CSI Parallels for Bug Investigation

Jon Bach
Manager, Corporate Intellect
jonb@quardev.com

STP Con
October 23, 2009

A Seattle CSI Detective

Abductive Inference

Abductive inference means finding the best explanation for a set of data.

1. Collect data.
2. Find several explanations that account for the data.
3. Find more data that is either *consistent* or *inconsistent* with explanations.
4. Choose the best explanation that accounts for the important data, or keep searching.

Jon Bach / Quardev Laboratories, Seattle, WA 98119

Parallel #1

“Look up, not just straight ahead…”

Parallel: Change your perspective when thinking of software tests to run.

Parallel #2

“Projectiles go through glass and leave different signatures…”

Parallel: The same is true for bugs… programs leave different signatures on how they use memory or install files.

Exercise: Psychic Reading

What do you see?

What's different?

What's the principle?

Key Ideas

How could this be fooling me?

Is anything changing?

Follow-up tests: *what if I try <...> ?*

Is there a logical principle at work?

Parallel #3

"Look in the garbage… we go into toilets quite a bit…"

Parallel: Bugs could reside in places we don't associate with normally having problems.

Parallel #4

"We have to do presumptive tests sometimes (like the bullet through rubber)…"

Parallel: We, too, have to check our basic perceptions to make sure that a bug is really what we think it is.

Exercise: This App Can Break

What do you notice?

What might be happening?

Why is my machine different?

Jon Bach / Quardev Laboratories, Seattle, WA 98119

Key Ideas

Is it isolated to my machine?

Is it too technical for me?

What tools might help?

Why have I never seen this before?

Parallel #5

"Detectives should cut their own path to an outside crime scene…"

Parallel: There is more than one way to reproduce or find a bug.

Parallel #6

"We take 'elimination fingerprints' to rule out different suspects… fingerprints can last a long time…might have been there months ago…"

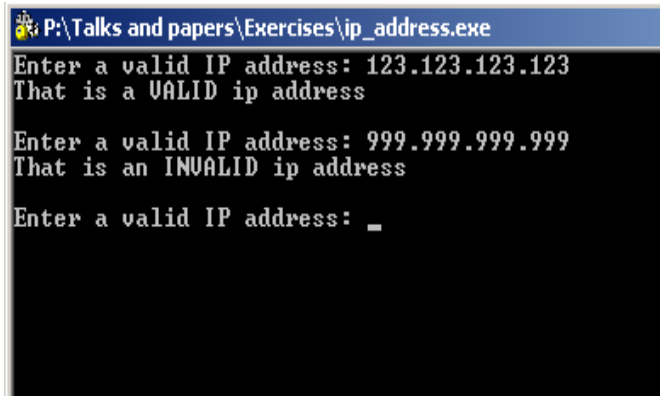
Parallel: We do follow-up tests or peripheral tests to rule out other causes. Beware of the Primacy Bias… a bug might have started weeks ago and only shows itself now.

Parallel #7

"Footwear impressions and fingerprints are there whether we see them or not…"

Parallel: The same is true for software defects… we don't break software, we find the breaks that are already there.

Exercise: IP Syntax Checker



Does it work?

What is the hidden feature?

What story does the data tell?

Key Idea

Think of different dimensions

Collaborate

MFAT vs. OFAT

Break the “rules”

Parallel #8

Tools: reflective UV imaging screen, forensic stepping plates, sifting screens; Detectives use photogrammetry – series of digital photographs in succession

Parallel: We have special tools as well (inControl, log file tracing, Spector, WME, dxdiag, TaskManager).

Parallel #9

"Sometimes you have to match the bullet even though the crime is solved…"

Parallel: Even though you have found the failure for the Programmer to repro, the root cause may be undetermined.

Parallel #10

"Two heads are better than 1…"

Parallel: Paired Testing; "fresh eyes find bugs".

Parallel #11

"Crime scenes might have CS gas residue…"

Parallel: We may be digging in an area that complicates our ability to find bugs.

Parallel #12

"We study different disciplines: entomology, odontology, etc…"

Parallel: We also study different domains… cognitive psychology for usability, brain physiology, and, well, Crime Scene Investigation!

Parallel #13

"Everybody's interested in coming in and going right to the dead body…"

Parallel: We tend to go right for the features that attract us or that are easy to test.

Parallel #14

"Juries expect a lot more, so in some cases, we have to entertain (re: animation) as well as inform…"

Parallel: Sometimes filing a bug is not enough, we have to be an advocate for what we find.

Parallel #15

"We must gather, document, and demonstrate in court that we did everything possible…"

Parallel: Software projects have "bug juries" that we are often called in to testify in front of to make our case.

Parallel #16

"Defense attorneys could discount elements of our case, so we have to be thorough and careful…"

Parallel: The same is true when we deal with programmers – we have to anticipate scrutiny.

Parallel #17

[Talked about how a boyfriend / girlfriend got into a fight and then violence happened…]

Parallel: We develop user stories and scenarios to test for bug pathologies in software.

