Design in Test Driven Development Methods

A Survey

Reuven Yagel
Software Engineering Department,
Azrieli - The Jerusalem College of Engineering, Israel
robi@jce.ac.il

AISTQ, 19 May, 2016 – Beer-Sheva
http://robi-y.github.io/presentations

Agenda

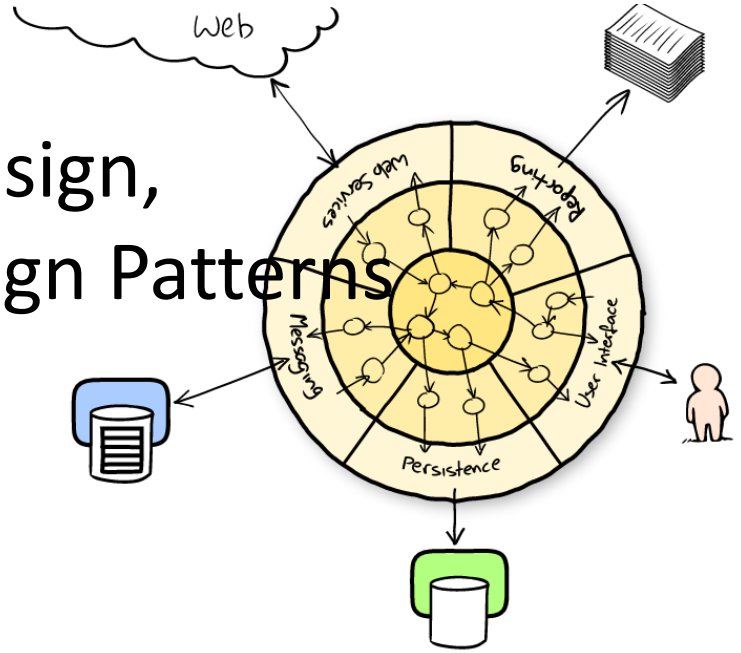
- Testing and xDD Family
- Design and Testing
- TDD / BDD: Pros & Cons
- ~~Demo~~
- Discussion / Conclusions

Testing Today

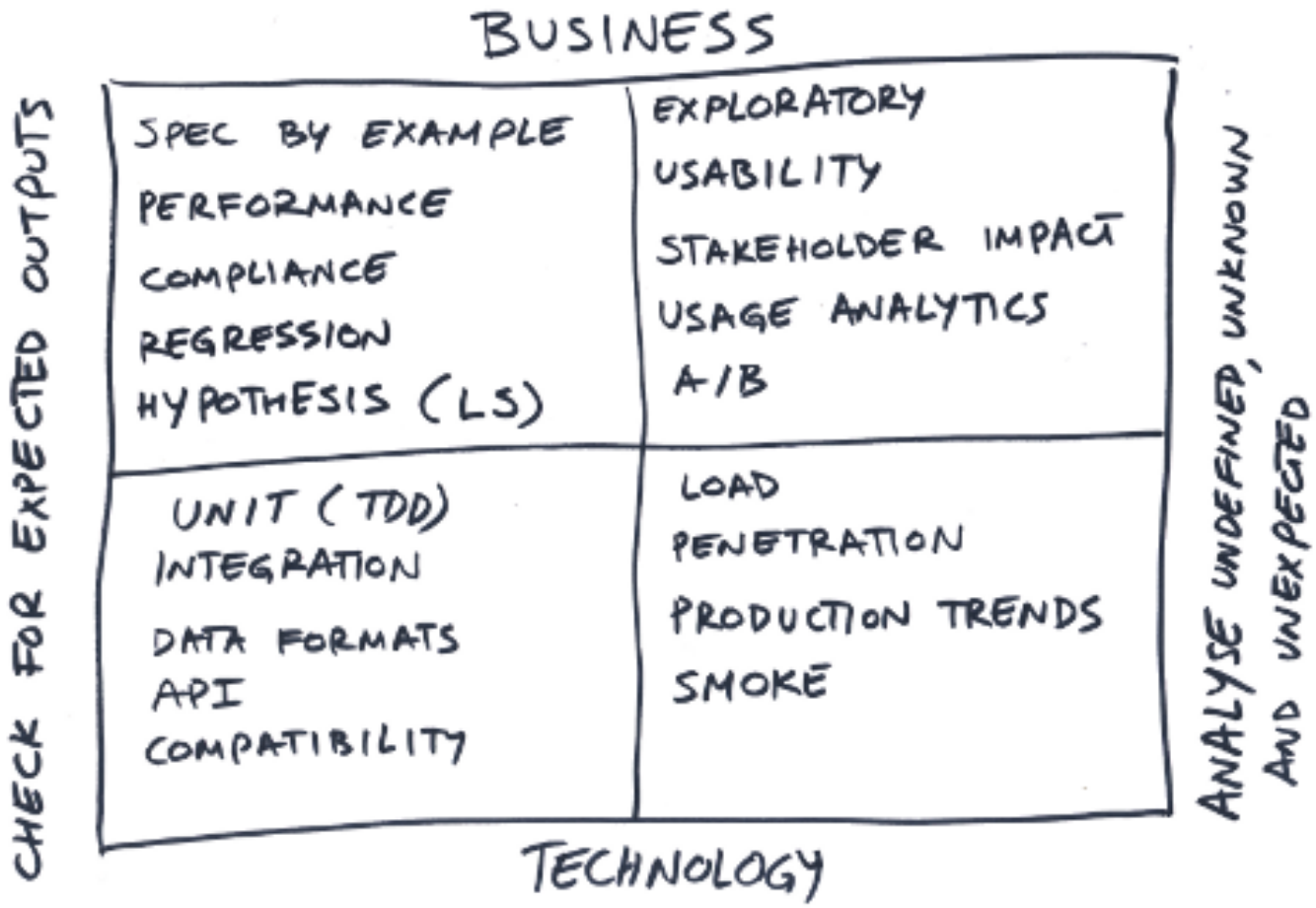
- Not a separate process in SE lifecycle
- Specification/ Modeling / Design / Documentation / Verification
- Two prominent branches in agile
  - Behavior Driven Design
  - Test Driven Design

