Validation and verification

SEng 321

N Ernst and D Damian

Outline

- Definitions of V&V activities
- V&V activities throughout the development life-cycle

Definitions

Software Verification and Validation (V&V) is the process of ensuring that

The software being developed or changed will **satisfy** functional and other requirements (**validation**) and

each step in the process of building the software yields the right products (**verification**)

Validation and verification

- Validation:
  - “Are we building the right product?”
  - The software should do what the users need

- Verification:
  - “Are we building the product right?”
  - The software should conform to its specification

Validation and verification

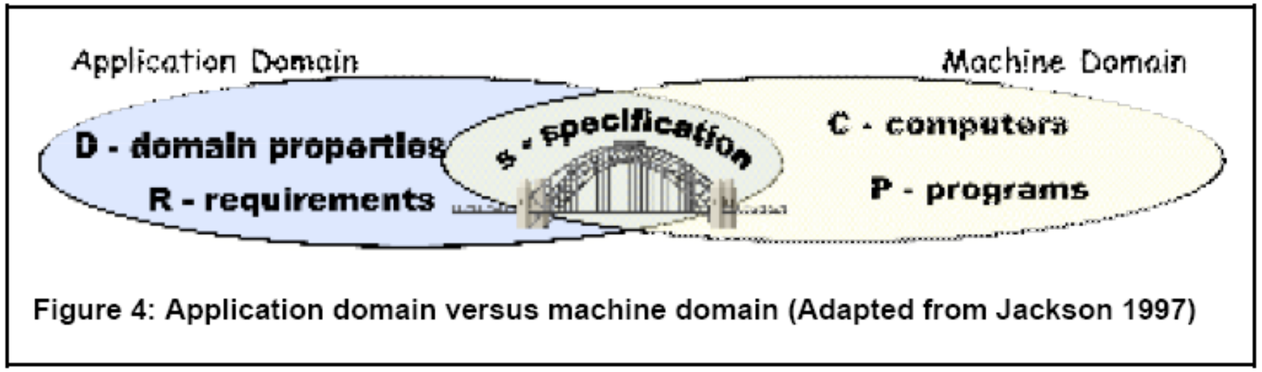


Figure 4: Application domain versus machine domain (Adapted from Jackson 1997)

