



Quardev Quarterly

Volume 1, Issue 2

September 2007

The Tester and Writer—Partners for Quality

Jon Bach, Manager of Corporate Intellect and Shelly Dillon, Technical Writing Manager, Quardev, Inc.

Inside this issue:

The Tester and Writer—Partners for Quality	1
Message Intended vs. Message Received	1
The Art of Bug Investigation	3
Upcoming Events	8

Imagine that it's time for a triage meeting – a meeting where talented people on a software project assemble to discuss the current status of the project, including risks and issues (the outstanding Bug List). It's likely the following 8 stakeholders would gather around a conference table:

1. the lead programmer
2. the marketing rep
3. the tech support lead
4. a program manager
5. a salesperson
6. the release manager
7. the test manager
8. and the documentation lead

Imagine vigorous discussion about what to fix and what not to fix. Everybody has a different opinion and many changes are discussed. Even-

tually, everybody agrees with the course of action. But look closely and you'll notice that two people may be quieter than the rest – the test manager (7) and the documentation lead (8).

These two are usually last to leave the room. After discussion has died down and everyone else is gone, they are usually still seated, looking over the notes they took because the discussions impacted them similarly – the tester has new tests to design and previous tests that now need tweaking; the writer has new modules to write and previous modules that need rewriting.

Any change in functionality means a change for both of them. That's why the relationship between them is important. While people in other project roles may be isolated, for the writer

(Continued on page 4)

Message Intended vs. Message Received

Kristelle Sim, Principal, Silver Wings Editing

You may have heard this before: everything you publish, print, or put up on your Web site sends a message about your company.

It's true. Every public document, Web site, Web page, ad, presentation, and brochure has a powerful impact on your customer's experience, and based on the quality of these items, your customers will create assumptions, make judgments, and ultimately, decide whether or not to buy what you're selling.

Yes, you know this is Marketing 101, so you've put lots of time and effort into your DVD presentation, or your tri-fold brochure. You've got your Web site designed with the right colors, and you've put in the right data to demonstrate your company's value and show everyone just how unique you are.

Problem is, most clients I've worked with have done all these things, and still have overlooked a major obstacle to getting their message across - an obstacle that

(Continued on page 2)



Message Intended vs. Message Received, Continued

(Continued from page 1)

can have a powerful impact on their potential customers:

The typo.

No matter how colorful the page, how brilliantly stated the purpose, how cool the Flash presentation, this silent *killer* of credibility can derail you. Two of them and you're going down. Three of them... now that's just sad.

I've seen them in best-selling books, Fortune 500 Web sites, and billion-dollar corporate brochures. I've seen hundreds of things, like:

- "if" instead of "it"
 - "there" instead of "their"
 - "it's" instead of "its"
 - a comma instead of a period
 - words initial-capped in some places, and not in others
 - quotation marks that begin, but don't end
- or,
- the inexplicable misspellings despite easily-available spelling tools

Now, you might be thinking: *What's the big deal? What's a little thing here and there? First, everybody makes mistakes, and second, it's not that important.*

Yes, everybody makes mistakes - but you don't have to let your mistakes get into print. And, yes, you're right... the tiny error isn't important, but the IMPACT is!

Here's an example:

Last week, I was handed a very high-level aerospace technology brochure, written primarily in Gobbledygook. Maybe this company assumed that its clients were fluent in the chosen jargon (assumptions about the reader are the topic for another article), but despite that, I noticed that they used the word "compliment" instead of "complement" - not just once, but twice. The context was obvious to me. They meant *complement*, as in "something that completes," not *compliment*, as in "a flattering comment."

I was perplexed. This brochure was about aerospace technology, an anchor for our technological future, and they didn't know which word to use?

Yes, it's just a little mistake. But with a mistake like this, the "Fog of Doubt" can start to creep in, and

with it, some common assumptions and judgments that your target audience might have, like:

1. They don't care enough to mind the details in a brochure... how do I know they'll care enough about the details in the machine I'm buying?
2. They glossed over this error because of their rush to get this out... what will they gloss over in the rush to finish my million-dollar airplane?
3. This is a sixth-grade vocabulary mistake. Do they think I'm not educated enough to know proper English?
4. I'm a high official in charge of a billion-dollar budget for our defense systems. Should I put my reputation in the hands of a company that can't be trusted to proof their own brochure?
5. If something like THIS has gotten past them, does that mean I'm going to have to watch them like a hawk on everything?

Once these assumptions and judgments are made (and they're made almost instantly in the flash of a thought), they lead inevitably to decisions. And your client's decisions are important ones - decisions to respect, or to trust, or to buy... or NOT. If there's too much of this fog clouding your client's mind, your message won't get through it, and their decisions will be affected by it.

