INTERNATIONAL MONTHLY FOR NEXT GENERATION TESTERS

Tea-time with Testers

JANUARY 2012 | YEAR 1 ISSUE XII

Articles by-

Jerry Weinberg

Karen Johnson

Joel Montvelisky

Bernice Ruhland

Samarjeet Mohanty

T Ashok

TEA-TIME WITH TESTERS

First Indian Testing Magazine to reach 87 Countries in the world !

Created and Published by:

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

Editorial and Advertising Enquiries:

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

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

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

Editorial

New Year and the New Beginning!

Dear Readers,

It's giving me an immense pleasure to be able to talk with you in this 12th issue of Tea-time with Testers.

I still remember *The January of 2011* when I had spent entire month with almost no sleep. There were so many questions like where to get articles from? How to design the magazine? How to publish it online? How to build a website? and most importantly What should be my focus?

I was not sure if my ideas would be welcomed in community. At one point I even thought to give up but my desire for creating one *different testing magazine* was so strong that it did not allow me to sleep. I woke up again and started working.

As I have always admitted, "A strong will opens all doors of opportunities for you" and so it did for me. I met Pratik, we brainstormed and decided to give it a go. Our frequent Tea-breaks gave us many ideas and so the magazine-name as well (thank you all for loving it ☺). Mail-id got created followed by our website. The only question was of articles and the only reason we could get them is "Our Lovely Testing Community".

We have managed to complete one year only because of our testing community, many legends who made their contribution to it and their religious followers like you all !

31st Jan 2011 was the day when we made our first public appearance. On same day of 2012 and with lots of mixed emotions, I declare successful completion of our 1st year of community service. And we are committed to serve you better for years to come.

It's New Year and New beginning for us. My sincere thanks to our all contributors, column writers and YOU dear readers. This 12th issue of Tea-time with Testers is dedicated to all of you.

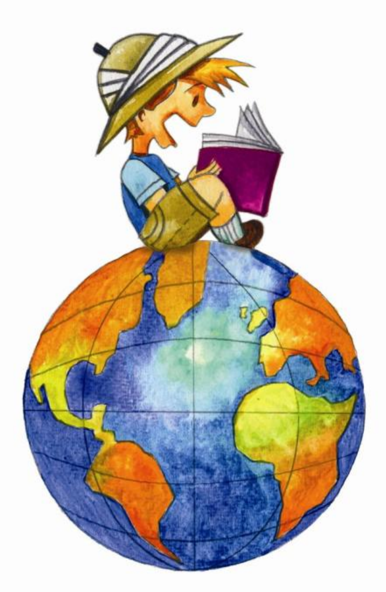
(*Oh Yes! Our next issue is going to be very special for obvious reason* ☺.) Stay tuned and enjoy reading!

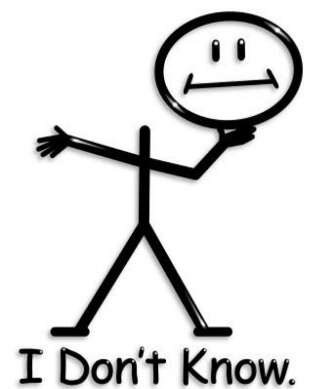
Yours Sincerely,

Bhamare

- **Lalitkumar Bhamare**

QuickLook





Editorial

What's making News?

Tea & Testing with Jerry Weinberg

Speaking Tester's Mind

As a Tester, "Do You Think Globally?" - 18

In the School of Testing

Developing A Career Plan - 24

SPE- Implement It To Avoid Getting Sucker Punch-29

Stop Being A Non Technical Tester – 33

T ' Talks

Single Entry, Single Exit - Singular Purpose – 40

Tool Watch

Testing Puzzle – S.T.O.M. Contest

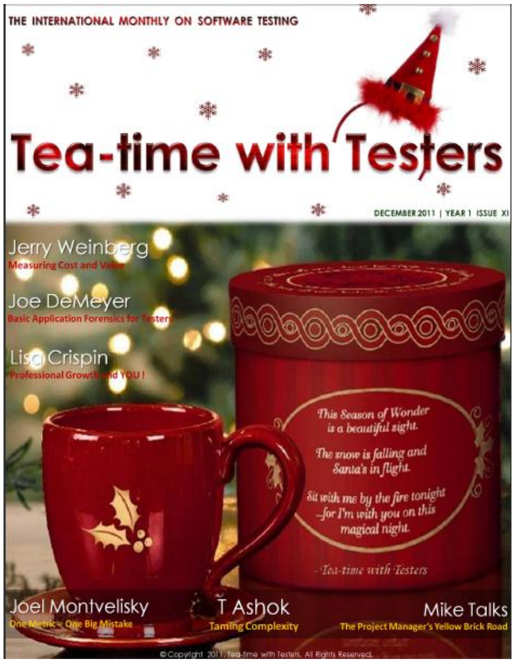
Our Testimonials

Family de Tea-time with Testers

Testing Puzzles
by Sebi

Crossword
by

QUALITY TESTING
Quality is delighting customers


THE INTERNATIONAL MONTHLY ON SOFTWARE TESTING
Tea-time with Testers
DECEMBER 2011 | YEAR 1 ISSUE XI
Jerry Weinberg
Measuring Cost and Value
Joe DeMeyer
Basic Application Forensics for Testers
Lisa Crispin
Professional Growth and YOU !
This Season of Wonder is a beautiful sight.
The snow is falling and Santa's in flight.
Sit with me by the fire tonight ..for I'm with you on this magical night.
- Tea-time with Testers
Joel Montvelisky
One Metric = One Big Mistake
T Ashok
Taming Complexity
Mike Talks
The Project Manager's Yellow Brick Road
©Copyright 2011. Tea-time with Testers. All Rights Reserved.

December 2011

You're making a real contribution.

I've finally finished reading this December issue. Once again, I really like the visual presentation, but I want to comment on a few of the articles:

- I'm a huge fan of The Wizard of Oz (my "More Secrets of Consulting" is based on Dorothy's team), and I enjoyed the article.

- Lisa Crispin's article is great, the kind of thing we need more of if we are to stimulate the growth of professionalism in testing. This sort of essay has lasting value.

- I'm very pleased to find in the mind-map article that you've created a way to enlarge the diagrams. It makes all the difference in the readability of the article.

Also, though mind-maps are tools, this kind of generalized presentation (rather than details of some particular vendor's tool) is of lasting value.

- I always like Joel's case studies of applying intelligence to testing.

- And T. Ashok's article on complexity is the kind of philosophical essay that's always good. Testers need to understand complexity, as it's the foundation of the need for our profession. But we need a whole lot more on the subject of dealing with complexity--which is the basis of my General Systems Thinking series.

- I don't allow myself the luxury of becoming hooked by the puzzles and crosswords, but I think they are a good feature.

So, all in all, keep up the excellent work.

You're making a real contribution.

- **Jerry Weinberg**

I eagerly wait for every issue !

Dear Team,

Thank you so much for taking these much efforts and creating such a beautiful magazine.

I am so addicted to your magazine that I eagerly wait every month for it.

- **Durga .**

TO SEND YOUR LETTERS:

WRITE TO US AT –

teatimewithtesters@gmail.com

What's making News?

- find out the latest happenings in the technology world

mage: www.bigfoto.com

Test and Verification Solutions creates strategic presence in Asia Pacific market

Bristol (England) -- December 21 2011 –

Test and Verification Solutions announced today that it has expanded its global presence in Asia Pacific region with the recent opening of its development centre in Chennai, India and signing with Sales representatives from China mainland and Taiwan. The strategic presence in Asia Pacific region will help TVS deliver high quality software testing and hardware verification products and services to semiconductor companies and design houses in that region.

Dr. Mike Bartley Founder and CEO of TVS said, "We are excited about our strategic presence in Asia Pacific region. India, China and Taiwan today are the global hub for development and manufacturing. Some of the leading market research agencies predict continued and unabated growth in this region. We are confident that TVS products and solutions will address the pain points of our customers and help them not only reduce development costs, but also accelerate their go-to-market plans."

TVS India at Chennai will be the primary development centre. With ongoing customer activities that include offshore development centre (ODC), VIP development and onsite resource augmentation support, the Chennai centre serves some of the leading semiconductor companies worldwide. TVS will also open a development centre in Bengaluru in the forthcoming year and would look to provide exciting opportunities to the local talent in India.

TVS has signed an exclusive agreement with HyperSilicon for China mainland and Grand Technology Inc. for Taiwan to promote its products and services offerings. The proven track record of TVS coupled with HyperSilicon's and GTI's vast experience in promoting ASIC, SoC based solutions and EDA tools, will allow potential customers to access, through a local medium, the value additions that TVS offerings would bring to them.

About Test and Verification Solutions

TVS provides services and products to organisations developing complex products in the micro-electronics and embedded systems industries. Such organisations use TVS to verify their hardware and software products, employ industry best practice and manage peaks in development and testing programmes. TVS hardware verification services include onsite/offshore verification support and training in advanced verification methodologies. TVS also offers Verification IPs and its own Verification (EDA) signoff tool (asureSign™). TVS embedded software testing services includes methodology development, software test training and execution support services.

TVS has its headquarters in Bristol, England with offices in Europe and India.

