"Testers Rights: What Test should demand from others, and why?"

Keynote: Eurostar Conference, Amsterdam, December, 2003

Tom Gilb

50 minutes

Talk Outline:

- Testers should not be debuggers
- Testers should not get bad work dumped on them.
- Even if you don't believe we should be kind to testers; the entire project will take less time, and have more quality, if we do not mistreat the test function.
- We must not make testers the scapegoat for our upstream bad development processes.
- To clarify this position I offer a
  - –Testers Bill of Rights.
    - Derived from my Company Communication Bill of Rights, page 23, Principles of Software Engineering Management, 1988. Developed for CEO Wilmott, ICL, UK, 1982

Testers Bill of Rights:

- **1. Testers have the right to sample their process inputs, and reject poor quality work (no entry).**
  2. Testers have the right to unambiguous and clear requirements.
  3. Testers have the right to test evolutionarily; early as the system increments.
  4. Testers have the right to integrate their test specifications into the other technical specifications.
  5. Testers have the right to be a party to setting the quality levels they will test to.
  6. Testers have the right to adequate resources to do their job professionally.
  7. Testers have the right to an even workload, and to have a life.
  8. Testers have the right to specify the consequences of products that they have not been allowed to test properly.
  9. Testers have the right to review any specifications that might impact their work.
  10. Testers have the right to focus on testing of agreed quality products, and to send poor work back to the source.

Why should testers have any rights?

- Main Argument:
  - Because it will reduce total costs, time and increase quality at the same time.
- Real Reason (hidden agenda):
  - To make other project members do their own work properly in the first place.
- Altruistic Reason:
  - to make their workday more meaningful,
  - and to show them some due respect.

What are testers going to *do* with their ‘rights’?

- Use them to negotiate service agreements with the rest of the project
- Use them to train testers in rational expectations
- Use them to set their own test process entry and exit standards
- Use them to enhance their own defined processes
- Use them as a starting argument;
  - when they are ‘mistreated’ by other project members

Tester Objectives

- To determine that a software product actually meets all formal requirements.

- To determine that a software product does not have unexpected or dangerous behavior

- To provide data to allow release determination

Not Tester Objectives

- To debug
- To deal with poor requirements
  - –Guessing their intent

1. Testers have the right to sample their process inputs, and reject poor quality work (no entry).

- Requirements need to have an agreed standard of less than 1 Major defects per page
- Typical requirements today have 50 to 300 Majors per page
  - (a major defect can lead to wrong test)
- It takes about 15-30 minutes to take a 1 page sample and check it according to standards
  - Like 'unambiguous', 'clear enough to test'.
- If 1 or more defects are found - that work should be rejected as substandard
  - Useful to have agreed Entry levels in advance
  - The argument is that total cost is less, if requirements are clean.

“Rules”:
Best Practice Strong Advice

Three simple rules for inspecting a requirements document:

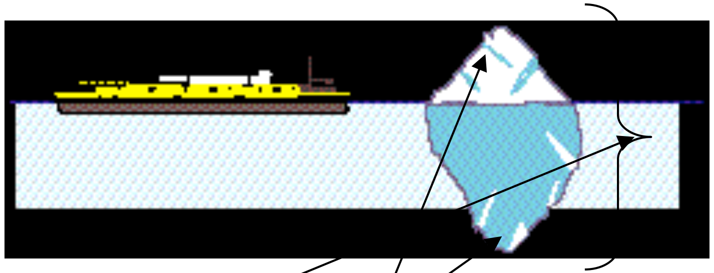
Three Rules for Requirements:

- 1. Unambiguous to intended Readership
  - *Including test planners*
- 2. Clear enough to test.
- 3. No Design specs (= ‘how to- be good’) mixed in
  - Mixed up in the Requirements
  - (Reqts. = ‘how good - to be’) not how to be good!
  - MARK Design as “D”
  - *Except if it is a conscious Design Constraint (which is a requirement type)*

Report for sample page 82

(reported inspection results on requirements document, 4 other managers)

- Total Majors
- Defects, alone
-
- M+m M Design
- ---------------------------
- 41, 24, D=1
- 33, 15, D=5
- 44, 30, D=10
- 24, 3, D=5

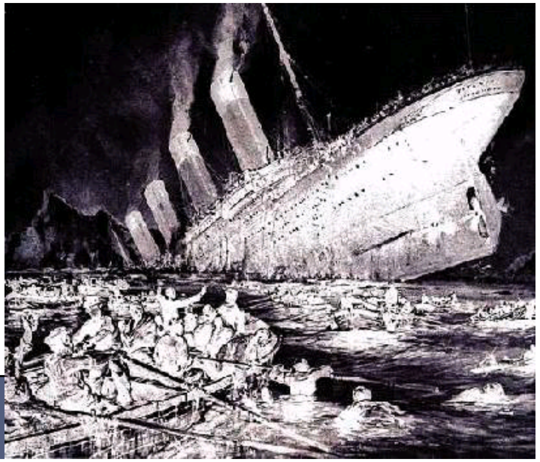


- Team would log unique Majors; about 2x30 = 60
  - total unique majors for the group
- Which is about 30% of total actually in document, so total this page is about 180 Majors (about 120 not identified yet)
- If we attempt to fix the 60 we log, and correctly fix 5/6, then 10 are failed fixes, so:
- The total remaining after inspection and editing = 10+120 =130 Majors per page.

Extrapolation to Total Majors in Whole Document

- Page 81: 120 majors/p
- Page 82: 180 Majors/p
- Average 150 Majors/page x 82 page = 12,300 Majors in the document.

- If a Major has 1/3 chance of causing loss
- And each loss is average 10 hours then total project Rework cost is about 41,000 hours loss.
- (This project was over a year late)
  - 1 year = 2,000 hour 10 people



2. Testers have the right to unambiguous and clear requirements.

- It is the job of requirements writers to write perfectly clear requirements
- It is NOT the job of the testers to ‘guess’ what an unclear requirement intends to say
- If requirements are poor, the tester needs to hand them back immediately for quality controlled rewrite (not guess or ask).
- If the requirement is not good enough for the tester, it is not good enough for any other purpose on the project (like estimating, design)
- If there is a systemic problem with bad requirements, then test management must work towards systemic improvement (standards and Quality Control).

