Rethinking Testing

towards your needs

cubox

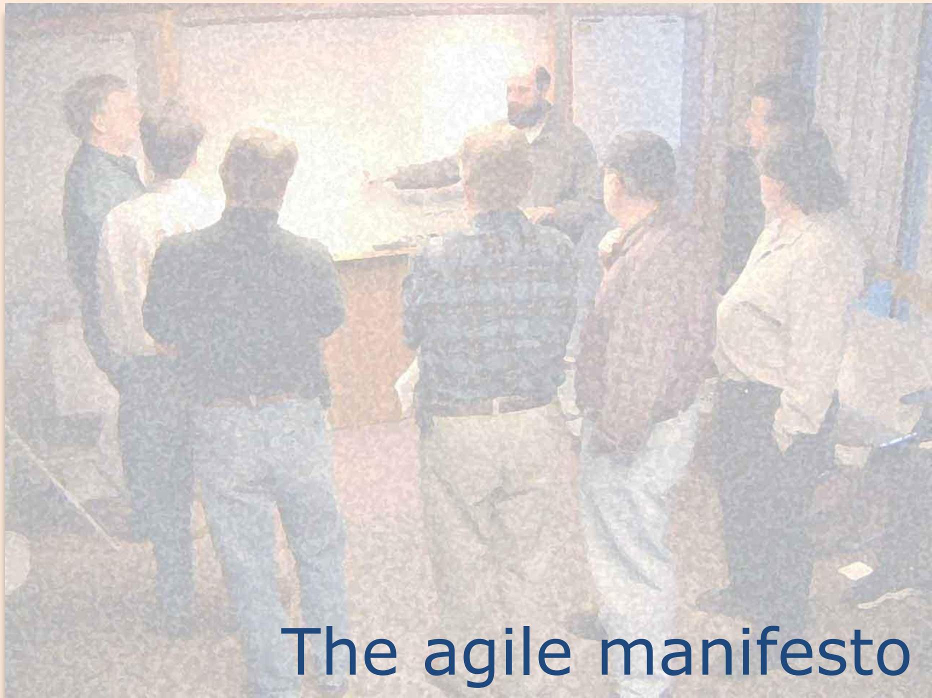
pablo@cuboxsa.com

ABN-AMRO Bank

9

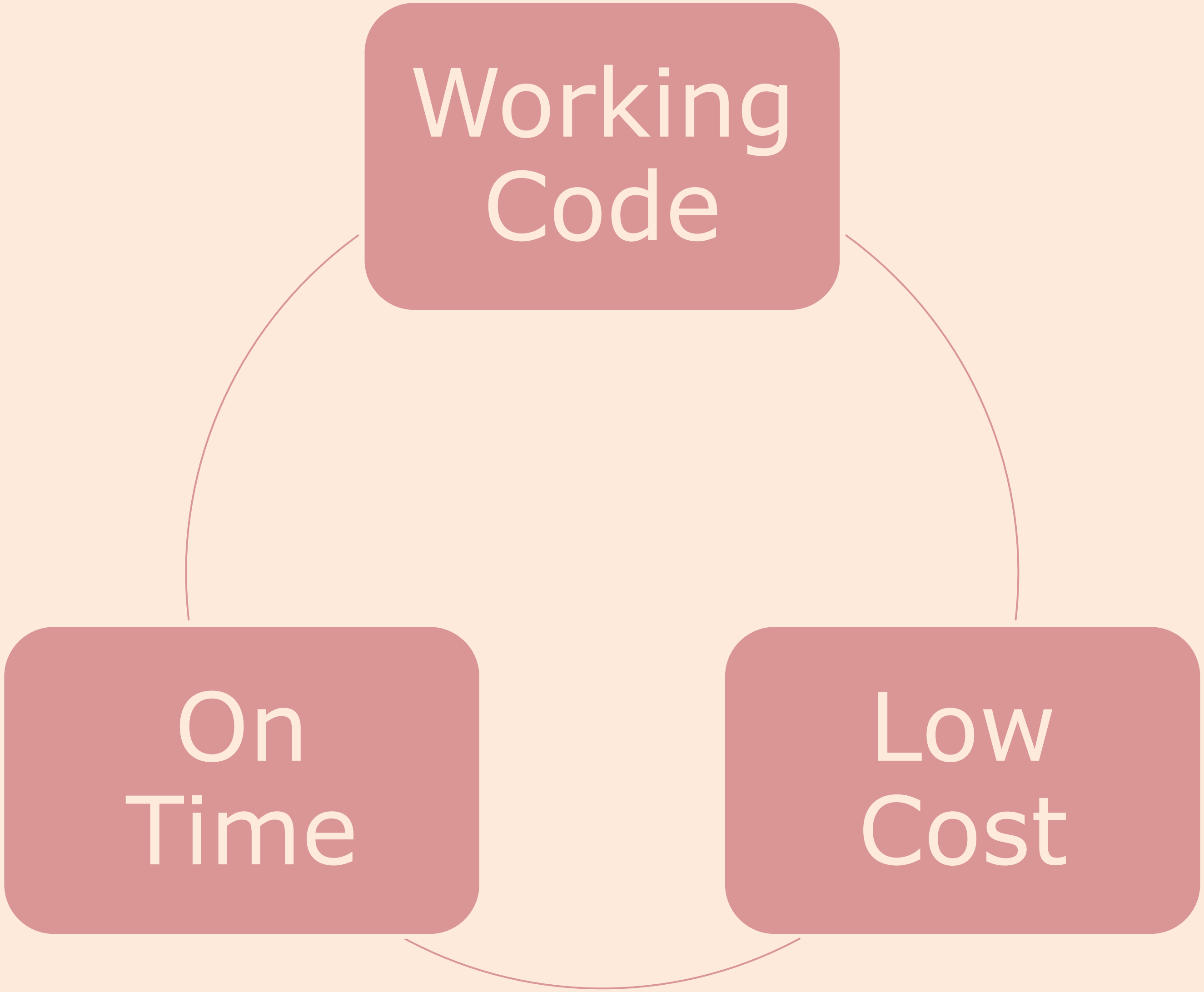
Agile Software Development
eXtreme Programming
Risk Management
Gerbil
Rspec
Functional
Testing Process
Exploratory Test
ISTQB Foundat
Continuous In
Crash Test
System Testing
Unit Testi
Software Lifecycle
figuration Testing
Integratio Test
Stress
Cucumb
Performan
Regression Testing
SCRU
QU
BehaviorD
Driven Developm
Design By Contrac
AgileP
WhiteB
Deliv
Box