Are you interested in publishing the news about your own firm, community, conference etc in Tea-time with Testers?

Feel free to write to us at: teatimewithtesters@gmail.com with "News Enquiry" in your subject line.

Announcing...

Smart Tester of the Month Awards !!!

❖ Winners for Testing Crossword :

Devaganaraja Gopalasetty, Hyderabad

Vijetha Thota,Chennai

Sonal Agarwal , Noida

Laxmi Kanth, Chennai

Ranjgarajan, Chennai

Kalyani Muthu, Chennai

Congratulations !

Discussion helps !

How about talking with us on Facebook?

Come ! Let’s have a nice Tea-time there !

Find us on Facebook

CLICK HERE

Look Who's Rising

Twelve Months

10000+ Readers

87 Countries

ONE Magazine

TEA-TIME WITH TESTERS

Subscribe here Right Away to get our all Issues for FREE

Measuring Cost and Value (Part 3)

1.4.3 Possible difficulties

This kind of study can be an enormous amount of work, to interview people and track down effects. Unlike some methods, the work involved in this one is impossible to estimate accurately before you start, because you don't know the scope of the impact. On the other hand, if you discover new effects, you have improved the requirements process.

The biggest reward is the understanding. Before embarking on this kind of study, ask yourself if one of the simpler methods would be just as effective. Ask your sponsor if one of those methods would be just as convincing.

This method can also be lots of fun if you like learning new things and interacting with people. However, the fun can be a trap if you lose sight of what you're trying to accomplish.

The final report can be another trap. You have so much data, you'll want to present it all, but that will turn off most executives. You must keep the supporting data in the background and present only the most important impacts in no more than one page.

1.4.4 Sample report #1

Sample reports 1 and 2 show contrasting applications of the detailed impact method. Report 1 is an after-the-fact report based on a detailed study of a pilot installation of a laser printer. It predicts ongoing savings for a typical work unit for each laser printer installed.

Report 2 estimates the impact of any rumour-reduction program by detailing the cost of disposing of one major rumour. The figures are based on interviews with a small sample of managers and staff in an organization of around 3,500 people.

Report 1 is a benefits report showing intangible benefits of a rather tangible change. Report 2 is a cost report showing tangible benefits of whatever intangible change might reduce the flow of rumours. Together, these two reports suggest the range and power of the detailed impact case study for making the value of a change project visible.

Impact Assessment of High-Speed Laser Printer

Replacing Several Dot Matrix Printers

Although the study identified dozens of effects of installing the Laser Printer, the following are the major ones in terms of cost saving or value added:

Clerical Staff (hours per week per person)

Trips to copier eliminated for letter copies 2.0

Reduced interaction with Print Shop 1.5 [Note 1]

Less printer setup and maintenance time 1.0

(4.5 hrs/wk x 3 staff x $30/hr standard cost x 52 wks) = $21,060 [Note 2]

Professional Staff [Note 3] (hours per week per person) Fewer iterations to finished product 1.0

(1.0 hrs/wk x 7 staff x $80/hr standard cost x 52 wks) = $29,120

Expenses (dollars per year)

Reduced printed pages due to higher print quality [Note 4] = $6,000

No longer keeping letterhead, or any special forms = $800

Reduced artwork contracting = $1,800

Increased printer supplies = ($200 extra cost) [Note 5]

(Net annual dollars saved over spent) = $8,400

Less Tangible Items (reported as important by interviewees) [Note 6]

(estimated dollars/year to get the same effect by another method) [Note 7]

Quieter environment: sound shield for impact printer = $300

Better company image in correspondence increased secretarial and professional time, printing costs = $8,000

Better quality handouts for presentations increased secretarial and professional time, art costs = $4,000

More working space (no impact printers, fewer supplies to store) @ standard costs per square foot = $2,000

(Total dollars) = $14,300

Summary

Our best estimates for the one year value of installing one laser printer in the work group of 7 professionals and 3 clericals is:

Lowest value [Note 8] (assuming no impact on professionals) = $30,000

Highest value [Note 9] (including many less noticeable effects) = $100,000

Most Easily Supportable value [Note 10] = $60,000

The estimated life of the laser printer in this environment is 3 years.

Detailed Impact Case Study 1: Analysis of the impact of the introduction of a new high-speed laser printer.

1.4.5 Notes on the detailed impact case study 1

Note 1: In analyzing effects, interruptions are often a critical feature, and one that is usually underestimated by workers. Studies of knowledge workers have shown that any interruption typically costs the equivalent of 15 minutes lost work, in addition to the time handling the interruption.

Note 2: Most large organizations have some system of standard costs. Use these when available rather than develop your own. It saves time, and it's more convincing.

Note 3: Professional staff savings should always be separated from clerical staff savings. Not only do professionals carry a different rate, but their work tends to be more elusive to quantify, which makes your analysis more easily questioned. Separating the labour categories keeps the questioning to a limited range.

Note 4: The biggest effects quite often turn up where nobody anticipated. In this case, careful interviewing revealed that "reports are smaller" with the laser printer. Sharper printing allowed smaller type sizes, as in this report.

Analysis developed an average reduction of 25% in the number of pages, with consequent reduction in copying, printing, storing, and mailing costs.

Note 5: Never neglect to show negative impacts when you find them. In this case, although the effect is smaller than most of the others, listing it shows that you have considered negative effects, and thus makes the report more convincing. Negative effects are also a check on your objectivity. If you haven't found any negative effects, you haven't looked hard enough. Go back and find some, and put at least one in your report to management.

Note 6: Leave out the intangibles unless somebody believable says they're important. Intangibles are usually small, and not worth getting into political fights over. If you think they ought to be important but nobody says so, you can ask direct questions in your interviews. Listen, don't sell.

Note 7: This is the most conservative way to put a value on intangibles. It assumes that if you want them, you could obtain them by some other change. Cost analysis of that method puts a bound on the value of the change you are analyzing. Putting a value on the benefit could give a large dollar amount, but that could easily be contested.

Note 8: The low estimate includes "hard" data only, whatever that means to your sponsor.

Note 9: The high estimate includes any effects you can make a reasonable argument for, without engaging in total fantasy. If it takes your high estimate to make your case, then you probably haven't made it.

Note 10: This means you can make this case without using arguments you think aren't as strong as they could be.

In other words, it's the one you're most willing to stand behind.

1.4.6 Sample report #2

Replacing old technology with new technology represents a rather concrete set of values, but the method is not limited to the tangible. To illustrate how the method can be used to evaluate largely intangible benefits, I've chosen a report (origin disguised) one organization made to justify (or not) a campaign to reduce rumour mongering.

Analysis of Cost of One Rumour or Savings from Its Early Elimination

Rationale

Management and others have expressed their annoyance with the constant circulation of rumours in our organization. Several methods have been proposed to squash rumours early in their life cycle. This report estimates the cost to the company of one typical rumour. The rumour we chose to examine was circulating last July/August. Although there were many versions of the rumour, the essential idea was this:

Management has decided to build an extension to building R4 where the parking lot is now. As a result, non-management people will lose their parking privileges and have to pay for parking. The nearest large lot is 3 blocks away, and costs $45 per month, with no in-out privileges. Managers will keep their parking privileges.

Here are some estimated costs of this rumour, based on interviews with a sample of 7 managers and 16 professional staff:

Management Time (Total hours by typical manager)

Discussions of rumour with subordinates 12.2

Management meetings to discuss handling rumour 4.5

Public meetings to discuss the rumour 1.7

(18.4 hrs/manager x 460 managers)=8,464 hours

(8464 hours x $90/hr standard cost)=$761,760

Professional Staff Time

Discussions of rumour with colleagues (average = 6.3 hrs/discussion) 2.1

Meetings with management to discuss the rumour 0.2

Public meetings to discuss the rumour 0.4

(2.7 hrs/staff x 3,060 staff) = 8262 hours

(8262 hours x $60/hr standard cost) = $495,720

Publication Expenses (dollars)

Cost of circulating 3 memos = $4,500

Cost of preparing 3 memos = $1,500

(Total Publication Cost) = $6,000

The total estimated hours dealing with this (totally false) rumour = 16,726

(This is over 8 person-years. For comparison, the entire Lambda Project took about 7 person-years.)

The total estimated cost of dealing with this rumour = $1,263,480

Detailed Impact Case Study 2: Analysis of cost of one rumour or savings from its early elimination

1.5 Single Greatest Benefit Method

A detailed impact case study may not be practical for all situations. The single greatest benefit method is one approach to putting a floor estimate on value, while keeping the expense and time of the method down to practical levels. The basic question of the single greatest benefit method is this:

What is the one greatest benefit that you can attribute to this change?

1.5.1 Key ideas

The method achieves its speed and savings by keeping the attention focused on a small number of large benefits.

By choosing only one benefit, and choosing the greatest benefit, we guarantee that the analysis is conservative. In other words, if the sample is at all representative, the estimate of total benefits will be on the low side. That's why this method is most appropriate when the expected benefits are large.

Most people can rather easily identify the largest benefit. If they can't, then the benefit must not be that important.

Because there is only one benefit per interviewee, and because it is a large one, you can afford to spend time tracking down the dollar value and reporting it in a way that convinces management.

1.5.2 Possible difficulties

Some people won't be able to say which is the largest benefit. Don't get hung up on this. Just have them pick one of the largest.

