



**Software Testing:  
a profession of paradoxes?**


Zeger Van Hese  
zeger.van\_hese@ctg.com

Copyright © CTG, Inc.




**« *Life is a riddle...***

***...unfortunately the answer is  
not written on the back of anything »***



**(old Chinese proverb)**

Copyright © CTG, Inc.



## Overview

- Goal
- Definitions
- Catch-22 of testing
- Paradoxes in software testing
  - Testing realities
  - Pesticide paradox
  - Test reporting
  - Test team
  - Test automation
  - Unit testing
  - Usability testing
- Conclusion
- Q&A



## Goal

Awareness

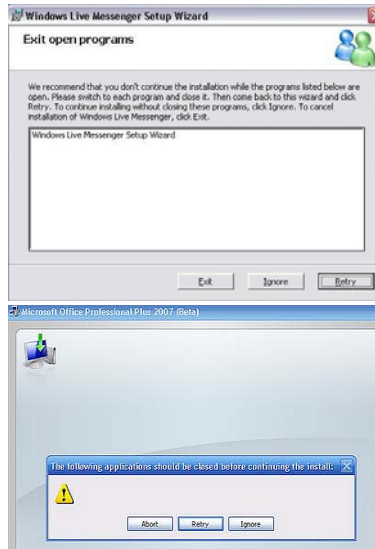
Food for  
thought

Solution ?

## Some definitions

### Catch-22

- Satirical fiction novel Joseph Heller, 1961
- Trap preventing soldiers from leaving the army
- Universal meaning:
  - No-win situation
  - Situation in which you cannot obtain A without B, but B requires A
- Job seeker's catch-22



Copyright © CTG, Inc.

5



## Catch-22 of software testing

*We end testing when we're finished...  
... but we're never finished !*

- Main cause: complexity of software
- Testing is potentially endless
- Testing is “stopped” - not “finished”
- When to stop?
  - Pessimistic
    - Time - budget – test cases exhausted
  - Optimistic
    - Reliability meets requirements
    - Benefit from continuing < testing cost



Copyright © CTG, Inc.

6



## Some definitions - cont

### Catch-22

- Satirical fiction novel  
Joseph Heller, 1961
- Trap preventing soldiers from leaving the army
- Universal meaning:
  - No-win situation
  - Situation in which you cannot obtain A without B, but B requires A
- Job seeker's catch-22



### Paradox

- Apparently true statement that leads to situation which defies intuition
- Purpose: arrest attention & provoke fresh thought
- Zeno's paradox



- Happiness or a ham sandwich?  
(Smullyan, 1978)

Copyright © CTG, Inc.

7



## Some definitions - cont

### Happiness or a ham sandwich?

*Which is better, eternal happiness or a ham sandwich?*

*It would appear that eternal happiness is better, but think again!*

*After all, nothing is better than eternal happiness, and a ham sandwich is certainly better than nothing.*

*Therefore a ham sandwich is better than eternal happiness.*

### Paradox

- Apparently true statement that leads to situation which defies intuition
- Purpose: arrest attention & provoke fresh thought
- Zeno's paradox



- Happiness or a ham sandwich?  
(Smullyan, 1978)

Copyright © CTG, Inc.

8



## Paradoxes in software testing: 1. Testing realities

### Who watches the watchmen?

- Gatekeepers of quality?
- Humans err - testers are human
  - Inattention blindness
  - Classic testing mistakes (M. Heusser)
  - Thinking errors (M. Bolton)

*"I was most surprised the day I realized the paradox of - how am I going to write tests for the tests that I'm writing?"*  
(Sara Ford)

### Testing can't show that bugs don't exist

- No guarantee there are no more bugs to be found

### Testing influences its own outcome

- Observer effect
- Log files - debugging



## Paradoxes in software testing: 1. Testing realities *cont*

### Failure breeds success

- Scientific proof  
(*University of Exeter, A.J. Wills, Journal of Cognitive Neuroscience - Jan 2007*)
- We learn more from failures than from successes:
  - Lateral thinking
  - More experienced
  - Honest
  - Thick-skinned

### The biggest bugs are the hardest to find

*Intermittent bug = a mysterious and undesirable behavior of a system, observed at least once, that we cannot yet manifest on demand (James Bach)*

The more bugs you find...