"Our highest priority is to satisfy the customer through the early and continuous DELIVERY of VALUABLE software."



The agile manifesto

The Customer



Working Code

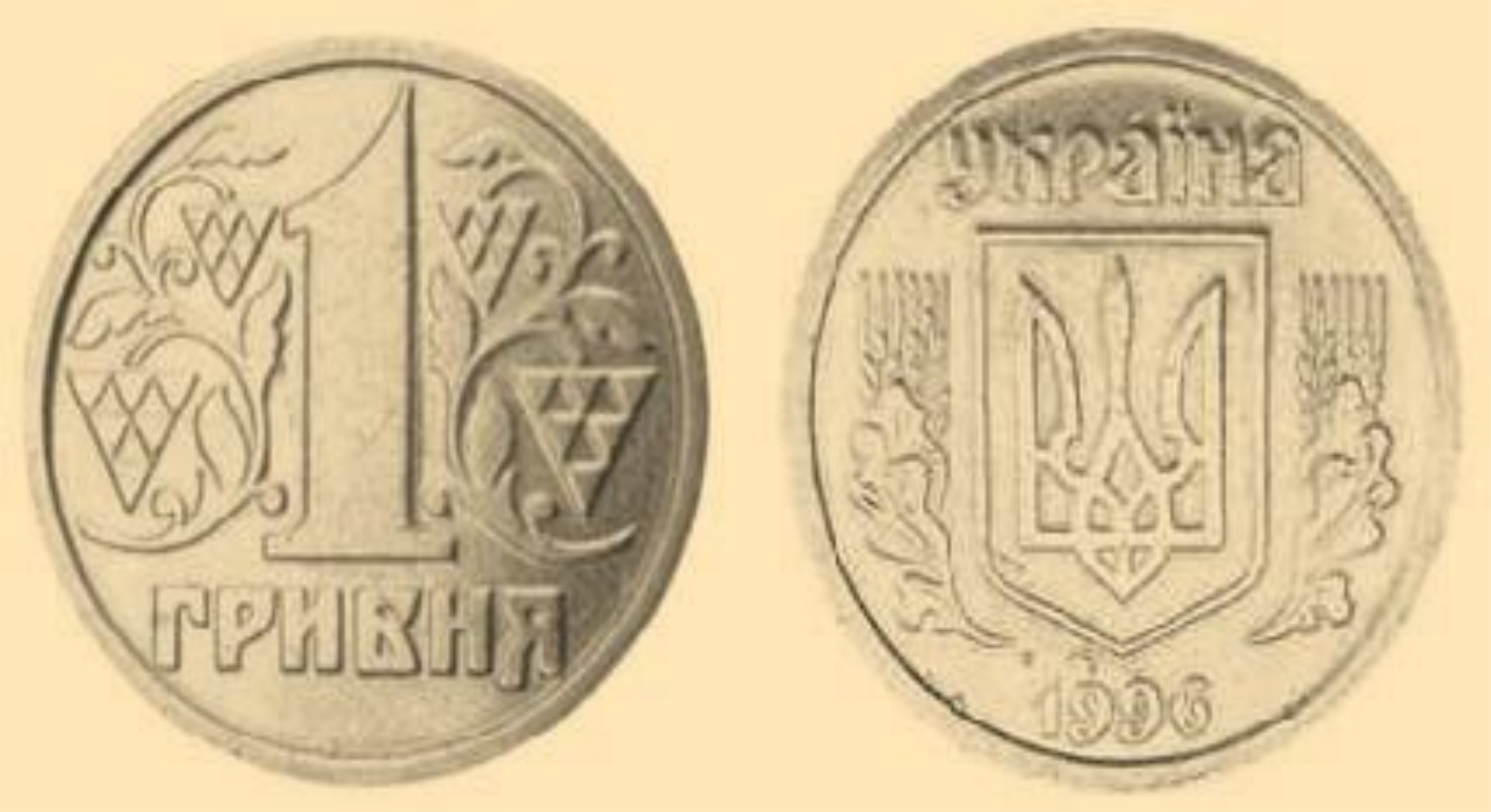
On Time

Low Cost

The Needs of Customer

Working Code

Acceptance Testing



Public Domain Wikimedia Commons

Design by Contract

- by B. Meyer
- Code enforces a contract
- Language provides constraints
  - Before function call (Preconditions)
  - After function call (Post Conditions)
  - During all the function call (Invariants)

User.create! params[:user]

Pre conditions:

✓ params[:user] is not empty.

