

Tea-time with Testers

OCTOBER 2012 | YEAR 2 ISSUE IX

Jerry Weinberg
The Fish-eye Lens

Ben Kelly
The Testing Dead

Jerry Weinberg and Michael Bolton
Let's Talk Testing

T Ashok
'Great Expectations'- Excellence Requires Empathy

Joel Montvelisky
Ask yourself what were you hired to do?

Exclusive : Over a Cup of Tea with Keith Klain

There are more than 40 leading
organisations that host
Tea-time with Testers
on their knowledge portal.

WHAT ABOUT YOURS?

TEA-TIME WITH TESTERS

Subscribe [here](#) Right Away to get our all Issues for FREE



TEA-TIME WITH TESTERS

First Indian Testing Magazine to reach 97 Countries in the world !

Created and Published by:

Tea-time with Testers.
Hiranandani, Powai,
Mumbai -400076
Maharashtra, India.

Editorial and Advertising Enquiries:

Email: editor@teatimewithtesters.com
Pratik: (+91) 9819013139
Lalit: (+91) 9960556841

This ezine is edited, designed and published by
Tea-time with Testers.

No part of this magazine may be reproduced,
transmitted, distributed or copied without prior written
permission of original authors of respective articles.

Opinions expressed in this ezine do not necessarily
reflect those of the editors of ***Tea-time with Testers.***

Guest Editorial

Want to become invaluable to your project?

Software testing is a funny business. We all want to be taken seriously, get involved in strategic decisions, and not have our jobs turned into commodities. So what is our typical response? We offer maturity models, metrics and processes that distract projects from evaluating their products, move us further away from risk management and expedite the process of devaluing our jobs as testers! Software testing is critical to managing risk for any enterprise. Whether that risk is related to the market, financial exposure, or gaining (and hopefully keeping) a competitive edge, your job as a software tester is to gather information to help your team manage that risk. That's an incredibly important job! So if you are in a position that is diminishing in value or your team is on the slippery slope to commoditization, here are some suggestions to snap things back on track.

Understand and Align Yourself to the Business You Support

I know this sounds simple, but believe me, this is one of the most common shortcomings I find in software testers. Technology is a big part of what we do, but technology supports a business –and people make up that business! If you are not conversant in the business and people your technology supports, you are going to have a difficult time advocating on their behalf. A great way to stay relevant to your organization is to understand their strategic objectives and be able to articulate how your testing strategy is aligned to – and helping achieve those objectives. It may be difficult to believe, but I've reviewed hundreds of projects, and it consistently amazes me when people cannot relate their testing strategy to business objectives. People are far less likely to devalue something that is moving the ball forward for their business strategy.

Learn the Language of “Value” in Your Project

So after you've aligned yourself and your test strategy to your business objectives, you need to be able to convey that alignment in way that your project understands. People very often make decisions that are driven by bias – both positive and negative bias. A good software tester is constantly trying to jar themselves out of their own bias to be able to empathize with other viewpoints and accommodate them in your test approach. Understanding value to your client can be a singular thing, but how that value is achieved, measured, and validated will have a “one to many” relationship with as many different people there are in your team! Understanding how your project manager, developers, BA's, business partners all define and articulate that value is integral to being successful as a tester. Learn the language of your project team and you won't be ignored!

Simplicity is the Essence of Perfection

Technology and the interpersonal relationships need to create software are incredibly complex by their very nature. Add into that the noise created by life in the 21st century – social media, internet, etc. and you have a powerful combination of short attention spans coupled with endless sources of distraction. In order to rise above the din, a good software testing must distill their message down to the simplest of terms in order to quickly ascertain risk probability and any required actions. Over the course of my career, I find testers having trouble getting to the point in their communication largely because the testing industry, their employer, and sometimes the development community, have fed an inferiority complex that testers are second class citizens. This gives us a tendency to try to validate ourselves by over complicating issues and providing far too much context instead of directly addressing issues. When speaking to people about test results or your approach, always remember this quote by Colin Powell – “Great leaders are almost always great simplifiers, who can cut through argument, debate and doubt, to offer a solution everybody can understand.”

Don't Wear Your Heart on Your Sleeve

Software testing is a deconstructive process that lives in a largely constructive lifecycle, and often times, it's the only attempt to critically review plans and designs for potential failures. By its very nature, the craft of testing is going to bring to light risks, issues, and the potentially negative impact of conscience decisions people have made about how to build their products. Guess what - that information might possible upset some people! So what! You cannot take their reactions personally, or you will not be able to effectively do your job as a tester. If you allow yourself to connect information you have learned about your products to who you are as a person, you are less likely to be able to objectively assess and report on risk. Don't take it personally! Software projects (and life) are filled with countless variables that constantly change and impact our ability to reason and decide on what to do, so don't take the bait and make it about yourself!

Finally, I want to say thanks to the folks at Tea-time with Testers for giving me the opportunity to this Guest Editorial. They are making a tremendous contribution to the software testing community, and I wish them all the best for the future!

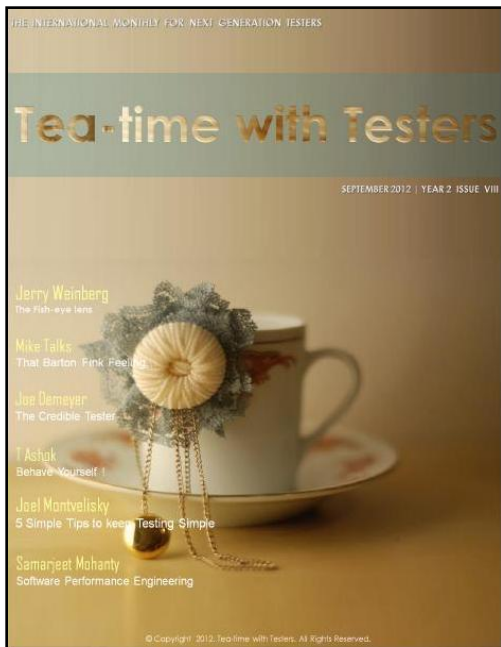


I am sure that you'll enjoy this issue of TTWT, as always.

Thanks!

A handwritten signature in black ink, appearing to be 'Kl' followed by a stylized flourish.

Keith Klain



September 2012

I've been reading over the September issue of Tea-time with Testers.

Here's some random thoughts inspired by the issue:

I keep noticing the expression "when the test failed" when the writer really wants to say, "when the test succeeded in turning up an issues." The test, in fact, succeeded in doing exactly what it was designed to do. Language is important, and shapes thinking.

Joe DeMeyer's article on The Credible Tester is great, full of specific suggestions on how to enter a team as a respected leader. I would warn the readers, however, about the effect to asking too many questions. Question-asking can easily be seen as controlling behavior, which alienates many people. So can suggestions. In addition to Joe's marvelous questions, make sure you encourage people to ask you questions--about anything they want to know. And encourage them to suggest things to you--and show that you're listening to their suggestions and taking them seriously.

You won't become a real member of the team unless these interactions can be symmetrical, so nobody puts themselves above the other members of the team.

Joel Montvelisky's 5 keys to simplifying the testing task is so completely right on that I can add only one suggestion:

Print out a large poster listing Joel's 5 points, and post copies at any place in your environment where you discuss your work. Then use them as a checklist for any proposed action. Take a few minutes as you prepare to finalize a decision to check:

- Have we broken the task into small, manageable pieces?
- Have we addressed all the important aspects? (You might make a list of these.)
- Have we categorized this task based on importance, complexity, and last start date?
- Have we said NO when NO is the right thing to say?
- Have we made everything about this work visible to all?

Keep doing these things, and they will become part of your culture.

T. Ashok has once again provided wise words to follow, to which I would add only one significant point. Behavior rules should not simply describe what a system must do, but also describe what the system must NOT do.

