Risk based testing

How to choose what to test more and less

by Hans Schaefer

Software Test Consulting, Norway

hans.schaefer@ieee.org

http://home.c2i.net/schaefer/testing.html

- 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
- Some more stuff, if there is time

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

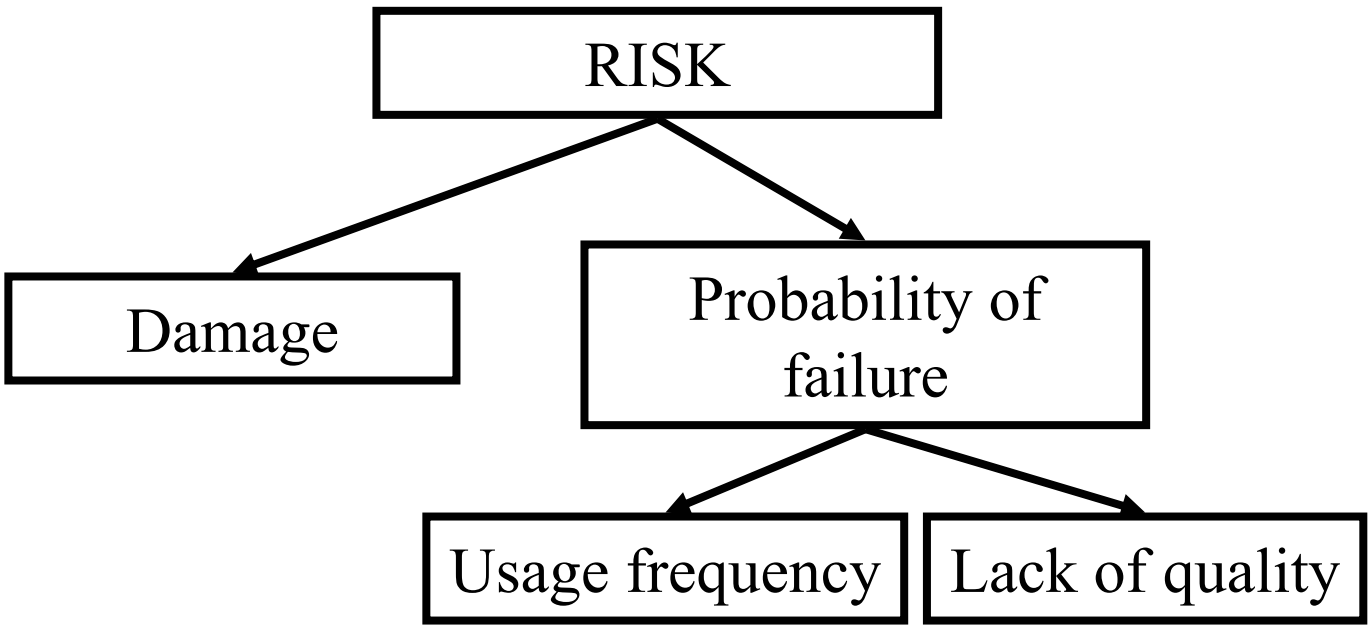
Strategy

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

No risk -> no test

Business based decision

What is risk?



