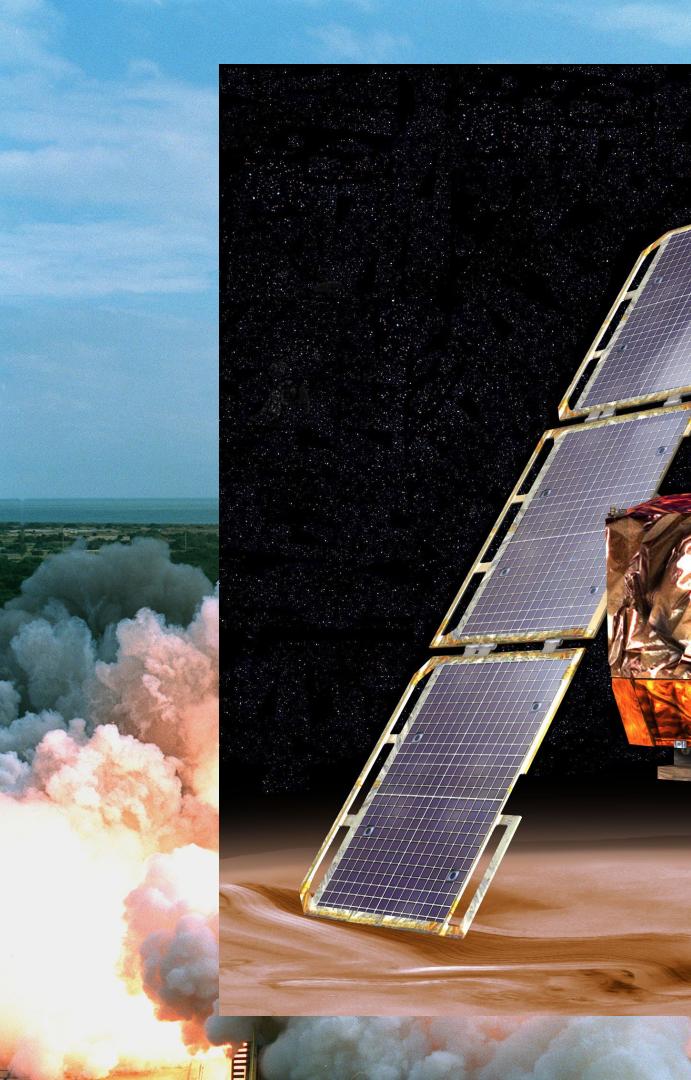
```
1 passing (9ms)
```

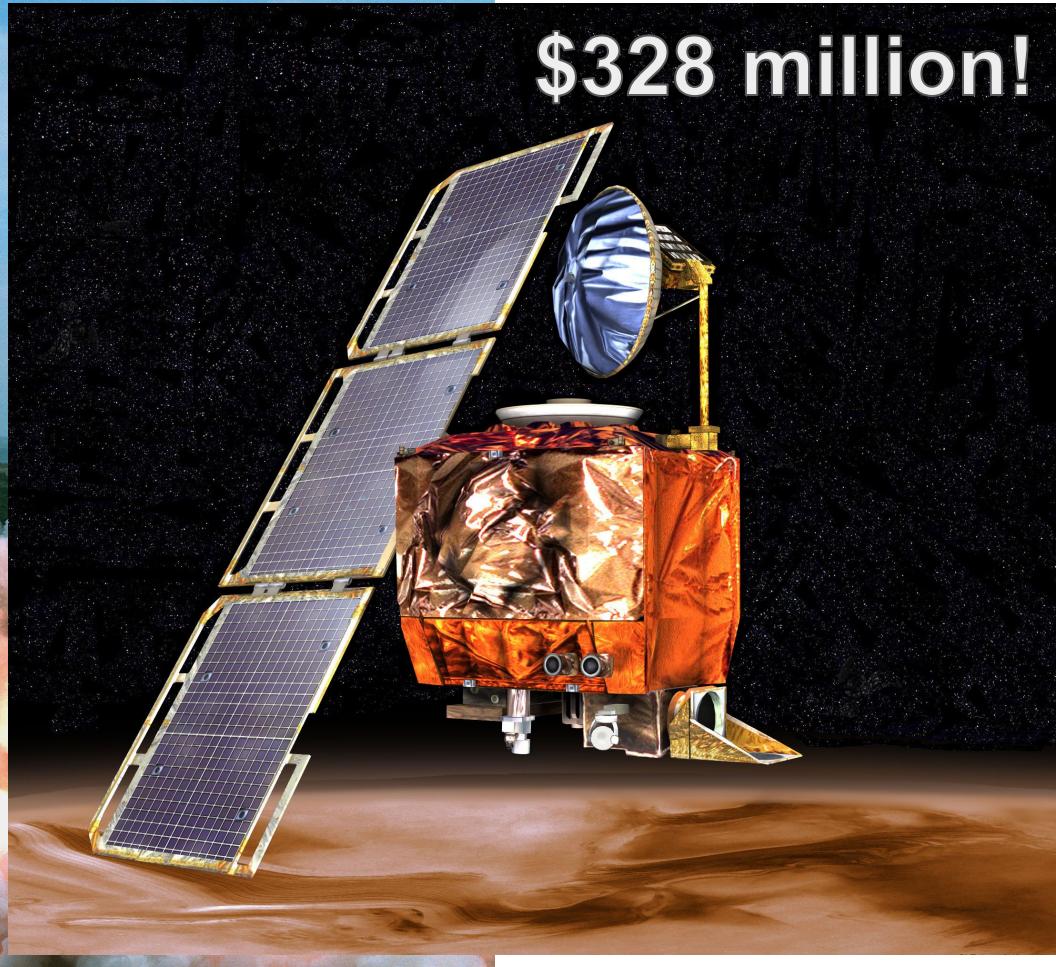
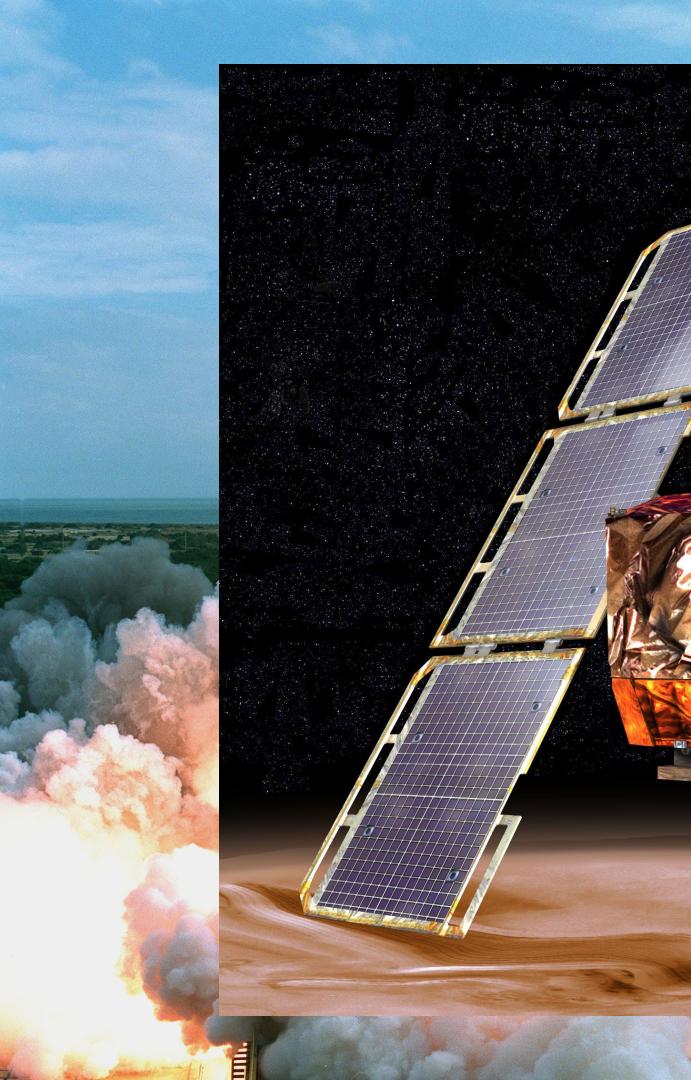












\$328 million!

Integration testing



2 UNIT TESTS, 0 INTEGRATION TESTS

via reddit.com/r/programmerhumor



Integration testing

Testing more than individual components

Integration testing

Testing more than individual components

Multiple approaches

- Big Bang
- Top Down
- Bottom Up

Integration testing

Testing more than individual components

Multiple approaches

- Big Bang
- Top Down
- Bottom Up

May use fixtures (stubs and mocks)

Test fixtures - Stubs

‘Stub’ out functionality

Reply to calls with canned responses

Allows testing without dependencies

Test fixtures - Stubs

‘Stub’ out functionality

Reply to calls with canned responses

Allows testing without dependencies

`pip install mock` (Python 2)

```
from unittest.mock import MagicMock

dependency.method = MagicMock(return_value=5)

...
dependency.method(10)

>>> 5
```

Test fixtures - Mocks

Basically stubs with expectations

```
from unittest.mock import MagicMock

dependency.method = MagicMock(return_value=5)

...
dependency.method(10)

...
dependency.method.assert_called_with(10)
```

Let's test ... Conway's Game of Life

Cellular automaton

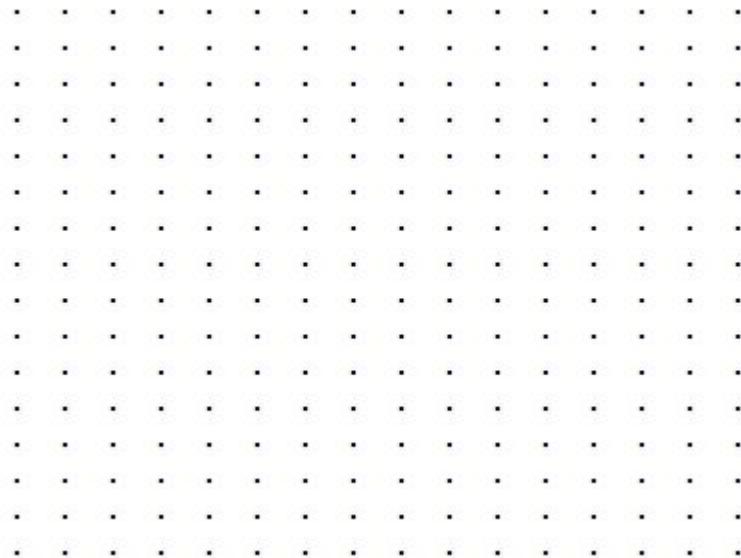
Set initial state and watch the system evolve

Complicated behavior from simple rules



Let's test ... Conway's Game of Life

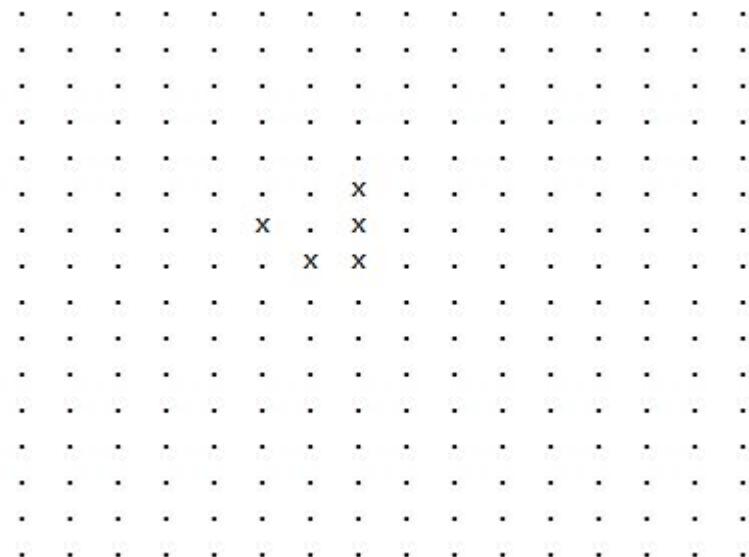
Grid of cells



Let's test ... Conway's Game of Life

Grid of cells

Each cell can be on (**alive**) or off (**dead**)



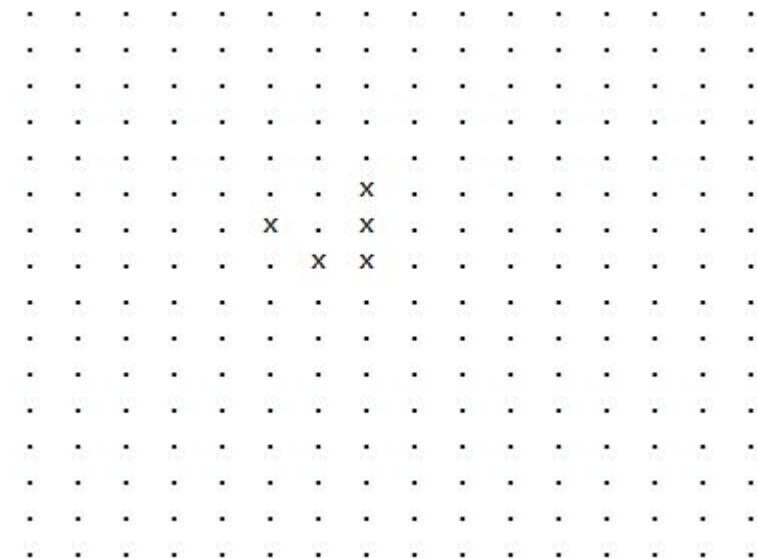
Let's test ... Conway's Game of Life

Grid of cells

Each cell can be on (**alive**) or off (**dead**)

Each 'step', apply rules:

- Live cells with < 2 neighbours **die** (under population)
- Live cells with 2 - 3 neighbours **live** (stable)
- Live cells with > 2 neighbours **die** (over population)
- Dead cells with 3 neighbours **live** (reproduction)



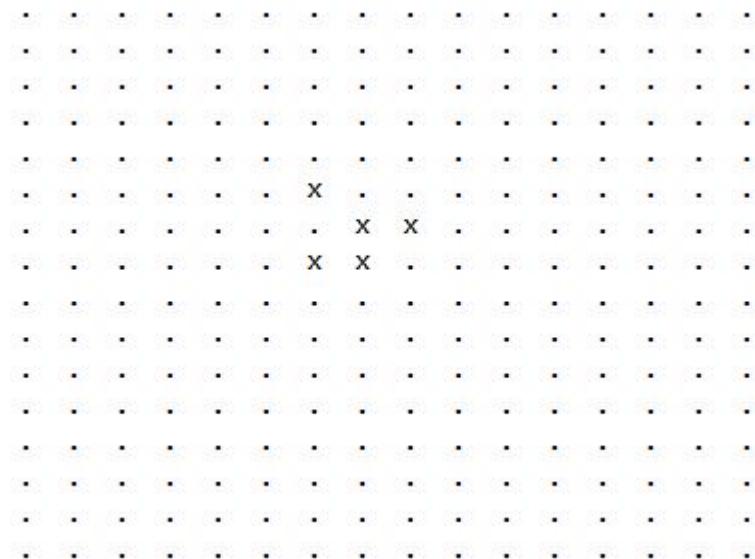
Let's test ... Conway's Game of Life

Grid of cells

Each cell can be on (**alive**) or off (**dead**)

Each 'step', apply rules:

- Live cells with < 2 neighbours **die** (under population)
- Live cells with 2 - 3 neighbours **live** (stable)
- Live cells with > 2 neighbours **die** (over population)
- Dead cells with 3 neighbours **live** (reproduction)



`py.test`

`pip install pytest`

Similar to unittest

Less boilerplate

Supports running unittest tests

py.test

```
$ pytest -v test_game_of_life.py
```

```
===== test session starts =====
```

```
platform linux2 -- Python 2.7.12, pytest-3.0.6, py-1.4.32, pluggy-0.4.0 --
```

```
/home/steve/Dev/dsr/dsr-testing-lab/exercise-pytest/venv/bin/python
```

```
cachadir: .cache
```

```
rootdir: /home/steve/Dev/dsr/dsr-testing-lab/exercise-pytest, infile:
```

```
collected 1 items
```

```
test_game_of_life.py::test_advance_should_return_a_set PASSED
```

```
===== 1 passed in 0.00 seconds =====
```

Exercise







Figure 3: Correctly Calculated Range Gate

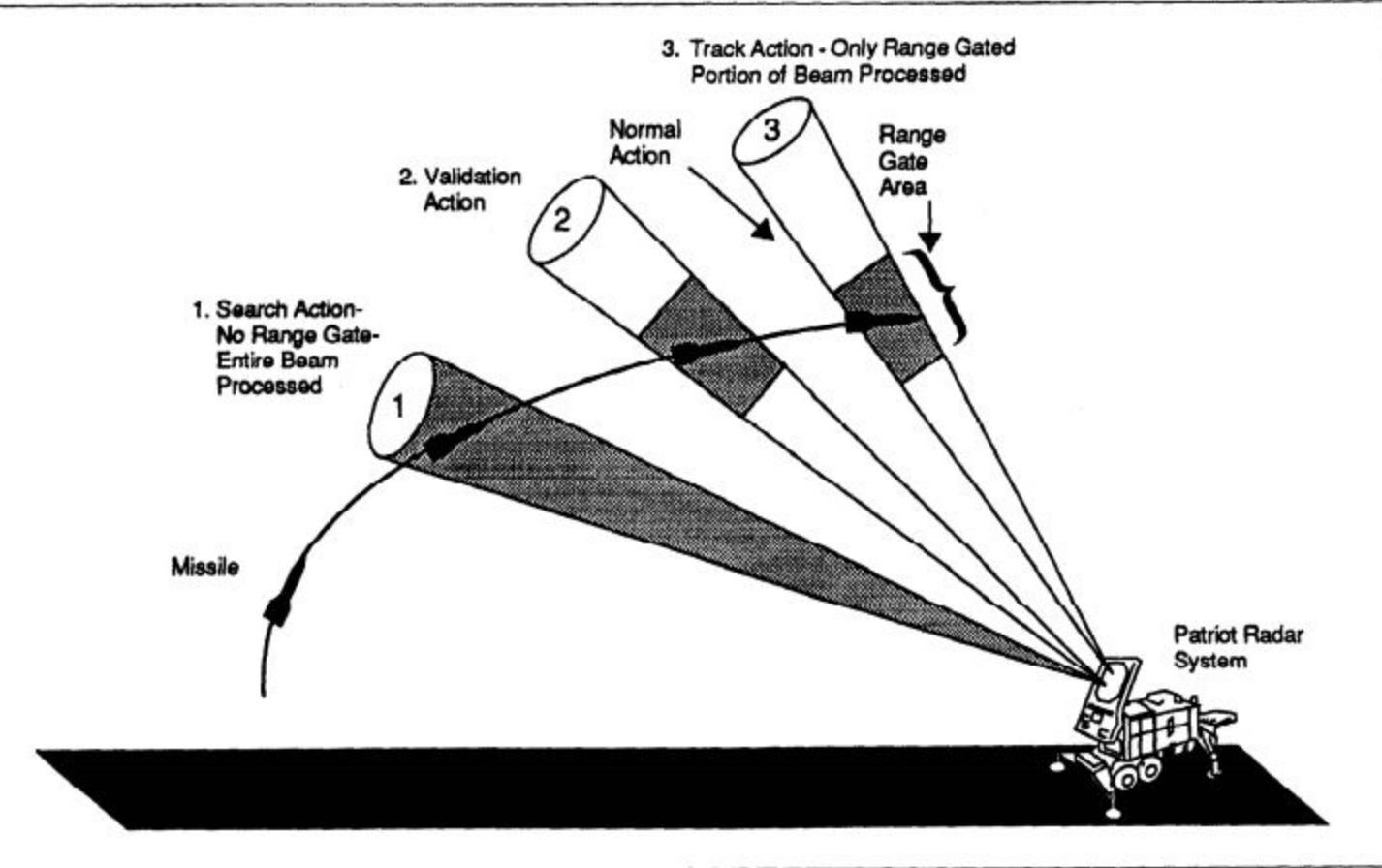


Figure 4: Calculated Range Gate After Approximately 8 Hours

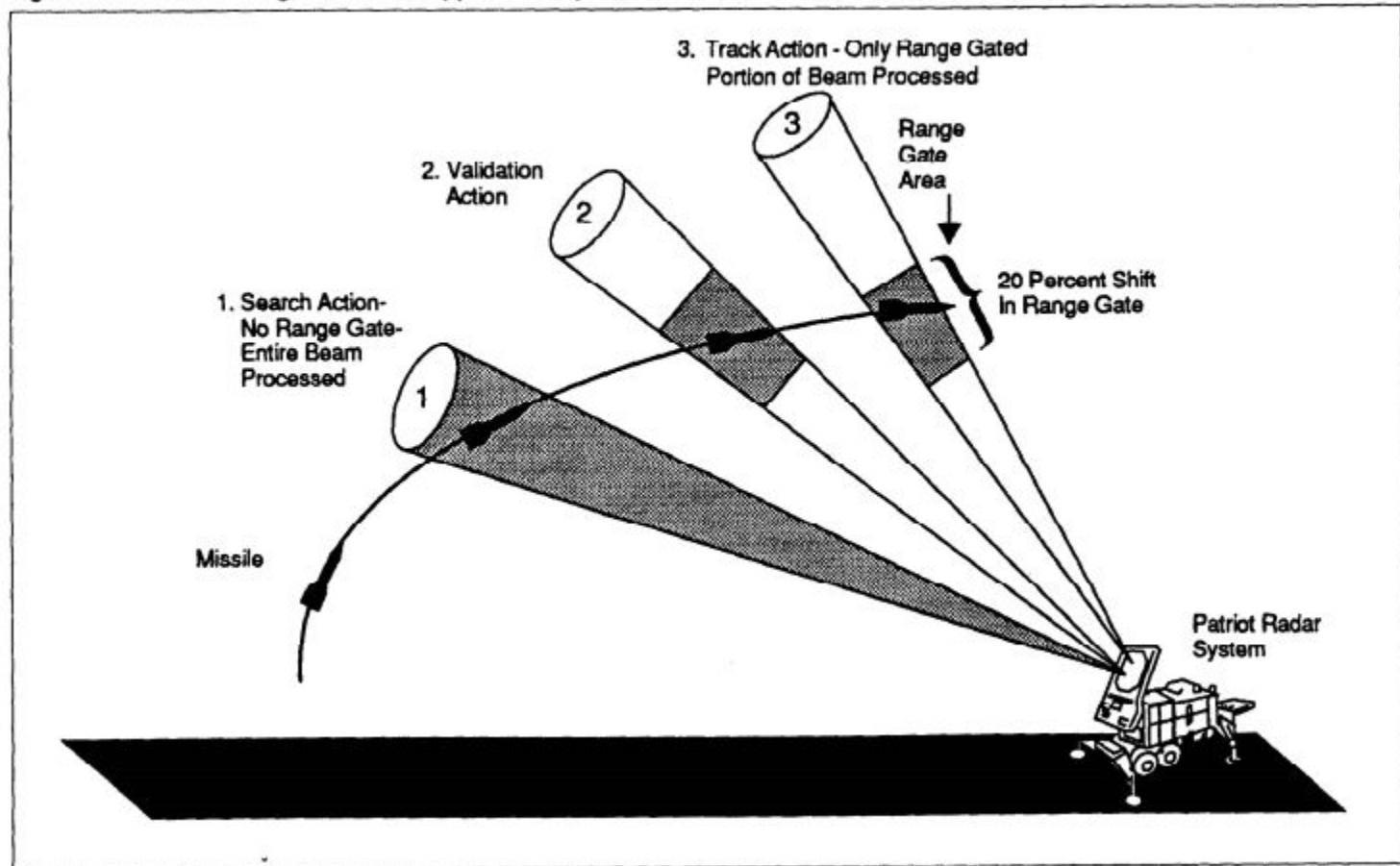
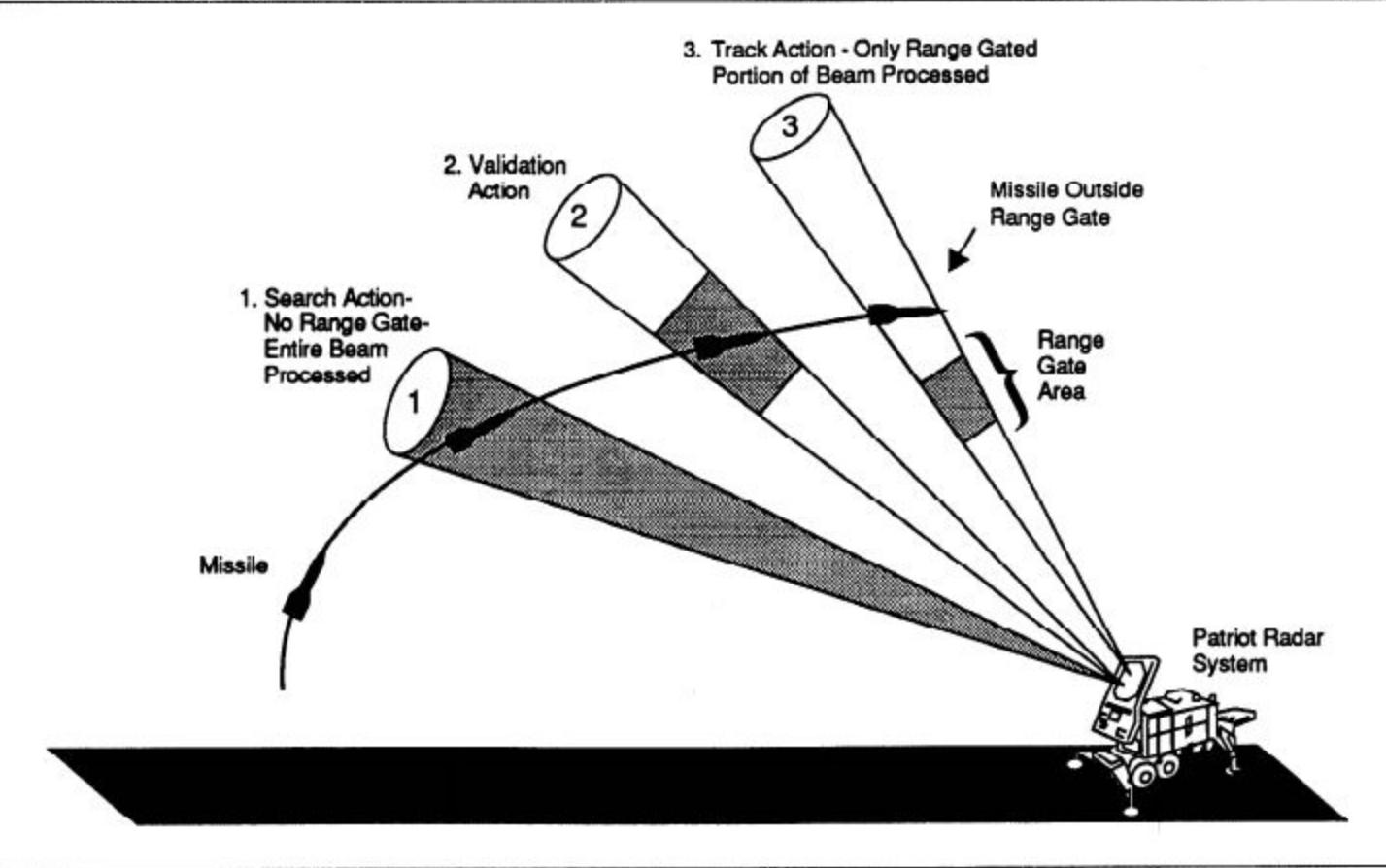


