Software Quality Engineering:

Testing, Quality Assurance, and Quantifiable Improvement

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

Chapter 8. Coverage and Usage Testing Based on Checklists and Partitions

- Checklist-Based Testing
- Partitions and Partition Testing
- Usage-Based Testing with Musa's OPs
- OP Development: Procedures/Examples

Checklists for Testing

- Ad hoc testing:
  - ▷ "run-and-observe"
  - ▷ How to start the run?
  - ▷ Areas/focuses of "observations"?
  - ▷ Implicit checklists may be involved.
- Explicit checklists:
  - ▷ Function/features (external)
  - ▷ Implementation (internal)
  - ▷ Standards, etc.
  - ▷ Mixed or combined checklists

Checklists for Testing

- Function/feature (external) checklists:
  - ▷ Black-box in nature
  - ▷ List of major functions expected
  - ▷ Example: Table 8.1 (p.105)

- Implementation (internal) checklists:
  - ▷ White-box in nature
  - ▷ At different levels of abstraction
    – e.g., lists of modules/components/etc.

- Related: cross-cutting features/structures:
  - ▷ Multiple elements involved.
  - ▷ Examples: call-pairs, diff. parts that cooperate/collaborate/communicate/etc.

Checklists for Testing

- Other checklists:
  - ▷ Related to certain properties
    - e.g., coding standards,
  - ▷ Combining (esp. for large products):
    - hierarchical list, e.g., refined Table 8.1
    - "X"-like, e.g., Table 8.2 (p.106)

- Possible drawbacks:
  - ▷ Coverage: need to fill "hole".
  - ▷ Duplication: need to improve efficiency.
  - ▷ Complex interactions not modeled.
  - ▷ Solutions: Partitions and FSMs.

Checklists to Partitions

- Partitions: a special type of checklists
  - ▷ Mutually exclusive ⇒ no overlaps.
  - ▷ Collectively exhaustive ⇒ coverage.
  - ▷ Address two problems of checklists. (Third addressed by FSMs in Ch.10.)

- Motivational examples:
  - ▷ Solution to: $ax^2 + bx + c = 0$,

$$r = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}.$$

  - ▷ Input: a, b, c; Output: r.
  - ▷ 32 bits floating point numbers.
  - ▷ Input combinations:

$$2^{32} \times 2^{32} \times 2^{32} = 2^{96}$$

  - ▷ Reduce to 3 partitions: Table 8.3 (p.108)

Partitions: Formal Definitions

- Partition of set S into subsets $G_1,\ G_2,\ \ldots,\ G_n$ ($G_i \subset S$):
  - ▷ G_i's are mutually exclusive:

$$\forall i, j,\ i \neq j \Rightarrow G_i \cap G_j = \emptyset$$

  - ▷ G_i's are collectively exhaustive:

$$\bigcup_{i=1}^{n} G_i = S.$$

- Each G_i forms an equivalent class.
  - ▷ Formal conditions sometimes possible: – formally defined by relations (next).
  - ▷ Often implicit by membership to G_i

Partitions: Formal Definitions

- Relation: An association of interest to some observers among objects.
  - ▷ $\mathcal{R}(A_1, A_2, \ldots, A_n)$
  - ▷ *Binary* relations: $\mathcal{R}(A, B)$ or $A\mathcal{R}B$. most commonly used relations.

- Relational properties
  - ▷ *Transitivity:* $A\mathcal{R}B \wedge B\mathcal{R}C \Rightarrow A\mathcal{R}C$ e.g., "$>$" relation.
  - ▷ *Symmetry:* $A\mathcal{R}B \wedge B\mathcal{R}A$ e.g., "is-neighbor-to" relation.
  - ▷ *Reflexivity:* $A\mathcal{R}A$ e.g., "$=$" relation.

- *Equivalence relation:* All the above properties hold.

Partition-Based Testing

- Basic idea:
  - ▷ Sampling from partitioned subsets
  - ▷ Coverage of partitions: uniform
  - ▷ Testing based on related problems:
    - – usage-related problems (later)
    - – boundary problems (Ch.9)

- Different types of partitions and related partition-based testing:
  - ▷ Pure membership based partitions:
    - – e.g., components in a subsystems
    - – direct sampling, e.g., one component from each subsystem for coverage
  - ▷ Properties/relations used in definitions:
    - – direct predicates on logical variables
    - – vs. operations on numerical variables
  - ▷ Combinations
  - ▷ Testing for latter two: Next

Partition-Based Testing

- Testing predicates on logical variables:
  - ▷ Logical variable P as input.
  - ▷ Two partitions/test-case: P=T, P=F.
  - ▷ $P \wedge Q$, with two partitions (outcomes).
  - ▷ $P \wedge Q = T$, with $P = T$ and $Q = T$.
  - ▷ $P \wedge Q = F$, one test case selected from three pairs: (P=T, Q=F); (P=F, Q=T); (P=F, Q=F).