So what can you do to increase the likelihood that your message gets through clearly, without the fog? Depending on your business size, here are some ideas:

- First, make a commitment to having only the highest-quality, error-free materials in every area of your business. (It sounds like a no-brainer, but it's rarely done!)
- Hire the best writing staff you can get, and make sure that no one proofs their own writing.
- If you have a lot of materials, develop a style sheet for them which describes common usage so that even new writing staff can easily step in and know what is capitalized, what terms are used and how, etc.
- Spend the money on a good outside copyeditor/proofreader if you find that your current staff is overwhelmed. Outside eyes can save you!
- If you're a one-person show, don't send out something just after you've written it. Take a break and do something else, then come back and see your document with "fresh eyes." Try to see what's really there - not just what you "meant" to write!

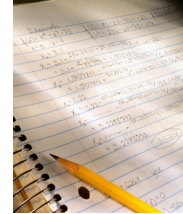
Message Intended vs. Message Received, Continued

(Continued from page 2)

Ralph Waldo Emerson once said: "What you do speaks so loud, I can't hear what you're saying."

So what are your Web sites, brochures, and manuals doing? Are they helping your message come through clearly, or is your customer put off by the fog?

Kristelle Sim (kristelle@dashoflife.com) is the principal for Silver Wings Editing (<http://www.silver-wings-editing.com>) and has been an editor since she was 8. She cannot read anything without proofing it.



Tech Talk: The Art of Bug Investigation

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

Testers find problems. The art of the search compels us like famous explorers.

But what do we do when we actually find something?

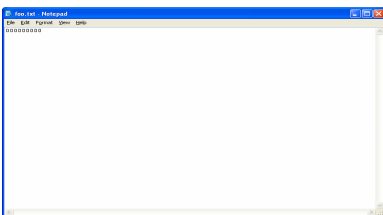
Bug investigation is something we don't talk about enough in our business, but it's the activity that contains some of the most important and compelling lessons about testing.

In that spirit, let me host an exercise:

If you've got a PC near you running Windows, open Notepad.

Type "this app can break," (all letters must be lowercase) save the file, close it, and re-open it.

You'll probably see a series of rectangles.



This is a bug, right? It's obvious that Notepad was not meant to do this to your file. Some testers would file a report titled "Rectangles seen in place of text when saving file," but that's just the failure. What's the underlying fault?

Bug investigation is a journey. It always starts with a conjecture.

For example: "Did someone hack my system long ago and is this some kind of time bomb or virus?"

If your conjecture is that it's a virus, you could run a virus check.

If the scan turns up negative, here are some choices you could make:

- Decide if you believe the virus check is accurate.
- Find out if there is a set of virus definitions you haven't yet downloaded.
- Try a scan with another anti-virus program and see what it finds.
- Look for evidence of a rootkit, or some other "can opener" a hacker has placed on your machine.

If the scan turns up positive, you have to investigate whether those viruses would create such a condition.

But let's put that aside for now and assume it isn't a virus. We'd need another conjecture to fuel our investigation.

Could this be some sort of issue within Notepad's code?

If so, here are some investigative choices:

- Type into the corrupted file to see if it accepts data
- Try it on another PC
- Reboot and try it again
- Research Knowledge Base articles online

If you try choice #1, you'll see that you can indeed type in there, but what to do next?

(Continued on page 6)

"Bug investigation is something we don't talk about enough in our business, but it's the activity that contains some of the most important and compelling lessons about testing."

The Tester and Writer—Partners for Quality, Continued

and the tester to be in silos could mean big risks to the project.

The relationship between a writer and a tester is virtually identical to the relationship between a writer and an *editor*. Editing is the discovery of problems – as is testing. Editing is about providing suggestions for better usage – testers do that, too. Editing is a process of conformance to style – testing often involves conformance to specifications.

This article is being written by a tester (Jon) and a writer (Shelly). As we planned this article, we talked with each other about the differences in our jobs.

For example, from Shelly’s perspective, writers may find problems for testers to report, but it’s not the writer’s job to file bugs.

From Jon’s perspective, writers want the product to work so they can document its behavior. Testers want to find the *failures* in the software.

Other than that, we did not come up with any significant differences between our roles! What came up for us in planning this article is that we had more in common than we realized.