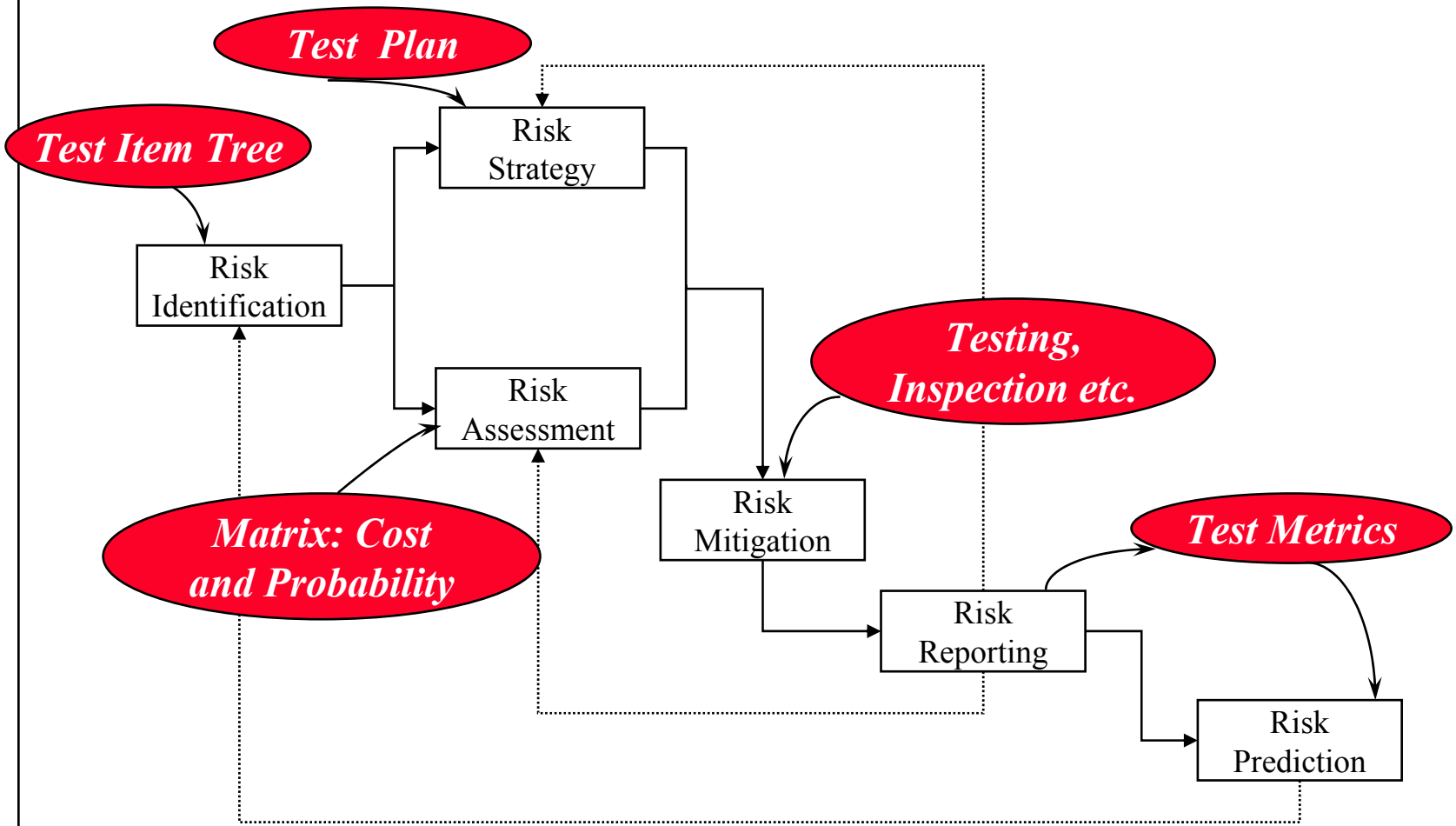
Risk:= You don't know what will happen but you do know the probabilities
Uncertainty = You don't even know the probabilities.

Risk definition

- Damage
  - financial, loss of (faith of) clients, damage to corporate identity
  - impact on other functions or systems
  - detection and repair time
- Probability of failure
  - globally = (estimated) size * complexity
  - in detail = knowledge of development project (just before testing)
- Risk = Damage * Probability

Jukka Talvio von F-Secure: "It shall be the duty of managers to make decisions and the duty of engineers to make them informed ones."