Where Does Design Come From?

- Documents, Diagrams (a.k.a Waterfall, Feynman Method,..)
- Test Driven
  - Emergent Design
- Others: Domain Driven Design, OO Design Principles, Design Patterns
  - E.g. ports and adapters



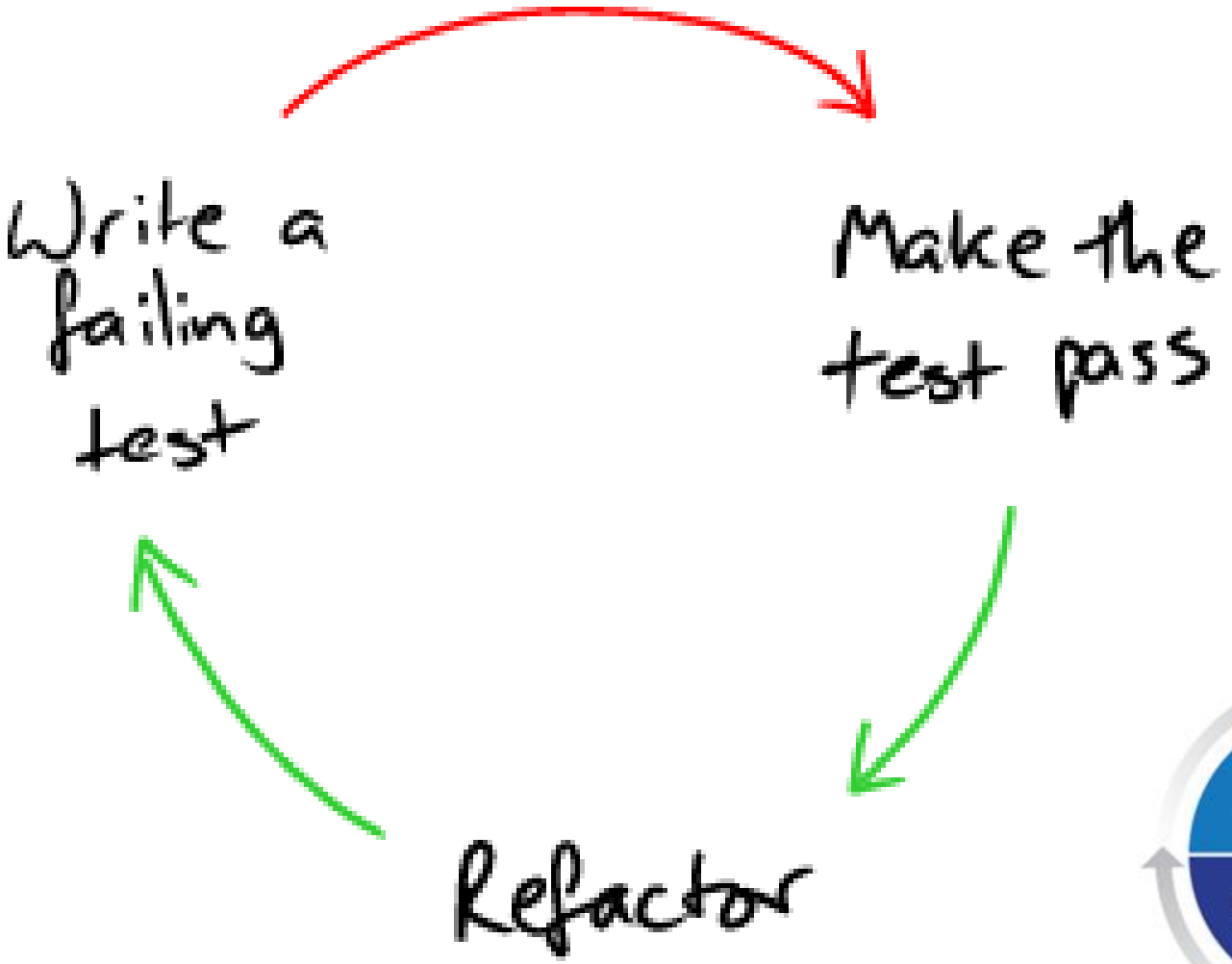
Testing Quadrant [Marick, Crispin, Adzic]