Post conditions:

✓New instance returned is successful.

✓Raise exception w/errors otherwise.

Invariants:

✓ params[:user] is not modifiable.

✓Database connection available.

Design by Contract

- Few implementations:
  - Native in Eiffel
  - As add-on like:
    - Java “assert”
    - .Net code contracts
- In Ruby
  - A.Hunt created DBC.rb (~2000)
  - Lost in translation

Code that verifies Code

- Focus on the valuable outputs

Automated Testing

jUnit

- by Gamma & Beck since 1997
- by the time Java was “cool”
- Isolated Components
- TestCases & TestSuites

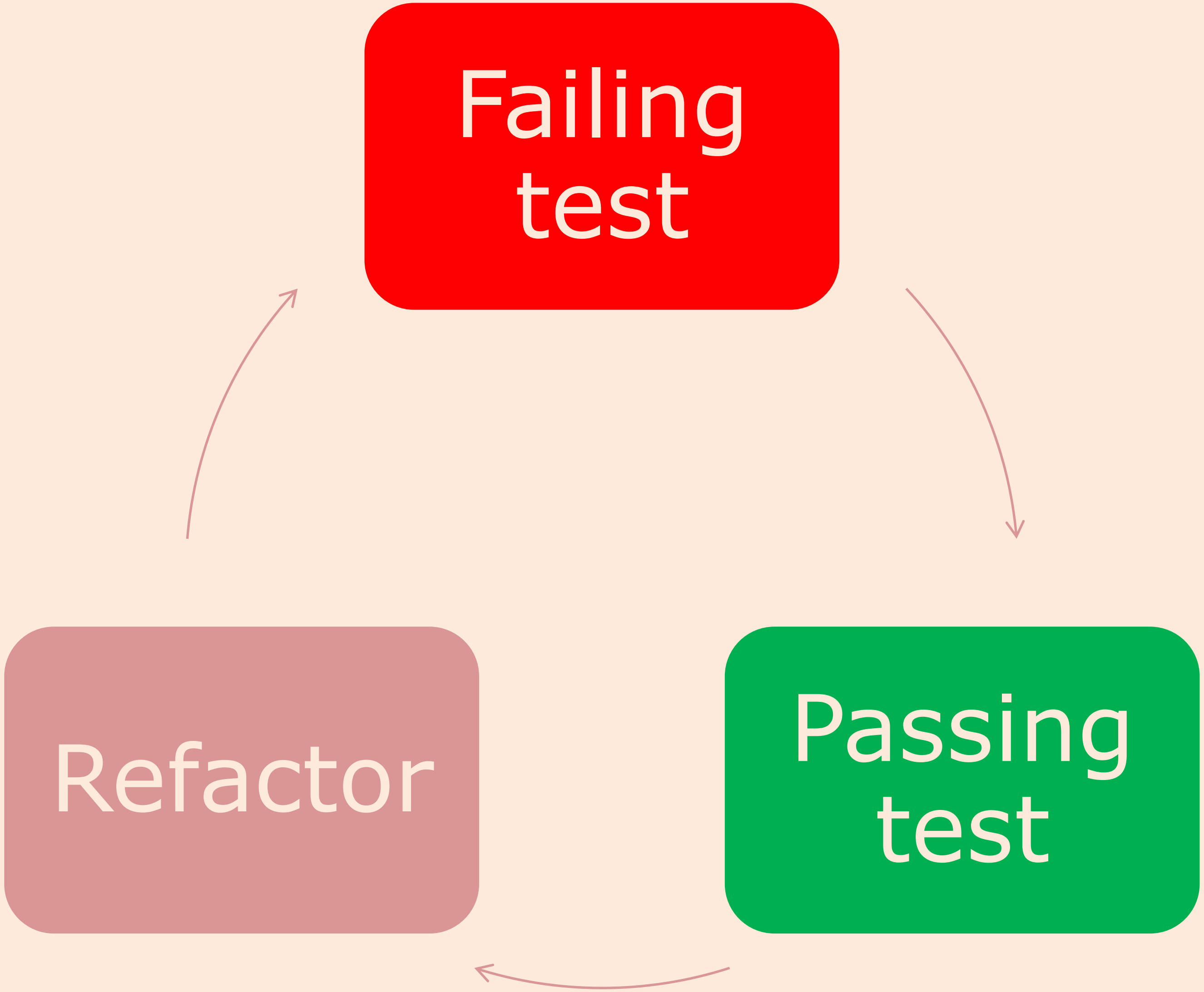
Test Driven Development (TDD)

- by Beck.
- Test First Design and then program.
- Small requirements.
- Verify functionality w/automated tests.