Is this good enough to test for the Usability sub-quality ‘Intelligibility’ for a spy plane?

“High ability for an operator to correctly interpret the meaning of given information.”

How would you test this?

A clear real specification in 'Planguage'

Usability.Intelligibility:

Ambition: High ability for an operator to <correctly> interpret the meaning of given information.

Scale: Percentage Probability of <objectively correct> interpretation(s) of a defined [Set of <Inputs>] by a defined [Individual Person: Default: Trained Operator] within a defined [Time Period].

Meter [Acceptance]: Use about 10 Trained Operators, and use about 100 <representative sets of information per operator within 15 minutes?> <- MAB.

Comment [Meter]: "Not sure if the 15 minutes are realistic" <- MAB.

Comment [Meter]: "This is a client & contract determined detail" <- MAB.

M1: Past: [XXX, 20 Trained Operators, 300 <data sets>, 30 minutes]: 99.0% <- Acceptance Test Report from XXX, MAB.

Record [XXX]: 99.0%. "None other than XXX known by me" <- MAB.

Fail [First Delivery Step]: 99.0%? <- MAB.

Fail [Acceptance]: 99.5%? <- MAB.

Goal [XXX, 20 Trained Operators, 300 <data sets>, 30 minutes] 99.9% <- LN.

========== More User Defined Terms ================

Acceptance: Defined As: Formal Acceptance Test as defined by our contract with Customer XXX.

First Delivery Step: Defined As: By end of November this year (The results of the first evolutionary result cycle will be integrated into the system and will be producing useful results).

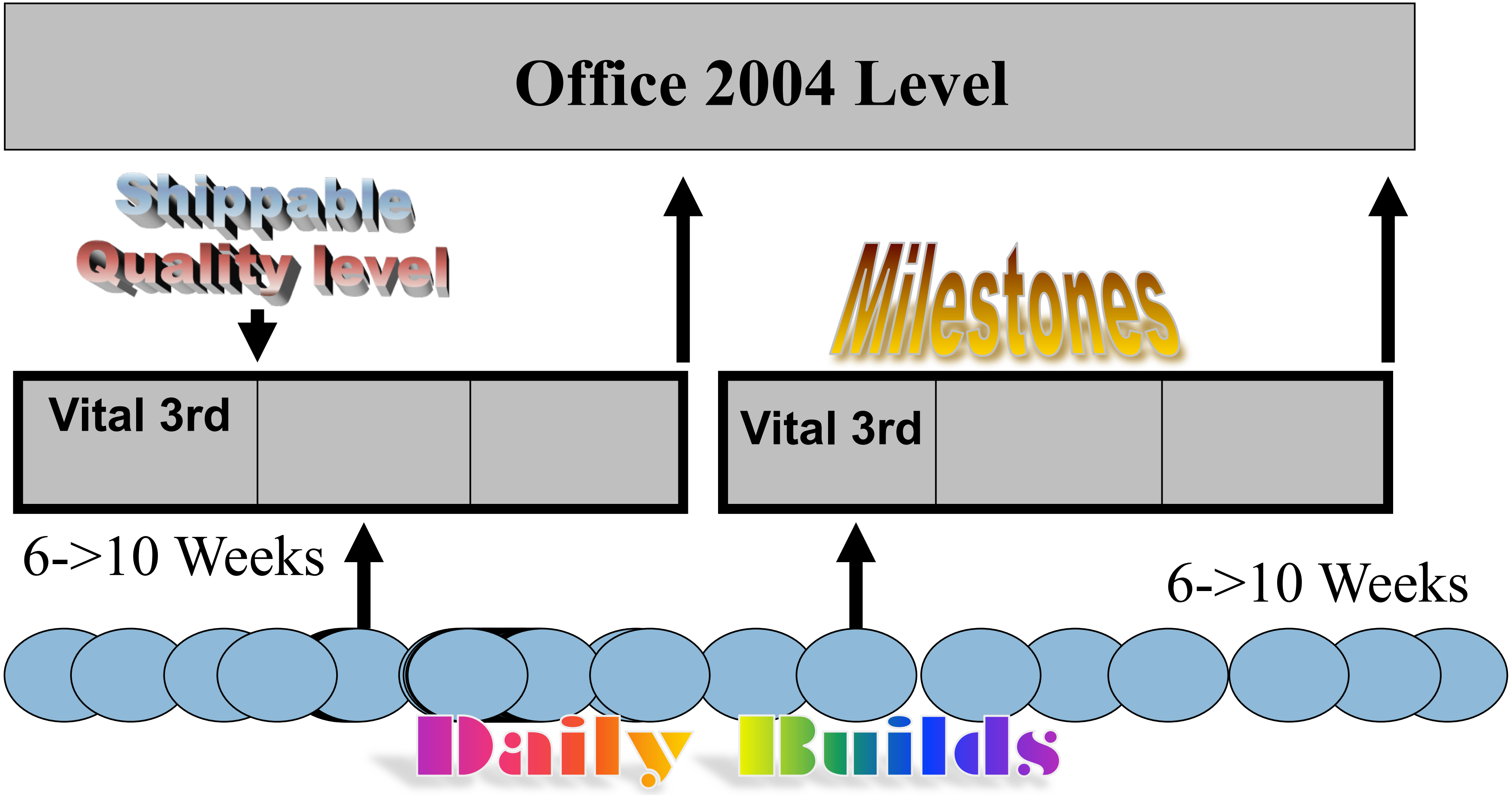
3. Testers have the right to test evolutionarily; early as the system increments.

- Testing should not be dumped on testers at the end of a (late) project
- Testers should be involved in Evolutionary integration testing early and frequently
  - Like every week of the project!
  - Like daily

Milestone Control (Ms)

- “Mike Conte, a senior program manager for Office
- “We actually break our development into three separate milestones.
- They might be six week milestones, [or] they might be ten-week milestones …
- At the end of the milestone our goal is to get all the features for that milestone that have been implemented … for that milestone at zero bugs….
- And then, when we get to the point where we get to ‘ship quality’, we can move on to the next milestone.
- The point of this is that we never get so totally out of control that we’re at the end of a project and we have so many thousands of bugs that we can’t ever tell when we’re going to finish it.”
- Source:Microsoft Secrets   page 200

Multiple Levels of Microsoft Evo



Office 2004 Level

Shippable Quality level

Milestones

Vital 3rd

Vital 3rd

6->10 Weeks

6->10 Weeks

Daily Builds

2003 Report from Microsoft (Craig Larman 28 Feb 2003)

- “However, MS does do more-or-less pure incremental dev, I discovered.
- A typical lifecycle is a big up-front req + des step, followed by 3-4 iterations of dev, each of 3 or 4 months in length.
- It is not pure incremental dev, as the later (long) iterations are not fully specified,
- but it is close to pure incremental dev.”

