Reporting Skills and Software Testing

Karen N. Johnson
Software Test Management Inc.
karen@karennjohnson.com

ABSTRACT

Some of the very skills journalists learn are skills that can be helpful to software testers. This paper focuses on those skills, and how software testers can learn and apply those skills to our work in software testing.

Interviewing. Software testers need to gather information; we need to be able to interview our team mates and subject matter experts to learn more about the products we're testing. We need to know how to build a rapport with different people and how to interview people in order to gather information.

Investigation. Testers need a sense exploration. We learn technologies and software products at a fairly rapid rate. What investigation and exploration skills can we learn from journalists?

Notetaking: Journalists are trained to take notes. Testers can benefit from taking good notes before, during and after testing. How can testers become adept at recording information?

Reporting: We test, we learn. And we gain opinions. How do we remain objective? Do we try to be objective? Do we report project status objectively or with our opinion? Do we know when we've developed a bias?

I graduated college with a major in journalism and worked as a reporter for two years. I've also been in software testing for more than two decades and it is based on this background and experiences that I've written this paper.

1. Interview Skills

Journalism students learn about asking questions. They start with the five W's: who, what, where, when and why. They get the facts, they get the basics and they drill in to learn more.

Journalists learn how to ask questions that best elicit information. They learn how to ask questions that are open-ended to encourage the person being interviewed to tell more, to elaborate. They learn to watch for the other person's reactions and to continue down the same line of questioning and they also learn when to shift questions and move the conversation along or down another path.

Whenever possible, journalists prepare their questions and learn about the subject or the person before investigating a story. Software testers often have limited time with the subject matter experts they need to interview – and like journalists, testers should try to prepare their questions in advance.

Despite the best of intentions (and sometimes very good execution), all the information testers need is not always written down. And as we learn about the software we're working with, we have more questions.

Learning skills is not just about the mechanics of the skill. In addition to the concrete skill, we also need to learn the finesse of using skills successfully. Asking people questions and continually quizzing someone for information can be off-putting. We need to learn how to make the other person comfortable and to blend questions into a conversation to collect the information we need.

1.1 Establish a Connection

In software testing, testers need to learn about the product, the customer, service level agreements and required or surrounding regulations. We gather a significant amount of information by asking questions and talking with software developers, system architects and database administrators. We talk to subject matter experts. We need to be able to talk with a range of people to gather the information. We need to become skilled at establishing a rapport with people quickly and in such a way that we're able to interview people to gain information. We might not realize that many of our conversations with subject matter experts are interviews.

To have and host a "successful" interview – one where you gather the information you need, then you need to build a rapport with the other person. Everyone is unique. So how you build that rapport is doesn't follow a template or a defined routine. Some people take time to warm up. Some people never warm up. Some people are not good at communicating. And some people withhold information.

1.2 Don't put people on the spot

I was shy when I was younger and starting out as a reporter, but I

I learned to be less self conscious. I learned techniques to help shift the focus from me to the other person and even better if I could shift the energy to the topic at hand and leave the person I was interviewing in a dynamic such that they didn't feel on the spot or self conscious either.

Asking questions can put our subject matter experts in a situation where they *feel* like the center of attention and that can be uncomfortable. When people are uncomfortable they frequently talk less, and we may have to work harder to get information. The result can be short stunted interviews where we don't gather the information we need.

There are tactical ways to let the subject, let the material take the center stage and avoid having your subject matter experts feeling like they're being interrogated. To help keep the focus on the subject and not on the expert, I often use whiteboards or diagrams, as the expert and I focus on the diagram and recording product details accurately, the other person may feel less on the spotlight and may be more forthcoming with information.

We might also experience working with people who enjoy being in the spotlight and may "take over" a conversation. So we also need to learn how to pull a conversation back on track. We need to know when to be the driver in a conversation and keep a conversation going entirely off course.

When do you need these almost opposite skills? It takes time and it takes practice. Learning to host a successful interview (successful begin defined as an interview where you get the information you were seeking – provided the person has the information) takes experience. There is not shortcut to gaining experience however like other skills, when you are conscience of the skill you are trying to improve upon and consciously apply techniques, you can become more skilled more rapidly. This is what is meant by intentional practice. We become aware of the different skills that we need in our work and we practice.