- Testing comparisons on numerical variables and combinations:
  - ▷ $x > 0$, many possible test cases.
  - ▷ Combination similar to above, e.g.,
    - $(x > 0) \wedge (y < 100)$, select x, y values individually;
    - $(x > 0) \wedge (x \leq 100)$, select x value to satisfy both conditions.

Partition-Based Testing

- Testing multiple sets of partitions:
  - ▷ Divide-and-conquer.
  - ▷ Model as stages.
  - ▷ Combination (cross-product) of the stages.
    - – e.g. binary partitions P followed by Q: four combinations: TT, TF, FT, FF.

- General: an m-way partition followed by an n-way partition: $m \times n$ combinations.

- Coordinated sensitization often needed, similar to for $(x > 0) \wedge (x \leq 100)$ above.

Partition-Based Testing

- Extensions to basic ideas:
  - ▷ Sampling from partitioned subsets.
  - ▷ Coverage of partitions: non-uniform?
  - ▷ Testing based on related problems:
    - – usage-related problems?
    - – boundary problems?

- Usage-related problems:
  - ▷ More use ⇒ failures more likely
  - ▷ Usage information in testing
    ⇒ (Musa's) operational profiles (OPs)

- Boundary problems:
  Input domain boundary testing (Ch.9).

Usage-Based Testing

- Usage based statistical testing (UBST) to ensure reliability.

- *Reliability:* Probability of failure-free operation for a specific time period or a given set of input under a specific environment
  - ▷ Reliability: customer view of quality
  - ▷ Probability: statistical modeling
  - ▷ Time/input/environment: OP

- OP: Operational Profile
  - ▷ Quantitative characterization of the way a (software) system will be used.
  - ▷ Generate/execute test cases for UBST
  - ▷ Realistic reliability assessment
  - ▷ Development decisions/priorities

UBST: General Issues

- General steps:
  - ▷ Information collection.
  - ▷ OP construction.
  - ▷ UBST under OP.
  - ▷ Analysis (reliability!) and followup.

- Linkage to development process
  - ▷ Construction: Requirement/specification, and spill over to later phases.
  - ▷ Usage: Testing techniques and SRE

- Procedures for OP construction necessary

OP: Basic Concepts

- Profile: Disjoint alternatives and their associated probabilities.
  - ▷ Key: flat and sum to 1.
  - ▷ Occurrence or weighting factors.
  - ▷ Representation: graphs and tables
    – Table 8.4 (p.112) and Fig 8.1 (p.113).
  - ▷ Different types of profiles.
  - ▷ OP: operational profile.
  - ▷ Often sorted in decreasing probabilities.

- General observations:
  - ▷ Uneven distribution: basis for UBST (otherwise uniform sampling adequate)
  - ▷ #operations$\uparrow\uparrow$ $\Rightarrow$ cutoff threshold.

OP Usage

- Usage of OPs in UBST:
  - ▷ Pure random sampling rare
    - – requires dynamic (on-the-fly) decisions
    - – might interfere with system functions
  - ▷ More often: pre-prepared test cases
    - – "pseudo" randomness
  - ▷ Other variations:
    - – normal cases and then perturbations
    - – use of adjustable thresholds

- OP and SRE (s/w reliability engineering):
  - ▷ SRE assumes OP-based UBST.
  - ▷ OP sometimes directly used in reliability evaluations and improvement.

UBST: Primary Benefit

- Primary benefit:
  - ▷ Overall reliability management.
  - ▷ Focus on high leverage parts
    ⇒ productivity and schedule gains:
    – same effort on most-used parts
    – reduced effort on lesser-used parts
    – reduction of 56% system testing cost
    – or 11.5% overall cost (Musa, 1993)

- Gains vs. savings situations
  - ▷ Savings situation: AT&T (above)
    – reliability goal within reach
    – not to over test lesser-used parts
  - ▷ Gains situation: more typical
    – re-focusing testing effort
    – constrained reliability maximization

UBST: Other Benefits

- Introducing new product
  - ▷ Highly-used features quickly
  - ▷ Lesser-used: subsequent releases

- Better communications/customer relations
  - ▷ Customer perspective & involvement
    ⇒ closer ties to customers
  - ▷ More precise requirement/specification
  - ▷ Better training focus

- High return on investment:
  - ▷ OP cost, "average" 1 staff-month
    – 10 developers, 100KLOC, 18 months
    – sub-linear increase for larger ones
  - ▷ Cost-benefit ratio: 10

Developing OP

- One OP or multiple OPs?
  - ▷ One OP for each homogeneous group of users or operations:
    - – user group or market segmentation
    - – groups of operations (op. modes)
  - ▷ Fundamental differences ⇒ split
  - ▷ Hybrid strategy often useful:
    - – develop separate OPs
    - – merged OP for overall picture
    - – both types offer valuable info.
- Generic methods: Information sources.
  - ▷ Actual measurement.
  - ▷ Customer surveys.
  - ▷ Expert opinion.