ARE DAILY BUILDS FOR WIMPS?

- "Evo does relatively continuous integration, but as a communicator you should know that "continuous integration" has a relatively precise technical meaning in the software dev community these days:
- it is now understood as defined in XP (http://www.martinfowler.com/articles/continuousIntegration.html)
- and then fully automated by a couple of guys at ThoughtWorks in the open source tool called CruiseControl http://cruisecontrol.sourceforge.net/

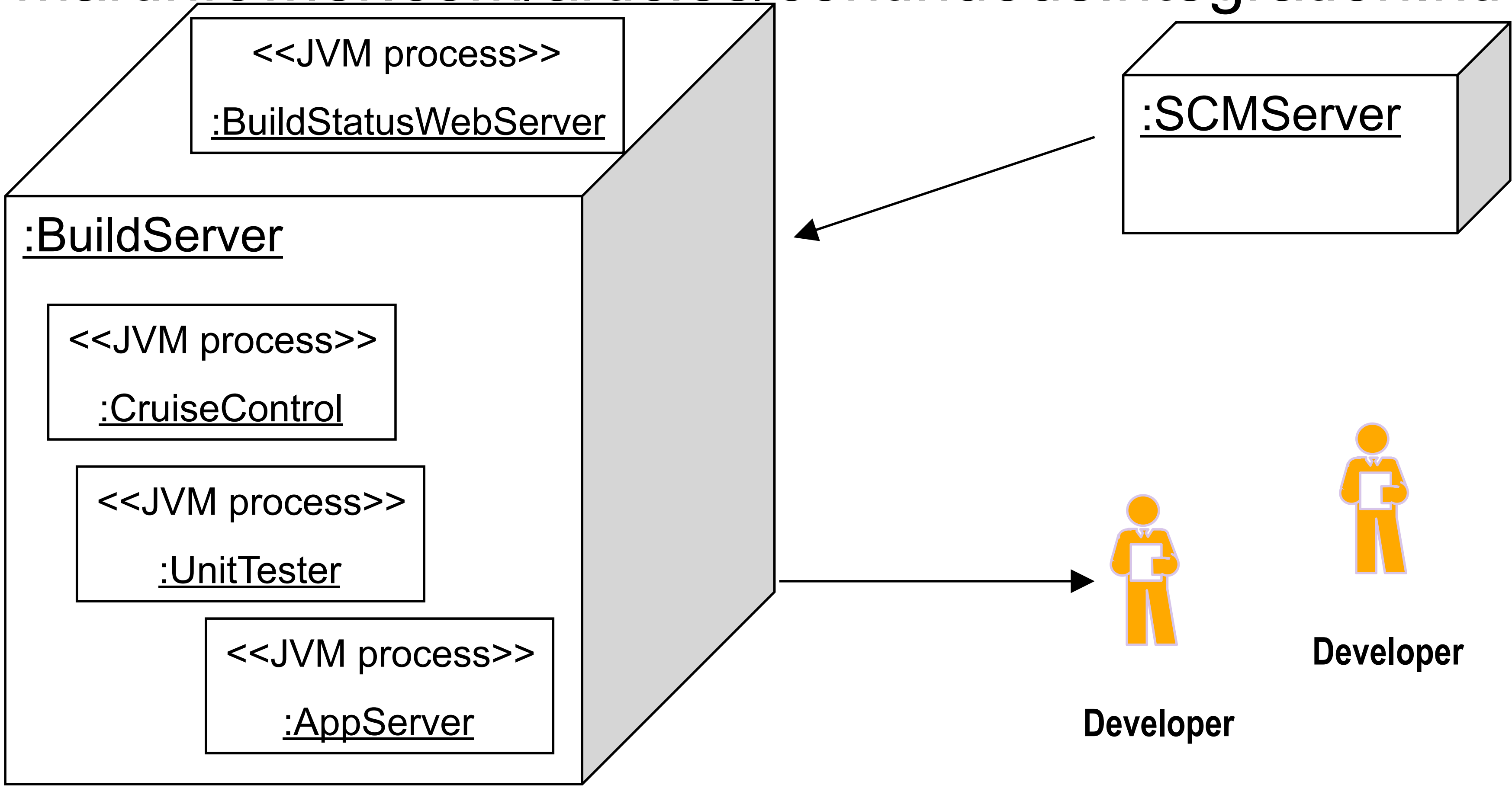
- Now in software dev, it means to have a separate build machine running 24/7 with a daemon process (CruiseControl) that wakes every N minutes (usually around 15 min).
- This kicks off an Ant task that queries the source control system, and if any new check-ins, then it pulls across the entire configuration to the build machine, then
  - 1) compiles it all, 2) runs all automated unit tests, 3) creates the composite app server files, 4) bounces the app server, 5) loads the app into the server, 6) runs all automated acceptance and integration tests.
  - Note the elapsed time between total system integration is usually in the 5-15 min range, making it much more "continuous."

- So, now the software dev community has a saying: "Daily builds are for wimps." ;) meaning daily is not nearly frequent enough.

- If at any point, there is failure, CruiseControl automatically sends email to the new code contributors informing them of failure. Then it updates the continuous integration web page with the build time, results, and contributor names."
- CRAIGLARMAN.COM 14 MARCH 2003

Continuous Integration with Cruise Control or Anthill

- CruiseControl or Anthill: Free OSS CI tools
- martinfowler.com/articles/continuousIntegration.html



4. Testers have the right to integrate their test specifications into the other technical specifications.

- So that the right hand knows what the left one is doing - in spite of the change process
- Specifically, we need to be able to put test info in requirements and design specifications
  - At least as a detailed cross reference
  - For each individual requirement and design
- Why?
  - To sense absence of test planning for some specs
  - To sense need of change in test plans when reviewing changes in specs
  - To give testers constant access to all upstream detail and change
  - So the right hand knows what the left one is doing

Integration of Test Planning: A simple quality requirement *without* integration of test information

Availability:

Version: December 24th 2004.