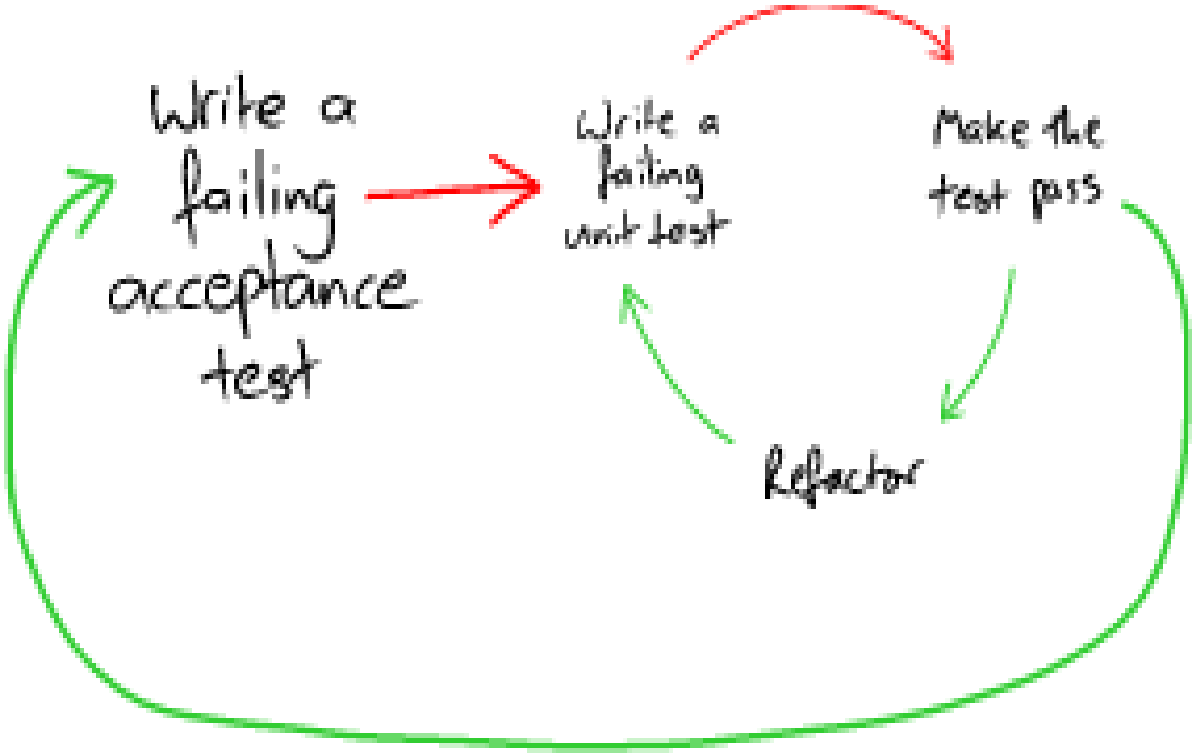
Narrowing: Agile Testing

- TDD [Beck, XP], classicist, bottom-up?, Detroit school
- BDD [North, GOOS], mockist, out-side-in, London School
- Tests drive the implementation
- Not separated
- “TDD is one of the most referenced, yet least used agile practices in industry” [Oram & Wilson, ‘10]