Risk Analysis and Testing



Risk analysis

- Applicable on the level of
  - system
  - subsystem
  - individual function or module **(e.g. insert new entry into phone database)**

Fundamental problems:

- Difficult to measure
- Failure to account for risk compensation (people compensate for greater safety by taking more risks)

Risk analysis

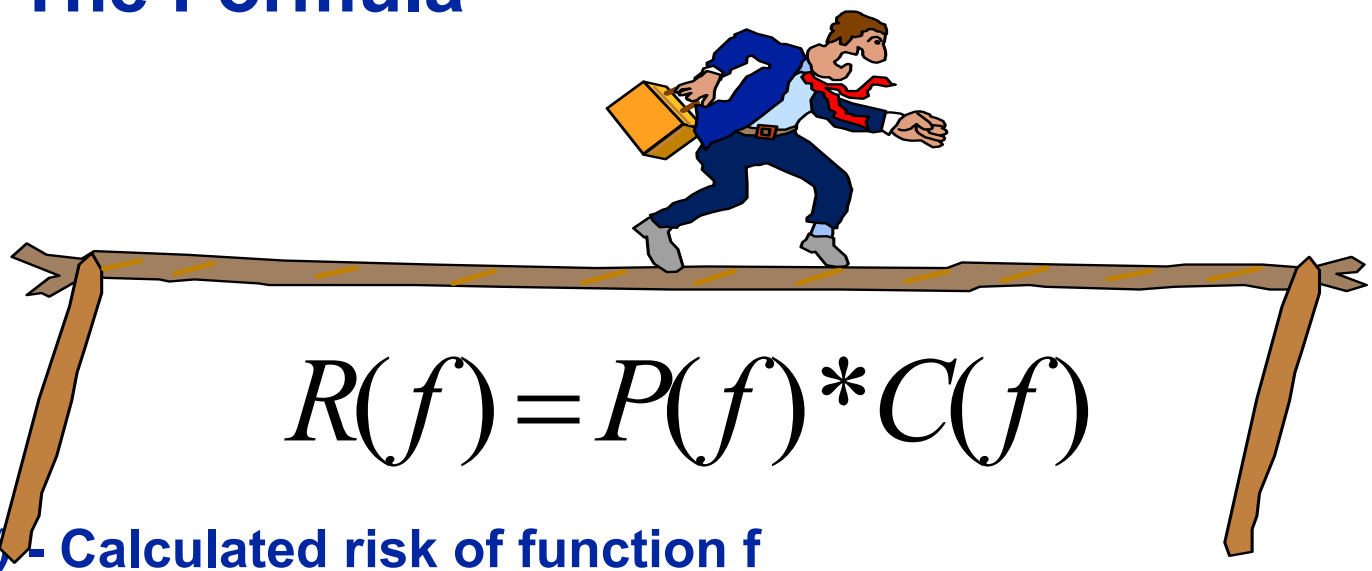
- Risk analysis should lead to a limited number of classes of approximately equal risks

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

See ISO/IEC 9126-1

Risk Based Testing - Theory

- TThe Formula



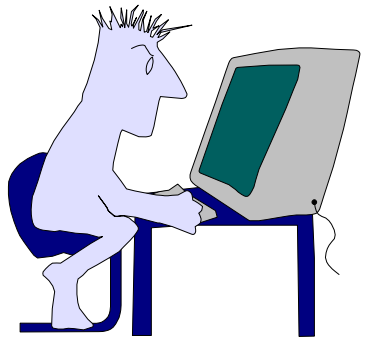
$$R(f) = P(f) * C(f)$$

- *R(f)* - Calculated risk of function f
- *P(f)* - Probability of a fault in function f
- *C(f)* - Cost related to a fault in function f

Risk based Test - Practice

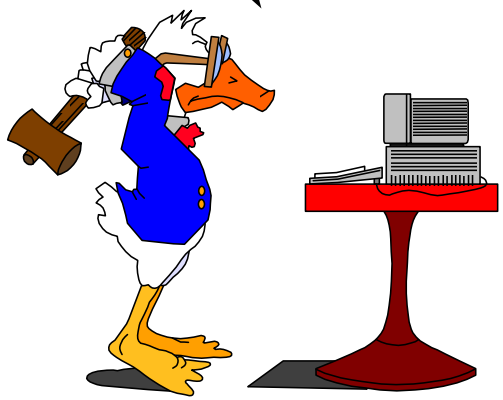
Before the Test: Identify what is critical

1



Test identifies areas with lots of detects

2



Extra Testing:
- Extra Test by product specialist
- automated regression test
-

3

Prioritization for the first test

Product Risks: What to think about

- Which functions and attributes are critical (for the success of the product)?
- 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?