Developing OP

- Actual measurement for OP construction:
  - ▷ Most accurate but also most costly.
  - ▷ Limitations for new products.
  - ▷ Legal/IP issues.

- Overcoming difficulties for new products:
  - ▷ Measurement for similar products.
  - ▷ Necessary adjustment.

- Overcoming legal/IP difficulties:
  - ▷ Similar to new product strategy above?
  - ▷ Voluntary participation:
    - "out" participation: beta testing,
    - "in" participation: ECI in IBM
  - ▷ Use of existing logs/records/etc.

Developing OP

- Customer surveys:
  - ▷ Less accurate/costly than measurement.
  - ▷ But without the related difficulties.
  - ▷ Key to statistical validity:
    - – large enough participation
    - – "right" individuals completing surveys
  - ▷ More important to cross-validate
    - – see example study in Section 8.5.
- Expert opinion:
  - ▷ Least accurate and least costly.
  - ▷ Ready availability of internal experts.
  - ▷ Use as a rough starting point.

Developing OP

- Who should develop OP?
  - ▷ System engineers
    – requirement ⇒ specification
  - ▷ High-level designers
    – specification ⇒ product design
  - ▷ Planning and marketing
    – requirement gathering
  - ▷ Test planners (testing)
    – users of OP
  - ▷ Customers (implicitly assumed)
    – as the main information source

- Development procedure (2 variations)
  - ▷ Top-down/Musa-1: (Musa, 1993)
  - ▷ Musa-2: Musa 1998 book (Chapter 3)
  - ▷ Both covered in SQE book.

OP Development: Musa-1

- One OP for each homogeneous group of users or operations.

- General idea:
  - ▷ Top-down: user/usage groups.
  - ▷ Focus: external users and their usage.

- Generic steps:
  1. Find the customer profile.
  2. Establish the user profile.
  3. Define the system modes.
  4. Determine the functional profile.
  5. Determine the operational profile.

- First two steps external view; last three steps internal view.

Musa-1.1: Finding the Customer Profile

- Differentiate customer from users
  - ▷ Customer: acquisition of software
  - ▷ User: using software
- Weight assignment:
  - ▷ By #customers
  - ▷ By importance/marketing concerns, etc.
  - ▷ Example: Table 8.5 (p.118)
- Split or merge?
  - ▷ Fundamental differences: split.
  - ▷ Else, use weighting factors to merge.

Musa-1.2: Establishing the User Profile

- Breakdown of customer groups
  - ▷ Different usages of user groups
  - ▷ Merging similar users across customers

- Weighting factor assignment and comprehensive user profile derivation:
  - ▷ User weights within customers:
    - – by users (equal usage intensity)
    - – by usage frequency
  - ▷ Comprehensive: weighted sum
  - ▷ Example: Table 8.6 (p.119)

Musa-1.3: Defining System Modes

- System mode
  - ▷ A set of functions/operations
  - ▷ For operational behavior analysis
  - ▷ Practicality: expert for system mode
- Example modes
  - ▷ Business use mode
  - ▷ Personal use mode
  - ▷ Attendant mode
  - ▷ System administration mode
  - ▷ Maintenance mode
  - ▷ Probabilities (weighting factors)

Musa-1.4: Determining Functional Profile

- Identifying functions
  - ▷ Function: high-level task/work of the projected system in the requirement.
  - ▷ Input domain partitions/combinations
  - ▷ Hardware/OS/system configuration
  - ▷ Base on environmental variables

- Creating/consolidating function list
  - ▷ From system requirement
  - ▷ From prototypes/previous release/user manual etc.

