Software Test Costs and Return on Investment (ROI) Issues

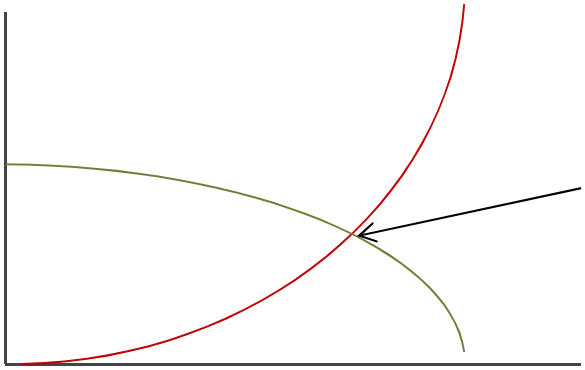
Bob Hunt, Galorath

Tony Abolfotouh, John Carpenter; Robbins Gioia

March 2014

Introduction

- The recent Affordable Health Care Web Site issues have once again raised the issue of adequate testing (not the first such problem, just the most visible "recent" problem)
- There is no such thing as "bug-free" software
- "As a matter of cosmic history. It has always been easier to destroy than to create." Spock from Star Trek II: The Wrath of Khan
- Software Testing is a Risk Based exercise



- How to Drive your Test Cost to "0" – Don't test
- How to drive your user base to "0" – Don't test

Types of Software Testing

- Black Box, White Box, Static, Dynamic
- Over 40 specific others tests types can be identified; not all are used all the time
- Independent Validation and Verification (IV&V) is a Special Case of independent testing that I am considering as different from internal Validation and Verification (V&V) (see next slides)
- Some "new" Test Types: Poka Yoke (error elimination), Kaizen(continuous improvement)
- Test plans seem to often be "ad Hoc"
- Test cost not usually directly estimated

Pesticide Paradox of Testing

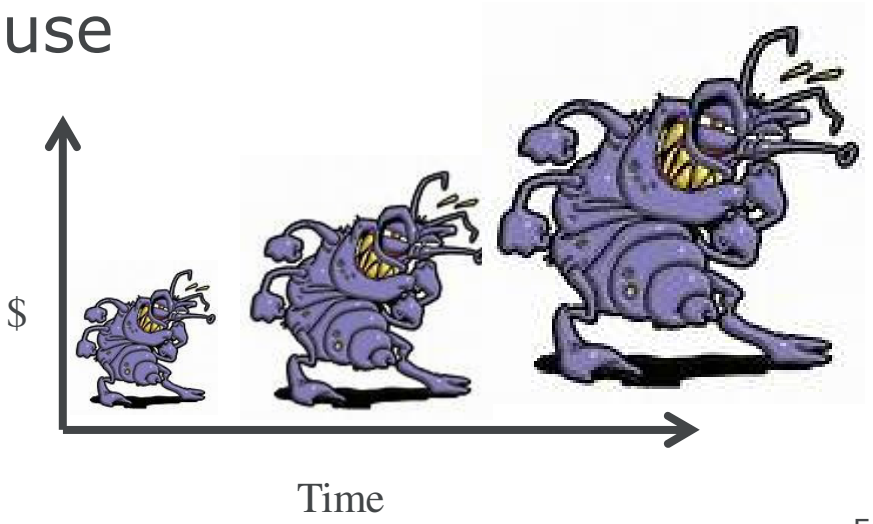
- In 1990, Boris Beizer, in his book *Software Testing Techniques*, Second Edition, coined the term "*pesticide paradox*" to describe the phenomenon that the more you test software, the more immune it becomes to your tests.

- In the spiral model of software development, the test process repeats each time around the loop. With each iteration, the software testers receive the software for testing and run their tests. Eventually, after several passes, all the bugs that those tests would find are exposed. Continuing to run them won't reveal anything new.

- Continually change and adapt testing processes

Cost to Fix Bugs Grows With Time ?

- Generally accepted Axiom: While costs vary by project and environment, the costs to fix defects follow what has come to be known as the "1:10:100" rule. (Achieving Software Quality Using Defect Filters; Randall Rice)
- A defect that costs $1 to fix in requirements or design costs $10 to fix in a traditional test phase and $100 to fix after the product goes into production (live) use
- Is this true?
- Not use in this brief



IV&V Overview

What is Independent Verification and Validation (IV&V)?