Scale: % uptime.

Goal [Initial Delivery, Pilots] 99%

Goal [Major Customers,Contracted] 99.9%

Integration of Test Planning specs

(can you see the problem now?)

Availability:

Version: December 24th 2004.

Scale: % uptime.

Meter: built-in software log computes and reports availability.

Test Plan: Test Plan [Availability, June 7 2004].

Test Cases: Test Scripts [Availability, Oct.8 2004].

Goal [Initial Delivery, Pilots] 99%.

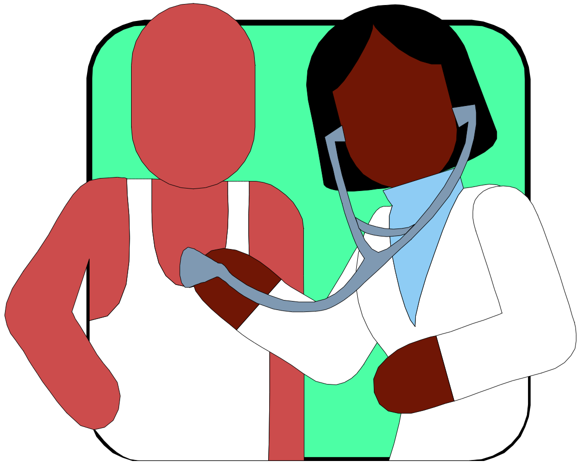
Goal [Major Customers,Contracted] 99.9%.

5. Testers have the right to be a party to setting the quality levels they will test to.

- The depth of testing determines the resources needed to do the job.
- Test cannot have a level of testing imposed on it when it is not given adequate people, time and machine resources to do the job properly
- Testers are the experts on these costs and on the levels of testing we can get for our resources
- So - testers must be a party to the decisions about resources needed for certain levels of quality of testing - under defined conditions
  - Like high quality requirements as input.
- Test also has to shout loudly when
  - planned resources are not available
  - Defined conditions are not true (bad requirements etc.)

Setting the acceptable levels

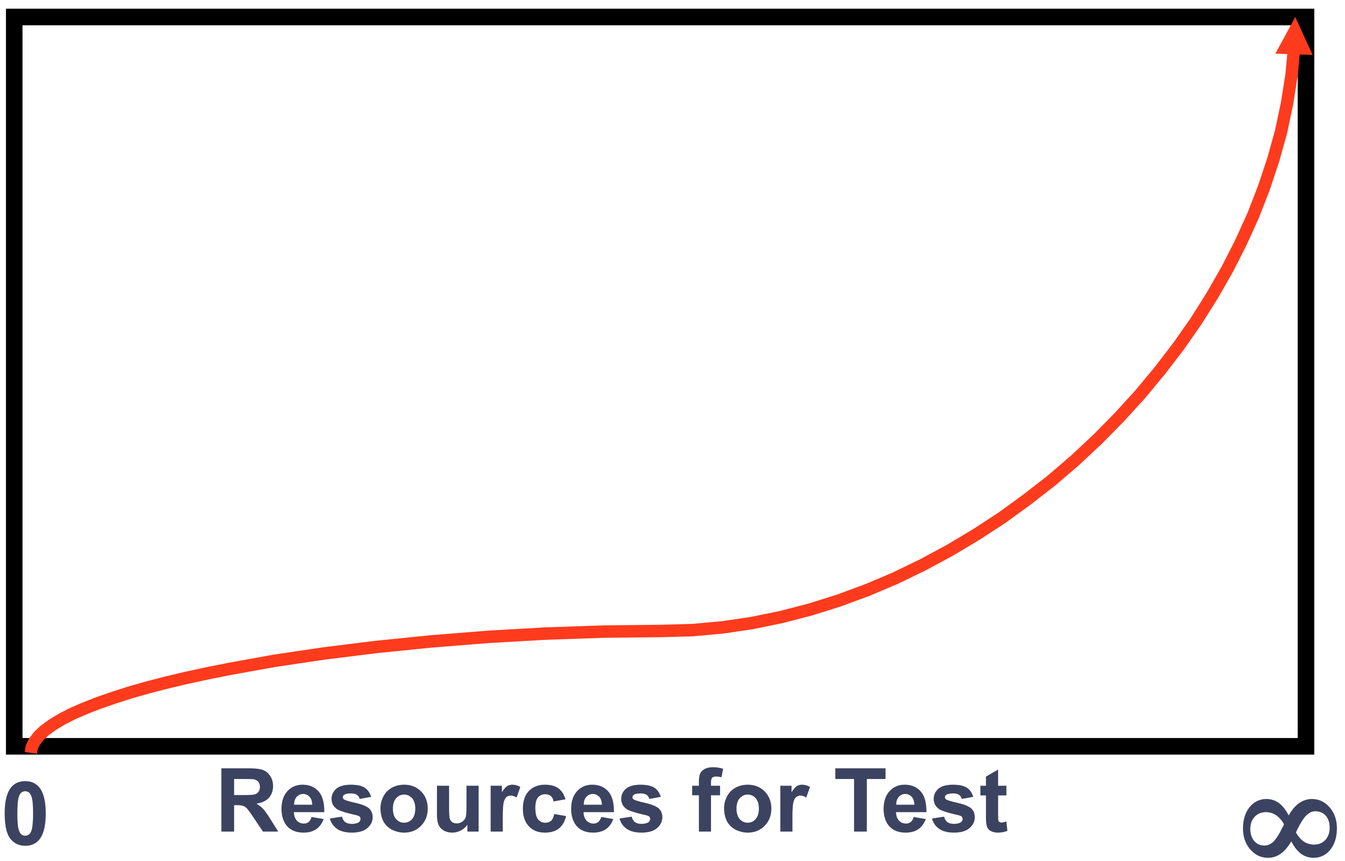
- Entry Condition to Test Planning
  - –Requirements will not exceed 1 major defect per page (300 NC Words) when sampled.



6. Testers have the right to adequate resources to do their job professionally.

- It is the responsibility of the test group to know what resources are in fact needed to do their work to an agreed standard.
  - Give project management a menu - with prices
- It their right to point out the limitations and risks that apply if they are forced to live with less than adequate resources
  - And make sure the resource providers, not test, take responsibility for the problems that might occur.

