Software Testing

Part 1 of 4

Software Testing

- *Error*: mistake made by the programmer/ developer
- *Fault*: a incorrect piece of code/document (i.e., bug)
- *Failure*: result of a fault

- Goal of software testing: Cause failures to uncover faults and errors
- Develop tests
- Execute tests

Quality & Testing

- Software Quality Assurance (SQA)
  - Evaluations to be performed
  - Audits and reviews
  - Standards
  - Procedures for error tacking/reporting
  - Documentation to be produced
  - Feedback
- Verification and Validation
  - Independent group (NASA IV&V)

Verification & Validation (V&V)

- **Verification**: The software should conform to its specification (Are we building the product right?)

- **Validation**: The software should do what the user really requires (Are we building the right product?)

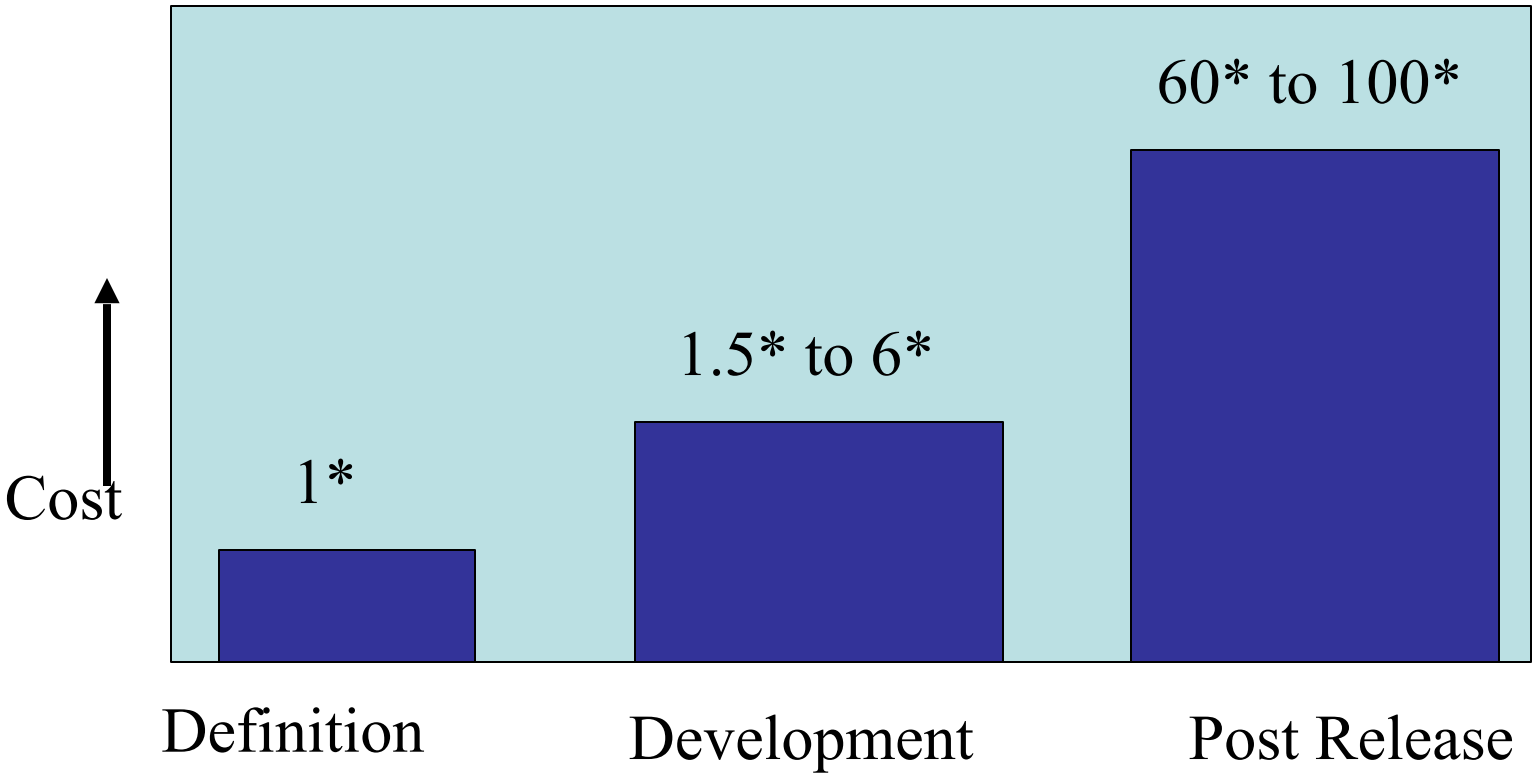
V & V Goals

- Verification and validation should establish confidence that the software is fit for its purpose
- This does NOT mean completely free of defects
- Rather, it must be good enough for its intended use and the type of use will determine the degree of confidence that is needed