Figure 5: Incorrectly Calculated Range Gate



Systems testing

Systems testing

Test the system **as a whole**

Step up from integration testing

Aim to test close to the real world

Many sub types

Generally done by dedicated testers

Performance + Load testing

Performance + Load testing

Performance: test performance of systems (duh)

Load: test behaviour under load

Also: stress, volume, scalability, recovery testing

Performance + Load testing

Performance: test performance of systems (duh)

Load: test behaviour under load

Also: stress, volume, scalability, recovery testing

Relies on tooling





ARTILLERY.IO

```
config:
  target: 'https://artillery.io'
phases:
  - duration: 60
    arrivalRate: 20
defaults:
  headers:
    x-my-service-auth: '987401838271002188298567'
scenarios:
  - flow:
    - get:
      url: "/docs"
```

```
Scenarios launched: 5
Scenarios completed: 5
Requests completed: 58
RPS sent: 0.86
Request latency:
    min: 102.4
    max: 3067.5
    median: 325.5
    p95: 2118.5
    p99: 3020
Scenario duration:
    min: 56745.4
    max: 67339.1
    median: 59275.6
    p95: NaN
    p99: NaN
Codes:
    200: 58
```

config:

target: 'https://artillery.io'

phases:

- duration: 60
- arrivalRate: 20

defaults:

headers:

x-my-service-auth: '987401838271002188298567'

scenarios:

- flow:

- get:
 - url: "/docs"



[Reset Stats](#)

[Statistics](#) [Failures](#) [Exceptions](#)

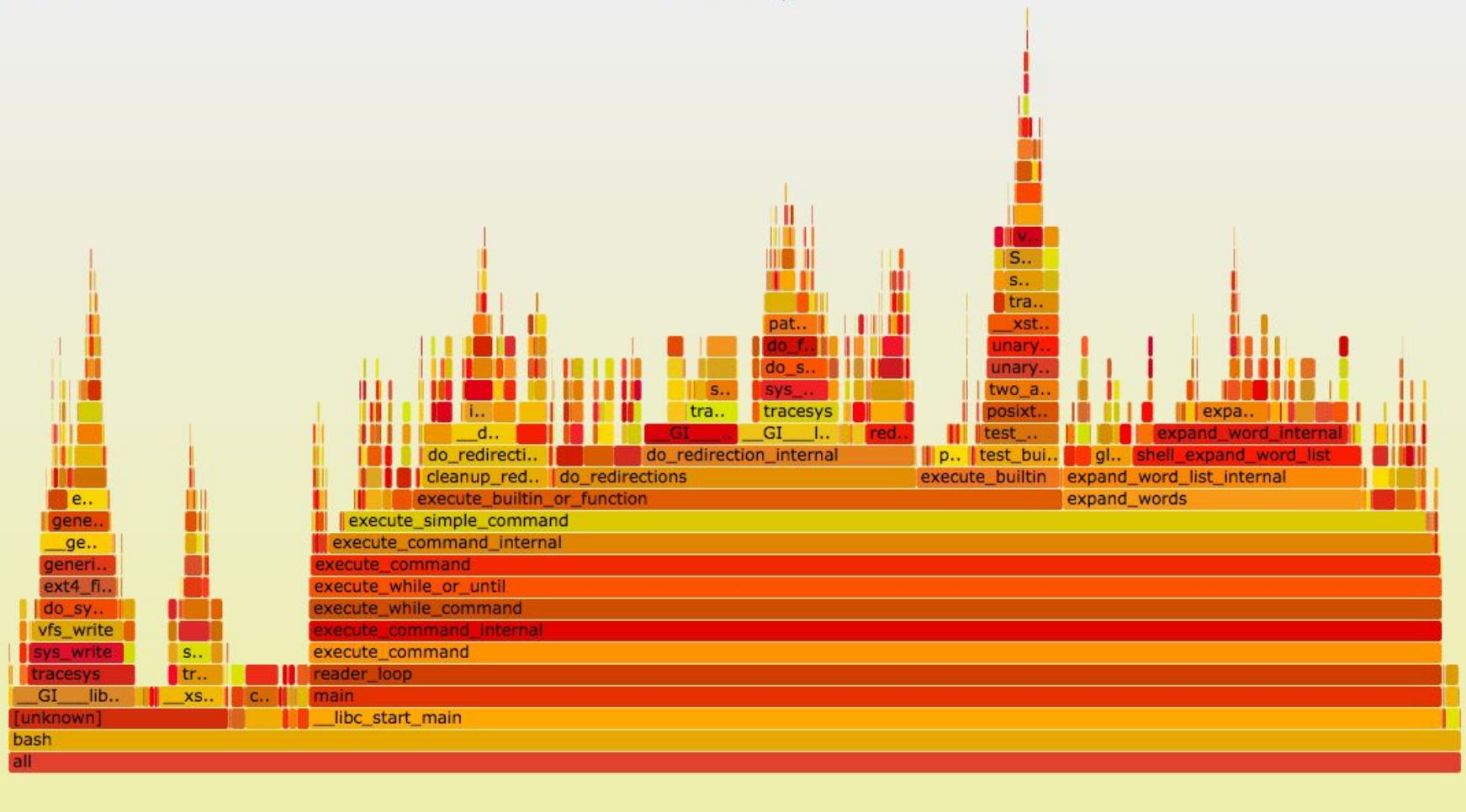
Type	Name	# requests	# fails	Median	Average	Min	Max	Content Size	# reqs/sec
GET	/	1831	0	21	21	4	38	19947	18.3
GET	/blog	608	0	25	26	3	49	19841	6.9
GET	/blog/[post-slug]	612	0	14	15	2	27	19858	7.8
GET	/forum	573	0	26	26	3	49	20209	5.5
GET	/forum/[thread-slug]	596	0	30	30	6	55	20209	5.3
POST	/forum/[thread-slug]	71	0	62	63	13	120	11188	0.6
POST	/forum/new	64	0	59	58	6	108	3272	0.7
GET	/signin	3439	0	26	26	3	49	19850	31.3
Total		7794	0	26	25	2	120	19711	76.4

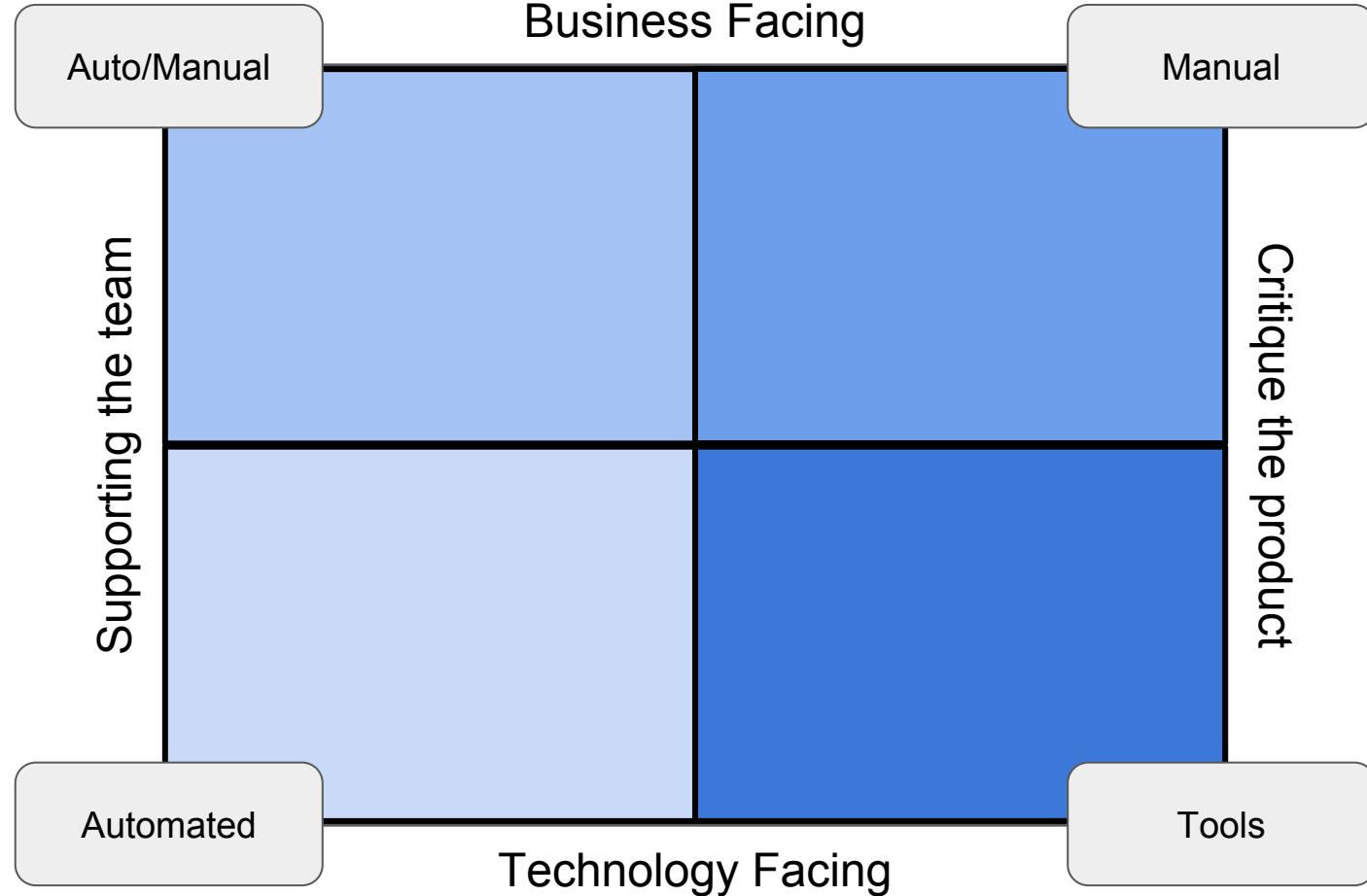
[Download request statistics CSV](#)

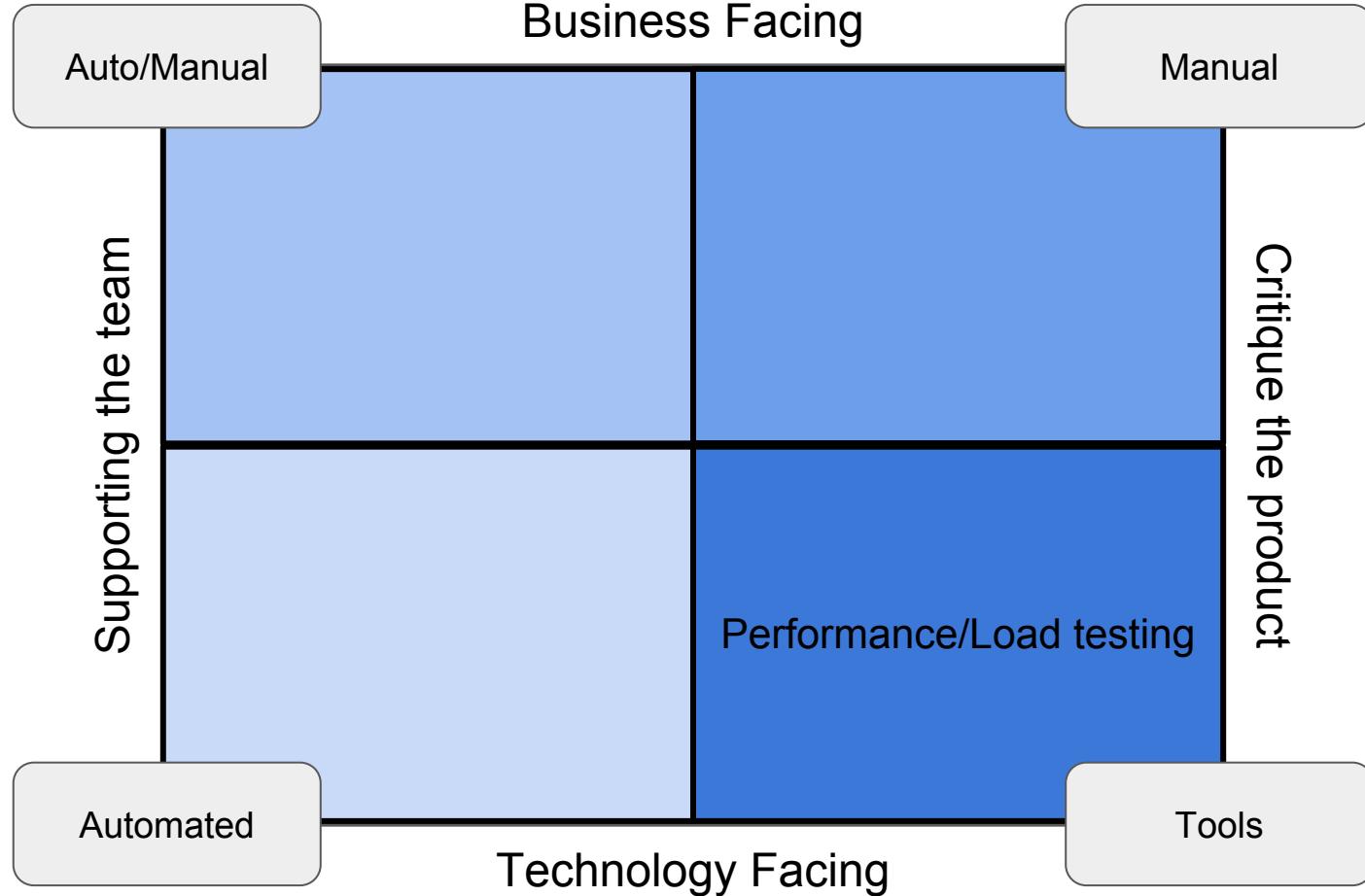
[Download response time distribution CSV](#)

[Reset Zoom](#)

Flame Graph

[Search](#)





Usability testing

Usability testing

Can people use our product?

“Consumability”