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

(0 - 1 - 2 - 3 - 4 - 5)

Calculate the weighted sum (impact * probability).

The spreadsheet does not contain surprise, but that can be added.

Spreadsheet

http://home.c2i.net/schaefer/testing/Riskcalc.xls

Example

Area to test	Usage frequency	Visibility	Complexity	Geography	Turnover	SUM
Weight	3	10	3	1	3	
Function A	5	3	2	4	5	1125
Function A performanc e	5	3	5	4	5	1530
Function B	2	1	2	2	5	368
F B usability	1	1	4	2	5	377
Function C	4	4	3	2	0	572
Function D	5	0	4	1	1	240

What is the formula?

Risk = Impact * Probability

Impact =
(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)

The mathematics behind it

Actually, we MAY work on a logarithmic scale: We may ADD instead of multiply! (Choosing “points” [1 to 5] intuitively, often leads to logarithmic value assignments)

However: It works well enough.

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!

A shorter method:
Determine the relative importance of quality characteristics

(They depend on value and possible damage).

Functionality	**50**	
Reliability	**20**	
Usability	**20**	
Efficiency	**5**	
Maintainability	**5**	
Portability	**0**	

Selecting test techniques

Subsystem X, Example

Reliability	30	State trans test Boundary value, branch coverage
Usability	40	Paper review, Usability lab
Efficiency	10	No test
Flexibility (maintain)	20	Design review Monitoring of repairs

What to do if you do not know anything about the product?

Run a test.

First a breadth test, everything a little.

Then prioritize a more thorough test for the second test cycle.

Prioritization of further test cycles

Fault- and Coverage analysis

Analysis of test coverage

Have all (important) functions been covered? (Benefits!)

Exception handling?

States and transitions?

Important non functional requirements?

Is test coverage as planned?

Extra Check or Test where coverage differs from expected coverage!

Follow-up of code coverage

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 war 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

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, more parallel work

Lack of knowledge

-> Test of earlier versions

Risks AFTER Testing

THESE SHOULD NOT HAPPEN…

Customer has trouble.

Customer uses the product in new ways.

Analysis of necessary reliability!

Good contact with support!

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 Prioritisation

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!