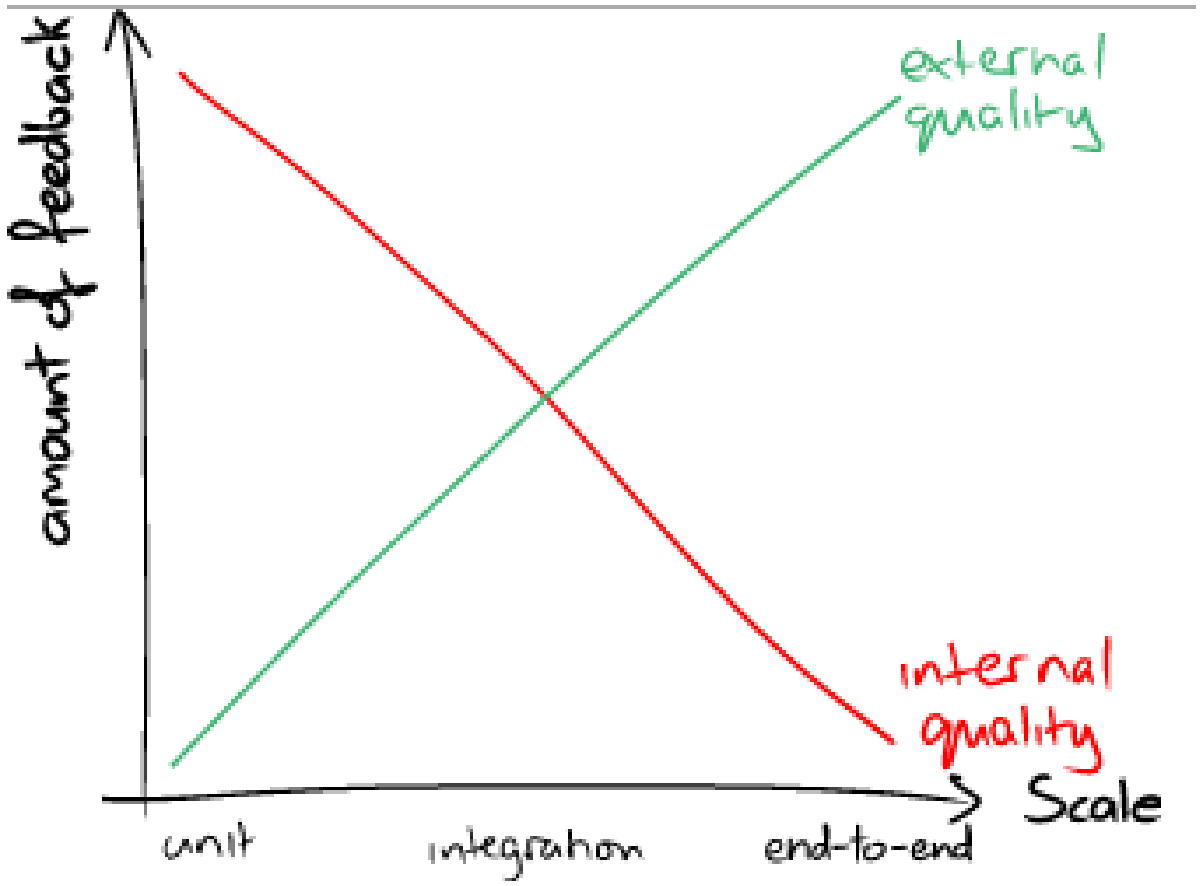
Simple TDD cycle



TDD with acceptance- and unit-test cycles

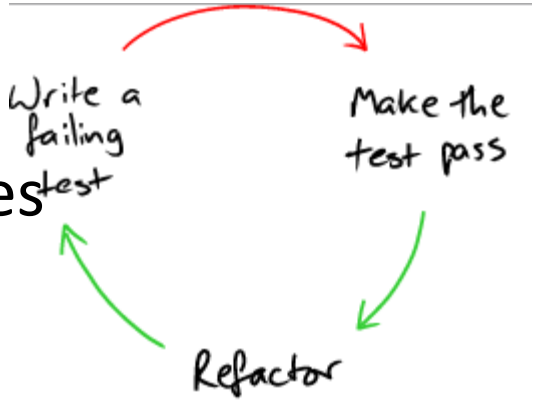


Internal & External Quality



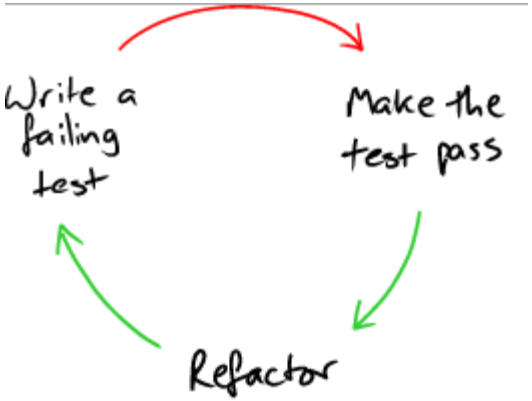
TDD

- Design mainly at refactoring stage
  - Discover other classes
  - Emergent, no over engineering (BDUF)
- State Based
- Simple and clear (beginners, modular design, e.g. 4 simple rules of design http://c2.com/cgi/wiki?XpSimplicityRules)
- Fits where clear inputs/output, e.g., algorithms
- High test coverage / regression
- Mocks mainly for external / slow entities