Really difficult to do with team members

Accessibility testing

Copyrighted Material

Steve Krug

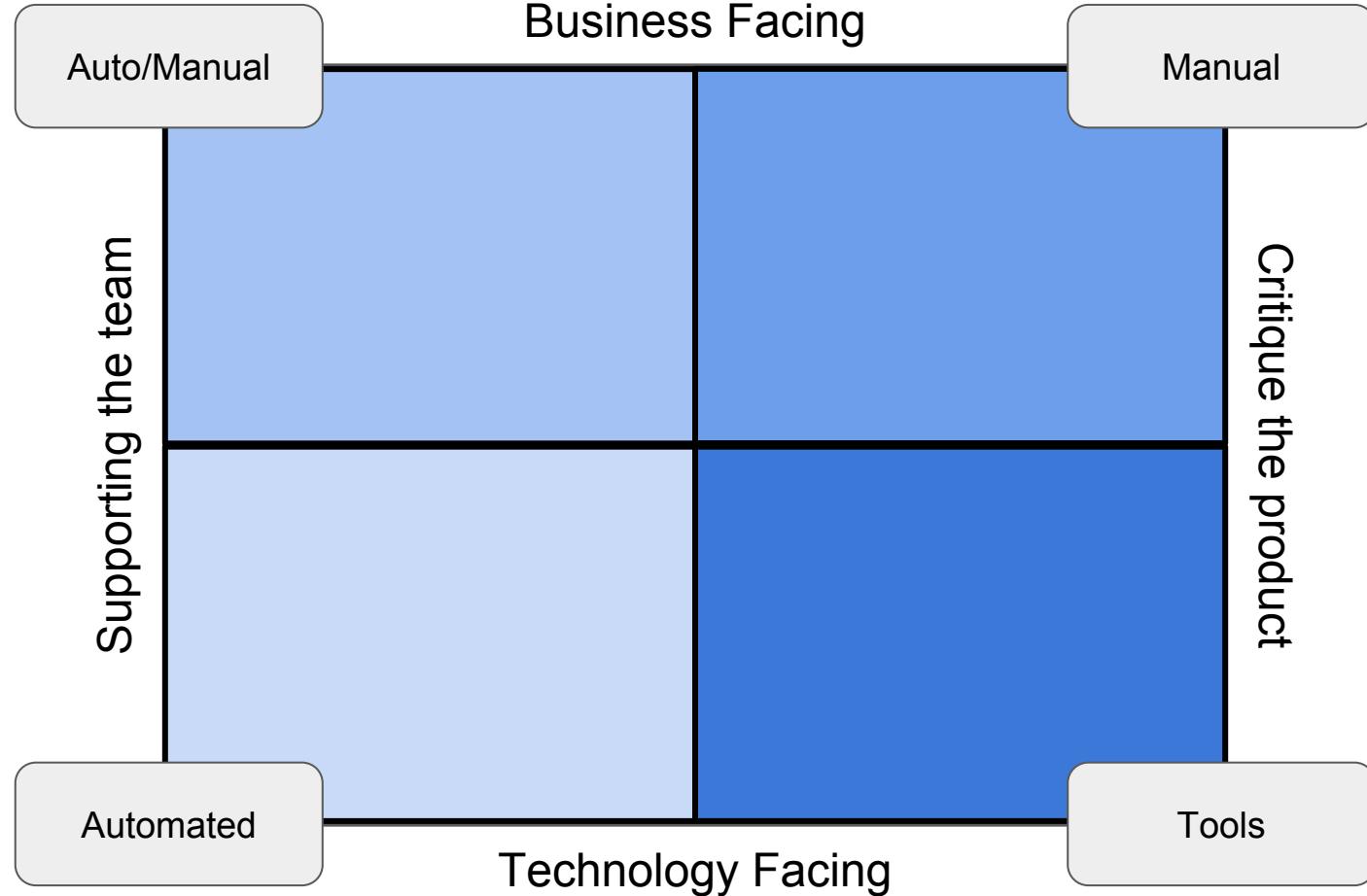


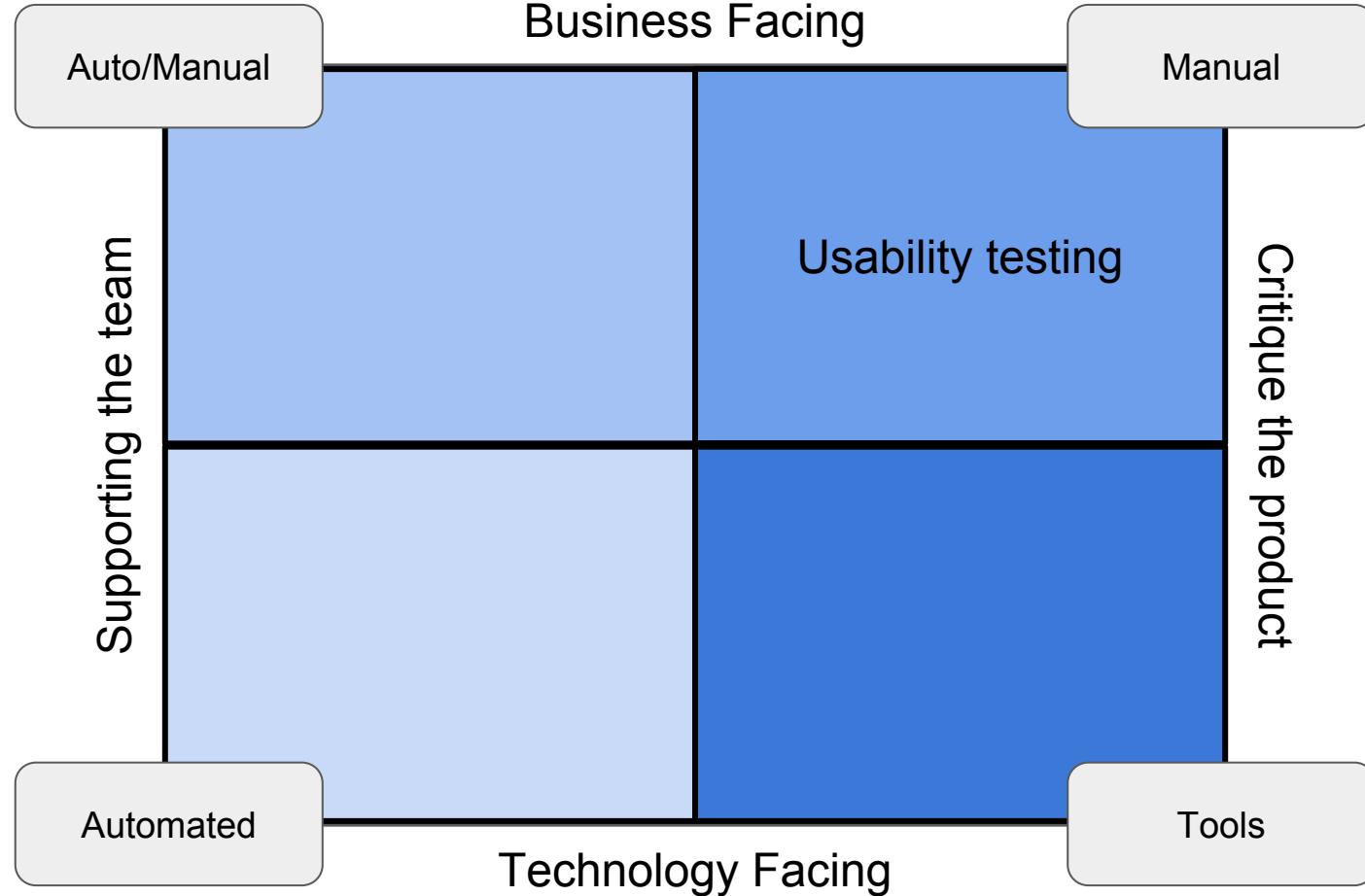
DON'T MAKE ME THINK

Revisited

and Mobile

A Common Sense Approach to Web Usability





(User) Acceptance Testing

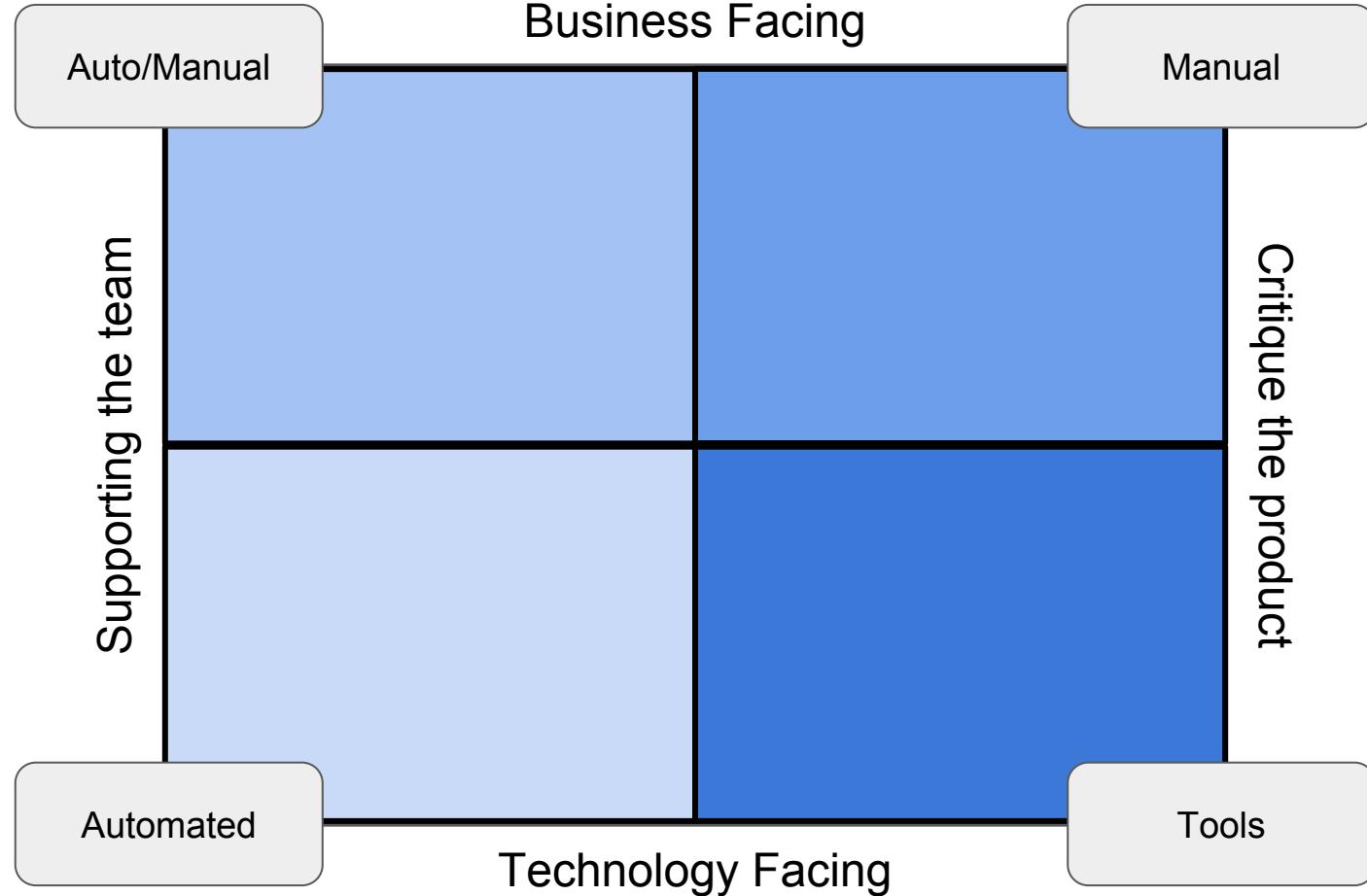
(User) Acceptance Testing

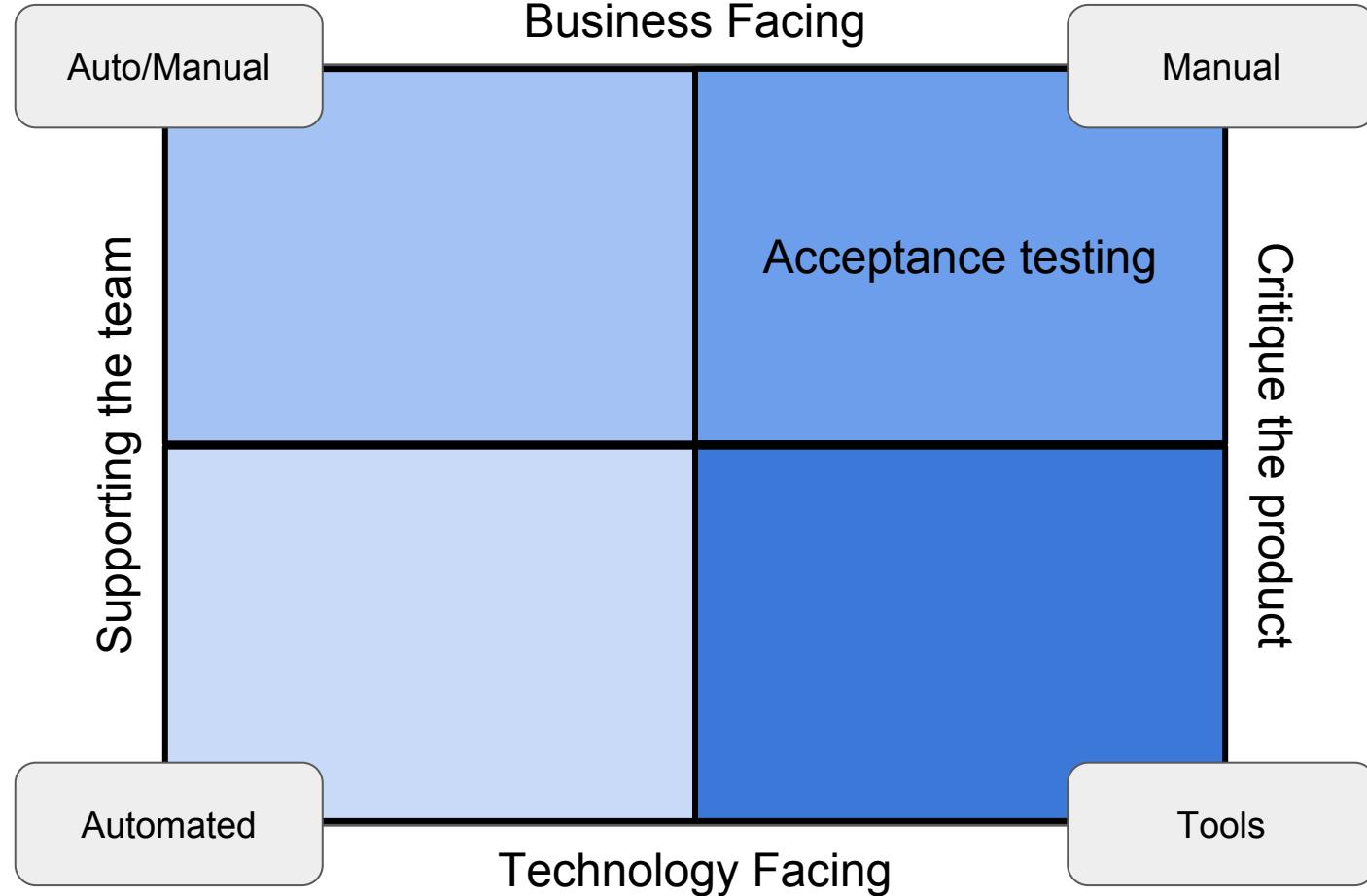
Gates (Waterfall)

Client sign off (UAT)

Operational acceptance

Regulation/contract acceptance





Exploratory testing

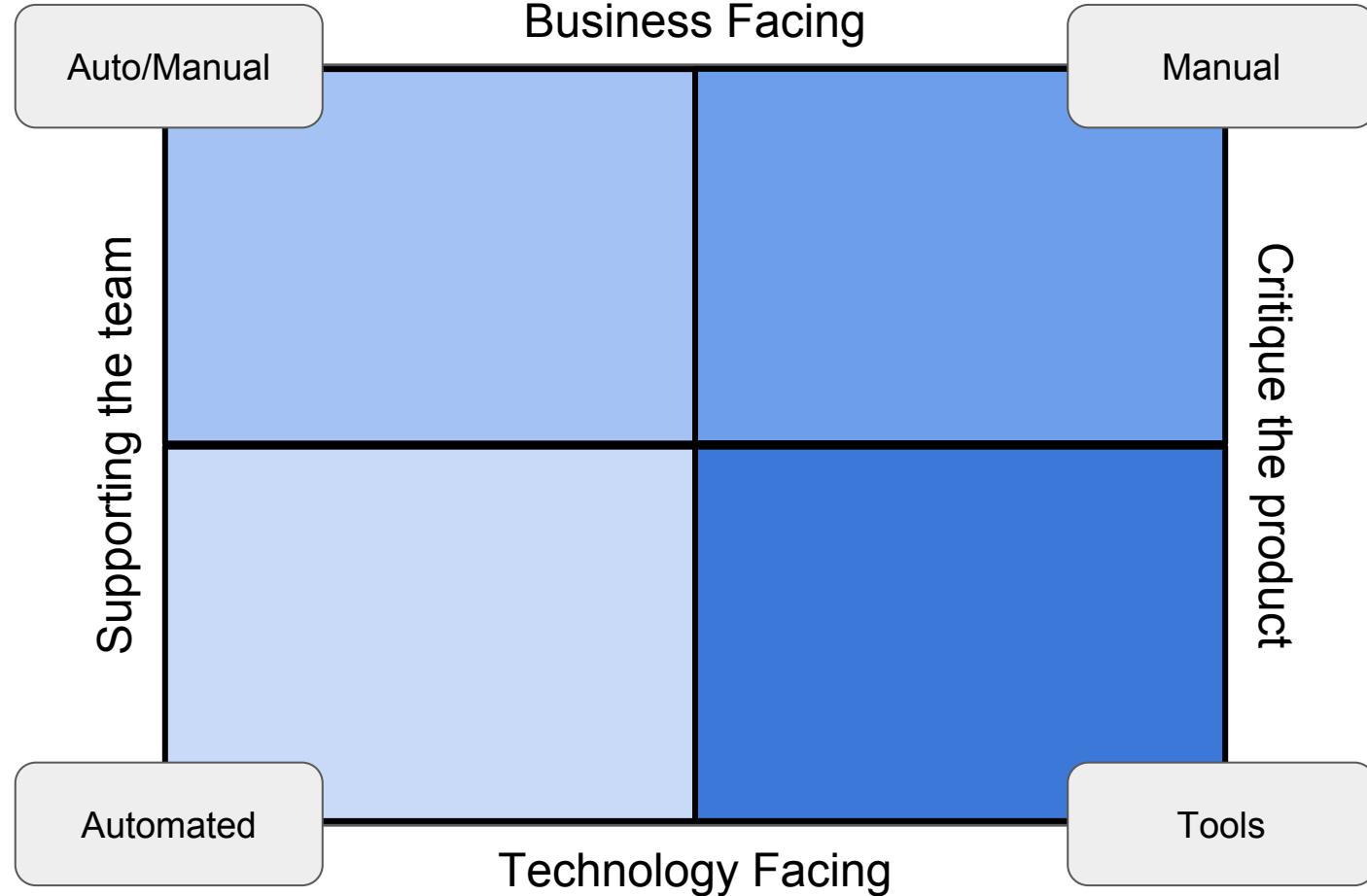
Exploratory testing

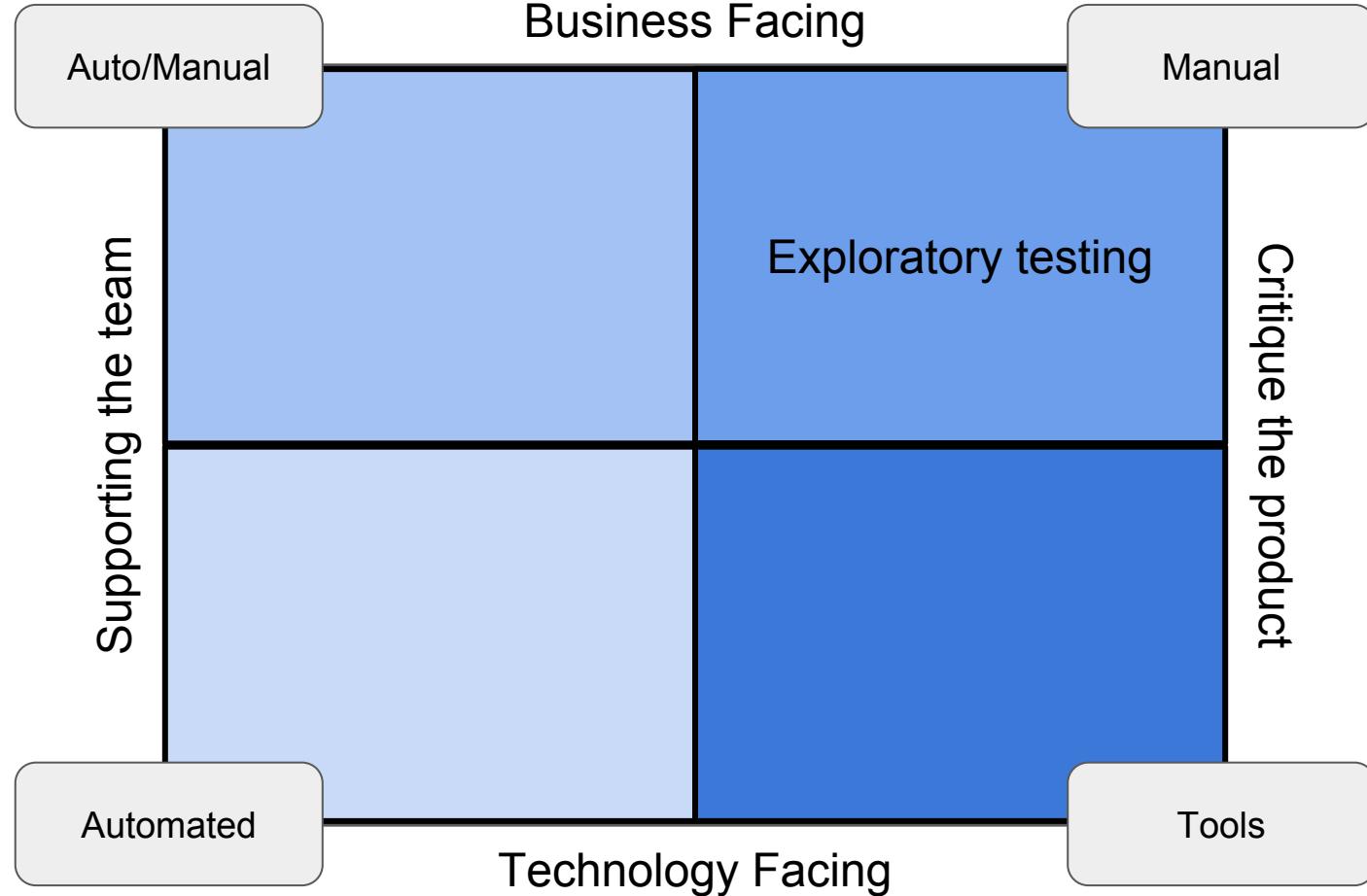
Generally unstructured, time-boxed exploration

Minimum planning, maximum execution

Can be hugely productive/informative, but unpredictable

Experience based





Security testing

Security testing

Don't get hacked!

Security testing

Don't get hacked!

Data protection regulations

Security testing

Don't get hacked!

Data protection regulations

Vulnerability scanning



Security testing

Don't get hacked!

Data protection regulations

Vulnerability scanning

Security reviews/audits



Security testing

Don't get hacked!

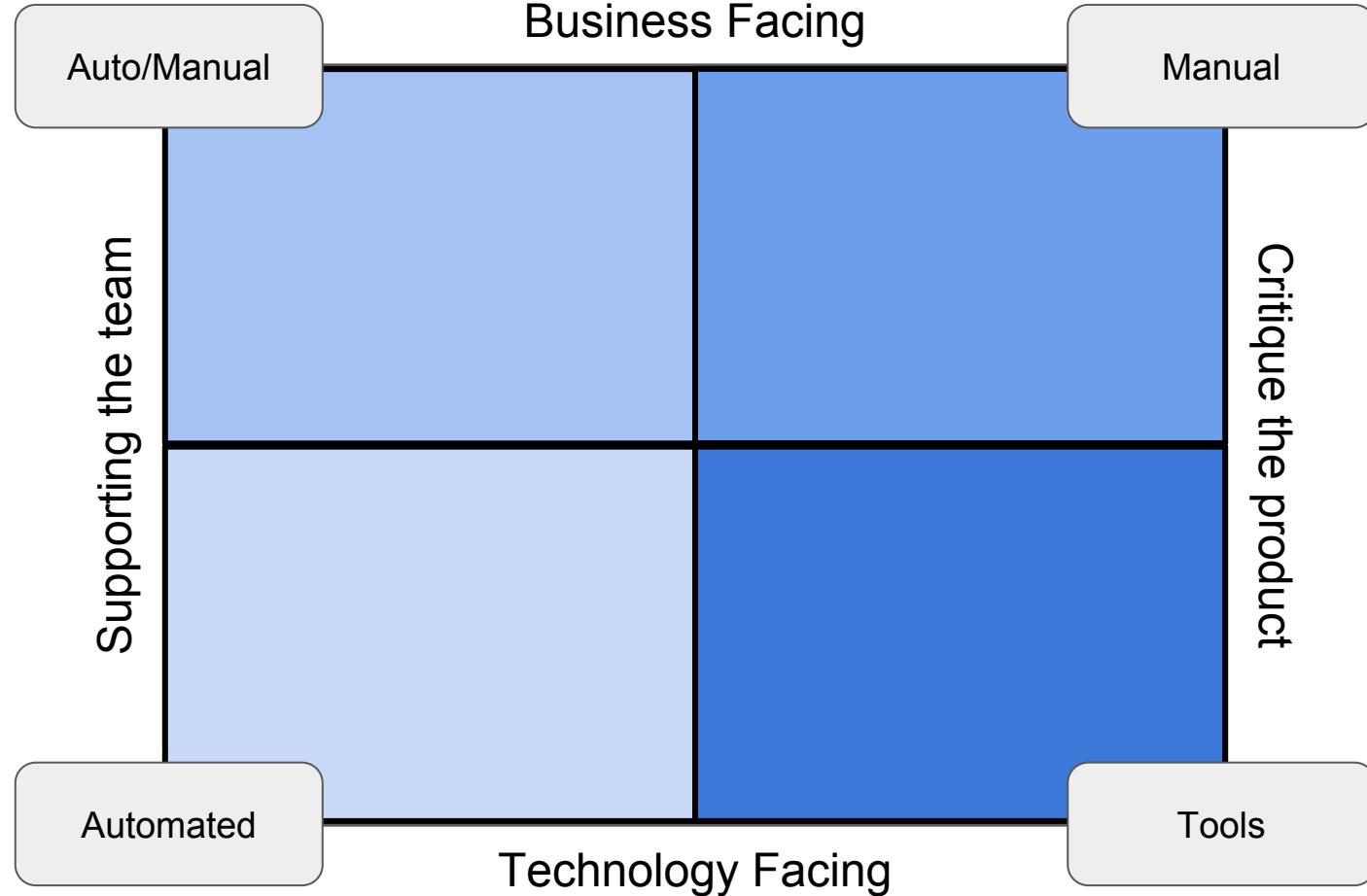
Data protection regulations

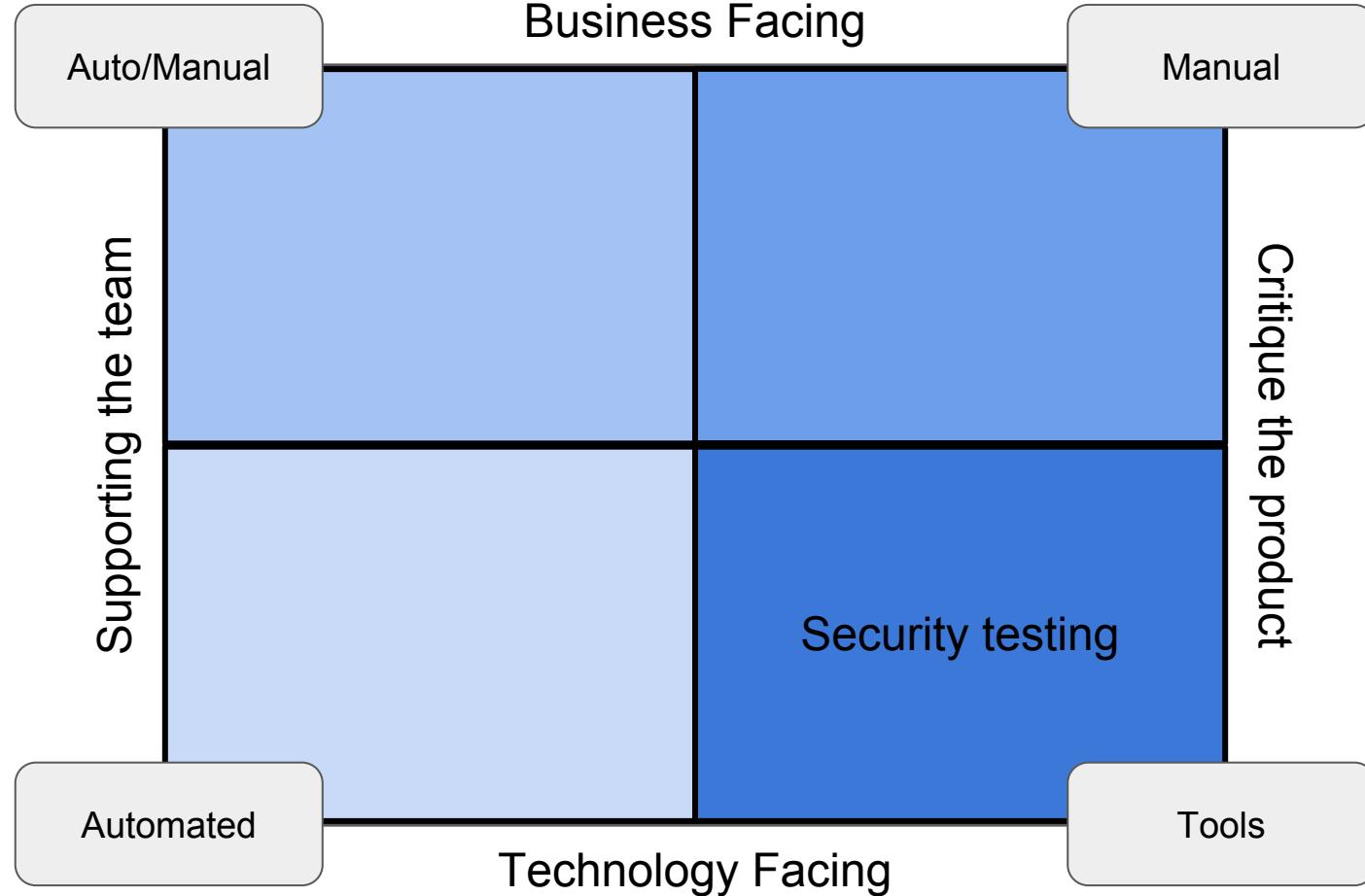
Vulnerability scanning

Security reviews/audits

Penetration testing







Alpha/Beta testing

Alpha/Beta testing

What's the difference between Alpha and Beta?

Alpha/Beta testing

What's the difference between Alpha and Beta?

↖_(ツ)_↗



Alpha/Beta testing

What's the difference between Alpha and Beta?

↖_(ツ)_↗

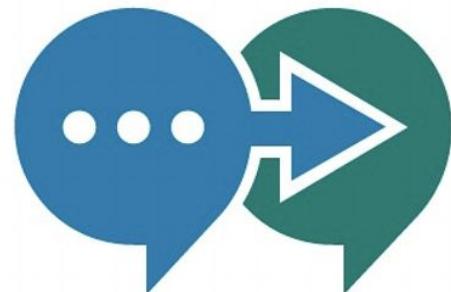


Alpha/Beta testing

What's the difference between Alpha and Beta?

↖_(ツ)_↗

Generally, Beta is more mature



Alpha/Beta testing

What's the difference between Alpha and Beta?