“Classical” lifecycle model

- Requirements Phase
- Specification Phase (Analysis)
- Planning Phase
- Design Phase
- Implementation Phase
- Integration and Testing
- Maintenance
- Retirement

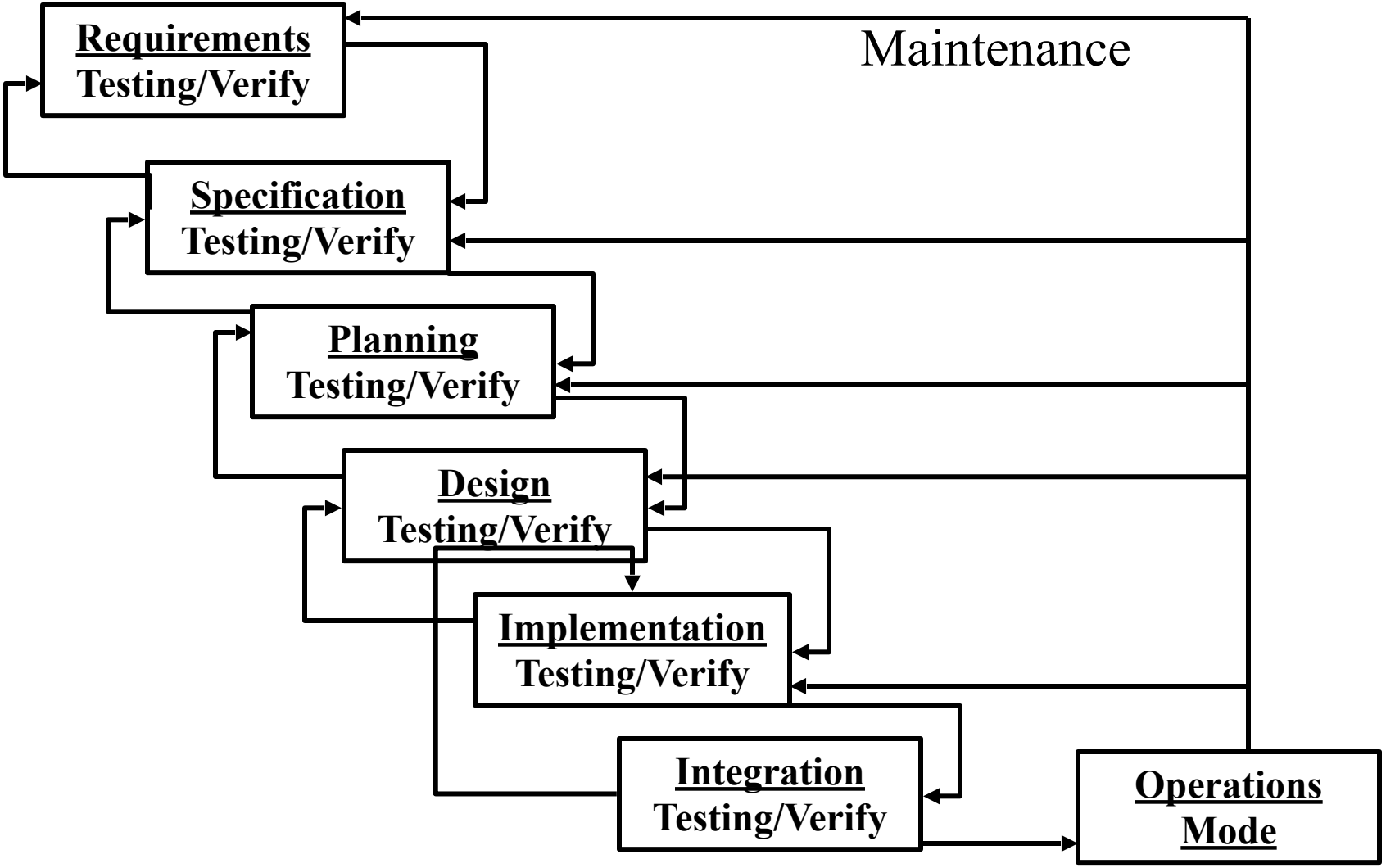
Cost to fix faults



The V & V process

- Is a whole life-cycle process - V & V must be applied at each stage in the software process.
- Has two principal objectives
  - The discovery of defects in a system
  - The assessment of whether or not the system is usable in an operational situation.

