Software Quality Engineering:

Testing, Quality Assurance, and Quantifiable Improvement

Jeff Tian, tian@engr.smu.edu
www.engr.smu.edu/~tian/SQEbook

Part I. Overview and Basics

- General Book Information
- Quality: View/Measure/Model
- QA Activities/Alternatives
- From QA to SQE

Ch.1: Main Problems Addressed

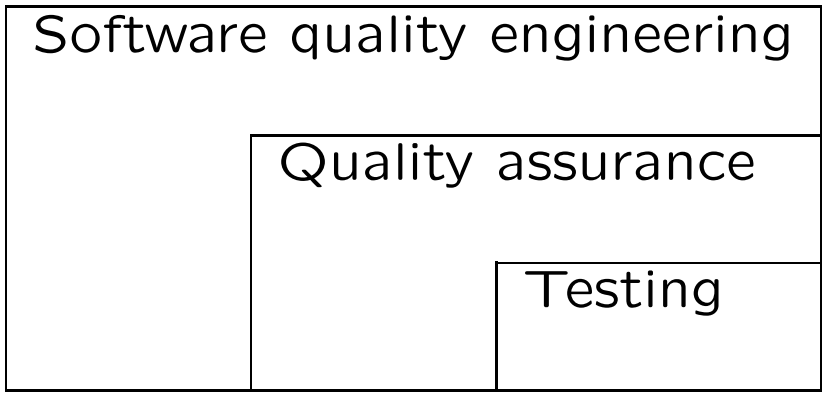
- Deliver software system that...
  - ▷ does what it is supposed to do.
  - ▷ does the things correctly.
  - ▷ show/demonstrate/prove it ("does").

- Major difficulties for the above:
  - ▷ Size: MLOC products common
  - ▷ Complexity
  - ▷ Environmental stress/constraints
  - ▷ Flexibility/adaptability expected
  - ▷ "no silver bullet", but...
    SQE (software quality engineering) helps

Ch.1: SQE as Answer

- Major SQE activities:
  - ▷ Testing: remove defect & ensure quality
  - ▷ Other QA alternatives to testing
  - ▷ How do you know: analysis & modeling

- Scope and content hierarchy: Fig.1.1 (p.6).



Ch.1: Book Contents

- QA alternatives/activities:
  (and mapping to our Parts/Chapters)
  - ▷ Testing (Part II)
  - ▷ Other alternatives (Part III):
    - – defect prevention (Ch.13)
    - – inspection, review, analysis (Ch.14)
    - – formal verification (Ch.15)
    - – defect containment (Ch.16), etc .
  - ▷ Analysis and improvement (Part IV)

- Issues in different QA alternative
  - ▷ Applicability and effectiveness
  - ▷ Dealing with quality problems/defects:
    - – prevention/removal/tolerance
  - ▷ Cost
  - ▷ Comparison (Ch.17) and
    improvement (Part IV).

Ch.1: Usage and Readership

- Part I (overview/concept) should precede other (possibly parallel) parts.

- Dependency within each parts:
  - ▷ Essential: prior knowledge
  - ▷ Non-essential:
    - – simple to complex
    - – process/external order or sequence
    - – top-down (and bottom-up?), etc.
  - ▷ Details: Fig 1.2 (p.10)

- Background knowledge needed:
  - ▷ CS/SE: object of study
  - ▷ math/statistics: modeling/analysis.
  - ▷ Details: Section 1.4

Ch.2: General Quality Views

- In Kitchenham & Pfleeger (1996):
  - ▷ Transcendental view: seen/not-defined.
  - ▷ User view: fitness for purpose.
  - ▷ Manufacturing view: conform to specs.
  - ▷ Product view: inherent characteristics.
  - ▷ Value-based view: willing to pay.

- In Prahalad & Krishnan (1999):
  - ▷ Conformance/adaptability/innovation
  - ▷ Traditional: conformance only
  - ▷ $\Rightarrow$ service, manage expectations:
    - 0 defect $\rightarrow$ 0 defection
  - ▷ Domain specific (for info. age?):
    - specificity, stability, evolvability

Ch.2: Quality Frameworks

- In various frameworks/mega-models
  - ▷ McCall: factors, criteria, and metrics
  - ▷ Basili: GQM (goal-question-metric)
  - ▷ SEI/CMM: process focus/levels
  - ▷ ISO 9000 series of standards
  - ▷ Dromey: component reflects Q-attributes

- ISO 9126 quality characteristics:
  - ▷ Functionality: what is needed?
  - ▷ Reliability: function correctly.
  - ▷ Usability: effort to use.
  - ▷ Efficiency: resource needed.
  - ▷ Maintainability: correct/improve/adapt.
  - ▷ Portability: one environment to another.
  - ▷ Adaptation in corporate definitions.
    - e.g. IBM's CUPRIMDSO.