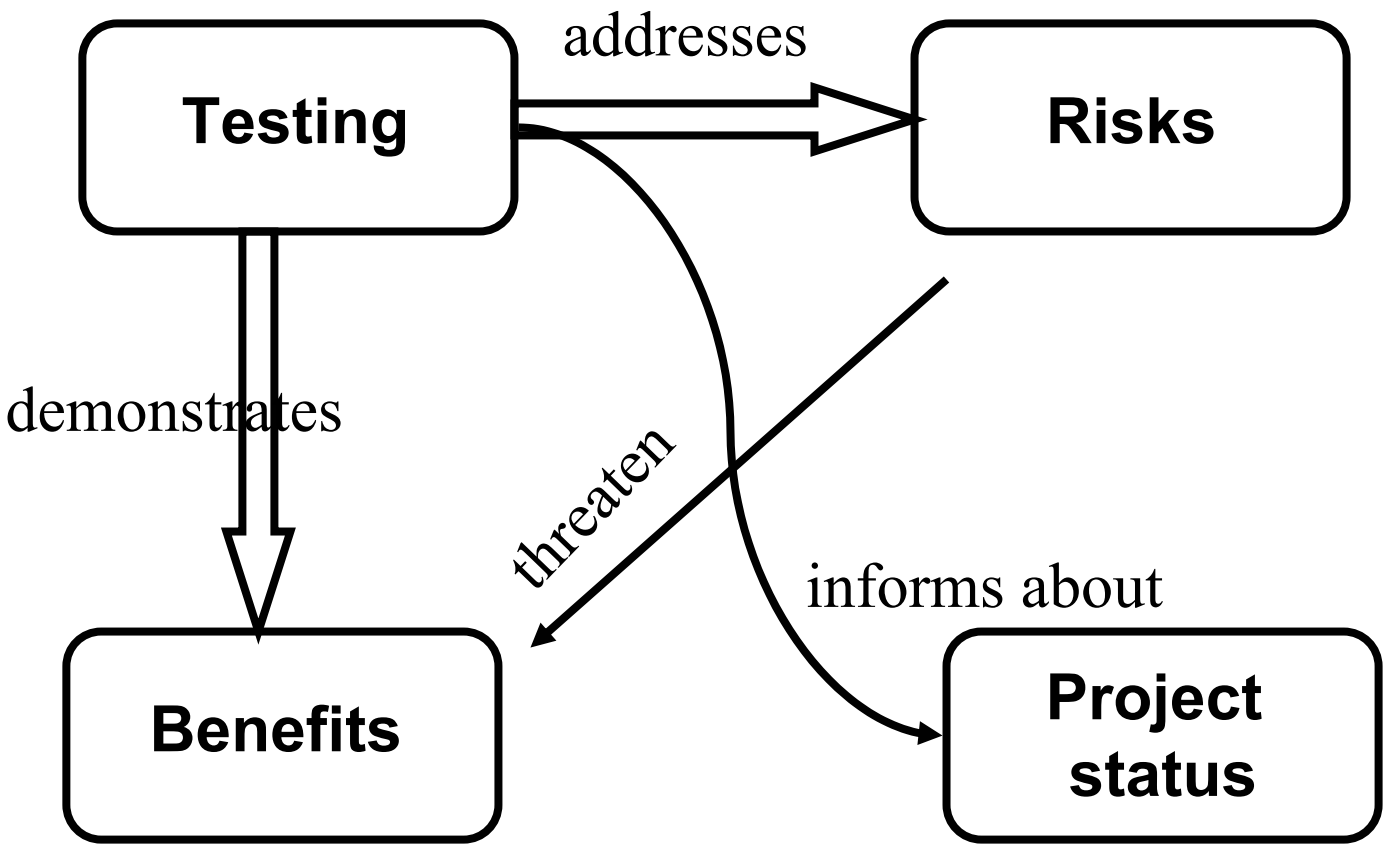
- Who pays for unit testing?
- What about test entry criteria?
- Less documentation