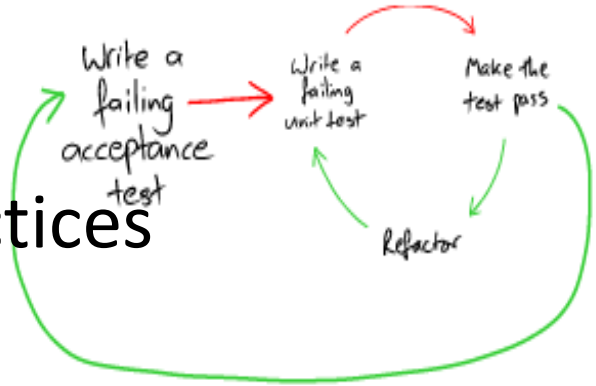
TDD Challenges

- No big picture (design)
  - Unit under test can grow
  - wasted code, rework, e.g., assumed collaborators
  - Test maintenance, become integration
  - Skipping refactoring
- Missing behavior
  - Redundant state accessors

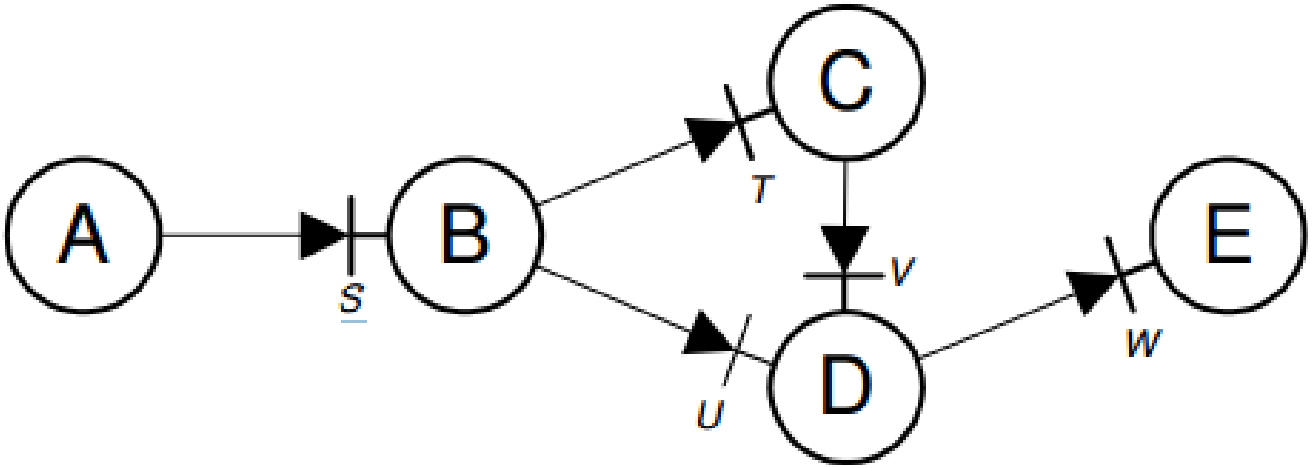


BDD

- Start from feature/acceptance test
  - The unit is behavior
  - Guides implementation
  - Behavior and collaborations based (OOP)
- Design starts at Red (testing) stage
  - Shorter refactoring
- Extensive use of mocks (test doubles)
- Fits business application with specs / user stories
- BDD per se is ignorant to dev. practices

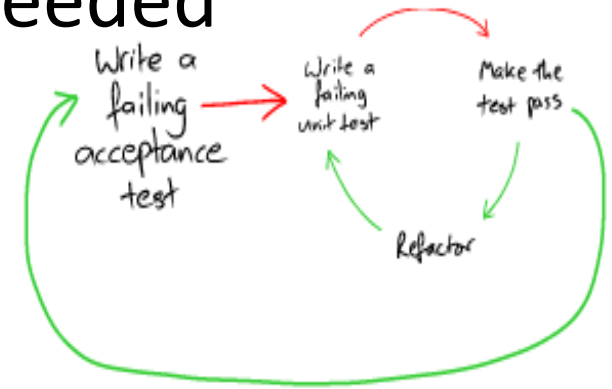


Iterative Discovery [Jmock04]