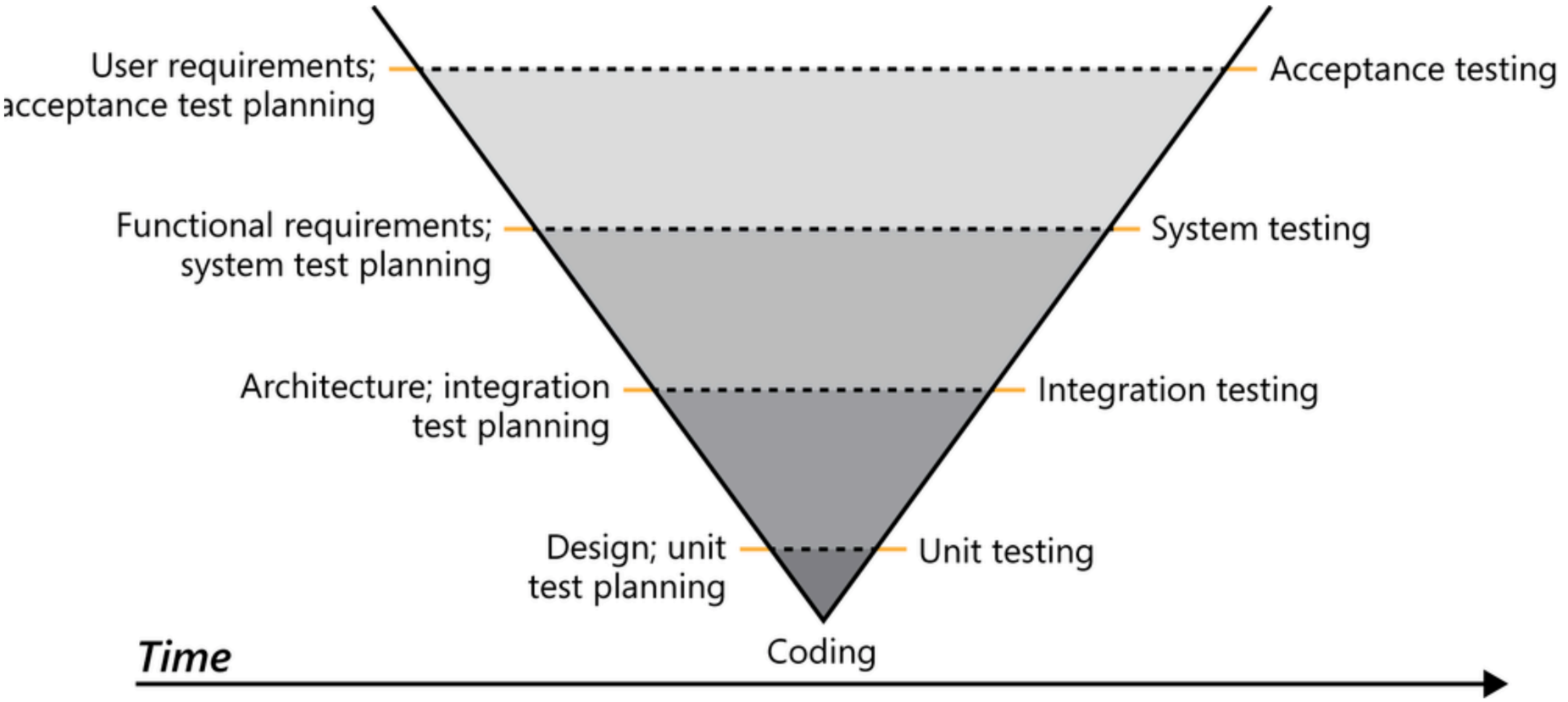
- Validation criteria:
  - Discover and understand all **domain properties**
  - Discover and understand all **requirements**
- Verification criteria:
  - The **program** satisfies the **specification**
  - The **specification**, given the **domain properties**, satisfies the **requirements**

A V&V Plan

- Major V&V activities:
  - Reviews, inspections and walkthroughs
    - At all stages
  - Testing
    - On the code

- The V&V Plan should cover all V&V activities during all phases of the life cycle

The V Model


User requirements;
acceptance test planning
Acceptance testing
Functional requirements;
system test planning
System testing
Architecture; integration
test planning
Integration testing
Design; unit
test planning
Unit testing
Coding
Time

Development stages and V&V activities

Requirements. Requirements must be reviewed with the client; rapid prototyping can refine requirements and accommodate changing requirements.

Specification. The specifications document must be checked for feasibility, traceability, completeness, and absence of contradictions and ambiguities. Specification reviews (walkthroughs or inspections) are especially effective (see SQA lecture notes).

Design. Design reviews are similar to specification reviews, but more technical. The design must be checked for logic faults, interface faults, lack of exception handling, and nonconformance to specifications.

Implementation. Code modules are informally tested by the programmer while they are being implemented (desk checking). Thereafter, formal testing of modules is done methodically by a testing team. This formal testing can include nonexecution-based methods (code inspections and walkthroughs) and execution-based methods (black-box testing, white-box testing).

Development Stages and V&V

Integration. Integration testing is performed to ensure that the modules combine together correctly to achieve a product that meets its specifications. Particular care must be given to the interfaces between modules. The appropriate order of combination must be determined as top-down, bottom-up, or a combination thereof.

Product Testing. The functionality of the product as a whole is checked against its specifications. Test cases are derived directly from the specifications document. The product is also tested for robustness (error-handling capabilities and stress tests). All source code and documentation are checked for completeness and consistency.