I really like **Mike Talks's** essay. As usual, he's right in choosing an important and neglected topic. As Dani Weinberg is one of the world's best professional dog trainers, I discussed his analogy with her. It's a powerful analogy, and Mike's right as far as he goes--but there's more to dog training that can teach us about training experienced professionals. For instance, one of the strongest principles of successful training is not "learning from failure," but "learning from success."

That's essentially what Mike is talking about when he suggests that you might want to start by doing some of the things these experienced trainers already do successfully. You then want to design iteratively a series of small steps, each of which is pretty much guaranteed to be successful, as seen by the participants.

All in all, a super issue. Keep up the fine work.

- **Gerald M. Weinberg**

We will miss you Ola !

Ola Hylten, a known personality in testing community as entrepreneur with Let's Test conference passed away this October. He was known for his passion towards testing.

We will always miss him for his contribution to the community and TTWt.

With Ola's demise, our community has lost one passionate tester. We pray for his soul and take inspiration from his contribution to the community.



QuickLook



Testing Puzzles
by Sebi



Crossword
by



Editorial

What's making News?

Tea & Testing with Jerry Weinberg

Speaking Tester's Mind

The Zombie Menagerie - 17

In the School of Testing

Let's Talk Testing - 25

Ask yourself what were you hired to do? -35

T' Talks

"Great Expectations" - Excellence requires empathy - 39

Over a cup of Tea with Keith Klain

Testing Puzzle – S.T.O.M. Contest

Our Testimonials

Family de Tea-time with Testers



What's making News?

- find out the latest happenings in the technology world

Quality Testing crosses milestone of 10,000 Registered Members !

Tuesday, November 1st, 2012:

Quality Testing (www.qualitytesting.info), a leading online-social network for software testers has crossed a big milestone of reaching 10,000+ registered members.

With 10,541 registered users, Quality Testing has now become one of the largest online communities of software test professionals. With such huge user-base, Quality Testing has also become largest testing community of Asia.

Quality Testing's motive is to benefit the testing community by providing an open platform for knowledge sharing and networking between Software Testing Professionals worldwide. Quality Testing has attracted many enthusiastic test professionals from 153 countries with major registrations done from India and USA.

More than 3800 professionals visit Quality Testing every day. This community is currently accessible to members as well as non-members. Quality Testing considers its user base and forums (around 8500 discussions) as its valuable assets.

Quality Testing also conducts online training courses (Quality Learning) and runs a jobs board (Quality Jobs Portal). That's not just it. You will find content services, hundreds of technical videos, discussions, featured interviews, information on testing tools, details of software testing events from across the world, a weekly news letter, blogs, and specially created groups on various subject areas in software testing in this terrific community.



Speaking on the occasion, **Mr. Kiran Kumar** (Founder and Director of Quality Testing) said, "We are happy that we have reached a one milestone. This is largest non-commercial and non-profit social network for software testing professionals across the world. Reaching this milestone has made us more responsible and we are committed towards making Quality Testing a most definitive platform for social networking between testing professionals."

Kiran can be contacted on kiran@qualitytesting.info or on twitter [@chkirankumar](https://twitter.com/chkirankumar)

Tea-time with Testers is in strategic partnership with **Quality Testing** and we congratulate them for this candid achievement.

Are you interested in publishing the news about your own firm, tools, community and conferences in **Tea-time with Testers**?

Then write to us at:

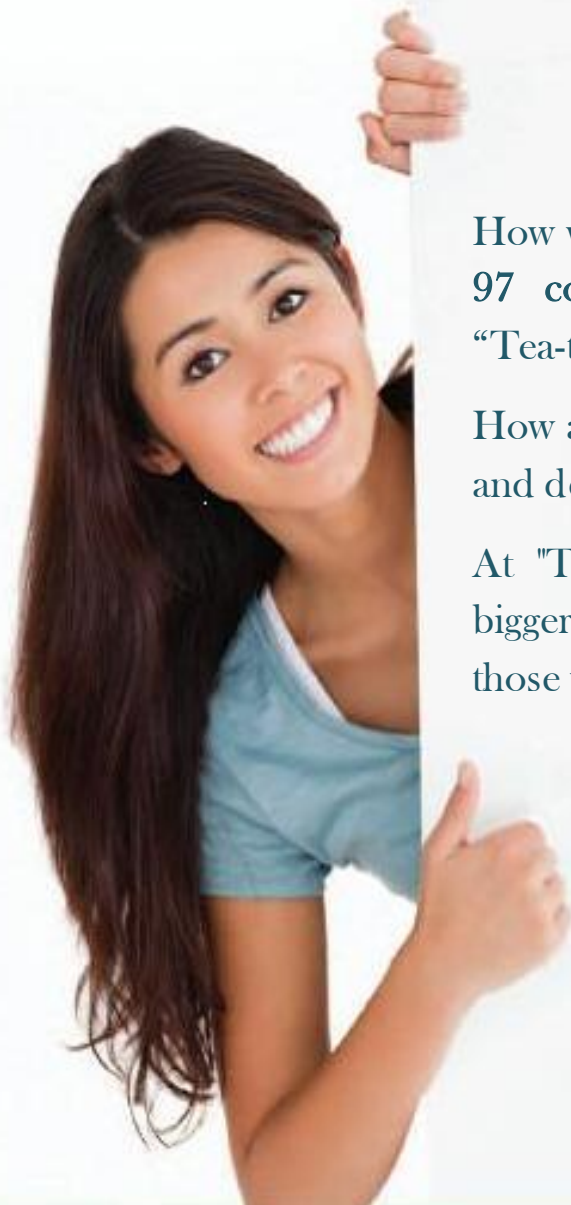
contact@teatimewithtesters.com

with "**News Enquiry**"* in your subject line.



*Conditions Apply

Want to connect with right audience?



How would you like to reach over **19,000** test professionals across **97 countries** in the world that read and religiously follow "Tea-time with Testers"?

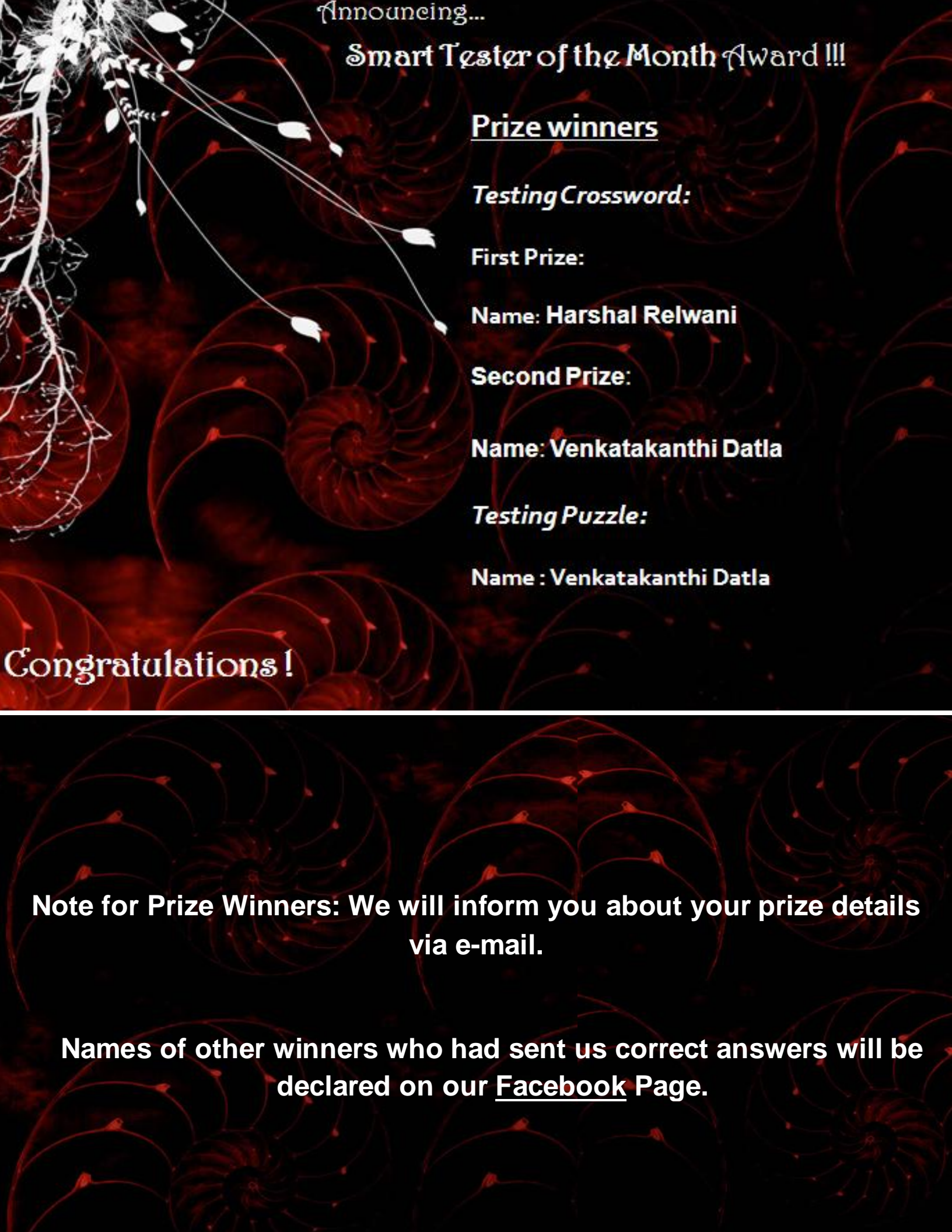
How about reaching industry thought leaders, intelligent managers and decision makers of organizations?