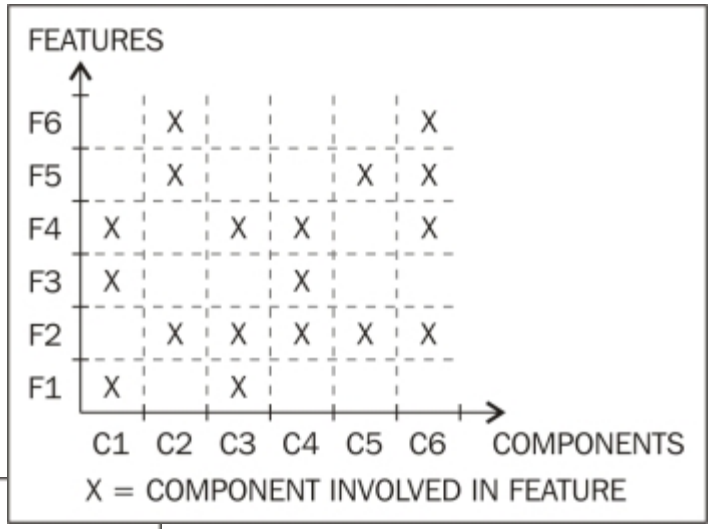


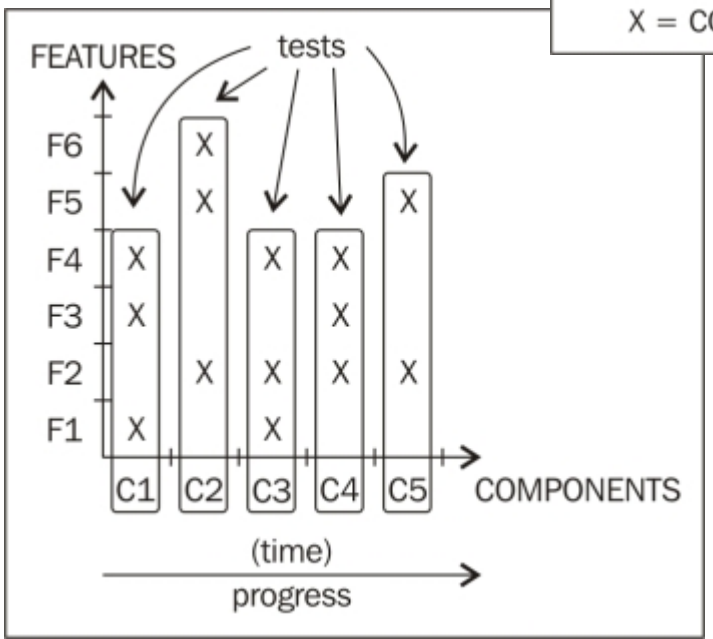
BDD Challenges

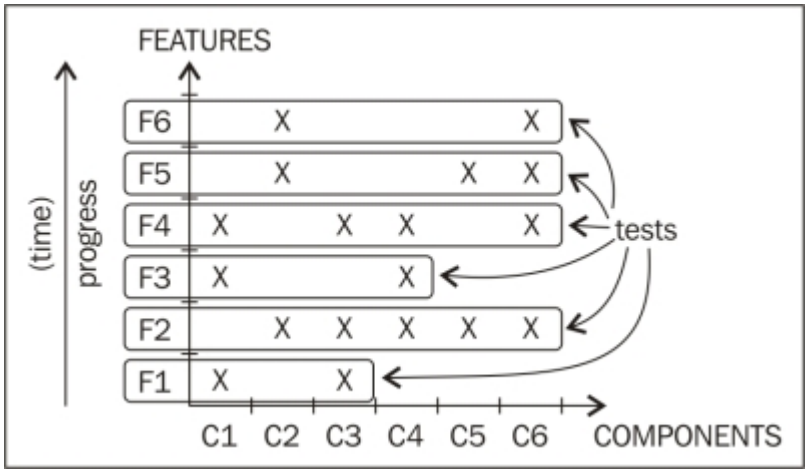
- Duplication
  - Testing the design
  - Cost of refactoring
- More complex tests
- Harder when no clear use cases
- Still higher level design skills needed
  - E.g., OOD, DBC, DDD



TDD vs. BDD [Amodeo]