Ch.2: Defining Quality

- Quality: views and attributes

View	Attribute	
	Correctness	Other
Customer (external)	Failures: reliability safety etc.	Maintainability Readability Portability Performance Installability Usability, etc.
Developer (internal)	Faults: count distr class etc.	Design Size Change Complexity presentation control data, etc.

- SQE focus: correctness-related.

Ch.2: Defect and Quality

- Defect/bug definition
  - ▷ Failure: external behavior
    – deviation from expected behavior
  - ▷ Fault: internal characteristics
    – cause for failures
  - ▷ Error: incorrect/missing human action
    – conceptual mistakes
  - ▷ Bug/debug: problematic terms, avoid

- Relations (not necessarily 1-1):
  errors ⇒ faults ⇒ failures: Fig 2.1 (p.21)

- Defect handling/resolution: Chapter 4.

Ch.3: Defect vs. QA

- QA: quality assurance
  - ▷ QA as dealing with defects.
  - ▷ Focus on correctness aspect of Q.
  - ▷ Many activities: testing & others
  - ▷ How $\Rightarrow$ classification
- How to deal with defects:
  - ▷ Prevention
  - ▷ Removal (detect them first)
  - ▷ Containment
- Classification illustrated: Fig 3.1 (p.30)

Ch.3: Error/Fault/Failure & QA

- Preventing fault injection
  - ▷ Causal/statistical/etc. analyses based
  - ▷ Preventive measures:
    – education, technology, process, tools
  - ▷ Formal verification (faults absent)

- Removal of faults
  - ▷ Inspection: faults discovered
  - ▷ Testing: failures trace back to faults

- Tolerance of faults
  - ▷ Local failure $\not\Rightarrow$ global failure
  - ▷ Dynamic measures to tolerant faults

Ch.3: Defect Prevention Overview

- Error blocking
  - ▷ Error: missing/incorrect actions
  - ▷ Direct intervention
  - ▷ Error blocked
    $\Rightarrow$ fault injections prevented
  - ▷ Rely on technology/tools/etc.

- Error source removal
  - ▷ Root cause analysis
    $\Rightarrow$ identify error sources
  - ▷ Removal through education/training/etc.

- Details: Chapter 13.

Ch.3: Formal Verification Overview

- Motivation
  - ▷ Fault present:
    – revealed through testing/inspection/etc.
  - ▷ Fault absent: formally verify.

- Basic ideas
  - ▷ Behavior formally specified:
    – pre/post conditions, or
    – as mathematical functions.
  - ▷ Verify "correctness":
    – intermediate states/steps,
    – axioms and compositional rules.
  - ▷ Approaches: axiomatic/functional/etc.

- Details: Chapter 15.

Ch.3: Inspection Overview

- Artifacts (code/design/test-cases/etc.) from req./design/coding/testing/etc. phases.

- Informal reviews:
  - ▷ Self conducted reviews.
  - ▷ Independent reviews.
  - ▷ Orthogonality of views desirable.

- Formal inspections:
  - ▷ Fagan inspection and variations.
  - ▷ Process and structure.
  - ▷ Individual vs. group inspections.
  - ▷ What/how to check: techniques .

- Details: Chapter 14.

Ch.3: Testing Overview

- Product/Process characteristics:
  - ▷ Object: product type, language, etc.
  - ▷ Scale/order:
    unit, component, system, ...
  - ▷ Who: self, independent, 3rd party

- What to check:
  - ▷ Verification vs. validation
  - ▷ External specifications (black-box)
  - ▷ Internal implementation (white/clear-box)

- Criteria: when to stop?
  - ▷ Coverage of specs/structures.
  - ▷ Reliability $\Rightarrow$ usage-based

- Much, much more in Part II.

Ch.3: Fault Tolerance Overview

- Motivation
  - ▷ Fault present but
    removal infeasible/impractical
  - ▷ Fault tolerance ⇒ contain defects
- FT techniques: break fault-failure link
  - ▷ Recovery: rollback and redo
  - ▷ NVP: N-version programming
    – fault blocked/out-voted
- Details: Chapter 16.

Ch.3: Safety Assurance Overview

- Extending FT idea for safety:
  - ▷ FT: tolerate fault
  - ▷ Extend: tolerate failure
  - ▷ Safety: accident free
- Safety related concepts:
  - ▷ Accident: failure w/ severe consequences
  - ▷ Hazard: precondition to accident
- Safety assurance:
  - ▷ Hazard elimination/reduction/control
  - ▷ Damage control
- Details: Chapter 16.

Ch.4: QA in Context

- QA and the overall development context
  - ▷ Defect handling/resolution
  - ▷ Activities in process
  - ▷ Alternative perspectives: verification/validation (V&V) view

- Defect handling/resolution
  - ▷ Status and tracking
  - ▷ Causal (root-cause) analysis
  - ▷ Resolution: defect removal/etc.
  - ▷ Improvement: break causal chain

Ch.4: Defect Measurement and Analysis

- Defect measurement:
  - ▷ Parallel to defect handling
  - ▷ Where injected/found?
  - ▷ Type/severity/impact?
  - ▷ More detailed classification possible?
  - ▷ Consistent interpretation
  - ▷ Timely defect reporting
- Defect analyses/quality models
  - ▷ As followup to defect handling.
  - ▷ Data and historical baselines
  - ▷ Goal: assessment/prediction/improvement
  - ▷ Causal/risk/reliability/etc. analyses
- Details in Part IV.

