

# **Software Quality Engineering:** Testing, Quality Assurance, and Quantifiable Improvement

Jeff Tian, [tian@engr.smu.edu](mailto:tian@engr.smu.edu)  
[www.engr.smu.edu/~tian/SQEbook](http://www.engr.smu.edu/~tian/SQEbook)

## **Chapter 6. Testing Overview**

- Testing: Concepts & Process
- Testing Related Questions
- Major Testing Techniques

---

## Testing and QA Alternatives

---

- Defect and QA:
  - ▷ Defect: error/fault/failure.
  - ▷ Defect prevention/removal/containment.
  - ▷ Map to major QA activities
  
- Defect prevention:  
Error blocking and error source removal.
  
- Defect removal:
  - ▷ Testing – Part II, Ch.6-12.
  - ▷ Inspection, etc.
  
- Defect containment: Fault tolerance and failure containment (safety assurance).

---

## QA and Testing

---

- Testing as part of QA:
  - ▷ Activities focus on testing phase
  - ▷ QA/testing in waterfall and V-models (Fig 4.1, p.45 and Fig 4.2, p.49)
  - ▷ One of the most important part of QA – defect removal: Fig 3.1 (p.30)
  
- Testing: Key questions:
  - ▷ Why: quality demonstration vs. defect detection and removal
  - ▷ How: techniques/activities/process/etc.
  - ▷ View: functional/external/black-box vs. structural/internal/white-box
  - ▷ Exit: coverage vs. usage-based

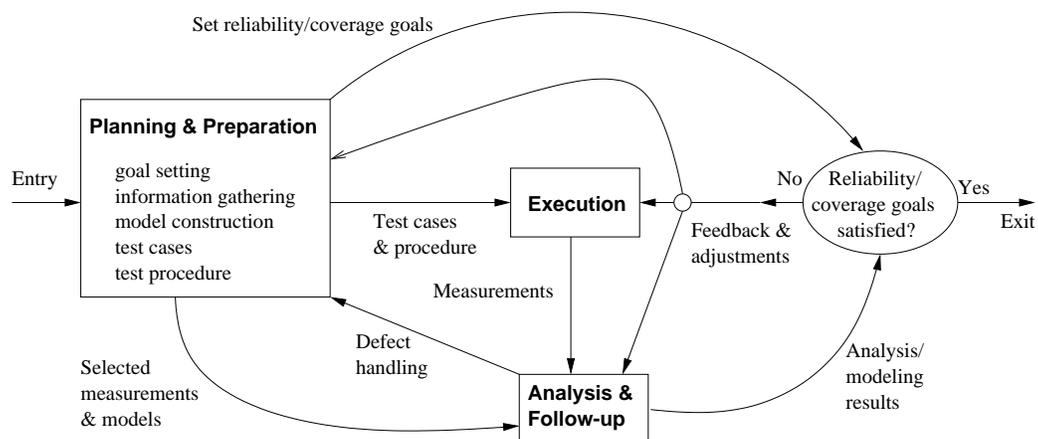
## Testing: Why?

---

- Original purpose: demonstration of proper behavior or quality demonstration.
  - ≈ “testing” in traditional settings.
  - ▷ evidence of quality or proper behavior.
  
- New purpose: defect detection & removal:
  - ▷ mostly defect-free software manufacturing vs. traditional manufacturing.
  - ▷ flexibility of software (ease of change; sometimes, curse of change/flexibility)
  - ▷ failure observation  $\Rightarrow$  fault removal.  
(defect detection  $\Rightarrow$  defect fixing)
  - ▷ eclipsing original purpose

## Testing: How

- How? Run-observe-followup (particularly in case of failure observations)
- Refinement  
⇒ generic process below (Fig 6.1, p.69)



- Generic testing process as instantiation of SQE process in Fig 5.1, p.54.

---

## Testing: Activities & Generic Process

---

- Major testing activities:
  - ▷ test planning and preparation
  - ▷ execution (testing)
  - ▷ analysis and followup
  
- Link above activities  $\Rightarrow$  generic process:
  - ▷ planning-execution-analysis-feedback.
  - ▷ entry criteria: typically external.
  - ▷ exit criteria: internal and external.
  - ▷ some (small) process variations
    - but we focus on strategies/techniques.

## Testing: Planning and Preparation

---

- Test planning:
  - ▷ goal setting based on customers' quality perspectives and expectations.
  - ▷ overall strategy based on the above and product/environmental characteristics.
  
- Test preparation:
  - ▷ preparing test cases/suites:
    - typically based on formal models.
  - ▷ preparing test procedure.
  
- More details in Chapter 7.

---

## Testing: Execution

---

- General steps in test execution
  - ▷ allocating test time (& resources)
  - ▷ invoking test
  - ▷ identifying system failures  
(& gathering info. for followup actions)
  
- Key to execution: handling both normal vs. abnormal cases
  
- Activities closely related to execution:
  - ▷ failure identification:  
test oracle problem
  - ▷ data capturing and other measurement
  
- More details in Chapter 7.

---

## Testing: Analysis and Followup

---

- Analysis of testing results:
  - ▷ result checking (as part of execution)
  - ▷ further result analyses
    - defect/reliability/etc. analyses.
  - ▷ other analyses: defect ~ other metrics.
  
- Followup activities:
  - ▷ feedback based analysis results.
  - ▷ immediate: defect removal (& re-test)
  - ▷ other followup (longer term):
    - decision making (exit testing, etc.)
    - test process improvement, etc.
  