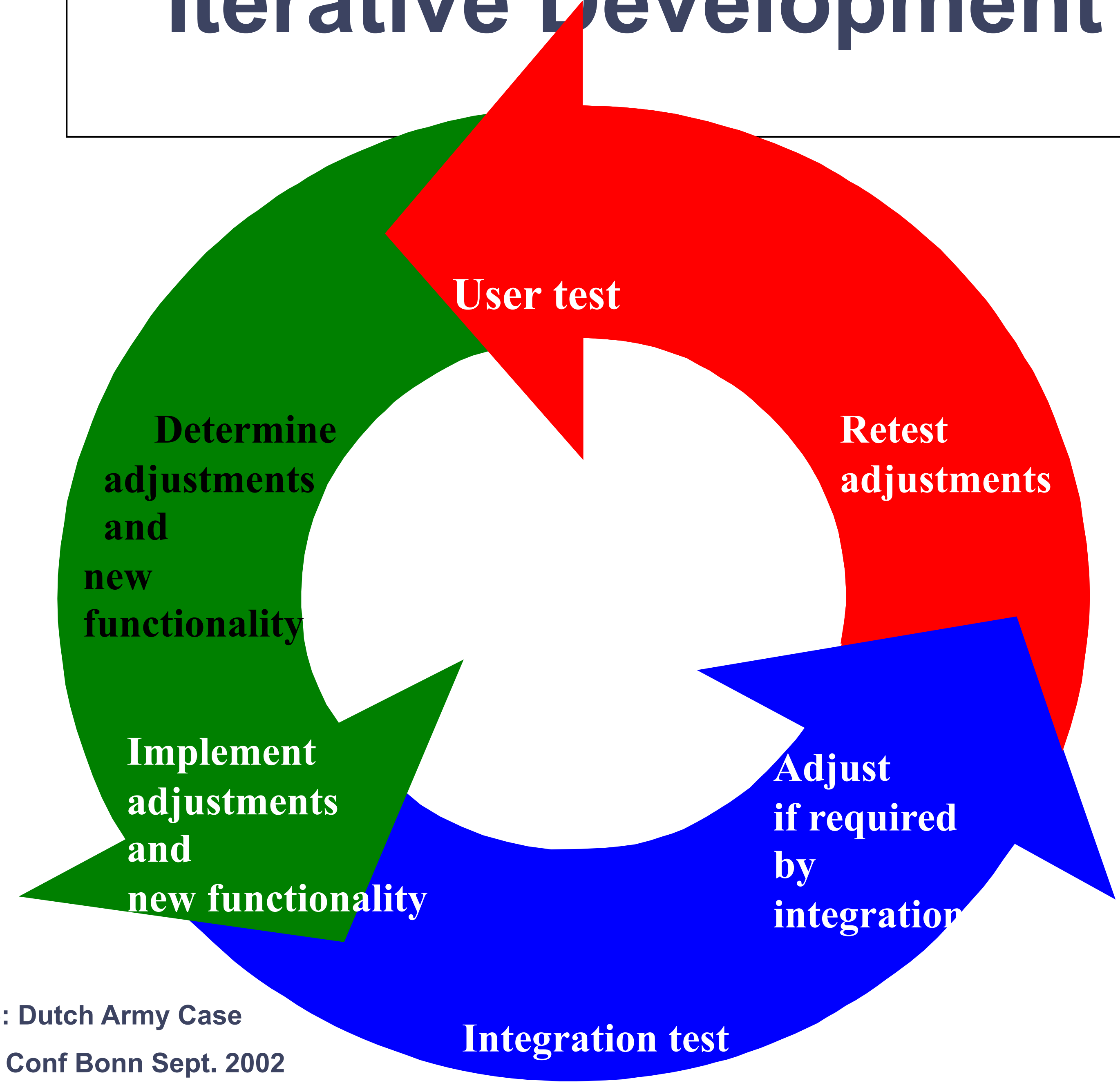
Don’t compromise on resources to do the job right
Make resource suppliers aware that they can only get what they are willing to pay for.



7. Testers have the right to an even workload, and to have a life.

- It is unnecessary and unfair to get the delayed work of other developers, before a fixed deadline - and then get unreasonable pressure to finish on time
- The solution is organizational. The development and test process must be EVOLUTIONARY
  - Test and iterative integration is an early and frequent process.
  - Probably daily, weekly increments throughout the project

