

Quardev Quarterly

Volume 1, Issue 1

June 2007

Quardev, Inc.

What Do You Mean “There’s Nothing to Test”?

How QA Can Help Keep Projects On Target Before Coding Starts

By Fred Hagan, Managing Test Lead, Quardev, Inc.

Over the past few years, it’s been apparent that the traditional view of QA’s role in software development is changing. More and more, the view that QA is not merely an evaluative activity that comes at the very end of the project lifecycle seems to be taking hold. Steve McConnell’s statement in *Rapid Development* that up to 70% of the defects in an application can be detected before any code is written is becoming widely known in the world of QA and development process models. And that’s good news.

However, while many organizations may agree that QA should be involved very early on in the development lifecycle, it’s sometimes little more than lip service. Some may wonder exactly what it means to involve QA from the initial phases of a project, and how one goes about doing it. Other than producing a very high-level and tentative draft of a test plan, what can QA really contribute before there’s any code written? What do you test when there’s nothing to test?

Requirements, Design, and Risk

Now, when one is testing an application or system, presumably the tests are based on the requirements and specifications. But what’s sometimes forgotten is that the requirements and specification documents themselves are legitimate test items. James Bach, principal consultant at Satisfice, Inc. and one of the pioneers of Exploratory Testing as an actual discipline, has made this observation: “Because of incompleteness and ambiguity [of requirements], testing should not be considered merely as an evaluative process. It is also a process of exploring the meaning and implications of requirements.” (Bach, James. “Risk and Requirements Based Testing” in *Computer*, June 1999)

As Bach’s statement suggests, the documented requirements of a system or application, whether documented explicitly or implied, are project artifacts worthy of QA scrutiny. In fact, the requirements as documented may, and probably do, have

(Continued on page 2)

Inside this issue:

Writing with Controlled Language	3
STAREAST Recap	3
Should I be Certified?	4
Tech Talk: Web 2.0	5
Upcoming Events	6

SASQAG Celebrates its 10th Anniversary!

This year, the Seattle Area Software Quality Assurance Group (sasqag.org) celebrates 10 years of support to the Pacific Northwest Quality Assurance and Quality Control community. Since the first public meeting in the fall of 1997, SASQAG has continued to hold monthly educational and networking meetings on the third Thursday of each month (except December).

SASQAG’s purpose is to promote professional software quality practices in a non-commercial atmosphere. SASQAG operates independently from any specific software

quality organization, choosing instead to take advantage of associating itself with the full range of local, regional, national and international organizations whose missions align with ours.

Visit <http://www.sasqag.org> to see what anniversary activities and programs are being planned throughout 2007-2008. Membership is free and only requires an email to membership@sasqag.org with your email address, your name, company, and the kind of work you do.

What Do You Mean “There’s Nothing to Test”?, Continued

“bugs” that should be logged and tracked the same as bugs that will be reported later on against the actual code. Most projects would benefit from having requirements- and design-phase bugs as *major deliverables* from QA during the project’s early stages. Why?

Well, to begin with, consider the following scenarios. Any of them sound familiar?

The members of the development team and the customer, or end user, may come from different worlds, and may speak and understand different cultural tongues in accord with the point of view of their jobs. So it’s possible that everyone may think they all understand each other perfectly – until the product is actually created.

Developers may try to make requirements fit an already partially existing solution rather than building a new solution specific to the requirements..

The client may change his mind about what he wants, or may add new requirements late in the process, causing design and functional specs to change even after the coding has started. In this scenario, the programmers are trying to code to a moving target, so all bets are off.

Requirements might be “analyzed” by programmers or engineers who lack the domain knowledge to truly *get* the business need or the reason for certain requirements. In such cases, the development team just doesn’t fully understand the problem that the requirement is trying to solve. And nobody even knows there’s a misunderstanding until later. Requirements that are *implied* rather than explicitly stated may contribute to this problem.

Testing in the Requirements Gathering and Design Phases

Three things: Quality Criteria, Techniques, Deliverables. Any conversation about testing in the Requirements and Design phases should break out into those three topics. If the Requirements and design document are artifacts to be tested before code, then what are the **quality criteria** that apply to them? On what basis are they evaluated as “good” or “not good,” and with what sort of **test techniques** do we identify and articulate “bugs” to be reported against them? And finally, what QA artifacts do we create as **deliverables** or **work products** specific to the requirements definition and design phases of the project?