At "Tea-time with Testers", we're all about making the circle bigger, so get in touch with us to see how you can get in touch with those who matter to you!

ADVERTISE WITH US

To know about our unique offerings and detailed media kit

write to us at sales@teatimewithtesters.com



Announcing...

Smart Tester of the Month Award !!!

Prize winners

Testing Crossword:

First Prize:

Name: Harshal Relwani

Second Prize:

Name: Venkatakanthi Datla

Testing Puzzle:

Name : Venkatakanthi Datla

Congratulations!

Note for Prize Winners: We will inform you about your prize details via e-mail.

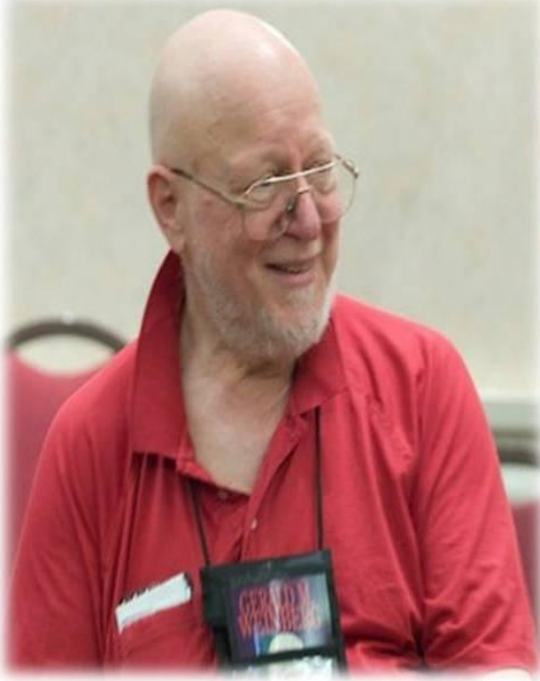
Names of other winners who had sent us correct answers will be declared on our Facebook Page.

Don't miss the action...



join us on FACEBOOK !

Tea & Testing



with

Jerry Weinberg

The Fish-Eye Lens (Part 5)

The Crooked Channel Cleanser

The Fish-Eye Lens is a metaphor for the channels through which I perceive the context in which I'm working, but whether they are real or metaphorical, lenses must be kept clean. Let me give a funny example.

Up at our cabin in Pecos, we're connected to the world by one thin telephone line through the mountains. A few years ago, we had a peculiar problem in which our phone kept dialing wrong numbers. We have several extensions, and all did the same thing, so I figured it wasn't the handset, but something further down the line.

I tried to call repair (114), but because the phone was misdialing, I kept getting directory assistance (113). I tried to tell the operator that my phone was misdialing, and she told me I'd have to call 114 for repairs. After several cycles like this, I gave up and drove the Jeep down the mountain ten miles to the phone company where I could describe my problem face-to-face to Drew, the repair man who often came to our house when there was trouble.

Drew, as usual, was friendly and helpful. But he was puzzled. "Why didn't you just call me on the repair number?" he asked. "You could have saved yourself the trip." He couldn't seem to grasp the concept that the communication system itself was adding error to the situation.

I wasn't really surprised by Drew's reactions, because when something like this happens (often unnoticed) in personal interactions, people rarely grasp what is going on. I call this kind of mystery a Crooked Channel Conundrum.

It's cleared up by applying the Crooked Channel Cleanser:

When you're having trouble understanding what you're receiving, first check that your channel is congruent.

In other words, I try to keep my Fish-Eye Lens polished, or I won't know if I'm seeing the world or merely seeing the distortions of the Lens. If I am straight with people, I can debug my crooked interactions with them—much like providing a constant environment for debugging software, but where I am the test environment.

In off-line computer systems, it's easier to create a consistent, clean test environment. Similarly, in human-to-human communications, consistent clear communication is easier in written letters, where you have time to think things through and remove any incongruence. The trade-off, of course, is that if there is incongruence, you don't have the quick feedback to correct it.

In e-mail, I get faster feedback, but I may be going too fast, and on-line, or face-to-face, I have a much harder time keeping some crookedness from creeping in without my noticing. So, I try to be as congruent as I can in order to be sure that what's coming to me is not my own incongruence distorting what was really out there.

And that's why I need my Gyroscope.

[Back To Index](#)



Experiential Learning: Beginning

Gerald M Weinberg

"Telling is not Teaching"

Read Jerry's new series, 'Experiential Learning' !

Do not forget to check out -

[Volume 1: Beginning](#)

[Volume 2: Inventing](#) is now available.

[Volume 3: Simulation](#) is also available.

When you buy this book, you get it in PDF, EPUB and MOBI formats, so you can read it on your computer, iPad, Kindle or other ebook reader!

If you buy the book, you get Jerry's all the Leanpub updates to the book for free!

Biography

Gerald Marvin (Jerry) Weinberg is an American computer scientist, author and teacher of the psychology and anthropology of computer software development.



For more than 50 years, he has worked on transforming software organizations. He is author or co-author of many articles and books, including *The Psychology of Computer Programming*. His books cover all phases of the software life-cycle. They include *Exploring Requirements*, *Rethinking Systems Analysis and Design*, *The Handbook of Walkthroughs*, *Design*.

In 1993 he was the Winner of the **J.-D. Warnier Prize for Excellence** in Information Sciences, the 2000 Winner of **The Stevens Award** for Contributions to Software Engineering, and the 2010 **Software Test Professionals first annual Luminary Award**.

To know more about Gerald and his work, please visit his Official Website [here](#).

Gerald can be reached at hardpretzel@earthlink.net or on twitter [@JerryWeinberg](#)

More Secrets of Consulting is another book by Jerry after his world famous book **Secrets of Consulting**.

This book throws light on many aspects, ways and tools that consultant needs.

“Ultimately, what you will discover as you read this book is that the tools to use are an exceptionally well tuned common sense, a focus on street smarts, a little bit of technical knowledge, and a whole lot of discernment”, says **Michael Larsen**.

More Secrets is definitely useful not only to consultants but to anyone for building up his/her own character by implementation of the tools mentioned in day to day life.

Its sample can be read online [here](#).

To know more about Jerry's writing on software please click [here](#).