- Verification answers the question, "Are we building the product right?" Verification is the process of determining whether or not the software products of a given phase of the SDLC fulfill the established requirements for that phase.
- Validation answers the question, "Are we building the right product?" Validation evaluates the software products throughout the SDLC to ensure those products meet the mission and customer's needs.

Verification often a by-product of bugs i.e. Microsoft

Validation a by-product of mission failure i.e. AF C130J

What is Independence?

- **IEEE** defines independence in IV&V as three parameters:
- **Technical independence** is achieved by IV&V practitioners who use their expertise to assess development processes and products independent of the developer.
- **Managerial independence** requires responsibility for the IV&V effort to be vested in an organization separate from the organization responsible for performing the system implementation.
  - The IV&V effort independently selects the segments of the software and system to analyze and test, chooses the IV&V techniques, defines the schedule of IV&V activities, and selects the specific technical issues and problems to act upon.
  - Most projects view V&V as sufficient and do not recognize the added value the independence brings.
- **Financial:** Typically funded from Corporate General & Administrative (Expense). Projects may directly fund services.

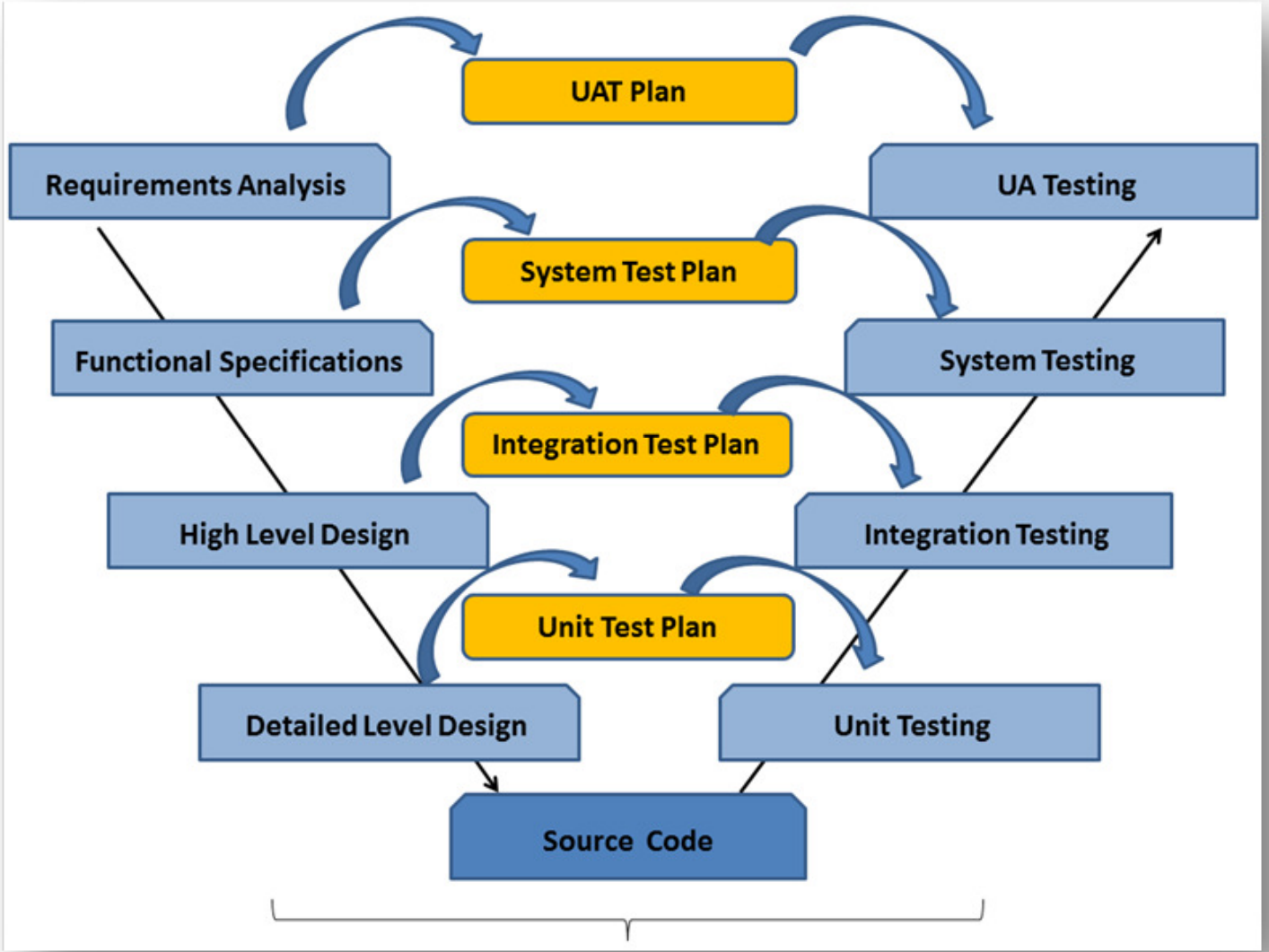
What Should You Pay for IV&V

- You would expect IV&V cost to at least equal internal defect discovery costs – unless less testing is required
- An increase factor of up to 2.0 is not unrealistic for "complete" IV&V versus Internal Software Project Testing (source Bob Hunt)
- IV&V is, conceptually, pretty simple:
  - Identify the products and processes to undergo verification and validation (preferably before or in the early stages of development),
  - Determine the criteria with which each of those products and processes can be evaluated (starting with standards where available e.g. the IEEE-1012-2004 standard for software verification and validation),
  - Assess the products/processes while in production and upon completion to verify they meet the predefined criteria and note where they don't,
  - Re-assess products/processes when deficiencies have been addressed.
- IV&V Vendors are very protective of any internal methodology they use to estimate IV&V costs - Then they guess

Some Questions

- What does the Test Plan show?