- More details in Chapter 7 (for activities) and Part IV (for mechanisms/models/etc.).

## Testing: How?

---

- How to test?
  - refine into three sets of questions
    - ▷ basic questions
    - ▷ testing technique questions
    - ▷ activity/management questions
  
- Basic questions addressed in Ch.6:
  - ▷ What artifacts are tested?
  - ▷ What to test?
    - from which view?
    - related: type of faults found?
  - ▷ When to stop testing?

---

## Testing Technique Questions

---

- Testing technique questions:
  - ▷ specific technique used?
  - ▷ systematic models used?
    - related model questions (below)
  - ▷ adapting technique from other domains?
  - ▷ integration for efficiency/effectiveness↑?
  
- Testing model questions:
  - ▷ underlying structure of the model?
    - main types: list vs. FSM?
  - ▷ how are these models used?
  - ▷ model extension?
  
- Major techniques: Chapters 8–11.

---

## Test Activity/Management Questions

---

- Addressed already: Generic process and relation to QA and software processes.
  
- Other activity/management questions:
  - ▷ Who performs which specific activities?
  - ▷ When can specific activities be performed?
  - ▷ Test automation? What about tools?
  - ▷ Artifacts used for test management?
  - ▷ General environment for testing?
  - ▷ Product type/segment?
  
- Most questions answered in Chapter 7.  
Integration issues addressed in Chapter 12.

---

## Functional vs. Structural Testing

---

- Key distinction: Perspective on what need to be checked/tested.
  
- Functional testing:
  - ▷ tests external functions.
    - as described by external specifications
  - ▷ black-box in nature;
    - functional mapping: input  $\Rightarrow$  output
    - without involving internal knowledge
  
- Structural testing:
  - ▷ tests internal implementations.
    - components and structures.
  - ▷ white-box in nature;
    - “white” here = seeing through  $\Rightarrow$  internal elements visible.
  - ▷ really clear/glass/transparent box.

---

## Black-Box vs. White-Box View

---

- Object abstraction/representation:
  - ▷ high-level: whole system  $\approx$  black-box.
  - ▷ low-level: individual statements, data, and other elements  $\approx$  white-box.
  - ▷ middle-levels of abstraction:
    - function/subroutine/procedure, module, subsystem, etc.
    - method, class, super-class, etc.
  
- Gray-box (mixed black-/white-) testing:
  - ▷ many of the middle levels of testing.
  - ▷ example: procedures in modules
    - procedures individually as black box,
    - procedure interconnection  $\approx$  white-box at module level.

---

## White-box Testing

---

- Program component/structure knowledge (or implementation details)
  - ▷ statement/component checklist
  - ▷ path (control flow) testing
  - ▷ data (flow) dependency testing
  
- Applicability
  - ▷ test in the small/early
  - ▷ dual role of programmers/testers
  - ▷ can also model specifications
  
- Criterion for stopping
  - ▷ mostly coverage goals.
  - ▷ occasionally quality/reliability goals.

---

## Black-box Testing

---

- Input/output behavior
  - ▷ specification checklist.
  - ▷ testing expected/specified behavior
    - finite-state machines (FSMs)
  - ▷ white-box technique on specification
    - functional execution path testing.
  
- Applicability
  - ▷ late in testing: system testing etc.
  - ▷ suitable for IV&V
  - ▷ compatible with OO/Reuse paradigm
  
- Criteria: when to stop
  - ▷ traditional: functional coverage
  - ▷ usage-based: reliability target

## When to Stop Testing

---

- Resource-based criteria:
  - ▷ Stop when you run out of time.
  - ▷ Stop when you run out of money.
  - ▷ Irresponsible  $\Rightarrow$  quality/other problems.
  
- Quality-based criteria:
  - ▷ Stop when quality goals reached.
  - ▷ Direct quality measure: reliability
    - resemble actual customer usages
  - ▷ Indirect quality measure: coverage.
  - ▷ Other surrogate: activity completion.
  - ▷ Above in decreasing desirability.

---

## Usage-Based Testing and OP

---

- Usage-based statistical testing:
  - ▷ actual usage and scenarios/information
  - ▷ captured in operational profiles (OPs)
  - ▷ simulated in testing environment  
(too numerous  $\Rightarrow$  random sampling)
  
- Applicability
  - ▷ final stages of testing.
  - ▷ particularly system/acceptance testing.
  - ▷ use with s/w reliability engineering.
  
- Termination criteria: reliability goals

---

## Coverage-Based Testing

---

- Coverage-based testing:
  - ▷ systematic testing based on formal (BBT/WBT) models and techniques
  - ▷ coverage measures defined for models
  - ▷ testing managed by coverage goals
  
- Applicability
  - ▷ all stages of testing.
  - ▷ particularly unit and component testing.
  - ▷ later phases at high abstraction levels.
  
- Termination criteria: coverage goals

---

## Steps in Systematic Testing

---

- Instantiation of Fig 6.1 (p.69), but,
  - ▷ with a formalized strategies/goals,
  - ▷ based on formal models and techniques,
  - ▷ managed by termination criteria.
  
- Steps in model construction and usage:
  - ▷ Define the model, usually represented as graphs and relations.
  - ▷ “Check” individual elements:
  - ▷ “Test”: derive (sensitize) test cases and then execute them.
  - ▷ Result checking and followup.
  
- Specifics on model construction and usage in individual testing techniques: Ch.8–11.