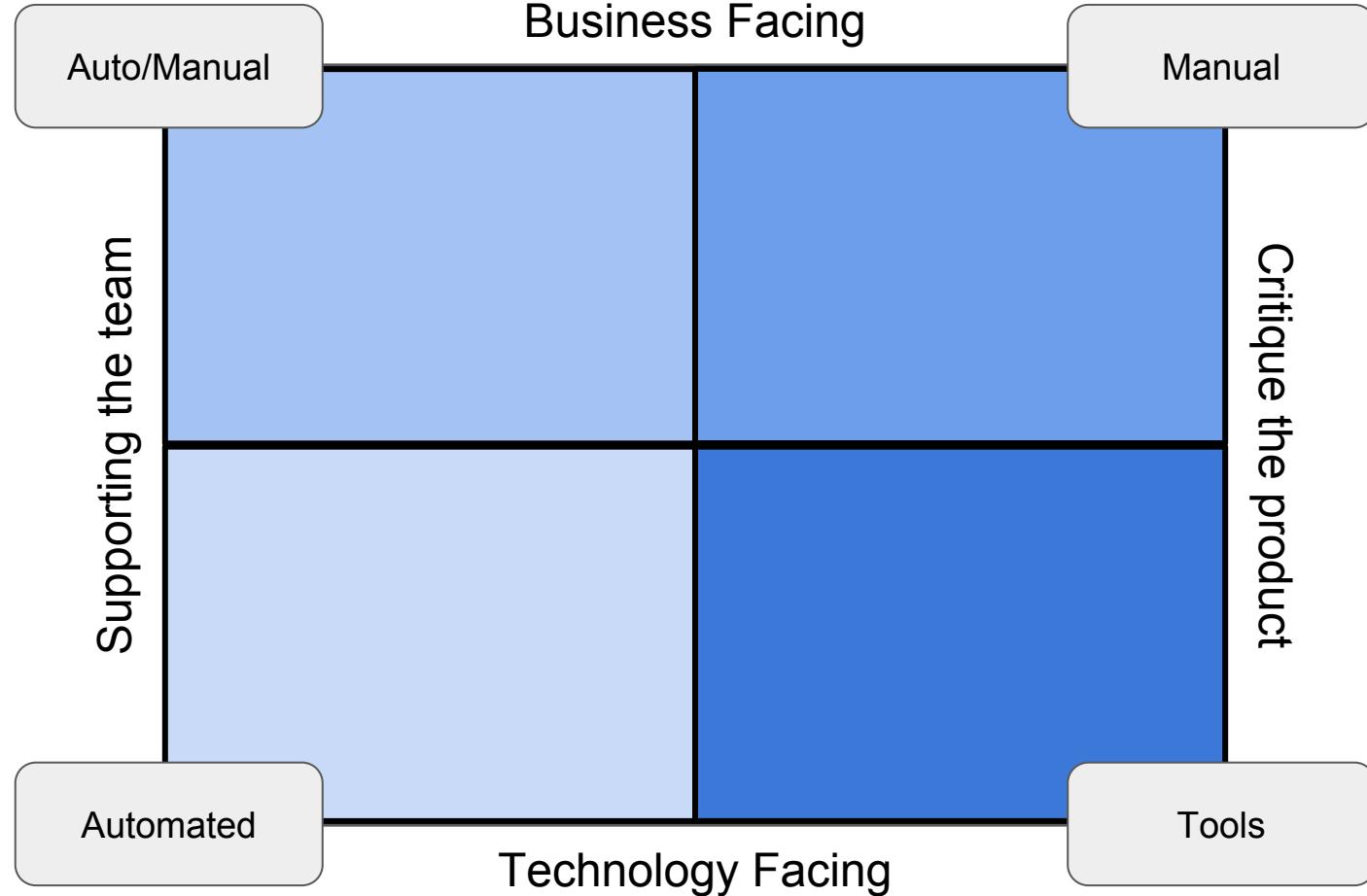
↖_(ツ)_↗

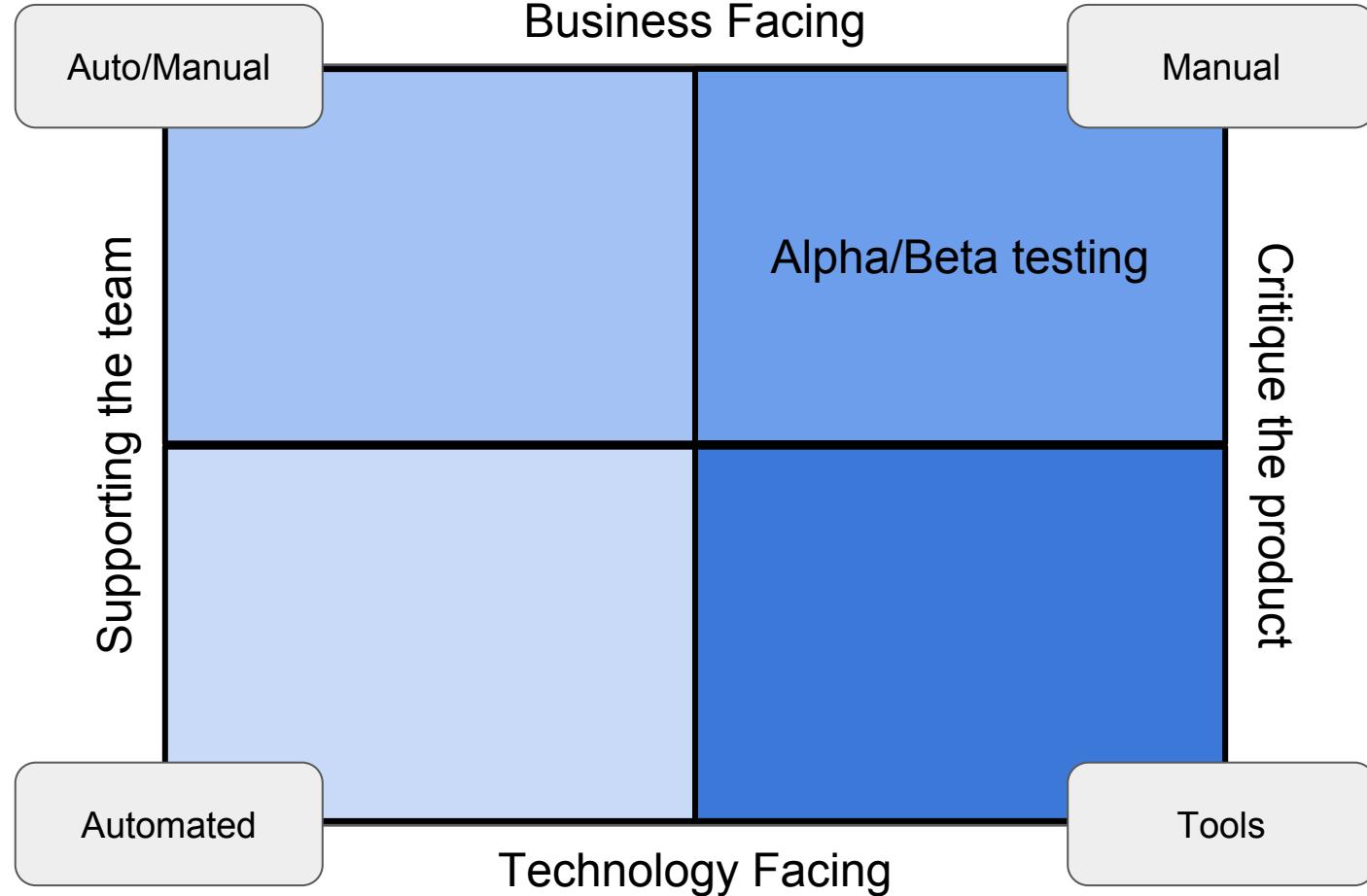
Generally, Beta is more mature

Goals include:

- Testing in close to 'real' environment
- Gain feedback from users (UAT)
- Prepare for release







Defect Management

Defect reporting

Tech support please, when
you have the time 😊. I cannot
search for anything via
Google on my phone but can

Tech support please, when
you have the time 😊. I cannot
search for anything via
Google on my phone but can

Help I can't download
anything

Tech support please, when
you have the time 😊. I cannot
search for anything via
Google on my phone but can

Help I can't download
anything

Phone doesn't work

Tech support please, when
you have the time 😊. I cannot
search for anything via
Google on my phone but can

Help I can't download
anything

Phone doesn't work

Sent from Messenger

Q: What is our goal when reporting a defect?

Q: What is our goal when reporting a defect?

A: Provide enough information to make the defect **actionable**

IEEE 1044-2009

Defect ID - Unique identifier for the defect.

Description - Description of what is missing, wrong, or unnecessary.

Status - Current state within defect report life cycle.

Asset - The software asset (product, component, module, etc.) containing the defect.

Artifact - The specific software work product containing the defect.

Version detected - Identification of the software version in which the defect was detected.

Version corrected - Identification of the software version in which the defect was corrected.

Priority - Ranking for processing assigned by the organization responsible for the evaluation, resolution, and closure of the defect relative to other reported defects.

Severity - The highest failure impact that the defect could (or did) cause, as determined by (from the perspective of) the organization responsible for software engineering.

Probability - Probability of recurring failure caused by this defect

What goes in a defect report?

What goes in a defect report?

Expected/actual results

Steps to reproduce

Supporting materials (screenshots, logs, environment etc.)

Impact/severity

Input from tester (thoughts, intuitions)

How do you write a defect report?

How do you write a defect report?

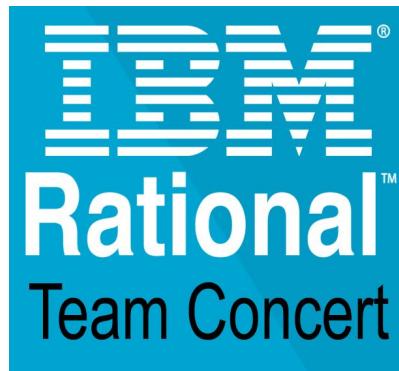
Clear

Accurate

Objective

Defect Management Systems

Defect Management Systems





Teams in Space

Scrum: Teams in Space ▾

Backlog

Agile board

Releases

Reports

All issues

Components

Add-ons

PROJECT SHORTCUTS

Mars Team HipChat Room

Space Station Dev Roadmap

Teams in Space Org Chart

Orbital Spotify Playlist

Hyperspeed Bitbucket Repo

+ Add shortcut

TIS-70 Scrum Board

QUICK FILTERS: Critical partners Only my partners Recently updated

12 To do

2 In progress

3 Done

▼ TIS Developer Love 3 issues

TIS-37

- ↑ Service should return prior trip details and info

SeeSpaceEZ plus

TIS-10

- ↑ Bad JSON data coming back from hotel API

SeeSpaceEZ plus

TIS-8

- ↑ Requesting flights is now taking > 5 seconds

SeeSpaceEZ plus

▼ Everything Else 21 issues

TIS-68

- ↑ Homepage footer uses an inline style-should use class

Large Team Support

TIS-20

- ↑ Engage Saturn Shuttle lines for group tours

Space Travel Partners

TIS-17

- ↑ Engage Saturn's Rings Resort as preferred

Space Travel Partners

TIS-56

- ↑ Add pointer to main css file to create child themes

Large Team Support

TIS-12

- ∅ Create 90 day plans for all departments in Mars office

SeeSpaceEZ plus

Create Issue

Configure Fields ▾

Project* 1

Issue Type* Bug 2

Some issue types are unavailable due to incompatible field configuration and/or workflow associations.

Summary* 3

Priority* 4

Component/s

Start typing to get a list of possible matches or press down to select.

Affects Version/s

Start typing to get a list of possible matches or press down to select.

Environment 5

For example operating system, software platform and/or hardware specifications (include as appropriate for the issue).

Description 6

Steps to reproduce:

- * Open PCGen.
- * Load the 35e data set.
- * Create a new character.
- * Click the "Add class" button.
- * Select "Fighter"
- * The program crashes.



Theme

▾

Labels

Begin typing to find and create labels or press down to select a suggested label.

Create another

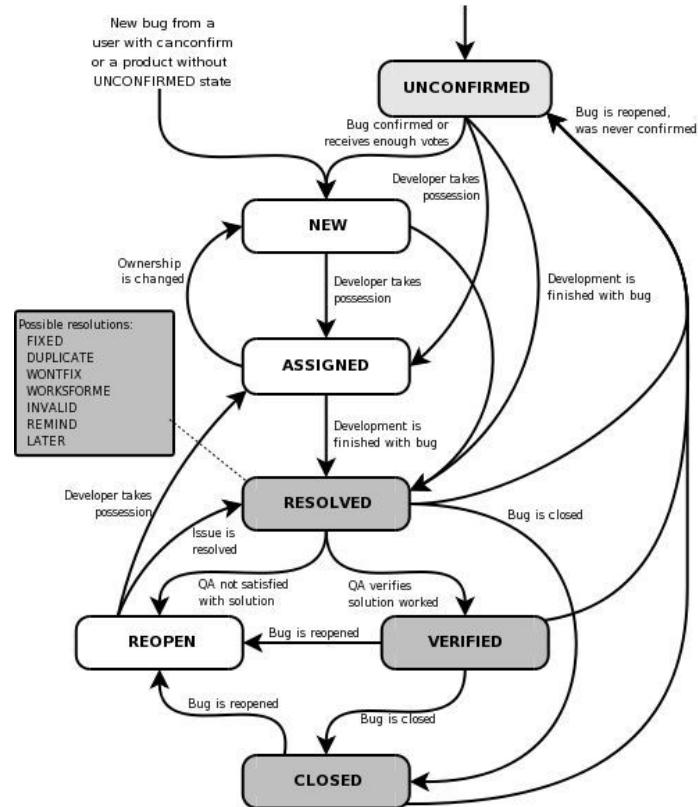
Create

Cancel

Do you need a Defect Management System?

Do you need a Defect Management System?

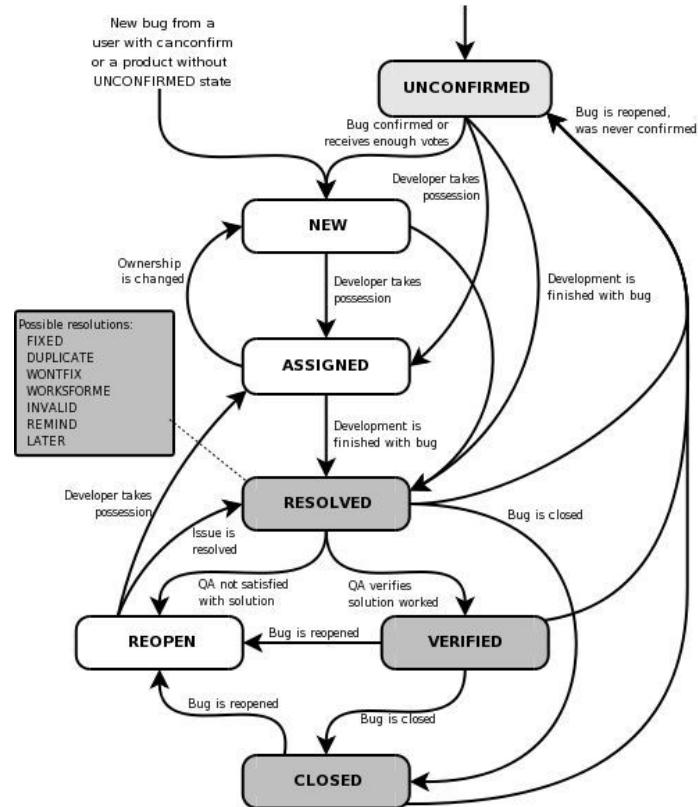
Useful for tracking



Do you need a Defect Management System?

Useful for tracking

Useful for auditing and traceability

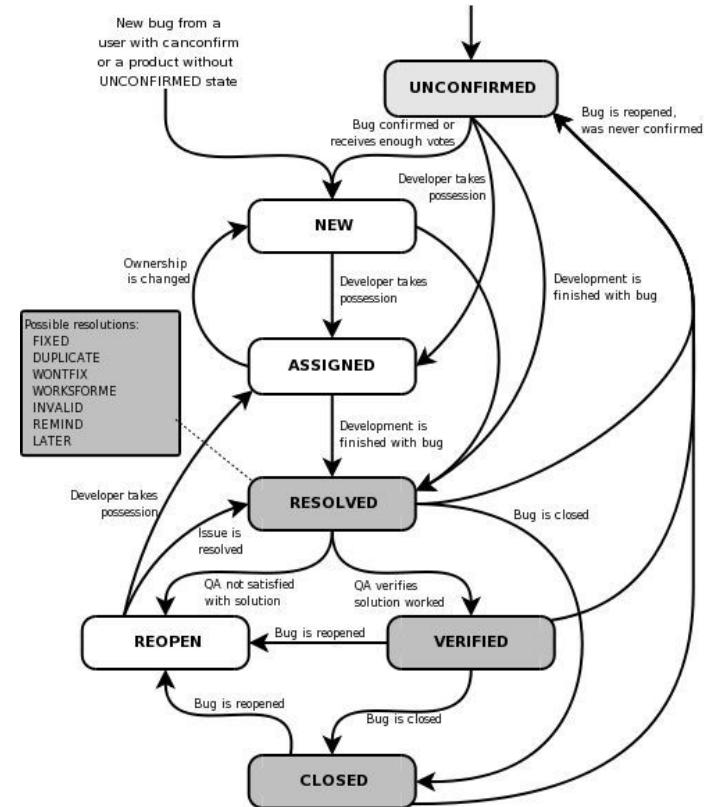


Do you need a Defect Management System?

Useful for tracking

Useful for auditing and traceability

Less relevant for Agile processes



Do you need a Defect Management System?

Useful for tracking

Useful for auditing and traceability

Less relevant for Agile processes

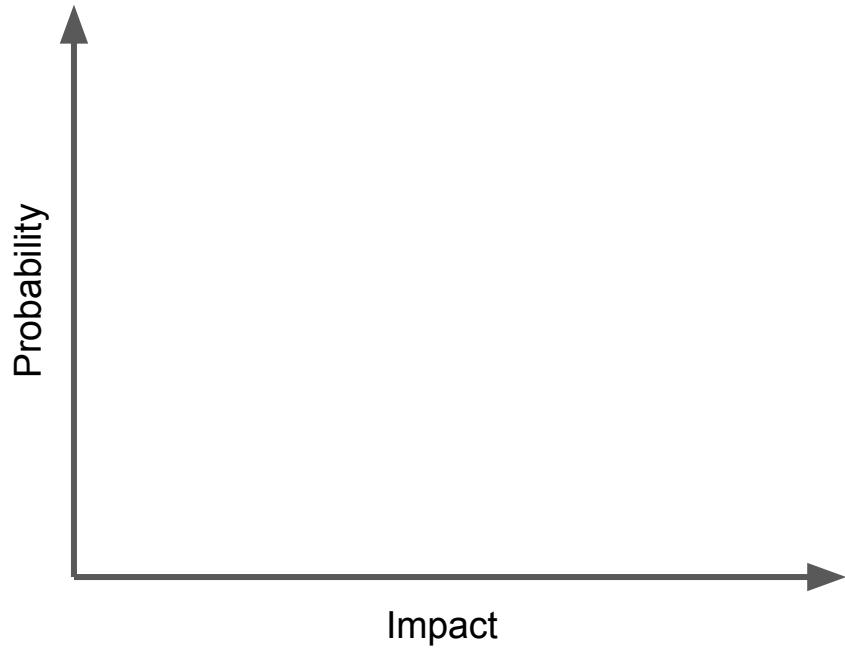
Communication is key!

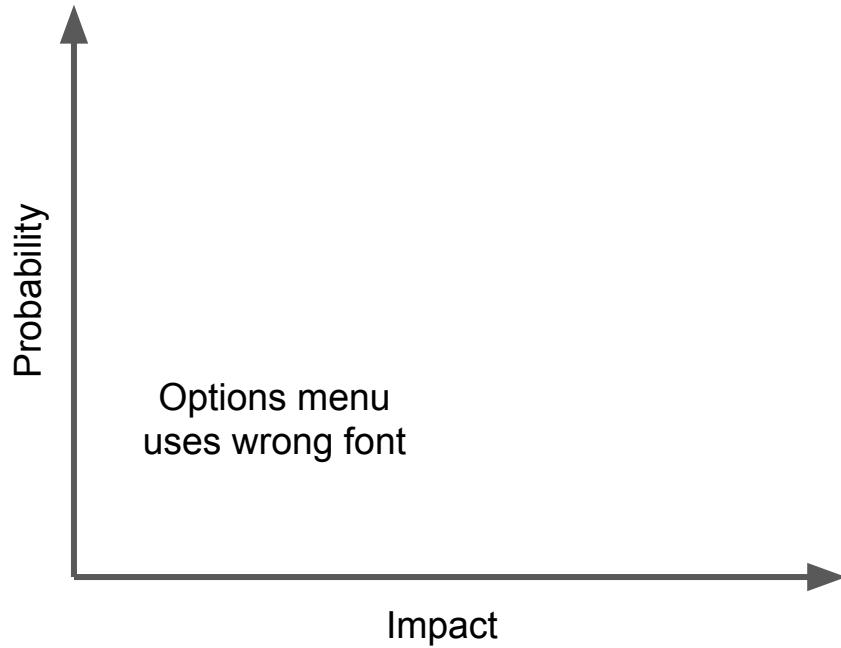
“Defect tracking systems
don’t promote
communication between
programmers and testers.”

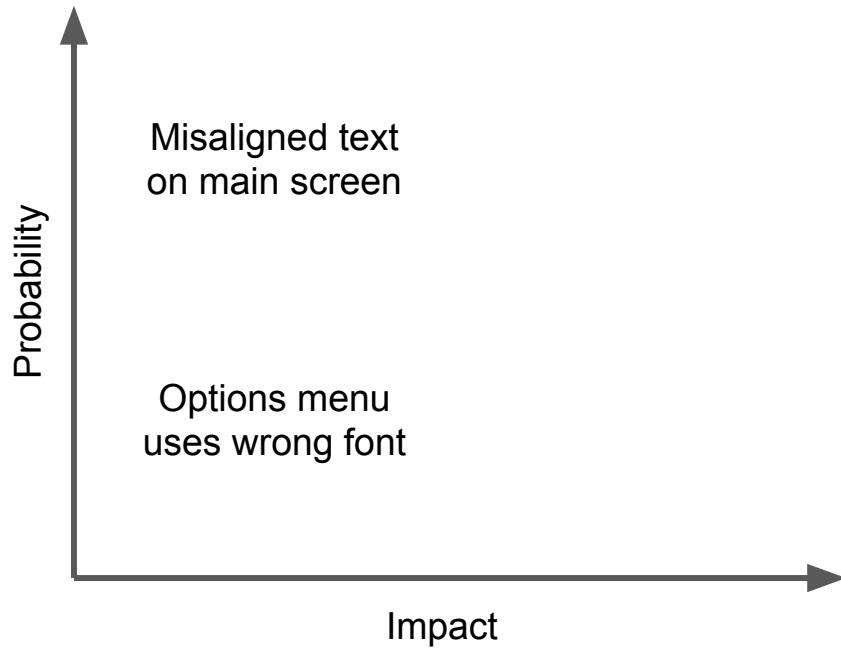
Lisa Crispin, Agile Testing

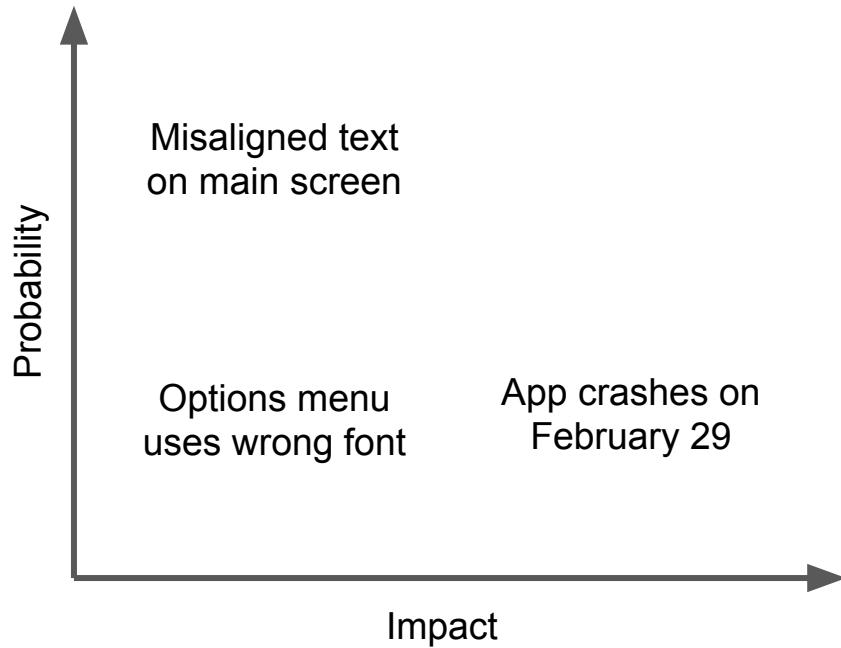
Defect prioritisation

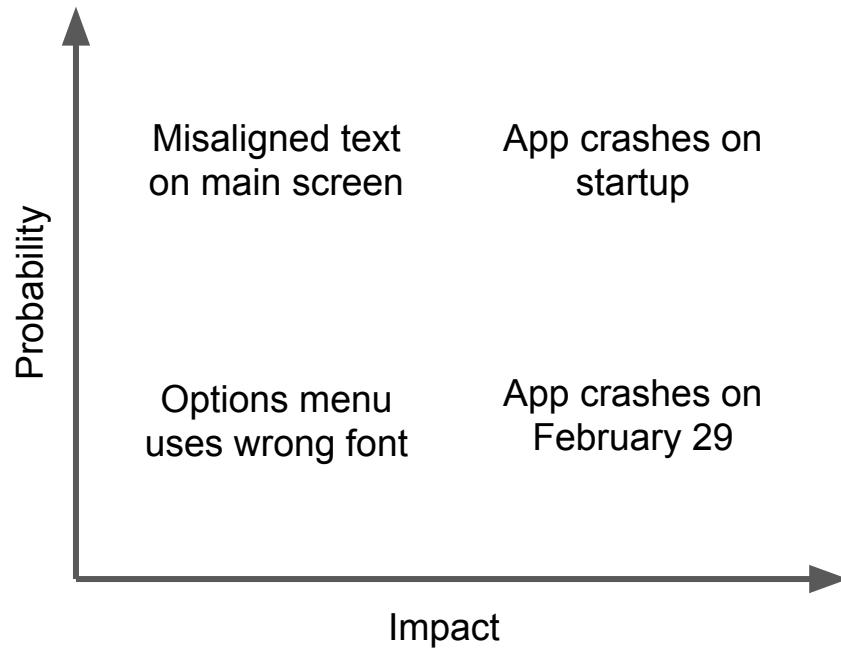
Defect prioritisation (how bad is it?)