- Is there a test plan?
- If the project is using Agile Development or Agile Acquisition, how many staff on the Scrum Team are identified as testers?
- Is there a specific “Sprint” dedicated to testing
- Does the software meet the requirements that guided its design and development and work as expected?

There Is A Process



Some Testing Data

- Hailpern and Santhanam: "... debugging, testing, and verification activities can easily range from 50 to 75 percent of the total development cost." (Software debugging, testing and verification by Hailpern and Santhanam, 2002, seehttp://www.research.ibm.com/ journal/sj/411/hailpern.pdf)
- Data from The Case for Automated Software Testing (Bernie Gauf and Elfriede Dustin, IDT; Software Tech News)

Table 1. Survey result for question: "Time currently spent on testing in relationship to overall software development lifecycle"

Question Response Options	*% of Respondents*
Less than 30 %	28%
30 % - 50%	**48.6%**
50% - 75 %	**14%**
75% - 100%	7.00%
Other (please specify)	3.75%

- Most Automated models estimate total development cost and then distribute a percentage (20 to 50%) of the total cost as testing by SDLC phase and type of program

Some Defect Data

- Assume 7 to 10 delivered defects per KSLOC
- Code Complete by Steve McConnell:
- (a) Industry Average: "about 15 - 50 errors per 1000 lines of delivered code.
- (b) Microsoft Applications: "about 10 - 20 defects per 1000 lines of code during in-house testing, and 0.5 defect per **KLOC** (**KLOC** IS CALLED AS 1000 lines of code) in released product (Moore 1992).

  (c) "Harlan Mills pioneered 'cleanroom development', a technique that has been able to achieve rates as low as 3 defects per 1000 lines of code during in-house testing and 0.1 defect per 1000 lines of code in released product (Cobb and Mills 1990). A few projects - for example, the space-shuttle software - have achieved a level of 0 defects in 500,000 lines of code using a system of format development methods, peer reviews, and statistical testing."
- Capers Jones assumes 5 to 7 defects per function point

What testing can do

- "Today a typical application of 1,000 Function Points will contain 5,000 defects and deliver about 750 defects to customers using the normal waterfall approach. It is theoretically possible therefore to cut the defect potential down to 2,500 and deliver only 25 defects to customers using state-of-the-art defect prevention and removal methods." The Economics of Software Quality; Jones and Bonsignour, page 189