Here are 10 important categories of perspectives that testers and writers share:

- 1) **Users** — Since there may not be convenient access to end-users, writers and testers need models, use cases, and scenarios to keep in mind. Testers and writers are often involved in usability testing – a process of bringing targeted users in and seeing how they would use the software. If either the writer or tester was not part of that process, each could ask the other logic-oriented questions like, “Would users do this?”
- 2) **Expectations** — Writers and testers may decide to talk about features or behaviors they have seen during the course of their work that were not in the specification. These could be things that were *supposed* to be in the spec or things that were not designed to work a certain way, in effect, answering the questions: Is this a bug or by design?
- 3) **Obstacles** — Both want to know the troublesome or blocking areas so we can avoid writing or testing in those areas for the time being. Knowing the areas to avoid and taking time to focus on more productive tasks – such as speaking to the writer about how things are supposed to work once they are fixed (for the tester) or asking for an older build or existing test cases (for the writer) – is a huge time saver.
- 4) **Information** — Each of us might ask the other, “Have you been to any meetings (or received any email) that might shed some new light on features or decisions *about* features? Or, more simply, “Do you have any versions of documents newer than mine that could help me understand functionality?”
- 5) **Collaboration** — Both testers and writers are in service to project stakeholders (anyone with a vested interest in the success of the project) – getting them to give us relevant information to do our jobs effectively. We know that not everyone is invited or can attend all the necessary meetings, nor would every important issue be found in the bug database. Therefore, each day is a potential surprise for both the tester and the writer, so keeping in close contact is valuable.
- 6) **Value** — Testers and writers are



“What came up for us in planning this article is that we had more in common than we realized.”

The Tester and Writer—Partners for Quality, Continued

often asked to justify their value and we both might go about it the same way by saying, “Without giving us the tools to do our jobs, it might lead to an error in omission, which in turn leads to an expensive call to tech support. Both of us want to provide the user with software that works (defined here as “meets the right requirements to the right degrees”).

- 7) **Marketing** — Knowing the marketing message is important for each of us. Testers want to be told what promises the marketing staff is making so they can test those claims. Writers want to be able to take what the marketer envisions and turn it into something the reader can “see.”
- 8) **Scope** — We need an anchoring mission. There needs to be meaning in the information we gather about the Who (the audience) and How (the plan), and we have to know What (the scope).
- 9) **Delivery** — We both need to define the ways we will deliver artifacts in accordance with our mission. For the writer, this could be in the form of user manuals, specifications, user guides, training guides, whitepapers, or development documents. For the tester, it is status dashboards, bugs, coverage metrics, risk checklists, test plans, scenarios, or test cases.
- 10) **Risk** — Last, there is that triage meeting. Testers and writers will be listening closely, taking notes, and thinking “What do I need to change now that decisions have been made about what to fix and what not to fix?” Even if a decision was deferred, both will have to be vigilant about checking back with the stakeholders, because it will impact both of their strategies for achieving their writing and testing missions. At Quardev, we have daily “huddles” – 15 minute meetings that include the test leads and tech leads – where we talk about our daily pro-

ject goals in case it impacts others.

In the end, the whole team, from program manager to marketing, is trying to deliver the best product possible – one that will inspire customers and potential customers. Within this team there is great opportunity, and a natural tendency, for testing and writing to collaborate and support each other’s mission.

What we want to impart is that the mentality of both testers and writers is about providing service. After collecting, organizing and processing information, if the two are kept separate from each other and stay in their respective departments or if communication is strained in some way, chances are good that inconsistent messages will leak out to the user, either in the form of bugs or in confusing documents.

Tech Talk: The Art of Bug Investigation, Continued

(Continued from page 3)

Should you:

1. Save the file?
2. Close and reload the file?
3. Save a copy of it?

If you save a copy of it, you'll see that when it's re-opened, it is still corrupted.

Conjecture:

1. Maybe the text is corrupted?
2. Maybe it's being displayed in a certain way?
3. Maybe it was saved to the wrong place?

Going down any of these new paths is called "branching" – forming new paths to explore based on new context.

How to get context? Try to refute your own conjectures.

For example, conjecture #2 is unlikely because Notepad has no word processing logic built into it like its more sophisticated cousin, Microsoft Word. It's just a text editor.

To know this, you have to have an idea of oracles – methods or principles you use in your testing and investigations to identify problems. Oracles can be specifications, emails from stakeholders, comparisons with past products, or competing products. If whatever oracle you consult says Notepad is just a text editor, then you can likely rule out corrupted text.

(At this point in your investigation, how are you logging your travels, ideas and refutations? Lab procedures, the way you follow your logic to capture and present important details in your testing, is another factor that can make a good investigation.)

What about tools that might help?

(At this point, you may decide to get technical, like using an older program called "List" which lists the context of files. If you did, you'd notice that there are spaces between the characters.)

This may cause you to have a conjecture that maybe when viewed in hexadecimal, the file will give you a clue to its behavior?

If so, you'd see that the first two characters are FF and FE.

Is that significant?

In these days of the ubiquitous nature of Google and MSN Live Search, you can find out fairly easily. My search brought me to http://en.wikipedia.org/wiki/Byte_Order_Mark where I learned that, yes, it could be very significant.