Other people won't be able to think of any benefits. You can suggest some benefits that others have found as a way of triggering their thoughts, but don't get hung up on this, either. You're looking for large benefits, and those will be easily remembered, if there are any. If there aren't any benefits for some of the people, that's okay, as the large benefits from others will make up for them.

If nobody can think of large benefits, perhaps the goals of the change were unrealistic. Consider using a method that detects small improvements, such as the Subjective Impact Method.

1.5.3 Example Report #3

There are many practical applications of the single greatest benefit method. For instance, here's a rather spectacular but not atypical case:

A system of specification reviews was introduced, primarily to stem the high cost of fixing errors late in the software development cycle, or after shipment of software. The plan was to study the first fourteen reviews by running down the Issues Lists with the developers. In making appointments, the analyst discovered that one of the products specified was no longer scheduled for development. Tracking down the history, he discovered that the specification review had turned up so many serious issues that management had decided to scrap the project. The analyst then turned to estimating what was saved, producing the report entitled "The Case for Specification Reviews."

to be continued in next issue…

Back To Index

Biography

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

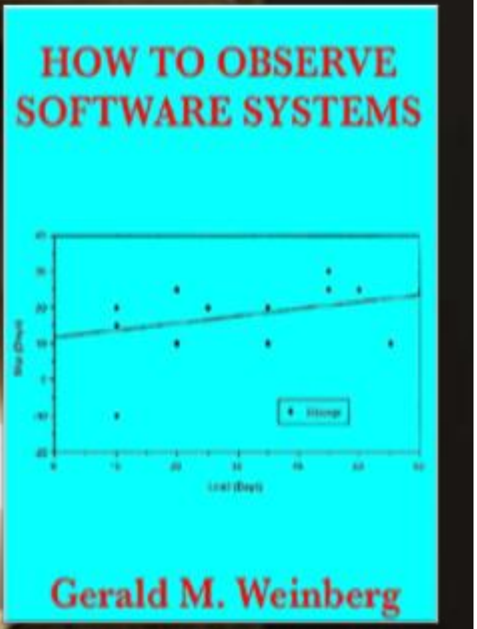
For more than 50 years, he has worked on transforming software organizations. He is author or co-author of many articles and books, including The Psychology of Computer Programming. His books cover all phases of the software life-cycle. They include Exploring Requirements, Rethinking Systems Analysis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

HOW TO OBSERVE SOFTWARE SYSTEMS is one of the most famous books written by Jerry.



This book will probably make you think twice about some decisions you currently make by reflex. That alone makes it worth reading. "Great to understand the real meaning of non linearity of human based processes and great to highlight how some easy macro indicator can give info about your s/w development process." An incredibly useful book. Its sample can be read online here.

TTWT Rating: ★★★★★

To know more about Jerry's writing on software please click here .

Speaking Tester's Mind

- straight from the author's desk

As a tester do you think globally?

by Karen Johnson

Money! Money is one of those few rare words that instantly draw peoples' attention. When we think about money we either think about it in vague large terms like how life would be if we were "rich" or if we won the lottery. Alternately when we think about money, we think specifically in "dollars and cents" and very quickly, without even realizing it, we're thinking in our own local currency. Even the expression "dollars and cents" is an American term referring to the United States currency.

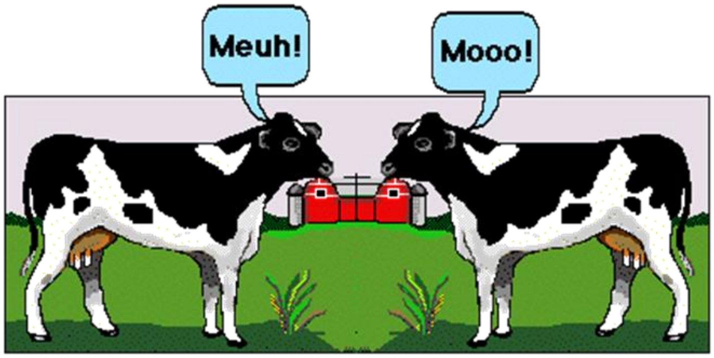
When you work in an international world and maintain a client base located around the globe, you need to think about money in multiple currencies. One of the first steps in thinking globally, is to stop thinking solely from your own perspective which is not necessarily easy as local customs, as well as currency and language tend to be pervasive. As an American, I've come to recognize that our own physical landmass of the United States is so geographically large that many Americans never travel outside of our country which contributes to being less inclined to make mental translations for currency, language or time differences.

On the other hand, in a world that seems increasingly global, we also seem to be losing some individuality. I've had the amazing good fortune to travel to New Zealand twice which is about as far from my home as it gets and what did I find but Kentucky Fried Chicken and Starbucks. It seems currency and language, as well as hard to describe local customs are some of the last signs of local flavor we have left and I feel we should savor the differences. Part of appreciating the differences is fully supporting the differences – which of course translates in the software world as supporting "localisation" or perhaps you spell that as "localization?"

So *potato or potatoe* – of course this expression makes more sense when vocalized than written, but you get the point. We may have differences but we need to recognize and appreciate them and in software, we need to localize and translate those differences – preferably gracefully.

In the case of currency, there are three bits of information I need. I need to know what the currency name is– meaning what are dollars in another currency? Is the currency referred to as euros, pesos, rupees or yen? I also need to know what the currency code is – here in the US, we see references to USD and barely think about it but this code denotes what the currency is such as AUD for Australian Dollar, SEK for the Swedish Krona and LKR for the Sri Lanka Rupee. No worries if you can't memorize this information, it is easy enough to look up. I found this website offers a comprehensive up to date list: http://www.xe.com/symbols.php. In exchanging currency in different countries myself, I've learned that understanding what decimal symbol is used can be confusing – as an American I think in decimal points but other countries use a decimal comma. Again *potato or potatoe,* see this Wikipedia entry for more: http://en.wikipedia.org/wiki/Decimal_mark or look up currency information through a foreign exchange agency or bank.

As I share the link on decimal marks, I noticed the language tag of "en" in the URL and recognize that I am sharing an English translation of the website. See the Iana.org website for a list of HTML tags for languages: http://www.iana.org/assignments/language-subtag-registry Humorous that I would share such a link in an article on localization because this is exactly what we don't want to do to users of our websites and mobile sites, force a language selection. On websites that are used to accommodating multiple languages, you'll often see a flag selection so users can choose a language based on the flag they click on which removes a dependency on reading the words next to a language selection dropdown. Notably tourism and hotel websites seems to be best at providing this option. What does your site provide for users who do not speak the same language as you? Is your site translated? Is your site available in multiple languages?



Language localization is a topic larger than it may seem since localization is not just a case of translating the content but also a process of adapting for cultural uses of the language. L10N – of course there is an acronym although more accurately this is called a numeronym because the abbreviation includes digits, refers to exactly this process – of translating not just the language and text but translating to ensure the meaning is accurately carried into the another language.

How often in movies do we see people thrown into circumstance while traveling where they don't know the language and not only mispronounce but misuse words – and insult someone in the process which is often portrayed in movies as humorous. People may forgive an unknowing tourist but will potential clients forgive a business, your business when your website fails to localize content appropriately? Why find out? This is not scenario for which you want to be begging for forgiveness.

So as great as it is that your company has paid for localization, what if you work as a software tester and you don't know other languages and yet you are asked to make sure the website "works" in other languages. Tough situation, a situation I face most days as one of my longtime clients has a website translated into more than two dozen languages and I all know is English and a small amount of Spanish and Latin from school days long ago.

The good news is that I don't have to read to understand page not found or a timeout error. My point is you don't have to be able to read and write in a language in order to see some error situations. In fact, I've written on the topic of testing languages that are orientated from right to left (RTL) such as Arabic and Hebrew (see:here) highlighting checkpoints centered (no pun intended) on user interface considerations such as dropdown fields, scrollbars, data entry fields, checkboxes and radio buttons as well as ordered and unordered lists. It takes a bit of adjusting to check orientation in the opposite direction than what you are familiar with – one of the most obvious examples is the search box as you type characters, you will find the characters appear in the opposite side from what you are used to if you work with a left to right language (which is the majority of languages).

I've blogged on the topic of search testing in different languages– originally posted on www.TestingReflections.com, this post can also be found on my website here . Since that time, I've added to the content I keep "on hand" to test in "other" languages – quickly having a valid search string in a variety of languages is helpful. It's also helpful to become familiar with what a language looks like so you can recognize issues – one issue is whether or not your browser can handle a language, you may find you have to install a language pack. In the case of older versions of Internet Explorer, I've found several languages that are not supported at which I can detect whenever I see a discouraging string of small boxes such as: □□□□□□□□□□ . If you find a string of small boxes it generally means the language is not rendering accurately. When I look up the language Tamil on Wikipedia, I see these boxes instead of seeing Tamil in the list of languages supported on the left navigation bar. A bit of irony that I cannot see Tamil on a page explaining the language itself.

Language and currency are not the only aspects of localization, there is date formatting to consider although this is usually a less onerous task. I find I frequently am confused and probably confuse other people by referring to dates in MM/DD/YYYY format when throughout Europe and other parts of the world; dates are often referred to in DD/MM/YYYY format. If you are testing a website such as a hotel reservation system, it is essential not to confuse what date is being referred to. In emails, I try to remember to spell the month instead of using a numerical reference. The W3C organization as more information on both localization and internationalization: http://www.w3.org/International/questions/qa-i18n.