- **This implies a potential defect delivery reduction of 97%!**
- So, how much testing should we do and what should it cost

A **Proposed** Matrix To Estimate How Much Testing Should be Done

System Criticality	A	SDLC Phase*	B	Test Requirement Score = A*B	Score Impact	Recommended Test Level
Low	1	Requirements	1		1 to 10	Min # TCa**
	2					
	3		2			
	4					
Normative	5	Architecture	3		11 to 19	Average # TCa
	6					
	7		4			
	8					
	9			21	20 to 29	Max # TCa
Critical/loss of life	10	Construction	5		30	Max # TCa plus IV&V

* Assume anything after fieldinng would be in Mainatanance

** TCa = Test Cases

- This is the first cut at an evaluation tool, need data to refine
- Apply it to your system and see if it makes sense

Test Cost Equation
(Function Point Based)

- Test Cost = (#TCa)*(hr/TCa)*($/hr)
- $/hr = your program estimate
- Hr/TCa = 2.25 hr (broad based average from Economics of Software Quality; Jones and Bobsignour, page 340, Table 5.9)
- #test cases = FP**(1.05 for Min, 1.2 for Average, 1.3 for Max) for programs with FP counts between 100 and 100,000 hr (Economics of Software Quality; Jones and Bobsignour, page 335) – other data sets show some variation, and the average number appears to be higher than many current Government programs are executing)
- Average Test Cost = 2.25(#FP**1.2)(hourly rate)
- Other possible measures could be #LOC; #artifacts; #files, functions, classes, tables; …

Why Function Points as the Metric

Defects are thought to come from

- Coding (17%)
- Requirements
- Architecture
- Hardware
- Design
- Other Non-coding issues

LOC measures may not address non-coding defects

Can I Backfire TO get to SLOC

Backfiring Function Points to Source Lines of Code is not recommended due to great variations in language and style, but...

Partial List from Economics of Software Quality p. 67

Language	SLOC per FP
Basic Assembly	320
C	160
COBOL	107
Ada95	53
Oracle	40
Pearl	36
C++	32
AVERAGE	**58**

Cost Implications

This chart applies the cost per test case data to the Test Case Criteria matrix presented earlier

System Criticality	A	SDLC Phase*	B	Test Requirement Score = A*B	Score Impact	Recommended Test Level	Cost Implications
Low	1	Requirements	1		1 to 10	Min # TCa**	2.25(#FP**1.05)(hr rate)
	2						
	3		2				
	4						
Normative	5	Architecture	3		11 to 19	Average # TCa	2.25(#FP**1.2)(hr rate)
	6						
	7		4				
	8						
	9			21	20 to 29	Max # TCa	2.25(#FP**1.3)(hr rate)
Critical/loss of life	10	Construction	5		30	Max # TCa plus IV&V	2.25(hr rate)(#FP**1.3+ #FP**1.2)

* Assume anything after fieldinng would be in Mainatanance

** TCa = Test Cases

Assume IV&V equals average Test Costs

Preliminary Conclusions

- We can develop a "Test Critically Matrix" to determine the level of testing/number of test cases required
- Using broad based averages, we can determine the expected Test Cost required to meet the "Test Critically Matrix"

Now

- What is the Return on Investment?

Return On Investment (ROI)

ROI is a performance measure used to evaluate the efficiency of an investment or to compare the efficiency of a number of different investments. To calculate ROI, the benefit (return) of an investment is divided by the cost of the investment; the result is expressed as a percentage or a ratio.

A return on investment formula can be expressed as:
ROI = (Gain from Investment – Cost of Investment)/Cost of Investment

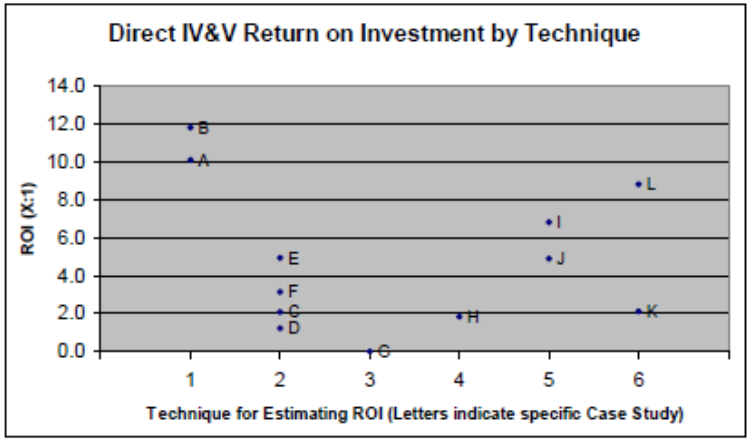
For Software/IT Systems the key questions become:

1. What was the gain – Tangible and Non-tangible or fiscal and non-fiscal? – TBD. Program unique. …. No generic metric developed YET
2. What was the cost of the investment? Take this from the Cost Matrix

You might say I am halfway there now

NASA IV&V Costs and ROI Data

- Data from March 2008 KPMG WebEx
  - QA/IV&V can typically cost 5 to 10 percent of the **total cost of an IT project**, depending on the complexity of the project and the specific scope of QA/IV&V activities*
  - Several studies indicate that the ROI on QA/IV&V investments can **



- *NASA PAE 2008 Study show ROI values from 1.5 to 12; but says,* "There is a wide range of opinions and studies regarding the cost effectiveness of IV&V. The study team was **unable to identify a common methodology for calculating ROI,** and individual Case Studies using the same methodology resulted in a range of values.

**Estimating Direct Return on Investment of Independent Verification and Validation using COCOMO-II, J.B. Dabney, G. Barber, and D. Ohi, 2006*
***A Case Study of IV&V Return on Investment (ROI), R.A. Rogers, D. McCaugherty, F. Martin, NASA, October 2000*

Alternative IV&V Cost Equation

- Assumptions:
  - A Software Development program has a Total Development Cost of $**X** =($x/SM)(SM)* (simplistic express of software equations)
  - Assume the Cost of Internal Software Testing (IST) is 30% of the total Software Development cost, or IST = (0.3)($**X**)= (0.3)($x/SM)(Zsize)
  - IV&V cost equals IST or IV&V=(1)(0.3)($x/SM)(Zsize)
  - There are Y delivered defects per size metric
  - The cost per size metric to rewrite code is the same as the cost per size metric to generate original code
  - The Cost of Defect Recovery and Removal follows the "1:10:100" rule and all "**Y**" defects would have been found after fielding
- **The simple cost Break Even Point is reached when:**
  - IV&V cost = IV&V Savings
  - (1)(0.3)($x/SM)(Zsize)= (100)(**Y**)($x/SM)
  - (0.3) Zsize = 100**Y**
  - (0.3)Zsize/100 = **Y**

*SM – size metric could be expressed as Lines of Code, Function Points, …

IS Testing Worth the Cost

- It depends on who is paying for it , but the answer is yes
- When critical safety and life issues are at stake (and accidents are averted), the answer is again yes
- But what does it really cost –
  - Although numbers vary by project and environment, the costs to fix defects average what has come to be known as the "1:10:100" rule. (Achieving Software Quality Using Defect Filters; Randall Rice)
  - A defect that costs $1 to fix in requirements or design costs $10 to fix in a traditional test phase and $100 to fix after the product goes into production (live) use

  Most ROI studies assume any defects found in Testing would have made it to production (live usage)

Closing Thoughts

- No software is delivered defect free
- You can reduce Testing to no cost by not testing (this could reduce your users to zero also)
- Key Benefits:
  - Early detection means saved/avoided costs
  - Improved quality
  - Lower Total Cost of Ownership (TOC) cost
  - Reduced management burden
- Testing is hard work that takes expertise independence, and a degree of rigor -
- Understand you need competent, experienced testing professionals but this is not a mystical art.

- *Bottom Line: Robust Testing is worth the cost, but an effective ROI metric remains allusive*

Questions

Please contact:

- Bob Hunt, Vice President, Galorath Inc
  - Email: bhunt@Galorath.com
  - Phone: 703.201.0651
- *Tony Abolfotouh/John Carpenter - Independent Analysis Team Lead, Robbins-Gioia LLC*
  - Email: TONY.ABOLFOTOUH@CBP.DHS.GOV
  - Phone: 757.262.7462