Risk based testing
How to choose what to test more and less

by Hans Schaefer
hans.schaefer@ieee.org

- What is risk
- Factors determining damage
- Factors determining probability
- A simple method to calculate risk
- Risk management in test projects: Risks before, during and after the test
The duty of testing

“It shall be the duty of managers to make decisions and the duty of engineers to make them informed ones.”

Jukka Talvio, Development Manager, F-Secure
Why this presentation

Because testing is always under pressure
Testing is the last thing done in a project (“caboose effect”)
You must be able to cut down the least important things
Use optimal amount of resources for testing
Test Effort

Cost of loss / damage

Cost of testing

Level of assurance

GOAL

Cost of loss / damage
Strategy

Objective: Find the most important defects as early as possible at the lowest price

No risk -> No test

Business / user / client based decision
What is risk?

The product of the **probability**, that something negative, a failure, will happen, and the **cost**, (damage) of the consequences which will then happen.

Risk := You don’t know what will happen but you do know the probabilities.

Uncertainty = You don’t even know the probabilities.
Determining probability of failure

Probability of failure

Quality
(failure rate / defect density)

Functional volume
(how much is “in there”)

Probability = defect density / volume
Risk definition

- **Damage**
  - Catastrophic: Loss of lives, loss of license
  - Financial, loss of (faith of) clients, damage to corporate identity
  - Impact on other functions or systems
  - Detection and repair time

- **Probability of failure**
  - Black box = complexity or (estimated) size
  - White box = knowledge of development project (just before testing)

- **Risk = Damage * Probability**

\[
R(f) = P(f) \times C(f)
\]
Risk handling

Risk

Do something with it
Minimize loss

Prevent loss
Avoid QA QM methods

Fight loss
Review Test V&V

Ignore for now
Pay for loss

Retention
React later Pay damage Repair Reserves

Transfer
Alternatives Insurance

© 2005 Hans Schaefer
Risk Analysis and Testing

- Test Plan
  - Test Requirements
    - Risk Identification
    - Risk Strategy
    - Risk Assessment
  - Risk Mitigation
  - Risk Reporting
  - Risk Prediction
- Testing, Inspection etc.
- Matrix: Cost and Probability
  - Test Measurement Data

Risk based testing
Risk analysis

- Applicable on ANY level

- Fundamental problems:
  - Difficult to measure
  - Failure to account for risk compensation (people compensate for greater safety by taking more risks)
Risk analysis

- Risk analysis should lead to a **limited number of classes of approximately equal risks (3-5)**

- **Quality characteristics:** What is the probability that failures will happen and the damage for
  - functional defects
  - bad performance
  - bad usability
  - low maintainability
  - ...

ISO/IEC Std 9126 as checklist
Risk based Test - Practice

First time through
Before the Test: Identify what is critical

1. “Top-risk list”

Test identifies areas with lots of detects

2.

Second time through: Extra Testing:
- Extra Test
- automated regression test

3.
Prioritization for the first test
First Black Box Analysis: High Level Prioritization: Determine the relative importance of quality characteristics

*(Depending on value and possible damage)*.

<table>
<thead>
<tr>
<th>Quality Characteristics</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Functionality</td>
<td>50</td>
</tr>
<tr>
<td>Reliability</td>
<td>20</td>
</tr>
<tr>
<td>Usability</td>
<td>20</td>
</tr>
<tr>
<td>Efficiency</td>
<td>5</td>
</tr>
<tr>
<td>Maintainability</td>
<td>5</td>
</tr>
<tr>
<td>Portability</td>
<td>0</td>
</tr>
</tbody>
</table>
Product Risks: What to think about

Damage factors:

- Which functions and attributes are critical? (essential for the business success to reduce the business risk).
- How visible is a problem in a function or attribute? (for customers, users, people outside)
- How often is a function used?
- Can we do without?
- Legal consequences
- Plain cost
Failure probability:
What is (presumably) worst?

- Complex areas
- Changed areas
- Number of people involved
- Turnover
- New technology, solutions, methods
- New tools
- Time pressure
- Areas which needed optimizing
- Areas with many defects before
- Geographical spread
- History of prior use
- Local factors
Do not forget

Can we test ONLY PART of the product?

Other versions later?

Fight time pressure!
How to calculate priority of risk areas?

Assign weights to the chosen factors. (1 - 3 - 10)
Assign points to every area and factor  
(1 - 2 - 3 - 4 - 5)
Calculate the weighted sum (damage * probability).
Damage may even be in absolute terms!

Spreadsheet
Download: http://home.c2i.net/schaefer/testing/riskcalc.hqx
Example

Risk based testing

<table>
<thead>
<tr>
<th>Area to test</th>
<th>Usage frequency</th>
<th>Visibility</th>
<th>Complexity</th>
<th>Geography</th>
<th>Turnover</th>
<th>SUM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weight</td>
<td>3</td>
<td>10</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>Function A</td>
<td>5</td>
<td>3</td>
<td>2</td>
<td>4</td>
<td>5</td>
<td>1125</td>
</tr>
<tr>
<td>Function A perf.</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>1530</td>
</tr>
<tr>
<td>Function B</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>5</td>
<td>368</td>
</tr>
<tr>
<td>F B usability</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>2</td>
<td>5</td>
<td>377</td>
</tr>
<tr>
<td>Function C</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>572</td>
</tr>
<tr>
<td>Function D</td>
<td>5</td>
<td>0</td>
<td>4</td>
<td>1</td>
<td>1</td>
<td>240</td>
</tr>
</tbody>
</table>

Functional volume ignored in this example.
What is the formula?