Iterative Development



User test

Retest adjustments

Adjust if required by integration

Integration test

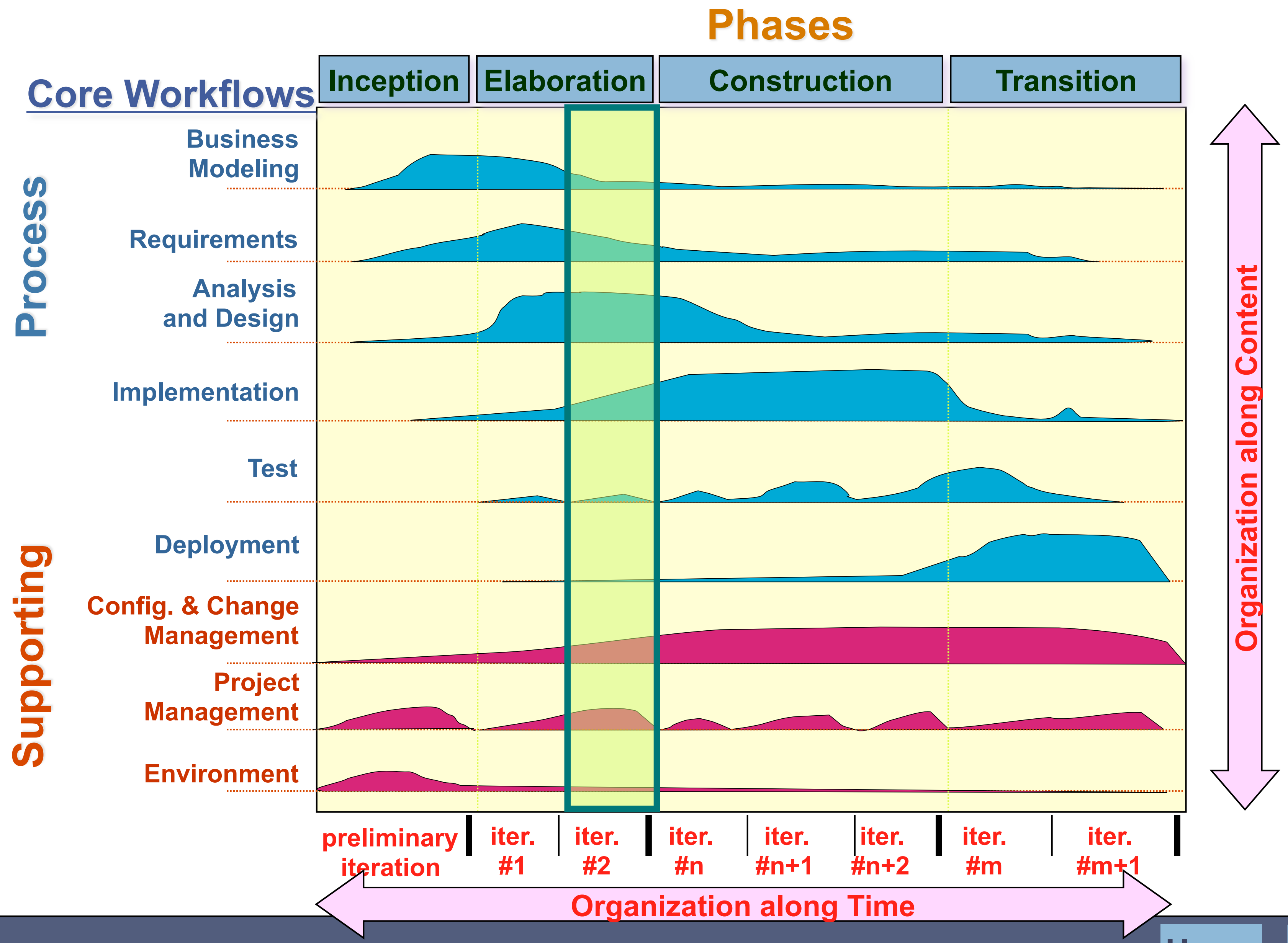
Implement adjustments and new functionality

Determine adjustments and new functionality

Source: Dutch Army Case

NATO Evo Conf Bonn Sept. 2002

The Unified Software Development Process/ RUP SE



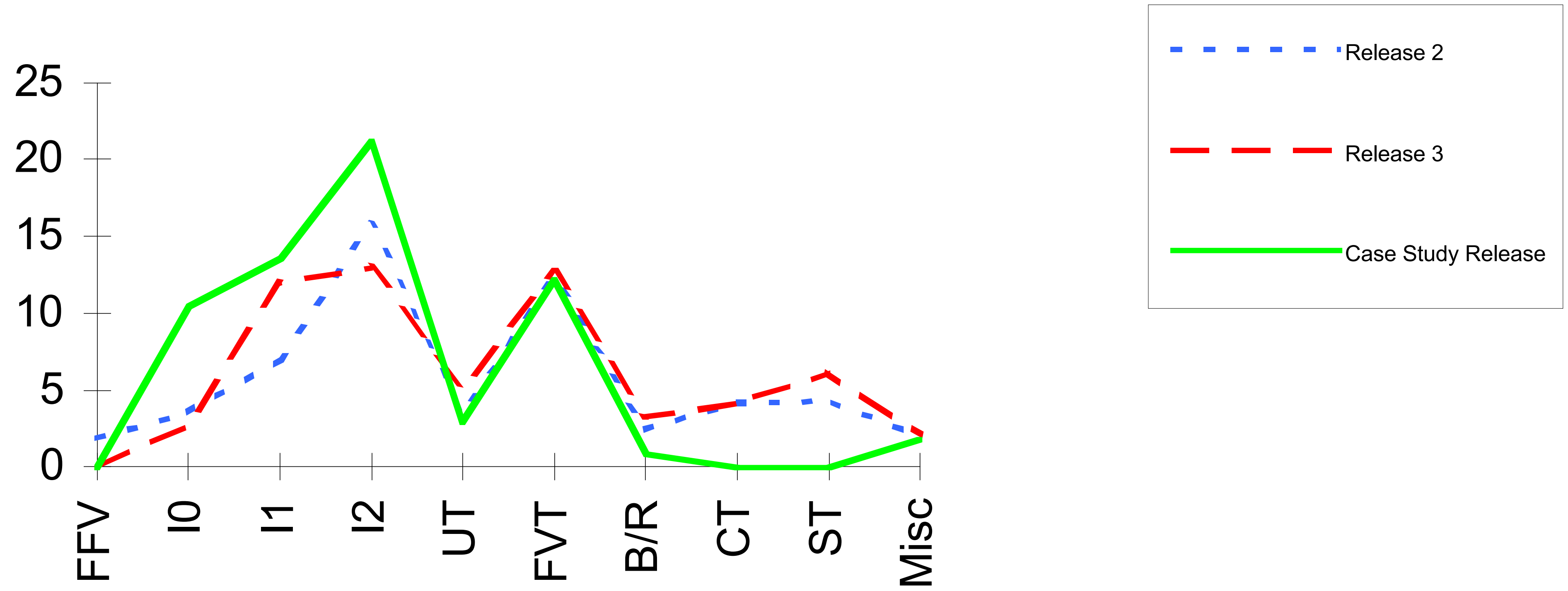
Source: P. Kruchten, ***The Rational Unified Process***, *Addison-Wesley, Reading, MA, 1998.*

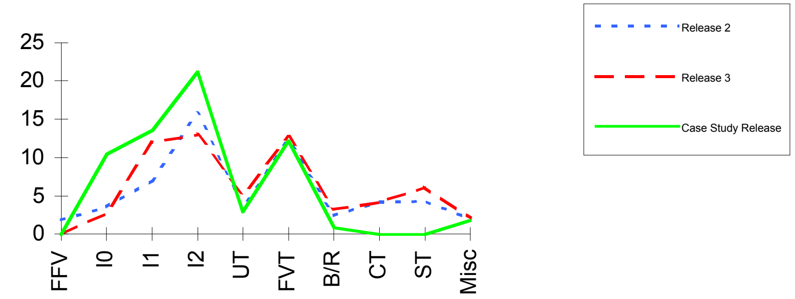
8. Testers have the right to specify the consequences of products that they have not been allowed to test properly.