Severity

How bad is it?

Blocker

Critical

Major

Medium

Minor

Trivial

Priority

How quickly should we fix it?

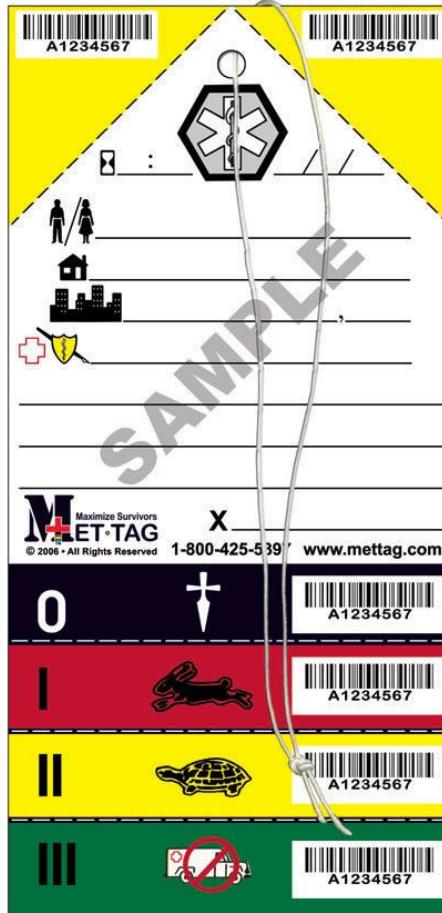
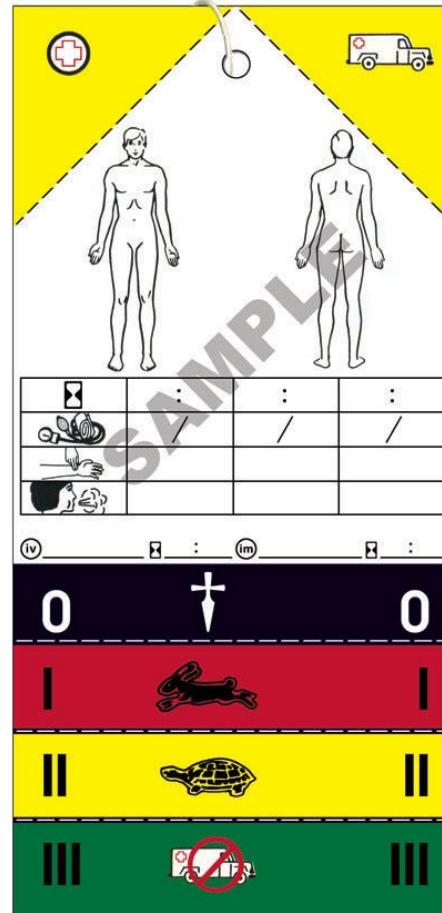
High

Medium

Low



SIGNAL CORPS
SP

FRONT**BACK**

Bugzilla@Mozilla

Open Bugs

24525

Critical

- The defects that need to be fixed **immediately** because it may cause great damage to the product

High

- The defect impacts the product's **main** features

Medium

- The defect causes **minimal** deviation from product requirement

Low

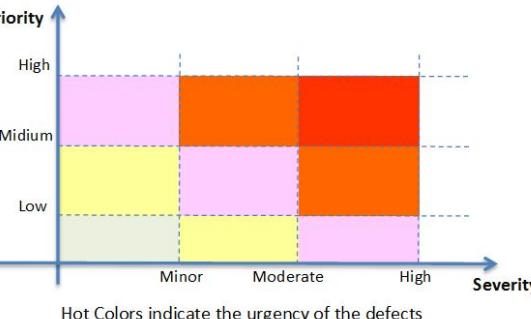
- The defect has **very minor** affect product operation

SEVERITY

3 - LOW 2 - MEDIUM 1 - HIGH

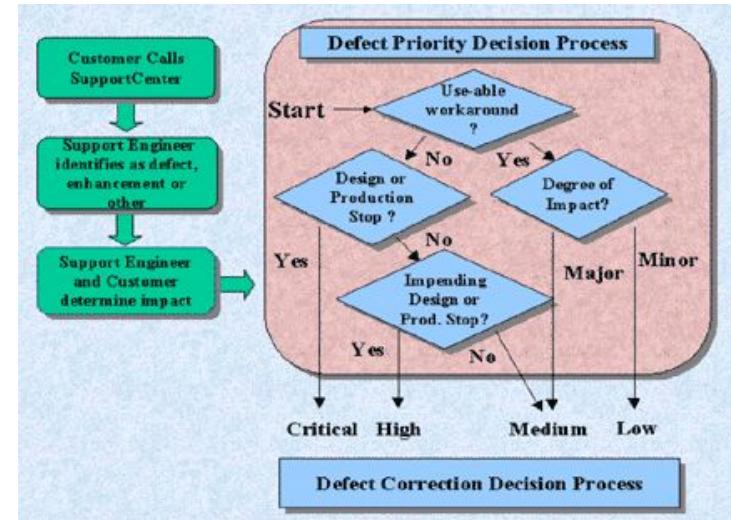
	P2	P1
P3	P2	P2
P4	P3	

Priority ↑
3 - LOW 2 - MEDIUM 1 - HIGH
URGENCY ↓



EXAMPLE RISK

		Probability				
		Very High	High	Medium	Low	Very Low
Consequence	Very High	Very High	Very High	Very High	High	High
	High	Very High	High	High	Medium	Medium
	Medium	High	High	Medium	Medium	Low
	Low	High	Medium	Medium	Low	Very Low
	Very Low	Medium	Low	Low	Very Low	Very Low



Defect Matrix

	Users affected			
Type	100% – 75%	75% – 50%	50% – 25%	< 25%
Crash	High	High	High	High
Functionality not available	High	High	Medium	Medium
Functionality partially available	High	Medium	Medium	Low
Performance	Medium	Low	Low	Low
Cosmetic	Low	Low	Low	Low

Defect Matrix

	Users affected			
Type	100% – 75%	75% – 50%	50% – 25%	< 25%
Crash	Fix now	Fix now	Fix now	Fix now
Functionality not available	Fix now	Fix now	Fix now	Fix later
Functionality partially available	Fix later	Fix later	Fix later	Fix later
Performance	Fix later	Fix later	Fix later	Fix later
Cosmetic	Fix later	Fix later	Fix later	Fix later

Defect Matrix

Type	< 25%
Crash	Fix now
Functionality not available	Fix later
Functionality partially available	Fix later
Performance	Fix later
Cosmetic	Fix later

Fix now

or

Fix later



Test metrics

What are metrics?

What are metrics?

A way of understanding our effectiveness

A way to identify problems

A guide to success

What are metrics?

A way of understanding our effectiveness

A way to identify problems

A guide to success

“A pit of **wasted effort** ...
numbers for the sake of
numbers ... **actively
harmful**”

Lisa Crispin, Agile Testing

Code coverage

Code coverage

How much of the code has been tested (covered)?

Several ways of measuring

Generally summed over a test suite

Code coverage - Types

Function coverage

How many functions/methods have been called?

Statement coverage

How many executable statements have been tested?

Branch coverage

Has every branch in program flow been tested?

```
def greet(name):  
    print 'Hello', name  
  
def comment_on_age(age):  
    print 'In ten years, you will be', (age + 10)
```

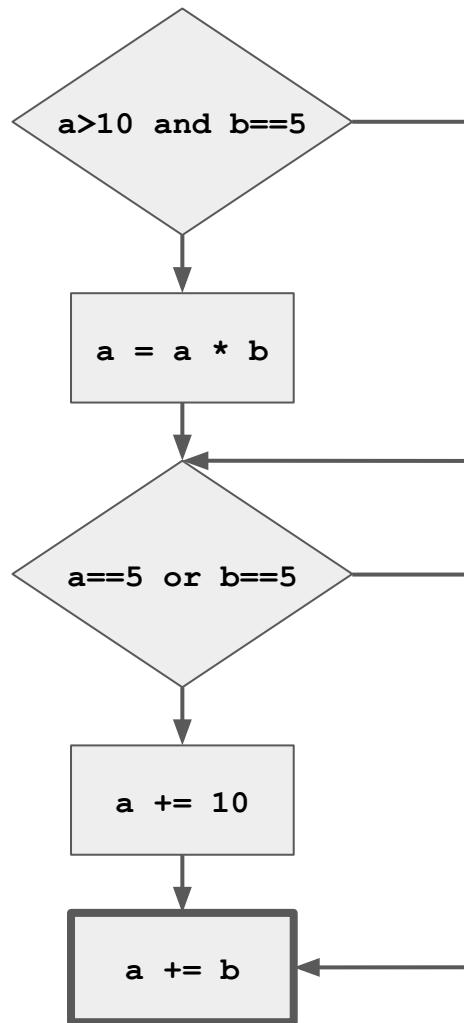
```
def func(a, b):

    if (a > 10 and b == 5):
        a = a * b

    if (a == 5 or b == 5):
        a += 10

    a += b
```

```
def func(a, b):  
  
    if (a > 10 and b == 5):  
  
        a = a * b  
  
    if (a == 5 or b == 5):  
  
        a += 10  
  
    a += b
```



```
def calculate_interest(balance):

    interest = 0.1

    if (balance > 1000):

        interest = 0.05

    if (balance > 50000):

        interest += 0.1

    else:

        interest += 0.05

    balance = balance + (1 * interest)
```

```
def calculate_interest(balance):

    interest = 0.1

    if (balance > 1000):

        interest = 0.05

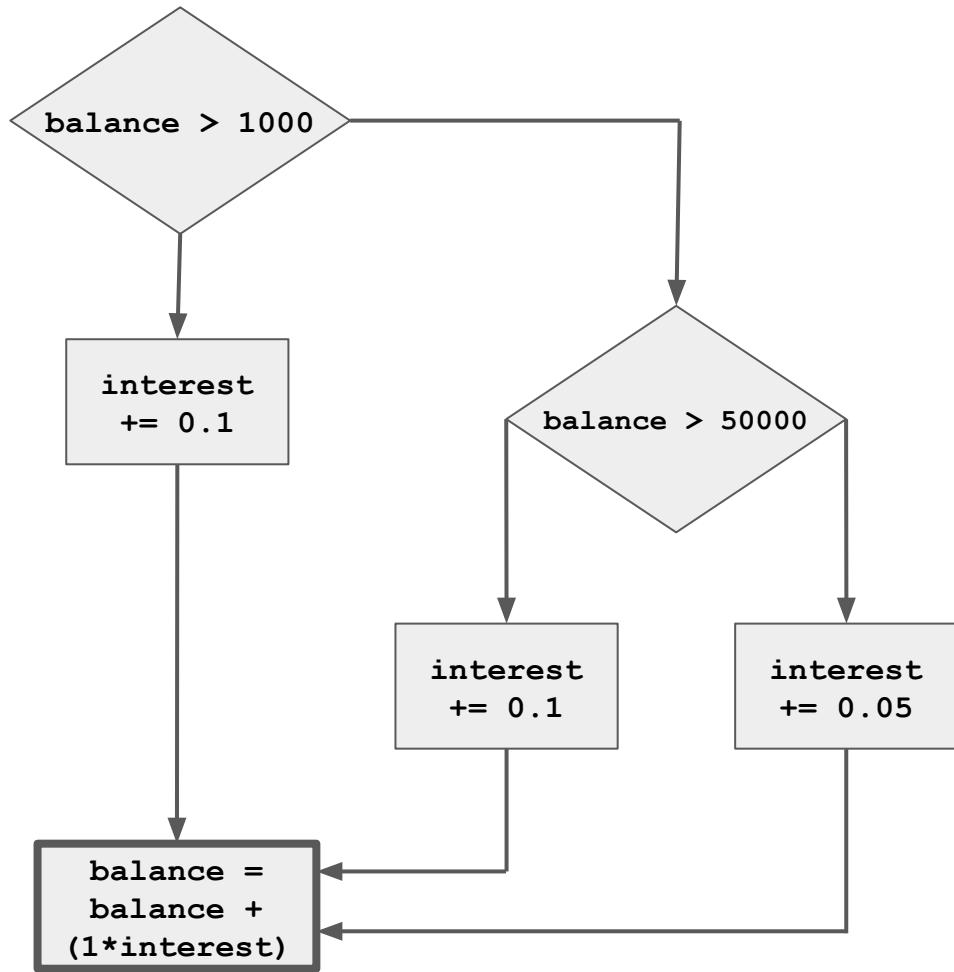
    if (balance > 50000):

        interest += 0.1

    else:

        interest += 0.05

    balance = balance + (1 * interest)
```



```
def calculate_speed(speed, drag):

    if (speed > 100):

        if (drag > 50):

            return speed - (drag * 2)

        else:

            speed -= drag

    else:

        if (drag > 50):

            speed -= (drag * 0.5)

    return speed - (drag * 0.3)
```

```
def calculate_speed(speed, drag):

    if (speed > 100):

        if (drag > 50):

            return speed - (drag * 2)

        else:

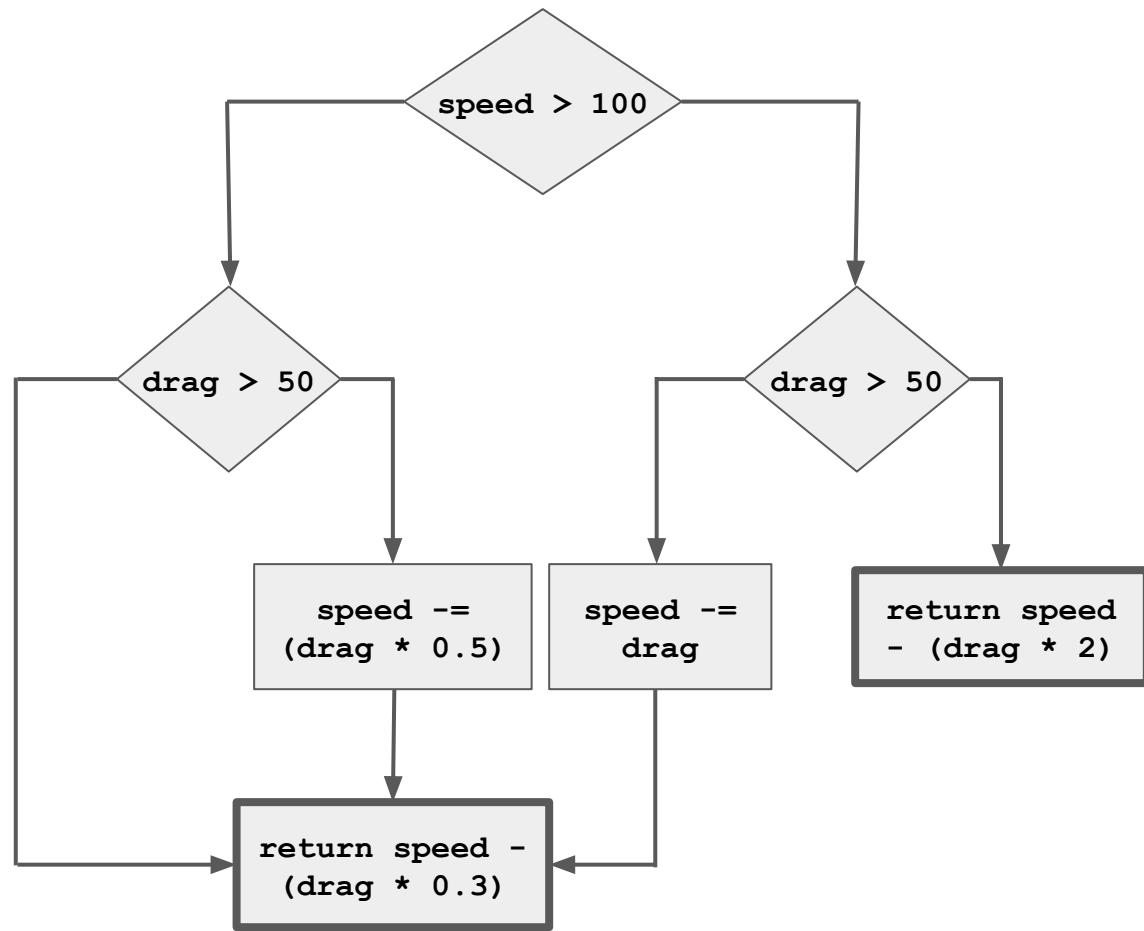
            speed -= drag

    else:

        if (drag > 50):

            speed -= (drag * 0.5)

    return speed - (drag * 0.3)
```



```
288         this.imagePlus = imagePlus;
289
290         this.stackMin = stackMin;
291         this.stackMax = stackMax;
292
293         this.bidirectional = bidirectional;
294         this.definedGoal = definedGoal;
295         this.startPaused = startPaused;
296
297         this.imageType = imagePlus.getType();
298
299         width = imagePlus.getWidth();
300         height = imagePlus.getHeight();
301         depth = imagePlus.getStackSize();
302
303     {
304         ImageStack s = imagePlus.getStack();
305         switch(imageType) {
306             case ImagePlus.GRAY8:
307             case ImagePlus.COLOR_256:
308                 slices_data_b = new byte[depth][];
309                 for( int z = 0; z < depth; ++z )
310                     slices_data_b[z] = (byte []) s.getPixels( z + 1 );
311                 break;
312             case ImagePlus.GRAY16:
313                 slices_data_s = new short[depth][];
314                 for( int z = 0; z < depth; ++z )
315                     slices_data_s[z] = (short []) s.getPixels( z + 1 );
316                 break;
317             case ImagePlus.GRAY32:
318                 slices_data_f = new float[depth][];
319                 for( int z = 0; z < depth; ++z )
320                     slices_data_f[z] = (float []) s.getPixels( z + 1 );
321                 break;
322             }
323         }
324
325         Calibration calibration = imagePlus.getCalibration();
326
327         x_spacing = (float)calibration.pixelWidth;
```

JSCover

[Summary](#) [Source](#) [About](#)
 Show missing statements column

File	Statements	Executed	Branches	Branches Hit	Functions	Functions Hit	Coverage	Branch	Function
Total:	23	1032	350	8436	1023	1781	439 33%	12%	24%
/experiences/healthcare/js/healthcare/config.js	2	2	0	0	0	0	100%	N/A	N/A
/experiences/healthcare/js/healthcare/confighandler.js	24	11	12	5	4	2	45%	41%	50%
/experiences/healthcare/js/healthcare/throttlingService.js	3	3	0	0	1	0	100%	N/A	0%
/experiences/healthcare/js/healthcare/dependencies/ga_custom_code.js	249	144	180	60	41	22	57%	33%	53%
/experiences/healthcare/js/healthcare/test-constructor.js	30	27	26	11	4	3	90%	42%	75%
/experiences/healthcare/js/healthcare/dependencies/detectmobilebrowser.min.js	1	1	6	0	2	0	100%	0%	0%
/experiences/healthcare/js/healthcare/dependencies/ieFixes.js	62	10	66	9	11	0	16%	13%	0%
/experiences/healthcare/js/healthcare/dependencies/select2_min.js	2	1	1282	110	250	52	50%	8%	20%
/experiences/healthcare/js/healthcare/dependencies/underscore.min.js	1	1	496	71	151	32	100%	14%	21%
/experiences/healthcare/js/healthcare/dependencies/handlebars.min.js	2	2	570	154	165	66	100%	27%	40%
/experiences/healthcare/js/healthcare/dependencies/backbone.min.js	1	1	556	117	117	51	100%	21%	43%
/experiences/healthcare/js/healthcare/dependencies/backbone-validation.min.js	1	1	168	4	58	6	100%	2%	10%
/experiences/healthcare/js/healthcare/dependencies/backbone.localStorage-min.js	1	1	70	25	23	11	100%	35%	47%
/experiences/healthcare/js/healthcare/dependencies/backbone.layoutmanager.js	316	32	186	11	63	4	10%	5%	6%
/experiences/healthcare/js/healthcare/dependencies/jquery.autotab.min.js	11	1	256	6	20	1	9%	2%	5%
/experiences/healthcare/js/healthcare/dependencies/placeholders.jquery.min.js	1	1	164	3	31	4	100%	1%	12%
/experiences/healthcare/js/healthcare/dependencies/mediator.js	132	30	102	6	24	4	22%	5%	16%
/experiences/healthcare/js/tlhealth.min-63b54fa1b3e21ac0bbfb7fdff56fc844.js	5	2	2494	88	488	46	40%	3%	9%
/experiences/devmode/js/toolbar.js	101	32	42	3	22	5	31%	7%	22%
/experiences/healthcare/js/healthcare/dependencies/ga_invoke.js	84	44	76	26	20	15	52%	34%	75%
/js/widget/webengage-min-v-4.0.js	1	1	1684	314	286	115	100%	18%	40%
/gz.js	1	1	0	0	0	0	100%	N/A	N/A
/webengage-files/webengage/76aa436/v3.js	1	1	0	0	0	0	100%	N/A	N/A

`coverage.py`

`pip install coverage`

Specify what source files to monitor

`coverage run my_program.py`

Output a report

`coverage report -m`

`$ coverage report -m`

Name	Stmts	Miss	Cover	Missing

<code>my_program.py</code>	20	4	80%	33-35, 39
<code>my_other_module.py</code>	56	6	89%	17-23

<code>TOTAL</code>	76	10	87%	

`coverage.py`

`pip install coverage`

Specify what source files to monitor

`coverage run my_program.py`

Output a report

`coverage report -m`

Coverage report: 37.59%

<i>Module ↓</i>	<i>statements</i>	<i>missing</i>	<i>excluded</i>	<i>branches</i>	<i>partial</i>	<i>coverage</i>
cogapp/__init__.py	2	0	0	0	0	100.00%
cogapp/__main__.py	3	3	0	0	0	0.00%
cogapp/backward.py	19	8	0	2	1	57.14%
cogapp/cogapp.py	427	197	4	176	26	47.10%
cogapp/makefiles.py	28	20	3	14	0	19.05%
cogapp/test_cogapp.py	704	486	6	6	0	30.99%
cogapp/test_makefiles.py	55	55	0	6	0	0.00%
cogapp/test_whiteutils.py	69	69	0	0	0	0.00%
cogapp/whiteutils.py	45	3	0	32	3	92.21%
Total	1352	841	13	236	30	37.59%

coverage.py v4.3.2, created at 2017-01-16 18:24

pytest-cov

Plugin for [pytest](#)

`pip install pytest-cov`

Specify what source files to monitor

`--cov=game_of_life`

```
$ pytest test_game_of_life.py --cov=game_of_life
=====
plugins: cov-2.4.0
collected 1 items

test_game_of_life.py .

-----
coverage: platform linux2, python 2.7.12-final-0 ----
Name          Stmts    Miss   Cover
-----
game_of_life.py      39      26    33%
=====
1 passed in 0.03 seconds =====
```

pytest-cov

Plugin for `pytest`

`pip install pytest-cov`

Specify what source files to monitor

`--cov=game_of_life`

```
$ pytest test_game_of_life.py --cov=game_of_life  
--cov-report=html
```

Coverage report: 33%

<i>Module ↓</i>	<i>statements</i>	<i>missing</i>	<i>excluded</i>	<i>coverage</i>
game_of_life.py	39	26	0	33%
Total	39	26	0	33%

coverage.py v4.3.4, created at 2017-02-02 06:44

Is code coverage a useful metric?

Is code coverage a useful metric?