“the programmer should let correctness proof and program grow hand in hand”

E.Dijsktra 1972

The TDD Cycle

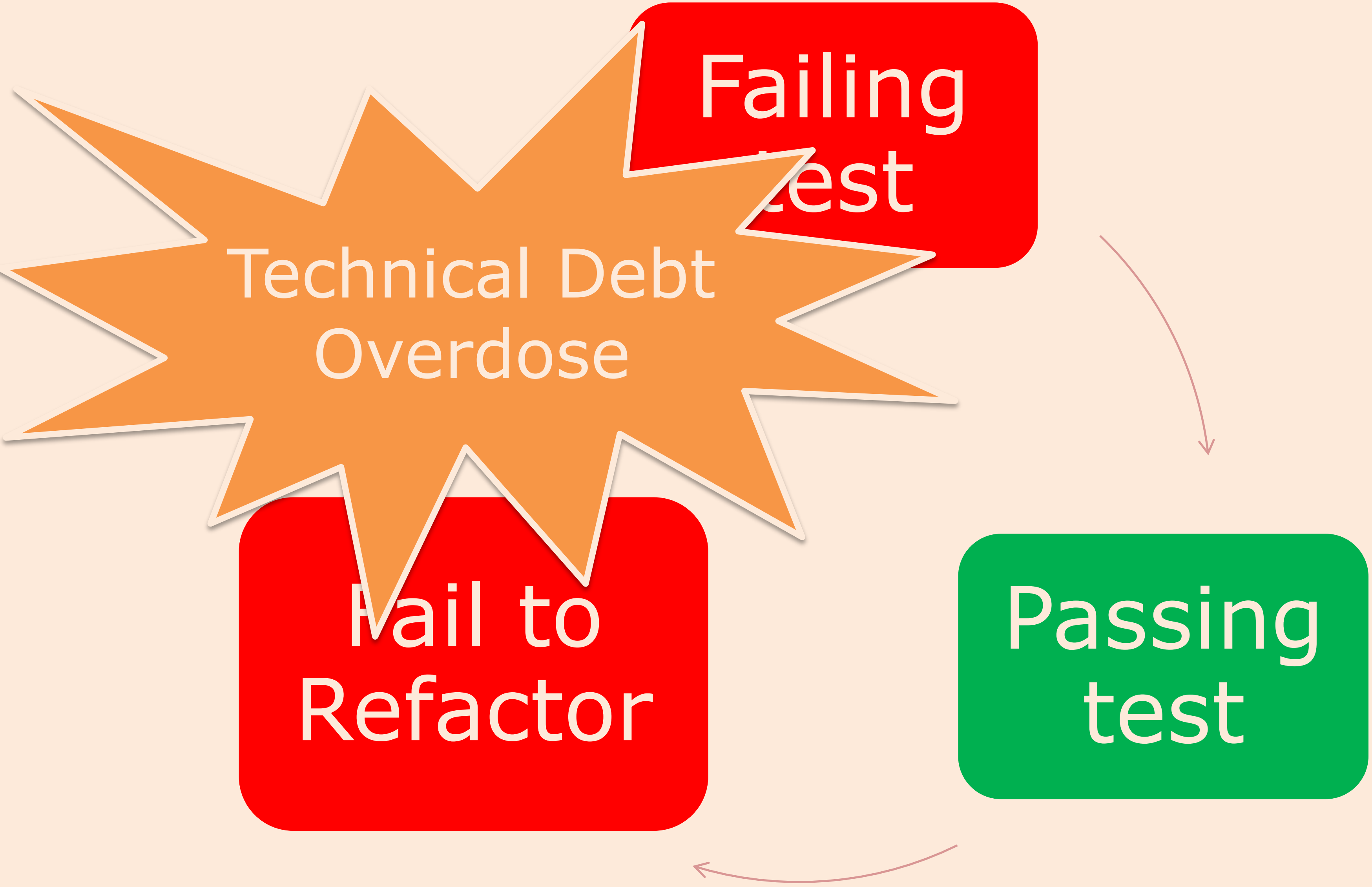


Failing test

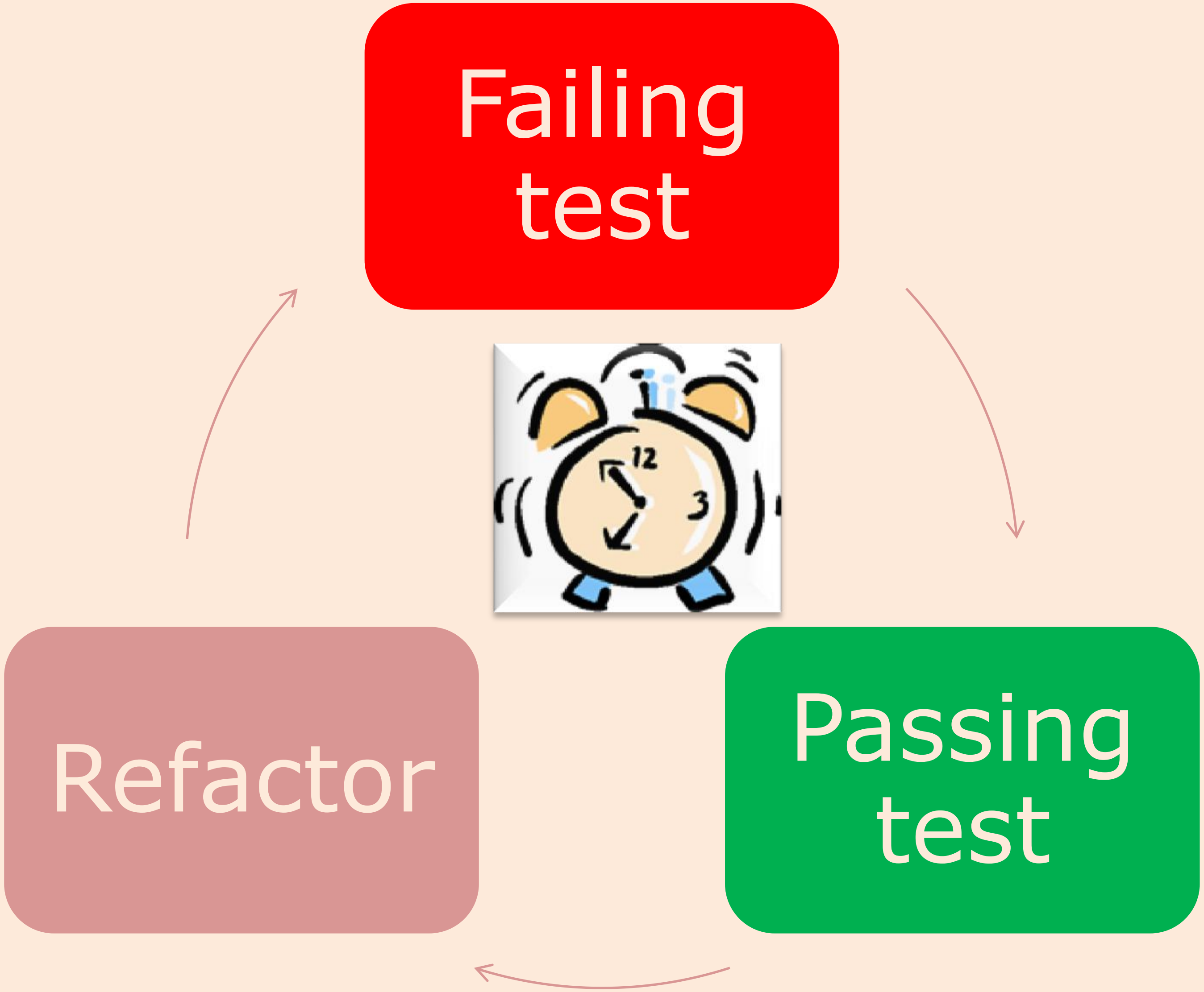
Passing test

Refactor

No Refactor