Acceptance Testing. The software is delivered to the client, who tests the software on the actual hardware, using actual data instead of test data. A product cannot be considered to satisfy its specifications until it has passed an acceptance test. Commercial off-the-shelf (or shrink-wrapped) software usually undergoes alpha and beta testing as a form of acceptance test.

V&V: on and about requirements

- **Validation** techniques: prototypes, reviews, walkthroughs, inspections

- Inspections of the requirements document for:
  - completeness and consistency
  - Conformance to standards
  - Conflicts
  - Ambiguous requirements

A slight detour: Technical Reviews

- Technical reviews conducted during:
  - specification
  - design
  - Coding

- Other names:
  - (Fagan) inspections
  - Walkthroughs
  - Round robin reviews

Benefits of technical reviews

Errors found	Number	Cost unit	Total
Reviews conducted			
During design	**22**	**1.5**	**33**
Before test	36	6.5	234
During test	15	15	315
After release	3	67	201
			783
No reviews conducted			
Before test	22	6.5	143
During test	82	15	1238
After release	12	67	804
			2177

[Roger Pressman, Software Engineering: A practitioner's approach, 1997]

Characteristics of technical reviews

- Well defined roles
- Typically no more than 6-7 participants
- Duration: no more than 2 hours
- Well-defined and agreed outcome
- Seek to uncover errors close to the time they might have been produced

Roles in technical reviews

- Walkthroughs
  - Leader
  - Recorder
  - Reader (implementer)
  - Author (designer)
  - Other reviewers

Fagan inspections

- Moderator
- Designer
- Coder/implementer
- Tester

Fagan inspection process-- Phases

- Planning
  - Prepare material, educate inspectors, schedule the meeting
- **Overview and Preparation**
  - Present the overview to participants, read the material to identify defects
- **Inspection meeting**
  - Find defects! Note but do not solve problems
- **Process improvement**
- Learn from current inspection to improve next inspection
- **Rework**
- Author reworks all defects
- **Follow up**
- Moderator verifies all fixes; product is re-inspect if 5% of document is reworked.

Requirements reviews

- Round robin more useful
  - Encourages equal participation
  - Participants roughly at the same level of knowledge

Guidelines for technical reviews

- Develop a checklist for each work product that is likely to be reviewed.
- Review the product, not the producer.
- Set an agenda and maintain it.
- Limit debate and rebuttal.
- Enunciate problem areas, but don't attempt to solve every problem noted.

Guidelines for technical reviews

- Take written notes.
- Limit the number of participants and insist upon advance preparation.
- Review your earlier reviews and the current review process
- Allocate resources and time schedule for technical reviews
- Conduct meaningful training for all reviewers.

V&V: software architecture (preliminary design) phase

- Updating the preliminary version of the **Acceptance Test Plan** and the **verification matrix.**

- informal reviews and walkthroughs or inspections of the preliminary software

V&V: software detailed design phase

- Complete the **Acceptance Test Plan** and the **verification matrix**, including test specifications and unit test plans.

- Conduct **informal reviews and walkthroughs or inspections** of the detailed software

V&V: implementation phase

- Code inspections and/or walkthroughs.
- Unit testing software and data structures.
- Locating, correcting, and retesting errors.
- Development of **detailed integration and product test procedures**

Code testing

- There are two basic types of methodical testing:
  - nonexecution-based testing: the module is reviewed by a team: inspections, walkthroughs
  - execution-based testing: the module is run against test cases.