In the mobile environment, localization brings another set of challenges to testing when the keyboard itself whether QWERTY or touch may not support other languages. How can I test Arabic searching on a mobile device when I cannot get my hands on a mobile device with an Arabic keyboard? Some testing challenges indeed. Some of these language issues I've worked around by using products such as Device Anywhere or Perfecto Mobile but neither vendor offers a complete solution when it comes to languages and devices.

I've tested multiple languages with SMS texting as well. In 2010, I devised a heuristic to help me think through text testing – the heuristic is **RSTLLLL**. The original article appeared on Search Software Quality here.

The heuristic works as follows:

Reply: Can I send a reply to the sender? Does the sender receive my message?

Sender: Who is the sender? Does the sender's information appear correctly? If the sender is already loaded into your address book, the sender's name might appear "resolved." When the sender is a person, it would be the person's name as stored in your address book. Does your application have a name? How does the sender information appear on a phone?

Timestamp: What time is the message stamped? Whose time is used? Is it the time the application sends the message or is it the timestamp from when the telecom company sends the messages out?

List: Was the text sent to one user or was the message sent as part of a list? Were there supposed to be multiple people who received the message, did everyone on the list get the message?

Links: Does the message include a link? Does the link work?

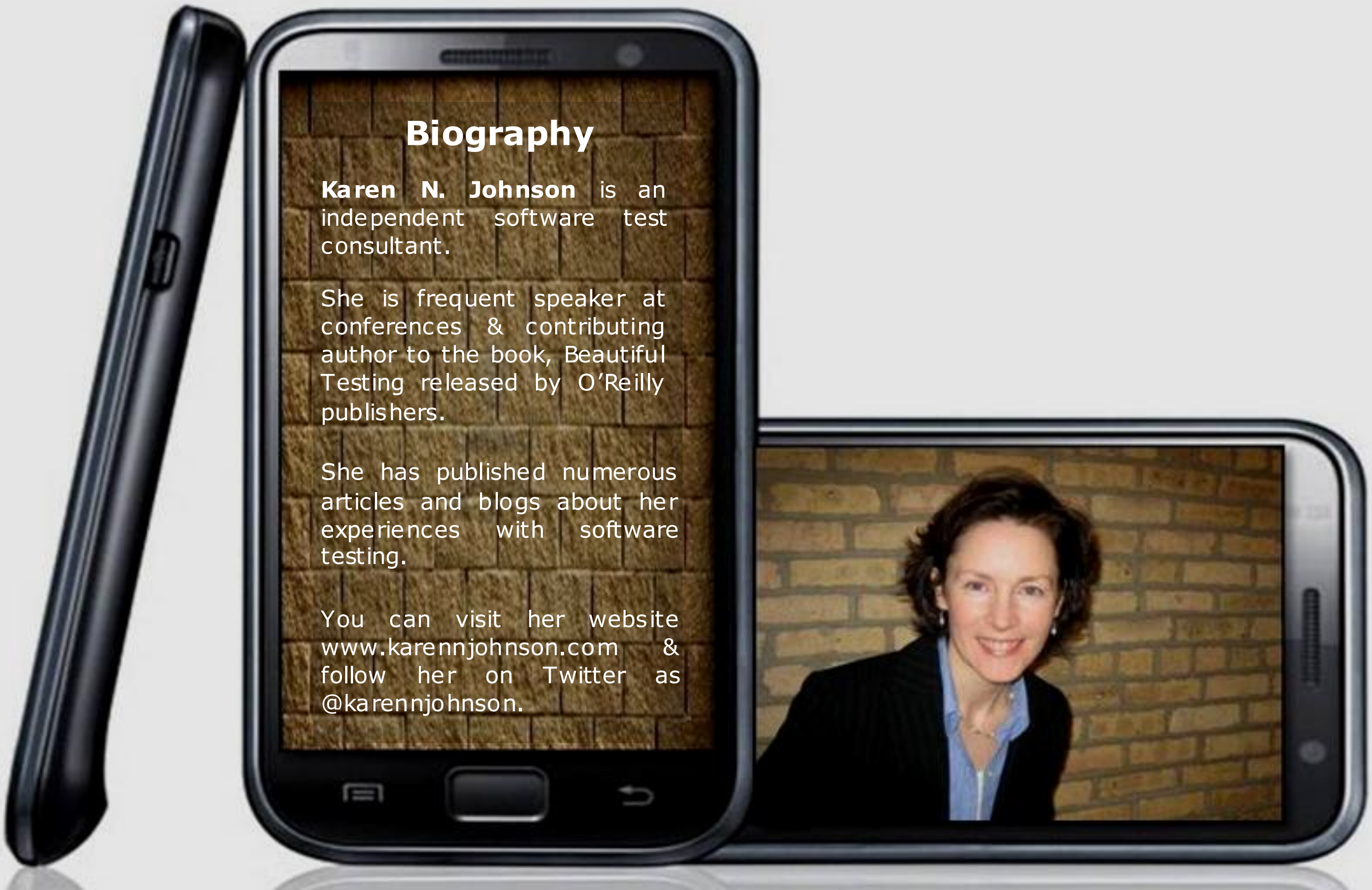
Language: What language was the text sent in? What about Unicode characters? What about non-Latin characters?

Length: Messages end at 160 characters. Did I receive the entire message? What if a link was included as part of the message and the link was included at the end of the message, did the link get truncated?

As we live and work in an increasingly global world, we learn about each other's differences. In Paris, my favorite coffee is referred to as a café au-lait while in New Zealand, I order a flat white and at home in America, I drink drip coffee, coffee from a French press (a taste I acquired in New Zealand) and espresso from a small manual Italian machine that is one of my more favorite possessions. From languages, to currency to date format differences and time zone changes, our websites and mobile applications (including texting) increasingly draw clients from around the globe.

Has your software been adapted for a global audience? As a tester do you think globally?

Back To Index



Biography

Karen N. Johnson is an independent software test consultant.

She is frequent speaker at conferences & contributing author to the book, Beautiful Testing released by O'Reilly publishers.

She has published numerous articles and blogs about her experiences with software testing.

You can visit her website www.karennjohnson.com & follow her on Twitter as @karennjohnson.

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

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

In the school of Testing
for your better learning & sharing experience

"Career Development and Learning Strategies for Testers" is a series of articles providing different approaches to develop testers' skills and knowledge from both a managerial and tester perspective.

By Bernice Niel Ruhland

2. Developing a Career Plan

The focus of this article is developing a career plan with approaches that a manager can implement for his department and testers can adopt for their own professional development. The tools and techniques discussed can be modified to individual needs to customize a personal approach to career planning.

Developing a career plan whether formal or informal can be helpful in guiding a tester's career and reviewing progress. A formal plan may be written with a manager that defines a specific career path within a company or a tester can develop his own plan which might highlight specific goals or learning opportunities. Regardless of the approach, a manager can coach his employees for career-progression and a tester can take ownership of his future through his actions. The end-result of career planning should be skills and knowledge improvement and not a formal document that requires a lot of maintenance. The following sections provide suggestions to get started with developing a plan.

What is Your 5-Year Goal?

In order to understand a tester's career aspirations, many managers ask the dreaded question "where do you want to be in 5-years". This is a difficult question to answer with evolving technology providing new opportunities. Sometimes testers are not aware of their potential and they may provide generic answers such as "I want to become a more skilled tester".

A better question might be: how satisfied are you, with where you are at, in your career? Based upon the response a discussion might continue with questions such as:

- What do you enjoy most about your job?
- If you could change something, what would it be?
- Based upon what you would change:
    - What skills do you need to improve?
    - What knowledge do you require?
    - What opportunities would be helpful?

- What project did you feel professional satisfaction and why?
- What type of testing do you enjoy and why? For example, exploratory testing, load testing.

Asking open-ended questions can foster a better conversation than asking for an unrealistic 5-year goal. Plus this approach starts to identify areas for improvement and strategies to bridge knowledge or skills.

How Much Do You Understand About the Product and Client's Usage?

To further define training needs, determine how much the tester understands about:

- The product under test.
- How the client uses the product.
- The type of decisions that client makes.

It may be helpful to create a product functionality checklist allowing him to rate skills to help determine training needs and approaches. This assessment allows the manager to identify testing assignments and perhaps adopt pair-testing allowing him to gain more knowledge by working with an experienced tester.

What are Your Strengths and Weaknesses?

Sometimes this is an easy question to answer as the tester may be aware of his capabilities and limitations. It can be helpful to create a list of strengths and weaknesses. Review the list of strengths to determine if you are capitalizing upon them. If the answer is no, what needs to change? Identify ways to bring those strengths into your testing.

Is there a weakness that might hinder career progression? A tester may be capitalizing upon his strengths but the weakness can be reducing promotional potential. For example, if he cannot present an answer on a bug report without becoming overly emotional can make him less influential in the company. Review the list of weaknesses and only work on the ones that can sidetrack a career. Everyone has weaknesses and it is not necessary to improve all of them. Be selective on what strengths and weaknesses you capitalize upon based upon the job and career aspirations.

