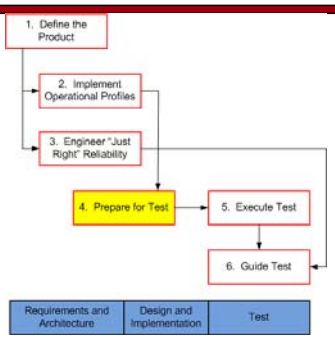
SRE: Preparing for Test



This lecture provides reference material for Chapter 4 of the book entitled "Software Reliability Engineering: More Reliable Software Faster and Cheaper" by John D. Musa © 2004

For PowerPoint version of the slides, contact Laurie Williams at williams@csc.ncsu.edu.

Preparing for test

- Consists of . . .
  - Preparing test cases and test procedures
  - Planning for automated tools
- To guide (in this order) . . .
  - Feature test – single executions of operations, with interactions between operation minimized. (Does the operation execute properly?)
  - Load test; performance test – attempt to represent field use and environment, with operations executing simultaneously and interacting
  - Regression test – feature test you conduct after each build involving significant change to reveal faults that may have spawned in the process of change

Some definitions

- Remember:
  - Operational profile: a complete set of operations with their probabilities of occurrence
- Test operational profile: operational profile modified to ensure the critical operations are tested adequately and to account for reused operations
- Test procedure: test controller for load test that sets up environmental conditions and invokes at various times test cases it randomly selects from the test case set based upon the test operational profile

Preparing for Test

- For each base product and variation:
  - Specify the new test cases for new operations for the current release, based on the operational profile
  - Specify the test procedure, based upon the test operational profile and traffic level
- Provide for:
  - Setup and invocation of test cases
  - Recording each run executed (operation, test case, invocation time and outputs)
  - Recording the number of natural or time units at frequent intervals so we can establish when failures occurred
  - Cleanup

Specify the new test cases for new operations for the current release

1. Estimate the number of new test cases that will be available for the current release.
2. Allocate the new test cases among the base product and variations.
3. Distribute the test cases of the base product and each variation among its new operations.
4. Detail the new test cases for each new operation
   - An operation is "new" if it has been modified in any nontrivial way (that might introduce one or more faults).

Then: add new test cases to test cases from previous releases and remove test cases of operations that might have been removed from this release

Step one: Planning the number of new test cases for current release

- Estimate the number you need based on:
  - History of your project's test cases/line of executable code
  - Industry data (depends on failure intensity reduction objective):
    - 2-3 test cases/thousand lines of code for moderate reliability
    - 20-33 test cases/thousand lines of code for high reliability
  - If most of the product involves reused operations (> 50%), use (N/R)*T
    - where N=occurrence probability of new operations, R=occurrence probability of reused operations, [N=R=1]; T=quantity of test cases from prior releases
- Estimate the number you have the capacity to prepare
  - Industry data: 0.4-16 hours/test cases (preparation)
- Determine the number you WILL prepare
  - Minimum of above two
  - At least one test case per operation! Combine operations?
  - Big discrepancy? Time to negotiate

Step two: Allocating new test cases

- U = F x S
  - U = unique new use
  - F = expected fraction of field use for the associated system vs. total use
  - S = sum of occurrence probabilities of the new operations that are functionally different from those of previously-considered associated systems

Table 4.3 Allocating the number of new test cases among the base product and its variations for Fone Follower

Associated system	*F*	*S*	*U*	*L*	*C*
Fone Follower	0.6	1	0.6	0.833	500
Fone Follower Japan	0.4	0.3	0.12	0.167	100

  - L = new test case allocation fraction = U/total U
    - Fone Follower = 0.6/0.72 = 0.833; Fone Follower-Japan = 0.12/0/72 = 0.167
  - C = number of new test cases = L x total number of test cases

Table from Musa, J., *Software Reliability Engineering*, 2004.

Step three: Distributing new test cases among new operations

- Based upon occurrence proportion for new operation
  - Proportion of occurrences of new operation with respect to occurrences of all new operations for a release.
  - First release: occurrence proportion = occurrence probability
  - Future releases: occurrence probability of operation/sum of occurrence probabilities of all new operations

Table 4.4 Occurrence proportions for Fone Follower base product

Operation	Occurrence probability	Occurrence proportion
Process voice call, no pager, answer	0.21	0.21
Process voice call, pager, answer	0.19	0.19
Process fax call	0.17	0.17
Process voice call, pager, answer on page	0.13	0.13
Process voice call, no pager, no answer	0.10	0.10
Process voice call, pager, no answer on page	0.10	0.10
Enter forwardees	0.09	0.09
Audit section - phone number database	0.009	0.009
Add subscriber	0.0005	0.0005
Delete subscriber	0.0005	0.0005
Recover from hardware failure	0.000001	0.000001
Total	1	1

Table 4.5 Occurrence proportions for Fone Follower 2 base product

Operation	Occurrence probability	Occurrence proportion
Process voice call, no pager, answer	0.168	0
Process voice call, pager, answer	0.152	0
Process fax call	0.136	0
Process voice call, pager, answer on page	0.104	0
Y	0.1	0.5
Z	0.1	0.5
Process voice call, no pager, no answer	0.08	0
Process voice call, pager, no answer on page	0.08	0
Enter forwardees	0.072	0
Audit section - phone number database	0.0072	0
Add subscriber	0.0004	0
Delete subscriber	0.0004	0
Recover from hardware failure	0.0000008	0
Total	1	1

Tables from Musa, J., *Software Reliability Engineering*, 2004.