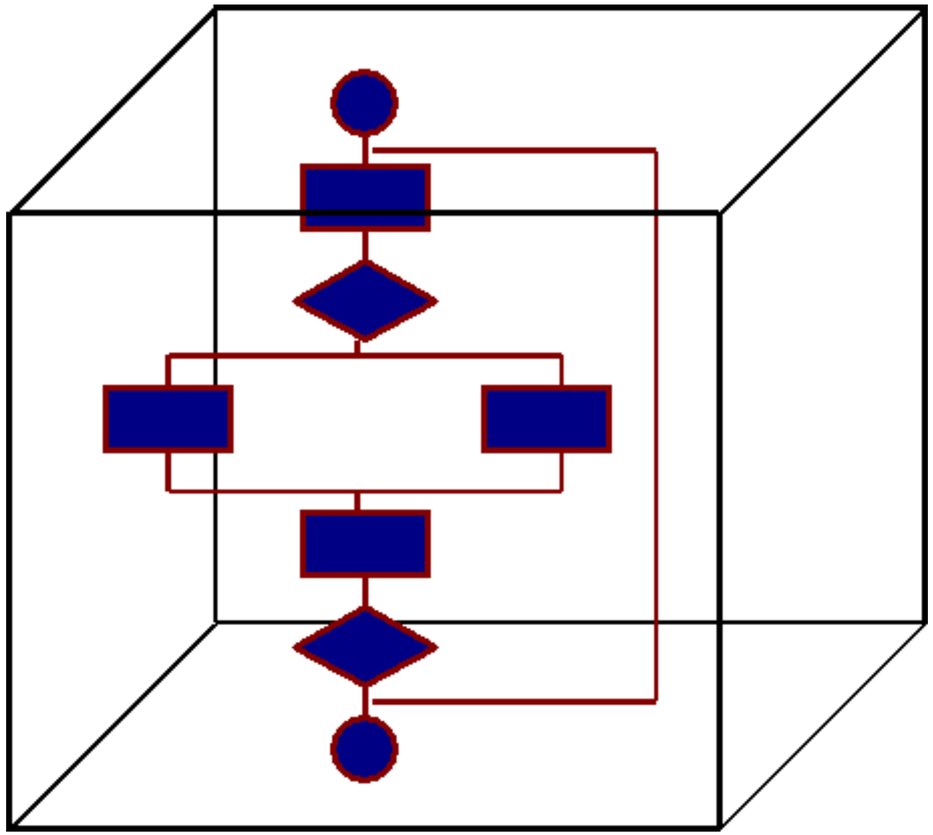
Execution-based testing

•

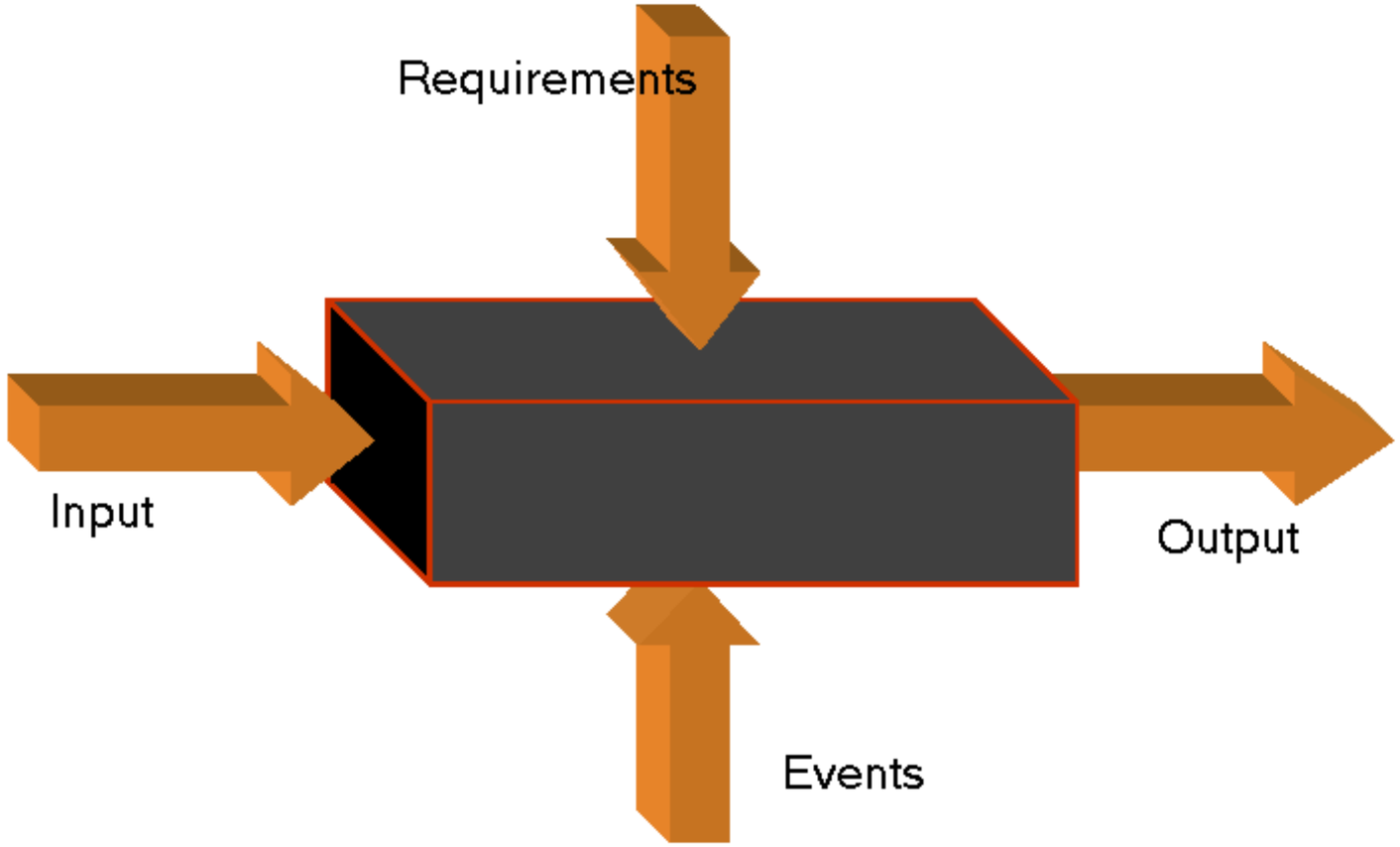
•Black-Box Testing: The code itself is ignored; the only information used in designing test cases is the specification document.

•White-Box Testing: The code itself is tested, without regard of the specifications.

White-box testing



Black-box testing



Black-box testing

•Exhaustive black-box testing is usually either impossible or unreasonable. The art of testing is to design a small, manageable set of test cases so as to maximize the chances of detecting a fault while minimizing the redundancy amongst of the cases.

•Equivalence Testing and Boundary Value Analysis
Equivalence testing, combined with boundary value analysis, is a black-box technique of selecting test cases in such a way that new cases are chosen to detect previously undetected faults. An *equivalence class* is a set of test cases such that any one member of the class is representative of any other member of the class.

•Suppose the specifications for a database product state that the product must be able to handle any number of records from 1 through 16,383. For this product, there are three equivalence classes:

- *Equivalence class 1:* less then one record.
- *Equivalence class 2*: from 1 to 16,383 records.
- *Equivalence class 3*: more than 16,383 records.

•**Testing the database product then requires that one test class from each equivalence class be selected.**

Equivalence Testing and Boundary Value Analysis

•A successful test case is one that detects a previously undetected fault. In order to maximize the chances of finding a new fault, a high-payoff technique is *boundary-value analysis*.

•Experience has shown that when a test case on or just to one side of a boundary of an equivalence class is selected, the probability of detecting a fault increases.

•Testing then should include members of the equivalence class, boundary elements and adjacent members.

V&V: integration phase

- Conducting tests per test procedures.
- Documenting test performance, test completion, and conformance of test results versus expected results.
- Providing a **test report** that includes a summary of nonconformances found during testing.
- Locating, recording, correcting, and retesting nonconformances.

V&V: software acceptance and delivery phase

- Test, analyze, inspect and demonstrate that the developed system meets its functional and non-functional requirements in house (product testing) and in the operational environment (acceptance testing)

- Locate, correct and re-test nonconformances (regression testing)