Failing test
Passing test
Fail to Refactor
Technical Debt Overdose

Running Suite takes too long



Failing test

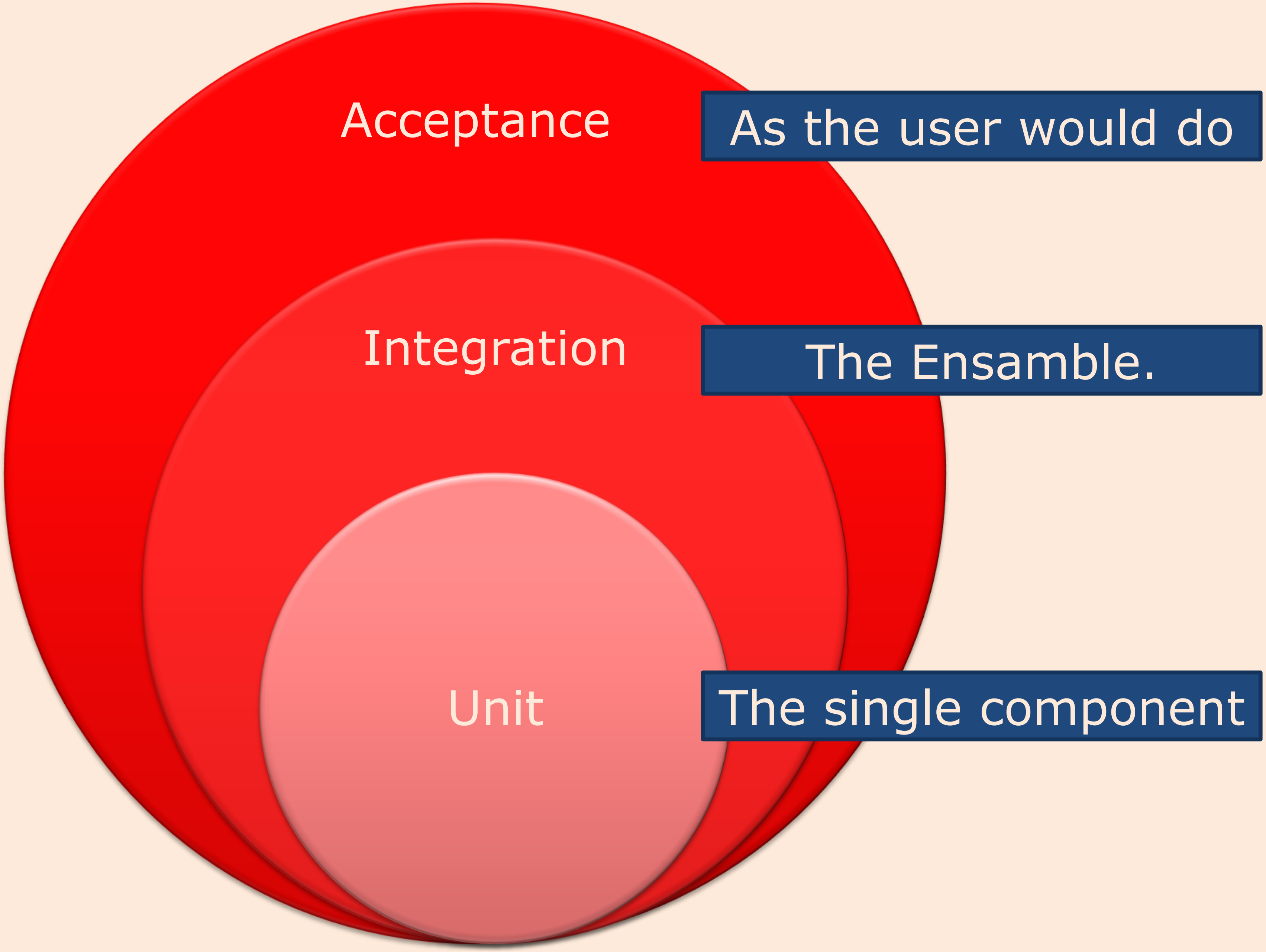
Passing test

Refactor

“correctness concerns turn out to be a very effective heuristic guidance”