MORE SECRETS OF CONSULTING



Gerald M. Weinberg

TTWT Rating: ★★★★★

A photograph of a green, conical pendulum bob hanging from a thin wire. The bob is positioned over a surface of light-colored sand. In the sand, there is a large, circular, swirling pattern that resembles a fingerprint or a stylized mandala. The text "Speaking Tester's Mind" is overlaid on the image in a large, blue, serif font with a white outline.

Speaking Tester's Mind

- straight from the author's desk



The Testing Dead

episode-1

The Zombie Menagerie

- A series by Ben Kelly

The zombie apocalypse has occurred. They walk among us even now – The Testing Dead. These dead-eyed, soulless creatures make sounds that seem human, but they're an empty shell inside and will bite you if provoked. Left unchecked, Zombie Testers will infect an organization with their disease. Zombie testing is any rote application of testing practice or methodology without regard for how appropriate it is in that context. It often looks like one or more 'skilled' testers churning out test cases for meatbot automations to execute, but there are no doubt those who identify with context driven methodologies who have missed the point and follow the same go-to patterns regardless of context.

While there is some amount of tongue-in-cheek in this analogy, it does describe actual patterns of dysfunction that I've observed. I want to be clear at this point that I'm having a go at a kind of behavior. I'm not trying to demonize people.

There are a number of different flavors of Zombie. See if you recognize any of them.

The Misled

These are the ones who finished whatever secondary or tertiary education they did and decided they were done learning for the rest of their life and could they please have a job where they could memorize and regurgitate the right answers like they did in school. Not particularly adventurous, they might have found some of the large amounts of crap online about software testing and decided that was just fine thanks.

Give me a recipe to follow or a template to fill out, but the dark gods forbid I should have to think for myself.



The Template Weenie

A variant of the Misled. They discovered some testing templates online or perhaps their company had some already put together. Their belief is that if they fill out these templates and no gaps are left, then good testing will have been done. If they've got all the requirements covered and tests all trace back to them, and the test plan is all filled out and the schedule is set properly, then we're all good. It appears that for them, reality is an obstacle to be managed with paperwork.

The Passenger



The passenger has fallen into testing but has no desire to be there. They have no desire to be a tester, but are using testing as a bridge to somewhere 'better' (typically programming, business analysis or project management). They tend to do only enough work to avoid reprimand and will often be found hanging around the group they're trying to break into as though they might be absorbed by osmosis.

I generally try and cut a deal with passengers should I encounter them. It is in our combined best interests to move them on, so I promise them I will do everything in my power to get them where they want to go if they agree to be the best tester they can be while they are with me. If they have a strong body of work I can show the manager of the group they want to transition to, and I can honestly talk about the strength of their efforts, then that tends to lend great credibility to their application. Sometimes that approach works, sometimes not. If they won't let you help them to get where they're going, you may wish to help them out the door instead.

The Apathetic

Similarly to the passenger, these zombies have no real desire to get better at testing, they simply want to turn up between 0900-1700, go home, rinse and repeat. They won't think about or do testing in their spare time, it is merely a job. To some degree there's nothing inherently wrong with this, but personally I'd rather work with inspired, passionate people that genuinely enjoy what they do and want to do it better.

In some respects having a few apathetic zombies around can make your life easier – they tend to be the ones who enjoy predictable monotony and there's often no shortage of that in testing. If you have repetitive work that is difficult to automate, these people can be handy.

The Confused



This lot thinks they're doing Quality Assurance when what they're doing is testing. Quality assurance is really a collection of roles and actions that have a direct bearing on the quality of the product, such as the hiring/firing of programmers, architects etc, Decision making about what to include or what to leave out, which design to go with, which vendor to go with and so on. In contrast, software testers reveal information about the product. Some testers write production code, but in their role as a tester, they do not directly influence the product itself. The Confused either do not get this, or vehemently believe that their role is to be the final bastion of quality before the software goes out into the world.

They tend to enjoy grandiose titles such as 'Quality Assurance Engineer' despite not doing quality assurance and not being an engineer. They also seem to actively position themselves as the gatekeepers of the software release decision, apparently blissfully unaware that it's a lose/lose situation for someone with the word 'quality' in their title to be attached to. If they say no to a release, they're either overridden by the people with real power (and who probably have a better business sense of what needs to occur), or they're seen as the ones holding everything up. If a release goes out and something screws up in production, they're the ones who get fingers pointed at them and asked questions like 'Why did you let that bug out?'

I've seen on testing forums questions like 'We had some bugs go to production. My manager is asking me why. Can someone give me some excuses I can tell them?' Wow, just wow. The level of non-comprehension about one's own job that this question requires is mind boggling.

The sadness doesn't end there though. Not only do the confused make their own life hard, but they like to make life harder for their non-testing peers also. Things like testing entry and exit criteria, based on arbitrary bug counts of varying severity (e.g. no more than 1 severity 1 bug and 5 severity 2) tend to make people's life unnecessarily difficult.

The Priest

A variant on the confused, these guys perform ritual testing.

It's testing theatre in much the same way that the TSA to airport security theatre. It may find some stuff, it may not. It gets applied to everything in the same way because that's how it has to be done. It's their religion. This is the way testing must be, for this is the one true way of testing. I'm not sure, but they may be an evolution of the template weenie.

Fortunately I haven't encountered too many of these.



The Horde

The horde probably resembles their traditional zombie counterpart more closely than any other zombie type. Although they are a large group, they share nothing more than proximity and brain death. A website (or app or other software) will be left out in the open like a sacrificial virgin. The horde descends upon this website under the guise of crowd-sourced testing whereupon individuals compete with the rest of the group in order to find something vaguely bug shaped upon the surface, like little zombie rhesus monkeys. They are paid by the bug, so when you take their bug away from them well, have you ever tried taking food away from a rhesus monkey? It's a bit like that.

They are largely incapable of following instructions unless said instructions are very, very precise. Instead, these zombies have specific go-to patterns they use to find bugs, such as turning on Internet Explorer's 'bitch about everything JavaScript related' mode. They will also report every single instance of the same bug despite the fact that they clearly have a single, common root cause. Their bug reports are somewhere between readable and atrocious because it doesn't matter what quality the bug report is, it just has to be first. Once they have exhausted their suite of patterns, they will immediately leave the victim, generally unmolested but quite free of lice or other surface irritants.

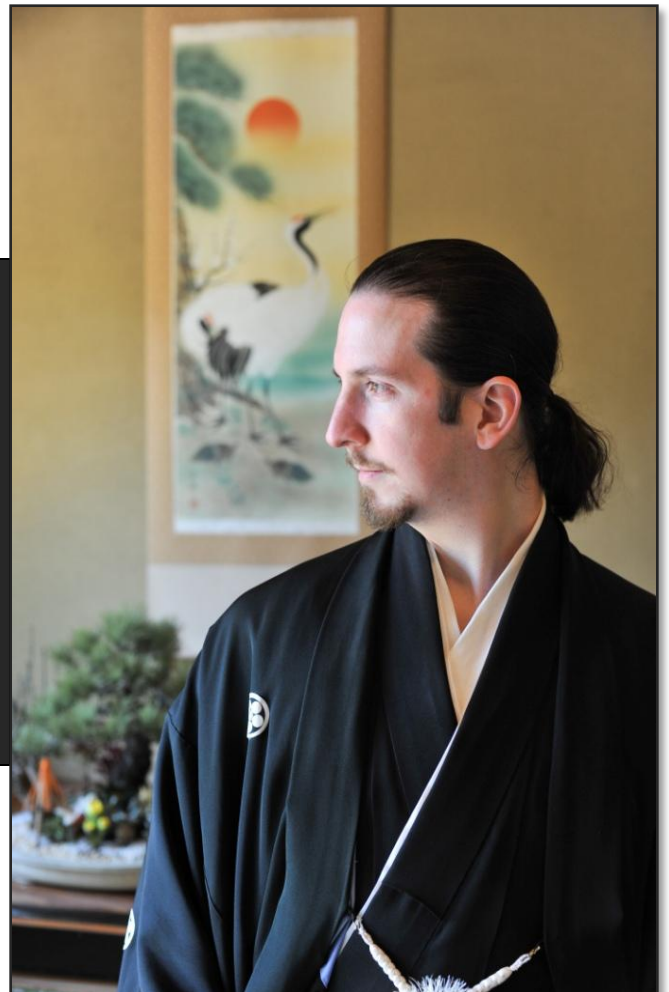
No doubt that there are more types of Zombie out there, but these are the ones I have encountered on my travels. There seems to be a common thread amongst zombie testers – the complete lack of desire to do anything differently to how they are doing it now. In a role that demands that we rapidly respond to a frequently changing environment that seems antithetical to how a tester should operate.

