Introduction to software testing

PBA Software Development

Test

Henrik Kühl

BUSINESS ACADEMY SOUTHWEST

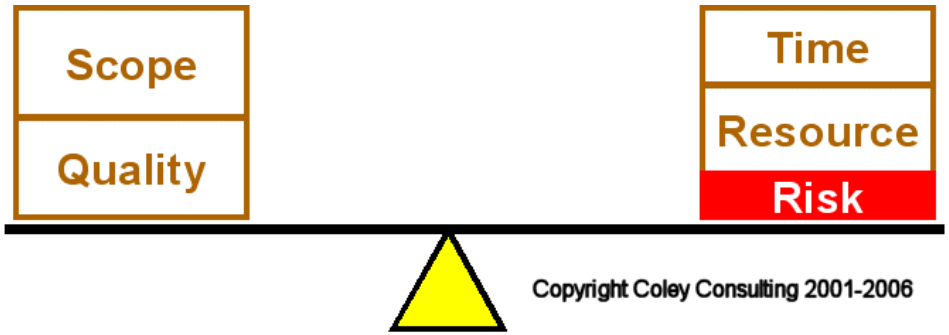
Agenda

1. Why should we test?
2. Course objectives
3. Test types
4. Test tools
5. Test techniques
6. Test processes/models
    a. Test process under the Waterfall model
    b. Test process under agile software development
7. Verification and validation

Why should we test?

1. Discover problems and solve them
2. To meet the stakeholder's expectations
   - Functional and non-functional requirements
3. You may be legally obligated to do so
   - contractual requirement or regulatory compliance
4. Reduce the risk of using the software
   - Mission- and life-critical systems

So are there any reasons not to test?



Course objectives

1. Test management
   - Strategies, models, planning, traceability, track progress (incl. defects)
2. Test techniques
   - Blackbox, whitebox, verification, validation, coverage
3. Test automation
   - Techniques and tools
4. Design for testability