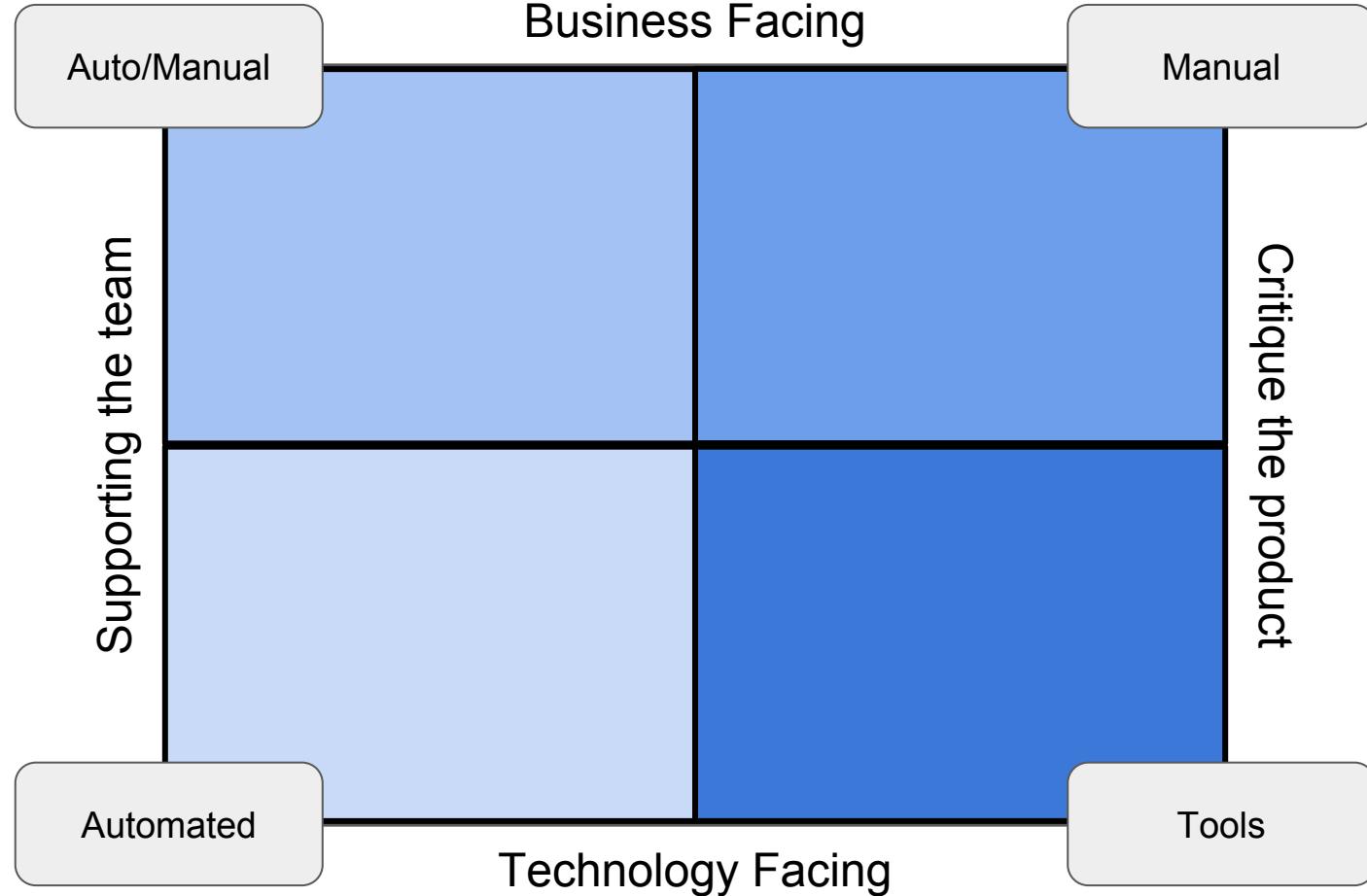
Bad (even broken) tests can increase coverage

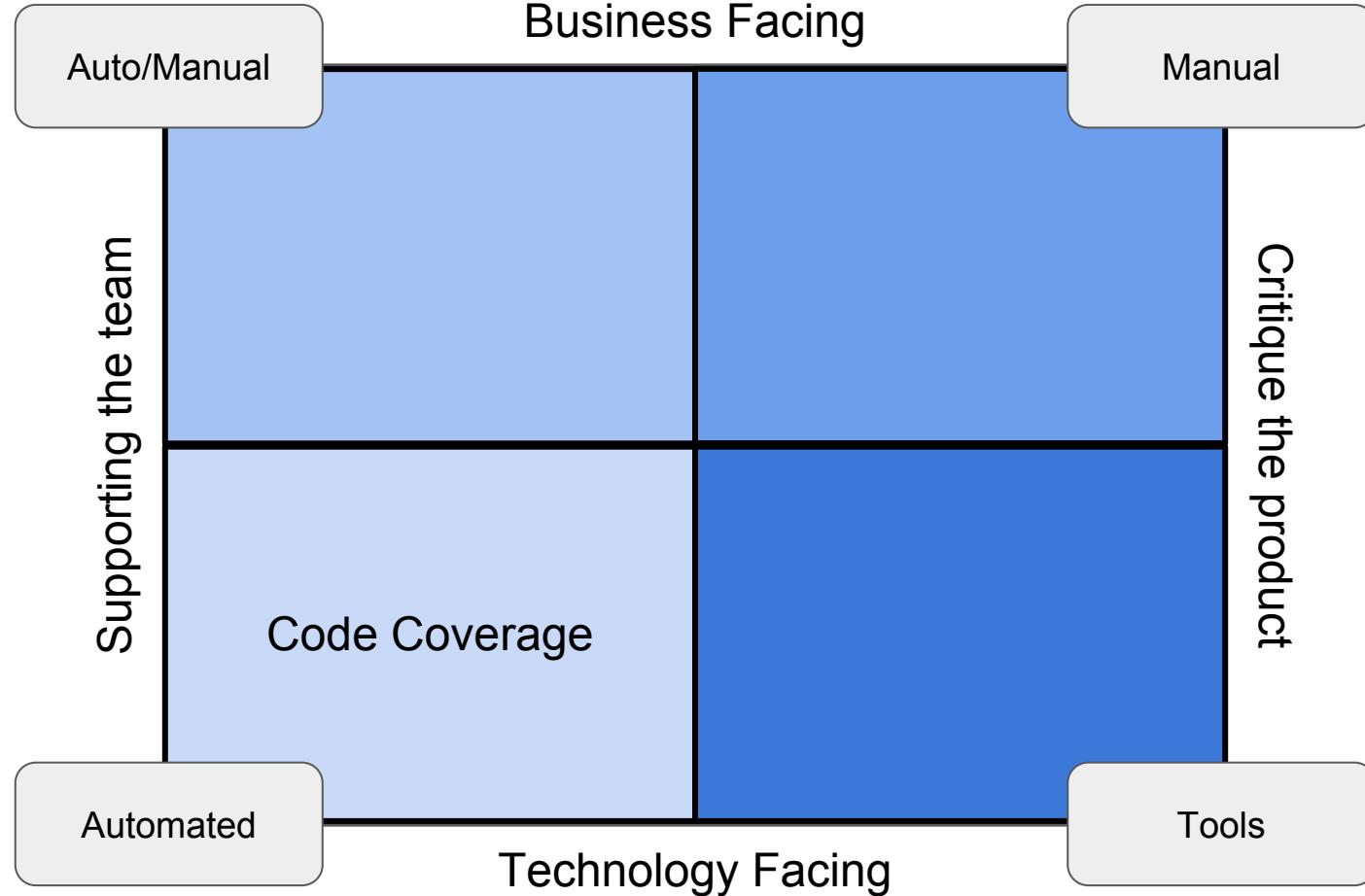
False sense of security?

Using coverage as a shipping gate leads to bad practices

Can give helpful cues about weak spots in your tests

Expect good coverage, don't **require** it (Brian Marick)





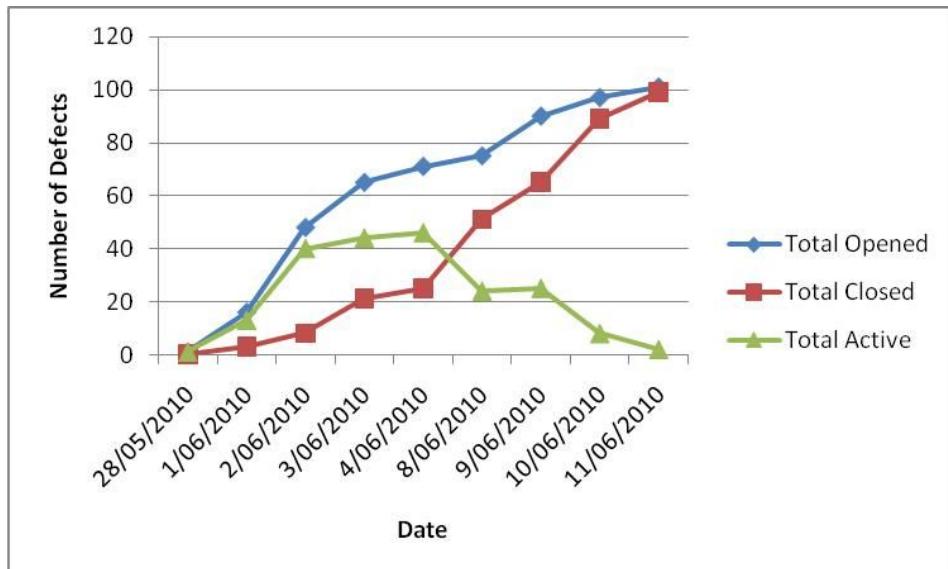
Defect metrics

Number of defects in each state

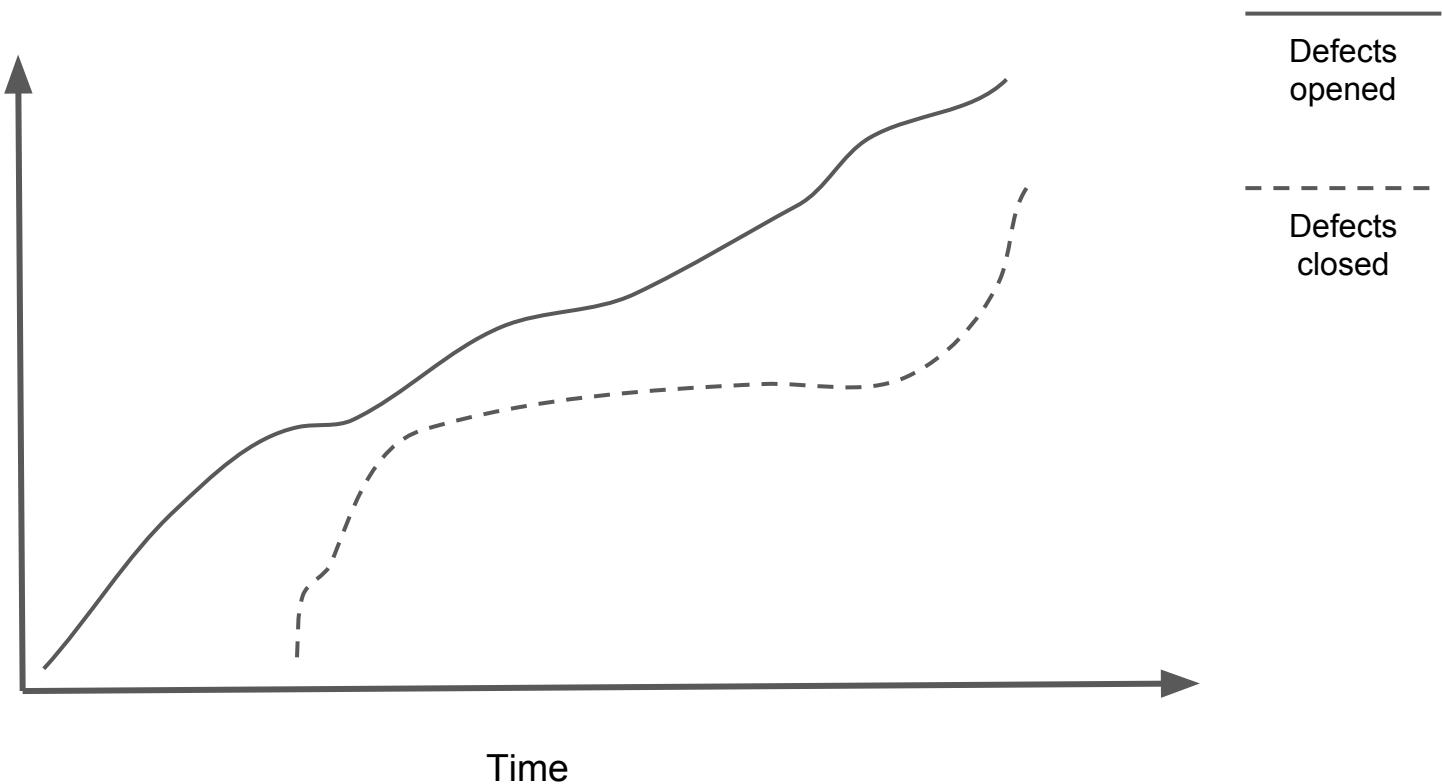
Rate of opening vs closing

Average age of a defect

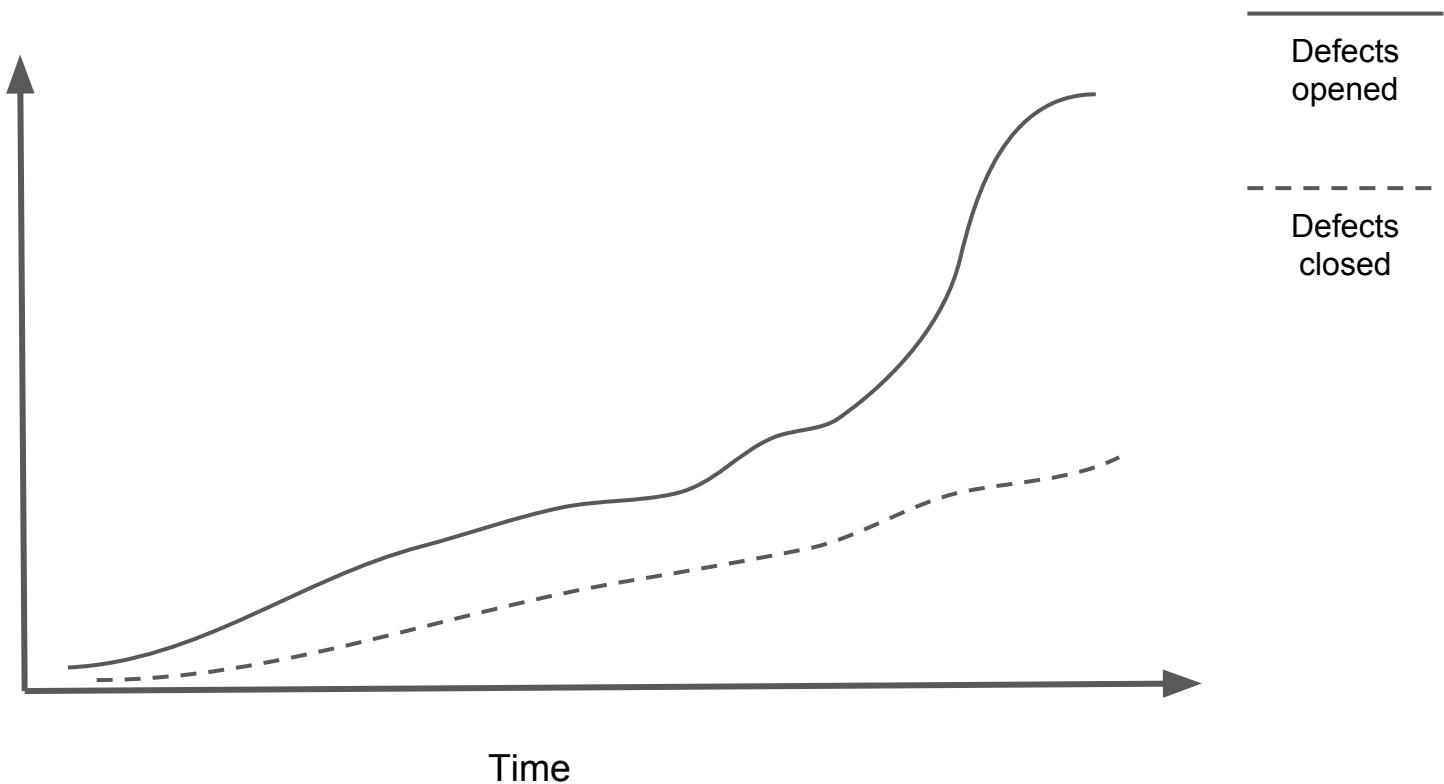
Per engineer/component counts...



Defect metrics

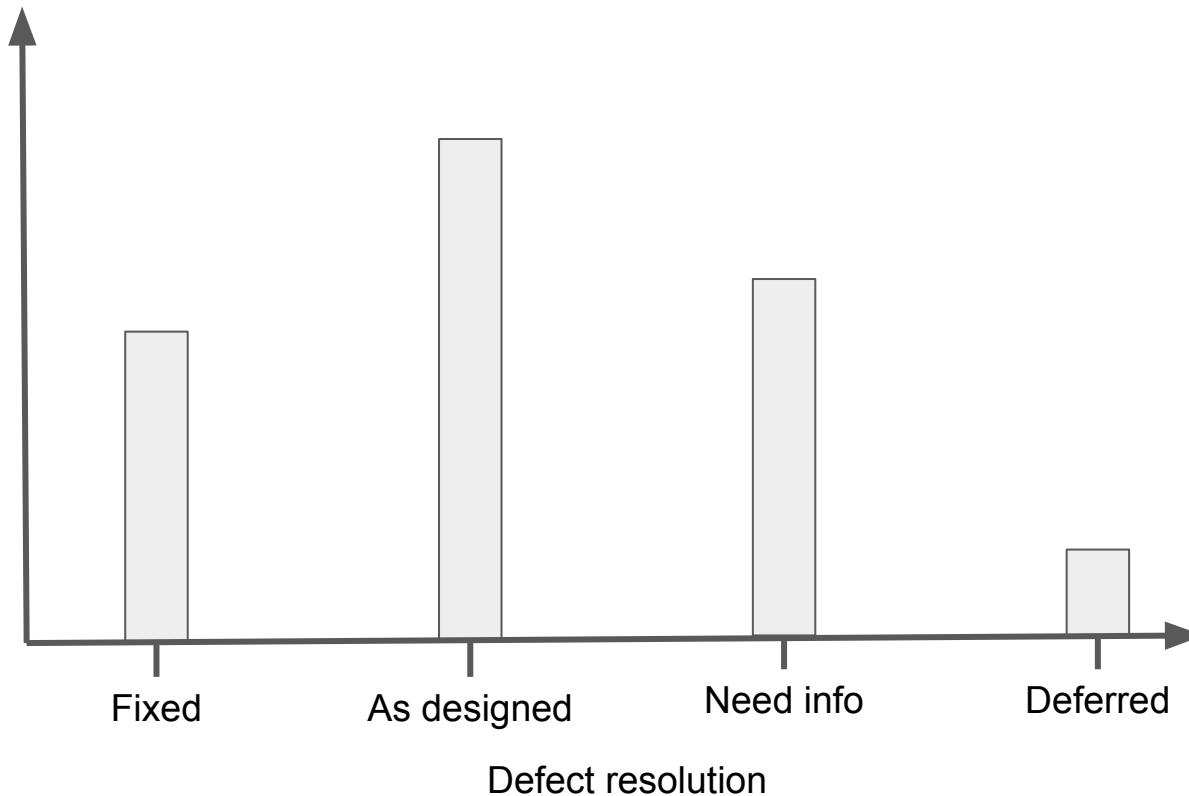


Defect metrics





Defect metrics



Are defect counts a good metric?

Useful for understanding velocity

Can be used to identify problems

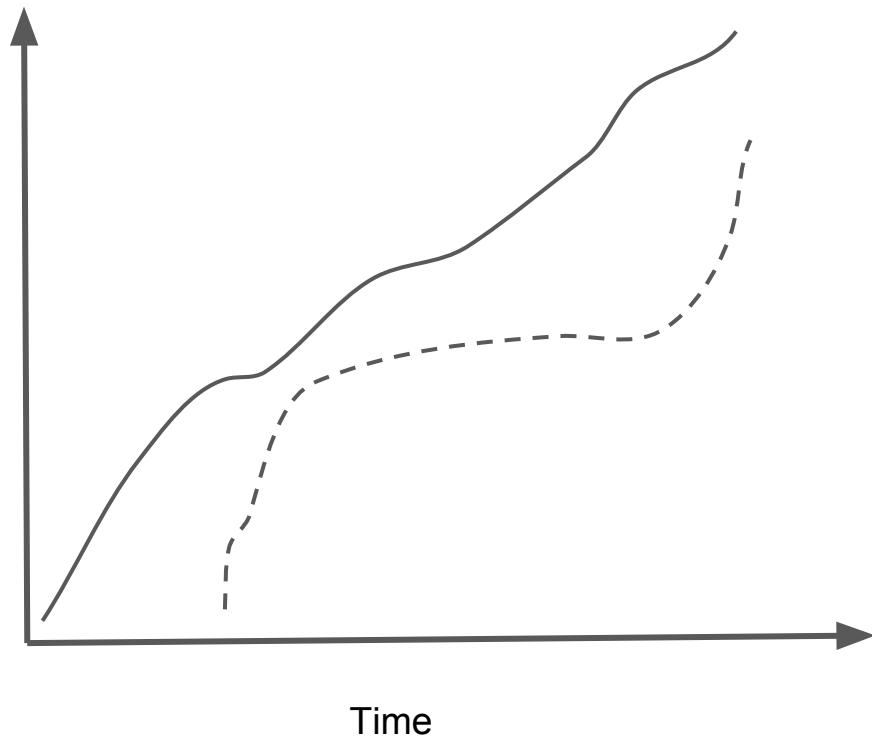
Targets are easy to misuse

Are defect counts a good metric?

Useful for understanding velocity

Can be used to identify problems

Targets are easy to misuse



“Test metrics can be useful if used properly or harmful if used poorly. Understand the metrics and who they might be useful to.”

“Test metrics can be useful if used properly or harmful if used poorly. Understand the metrics and who they might be useful to.”