In the next article I'll talk about why zombie testing is a problem for thinking testers and what we can do about it.

Ben Kelly is a software tester living and working in Tokyo, Japan.

He has done stints in various industries including Internet statistics , insurance and most recently online language learning.

When he's not agitating lively discussion on other people's blogs, he writes sporadically at testjutsu.com and is available on twitter @benjaminkelly.



[Back To Index](#)

Looking for RIGHT job in
Software Testing?

visit **Qualityjobsportal.com**

after all, it's your career we are talking about !



Do **YOU** have **IT** in you what it takes to be **GOOD** Testing Coach?

We are looking for skilled **ONLINE TRAINERS** for Manual Testing, Database Testing and Automation Tools like Selenium, QTP, Loadrunner, Quality Center, JMeter and SoapUI.

TEA-TIME WITH TESTERS in association with **QUALITY LEARNING** is offering you this unique opportunity.

If you think that **YOU** are the **PLAYER** then send your profiles to trainers@qualitylearning.in.

Click [here](#) to know more

There was a time when people did not have compass to find right direction. The only guide they had was that guiding star up in the sky.

Do you think that you are also stuck somewhere with technical issues? Do you need help in decision making or want guidance?

Well, the wait is now over . Introducing...

“The Guiding Star”

*The panel of our experts is now here to help you.
Send us your questions around software testing and our Guiding Stars will help you out.*



E-mail your question on –

theguidingstar@teatimewithtesters.com

Please Note :

1. This is not a job portal.
2. Typical interview questions will not be answered.
3. Questions should be on Software Testing or related topics only.

Do you have any Questions or Feedback on articles that we publish in Tea-time with Testers?

No Problemo! We will publish your Feedback/Comments and also the answers to your Questions that you have for our Authors.

Do write us your Feedback and Questions in below format and send it to teatimewithtesters@gmail.com :

- Your Name
- Your Brief Introduction
- Article Name
- Your Feedback or Questions if any

➤ Your Feedback or Question

Make sure to write **Feedback For < Article Name>** in your subject line.



A photograph of several students in a classroom, seen from behind, with their hands raised in the air. They are facing a chalkboard that has some faint writing on it. The students are wearing colorful shirts: light blue, red, orange, and green. The entire image is framed by a thick black border.

In the school of Testing

for your better learning & sharing experience

Let's Talk



Testing

We believe that, discussion between two experts sometimes imparts *that* knowledge which one would hardly get otherwise.

In our **April 2012** issue, we had published the discussion between [Fiona Charles](#) and [Jerry Weinberg](#). In this issue we are publishing an interesting discussion between [Jerry Weinberg](#) and [Michael Bolton](#) that happened over e-mail.

This discussion is based on Michael Bolton's '**Why is Testing Taking so long?**' article that we published in our **July 2012** issue. It started with Jerry's feedback on this interesting article.

Enjoy this *Testy Talk* and don't forget to thank [Jerry](#) and [Michael](#) for allowing us to publish this.

- Editor

Jerry:

Michael Bolton's article is wonderfully clear. I hope managers read it and understand its implications for evaluating test teams. I have one quibble, however. When discussing Team C (the module with lots of bugs), Michael says, "Alternatively, we could stop testing now," but then dismisses the idea. I think that when a piece of code is this bad, the proper action is to discard it and start over. Our experience shows that code this bad stays this bad, or gets worse, after all the bugs are *fixed*. It's error-prone, and stays error-prone, so it's best to start fresh. (And break up Team C.)

Michael:

Thank you for your kind words, and your wish that managers read it. That's a wish that I share.

With respect to your quibble, my simple answer is that we (the testers) don't run the project. Testers provide service to the project, in the form of information about the product and the project, but testers don't run the project. I've been a program manager. Program managers must weigh all kinds of information, and make decisions based on all kinds of reasons: make technical, business, and personal reasons. Good testers contribute information to those decisions, and good managers value that information, but the testers don't make the decisions. As the program manager or the development manager, I might indeed decide to toss Team C or its code; truly bad code is often the cause of truly bad problems, and my experience is like yours. That is, I agree with you on the principle that the right thing to do would be to start over. However, practice is sometimes another matter. If I (the program managers) made a different decision and if a tester told me that I should do the *proper* thing, I *might* thank him for his opinion, but only if I trusted him AND only if I were in a good mood. Otherwise, I would suggest that he avoid telling me how to do my job, and to avoid presuming what is *proper* when they he doesn't have the information I have. Quality is multi-dimensional; sometimes some customers would prefer a product with lots of bugs to a product they can't have at all. Sometimes contracts or even-more-senior managers or the stock market require that we deliver something that we're entirely happy with. Program managers (product owners, project managers) make the project decisions; testers don't.

Now: If you choose to interpret *we* as *the project team*, then one could make the argument that we would make the decision. But I've never seen a project team that's truly organized on the consensus or democratic model. Managers are unwise, in my opinion, to ignore the information that the team provides to them, but in the end, some single person has had final responsibility for business decisions in every place that I've worked. I can imagine other situations, but I haven't been in one.

Jerry:

You will notice that I didn't say the *testers* should make the decision to discard the code, or to break up the team. I totally agree with Michael that these actions are not part of the testers' job.

I should have made clearer that I was talking about what the *project* should do. The testers can advise the management (whatever their structure) about their experience with this sort of poor code, but it is indeed not their decision to make. Even when the testers are part of a team running the project, it would never be their decision alone.

But knowing about error-prone dynamics, and informing the project team about their likely consequences, is (in my opinion) part of a professional tester's job.

Michael:

"But knowing about error-prone dynamics, and informing the project team about their likely consequences, is (in my opinion) part of a professional tester's job."

I agree with that, in principle; learning about risk is central to testing work, and informing is generally good. There are at least two items where I'd like to be precise about a statement like yours. The first is about *error-prone dynamics*: Information can be delivered in various ways, and in various tones. Talking about *error-prone dynamics* can be very risky, since errors are ultimately made by people. People make choices that are shaped by logic, by emotion, and by politics, and those choices involve making decisions about whose values matter. *Error-prone dynamics* can usually be found everywhere in the project and in the business, from coding to project management choices to the organization of the company's mailroom. Determining what information to give, when to give it, and to whom it should be given can be tricky. To address that, we must know our mission in the project and our relationship to our many clients. So again, information is good, and identifying risk is good, but caution, context, collaboration are important things to think about. In the statement above, my second item to be careful about is the idea of *likely* consequences. As the old saying goes, prediction is very difficult, especially when it's about the future. I'd probably say *plausible* or *possible* consequences, myself.

Based on those refinements, I believe (or at least hope) we agree. Thank you for writing.

Jerry:

I agree with everything except your changing the word *likely* to *plausible* or *possible*. In my experience, *likely* is the right word and pussyfooting with those other words invites dangerous denial. Yes, testers must be diplomatic, but even more important is that they be effective at communicating risks.

Michael:

I'm glad that you've mentioned experience as a factor. For example, measured solely by time spent in software development, you have about double the experience that I've had, so in many respects your notion of likelihood is better informed, more possible and plausible than mine would be (and so your argument in favour of *likelihood* is more plausible than mine is!). For someone with only a few years of experience in development or testing, *likelihood* is backed less by direct experience and direct observation, more by stories and models that might be missing important factors.