Test types

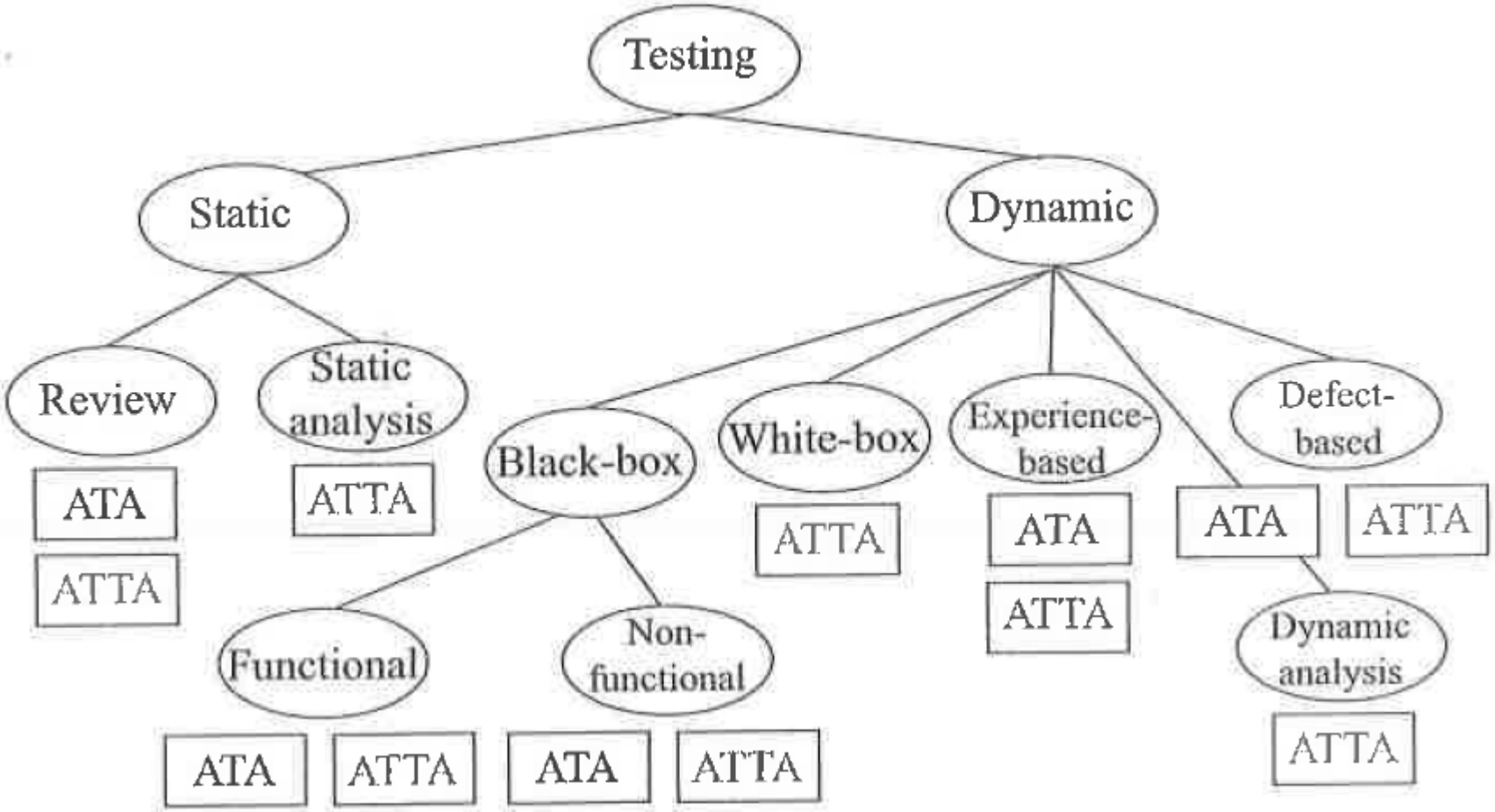
- Functional – what?
- Non functional - how or how well?
  - Usability
  - Efficiency
  - Security
  - Reliability
  - Maintainability
  - … and other forms of “itilities”
- Testing related to changes/maintenance
  - Confirmation and regression testing

Test tools

- Tools covered in this course
  - Unit testing
    - xUnit, Moq
  - Functional testing
    - SpecFlow
  - Code coverage
    - Coverlet
  - Continuous integration
    - Travis CI

- Other interesting tools
  - Non-functional testing (performance, security)
  - Functional testing (Selenium, Postman, …)
  - Test management (mostly commercial tools)
  - Static analysis (ReSharper, …)