Parallel #18

"We can't say 'this is what happened', but we can give a logical range of possibilities…"

Parallel: we're not always sure what the fault is, but we can suggest possibilities

Parallel #19

"Keep an open mind… don't make your evidence fit your theory…"

Parallel: be mindful of your biases… don't be fooled into thinking that this is a bug you've seen before

Parallel #20

"There is a high cost for processing evidence… homicides get priority…"

Parallel: There is a cost to doing tests… high risk features that lead to crash, hang or data loss get priority.

Intermittent Bugs: Observation

Bad observation
Irrelevant observation
Bad memory
Misattribution
Misrepresentation
Unreliable oracle
Unreliable communication

Link

Intermittent Bugs: System

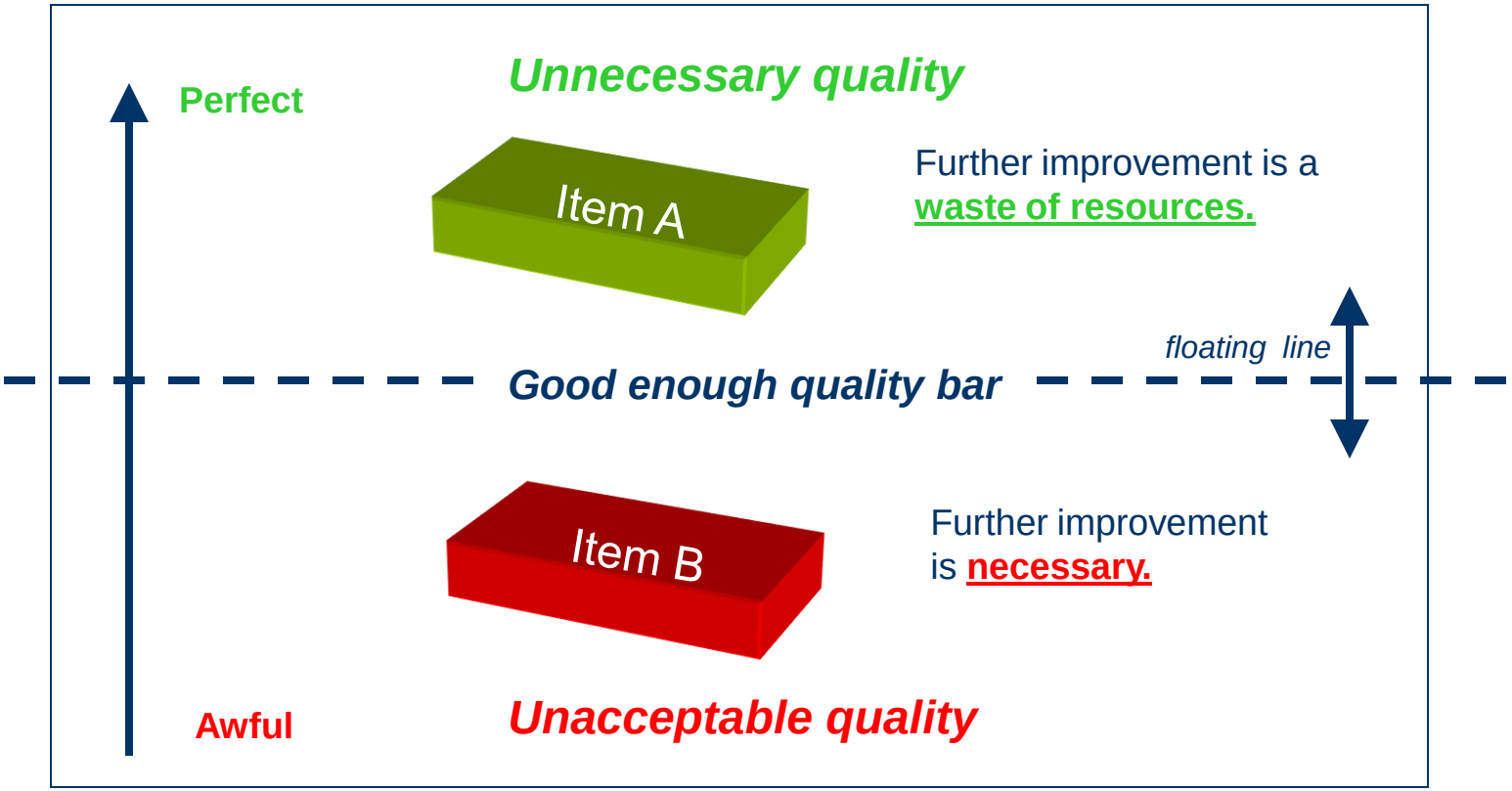
Purposeful change, and then back to original
Accidental change
Platform change
Flaky hardware
Trespassing system
Executable corruption
Component competition

Intermittent Bugs: Machine State

Frozen conditional
Improper Initialization
Resource denial
Progressive data corruption
Progressive destabilization
Overflow
Occasional functions
Different mode or option setting

Dynamic Quality Paradigm

It's more important to work on *Item B.*



Jon Bach / Quardev Laboratories, Seattle, WA 98119

Are We Done (Investigating?)

1) **Sufficient benefits**

2) **No critical problems**

3) **The benefits outweigh the problems**

4) **In the present situation, and all things considered, improvement would be more harmful than helpful**

The answer must be "Yes" to all four criteria

Jon Bach / Quardev Laboratories, Seattle, WA 98119