- Testers in fact have a professional obligation to send clear early written warning signals to project management about consequences.
  - For example:
    - bug rate predictions at customer sites, if released now
    - Load of work at help desks
    - Load of work doing bug fixing and regression testing

All consequent problems fleshed out - with concrete suggestion about what to do in practice about it.

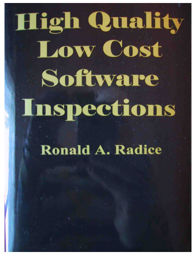
Projections Based on Experience Models: Radice.





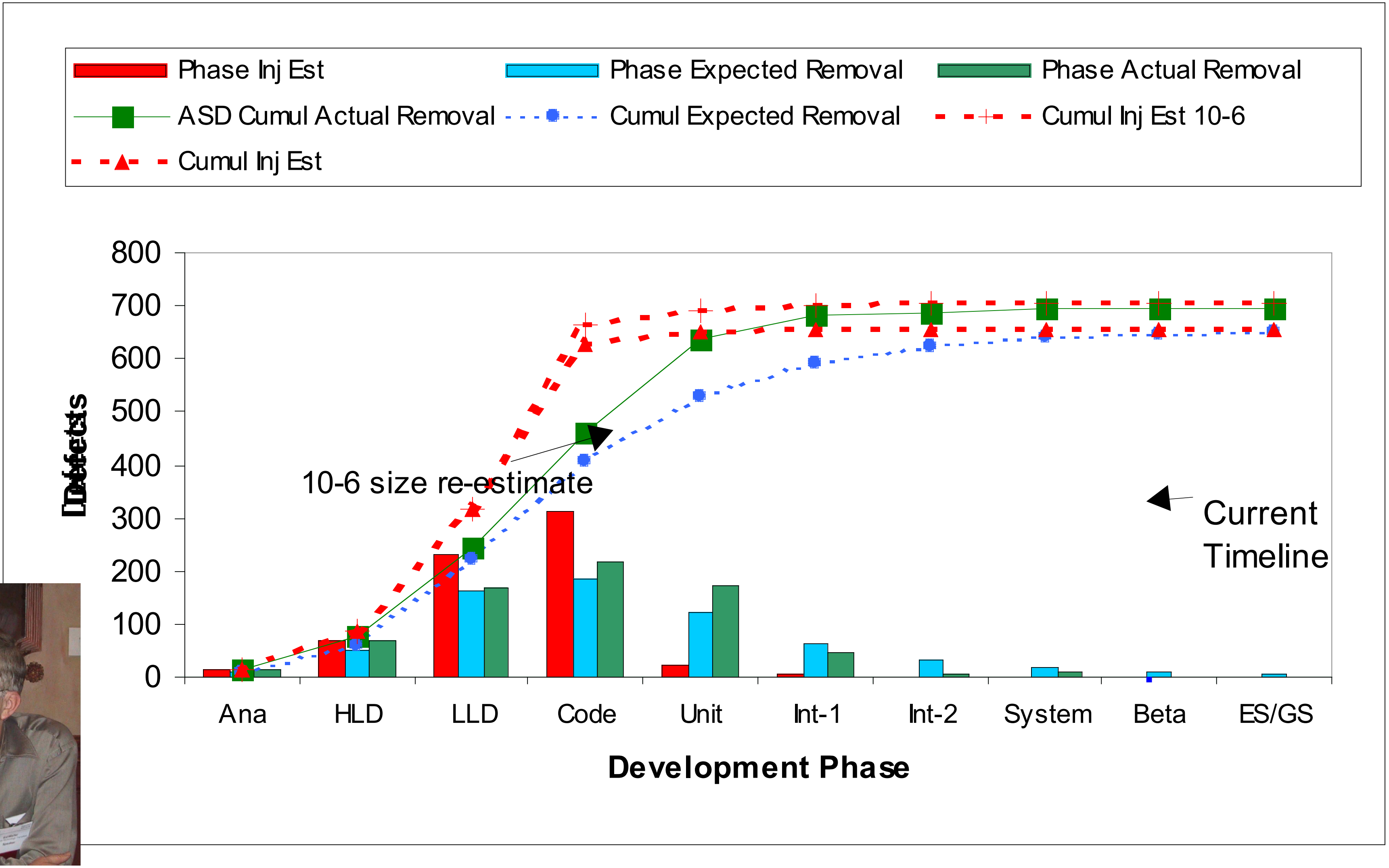
Real Example Projection

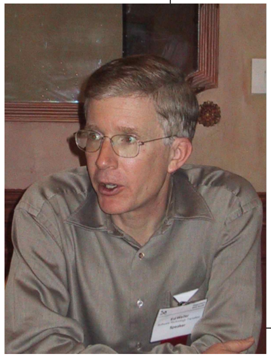
(Ron Radice: IBM)



- “Since our objective was to get the overrun in defects removed as soon as possible, we began to build a case which would use a subset of the recommended 25.9 KLOC as a sample to determine the remaining error density.
- **Based on where the product was at this point in the test cycle, we had projected a remaining defect volume of 24/KLOC.**
- This is to say that there was a latency of this amount in the product.
- We now needed to prove that this projection was credible.
- We argued that if the sample re-inspection would find ten errors per KLOC against the projected 24, the effectiveness would be 41% or about what it apparently was for the primary Inspections in this release.
- If we would find 12, it would be 50% effective.
- If it would find 16, it would be 67% effective, which was the best this product had historically been able to show.
- We, of course, had no way to pre-determine the effectiveness of these re-inspections.
- An effectiveness higher than 41% would be suspect, however, without reasons to demonstrate it should be higher. “