Risk = Damage * Probability

Damage =
(Weight for impact factor 1 * value for this factor +
Weight for impact factor 2 * value for this factor + + +
Weight for impact factor n * value for this factor)

Probability =
((Weight for probability factor 1 * value for this factor +
Weight for probability factor 2 * value for this factor + + +
Weight for probability factor n * value for this factor) /
functional volume)
The mathematics behind it

It works well enough.
We may actually be on a logarithmic scale (humans assigning points do so), which means we should ADD instead of MULTIPLY.

The highest weighted sums -> thorough testing
Middle weighted sums -> ordinary testing
Low weighted sums -> light testing

Make sure you use your head! Analyze unexpected results!
Selecting test techniques

Subsystem X, Example

<table>
<thead>
<tr>
<th>Reliability</th>
<th>high</th>
<th>State trans test Boundary value, branch coverage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reliability</td>
<td>low</td>
<td>One positive, one negative test case</td>
</tr>
<tr>
<td>Efficiency</td>
<td>high</td>
<td>Reliatic load test, stress test</td>
</tr>
<tr>
<td>Efficiency</td>
<td>low</td>
<td>No test</td>
</tr>
</tbody>
</table>
What to do if you do not know anything about the product?

Run a test.
Prioritize roughly by risk.

First a breadth test ("smoke test"), everything a little, risky items more. (Explore the product).
Then prioritize a more thorough test for the second test cycle.
Prioritization of further test cycles

Fault- and Coverage analysis
Analysis of defect detection percentage
Adaptive Testing - second test

Effort = length of bar

Test Area 5
Test Area 4
Test Area 3
Test Area 2
Test Area 1

Time

\( t_1 \)

\( t_2 \)

Time
Analysis of test coverage

Have all (important) functions been covered?
Exception handling?
States and transitions?
Important non functional requirements?

Is test coverage as planned?

Extra Check or Test where coverage differs from expected coverage!
How to analyze your test

Coverage against expected coverage

Is the code coverage under test as expected?

If some area is executed a lot more than expected, is that a symptom for performance problems? Bottleneck, error?

If an area was covered less than expected, is that area superfluous, or was the specification too “thin”?

Do an extra inspection of such areas!
Analysis of fault density

Facts:

Testing does not find all faults.
The more you find, the more are left.
Post-release fault density correlates with test fault density!

Defect prone units:
A Pareto distribution.
NSA: 90% of high severity failures come from 2.5% of the units.
Others: Typically 80% failures from 20% of the units.

Defects are social creatures, they tend to keep together!
What to use fault density for

- Measure the number of faults / 1000 lines of code.
- Compare with your own average.
- Spend extra analysis or test if the program under test is bad.
- Spend extra analysis if the program under test is “too good”.
Analysis of causes

If you have many defects with the same cause category, think about improving your way of working!

Typical for unit testing:
- Logic
- Computation
- Interfacing
- Data handling
- Input data problem
- Documentation
- Change
Analysis of defect detection

How effective is the already planned or done defect detection?
Or: How much chance is there that defects survive?

Probability for defects decreases:

New risk = old risk / detection percentage

Defect detection percentage = defects found / defects before detection measure * 100%
Another risk based approach: Project risks for the Tester

Risks BEFORE Test
Risks DURING Test
Risks AFTER Test
Risks BEFORE Testing

Bad Quality
- Many faults overlooked
- Blocking faults
- Too many new versions

  -> Requirements to, and follow up of quality assurance before test

Delays
  -> Alternative plans

Lack of knowledge
  -> Test of earlier versions
Risks AFTER Testing

THESE SHOULD NOT HAPPEN…

Customer finds faults.
Customer uses the product in new ways.

Analysis of necessary reliability!
Risks in the Test project itself

Bad management
Lack of qualification
Too few or the wrong people, too late
Bad coordination
Bad cooperation
Problems with equipment and tools

Medicine: Normal good project management.
How to make testing cheaper?

Good people save time and money
Good Prioritization

Try to get rid of part of the task...
Getting rid of work

Get someone else to pay for it or cut it out completely!
  - Who pays for unit testing?
  - What about test entry criteria?
  - Less documentation - more exploratory test

Cutting installation cost - strategies for defect repair
  - When to correct a defect, when not?
  - Rule 1: Repair only defects causing important failures!
  - Rule 2: Change requests to next release!
  - Rule 3: Install corrections in groups!
  - Rule 4: Daily build!

Less Test, should the customers pay ????
Test reporting, risks and benefits

**Diagram:**
- Testing → Risks (addresses)
- Testing → Benefits (demonstrates)
- Risks → Project status (threaten)
- Benefits → Project status (informs about)
Risk-based reporting

Risk-based reporting involves tracking residual risks from the start of the project. All risks are marked as 'open' at the beginning, with some high-priority risks (marked with a star) being addressed immediately.

The progress through the test plan is illustrated with a graph showing the planned end, today's status, and residual risks of releasing TODAY.
Risk-based reporting - Benefits

Progress through the test plan

Star

Benefits 'unachieved' at the start

Residual benefits not yet realized if releasing TODAY

Today

Planned end

Benefits

Risk based testing

© 2005 Hans Schaefer

Slide no. 41
References

IEEE Standard 1044-2002: Standard Classification for Software Anomalies
-You find them at sales@ieee.org

Rex Black, Managing the Testing Process, John Wiley, 2002. (includes CD with a test priority spreadsheet)


  http://home.c2i.net/schaefer/testing/risktest.doc

James Bach, Risk Based Testing, STQEMagazine, Vol1, No. 6,
  www.stqemagazine.com/featured.asp?stamp=1129125440


Thank you for listening

Questions?