Sequential model

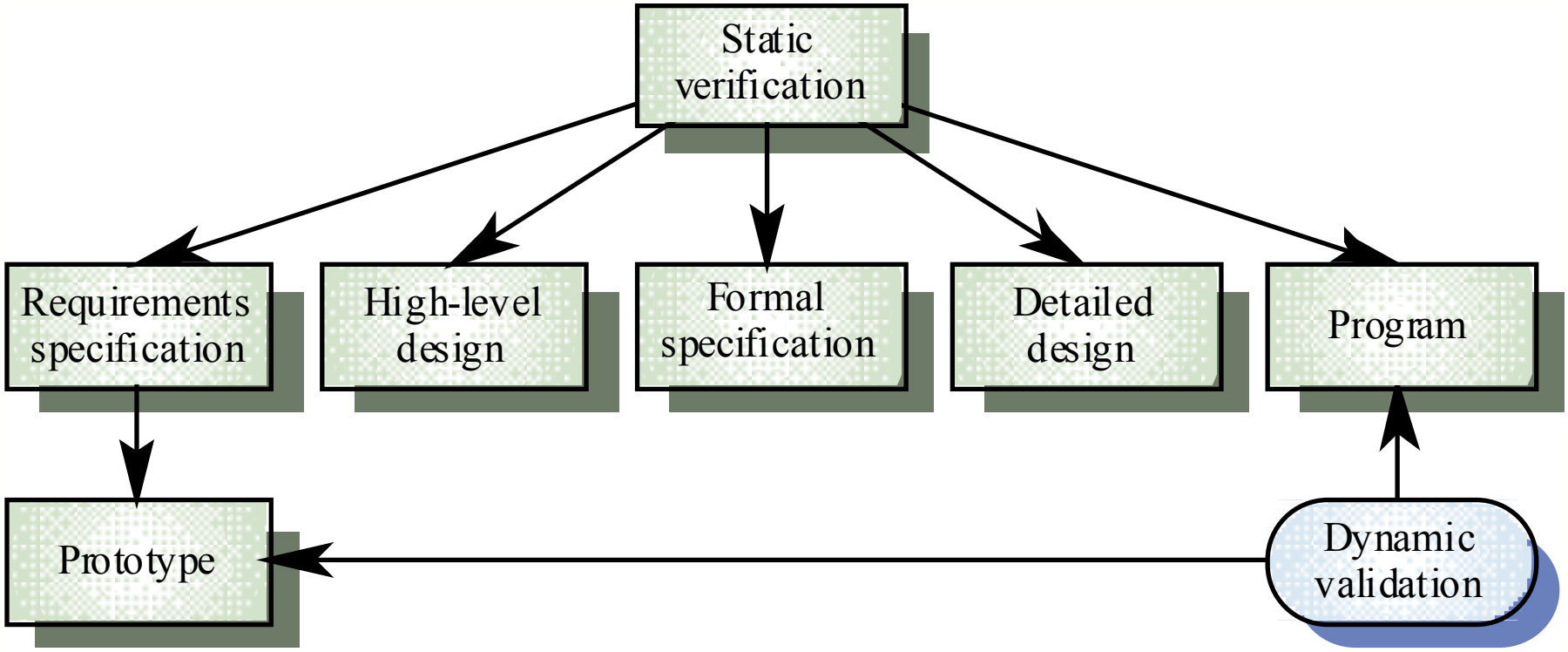


Static and dynamic verification

- *Software inspections and walkthroughs* -- Concerned with analysis of the static system representation to discover problems (static verification)

- *Software testing* - Concerned with exercising and observing product behavior (dynamic verification)
  - The system is executed with test data and its operational behavior is observed