Cutting installation cost - strategies for defect repair

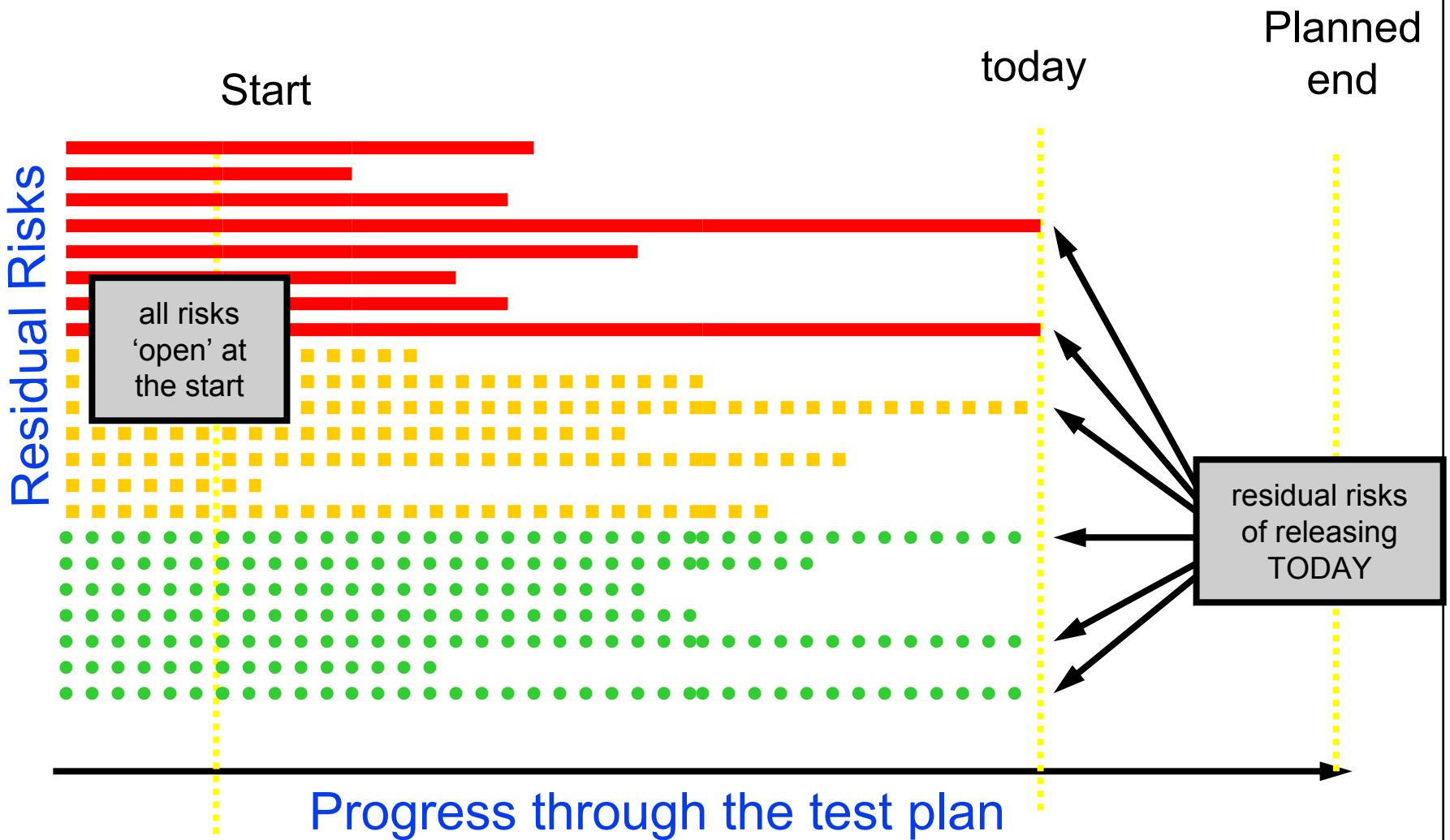
- When to correct a defect, when not?
- Rule 1: Repair only defects causing important failures!
- Rule 2: Change requests to next release!
- Rule 3: Install corrections in groups!
- Rule 4: Daily build!

Less Test, should the customers pay ????