Test techniques



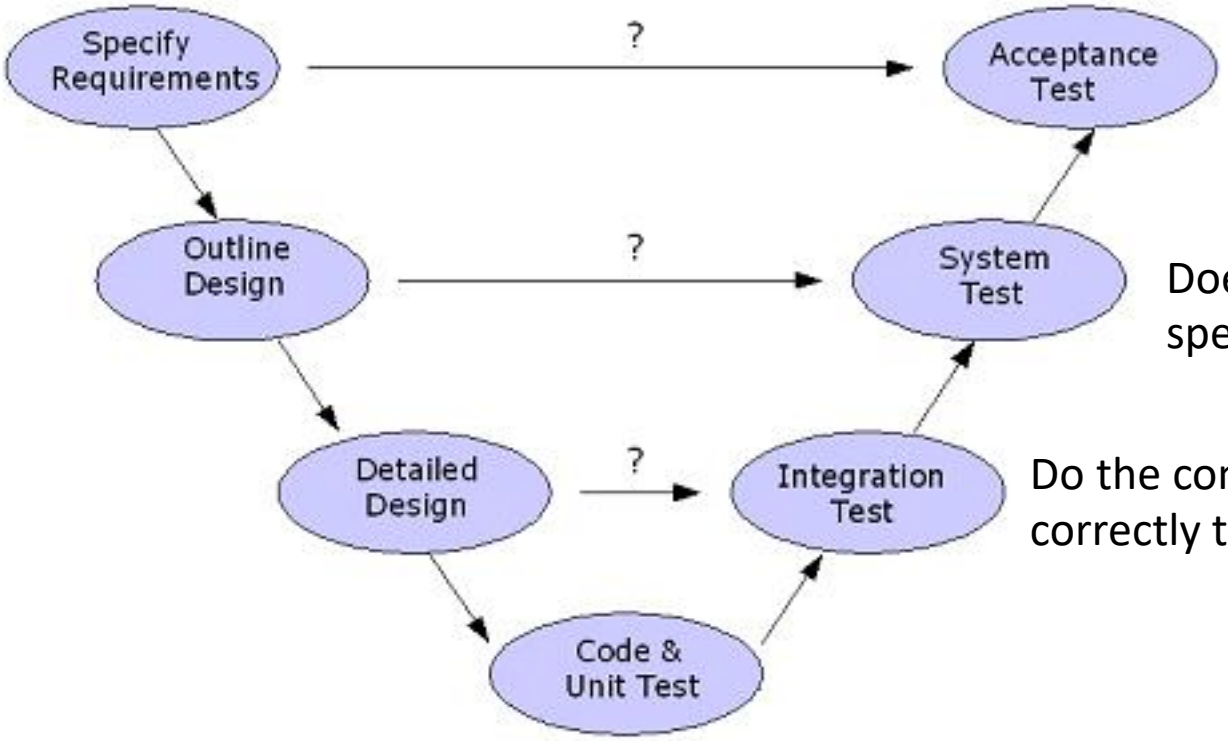
Static test

- Does not involve executing the software
  1. Review
     - A human is the defect finder
  2. Static analysis
     - A tool is used to analyse the test object and find possible defects and other issues.
     - The test object can be the code or the system architecture.

Dynamic test

- Involves executing the software
  1. Specification-based (black-box)
     - Tests are created based on an analysis of the specification and sometimes other relevant documents.
  2. Structure-based (white-box, code-based)
     - Tests are created based on an analysis of the systems internal structure (i.e. the code).
  3. Defect- and experience-based
     - Tests are created based on the testers intuition, experience and knowledge about potential defects.
  4. Dynamic analysis
     - Special tools are used to monitor the system as it runs, and log results. They are especially useful for finding failures that cannot be replicated.