1.3 Let Dialogs Roam

In the book, "The Art of the Interview" the author Lawrence Grobel adjusts his interviews based on the person and how the conversation is going. As an "expert" at interviews, he does his homework before talking to the person – in software testing this might mean: read the product specifications, read the online help, take the time to experience and explore the software and any other legwork you can do.

Do your homework. Be ready. Be prepared. Let the person see that you have done your research. Have your questions ready. Avoid driving the conversation like an interrogation.

Ask open ended questions. Avoid a staccato push and pull exchange. An exchange like the famous teenage and parent chat:

```
Parent: Where did you go?

Teenager: Out.

Parent: What did you do?

Teenager: Nothing.
```

In the "Art of the Interview" the author talks about his interview with one celebrity, Drew Barrymore. He commented that he could see the subject was getting too difficult, too hard for her to continue in that topic so he shifted the dialog with a different question. You might have a list of questions and for a variety of reasons find that you can continue asking questions in a linear path. You might have to jump around on the road map you had planned but it doesn't mean you won't arrive at the destination.

1.4 Tune into the Conversation Tempo

Conversations have a tempo. There is a speed of dialog, there is a speed of one person talks, the other person talks, we take turns and those turns can be high speed or slow. Conversations and more particularly interviews move along better when the tempo of the dialog is harmonious between the people talking.

If you want to keep the ebb and flow rolling in a conversation, tune into the cadence of the dialog and make sure you're in rhythm with the other person. As you get to know the people you work with or you pay more attention to your own interviewing skills, you learn how to adjust more quickly and tactfully.

While content matters more than tempo, you might find you can improve your conversations and interviews just a bit if you can subtly clue into the tempo and adjust as needed. When people feel rushed, they are more likely to skip past information.

2. Learn How to Investigate

Journalists are investigators. Journalists have opportunities to work like a detective. They are taught to be prepared. Have a notebook, have a pen, have a recorder and have a camera – essentially, have what you need. The reality for a journalist or a reporter, they often don't know when or where a story will break so they learn to keep their tools on hand.

Testers also need to be ready; we don't know when we will find an issue or when we might luck into a conversation or opportunity to talk with a subject matter expert, so we too have to have our tools on the ready.

Testing software is similar to investigative reporting, we don't know what we're going to find so it may be impossible to tell a tester how to prepare. Sometimes the best preparation is to have our tools on hand and to go forth. Investigating often means building a path, not following one.

One of the most important skills for a tester is to be curious. Be hungry to learn, hungry to explore and don't expect a path to be laid out for you. Maintain a childlike curiosity and hunger to learn. What does it mean to have a childlike curiosity? It's a combination of nonstop energy and interest, an innate desire to learn and explore.

See investigation as the adventure. Sometimes I'm eager to arrive in a certain state or condition with the software I'm testing and then a defect occurs and I am caught off guard – I'm forever being reminded: stay alert. Don't be impatient for the results.

One of the difficulties in trying to teach a journalist or a tester how to investigate is we're not even sure what, who or when we will be investigating. Our sources and resources "depend" on the story or the project, we learn to adapt. Learn how to change directions quickly because investigating and testing is not clearly laid out. And even when we map out plans, our plans may need to shift.

It helps if you can see alternatives paths and alternative ways to learn and explore. What is your learning style? Do you jump in and explore with software? Do you read and research? Do you use

a mixture of both? If you can't use your preferred method of learning or working, can you find another approach?

3. Take Notes

Journalists are taught to take notes. We're taught not trust our memory. Write and be ready to write. Being ready to write means having some way to record information with you and with you at all times. In the newspaper room that I worked in – we were trained to have a notebook and pen with us at all times and if possible and feasible, a camera and voice recorder if that made sense for our assignment. For me, I had a camera frequently.

In software testing, a reporter's notebook might be replaced with a project notebook. And more specifically, many testers have come to carrying and using a Moleskine notebook. And in a pinch, your phone might include a built in voice recorder. We have other tools too, such as video recording to track our steps and interactions with software.

I've found software developers don't appreciate having to repeat information so making the most of a session together includes thinking fast on your feet, asking questions, listening and taking good notes. What are "good" notes? Minimally, I would define "good" as - sufficient notes for you to remember the information.

In the most blunt, direct words short of photocopying pages from a project notebook, I blogged in 2008 about how I record notes, what I record in my project notebooks and my own personal short codes that help me record information as rapidly as I can.