E.Dijsktra 1972

Testing correctness - scales



Восточные специи
ЧИЛИ ОСТРЫЙ
для: первых и вторых блюд
ИМБИРЬ (молотый)
для: маринованных и квашенных овощей выпечки кондитерских изделий чая
ШАФРАН
для: плова супа рыбы кондитерских изделий
АДЖИКА
для: мяса супа салатов рагу
РЕГАН
МАЙОРАН
ДЛЯ плова
ДЛЯ рыбных блюд
ДЛЯ кур, окорочков, птицы, индюшатины
ДЛЯ подливок, отбивных, чинахи, гуляша
МУСКАТ
ДЛЯ ухи
САМБАР (острый)
ДЛЯ баранины
ПЕРЕЦ (душистый)
ДЛЯ супа, борща, харчо, шурпы
ПЕРЕЦ БЕЛ

By Jewgienij Bal, Wikimedia Commons

Plenty of Tools

Focus on Ruby

- Test-Unit
- RSpec
- Cucumber

Focus on Javascript

- QUnit
- Jasmine
- Gerbil

Plenty of Tools

Focus on Ruby

- Test-Unit
- RSpec
- Cucumber

Focus on Javascript

- QUnit
- Jasmine
- Gerbil

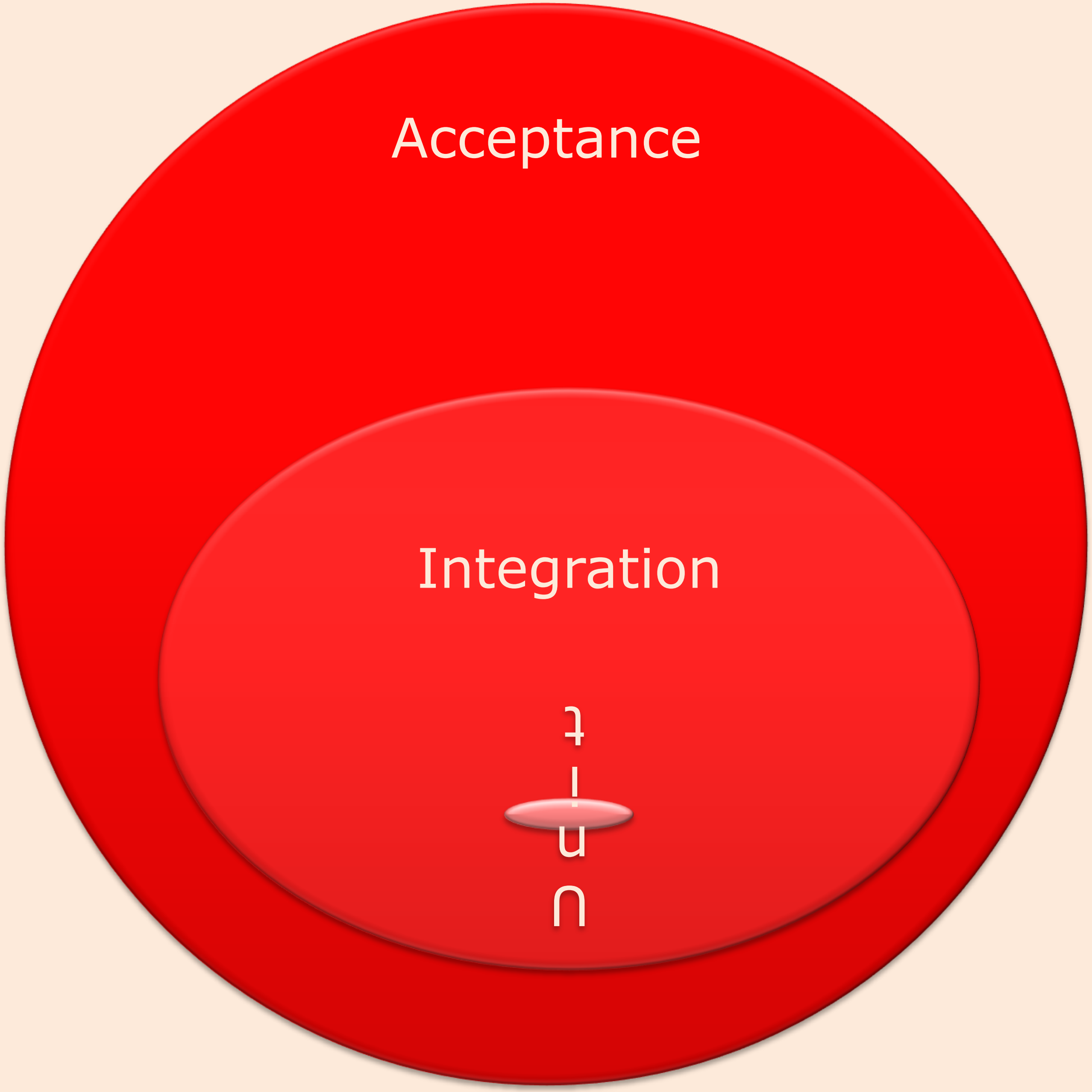
RSpec

- Rails ready out-of-the-box
- Acceptance testing:
  - Selenium driver
  - Capybara
  - `spec/integration`

What do we test?