Test process: Traditional waterfall testing
The V-model



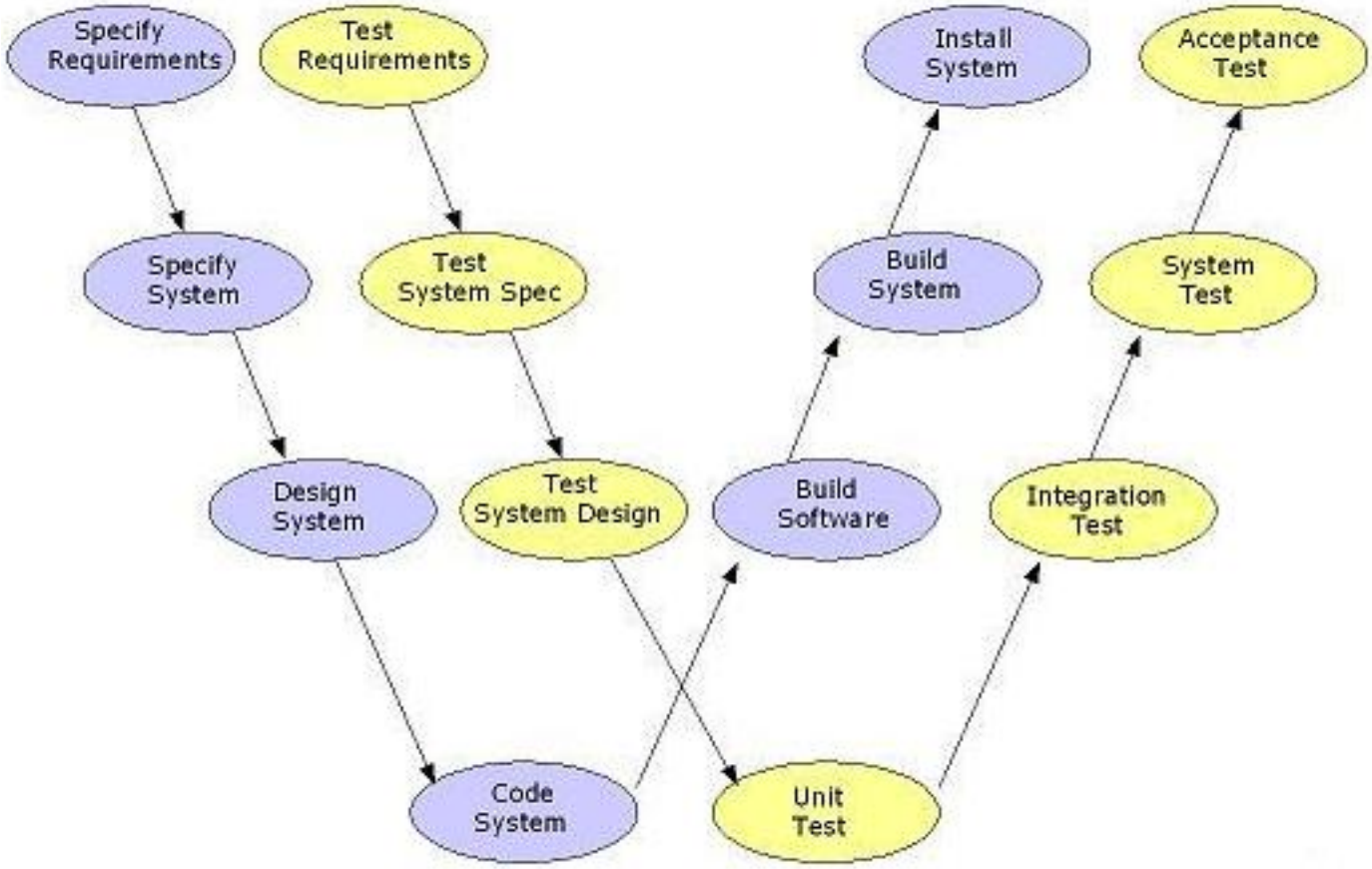
Can the system be accepted by all key stakeholders?

Does the system meet specified requirements?

Do the components work correctly together?

Test levels: Unit, Integration, System and Acceptance test

Test process: Traditional waterfall testing
The W-model



Test process: Agile testing

- The agile manifesto
  - *Individuals and interactions over processes and tools*
  - *Working software over comprehensive documentation*
  - *Customer collaboration over contract negotiation*
  - *Responding to change over following a plan*