There are situations where I find that note-taking while I'm talking to someone can be distracting. In many cases, I leave my notebook open widely on a table so that people can see what I'm recording and to avoid creating any angst about what's being captured. As a journalist, some people would ask to see my notes and it was standard for me to confirm a quote from my notes.

3.1 Consider the Source

Journalists are trained to: consider the source. An expression that's used often as a reminder for a journalist to consider whether the person they are interviewing has a bias or background that could impact the information being share. This expression is meant as a reminder to have a discerning ear and a reminder to verify facts.

It might seem the quote as little to do with software testing but when we consider that much of our information comes from other people and those people are frequently the designers and developers of the software we're testing – it is a reasonable and a prudent quote to keep in mind – consider the source.

Nearly everyone has an opinion and certainly the creator of software is going to have an opinion as well as you talk over design and potential flaws. It doesn't mean their opinion isn't highly valuable, helpful and holds a measure of accuracy and truth but it does mean there could be a bias and keeping a critical ear open as you listen can be helpful.

In addition to being mindful of bias, software testers need to challenge assumptions and beliefs of the software. Just because a product requirement states that the product is supposed to do something, doesn't mean that the software actually does that. We need to prove information.

3.2 Collect and Organize Artifacts

Between investigating, interviewing and exploring, reading and researching, we may collect a lot of information. Develop a way to collect information and to be organized. You may find information you gathered early on, has a value later as your knowledge increased. Can you find your notes from earlier in a project?

After years of capturing lots of notes, I found I needed a way to organize materials. So I've created my own organization style that works for me. Client notebooks are color-coded, usually I color code the files and folders I have as well. I use the same colors to block time and meetings on my calendar. Files, folders and color coding work for me. For me, I have my own way of organizing my materials. The organization helps especially when projects get busy.

4. Cope with Deadlines

Tight demanding deadlines are a reality for journalists. The same is true for software testers. As a journalist when I wrote articles for a daily paper, my deadline was 2am to the editor. The expectation was that I would then work with the editor until the copy was good to go – regardless of how late it became although the editors were also working with compressed time schedules. Software testing deadlines are often tight and tension filled as well.

When I began working with software, I found my tight deadlines didn't go away. In fact, I frequently was working under tight timeframes – so much so that I quickly began to believe that coping with deadlines is yet another key skill for a software tester.

5. Reporting Information

During and after a software release, testers are often the people on the team to report the status of the product. We are often reporters of product status. In journalism, we are trained to be mindful of fact vs. opinion and not to blur the two. And if we do blur the two – fact and opinion – we should at least be aware that we are doing so.

Defects need to be stated clearly and accurately without bias. Our project and product status reports also need to be accurate, free of bias or if our opinion is being sought, at least being mindful of what is a fact and what is an opinion can keep our writing clear and focused.

As you report information, remember that other people will be considering the source as well. If you want to become a trusted

source for information, you have to be able to report the facts, just like journalists do.

6. References

1. "The Art of the Interview", Lawrence Grobel
2. My blog on: Notetaking taking while testing software
   http://www.testingreflections.com/node/view/6536

7. Author's Background

I majored in journalism and minored in English in college and graduated with a Bachelors of Science in Journalism. I worked as a reporter for two years starting by taking any odd assignment I could get my hands on and eventually graduating to better and more desirable assignments. But newspapers pay poorly and being in the classic post college shoes of needing money to buy a car, getting married and wanting to buy a condo – it was time to strike out to make more money. The best paying work I could find was working as a marketing writer for a small software company, a company that would later went bankrupt and gave me a real world education of the software world. After a year of pumping out press releases, the company I worked at had an opening for a tech writer, a job title I thought sounded cool. I bucked up for and got the job. I took to the job quickly and feel for software. I was hooked.

I didn't have the background and I didn't have the academic insight into how software works or how things in general fundamentally are built or structured. But what I did have was a handful of skills that translated readily in my new job. I could learn fast and I could write well. I was alone for hours, in fact days and even months at a time to learn the software and write online help systems or quick start guides. Left alone to explore and left even more alone to write, it was a perfect job for me and I remained a tech writer for seven years. I came to love software finding it infinitely fascinating.

I found skills acquired from journalism and reporting readily translated in my work in software testing.