- Rails app:
  - Models
  - Controllers
  - Views
- Additionally:
  - Services
  - Components

Typical setup on Rails



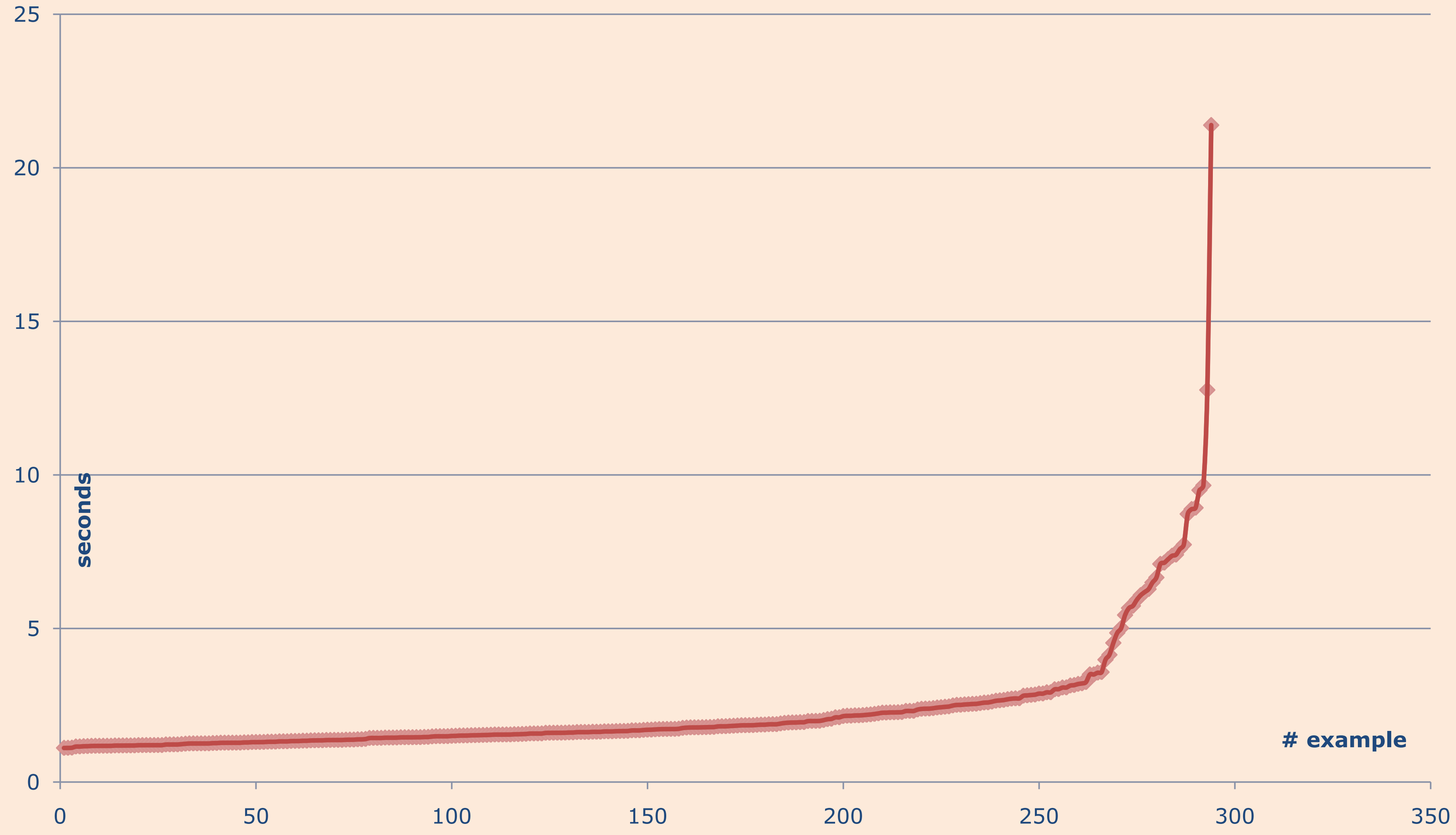
.rspec

```
--colour
--format Fuubar
--profile
```