The specific context in which we choose *likely* vs. *plausible* is another factor: for truly lousy code, where we have evidence of it being lousy, both your experience and mine suggest we should toss it and start

again. It's usually infested with even more bugs we don't know about, it's usually a maintenance nightmare, and (worst of all) it usually reflects a poor understanding of some problem we're trying to solve with the product. In cases like that, I agree that it's safe to predict likely (and severe) consequences.

I agree that it's crucial for a tester to be aware of risks and to communicate them, but (as I learned from you more than from anyone else) it's also important to be modest and cautious about our ability to know what's happening now in the rest of the product, and to know what will happen later. Accurate predictions can help prevent trouble; inaccurate ones can compromise a tester's credibility.



Jerry Weinberg has been practicing, teaching, lecturing, consulting, coaching and writing about software programming and testing since the 1950s. With decades of experience and accumulated knowledge he's written more than 80 books and has dedicated his life to helping others be the best testers they can be – despite ever changing testing trends.

After reading his "[Testing The Limits](#)" interview with **uTest** , you'll find out the biggest lessons Jerry's learned over the years, how his books remain top sellers 20 years after their release and what the biggest issues testers are facing today.

To stay updated with Jerry's work , visit his [website](#) or follow him on [Twitter](#).



Michael Bolton has over 20 years of experience in the computer industry testing, developing, managing, and writing about software & has been teaching software testing and presenting at conferences around the world for nine years.

He is the co-author (with senior author James Bach) of Rapid Software Testing, a course that presents a methodology and mindset for testing software expertly in uncertain conditions and under extreme time pressure.

Michael can be reached through his Web site, <http://www.developsense.com>

[Back To Index](#) 

His key-note at **STAR East 2012** had become talk of testing-community.

With 'Bridging the Gap - Leading Change in a Community of Testers', he showed us how we can adapt some principles to lead ourselves and our team to a new and better place.

Meet **Keith Klain**, the Head of **Barclays Capital Global Test Center (GTC)**.

When Keith took over GTC, he implemented some changes to put a system in place to foster testing talent and drive out fear by abandoning worthless metrics and maturity programs, overhauling the training regime, and investing in a culture that rewards teamwork and innovation.

The challenges of these monumental changes required a new kind of leadership—something quite different from traditional management.

We talked with Keith to find out how he is leading this change. Know it directly from him in this exclusive interview with **Tea-time with Testers**.

Over a Cup of Tea with Keith Klain



Before we start, we would like to know about your 'testing journey'.

I got my first real introduction to software testing as discipline working as a test analyst for a company based in Chicago called Spherion Technology. They were one of the few companies to have a Software Quality Management practice with a specific methodology and training centred on testing. Although I disagree with most of the methodology now, what it instilled in me was that you could have a career in software testing, and I basically worked my way up from test analyst, to automation engineer, to test manager to eventually running testing practices as a director for them in London and the US.

I've pretty much worked in financial services my entire career focused on software testing, so I don't know a lot about other industries. I do believe that my focus on the people side of technology has allowed me to have success in management and change programmes where other people have failed. And as well, I am a voracious self educator, which I think is essential to continual learning and adapting to change. I'm less interested in accomplishments or the next milestone, and have never really focused on those aspects of my job, but I will say running the Global Test Centre (GTC) has been the greatest and most rewarding role of my career.

If we ask you to answer 'Software Testing for you is?' in one line, what would your answer be?

Software testing is the essence of risk management.

You are proven leader with both strategic and tactical abilities in Test Process Improvement, Quality Assurance and managing remote test teams. Please tell us more about your leadership stories, your success mantra.

My personal belief is that leadership has to be grounded by principals that people can identify with and then mirror in their own behaviour. To me, honesty, integrity and accountability are essential to building a team and as well, they are what I look for in people in leadership positions. You can manage people regardless of your principals, but if they don't believe what you are saying – that you have honesty and integrity in how you deal with them, you won't get the kind of performance and willingness to change that we are seeing in the GTC. Also, it's not very inspiring to work for leaders who don't mimic the attitudes they want out of their teams!

What made you to bring a new model in GTC? It's quite uncommon, we would rather say, first initiative of its own kind. Could you throw light on key changes that you brought in and how did those changes help?

When I joined Barclays in 2010, the GTC was part of a very structured "factory model" development centre based in Singapore. There were spreadsheets filled with metrics and KPI's, and the organisation was striving for a manufacturing approach to software development. That model was pursued to such an extent that the business functions were just called "units" with no association to the business they supported!

Clearly we needed to fundamentally change the approach from not only an organisational perspective but operational as well, so the first thing we did was throw out all those useless and distracting metrics scorecards. The immediate effect that had was changing the focus of the test teams away from trying to measure every aspect of their jobs and improve numbers.

The second big change was to fully adopt and start implementing the **Context Driven Testing (CDT)** approach to testing through training and working with **James Bach** and his **Rapid Software Testing** methodology. This change is a bit longer in its implementation as it's not just about training and implementation, but equally about a paradigm shift for the project teams (as well as the testers) as to the purpose and value of software testing.

Well, when one tries to bring change, a lot of things are required to be changed starting from the processes to be followed till career building roadmap. What is the mantra to do this from a mid-management and lower level?

I would sum up our change mantra as: "Manage Your Own Expectations". I'm a big believer in personal empowerment for the "Mid-Management and lower level", and don't really feel there should be separation of responsibilities for change in any part of an organisation.

My view is that it is your own personal responsibility to realise the changes you want to see happen. Understand the value of the change and take ownership of getting things done and you'll start to see positive improvements.

How should one manage the changes during transition periods to make sure that he/she doesn't change or remove something which is essential?

What I've found effective is to catalogue everything that a project or programme does, then identify what the project or programme "needs", compare the two and then start asking why. The majority of times, people will have not put much thought into why they do things and as well, "group think" is a very powerful force in projects. From my perspective, if it's not regulatory required or moving us towards achieving value for our clients, it can probably be chucked out and no one would miss it!

Coming back to software testing, we see that many testers are still confused about their role in software development. How can we make testers understand that, 'Responsibility lies with us'?

That's partially because the software development industry can't make up its mind about testing's role!

It seems like every 5 years, the development or project management community comes up with some buzz word-laden variation of an iterative approach to delivery that diminishes, changes, or supposedly makes obsolete the role of software testing. Whether it be a more technically demanding testing role or a functional subject matter expert in a business process, having a holistic, value based approach to software testing that relies heavily on system thinking will always have a role in a project – and I think be essential to its success.

Most of the times testers are underrated by organisations and thus they lose confidence. How to increase their visibility and make them confident? What steps should leadership take to motivate testers?

There was a recent article in Forbes magazine about why top talent leave organisations, and they boiled it down to one reason: ***"Top talent leave an organization when they're badly managed and the organization is confusing and uninspiring."*** In addition to dealing with those generic problems, software testers typically have double the impact here, because as a community, we are particularly bad at articulating our value. As well, in my view the software testing industry (vendors, tools, associations, etc.) has had a large part in undermining our own credibility with our "clients."

To overcome these obstacles, I would offer the following advice:

- 1) Know who your clients are and what they value
- 2) Align yourself and your test approach to protecting that value
- 3) Know your organisation and their strategic objectives
- 4) Align yourself and your test approach to helping your organisation achieve its objectives.

And lastly, be able to clearly articulate each of those and how to identify them in your work.

It sounds simple, but I am consistently amazed at how few testers can do those basic things and don't let them permeate their approach to their job.

How do you compare maturity of Software testing compared to the quality processes in other industries in view of exponential growth of software industry?

I'm not a big fan of maturity measurement, and I definitely think you should try as best to compare apples to apples when drawing correlations between things.

What I would note, is that different industries have different risk profiles and therefore require different test approaches and techniques. Even within the financial services industry, we alter our approach by asset class or system as risk profiles are different from risk engines, to order management systems, to high frequency trading applications.