For documented requirements, the single most important quality criterion is that each requirement be testable. That is, can a test case involving specific

observable items or behavior, or specific actions by the user such as click steps or other inputs, be written for it? Is each requirement unambiguous, with clear pass/fail criteria? Do any requirements “imply” other requirements that are not explicitly stated?

If writing a reasonable test case is not possible, then that’s a bug against the requirement, as the requirement is untestable. Log it in your bug tracking system. The bug is resolved when the requirement is stated in a way that allows a specific test case to be designed and written for it.

The act of writing the test cases for each requirement will often reveal “implied” requirements or specifications that should be noted by the QA lead, verified with the customer and the business analyst, and then communicated to the PM and the designer. Although not technically bugs, it’s a good idea to document and track them in the bug database so that they will be taken into account and not forgotten about as the application is being designed.

Maybe they seem obvious, but these “obvious” requirements may not be so obvious to the programmers if they’re not spelled out explicitly in the specs from which they write their code. That increases the chances that they might build an application that works very well, but is not the application that the client wanted. We’re talking about thousands of man-hours and dollars spent to build the wrong product. Maybe it’s a near miss, but wrong all the same if it’s not exactly what the customer wanted.

Requirement and design spec bugs may be resolved and closed as soon as they are written or revised in the requirements and specs, which should take a very short time and be very inexpensive. Compare that time and expense to re-designing a portion of the product and re-writing who knows how much code farther downstream. That’s pretty easy math. Still think testing is what happens at the END of the development lifecycle?

To see the full article, please go to the following link on our Web site: www.quardev.com/articles/.

“Documented requirements of a system or application, whether documented explicitly or implied, are project artifacts worthy of QA scrutiny.”

Writing Content for Controlled Language Shelly Dillon, Manager Technical Writing Division, Quardev, Inc.

More and more we are coming into an era where content needs to be reused and repurposed as much as possible. This isn't to suggest that there is or will be less writing happening, but rather that there is so much writing and so many audiences for the same information that it is important to use content in new ways.

A common reuse of content is through translating a document into numerous languages for multiple audiences. While the same subject matter may be appropriate for each audience within each language, it is difficult, and expensive, to translate text riddled with jargon, filled with ambiguous words or awkwardly structured sentences. To complicate this even further, many translation services use some form of machine translation to translate common words once rather than paying for the same word to be translated multiple times.

Common Rules for Standardized Results

One way to help ensure content is ready for both machine translation and multiple audiences is to create a system for writing in a controlled language. While there is no such thing as an agreed upon or universal controlled language there are common rules that exist in some form that can be used as a basis for creating a customized controlled language policy. Once created, all writers and editors must be able to work within the model in which the vocabulary and syntax is greatly reduced.

Common rules for universal translation include the following:

- Write in short sentences
- Use standard English word order
- Include all optional words
- Use words and phrases consistently
- Avoid ambiguity

Global Audience vs. Machine Translation

When using a controlled language, it is important to understand how content will be used and repurposed. For example, some content may need to be prepared for a global English audience but not for machine translation, while other content must be prepared specifically for machine translation. It is ideal to have two sets of rules, the rules appropriate for the global English audience and those for machine translation. Writers and editors need to fully understand both sets of rules. Typically rules for the global English audience apply equally to machine translation but conversely machine translation rules do not always apply to writing for the global audience.

Global English rules that also apply to machine translation include the following:

- Don't use a "to be" verb when a stronger verb is appropriate
- Don't use nominalizations
- Avoid stacking prepositional phrases
- Avoid noun stacks (overly modified nouns)
- Avoid wordiness and jargon

(Continued on page 5)



“Writing and editing in a controlled language environment is not easy, and certainly not for everyone.”

STAREAST Recap

Jon Bach, Manager for Corporate Intellect, Quardev, Inc.

SQE, the producers of the nation's biggest testing conferences called STAR (Software Testing Analysis and Review), just wrapped up their STAREAST event in Orlando. I've been going to STAR since 2000 because I get to jam with so many speakers whose insight sharpens my notions about software testing.

When conference attendees return to their hotel rooms for the day, there are about 10 of us who linger in the restaurant with our notebooks open, challenging each other with testing problems, discussing project and personnel pathologies, and looking for inspi-

ration for the next “cool talk” we want to present. For me, it was the notion of “scripted vs. exploratory testing” and which is more effective. I'll have thoughts about that in an upcoming Blog.