- Determining occurrence probabilities
  - ▷ Measurement and adjustment
  - ▷ Functions $\Leftrightarrow$ operations

Musa-1.5: Determining OP

- Refining functional profile into OP
- Defining operations
  - ▷ Operation: implemented task/work that can be used as part of system test plan
  - ▷ Defining the input space
  - ▷ Partitioning input space into operations
  - ▷ Typically: 1 function $\Rightarrow$ n operations
- Obtaining occurrence probabilities
  - ▷ In-field measurement
  - ▷ Estimation for new systems or added functionalities using symbolic models or prototypes
  - ▷ Help from functional probabilities

OP Development: Musa-2

- One OP for each operational mode (testing under specific modes in practice)
- General idea:
  - ▷ Op. group: coarse → fine → individual.
  - ▷ Focus: internal users (testers).
- Generic steps:
  1. Identify initiators of operations.
  2. Tabular or graphical representation.
  3. Operations lists: initiators → consolidated.
  4. Determine the occurrence rate.
  5. Determine the occurrence probability.

OP Development: Musa-2

1. Identify initiators of operations
   - ▷ Who are the users of the system? human users, other hw/sw/network/etc.
   - ▷ Consolidate across organizations or customer types

2. Tabular vs graphical representation
   - ▷ Tabular: operation-probability pairs.
   - ▷ Graphical: stages/steps of operation
     – operation = a path in graph/tree
     – probability for branching
       (joint prob = product of indiv. prob.)
   - ▷ Example: Fig 8.2 (p.121)

OP Development: Musa-2

3. Operations lists:
   - ▷ Initiators ⇒ indiv. op. lists
   - ▷ Consolidation ⇒ overall op. lists
   - ▷ Proper granularity adjustment:
     – possible split/merge

4. Determine the occurrence rate
   - ▷ Measurement (and survey?)
   - ▷ Tabulation

5. Determine the occurrence probability
   - ▷ Normalized occurrence rate
   - ▷ $0 \leq p_i \leq 1$ and $\sum_i p_i = 1$

Comparison: Musa-1 vs. Musa-2

- Generic steps:
  - ▷ Musa-1: customer → user → sys. modes → functional → operational
  - ▷ Musa-2: initiator → representation → list → rate → probability

- Comparison
  - ▷ Size/environment/population differences.
  - ▷ One OP for each distinguished group
    - – Musa-1: user or operation group,
    - – Musa-2: operational modes.
  - ▷ Musa-1: 5 profiles, refined along.
  - ▷ Musa-2: different elements for 1 profile.

OP Construction: A Case Study

- Background:
  - ▷ Former CSE 5314 student
  - ▷ Course project: OP development
  - ▷ Application of Musa-1
  - ▷ Chruscielski/Tian: ISSRE'97 paper (IEEE-ISSRE'97 best paper award)

- Problem and key decisions:
  - ▷ Product: LMTAS/CSS
  - ▷ Product characteristics ⇒ OP type
    - – menu selection/classification type
    - – flat instead of Markovian
  - ▷ Result OP, validation, and application

OP Case Study

- Participants:
  - ▷ Software Product Manager
  - ▷ Test Engineers
  - ▷ Systems Engineers
  - ▷ Customers
  - ▷ Chruscielski: pulling it together
  - ▷ Tian: technical advising
  - ▷ Chruscielski/Tian: documentation

- Information gathering
  - ▷ Interview Software Product Manager to identify target customers
  - ▷ Customer survey/questionnaire to obtain customer usage information
  - ▷ Preparation, OP construction and followup

OP Case Study

- Customer profile:
  - ▷ US Air Force and other AFs
  - ▷ Similar customers/usage ⇒ one OP

- User profile: Table 8.7 (p.123)
  - ▷ User groups & marketing concerns.
  - ▷ Profile reflects both.
  - ▷ Idea applicable to other steps:
    - – profile can be importance weighted,
    - – trade-off impossible ⇒ dedicated OP.

- System modes
  - ▷ No significant difference in op.
  - ▷ Directly proceed to functional profile
  - ▷ General: some step may be by-passed

OP Case Study

- Functional/operational profile:
  - ▷ CSS: functions ≈ operations
  - ▷ Flat structure/choices
  - ▷ Implicit profile possible
  - ▷ Functional list
  - ▷ OPs: for both individual user groups and comprehensive

- Analysis and followup
  - ▷ Cross-validation: Peer review by Software Product Manager, System Engineers and Test Engineers
  - ▷ Classification of usage frequencies
    – Table 8.8 (p.134) found to be useful.
  - ▷ Followup actions

Alternative Usage Models

- Motivation: enhance flat OP
  - ▷ Complicated operations involve many steps/stages in the end-to-end chain
  - ▷ Ability to use existing models and structural information
  - ▷ Ability to use localized knowledge
  - ▷ Local information easy to gather

- Markov OP: Basic ideas
  - ▷ Markov chain for usage information
  - ▷ State: operations/functions
  - ▷ Transition: probabilistic
    - – reflects usage sequence/frequency
    - – history independent (Markovian)
    - – but reflects local usage info.
  - ▷ Details in Chapter 10.