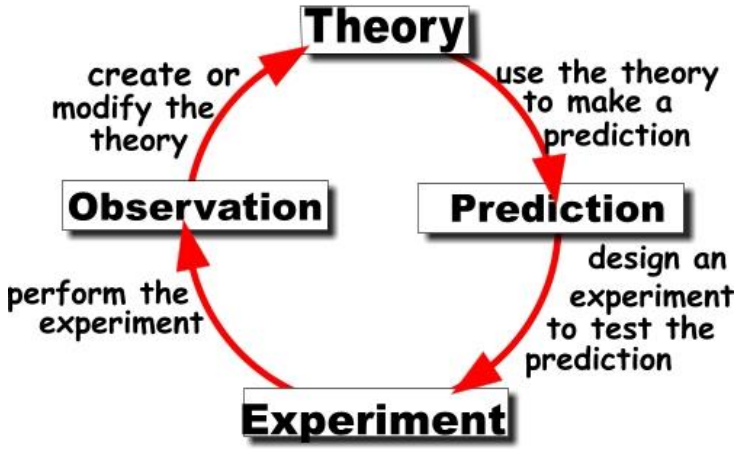






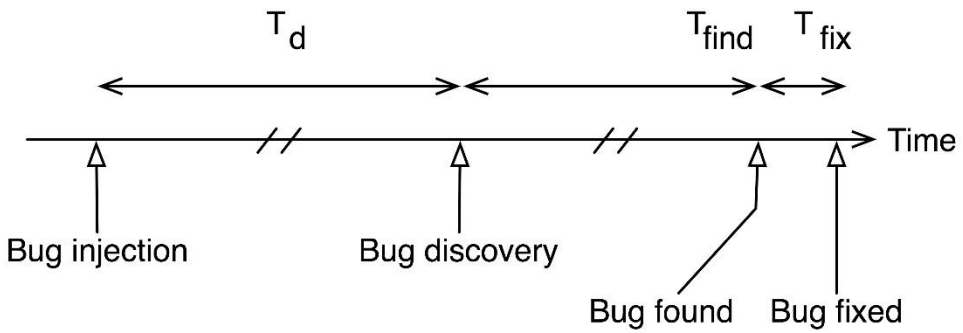
Why It Works?

- The Scientific Method



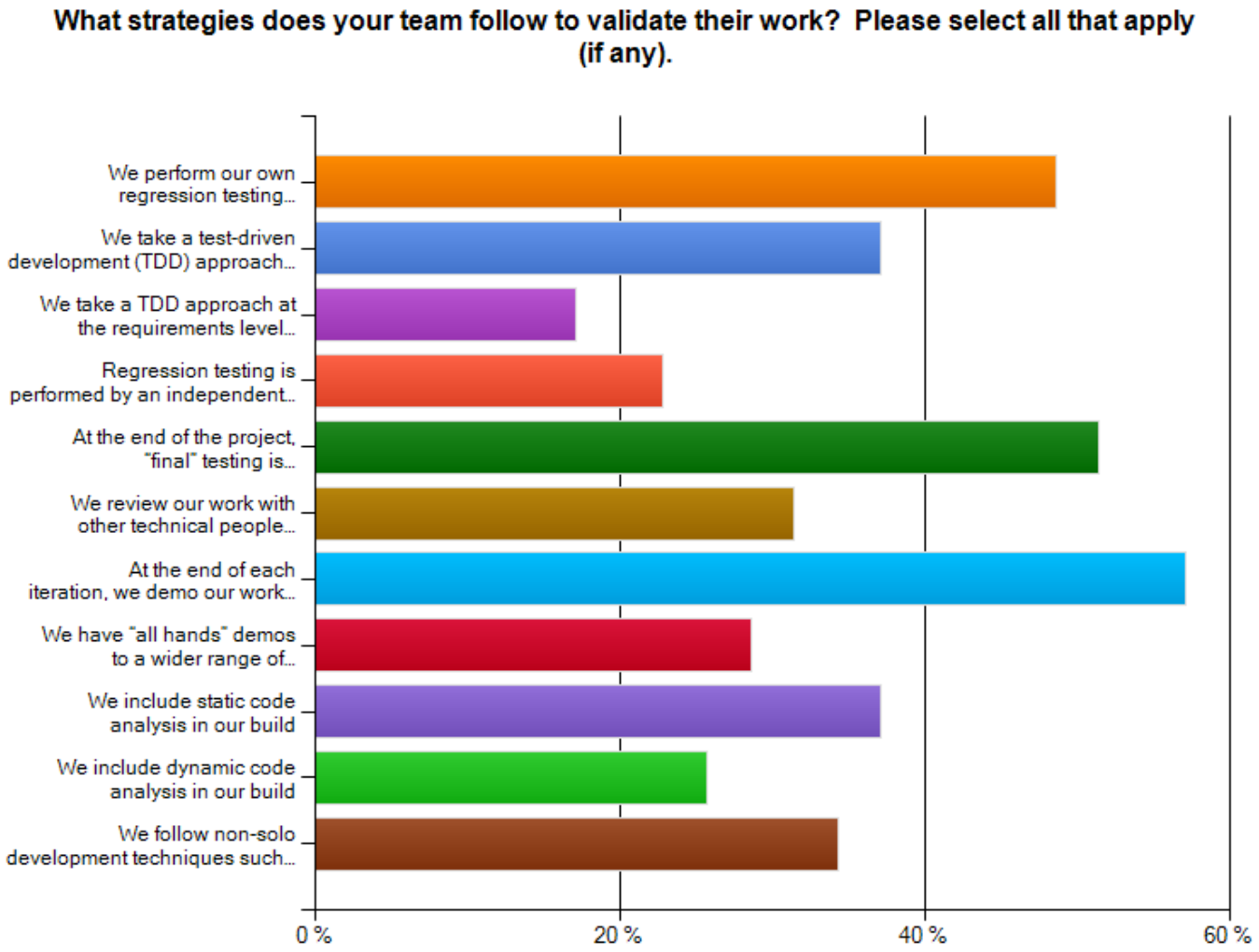
http://tomatosphere.org/teachers/guide/principal-investigation/scientific-method