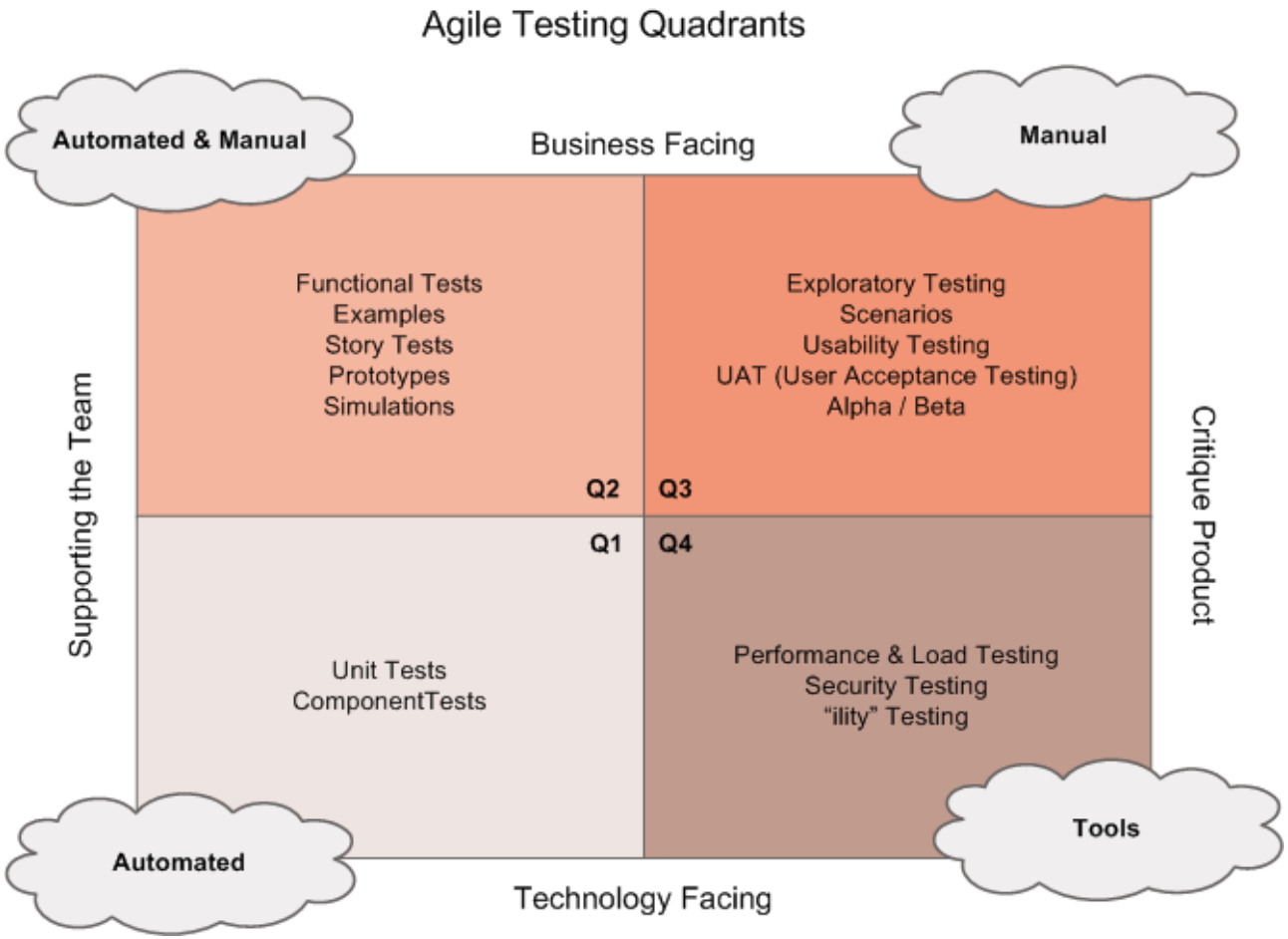
  While you should value the concepts on the right hand side you should value the things on the left hand side even more.

Test process: Agile testing

- Agile development recognizes that testing is not a separate phase, but an integral part of software development, along with coding.
- Testing and coding are done incrementally and iteratively, building up each feature until it provides enough value to release to production.
- Agile testing covers all types of testing.
- Most test planning and execution occur during iterations.

13th annual State of Agile Survey: 97% of respondents report that their organizations practice agile development methods.