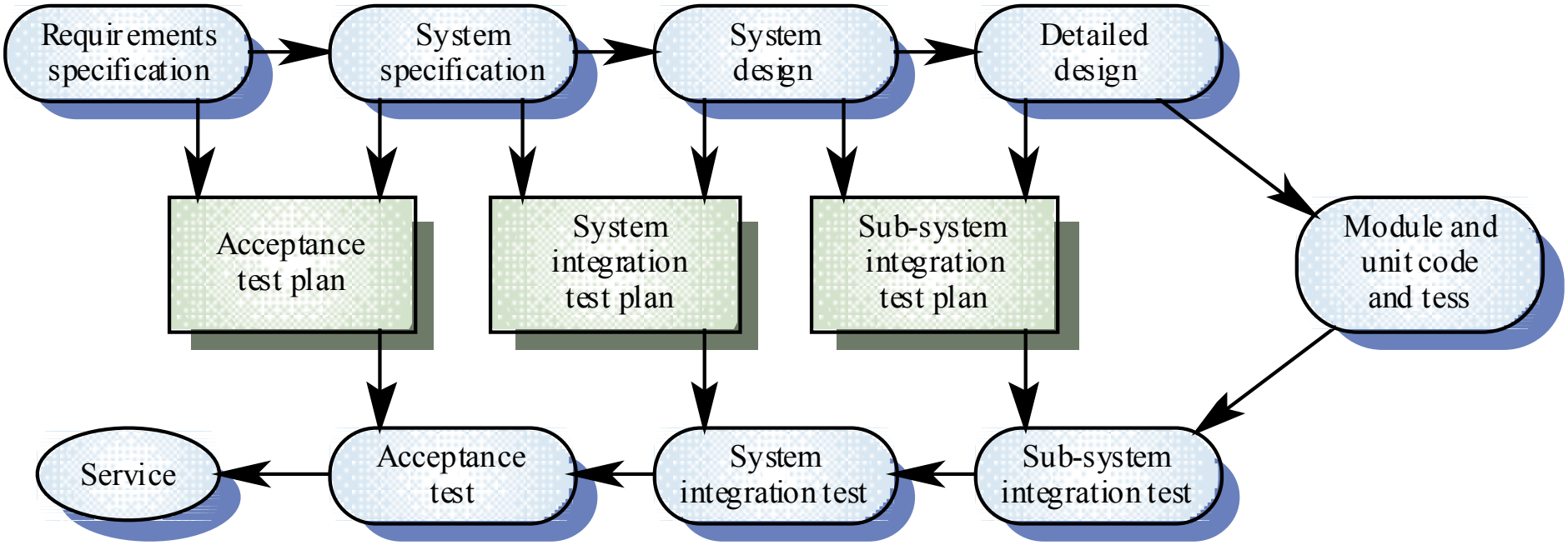
Static and Dynamic V&V



V & V planning

- Careful planning is required to get the most out of testing and inspection processes
- Planning should start early in the development process
- The plan should identify the balance between static verification and testing
- Test planning is about defining standards for the testing process rather than describing product tests

The V-model of development



Software Test Plan

- The testing process
- Requirements traceability
- Tested items
- Testing schedule
- Test recording procedures
- Hardware and software requirements
- Constraints

Walkthroughs

- Informal examination of a product (document)
- Made up of:
  - developers
  - client
  - next phase developers
  - Software Quality Assurance group leader
- Produces:
  - list of items not understood
  - list of items thought to be incorrect

Software Inspections

- Involve people examining the source representation with the aim of discovering anomalies and defects
- Do not require execution of a system so may be used before implementation
- May be applied to any representation of the system (requirements, design, test data, etc.)
- Very effective technique for discovering errors

Inspection Process

- Overview - of the document is made
- Preparation - participants understand the product in detail
- Inspection - a complete walk through is made, covering every branch of the product. Fault finding is done
- Rework - faults are fixed
- Follow - up check fixed faults. If more than say 5% of product is reworked then a complete inspection is done again.

- Statistics are kept: *fault density*

Inspection Success

- Many different defects may be discovered in a single inspection. In testing, one defect may mask another so several executions are required
- They reuse domain and programming knowledge so reviewers are likely to have seen the types of error that commonly arise

Inspections and Testing

- Inspections and testing are complementary and not opposing verification techniques
- Both should be used during the V & V process
- Inspections can check conformance with a specification but not conformance with the customer's real requirements
- Inspections cannot check non-functional characteristics such as performance, usability, etc.

Program Inspections

- Formalized approach to document reviews
- Intended explicitly for defect DETECTION (not correction)
- Defects may be logical errors, anomalies in the code that might indicate an erroneous condition (e.g. an un-initialised variable) or non-compliance with standards

Inspection Pre-conditions

- A precise specification must be available
- Team members must be familiar with the organization standards
- Syntactically correct code must be available
- An error checklist should be prepared
- Management must accept that inspection will increase costs early in the software process
- Management must not use inspections for staff appraisal

Inspection Procedure

- System overview presented to inspection team
- Code and associated documents are distributed to inspection team in advance
- Inspection takes place and discovered errors are noted
- Modifications are made to repair discovered errors
- Re-inspection may or may not be required

Inspection Teams

- Made up of at least 4 members
- Author of the code being inspected
- Inspector who finds errors, omissions and inconsistencies
- Reader who reads the code to the team
- Moderator who chairs the meeting and notes discovered errors
- Other roles are Scribe and Chief moderator

Inspection Checklists

- Checklist of common errors should be used to drive the inspection
- Error checklist is programming language dependent
- The 'weaker' the type checking, the larger the checklist
- Examples: Initialization, Constant naming, loop termination, array bounds, etc.

Inspection Rate

- 500 statements/hour during overview
- 125 source statement/hour during individual preparation
- 90-125 statements/hour can be inspected
- Inspection is therefore an expensive process
- Inspecting 500 lines costs about 40 man/hours effort (@ $50/hr = $2000)