How Can You Use a Job Description?

Review your job description to further understand expectations and if there is a potential of a higher-level position, review the requirements. Based upon this information, prepare a gap analysis to determine what you can do to improve skills and knowledge, and what requires a manager involvement.

Do You Need a Formal, Written Career Plan?

A career plan can be formally written defining goals and milestones to help a tester progress towards a specific job. An informal approach can be adopted by identifying testing skills and knowledge to improve. Some managers may use the company's annual review document to define goals to further a tester's career.

The format does not matter if it works. However, do not get into a trap of creating a formal document that contains goals or bullet points that do not lead to improving skills or career progression. All unnecessary administrative paperwork should be eliminated to spend time on conversations and making opportunities happen instead of updating documents.

What Tools Can You Use?

Reflection Journal:

A journal can be used to document what you have learned and areas for improvement. Every night write in the journal a lesson learned and something to do differently. For example, if a difficult conversation with a developer went well, identify why it went well to help you in the future. Perhaps you did more preparation for the meeting or were able to remain calm while discussing the testing results. Periodically review the journal to reflect upon the lessons learned to ensure they are being integrated into the appropriate projects.

Strength-Based Approach:

A strength-based approach, discussed in two popular books, can be adopted that provides both an assessment and program to follow. The book "Now, Discover Your Strengths" by Marcus Buckingham and Donald O. Clifton, Ph.D. has an Internet-based StrengthsFinder® Profile to identify your top 5 strengths. This book provides a brief overview of the strengths categories and how to use them for personal and professional development. Marcus Buckingham follow-up book "GO Put Your Strengths to Work" provides a short survey to determine how engaged your strengths are being utilized at the beginning and end of the program. A six-step, six-week plan helps a tester identify approaches to incorporate strengths into the work week and manage weaknesses that may be damaging.

A manager can use this approach to develop individual testers and to build a stronger testing team. If a manager does not adopt this approach, a tester can go through this program bringing back any lessons learned to his manager for further discussion.

Career Plan:

A career plan can include information such as goals, milestones, skill-gaps, knowledge-gaps, and training needs. The following website provides a sample career plan that can be modified. http://www.career-development-help.com/career-development-plan-template.html. Through a google search, additional career development templates can be located for more examples of written plans and assessments that provide in-depth approaches to assessing skills and career objectives.

How Can You Measure Progress?

It is important to periodically review a career plan to understand progression toward the goals and to determine if any corrective action is necessary. A career plan should be a living document that evolves with developing skills. Both the manager and the tester should review the plan prior to meeting in order to discuss progression and areas where more training or hands-on assignments are required. If a formal plan was written, it should be updated based upon the meeting and if a more informal approach or performance goals are used, summarize the meeting with an email.

Tips for Managers:

- Career development should be customized to individual testers based upon their skills, knowledge, and desire to accept more responsibility.
- Emphasis should be placed upon results and not developing a career plan document that is time-consuming to maintain or does not help progress skills.
- A one-size fits all approach does not work. Use different methods and do not mandate the same approach across all testers.
- Adopt a strength-based approach to ensure strengths are being utilized while weaknesses are being minimized when not important to success.

Tips for Testers Working With Managers on Career Development:

- A key aspect in any career development approach is the tester taking personal accountability to ensure skills and knowledge progresses regardless of a manager's involvement.
- Prepare for a meeting with your manager by reviewing suggestions discussed in this article to identify career goals, areas for improvements, and suggestions for bridging those gaps.
- Periodically review the plan for progress and any corrective action or changes based upon evolving skills and career aspirations.

Tips for Testers When a Manager Does Not Drive Career Development:

- A manager lack of involvement in career planning should never stop someone from furthering skills and knowledge.
- A tester can create a career plan by incorporating suggestions from this article. Once prepared, ask for a meeting with the manager to discuss your future.
- If the manager does not have time for a meeting, identify alternative approaches to gain training and project assignments. Try to start the process through email allowing the manager to respond when he has time. Other approaches include a working lunch meeting or a meeting scheduled before or after work hours which might work better for the manager.

Conclusion:

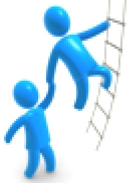
A manager should coach his employees to reach higher levels of performance through identifying and providing opportunities and tools to improve skills and knowledge. A tester needs to assume responsibility through capitalizing upon these assignments and taking ownership for career development. Employ an approach that makes sense whether it is a formal or informal career plan and remember any plan is a living document as it should continue to evolve. There are helpful tools such as the strength-based approach; however managers need to be careful on mandating methods across all testers as career development should be personalized. Regardless of the process, a tester needs to remember this is his career and it is important to take responsibility to ensure skills and knowledge continues to grow.

Bernice Niel Ruhland is a Software Testing Manager for a software development company with more than 20-years experience in testing strategies and execution; developing testing frameworks; performing data validation; and financial programming. To complete her Masters in Strategic Leadership, she conducted a research project on career development and onboarding strategies. She uses social media to connect with other testers to understand the testing approaches adopted by them to challenge her own testing skills and approaches.

The opinions of this article are her own and not reflective of the company she is employed with.

Bernice can be reached at:

LinkedIn: http://www.linkedin.com/in/bernicenielruhland
Twitter: bruhland2000
G+ and Facebook: Bernice Niel Ruhland



Back To Index

Software Performance Engineering :

implement it to avoid getting a Sucker Punch

by Samajeet Mohanty

Being my first article, I wanted to keep it simple and stick to the basics. In this abstract, I'll touch few pointers on a widely known, but not extensively used, methodology called *Software Performance Engineering* and try to explain its importance in the testing world. However, to begin with, let's look at few definitions.

Performance, by itself, means "The one thing that sums up all the hard work, time and thought put into a work of beauty & art [1]". Personally, if I apply the same conceptual meaning to software industry, it would come out as "*The one thing that sums up all the hard work, time and thought put into building & delivering an IT solution or IT Service that enables a Service Provider to stand out atop among the market competition*". ***Engineer***, on the other hand is regarded as someone who is a kick-ass uber-genius with godly math & science abilities [1] and who relates to things around him in a mathematical but socially inept way. Now, if we combine these two concepts into one and apply it to the software cosmos, the term we get introduced to is ***Software Performance Engineering (SPE)***. If I may, let's state it as *Sucker Punch Engineering*, in a good way.

Software Performance Engineering (SPE) is a systematic, quantitative approach to the cost-effective development of software systems to meet performance requirements. It is a software-oriented approach and focuses on architecture, design and implementation choices [2]. Before delving into SPE, let's briefly look at history for a bit.

The idea of Performance Engineering, in general, is as old as the automotive industry itself. The German manufactures, of say Porsche, very well knew the quality of vehicle they wanted to build and made sure that all necessary components were checked, re-checked and then checked again to ensure a high quality delivery. With the era of Internet revolution, one realized the power of BEING ONLINE. Soon enough, service providers tapped into the potential of reaching to their daily customers without being face-2-face. 1000's of websites were being built each day and end-users also felt it easier to deal with transactions from the comfort of their home. What the service providers failed to realize is that the online crowd at the other end of spectrum was growing exponentially and expecting the same level of online / website responsiveness, per se. Sadly, nobody thought of applying the same principle of Performance Engineering in their IT services.

Though much has evolved by now in 20th century, IT industries are still experiencing application related Performance issues every day and losing billions of dollars. This clearly indicates the lack of appropriate Performance Engineering strategy in place. As a real world example, one of my clients mandated an organization-wide Performance Engineering policy only 3 years ago, in spite of being in business for more than 20 years. However, they are still doing performance testing and not true performance engineering. In over 4 years of experience as a Performance Engineering Consultant, many project managers have clearly stated to me that they never even thought of introducing a performance test phase, let alone performance engineering, into their product Software Test Life Cycle (STLC) until end users reported performance issues after their application was LIVE in production.

What happened next ? Well !!

1. Loss of revenue
2. Loss of customers
3. Loss of productivity / effort already invested
4. And most importantly, Damage to the brand

Above list is only inclusive and not exhaustive.

Couple of reasons why project teams / higher management put the factor of 'performance' at backseat during STLC or even SDLC are:

1. Inadequate knowledge about Performance Engineering
2. Lower supply of competent Performance Engineers out there.

The projects stakeholders seem to believe that their newly built application can easily be supported on the hardware and software combination they have in place and hence, they really need to focus on the Functional piece of it. However, what they don't realize is that there are just too many factors like 3rd party components, other application dependencies, hardware limitations and so on that can affect their application's performance at any point in time. This is where a Performance Engineer needs to step in and educate the project team. S/He needs to clearly make the team understand the risks involved and their impact with every permutation & combination of each of these possible factors with clear mathematical justification. Only when explained in monetary terms, the project stakeholders consider this as a matter of concern and take necessary steps.