Test process: Agile testing

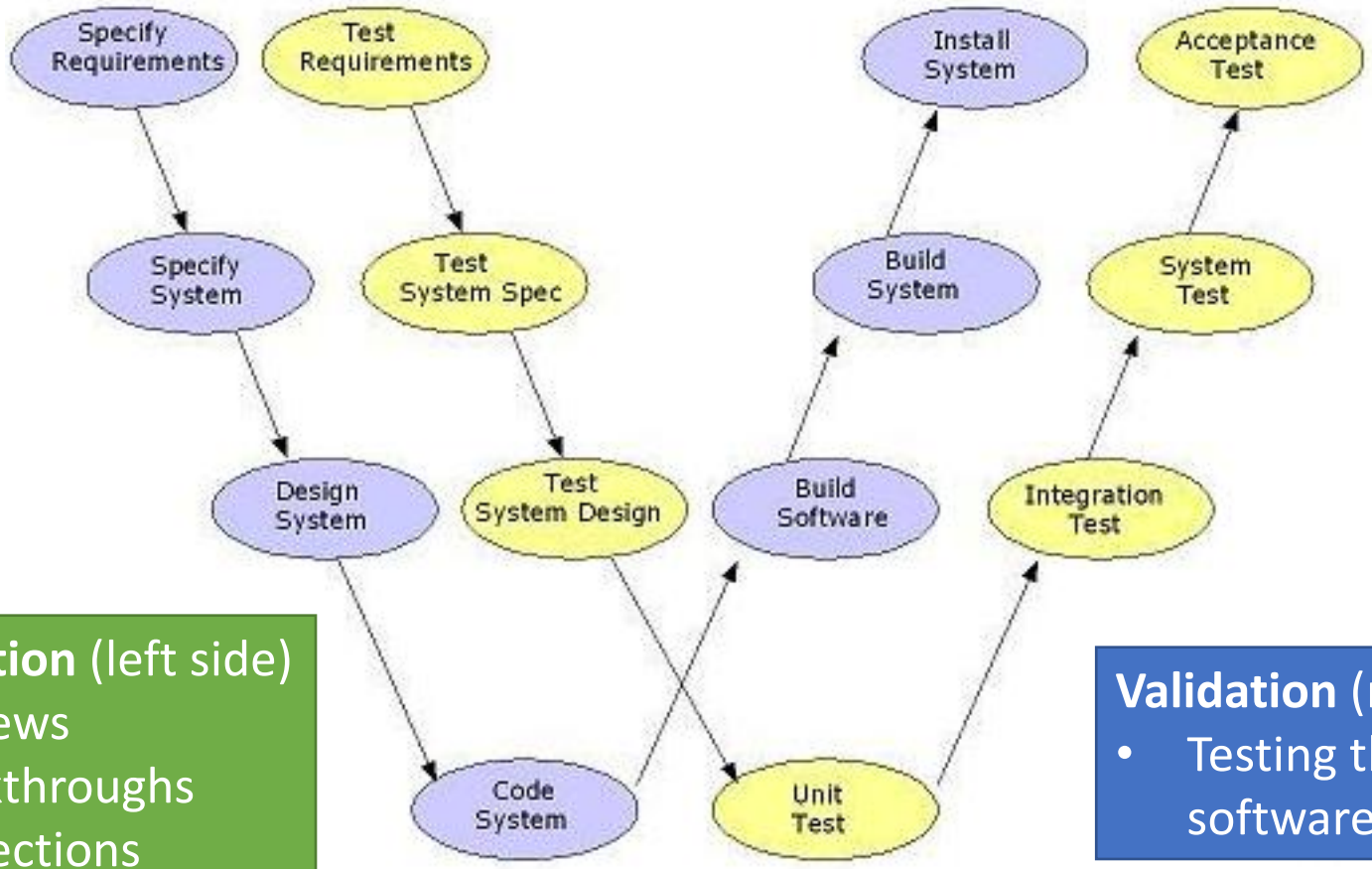


Verification and validation

- **Verification**
  - *Are we building the product right?*
  - The process of evaluating work-products (not the actual final product) of a development phase to determine whether they *meet the specified requirements* for that phase.
- **Validation**
  - *Are we building the right product?*
  - The process of evaluating software during or at the end of the development process to determine whether it *satisfies specified business requirements*.

- It is entirely possible that a product passes when verified but fails when validated. This can happen if the specifications fail to address the user's needs.