That's what I like about Context Driven Testing as it allows our testers to use their brains when choosing the test approach instead of mindless going about their jobs.



You have seen software testing since long, its situation in past and its state at present. Do you think that the exponential growth of software industry has lead to compromising on the quality processes?

Good grief, the last thing the software industry is lacking is processes!

I think we've seen an explosion in technology application and rates of adoption in the last 10 years and it's only going to get larger in scale and pace. What I think we'll see is a more practical application of some of the existing processes and a realisation that software development is NOT analogous to manufacturing.

A recalibration of client expectations and how we as a community relate to and articulate the risks and difficulties of delivering is probably what's needed more than new or better processes. Jerry Weinberg has written loads of great stuff about this and I would recommend either "Quality Software Management – Volume 2 First Order Measurement" or "The Psychology of Computer Programming". Jerry will forget more than I'll ever know about the subject!

AST (Association for Software Testing) is known for its efforts towards improving state of software testing. Being at the Board of Directors of such esteemed association, how do you see the state of software testing in next 5 years?

Well, I am not even seated onto the AST board yet, but was very flattered to be nominated and extremely honored to have been elected. I am looking forward to getting to know the other members of the board, and have already had some great exchanges with them on the working groups and objectives for the AST. I am a big advocate of software testers, and believe a vibrant community like our industry should be supported by networks for collaboration and idea exchanges. I believe the AST is the best positioned in the world to deliver just that, and I am excited to start working with them.

As for the future of testing over the next five years, I believe we'll continue to see a drive towards more agile process and team structures. The rate and magnitude of technology change will force organisations to adapt quickly whether it is to new regulations, market fluctuations or the ease at which customers can swap providers and services. I also believe there will be a big shake up in the test tool market as the older entrenched tools get replaced by lighter, more adaptable tools sets.

What is your opinion about 'Rapid Software Testing'? What made you to go for implementing RST across the GTC? Would you recommend it to leaders in other organisations?

I think my continual support and programme of work adopting the core principals are as good an endorsement as I can give of RST.

James Bach and the RST courses are analogous to what I call "an adenine shot to a tester's heart." Whether you are an experienced tester or someone new to the field, RST has something in there for you, and I have seen more "aha" moments within the GTC since we started doing the training.

I would, and often do recommend RST to leaders in other organisations and have had many discussions with them about how to manage the implementation.

As well, "Lessons Learned in Software Testing" is on our mandatory reading list for everyone in the GTC, and in fact we had 200+ copies sent to our team in India so everyone could have a copy on their desk! Ultimately, I believe RST and the CDT school of testing most accurately model how the testing process actually works and embodies the approach and mindset of someone pursuing software testing as a profession.

Treating our industry as a craft is something we take very seriously in the GTC and RST fits into that approach nicely.

What message would you like to give our readers?

Don't be afraid to try new things as you will learn more from your failures than your successes.

One of my favourite quotes from Herman Melville sums it up pretty good: "Failure is the true test of greatness. And if it be said, that continual success is a proof that a man wisely knows his powers, — it is only to be added, that, in that case, he knows them to be small."

Do you read 'Tea-time with Testers'? What is your opinion about us? Would you like to recommend it to your colleagues?

I do read 'Tea-time with Testers', albeit I am a recent arrival on the site. I am an avid **Jerry Weinberg** devotee, so it was fantastic to see him so involved in the content there.

I have already recommended it to my friends and contacts. Keep up the great work!





are you one of those
#smart testers who
know d taste of #real
testing magazine...?



then you must be telling your friends about ..



Tea-time with Testers

Don't you ? 😊



Tea-time with Testers !

first choice of every #smart tester !



testing intelligence

- *its all about becoming an intelligent tester*



an exclusive series by **Joel Montvelisky**

Ask yourself what were you hired to do?



Many times it feels like the people in the project don't really understand what you are doing and why? It can go as far as feeling the testing team is the only one stopping the product from being released to the field. Have you ever wondered if this is right?

I read a *nice post* in TestingReflections by John McConda. John talks about the fact that testers provide a service to the team, and that many times we suffer what he calls "genius envy", leading us to act as gatekeepers and/or play games of power with the rest of the project stakeholders.

I agree with him.

Not only that, but I can testify that I used to suffer of these “genius envy” attacks (I just love his definition) and was constantly looking for an escape route out of the testing Job misery I had gotten myself into.

After a while, as I started learning more about testing and how to do it right, I began perceiving the real value we could bring into the process. I remember how my personal perspective changed; I was suddenly able to explain to my friends and family what I did for a living without feeling the need to apologize for being second best in the team.

But I want to take this to another point, the point where we should define our tasks as services and ourselves as service providers.

Think for a moment about another service provider in a different professional area, a tailor for example. Whenever a client comes to his shop, the tailor knows he needs to understand the needs, likes and constraints of his customer in order to design and make the correct garment.

Why should testing be different? We know that our customers want to use our testing abilities, but we need to understand what are their objectives, needs and constraints in order to provide the correct set of test and services.

At this point I want to introduce something that helps me provide a better service to my development “customers”, my personal view of the testing team’s objective.

Based on my experience I see the goal of the testing team as follows:

“To provide correct and timely visibility into the product and process, in order to help the organization make tactical and strategic decisions; and to do this as close as possible to the defined constraints of schedule, functionality and costs”.

The main components of this definition are:

1. Visibility – testing is not aimed at finding all the bugs; it needs to provide a clear view of the status of the application (bug numbers are only a small part of it!).

2. Correct and timely – the information should be right and corroborated (if possible backed by facts and not only gut feelings); and it needs to be provided when it matters and is still relevant to the stakeholders and the project.

3. Help the Organization make decisions – we are not gatekeepers in charge of stopping the application from reaching the field. We are in charge providing inputs about the product and process that will help the Company to act correctly; these actions will be based in part on our information but also on additional factors and inputs such as marketing considerations, sales objectives and deals, operations, etc.

4. Work based on constraints – this is where reality enters the scene and we need to do our best under the current project constraints.

All the points above provide us only with a framework. The concrete definition of our objectives and tasks will be dictated by the goals, needs and constraints of the project as defined by its stakeholders.

The job of the Test Team Manager is then to understand what is needed, and to create and execute a plan that supplies these needs.

If we understand we are providing a service and we know who are customers are, then we can go to these customers and ask them what their needs are. If we know why they are "hiring" us we will know how to provide our service in the best possible way.

Just go and ask them why did they hire you?

[Back To Index](#)



Joel Montvelisky is a tester and test manager with over 14 years of experience in the field.

He's worked in companies ranging from small Internet Start-Ups and all the way to large multinational corporations, including Mercury Interactive (currently HP Software) where he managed the QA for TestDirector/Quality Center, QTP, WinRunner, and additional products in the Testing Area.

Today Joel is the Solution and Methodology Architect at [PractiTest](#), a new Lightweight Enterprise Test Management Platform.

He also imparts short training and consulting sessions, and is one of the chief editors of ThinkTesting - a Hebrew Testing Magazine.

Joel publishes a blog under - <http://qablog.practitest.com> and regularly tweets as [joelmonte](#)

Teach-Testing →

An Ambitious Campaign by Tea-time with Testers




Click here to Cast Your Vote



by





Call for Articles !

Have you got something to say?

yes, we are listening you...!!!

"Tea-time with Testers" firmly believes that one of the best ways to improve upon software testing is to listen to the lessons learned by others and their experiences too.

So, if you have an interesting story that you'd like to share with the world, contact us at teatimewithtesters@gmail.com.

Submit your articles, stories, thoughts around software testing.

now its your chance to be heard...!

Click [HERE](#) to read our Article Submission FAQs !

T ' Talks



T. Ashok exclusively on software testing

"Great Expectations" - Excellence requires empathy

What do we do when we find defects in software? We triage them based on severity and priority; fix the ones on the top, leaving some of the ones at the bottom possibly open.