At this point, as a reader of this article you might ask me:

1. What's the difference between Performance Testing and Engineering phases?
2. When should I implement Performance Engineering
3. And, its advantages

Now, the answer to all these questions deserves a separate article in itself which I'll do on a later date. However, for greater good of the reader community, I'll touch all the high notes here in the form of few abstract bullets. (**PE**, if used, will refer to *Performance engineering* in subsequent sections)

- PE phase enables the project team and in turn the testing team to achieve a higher level of quality for the delivered product.
- The processes involved in PE should be implemented during every phase of Software Development Life Cycle right from Requirements Gathering, Design, Coding, Testing till Maintenance steps. It is imperative that the project team engages an experienced performance engineer during each of these phases.
- Performance testing is ONLY a subset in the entire engineering methodology. It is generally performed in testing phase when the application is in User Acceptance Test cycle. It is important that the application has been signed-off by Functional testing team before it is simulated for performance. I'll cover few variations of performance tests like Load Test, Volume Test, Stress Test, and Endurance Test and so on with subtle but important differences in my upcoming columns.
- Implementing PE, in the right way, lets you build High Availability infrastructure with minimal downtime for all your applications.
- PE helps reduce the propagation of bugs throughout the SDLC phases.
- PE enables quick turn-around time for resolution of bugs identified during the testing phase.
- PE helps project team plan and manage future application capacity expectations from a hardware and software perspective.
- Following performance engineering methodology enables an organization to develop & follow a sturdy Quality Control / Assurance policy.
- Performance Engineering rather than Performance Testing is considered as an "Art and Science" because of its contribution to SDLC with the help of all mathematical goodness.

In conclusion, coming back to my earlier definition of **SPE**, it needs to be both Qualitative & Quantitative and applied at each phase of SDLC unlike Performance Testing which is done few weeks before the IT service is scheduled to go LIVE. If SPE methodology is not followed, it's just a matter of time before the project stakeholders get sucker punched by their customers.

References:

[1] → http://www.urbandictionary.com

[2] → http://www.perfeng.com/

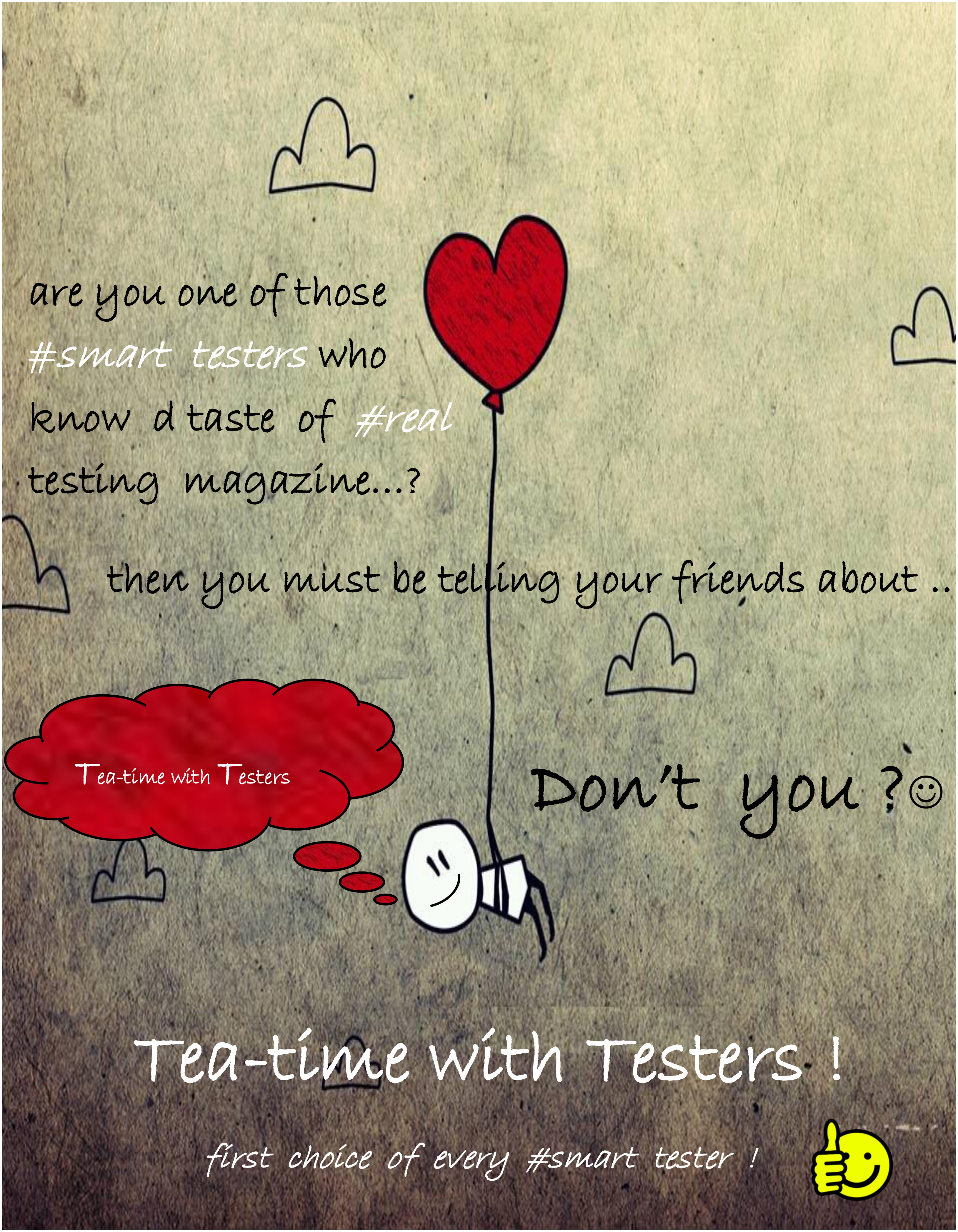
Back To Index

Samarjeet Mohanty (Samar) is a, self-proclaimed, practitioner of anything to do with Software Performance Engineering world.

He's quite experienced in Performance Engineering and Testing methodologies of software applications (BFSI) using industry standard tools and techniques. Apart from being an active participant in technical forum's concerning performance and generic testing QA, Samar likes to interact with creative-minded professionals from all walks of life to better understand their take in the related field.

If interested, connect on:

LinkedIn: http://ca.linkedin.com/in/samarjeetm
Twitter: http://twitter.com/SamarjeetM



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

then you must be telling your friends about ..

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**

Stop being a NON-Technical Tester!

Some days ago I posted an open question on twitter:

Do testers need to be as technically good as a programmer to be successful at their jobs?"

I got plenty of answers, so I will only post some of them representing the main opinions threads:

@TestAndAnalysis - Testing is a technical discipline that is different to programming and testers add a lot of value to projects

@huibschoots – No! It depends on the context they work in. But in general they need to have some basic technical skills (or the will to learn)

@datoon83 – I'd say that all people involved in the delivery need to talk 1 language – that of the domain and customers!

@klyr – Technical and non-technical testers have a different approach to testing, and will find different bugs.

@sgershon – Certainly do. Not sure about "more/less technical" as programmers, possibly they have to be "differently technical" than them.

@halperinko – @sgershon @joelmonte I normally look at it as in depth knowledge for devs, vs. System-wise knowledge for testers. (Some don't follow rules)

There were also a couple of tweeps who replied with blog posts of their own sharing their opinion on the subject.

@diamontip – my answer – do tester need to be as technical as programmers to be successful at their jobs? bit.ly/v5vcX

And

@adampknight – I personally dislike calling testers "technical" wrote about it here bit.ly/tVHDOB

To all the tweeps above and those I missed, thanks for the great feedback!

But as you correctly guessed I have my own opinion on the subject and I want to share it with you, so here it goes…

My definition of being "Technical"

Specially after reading Adam's post I feel the need to explain what do I mean by technical, or better yet how do I differentiate a Technical Tester from a Non-Technical Tester.

(If you read my previous blogs on "Why are some tester not really Professional Testers" then you should already have an idea…)

A Technical Tester is not afraid of doing most of the following stuff on a regular basis as part of his job (without any specific order):

- **Understand the architecture of the product he is testing**, including the pros & cons of the specific design, as well as the risks linked to each of the components and interfaces in the product.

He then uses this information to plan his testing strategy, to execute his tests and find the hidden issues, and also to provide visibility to his team regarding the risks involved in developing a specific feature or making a given change to the system.

- Review the code he needs to test.

He can do this on a number of levels, starting from going only over the names of the files that were changed, and all the way to reviewing the code itself. This information will provide valuable inputs to help decide what needs to be tested and how, as well as to find things about the changes that might have been missed by the developer or the documentation.

BTW, by code I mean SQL queries, scripts, configuration files, etc.

- Work with scripts & tools to help his work.

A technical tester should be able to create (or at least "play") with scripts to help him run repetitive tests such as sanity or smoke, and tasks such as configurations, installation, setups, etc.

He should also be able to work with free automation tools such as Selenium or WATIR (or any of the paid ones like QTP, SeeTest, TestComplete, etc) to create and run test scripts that will increase the stability of the product in development, and over time save time…

- Be up to date with the technical aspects of his infrastructure (e.g. browsers, databases, languages,etc)
He should read the latest updates on all aspects of his infrastructure that may have an effect on his work. For example new updates to his O/S matrix, known issues with the browsers supported by his product, updates to external products they integrate, etc.

With the help of Google alerts and by subscribing to a couple of newsletters anyone can do this by reading 5 to 10 email 2 or 3 times a week. The value gained from becoming an independent source of knowledge greatly exceeds the time invested in the efforts.

- Is able to troubleshoot issues from Logs or other System Feeds.