Bull AZ: Defect Removal Experience





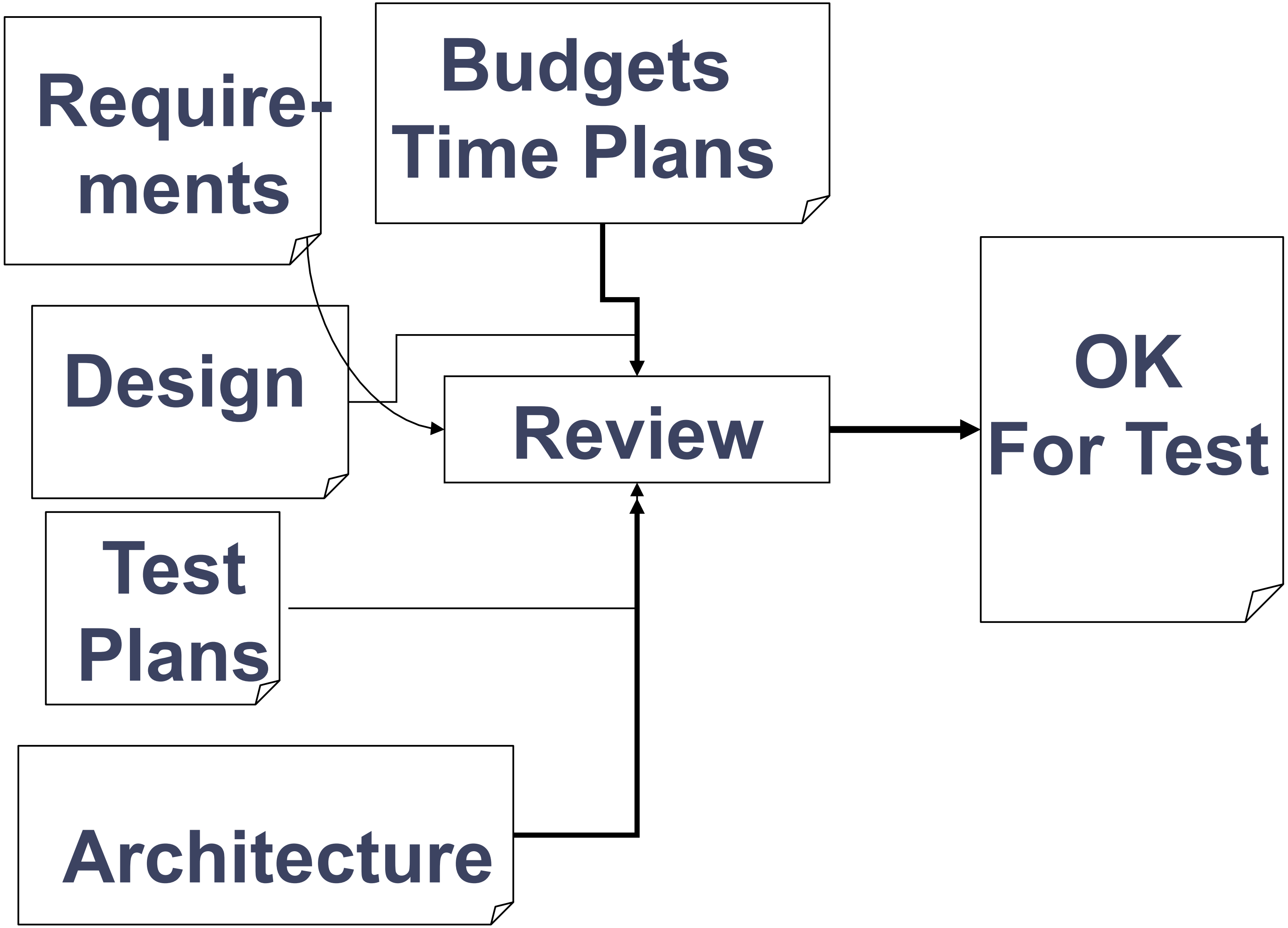
- Source: Ed Weller: **Practical Applications of Statistical Process Control**

9. Testers have the right to *content* review any specifications that might impact their work.

- Testers need to be in on requirements and design reviews
  - Including changes to these
- Testers need to be in on project budget and estimation plan reviews
- Testers need to be in on the Evolutionary delivery loop so that they constantly see the stakeholder experience with what they have allowed to be released at each Evo cycle.

Testers 'Content Review'

Combined with other reviews or against Test's own standards



Example of Design/Architecture Content Review Standard

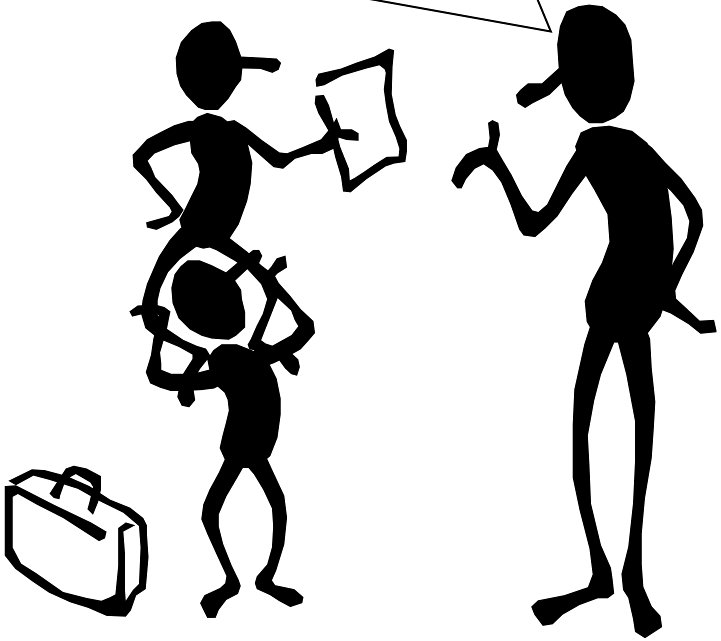
- Design and Architecture will:
  - –Contain interfaces to facilitate automated testing
  - –Contain provision to capture system state and transactions in case of failure
  - –Contain annotation regarding any level of testing requirements for the design that is planned, contracted, or envisaged.
  - –Contain annotation regarding previously known testing degree of any component planned, in any internal or external situation.

10. Testers have the right to focus on testing of agreed quality products, and to send poor work back to the source.

- Any work, even discovering sub-standard quality inputs to test must be charged back as a cost and delay, due to the developers who are not meeting defined exit conditions for their work
- Management needs to see who is really causing the delays!

Quality Control before Test Work

- Test can’t use your requirements, they have more than 10 Major defects/page. The limit is 1.
- So you are holding up testing, until you do your work to the agreed standard!



Last Slide

- If I have forgotten some testers rights
  - –Remind me!
  - –Tom@Gilb.com