Step three: More considerations

- Must have at least one test case per operation

Table 4.7 Setting test case minimums for Fone Follower base product

Operation	Occurrence proportion	Adjusted new test cases
Process voice call, no pager, answer	0.21	105
Process voice call, pager, answer	0.19	95
Process fax call	0.17	85
Process voice call, pager, answer on page	0.13	65
Process voice call, no pager, no answer	0.10	50
Process voice call, pager, no answer on page	0.10	50
Enter forwardees	0.09	45
Audit section – phone number database	0.009	5
Add subscriber	0.0005	1
Delete subscriber	0.0005	1
Recover from hardware failure	0.000001	1
Total	1	503

Table from Musa, J., *Software Reliability Engineering*, 2004.

Step three: And more considerations

- Critical operations – one for which successful execution adds a great deal of extra value and failure causes a great deal of impact with respect to human life, cost or system capability
- Acceleration factor (A):
  - FIO (system)/FIO (operation)
- Example:
  - FIO system = 100 failures/Mcalls
  - FIO "recover from hardware failures" = .025 failures/Mcalls
  - Acceleration factor (A)=4000
  - Test cases=
    - (500)(0.000001)(4000) = 2

Table 4.8 Distributing test cases for critical operation of Fone Follower

Operation	Occurrence proportion	Modified new test cases
Process voice call, no pager, answer	0.21	105
Process voice call, pager, answer	0.19	95
Process fax call	0.17	85
Process voice call, pager, answer on page	0.13	65
Process voice call, no pager, no answer	0.10	50
Process voice call, pager, no answer on page	0.10	50
Enter forwardees	0.09	45
Audit section – phone number database	0.009	5
Add subscriber	0.0005	1
Delete subscriber	0.0005	1
Recover from hardware failure	0.000001	2
Total	1	504

Table from Musa, J., *Software Reliability Engineering*, 2004.

Step three: What if there are too many test cases now?

- Try to do them all
- Redistribute by ratio:
  - Original number of test cases/new number of test cases
  - Make sure you don't go below one test case per operation
  - Combine operations if you need to
- Use your judgment if the number of equivalence classes for an operation is much lower than the number of test cases allocated to that operation
  - Equivalence class is a subset of runs of the operation that should yield the same failure behavior because identical processing occurs for them provided that execution involves identical sets of levels of indirect input variables.

Step four: Detailing test cases

- Determine equivalence classes for each operation and select input conditions for the equivalence class
- If there are more combinations of equivalence classes of direct input variable than test cases . . . Randomly select equivalence classes within operations with equal probability OR split the operations

Preparing test procedures

Table 4.11 Fone Follower – modifying occurrence probabilities for critical operations

Operation	Occurrence probability	Modified occurrence probability	Normalized modified occurrence probability
Process voice call, no pager, answer	0.168	0.168	0.1675
Process voice call, pager, answer	0.152	0.152	0.1515
Process fax call	0.136	0.136	0.1356
Process voice call, pager, answer on page	0.104	0.104	0.1037
Y	0.1	0.1	0.0997
Z	0.1	0.1	0.0997
Process voice call, no pager, no answer	0.08	0.08	0.0797
Process voice call, pager, no answer on page	0.08	0.08	0.0797
Enter forwardees	0.072	0.072	0.0718
Audit section – phone number database	0.0072	0.0072	0.0072
Add subscriber	0.0004	0.0004	0.0004
Delete subscriber	0.0004	0.0004	0.0004
Recover from hardware failure	0.0000008	0.0032	0.0032
Total	1	1.0032	1

Table 4.12 Fone Follower – adjusting occurrence probabilities for reused operations

Operation	Normalized modified occurrence probability	Adjusted occurrence probability	Modified occurrence probability
Y	0.0997	0.0997	0.2774
Z	0.0997	0.0997	0.2774
Process voice call, no pager, answer	0.1675	0.0335	0.0932
Process voice call, pager, answer	0.1515	0.0303	0.0843
Process fax call	0.1356	0.0271	0.0754
Process voice call, pager, answer on page	0.1037	0.0207	0.0576
Process voice call, no pager, no answer	0.0797	0.0159	0.0442
Process voice call, pager, no answer on page	0.0797	0.0159	0.0442
Enter forwardees	0.0718	0.0144	0.0401
Audit section – phone number database	0.0072	0.0014	0.0039
Recover from hardware failure	0.0032	0.0006	0.0017
Add subscriber	0.0004	0.0001	0.0003
Delete subscriber	0.0004	0.0001	0.0003
Total	1	0.3594	1

0.2

Table from Musa, J., *Software Reliability Engineering*, 2004.

Summary

- Test planning and test case execution takes time/resources
- Time/resources are limited.
- A test operational profile is used to allocate the limited amount of test cases that can be run such that
  - the faults most likely to be found by customers will be detected; and
  - critical functionalities are tested