Verification and validation (traditional testing)

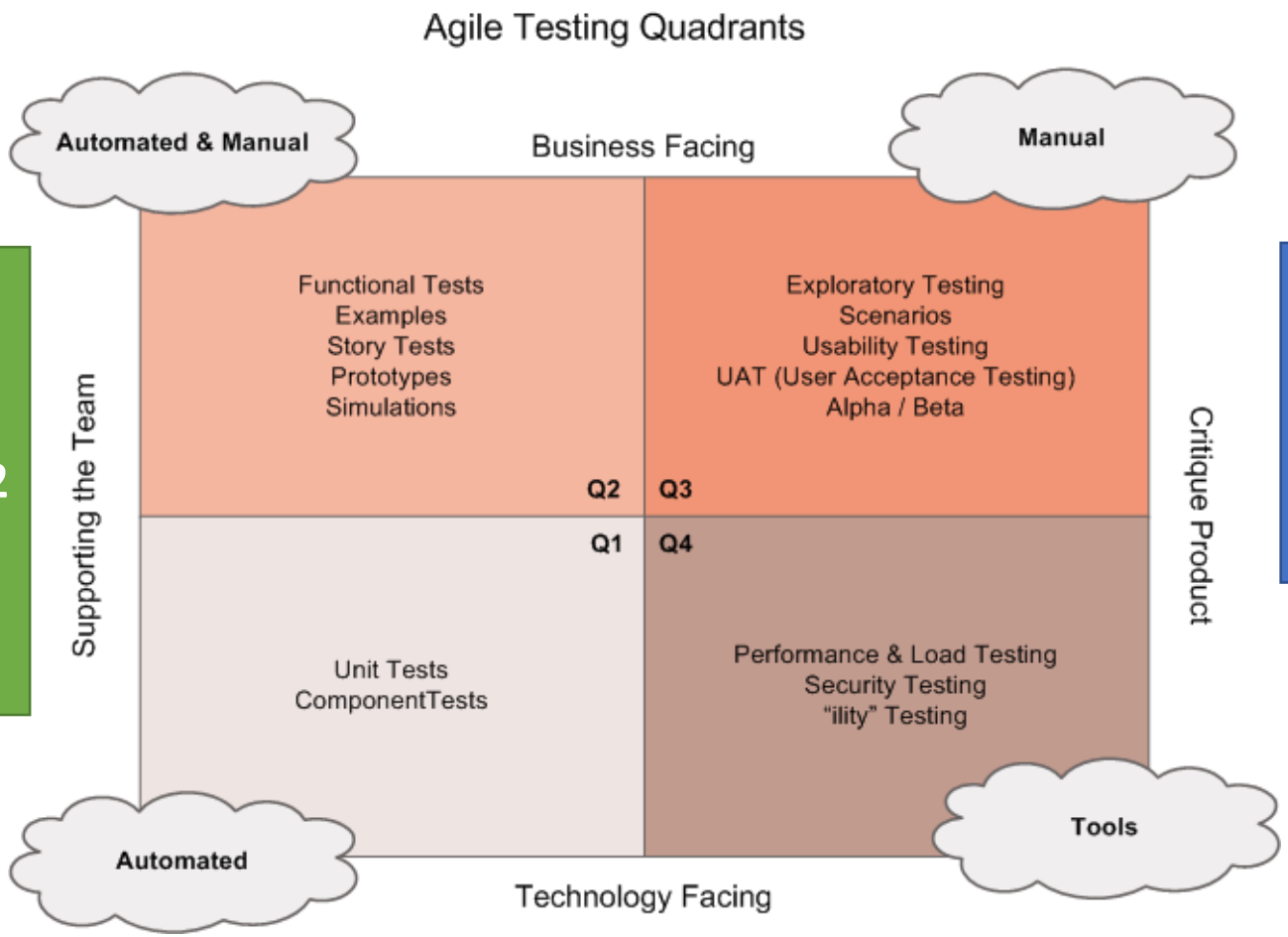


Verification (left side)
- Reviews
- Walkthroughs
- Inspections

Validation (right side)
- Testing the entire software product

Verification and validation (agile testing)

Verification
- All Q1 tests
- Most Q2 tests
- Some Q4 tests



Validation
- Most Q3 tests
- Some Q4 tests