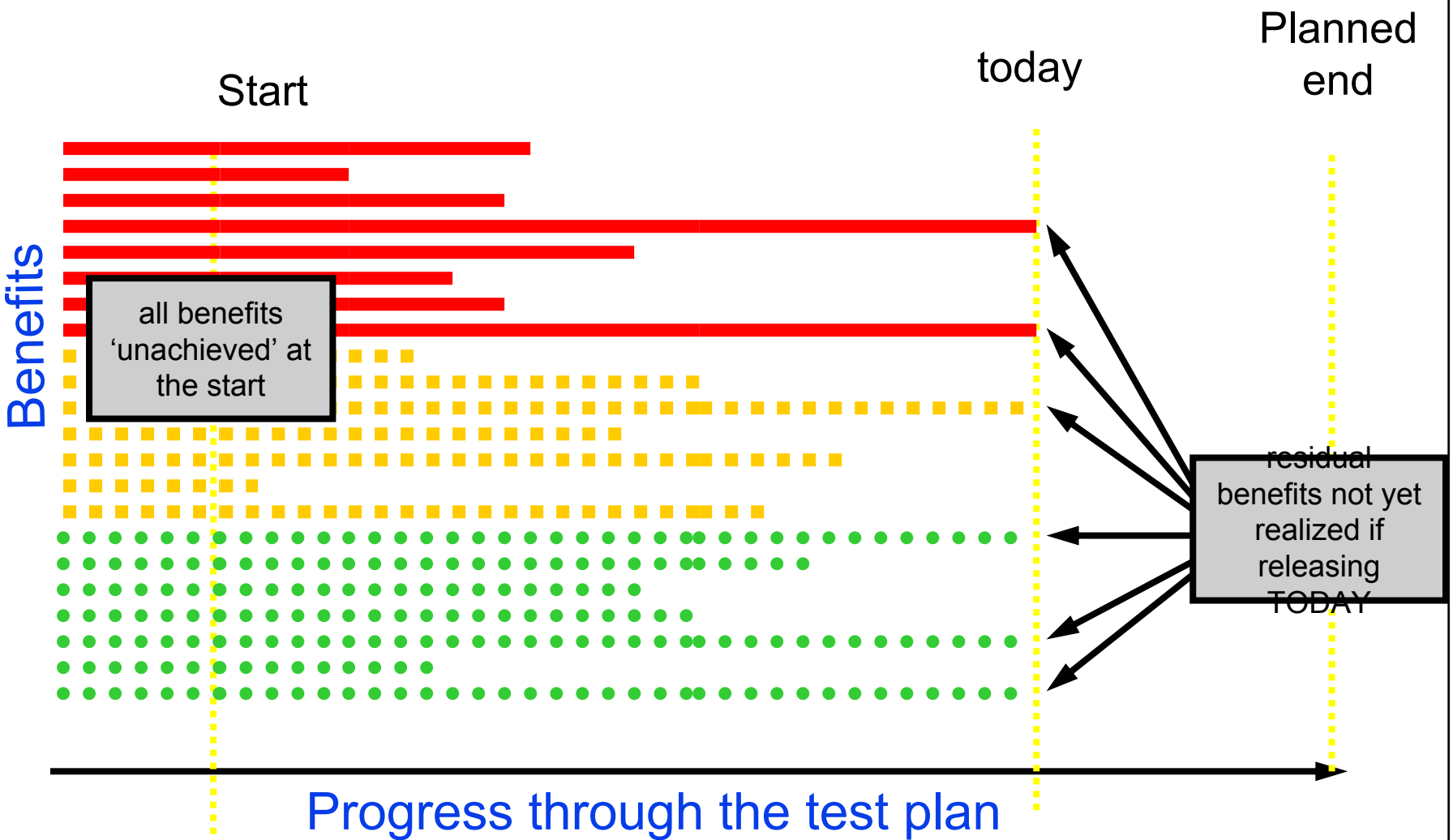
Test reporting, risks and benefits



Risk-based reporting - Risks



Risk-based reporting - Benefits



References

IEEE Standard 1044-2002: Standard Classification for Software Anomalies

IEEE Standard 1044.1-2002: Guide to Classification for Software Anomalies

Soon to come: IEEE Std. 16085 Standard for Software Engineering -
Software Life Cycle Processes - Risk Management

You find them at sales@ieee.org

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

Hall, Payson: A Calculated Gamble. In STQE Magazine No 1 +2 / 2003.

Stamatis, D.H., Failure Mode and Effect Analysis: FMEA from Theory to Execution, Amer Society for Quality, 1995.

Schaefer, Hans: „Strategies for Prioritizing Test“, STAR WEST 1998. http://home.c2i.net/schaefer/testing/risktest.doc

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

Felix Redmill in „Professional Tester“, April 2003. www.professional-tester.com

Tom DeMarco and Tim Lister, "Waltzing with Bears: Managing Risk on Software Projects”, 2003.

www.riskbasedtesting.com

Thanks for listening

Questions?