- Queuing Theory



http://www.renaissancesoftware.net/blog/archives/16

How Agile Are You? 2013 Survey Results, S. Ambler



ROI for Selected Practices

Practice	12-month ROI	36-month ROI
Test Driven Development	-	1000%+
PSP/TSP	-	800%
Formal Inspections	250%	600%+
Productivity Measurement	150%	600%
Process Assessments	150%	600%
Management Training	115%	550%
Scrum	-	500%
Process Improvement Program	-	500%
Technical Staff Training	90%	500%

Sources: Rico, et al 2009; DACS 2007; McConnell 2004; Jones, 1994.

Construx

15

Empirical Evidence?

- Realizing quality improvement through test driven development: results and experiences of four industrial teams (2008) [MSR 2008]
  - 40%-90% improvement
- Making Software, What Really Works, and Why We Believe It [Oram & Wilson ed. 2010], Chapter 12: How Effective Is Test-Driven Development?
  - "For practitioners looking for some actionable advice, our expert panel recommends taking the TDD pill, carefully monitoring its interactions and side effects, and increasing or decreasing the dosage accordingly."
- Also, e.g., http://www.jamesshore.com/Blog/AoA-Correction-Test-Driven-Development.html

References

- Amodeo E., 2015. Learning Behavior-driven Development with JavaScript, Packt Publishing.
- Beck, K., 2003. Test-Driven Development by Example, Addison-Wesley.
- Evans E., 2003. Domain-Driven Design: Tackling Complexity in the Heart of Software, Prentice Hall.
- Feathers M., 2004. Working Effectively with Legacy Code. Prentice Hall, 2004.
- Fowler M., 1999. Refactoring: Improving the Design of Existing Code, Addison-Wesley.
- Freeman S. and Pryce N., 2009. Growing Object-Oriented Software, Guided by Tests, Addison-Wesley.
- Freeman S., Mackinnon T., Pryce N., and Walnes J., 2004. Mock roles, not objects. In Companion to the 19th annual ACM SIGPLAN conference on Object-oriented programming systems, languages, and applications (OOPSLA '04). ACM, New York
- North, D., 2006. Introducing Behaviour Driven Development, Better Software Magazine. Available from: http://dannorth.net/introducing-bdd/.
- Vaccari M. and Bellettini C., 2015. TDD for Android, Leanpub, https://leanpub.com/tddforandroid

Web Resources

- Mancuso S., Screencast - Oustide-In TDD, http://codurance.com/2015/05/12/does-tdd-lead-to-good-design/
- Test Double Testing Notes, https://github.com/testdouble/contributing-tests/wiki
  Testing Quadrants
  - http://www.exampler.com/old-blog/2003/08/21/#agile-testing-project-1
  - http://lisacrispin.com/downloads/AdpTestPlanning.pdf
  - http://gojko.net/2013/10/21/lets-break-the-agile-testing-quadrants/
- Demo Repo: https://github.com/robi-y/BankAccountKata/

Conclusions

- Design and Testing
- Review TDD/BDD
- When / How to use?
  - Adapt
  - Test Infected
- Questions? Comments?
- Thanks