To see Jon Bach's Blog, “Notes, Bugs, and Issues,” go to Quardev's Web Site: <http://www.Quardev.com/blog>



Should I be certified?

5 Questions with Cem Kaner on Certifying Software Testers

Cem Kaner with Jon Bach



“I don't think the [certification] exams tell you much about the competence, knowledge, or skill of a tester.”

Do you see a trend toward certifying software testers?

Certifying software testers has become a significant business. Exams and review courses cost a lot of money—in some cases what seems to be an outrageous amount of money, which seems to buy a lot of advertising to convince people that they need to be certified or to favor certified testers in their hiring.

Some advertisements suggest that certification indicates the competence or professionalism of a tester. What do you think?

I don't think the exams tell you much about the competence, knowledge, or skill of a tester.

I have reviewed syllabi or study guides for the ASQ (American Society for Quality), ISEB (British Computer Society) and ISTQB (a group that seems primarily a marketing coalition for software testing trainers/consultants, who give themselves the grandiose name, International Software Testing Qualifications Board). The study materials strike me as outdated and oversimplified. The exams are completely or heavily multiple choice, which typically test memorizable knowledge or simple applications of basic skills. I've seen samples of ASQ and ISTQB exam questions and was not impressed by them.

I have a higher opinion of course-based certifications like UC Berkeley's and Magdy Hanna's IIST. The value of these lies in the courses, not the exams. I also think that the key people in QAI (Quality Assurance Institute) are trying to use certification to encourage people to develop a better understanding of what QAI considers good testing, rather than to turn a relatively easy buck. I haven't studied the new (essay style) QAI exam and don't yet have an opinion as to what the new certificate tells me about the knowledge or skill of the tester.

If you were hiring testers, would you pay more attention to someone who held one of these certificates?

I used to credit people who were certified with serious motivation—it takes time and money to prepare for these exams. But so many people have been sold on the idea that a certificate is an entry-level requirement for a testing job that this has become more of a chore for many people than a reflection of a commitment.

To the extent that someone certified by ASQ, ISEB or ISTQB adopted the attitudes and technical ideas that I perceive in their materials, I would expect to have a retraining cost. At this point, I would favor someone with no certification over someone with one of these certifications.

As to the university or IIST certifications, lots of people include lists of test-relevant courses

taken and books read in their resumes. I treat courses (and books) as courses—some are good, some are worthless, some are toxic. They are all good to know about. They make good topics for a detailed interview and the hiring decision is based on that discussion, not on the list or the associated piece(s) of paper.

What about the new "open" certification? Aren't you part of that?

A year ago, several leading testers met after the Conference of the Association for Software Testing (CAST) to plan a new model for testing certification. We want to honor the diversity of views in the field, not impose an orthodoxy. We are currently testing a question server that allows public review and criticism of each question in the database—we'll unveil the official 1.0 release at this year's CAST. An exam will be a targeted sample of the very large pool of questions in the database. By "targeted," I mean that we'll support several different exams that reflect several different views of testing. Sample exams will also be publicly visible, as will notes on the design of each exam. This is part of what we mean by "open." Additionally, questions will be open source (reusable by anyone who wants them for their courses or exams) and the exams themselves will be free (or perhaps for an administrative cost, if the candidate wants to take the exam in a supervised, identity-verified setting). Some (maybe all) of the exams will be based completely on courses and readings available for free on the web. Our goal is to make the exam, including preparation cost, affordable everywhere in the world.

The exam will still be computer-scorable and so, in my opinion, it will still not be a good test of skill. Some day, we will figure out how to cost-effectively run a low-cost skill-focused exam. But for now, we think that if a large segment of our community is going to take a computer-scorable exam, we can create one that is better—and free.

For those interested, how to get involved?

The Second Workshop on Open Certification (WOC 2) will be in Bellevue, right after CAST (July 12 to 14, 2007). We need help creating questions and designing and creating the exam server. We would also greatly appreciate corporate donations to cover the cost of some of the development and editorial staff. Most work will continue to be done by volunteers, but we're at the point where some modestly-paid staff could have a big impact on our rate of development.

For more information, contact Cem Kaner

(kaner@kaner.com) or Mike Kelly

(mike@michaeldkelly.com) or see [http://](http://www.freetestingcertification.com/)

www.freetestingcertification.com/

Tech Talk–Web 2.0

Pat Loughery, Managing Test Lead, Quardev, Inc.

By nature, the Web is in a constant state of change. Traditionally, transition in the online space has been rapid and disconnected, but recently a number of paradigm shifts have emerged in a way that changes many of the ways that the Web operates. This transition has been labeled Web 2.0.