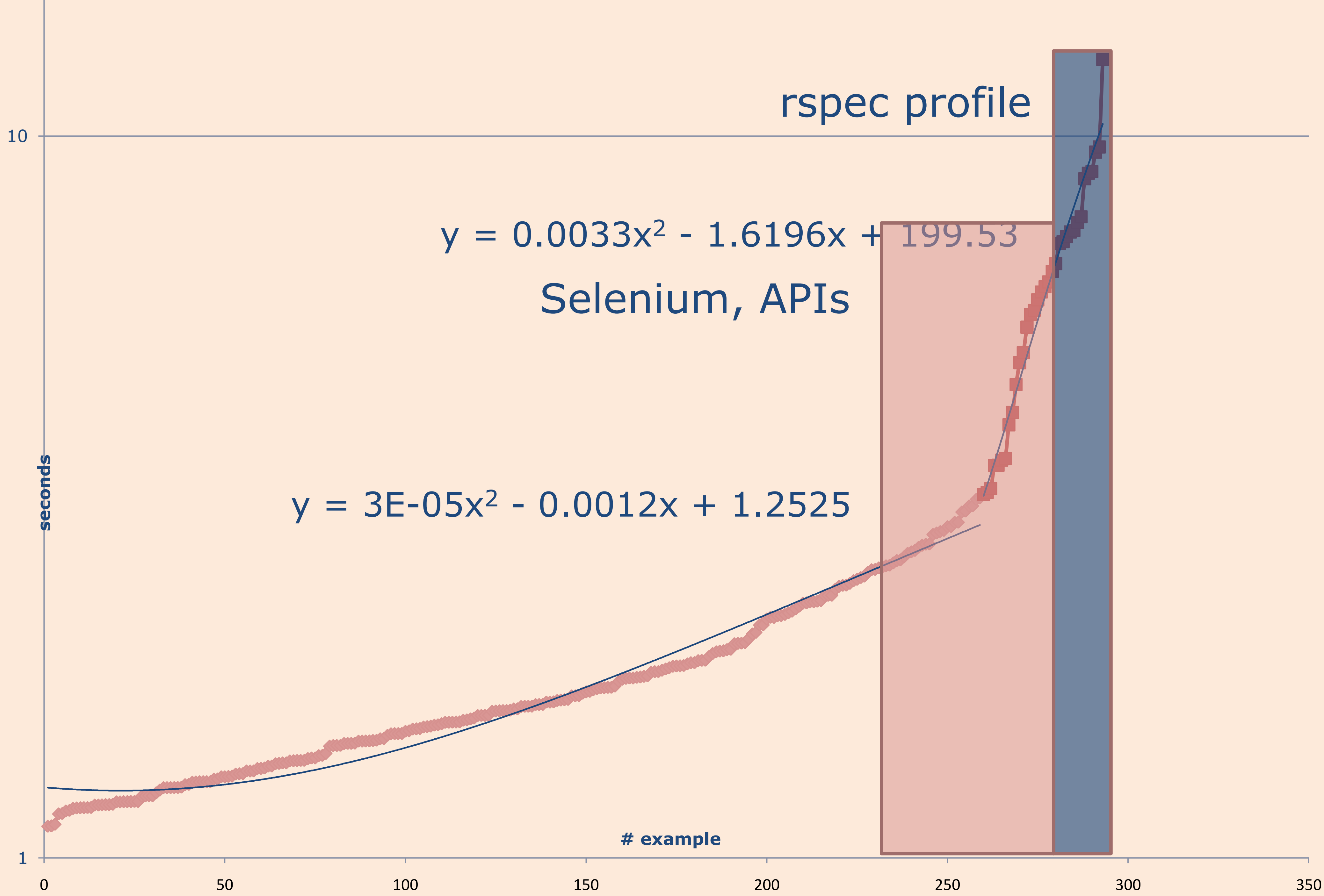
Additional gems:

- ✓Autotest
- ✓Specjour / Spork

Average runtime for test-suite for a small project (294 examples)



Two different sets of tests



rspec profile

$y = 0.0033x^2 - 1.6196x + 199.53$

Selenium, APIs

$y = 3E-05x^2 - 0.0012x + 1.2525$

Models && Persistence

- Object State & Transitions
- Custom methods and edge conditions
- Associations w/other objects
- Additional Validations
  - Expected fields.
  - Factories.

Controllers && Views

- Controllers:
  - Don’t test them, do acceptance tests.
  - Logic and state reside at Model level.
- Controllers that are APIs:
  - Don´t test them, do integration tests.
- Views
  - Do Acceptance testing for all actions available (not just CRUD).

Components

- Black Box Testing:
  - Test parameters and results.
  - Test correct invoking:
    - Send Email: ActionMailer parameters.
    - Draw a chart: HighChart parameters.

Services

- Test connectivity to services.
- Mocked for Integration Tests.
  - Save responses in fixtures.
  - Assume correct/wrong responses.

Javascript: Selenium driver

- Does a lot of things:
  - Creates an instance of browser.
  - Establishes connections.
  - Executes requests, renders responses.
  - Executes Javascript.
- Emulates the user experience
  - Browser “bugs”

Slow test suite

- Use a CI Server
  - Postpone long tests.
- Avoid Rails boot time
  - Decouple logic from Rails into POROs.
  - Use a no_rails_spec_helper.
- Avoid API calls lag
  - Mock external services.

By Nicolas M. Perrault , Wikimedia Commons

Claude Monet 77

Questions?

спасибі

cubox

¡MUCHAS GRACIAS!

References

- The Rspec Book, D. Chelimsky et al.
- TDD by Example, K. Beck
- Continuous Delivery, J.Humble
- The Pragmatic Programmer, A. Hunt
- The Humble Programmer, E. Dijkstra http://www.cs.utexas.edu/~EWD/transcriptions/EWD03xx/EWD340.html

Is Cucumber a need?

- Rspec is good enough.
- “Simplicity –
  the art of maximizing the
  amount of work not done –
  is essential.”
- Specs are easy to program than specs + steps.
- Customer Interaction vs tools.

Stress && Performace

- Staging environment.
- Monitor execution, APM:
  - New Relic
  - Air Brake

"…program testing can be a very effective way to show the presence of bugs, but [it] is hopelessly inadequate for showing their absence."

E.Dijsktra 1972