Ch.4: QA in Software Processes

- Mega-process: initiation, development, maintenance, termination.

- Development process components: requirement, specification, design, coding, testing, release.

- QA in waterfall process: Fig 4.1 (p.45)
  - ▷ QA in testing phase/sub-phases (V-model in sub-phases: Fig 4.2, p.49
  - ▷ Defect prevention in early phases
  - ▷ Defect removal in middle/late phases
  - ▷ Defect containment in late phases
  - ▷ Phase transitions: inspection/review/etc.
  - ▷ QA scattered throughout the process

Ch.4: QA in Software Processes

- Process variations and QA:
  - ▷ Alternative to waterfall
  - ▷ Iterative: QA in iterations/increments;
  - ▷ Spiral: QA and risk management;
  - ▷ Mixed/synthesized: case specific;
  - ▷ More evenly distributed QA activities
- QA in maintenance processes:
  - ▷ Focus on defect handling;
  - ▷ Some defect containment activities for critical or highly-dependable systems;
  - ▷ Data for future QA activities
- QA scattered throughout all processes

Ch.4: V&V

- Validation: w.r.t. requirement (what?)
  - ▷ Appropriate/fit-for-use/ "right thing"?
  - ▷ Scenario and usage inspection/testing;
  - ▷ System/integration/acceptance testing;
  - ▷ Beta testing and operational support.

- Verification: w.r.t. specification/design (how?)
  - ▷ Correct/ "doing things right"?
  - ▷ Design as specification for components;
  - ▷ Structural and functional testing;
  - ▷ Inspections and formal verification.

- V&V in software process: Fig 4.2 (p.49).

Ch.4: V&V vs DC View

- Two views of QA:
  - ▷ V&V view
  - ▷ DC (defect-centered) view in this book
  - ▷ Interconnected: mapping possible?

- Mapping between V&V and DC view:
  - ▷ V&V after commitment
    (defect injected already)
    $\Rightarrow$ defect removal & containment focus
  - ▷ Verification: more internal focus
  - ▷ Validation: more external focus
  - ▷ In V-model: closer to user or developer?

- Mapping: Table 4.1 (p.51)

Ch.5: QA to QE

- QA activities need additional support:
  - ▷ Planning and goal setting
  - ▷ Management:
    - – When to stop?
    - – Adjustment and improvement, etc.
    - – All based on assessments/predictions

- Assessment of quality/reliability/etc.:
  - ▷ Data collection needed
  - ▷ Analysis and modeling
  - ▷ Providing feedback for management

- Overall process: Fig 5.1 (p.54)
  - – Software quality engineering (SQE)

Ch.5: QE Activities

- Idea/activities similar to QIP.

- Major activities:
  - ▷ Pre-QA planning;
  - ▷ QA: covered previously (Ch.3 & 4);
  - ▷ Post-QA analysis and feedback (maybe parallel instead of "post-")

- Pre-QA planning:
  - ▷ Quality goal
  - ▷ Overall QA strategy:
    - – QA activities to perform?
    - – Measurement/feedback planning

Ch.5: Pre-QA Planning

- Setting quality goal(s):
  - ▷ Identify quality views/attributes
  - ▷ Select direct quality measurements
  - ▷ Assess quality expectations vs. cost
- Forming a QA strategy
  - ▷ Individual strength/weakness/cost of QA alternatives matched against goals
  - ▷ Measurement/feedback planning:
    - – define measurements & collect data
    - – preliminary choices of models/analyses
    - – feedback & followup mechanisms, etc.

Ch.5: Analysis and Feedback

- Measurement:
  - ▷ Defect measurement as part of defect handling process
  - ▷ Other related measurements
- Analyses: quality/other models
  - ▷ Data and historical baselines
  - ▷ Goal: assessment/prediction/improvement
  - ▷ Focus on defect/risk/reliability analyses
- Feedback and followup:
  - ▷ Frequent feedback: assessments/predictions
  - ▷ Possible improvement areas
  - ▷ Used in management and improvement
- Details in Part IV.

Ch.5: QE Context and Cost

- QE activities in software processes:
  - ▷ Different start/end time
  - ▷ Different sets of activities and focuses
  - ▷ In waterfall process: Fig 5.2 (p.61)
  - ▷ In other processes: slight variations

- QE activity/effort distribution/dynamics:
  - ▷ Different focus in different phases
  - ▷ Different levels (qualitatively)
  - ▷ Different build-up/wind-down patterns
  - ▷ In waterfall process: Fig 5.3 (p.63)
  - ▷ In other processes:
    similar but more evenly distributed