If you type new text into the file, you'll see that it is not in Unicode, just the old text, so somehow, Notepad is interpreting the old text as Unicode.

Online, on that same Wikipedia page, you might notice an entry for UTF 16, which refers to the ordering of bytes.

Ordering. Hmm...

Conjecture:

What if you use a Hex editor to change it to FF FF or FE FF? Will Notepad interpret the encoding differently when you re-open the file?

If you run "list" on the original file, you might see that it does not have the FF and FE.

Conjecture:

Maybe it adds the FF and FE later?

Action:

Start a new file with a new instance of Notepad. Call it a new name (here is where careful lab procedures come in handy so you don't get confused as to what your path is).

Conjecture:

What if you save the empty file as a file type "Unicode big endian" in the Save dialog?

If you do a "List," you can see that all it has in it is FE and FF!

Conjecture:

If we add text into the file now, perhaps it would be added in two-byte blocks.

(Just as you're getting the idea that this data wasn't corrupted, just changed in its representation, another tester might walk in with an idea: Create a new file and type: "this app can work". This is an example of "paired testing" – using someone else's brain to test because

“Going down any of these new paths is called “branching” – forming new paths to explore based on new context.”

Tech Talk: The Art of Bug Investigation, Continued

(Continued from page 6)

they may be biased in different ways than you are.)

You'll see that this idea is a good one because there is no corruption when the file is saved and re-opened!

Conjecture:

Change the text to "this app can brea" (no k).

This starts you on the path of a bisection search – taking point A and point B and narrowing down where between the two points that the behavior starts occurring. You could also try it on another machine (or a virtual machine).

You'll see that "this app can brea" works – no corruption.

Conjecture:

Put the k back in at the end of the phrase.

Result:

It fails.

Time to take a break from data gathering mode and kick into data analysis mode. This is a cognitive "polarity." There are times when you need more data to analyze and times when you have enough data and need to look for patterns.

So, let's analyze. What do we know? What can we see that's right in front of us?

On my machine, I see the following:

1. There are four words.
2. Two of the words are three letters long.
3. There are 18 total characters (including spaces).
4. The file has a name.
5. The file is saved.
6. The font is Arial.
7. The point size is 12.
8. The words are in English.
9. None of the letters are capitalized.
10. There are 5 vowels.
11. There are 10 consonants.
12. There is no punctuation.
13. There are no numbers.
14. The file has a certain size (in bytes).

From this, are there any new ideas for paths to take, any new conjectures?

(At this point, you may want to check how much your investigation is costing in terms of time or money. Can you reduce the cost of this investigation? Have you gotten far enough along where the people you are in service to wouldn't be annoyed?)

I used to struggle with how much investigation was enough. I mean, I could spend all day tracking down the cause of a bug. Is that a good thing or a bad thing?

Then I found a useful perspective – from the world of economics. It's called "satisficing" – to obtain a result that is good enough. "Good enough" in this case does not mean mediocre, it means not more quality than is necessary. For example, this article is not being typed on a top-of-the-line laptop, it's a PC that's 5 years old.

To know if you're "satisficing," or if you've done enough of "anything," all four of these conditions must be met:

- Your investigation has provided sufficient benefits to the stakeholders.
- Your investigation does not have critical problems.
- The benefits of your investigation outweigh any problems it might have created.
- Further progress in the investigation would likely be more expensive or time-consuming than helpful.

If any ONE of these items is false, then, by definition, your investigation is not good enough.

So, I can apply that model here with this article. Should I continue with the exercise, or have I done enough?

Let me leave you with this video link (<http://www.quardev.com/articles/video-from-james-bach-investigation>), created by my brother James, who narrates his actions during the same investigation to Grigori Melnik from the University of Calgary. You can also use a search engine to learn more about the problem – just type in "this app can break".



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



September 19, 2007, Triangle Park, NC
TISQA

Jon Bach will Keynote and present a Track Talk for this conference in Triangle Park, North Carolina.

See the conference Web site for more information: <http://tisqa.org/conference/>.

October 8-10, 2007, Portland, OR

PNSQC

The Pacific Northwest Software Quality Conference. Registration is now open.

See the conference Web site for more information: www.PNSQC.org.

October 22-26, 2007, Anaheim, CA

STARWEST

This conference put on by SQE is a premier conference on the West coast. Jon Bach will present an all-day tutorial and a track talk.

See the conference Web site for more information: www.sqe.com/starwest/.

*Wednesday, November 14, 2007,
Seattle, WA*

November WSA QA Workgroup

Details to come.

See more information on the QA Workgroup Web site: <http://www.QASIG.org> or visit <http://www.Quardev.com> for regular event updates