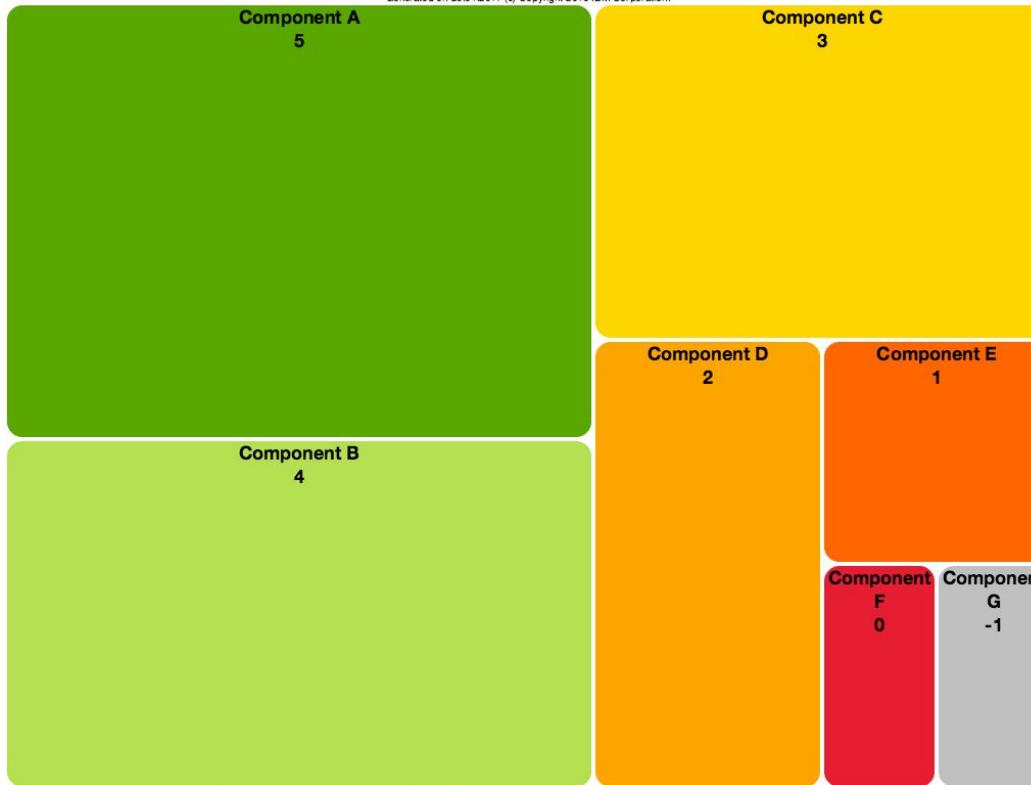
Steve Upton

Jon Tilt
@jontilt

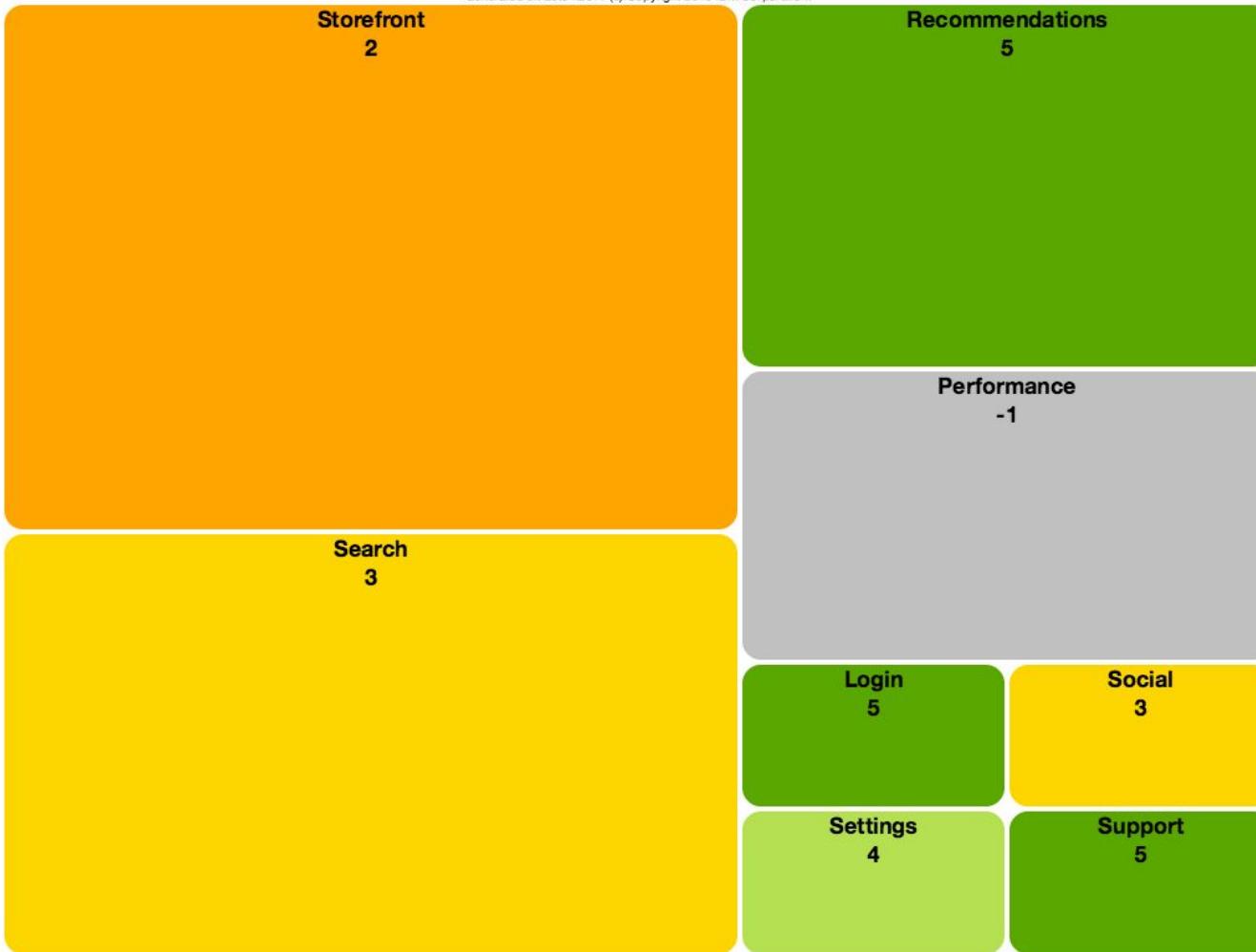


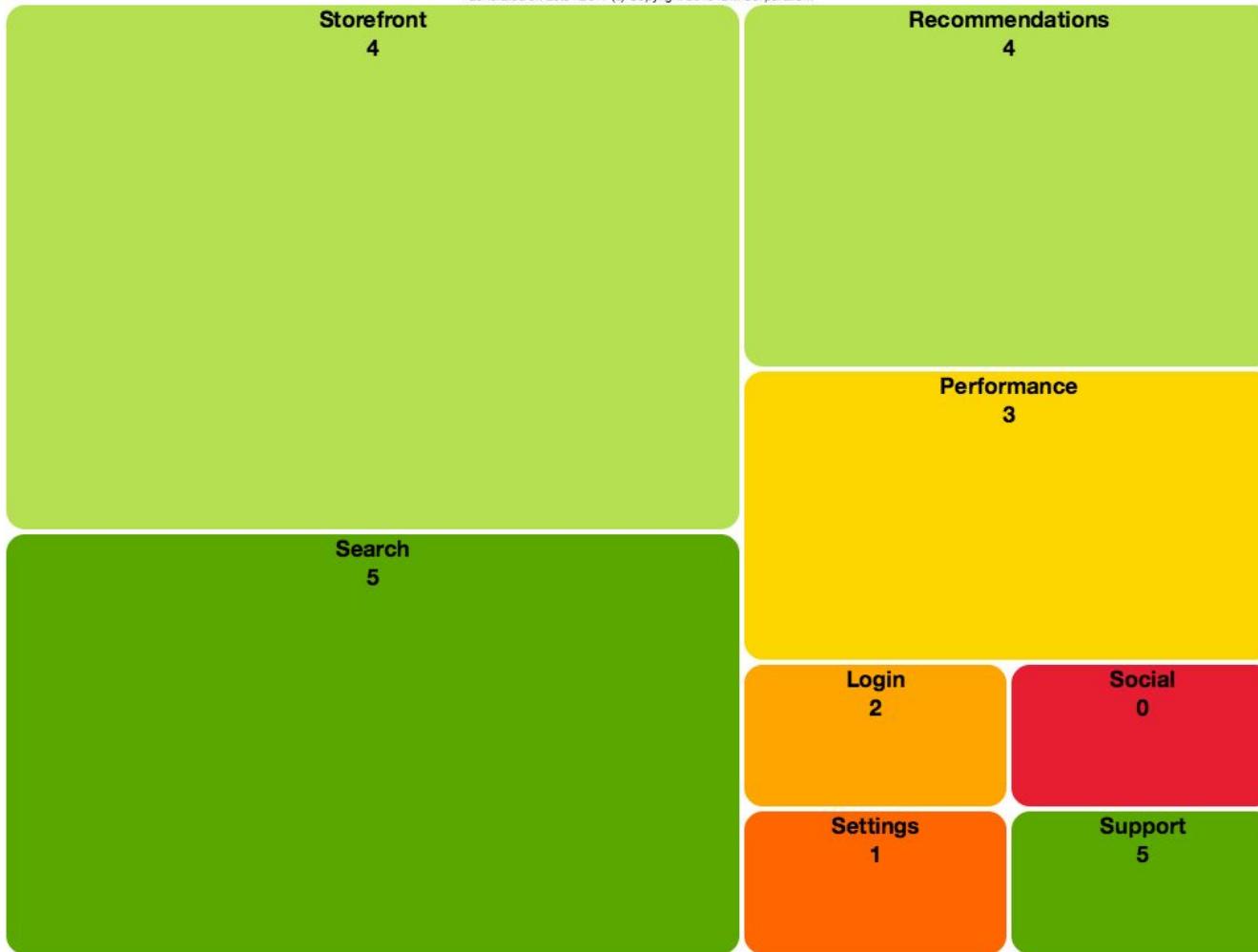
“Accurate and actionable information is more useful than **precise and detailed** information”

Jon Tilt



<http://confidencemap.mybluemix.net/index.html>





Project Coverage summary

Name	Classes	Conditionals	Files	Lines	Methods	Packages
Cobertura Coverage Report	14%  19/136	6%  791/14330	14%  19/136	9%  4048/43216	12%  177/1474	48%  10/21

Coverage Breakdown by Package

Name	Classes	Conditionals	Files	Lines	Methods
isp	10%  1/10	4%  36/982	10%  1/10	8%  238/3057	9%  10/109
isp.administrator	0%  0/5	0%  0/436	0%  0/5	0%  0/1465	0%  0/52
isp.administrator.admin	0%  0/1	0%  0/58	0%  0/1	0%  0/214	0%  0/8
isp.administrator.clinician	0%  0/4	0%  0/540	0%  0/4	0%  0/1755	0%  0/54
isp.administrator.forms	0%  0/14	0%  0/3366	0%  0/14	0%  0/10318	0%  0/360
isp.administrator.methods	0%  0/17	0%  0/468	0%  0/17	0%  0/1392	0%  0/68
isp.administrator.participant	0%  0/8	0%  0/1700	0%  0/8	0%  0/5284	0%  0/136
isp.administrator.researcher	0%  0/3	0%  0/306	0%  0/3	0%  0/1041	0%  0/32
isp.clinician	17%  1/6	11%  88/790	17%  1/6	16%  411/2588	7%  6/82
isp.clinician.forms	0%  0/2	0%  0/326	0%  0/2	0%  0/1027	0%  0/36
isp.clinician.methods	0%  0/18	0%  0/816	0%  0/18	0%  0/1826	0%  0/72
isp.clinician.participant	0%  0/8	0%  0/1186	0%  0/8	0%  0/3911	0%  0/108
isp.components	56%  5/9	39%  390/1008	56%  5/9	65%  1889/2914	74%  80/108
isp.components.downloads	0%  0/5	0%  0/110	0%  0/5	0%  0/317	0%  0/20
isp.components.html	100%  2/2	60%  37/62	100%  2/2	84%  92/110	75%  6/8
isp.components.menuBar	67%  2/3	17%  30/180	67%  2/3	41%  171/422	50%  6/12
isp.methods	40%  2/5	12%  27/218	40%  2/5	26%  144/547	32%  7/22
isp.participant	29%  2/7	11%  140/1268	29%  2/7	19%  756/3883	25%  32/126
isp.participant.methods	20%  1/5	3%  13/376	20%  1/5	7%  42/606	15%  3/20
isp.templates	100%  1/1	23%  6/26	100%  1/1	69%  81/117	78%  7/9
isp.templates.components	67%  2/3	22%  24/108	67%  2/3	53%  224/422	63%  20/32