In my recent discussions with a Japanese company, I encountered an interesting situation. The company had purchased a large software package that was customised and then deployed. During the process of customisation, they discovered quite a few bugs in the base code, a few hundred. Being Japanese, they were aghast at the quality of code and lost confidence in the software supplier.

Now I was curious. How could global vendor ship enterprise class financial software with fair number of open bugs? Were their tests less mature? Or did they release the system with open issues that they deemed they could fix later?

On the other hand, what kind of defects were encountered by the Japanese company? Being a stickler for perfection, were they less tolerant of even minor defects?

Digging into the details, I discovered that the number of post-release defects was about 500+ and all of these were functional in nature. A majority (60%) of these were possibly related to incorrect validation of inputs, issues in the user interface and so on. Is it that the vendor thought that these were not critical enough to fix or not so important to find (i.e. go after)?

If for a moment we do consider that the majority were indeed not deemed critical, how come the customer lost confidence in the vendor? Hmm.. This is what set me thinking...

Most often the rating of a severity and priority is driven by rating scale that is typically part of the SDLC process. The rating scale is a guideline while the actual value accorded to a defect is subjective, based on the triage team. Now think about this... The rating has to be done from the perspective of the customer, keeping in mind the different expectations of each customer. In this case, the customer being Japanese, their expectations on quality are very high and tolerance for any kind of defect very low. Hence even what is deemed as minor defect by the vendor is deemed 'not minor' by the customer resulting in low confidence.

So, it all comes to a simple thing - "Are you really sensitive?" Sensitive to what the end users expect? Being emphatic is possibly the most important social trait that is needed for being a good test engineer or a great software developer. To be in the end user's shoes, to think like them, to understand their feelings when they interact with your software is indeed very important to making key decisions in development/testing to deliver great software.

In all my interactions, I strive to look beyond the technical stuff by peering into the "person". What do they expect? How would it help them? What would they not like? The technique "Viewpoints" in HBT (Hypothesis Based Testing) is what I frequently use to viewing the system from the end user's view. It is indeed very challenging, interesting and also very frustrating as you discover how much you do not know/understand. But hey, it is fun!

So in closing, if the vendor had attempted to understand the social traits of the Japanese customer-their penchant for fine details, their low tolerance for any kind of defect, their zealous "customer orientation", the test strategy, test cases, test effort may have been quite different. And the confidence in the vendor would not have been lost.

It is not just about technology, process and tools. It is about the live person who uses your system. Never forget the HUMAN side of the equation. The social side.

Empathise. Understand their issues. Build with them in mind. See the joy when they use your system.

Absolute bliss. Have a great day.



T Ashok is the Founder & CEO of STAG Software Private Limited.

Passionate about excellence, his mission is to invent technologies to deliver "clean software".

He can be reached at ash@stagsoftware.com





OUR PARTNERS

Quality Testing



Quality Testing is a leading social network and resource center for Software Testing Community in the world, since April 2008. QT provides a simple web platform which addresses all the necessities of today's Software Quality beginners, professionals, experts and a diversified portal powered by Forums, Blogs, Groups, Job Search, Videos, Events, News, and Photos.

Quality Testing also provides daily Polls and sample tests for certification exams, to make tester to think, practice and get appropriate aid.



Mobile QA Zone

Mobile QA Zone is a first professional Network exclusively for Mobile and Tablets apps testing.

Looking at the scope and future of mobile apps, Mobiles, Smartphones and even Tablets, Mobile QA Zone has been emerging as a Next generation software testing community for all QA Professionals. The community focuses on testing of mobile apps on Android, iPhone, RIM (Blackberry), BREW, Symbian and other mobile platforms.

On Mobile QA Zone you can share your knowledge via blog posts, Forums, Groups, Videos, Notes and so on.



Testing PUZZLES

by Sebi



Claim your **Smart Tester of The Month Award**. Send us an answer for the Puzzle and Crossword below b4 20th Nov. 2012 & grab your Title.

Send -> teatimewithtesters@gmail.com with Subject: Testing Puzzle

**Exciting
PRIZE for 1st
three
WINNERS***

NOTE: S.T.O.M. contest comprises of Testing Puzzle + Crossword. To claim their prize, participants should to send answers both for puzzle and crossword.

*CONDITIONS APPLY.

“Find it if you can”

29102012=1

11041342=3

18084234=1

05011000=7

14125674=5

07040002=1

31051743=5

20091976=?

[Back To Index](#)



Biography



Blindu Eusebiu (a.k.a. Sebi) is a tester for more than 5 years. He is currently hosting European Weekend Testing.

He considers himself a context-driven follower and he is a fan of exploratory testing.

He tweets as @testalways.

You can find some interactive testing puzzles on his website www.testalways.com



TESTING CROSSWORD



1		2		3			4
5							
					6		
7				8			
	9						

Horizontal:

1. It is a web-based test case management tool (8)
5. A test suite that exercises the full functionality of a product but does not test features in detail (7)
6. Test cases are generated using the extremes of the input domain, is called ____ in short form (3)
7. The consequence of a test (6)
9. It is a web based bug tracking system (6)

Vertical:

2. It is a mobile test automation tool for Android, iPhone, Blackberry, Symbian & WindowsPhone7 (7)
3. It is a measure used in software testing, the first word (4)
4. A device, computer program, or system that accepts the same inputs and produces the same outputs as a given system. It is called _____ (8)
5. Ajax test runner for php, the first word (6)
6. Testing in which all branches in the program source code are tested at least once. It is called ____ in short form (2)
8. A white box testing technique that exercises program loops. It is known as ____, in short form (2)

T	E	S	T	I	T	E	M	
E		A				A		
S	O	F	T	W	A	R	E	C / P
T		S		E			M	
M				B	T			
A	P	I		L		T	D	T
K		T		O	M	C		
E	T	G		A		W	A	S
R				D			T	

Hex Mode allows to display bigger numbers in general as it accepts 64 bit maximum characters but zero is substituted for any number greater than 0xFFFFFFFFFFFFFFFF in windows calculator.



Join us on Facebook.

You are just a CLICK AWAY



Every Tester

who reads Tea-time with Testers,

**Recommends it to friends and
colleagues .**

What About You ?

Our Testimonials

Hi Team,

I regularly read your magazine and also solve puzzles and crossword.

With Tea-time with Testers, we have found fun filled way of learning and sharing and I thank you for that.

- Karthik

Thanks for giving us great issues every month. TTWT Rocks !

- Manali.

in ne>xt issue

articles by -

Jerry Weinberg

T Ashok

Joel Montvelisky

Martin Jansson

Bernice Ruhland

Ben Kelly

our family

Founder & Editor:

Lalitkumar Bhamare (Mumbai, India)

Pratikkumar Patel (Mumbai, India)



Lalitkumar



Pratikkumar

Contribution and Guidance:

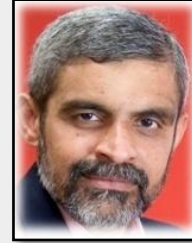
Jerry Weinberg (U.S.A.)

T Ashok (India)

Joel Montvelisky (Israel)



Jerry



T Ashok



Joel

Editorial | Magazine Design | Logo Design | Web Design:

Lalitkumar Bhamare

Image Credits- weiphoto.com and Favianna Rodriguez

Core Team:

Anurag Khode (Nagpur, India)

Dr.Meeta Prakash (Bangalore, India)



Anurag



Dr. Meeta Prakash

Testing Puzzle & Online Collaboration:

Eusebiu Blindu (Brno , Czech Republic)

Shweta Daiv (Mumbai, India)



Eusebiu



Shweta

Tech -Team:

Chris Philip (Mumbai, India)

Romil Gupta (Pune, India)

Kiran kumar (Mumbai, India)



Kiran Kumar



Chris



Romil

*// Karmanye vadhikaraste ma phaleshu kadachna |
Karmaphalehtur bhurma te sangostvakarmani //*

To get **FREE** copy ,
Subscribe to our group at



Join our community on



Follow us on



www.teatimewithtesters.com