## Paradoxes in software testing: 2. Pesticide paradox

*Every method you use to prevent or find bugs leaves a residue of subtler bugs against which those methods are ineffectual*  
(Boris Beizer)



- The more you test software, the more immune it becomes to your tests (cfr reaction of insects to pesticides)
- Software builds resistance to (repetitive) tests
- Fewer bugs being found - tests become ineffective
- Major drawback of automated testing
  - But: important in agile projects (regression)
- Solution?
  - Continually design new testcases
  - Model based testing

## Paradoxes in software testing: 3. Simpson's paradox

or a possible test reporting paradox

- Two or more sets of data lead to one conclusion when evaluated individually, but lead to an opposite conclusion when the sets are combined
- Beware of aggregated data!

Figure 1 Comparing System A and System B

Manual Tests	System A	System B
Number of tests passed	50	15
Total number of tests	200	100
% tests passed	25%	15%
Automated Regression Tests	System A	System B
Number of tests passed	85	300
Total number of tests	100	400
% tests passed	85%	75%

Figure 2 Combined Data for Manual and Automated Regression Tests

Combined Data	System A	System B
Number of tests passed	135	315
Total number of tests	300	500
Score	45%	63%

## Paradoxes in software testing:

### 4. Test Team paradox

- Testing = a procedure for critical evaluation  
*(American heritage dictionary)*
- Constantly critical attitude
- Successful teams need to be negative
- Key: critical focus – positive outlook
  - Highlight good points
  - Set clear goals
  - Try to find bugs early
  - Temper your enthusiasm
  - Try to bring bad news in a positive way

Source: William Echlin – the test team paradox



## Paradoxes in software testing:

### 5. Test automation paradoxes

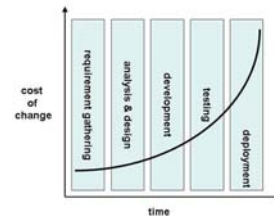
- N° 1: Bugs in automated tests
  - Test automation = software development
  - Who watches the watchmen – revisited
  - Tests for tests for tests for tests for...
  - Test suite failures: false alarms – silent horrors
- N° 2: Automated regression tests
  - Frequently changing sw → frequent retesting
  - Least reliable when we need them the most

Source: Brett Pettichord (2003)



## Paradoxes in software testing: 6. Developer testing paradox

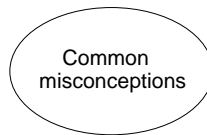
- Unit testing's benefits:
  - Best option to improve quality
  - Opportunity to catch bugs early
- But still... not a general practice



Boehm's cost of change curve

"Too time-consuming"

"It only proves that the code does what the code does"



"I don't need unit tests"

"Integration tests will catch the bugs anyway"

Sources: Alberto Savoia (2005) & IPL – "Why bother to unit test"

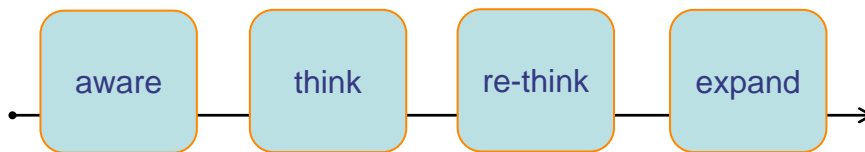
## Paradoxes in software testing: 7. Usability paradox

- Usability testing: important & expensive
- When do you plan?
  - Early?
    - Heading in right direction
    - Changes made when they're least expensive
    - But: no software
  - Later?
    - Software available to user – all valid feedback
    - But: no time/money to correct
- True story ("pretty dire")
- Prototyping

## Conclusion

*“How wonderful that we have met with a paradox.  
Now we have some hope of making progress”*

**Niels Bohr**



## Thank you for your time!

Questions/Comments ?  
[zeger.van\_hese@ctg.com]