iv&v australia The independent software testing specialists

Welcome to IV&V Australia Testing Newsletter

This e-newsletter provides a practitioner's view of how to manage and perform SOFTWARE TESTING in today's world.

IN THIS ISSUE – May 2011

Maximise the value of testing – focus on the feedback loop

In many organisations the testers are little more than gatekeepers. They don't get to test the application until shortly before it is scheduled for release, and although they might identify various types of defects, there is usually no time for fixing anything other than "showstoppers".

This sort of testing isn't really adding quality to the product – it just identifies what needs to be done to avoid disaster on release, and the testers are seen as the bearers of bad news.

For testing to have a real impact on the quality of the product, it must be part of an effective feedback loop. The testers must identify issues and feed them back to the developers, and the developers must act on them in a timely manner. With an effective feedback loop the testers become an integral part of improving the quality of the product.

Enabling the feedback loop

To be effective, the feedback loop must be built into the process and culture of the development project. This requires a number of key things to occur:

- Involve testers with reviewing the requirements as soon as they are available, as they need to use these as their test basis.
- Deliver software to the testers early enough for there to be time to act on the test results. As there is rarely enough time "at the end", this means functionality is best released to test in an incremental fashion.
- Start testing early in the implementation phase. If the developers don't get any feedback until late in the process, there is too much "investment" in the code to make it practical to address pervasive issues.

- Test areas of functionality which are critical, risky, or particularly difficult first. This maximises the time available to the developers to address the difficult issues.
- Have a quick test-and-report cycle. If the feedback is too slow, the developers are likely to have moved on to other parts of the system, and so are likely to be slow to act on any problems found.
- Ensure the developers are prepared to seriously consider all problems reported by the testers. In some projects the developers adopt a "we know best" attitude that effectively prevents feedback from occurring – the reports are made, but no serious attempt is made to act on them.

Managing the feedback loop

To be effective, the feedback loop needs to be managed.

Some projects get by with informal methods such as free-form emails and verbal communication, but this rarely works except for simple developments with very small teams. Beyond that, some sort of Defect Management system becomes vital.

A defect management system doesn't need to be complicated or expensive.

In small projects, a simple spreadsheet accessible to all team members can work just fine, and there are lots of free defect management tools out there. If you are looking for a simple tool, don't get carried away will bells and whistles – you just need something that is:

- Easy to enter new reports – too hard and they won't be reported
- Has a basic work flow – who fills in which information, permissions, etc
- Provides simple status reports – so that problems don't just get forgotten about and the process can be easily monitored.

It is important that the Defect Management system is visible to everyone on the team. A good way to do this is to use a web-enabled tool and put it on your intranet.

Monitoring the feedback loop

Once you have an effective feedback loop operating, it can become an extremely powerful source of information. With very little overhead it can provide some very useful metrics:

- Cumulative number of problems raised over time. Although this is a difficult metric to interpret on its own (Is a low rate good? It might be due to good software, but it could equally be due to bad testing!) the shape of the curve says a lot. You should be looking for the curve to flatten well before release, indicating that fewer new problems are being found (even though testing is continuing). If it is still going up at a steady rate, there are still many more problems to be found.
- Cumulative number of problems closed over time. The gap between problems raised and problems closed should be narrowing in the lead up to release and should be close to zero at release. If it is still widening, you are not ready yet!
- Average elapsed time to fix problems. If this is long, then perhaps the developers are not spending enough time on fixing problems (maybe they have been re-assigned to other tasks), and the feedback loop will break down.

Alternatively, it may indicate that there are serious design problems making fixes very difficult.

- Fix effectiveness (i.e., proportion of fixes passed on retest). This should be close to 100%. If not, the feedback loop is falling down – maybe more information is required from the testers, or maybe the developers are not paying serious attention to the problems that are being seen. This could also indicate serious design problems making bug fixing very difficult.

In conclusion

Without a feedback loop, the testers are merely helping to minimise the risk of disaster. With it, they can be an integral part of building quality into the product.

Focus on the feedback loop. Enable it, manage it, and monitor it and you will have a powerful mechanism for improving your software quality.

Upcoming Courses @ IV&V

Practical Test Techniques (2 day course)
- Sydney – 31st May/1st June
- Canberra – June 21st/22nd

Introduction to Scripting (1 day course)
- Sydney – June 15th
- Canberra – June 20th

End to End Software Testing (2 day course)
- On request

Test and Governance (2 day course)
- On request

For more information on these courses, visit our website or contact us at info@ivvaust.com.au

What is included? All catering (arrival tea/coffee, lunch, morning/afternoon tea), course notes and interactive examples.

Alice's Adventures

Chapter 9 – Back to work, so what's next? It's time to track progress

Alice is back to work after the Christmas holidays and long service leave, and she didn't think about work at all! She had better dust off the cobwebs and come up to speed.

Before Christmas, Alice, Max and their teams were thinking about how they were going to prove that their new "buddy system" approach to developing and testing the Whizzbang Gadget was leading to a better outcome than if they had struggled on as before by throwing problems back and forth over the wall to each other. What was going to make management sit up and take notice?

They decided that they would measure *defects found in the field*. If they found fewer defects in the field than on previous releases, then they could assume that they were finding more defects during their improved Dev/Test process – and hopefully also *reducing* the number of defects *because* of their improved process!

Alice and Max knew that this was a good metric, but in the meantime they needed to make sure that they would get there in the end – that they were doing the right things now (at Build 1) to lead to the right outcome on release (at the end of Build 3). So this is what they were doing:

- They'd worked out between them the most appropriate test techniques to be done by which team, at unit test, functional test and system test.

- Max's Dev Team were working closer (and earlier) with the Testers, so they were finding and fixing defects earlier. This would lead to less effort in bug fixing down the track when the system was larger and more complex and harder to change.

- Alice's Test team didn't have to write such basic tests as they'd done in the past because the Dev team were unit testing thoroughly, so they were able to concentrate on more complex tests and therefore find some of the more obscure bugs. They still did random checks, but overall felt that they were getting more effective testing for a reduced effort.

What they decided to do is to "snapshot" the test results at various points during the development process, so they could see the trends of the number of defects found, number still open, severities, etc. Hopefully the trend would show that defects were levelling off over time and being fixed on time, which would lead to confidence in release readiness down the track.

They also gave strict instructions to their staff about how to fill in their timesheets (rather than just put 40 hours on a generic task code like "Dev" or "Test"). Particularly if they could track effort on tasks such as rounds of bug fixing and regression test runs, they could see how much time was being spent on remediation.

They could then show their managers the trend over builds and how the whole software development and testing process had improved. Fewer bugs, happier teams!

Thought of the day

Testing: the slaying of a beautiful design by an ugly fact

FEEDBACK

Have you found this issue useful? We want to hear your comments and suggestions. Email us at info@ivvaust.com.au.
For more information about IV&V Australia, visit our web site at http://www.ivvaust.com.au.

If you do not want to receive further correspondence, please respond to subscribe@ivvaust.com.au with "unsubscribe" in the subject line.