He is aware of all the logs and feeds available in his system, and uses them to investigate more about any issue or strange behavior.

This information is helpful during testing to provide more information than simply writing "there is a bug with functionality X". And it will be critical if he is called to work on a customer bug, where he needs to understand complex issues quickly and without access to all the information.

In addition to the above, a technical tester should also be able to:

- Provide feedback and run the **unit tests** created by his programmer peers.

- Run **SQL Queries** on the DB directly to help verify his testing results.

- **Install and configure** the system he is testing. etc.

Sounds like Superman or MacGyver?

It may sound like this, but actually it's not!

As testers we work on projects that revolve around Software, Hardware, and/or Embedded products. The only way to do a good job in testing them is to have a deep understanding of both angles: technical and functional.

This doesn't mean that you need replace or have the same technical dept as your developers, or surpass your Product Marketing's knowledge of your users.

You need to achieve a balance, where you have "enough" knowledge and understanding of both these areas in order to do your job as a tester, or rephrasing the tweet from @halperinko – as testers we should achieve system-wide knowledge, as opposed to the in-dept knowledge required from the developers or the product marketing guys.

Is it black and white?

This time quoting "Cokolwiek" who commented on one of my latest posts, "Not everything is black and white". Meaning there is no standard to define how technical should a tester be on every project and product.

Like in many other situations, the best answer to how technical do you need to be is: "It Depends…"

You should be at least technical enough to do your job effectively and to talk the same language with the rest of your programming and testing peers.

What do I mean by that?

If you work on a software development firm then you should understand enough of the languages used by your developers to be able to read the code and understand their changes. If you work on a heavily DB-related project then you need to understand enough of SQL and database management. If you work on a Website development firm then you should know enough CSS, HTML and JS, and so it goes…

So if I am not Technical enough, should I quit testing???

Definitely not!

If you like testing and you are good at it, why should you quit? On the other hand, this is a great opportunity to improve your work and (as @diamontip wrote on his blog) increase your market value as a tester 😉

And the best part of it is… it's not very hard to become a more technical tester! Just start by asking questions, searching the web, reading books, etc.

I'm also confident that if you show the potential to increase the value of your current work to your manager, he won't mind you investing sometime during your day to learn these new trades (as long as you manage to explain why this is also good for him!).

So, stop making excuses for not been technical enough. Just make a decision (or a New Year's resolution…) to start working on improving your technical skills!

Go ahead and get rid of the NON-Technical Tester in you!

It will be worth your time and make your job more interesting and satisfying.

What do you think?

- Are there any negative sides to being a more technical tester?
- Maybe there are advantages for testers who used to be developers in the past?

Come and share your opinion or experience on this subject!

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

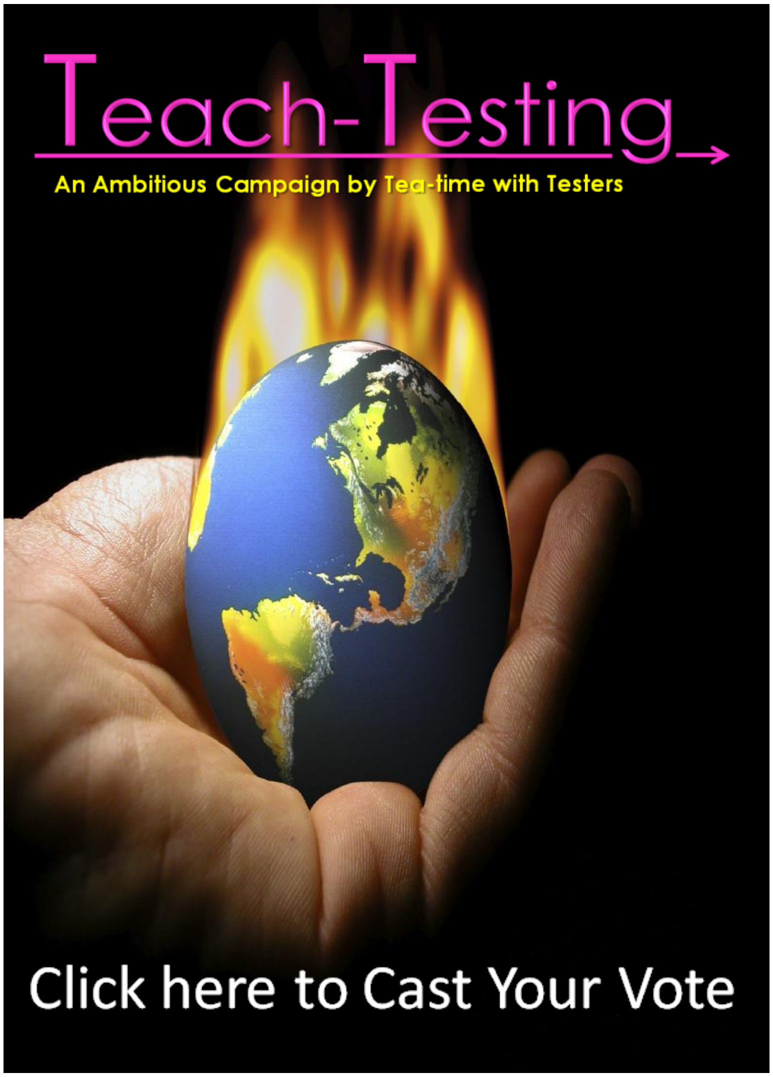
He's worked in companies ranging from small Internet Start-Ups and all the way to large multinational corporations, including Mercury Interactive (currently HP Software) where he managed the QA for TestDirector/Quality Center, QTP, WinRunner, and additional products in the Testing Area.

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

Back To Index


Teach-Testing
An Ambitious Campaign by Tea-time with Testers
Click here to Cast Your Vote

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 !

sciencephotogallery

Single Entry, Single Exit - Singular Purpose

Functional automation is seen as a key enabler for good testing, especially at a product/application level. Significant money, time, effort are spent to ensure we can do faster, cheaper and better by leveraging technology to do more. This is usually about choosing tools, devising architecture/framework and then cranking out automation code.

And then we discover the ice-berg; that we have to change code to be sync with evolving software i.e. we need non-trivial investment to modify automated scripts and ouch it is a pain! So, why does it become a pain? Because of the complexity of script? So what contributes to the complexity? The variety of conditions in the script to check the variety of behaviors.

As a consultant when I examine existing automation scripts, I am always amazed at how much some of the automation code tries to accomplish. In a recent engagement while examining the automation code of a product with rich browser-based GUI, I discovered that the automated script checks for data boundaries, default values of inputs, control enabling/disabling, control properties, and then functional behavior. I am sure that you can visualize the various conditions to check for and the associated long code that is rich in potentially uncovering a variety of issues. The developer of the script is proud of how much he/she has packed into the script whilst I am smiling and cringing inwardly. The "richness" is the problem!

We as human beings have an amazing ability of dealing with very many things at the same time. Replicating this in a machine is not easy. It is necessary to break a complex activity into multiple simpler activities so that a "machine/automaton can chew this". It is necessary to assess if an entity is in a state that it is fit enough to be delegated to a "machine or automaton". What does this involve? The act of simplification of the activity list by making activities "single minded" to reduce complexity.

So where are we going with this? Reduce the complexity of functional scenarios that are being automated by decomposing this into smaller single-minded scenarios. Taking the above browser-based product as an example, we can see that this is a collection of small scripts (1) for only checking data boundaries (2) to check the control properties (3) third for checking functional behavior and so on. What are we doing? Being single minded - (A) Writing scripts that are focused on uncovering one (or very few) types of potential defects (B) Ensuring that each scenario has only one kind of behavior i.e one type of output.

Voila, this means that script has minimal conditions in the code and hence it is more or less straight line code. Wait a minute... Does this not mean that the Cyclomatic complexity of the functional automation code is very low? And therefore easier to change/maintain? YES, you got it. Good development is about writing least amount of less complex code. Note that automation is development, hence good principles of development apply here too.

Single-entry, Single exit code i.e. code with ideally no conditions is the least complex code. Code with least complexity is less prone to bugs and is easily maintainable. A good developer lowers complexity by re-factoring code. The same thing an automation developer should adopt.

So as a script developer, what do I have to do? Assess the "fitness of automation" of scenarios that you have to automate. That is, simplify a scenario if it is attempting to do many things. Simplify it by decomposing it into various levels with each level focusing on a specific type of defect. For example a Level -1 script could focus on only checking on input correctness, while Level-2 script could check for interface correctness and Level-3 for functional correctness. In HBT (Hypothesis Based Testing), this is called "Fitness for Automation" with scenarios being "level-ized". This helps in making the scenario "chewable" by the "automation".

We now have scenarios that are single minded i.e. focused on finding a potential defect type, a scenario defined and understood as a singular behavior resulting in an output. When we convert these into automated scripts, we have a "single entry-single exit" script. This is a script with the least set of conditions (ideally zero) and therefore has "singular purpose".

It is often assumed that a good framework is a good enough for building flexible scripts. This is true, but represents only half the story. The other important half is the structure of a scenario and its "fitness for automation". A good structure wherein a scenario has a "singular purpose" enables the script to be "single entry-single exit code" i.e. least complex code.

