

Improving Software Quality

Bruce Logan

Brief Bio

- Originally from Zimbabwe
- School in Zimbabwe
- Studied Electrical Engineering

Career background

- Plant maintenance
- Stint at CSIR
- Systems integration & testing (Kentron)
- IT (technology & support)
- Software development
 - Project mgmt
 - Testing
 - Team mgmt
- Quality assurance

Systems & integration training

- Weapons systems – require systems approach
- System engineering training
- Complex systems testing – formal
- Written requirements – signed off by the customer

Software development experiences

- No written requirements – “make me one that looks like this!”
- Customer changes their wishes
- Expect that the schedule stays the same
- No time to do it right – end up patching & fixing
- Cost overrun – unhappy customer

V&V experiences

- Opportunity to set up V&V team
- Support from senior mgmt
- Project of limited scope
- Understand requirements in detail
- Draw up test spec according to requirements
- Skilled team
- Spectacular success
- Extend to other projects

V&V – success factors

- Limited scope
- Skilled team – *although not testers*
- Mgmt support – could build on early success
- Proved the formula repeatedly

Improving quality - overview

- ID areas for quick wins – get mgmt support
 - Formalise requirements?
 - Formalise test spec?
 - Feedback to developers?
- Follow through and get the wins
- Standardise relevant documentation
- Defect tracking & resolution
- Separate test environments
- Formalise process for delivering systems
 - Applicable to all sizes of systems

Improving Quality - suggestions

- Formalise the delivery process
 - Determine requirements
 - ID relevant documentation
 - Set out the gateways for progress
- Make teams part of the process
 - Make it positive – not a witch-hunt
 - Learn from mistakes
 - Make sure the process helps developers
- Engage with the customer – understand needs
- Get QA involved as early as possible

Improving quality – further suggestions

- ID your goal & focus on that
 - Don't get sidetracked or sucked in
- Keep QA function independent
- Tailor the generic process to fit your organisation

Summary

- Focus on the desired result
- QA is positive & a force for good
- QA can add large value on most projects
- Aim at building a quality process
 - “A quality process results in a quality product” – Ad Sparrius
 - “Problems downstream result from mistakes made upstream” – Ad Sparrius
- Early involvement – mistakes fixed more cheaply
- Tailor, tailor, tailor!

Discussions

- Group discussion lead by Stephen Quirke
- Insight:
 - Making the team part of the process is critical.
 - Management buy-in is very important.
 - Management leadership in formalising decisions is critical.
 - Standard structures for documents allow one to see the gaps.
 - Word template (.dot) works very well.
 - ISO – Say what you do and do what you say.

Discussions

- Something to add:
 - Work expands to fill the time available.
 - Because people do not understand the full process, developers use all the time for coding.
 - Similar for users, who do not understand the impact of changes.
 - Agile still build quality into the process, but do not inhibit changes in requirements.
 - Approach in QA should allow change.
 - Use cases are a good place to start test cases.
 - Be disciplined enough to throw away prototypes.
 - Often the cause of bugs later on.
 - Anger as a measure as a degree of meeting expectations.
 - Remember to calibrate client.
 - Quality testers/testing are important.

Discussions

- A question:
 - How do you capture learning in a way that will enable you to evolve?
 - How do you cut down scope to ensure that V&V team can be effective when you have a massive system to develop.
 - How do you implement QA when you do not have dedicated resources?
 - How do you make customers aware of their impact on delivery schedule if they keep on changing specification?
 - How do you marry agile principles with QA formality?
 - What does tailoring a process to an organisation actually mean?

Discussions

- A solution (not necessarily to the question above):
 - Addressing scope in massive projects:
 - Cut lowest priority features to deliver on time and budget.
 - Plan deliveries (iterations).
 - Changing user specifications:
 - Formalise change process and requests. Educate customer about impact on schedule, budget.
 - Incorporate time it takes to investigate change request.
 - Effect of context switching.