The term Web 2.0 was coined in 2003 and took on broad popularity in 2004 with the first Web 2.0 conference. Although there is no commonly accepted definition of the term, it generally includes some basic concepts.

Web 2.0 Basics

Software that improves as you use it. Web sites and applications identified as Web 2.0 are frequently social software sites that facilitate a high amount of interactivity with a low barrier for entry and participation. In these sites, it is the site user who is responsible for the content, and the site designer who is responsible for the framework. Wikipedia (www.wikipedia.org) is a great example of this concept. Wikipedia is a collaborative encyclopedia where anybody can add or modify an entry, and a system of checks and balances minimizes the risk of incorrect data being published – at least for very long.

Data is the driving force. In their current form, many Web 2.0 applications focus the most energy on data interaction and very little on user interface design. Web 2.0 applications tend to be sparsely designed but content-rich. Applications like Google Homepage (www.google.com/ig) or MySpace (www.myspace.com) show very simple (in the case of Google) or bauble-filled (in the case of MySpace) interface design, but the core of the system is the data that is being presented to the user.

The network as a platform. Web 2.0 applications are most frequently delivered to users as Web sites accessible through a browser. Usually no software needs to be installed.

The perpetual beta phase. Most Web 2.0 applications are incrementally improved, with very infrequent releases. Flickr (www.flickr.org) is an example of a feature-rich site that is several years into its beta phase. On Flickr, a photo sharing community, a user can interact with other users and their photos by adding comments, marking favorites, or adding labels. A photo owner can join a variety of topical groups and post photos to the group's photo pool, or participate in bulletin board style discussions. Another well-designed example of this concept is LibraryThing (www.librarything.com), a social software site for users to track, rate, and comment on other user's book collections.

Mash-ups, syndication and Ajax inspire a customizable user experience. One of the great strengths of Web 2.0 systems is that they are frequently highly modifiable. With syndicated data being presented in Ajax (or Ajax-like) forms, Web 2.0 sites have a buffet-style feel: take a bit of this and add a bit of that, and put it on your own unique plate. The Netvibes (www.netvibes.com) homepage is an effective starting point for this area. The myriad of applications built on top of Google Maps and the Google Maps API are fun and helpful tools – try RealEstateABC (www.realestateabc.com) for a feel of this experience.

Although Web 2.0 is a flexible label and there is no standard forthcoming, these concepts are widely accepted as identity markers for this new wave of software design and delivery. Enjoy your exploration in this rapidly changing area!

Pat Loughery is a Managing Test Lead at Quardev and is a social software junkie. Try taking away his flickr, LibraryThing, Last.fm, Wordpress blog, Twitter, Facebook, MySpace, del.icio.us or Postnuke sites and you may be bitten.

“Although Web 2.0 is a flexible label there is no standard forthcoming.”

Writing Content for Controlled Language, Continued

Machine translation-only rules include the following:

- Use sentence capitalization for headings and titles
- Avoid coordinating verb phrases
- Ensure all nouns have an article, an adjective, or some way of signaling to a machine that a noun is next
- Don't use parentheses or dashes for notes or instructions to the reader

Writing and editing in a controlled language environment is not easy, and certainly not for everyone. It can be difficult to adjust to the language constraints but if you can create the systems around the content development process, you can realize savings – from both reduced translation costs and the ability to more readily repurpose content.



Quardev, Inc.

Quardev, Inc.
3411 Thorndyke Ave W.
Seattle, WA 98119
Quardev.com

Phone: 206-547-7771
Toll Free: 866.305.4843
Fax: 206-547-7776
E-mail: info@Quardev.com

Integrity, Service, Solutions.



Calendar of Events–Quarter 3



July 9-11, 2007

Conference of the Association for Software Testing (CAST)

CAST is a testing conference right here in the Seattle area. It includes three days of facilitated presentations by several industry experts, including Cem Kaner, James Bach, Robert Sabourin, Esther Derby, Harry Robinson, and more.

See the conference Web site for more information: <http://www.associationforsoftwaretesting.org/conference/index.html>

Tuesday, July 10, 2007

July WSA QA Workgroup*

Special Edition: Tester Exhibition

On a special night and location – Tuesday, July 10, 2007, 6:30 PM at the Meydenbauer Center in Bellevue.

See more information on the QA Workgroup Web site: <http://www.QASIG.org>