COMP1531

🐶 Software Engineering
3.4 - Testing - Verification & Validation

In this lecture

Why?

- Understanding motivations and concepts of testing is a key in our conceptual understanding

What?

- Verification
- Static & Dynamic Verification
- Validation

Verification

Verification in a system life cycle context is a set of activities that compares a product of the system life cycle against the required characteristics for that product. This may include, but is not limited to, specified requirements, design description and the system itself.

Validation

Validation in a system life cycle context is a set of activities ensuring and gaining confidence that a system is able to accomplish its intended use, goals and objectives.

*ISO/IEC/IEEE 29148:2018

Verification

System has been built right

Validation

The right system has been built

Verification Types

- Verification can be broken up into two main types:
  - **Static verification** (what you know as "linting")
  - **Dynamic verification** (what you know as "testing")

Static verification is usually considered a more robust and reliable form of testing. however, in many cases we need to dynamically verify code too.

Static Verification

- Methods of static verification include:
  - (Linting) Style checking
  - (Linting) Type checking
  - (Linting) Logic checking. Includes:
    - Anti-pattern detection
    - Potential warnings
  - (Covered later) Key metric checking. Includes:
    - Code coverage
    - Coupling
    - Cyclomatic complexity
  - (Not covered) Formal verification
  - (Not covered) Informal reasoning

Formal Verification

- Proving (via Mathematics) that a piece of software has certain desirable properties
- Treats the software, or the algorithms implemented in the software, as a mathematical object that can be reasoned about.
- Typically involves tools like proof assistants, model checkers or automatic theorem provers.
- **Not something we cover in this course**

Formal Verification

- Tends to have a high cost in terms of effort
- E.g. to verify a microkernel
  - it took ~20 person years
  - and ~480,000 lines of proof script
  - for ~10,000 of C

Dynamic Verification

- Verification performed during the execution of software
- Often known as the "test" phase
- Typically falls into one of three categories:
  - Testing in the small
  - Testing in the large
  - Acceptance tests

Dynamic - Testing in the small

We often refer to small tests as **unit tests**, which the ISTQB defines as the testing of individual software components.

These can be white-box or black-box tests, and are written often by engineers who will implement work.

Dynamic - Testing in the large

Larger tests are tests performed to expose defects in the interfaces and in the interactions between integrated components or systems (ISTQB definition).

These tests tend to be black-box tests, and are written by either developers or independent testers.

Typically these tests fall into these categories:

- Module tests (testing specific module)
- Integration tests (testing the integration of modules)
- System tests (testing the entire system)

Dynamic - Acceptance Testing

ISTQB defines acceptance testing as formal testing with respect to user needs, requirements, and business processes conducted to determine whether or not a system satisfies the acceptance criteria and to enable the user, customers or other authorized entity to determine whether or not to accept the system.

This testing is always black-box, and is typically tested on the customer or user themselves.

Can either be performance based (trying to reach specific outcome) or stress testing.

“Testing shows the presence, not the absence of bugs” – *Edsger W. Dijkstra*

Feedback