27

MS

14

SUITES

27

TESTS

9

PASSED

5

FAILED

5

PENDING

5 FAILED HOOKS

8 SKIPPED TESTS

18.5% PENDING

40.9% PASSING

Master Test Suite

/test/test.js

Test Suite - Basic

/test/test.js

⌚ 2 ms ⚡ 2 ✓ 1 ✘ 1 ⚡ 0

Tests

✓ passing test

Show Code | 1 ms

✗ failing test

Show Code | 1 ms

AssertionError: expected false to be truthy

Show Stack

```
false.should.be.ok;  
done();
```



Test Suite - Nested Suites

/test/test.js

⌚ 0 ms ⚡ 2 ✓ 1 ✘ 1 ⚡ 0

Tests

✓ passing test

Show Code | 0 ms

✗ failing test

Show Code | 0 ms

AssertionError: expected false to be truthy

Show Stack



Nested Test Suite

/test/test.js

⌚ 0 ms ⚡ 1 ✓ 1 ✘ 0 ⚡ 0

Tests

✓ passing test

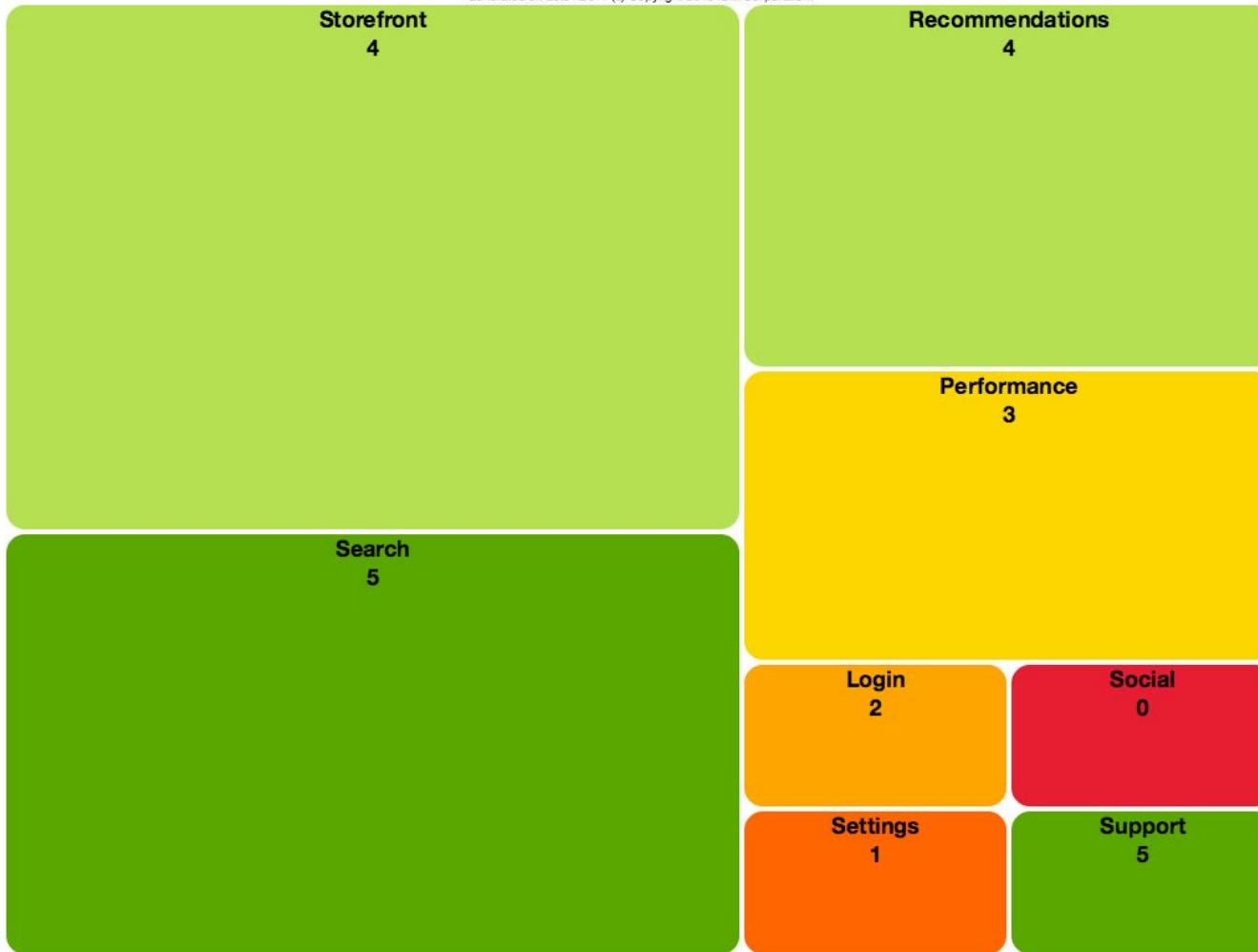
Show Code | 0 ms

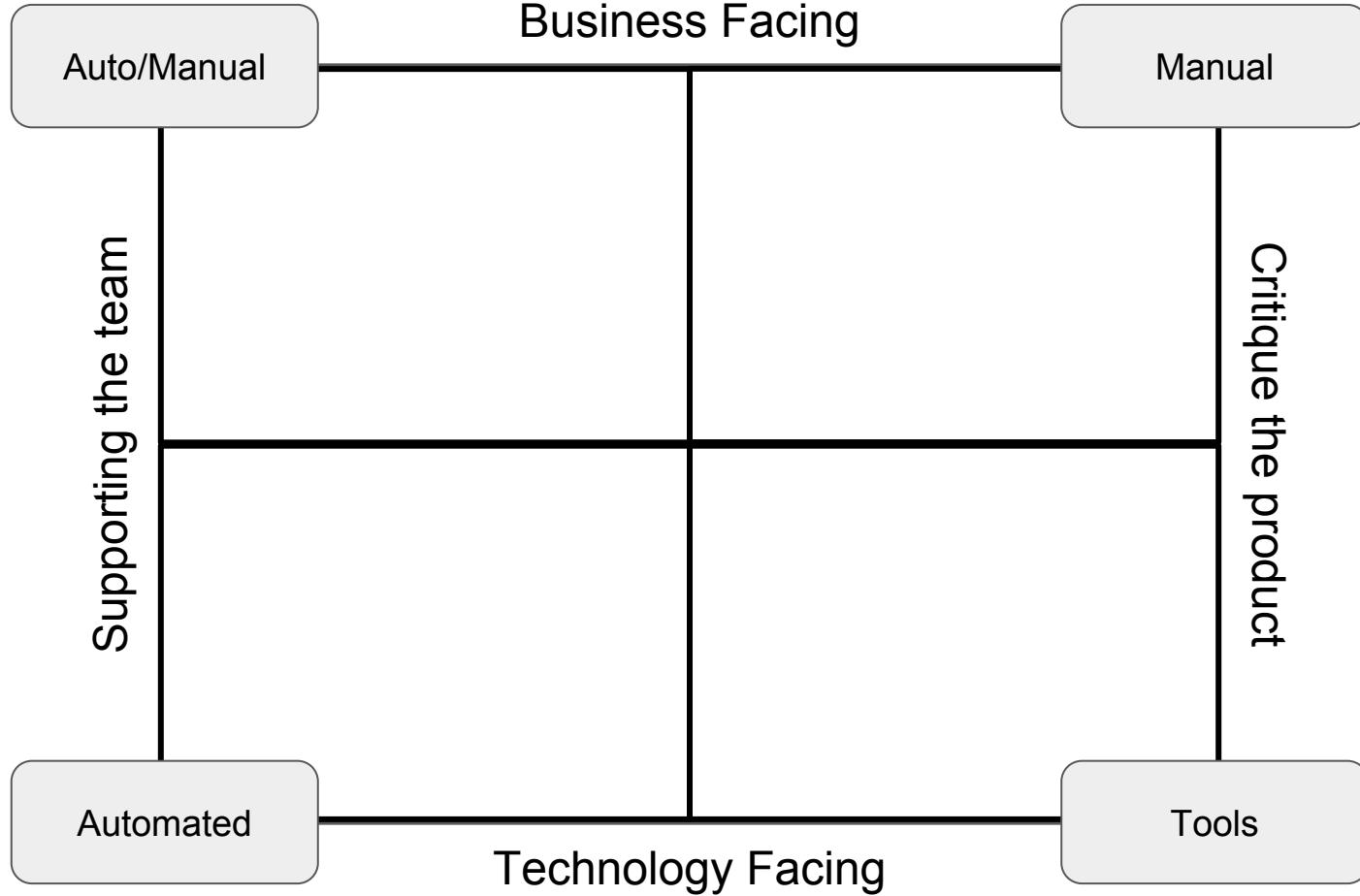


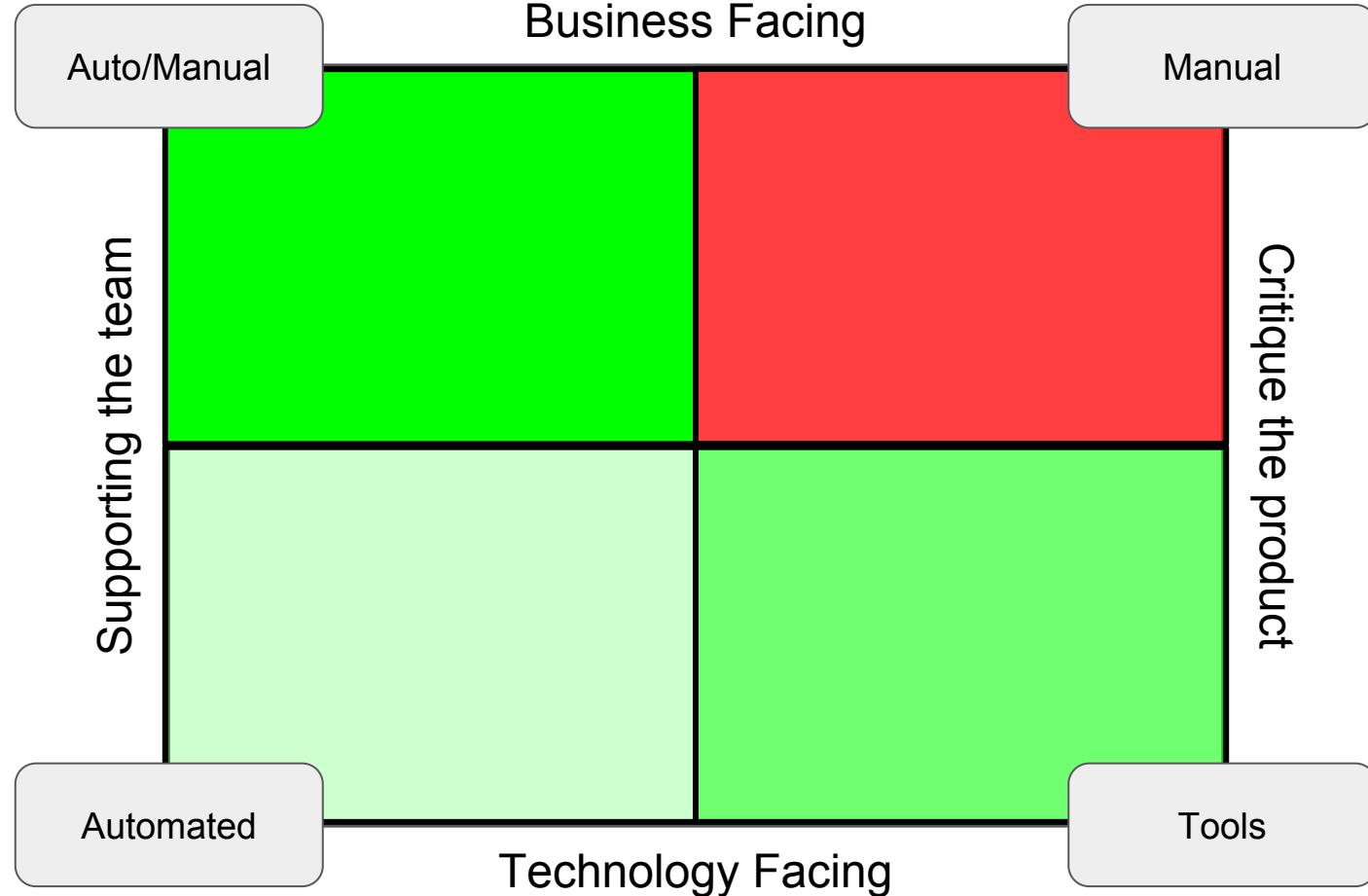
Nested Test Suite

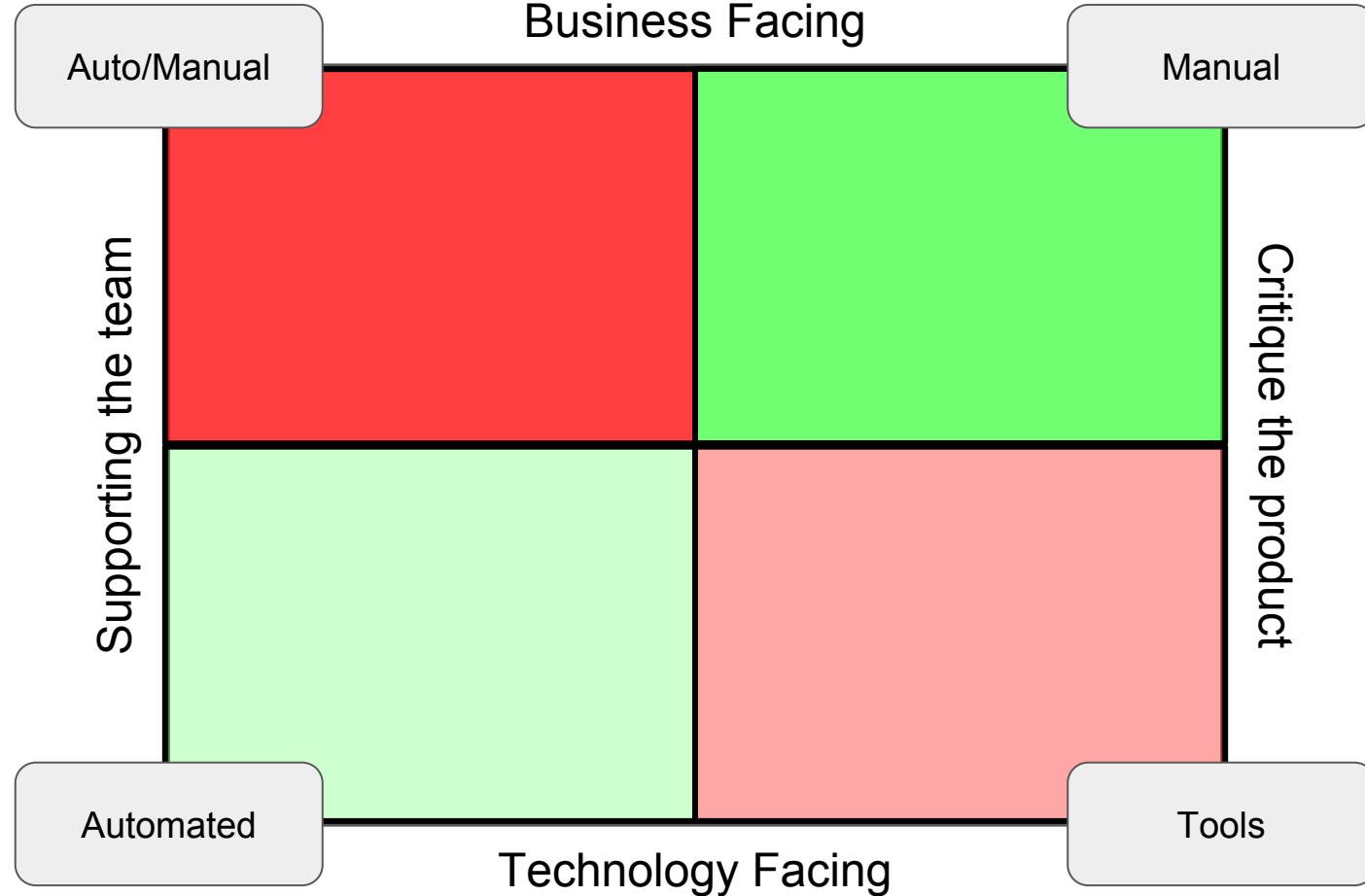
/test/test.js



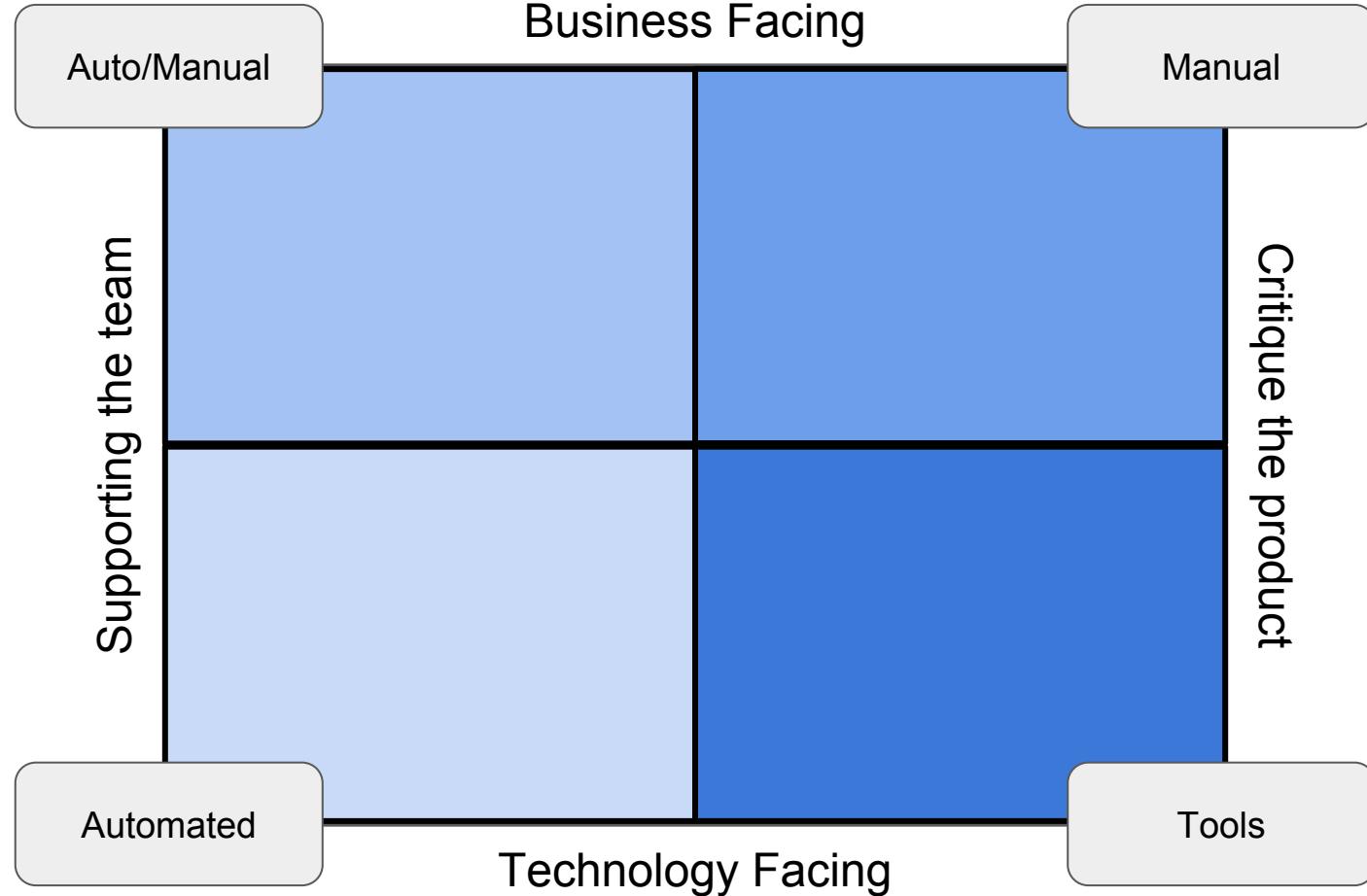


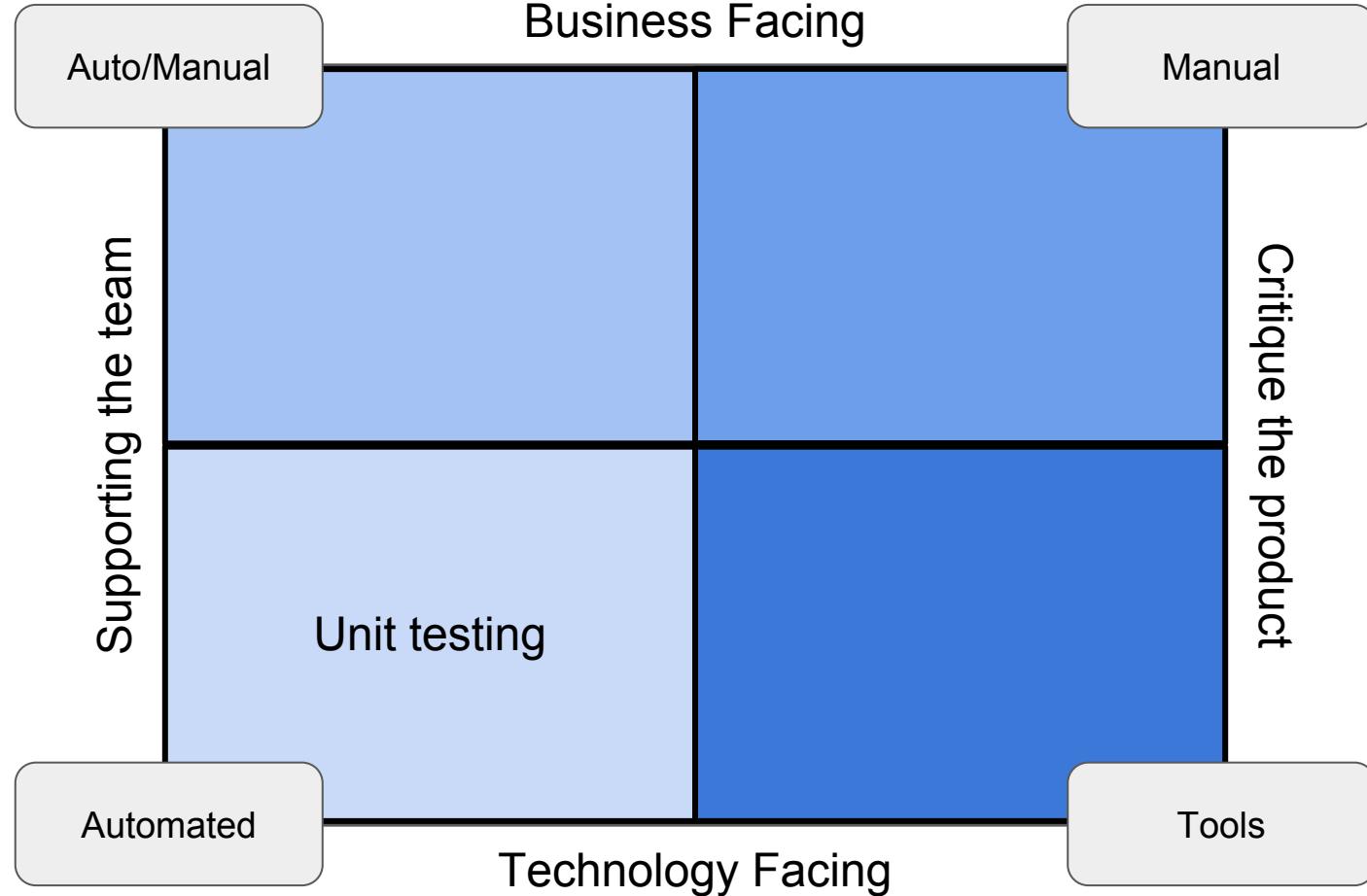






Testing for Data Science





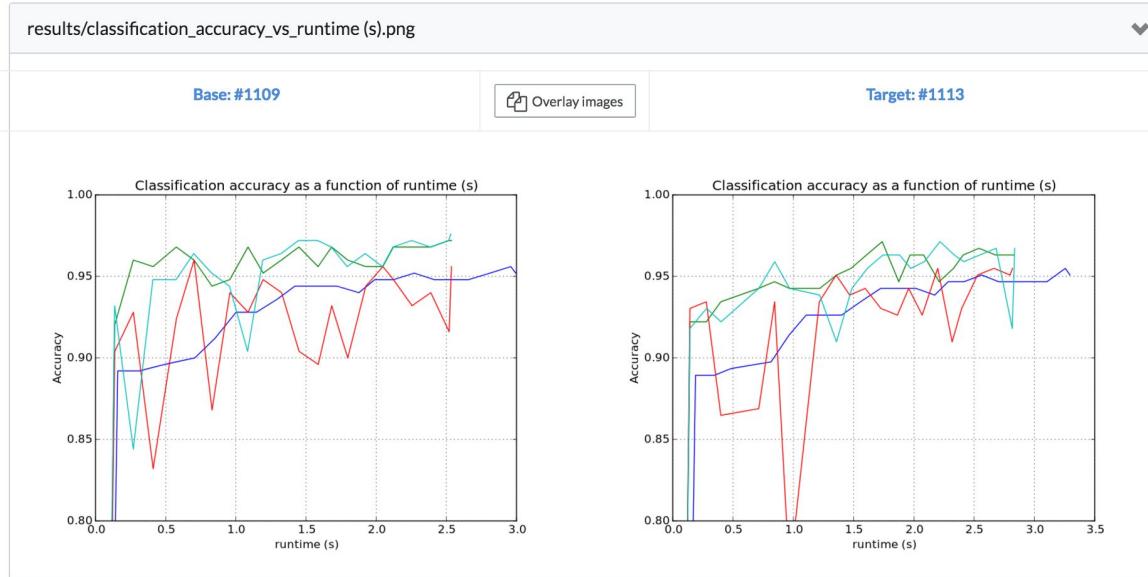
Comparing Run #1109 ...Run #1113

7 result files changed

Diagnostic statistics changed

Base		Target	
auc	0.791	auc	0.613
p-value	0.965	p-value	0.265

Result files changed



If a model looks too good to be true, then generally it is

Graham Williams

Data Mining with Rattle and R

The Art of Excavating Data
for Knowledge Discovery



Chapter 15: Model Performance Evaluation

Cross-validation

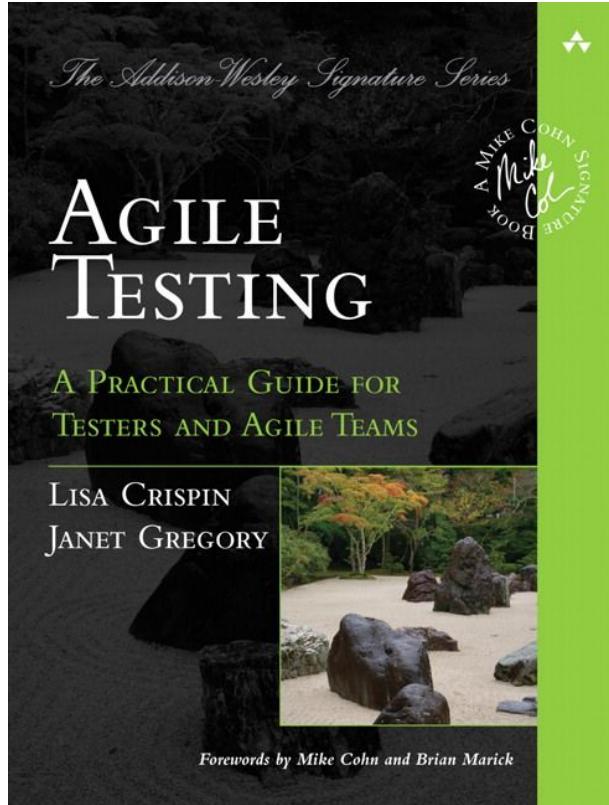
Error Rate

Questions?

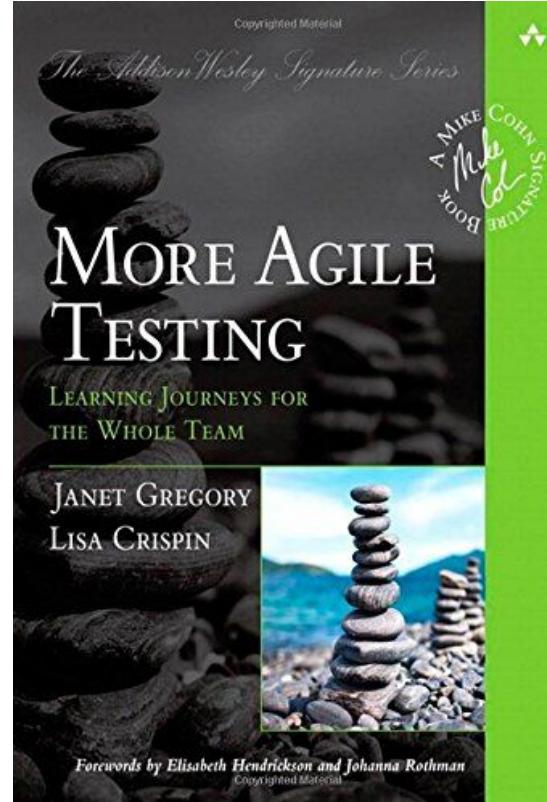
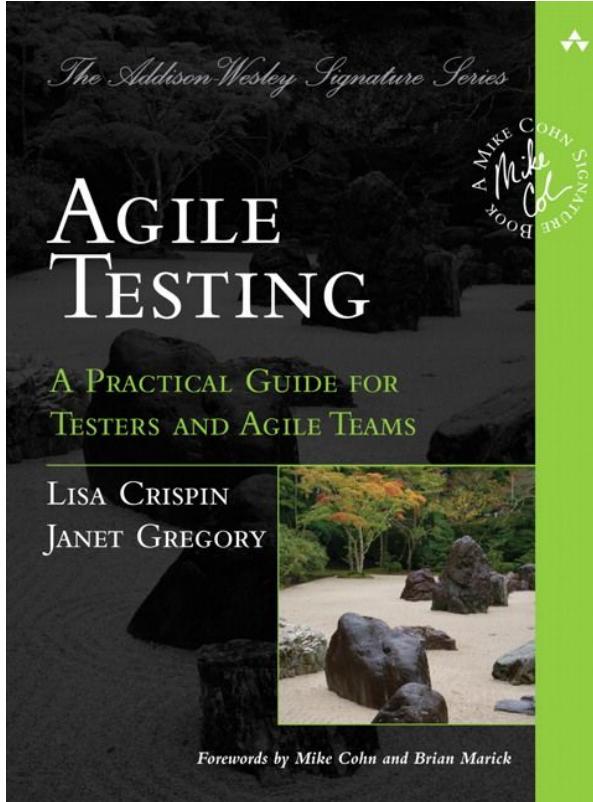
@Steve_Upton
steveupton.io

This work is licensed under a [Creative Commons Attribution 4.0 International License](#).

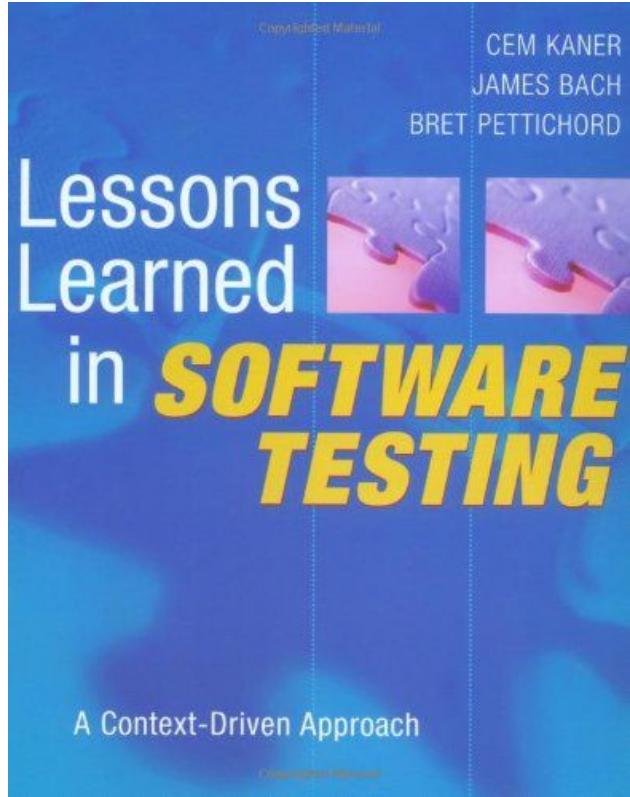
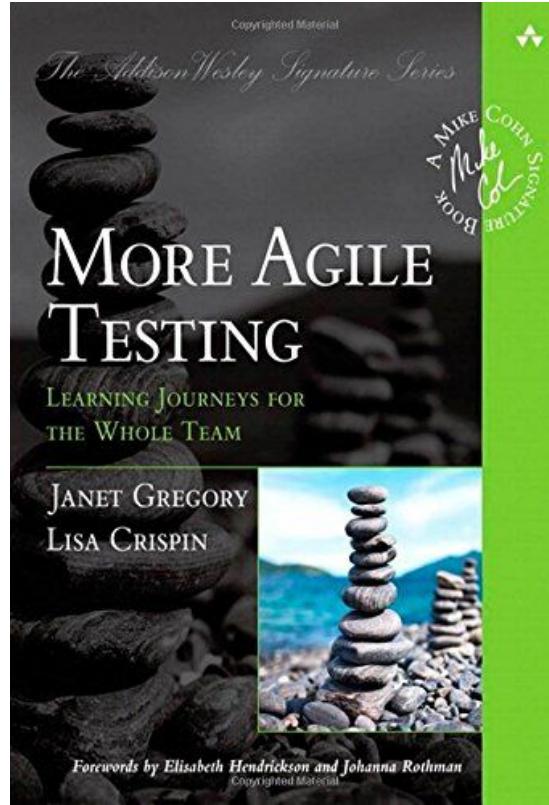
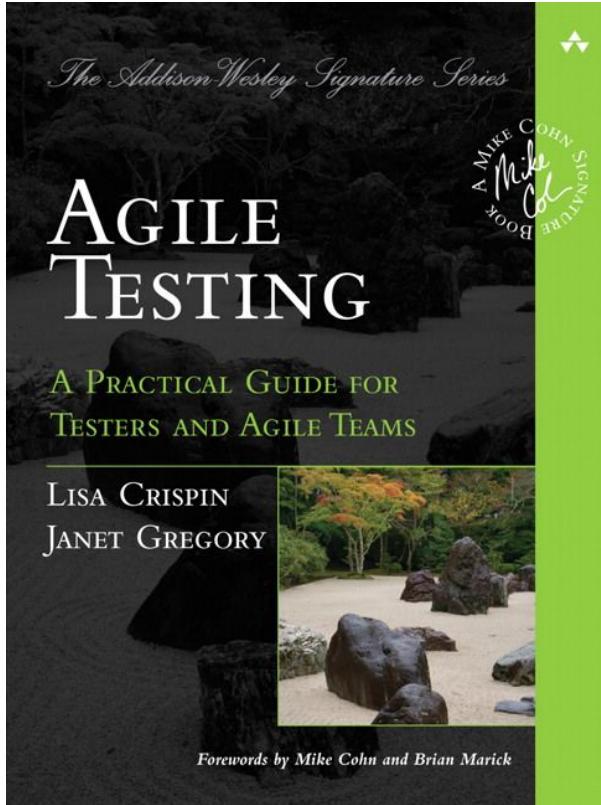
Books



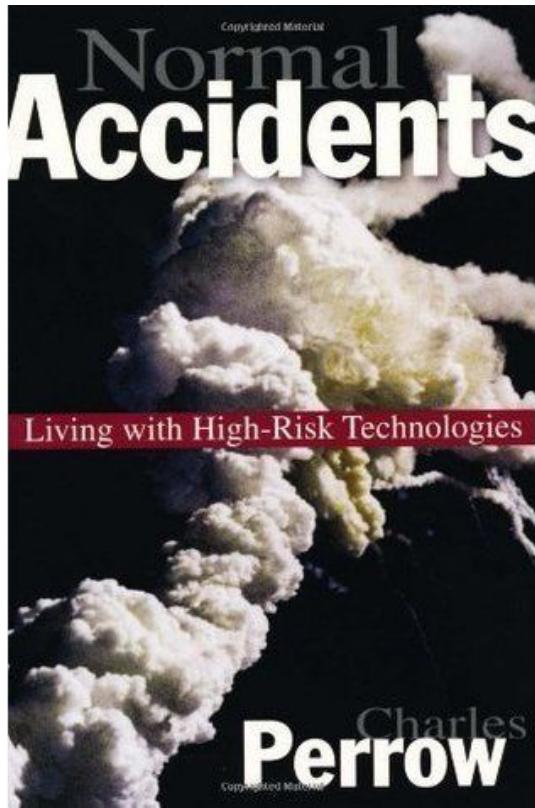
Books



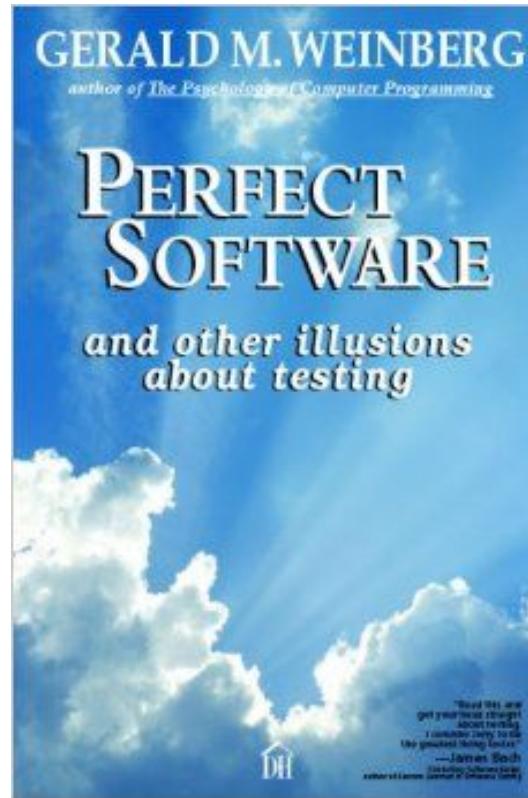
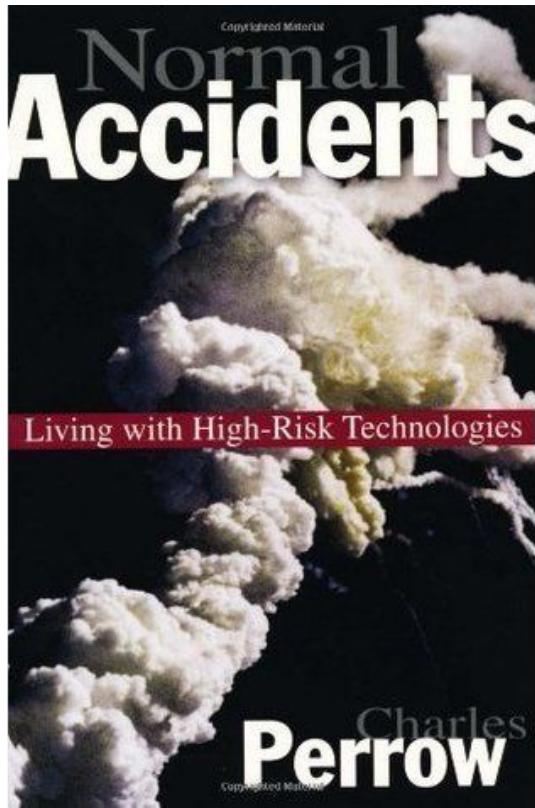
Books



Books



Books



Essential reading

<https://dannorth.net/introducing-bdd/>

Good reading

<http://www.example.com/testing-com/writings/coverage.pdf>

http://www.satisfice.com/articles/what_is_et.shtml

<http://www.predictiveanalyticsworld.com/patimes/four-ways-data-science-goes-wrong-and-how-test-driven-data-analysis-can-help/6947/>

<https://inviqa.com/blog/bdd-guide>

References

<http://sunnyday.mit.edu/accidents/Ariane5accidentreport.html>

<http://www.intel.de/content/dam/www/public/us/en/documents/specification-updates/desktop-6th-gen-core-family-spec-update.pdf>

<http://lisacrispin.com/2011/11/08/using-the-agile-testing-quadrants/>

<http://www.exampler.com/old-blog/2003/08/21.1.html#agile-testing-project-1>

<http://www.gao.gov/assets/220/215614.pdf>

<http://www.mysmu.edu/faculty/davidlo/papers/saner15-coverage.pdf>

<https://github.com/cucumber/cucumber-ruby/commit/a9398c13d5a53b71e582050de92ae49e3524412c>

References cont.

<https://shkspr.mobi/blog/2014/03/introducing-corkr-at-nhtg14/>

<https://shkspr.mobi/blog/copyright-terence-eden/>

<http://orlando.opensauce.it/corkr/>

Code examples

<https://technobeans.com/2012/04/16/5-ways-of-fibonacci-in-python/>

<https://wiki.python.org/moin/SimplePrograms>

Bad things

<http://stackoverflow.com/a/22291032/1214161>

<http://testingbasicinterviewquestions.blogspot.de/2015/03/best-difference-between-alpha-and-beta.html>

Image credits

US Launch Report

Extra material