As we saw in the Dec 2011 column ('Taming complexity') complexity is an enemy and friend. An enemy because it can faze you, freeze your thoughts and numb you. A friend because it depicts richness, the ability to do very many things. The trick is to be rich (level-ized) but not be fazed by it (singular).

A clear goal focus sharpens intellect. A scenario with a clear goal results in a "singular purpose" script that satisfies of the key attribute of "single entry-single exit" to enable you evolve with least pain.

So the next time you convert a scenario into an automated script, make the scenario "chewable" or be prepared to be "eaten".

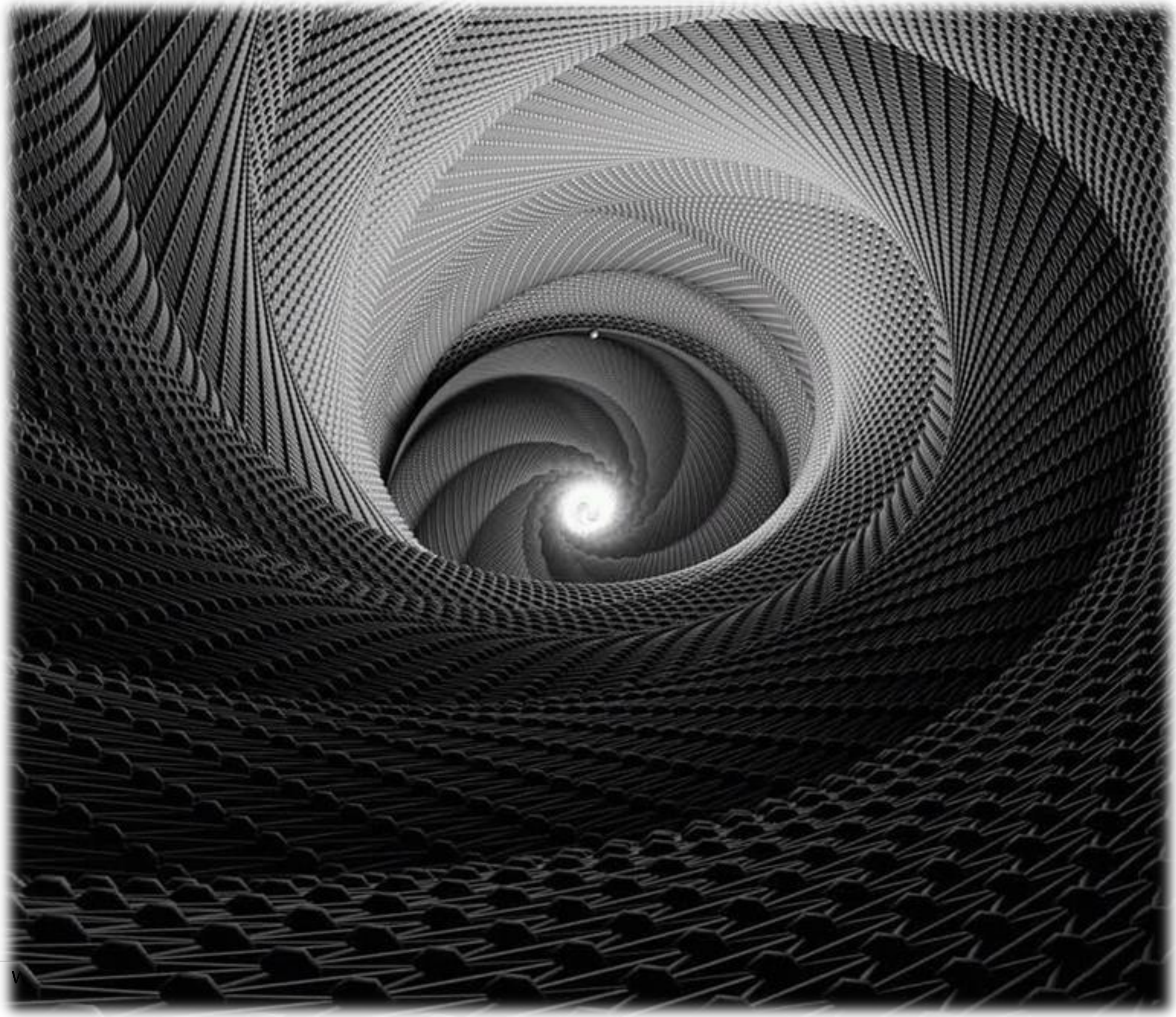
Have a great 2012.

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** .

Back To Index



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.

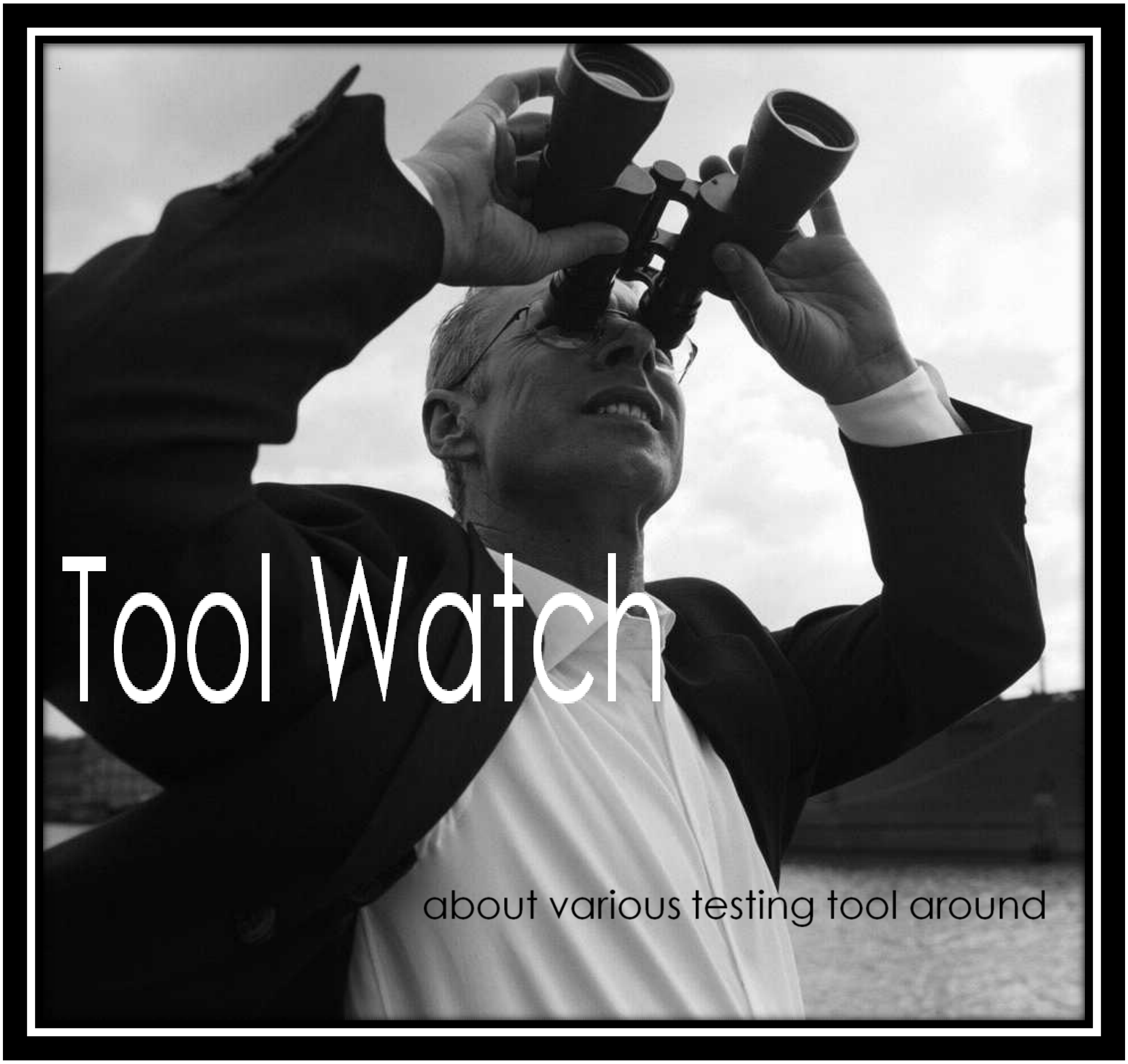
Quality Testing also provides daily Polls and sample tests for certification exams, to make tester to think, practice and get appropriate aid.

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.



Tool Watch

about various testing tool around

Rapid Reporter

PART 2

By Chris Philip

Session Debriefs

After the application is closed the session notes are ready to be reviewed. You will find all the files generated during the sessions at the session folder: *.CSV, *.JPG, *.RTF, *.HTM. The session folder will be the folder from which Rapid Reporter was executed, unless you pick a different folder (*See the Change the working directory or folders section for details on how it's done*).

One way to look at and review the session is by opening the appropriate *.CSV file (*you can find it according to the creation date or file name*) in a spreadsheet. By viewing it through a spreadsheet you can easily filter and look at one single type of note (*bugs, questions, ideas for next charter…*) or change the order the data is being shown (*you can always return back to chronologic order*). A spreadsheet also allows getting numbers like amount of sessions and bugs found etc.

Another way to look at the session is generating an HTML report. These are very useful because they include screenshots embedded – but at this moment do not present filtering or ordering options.

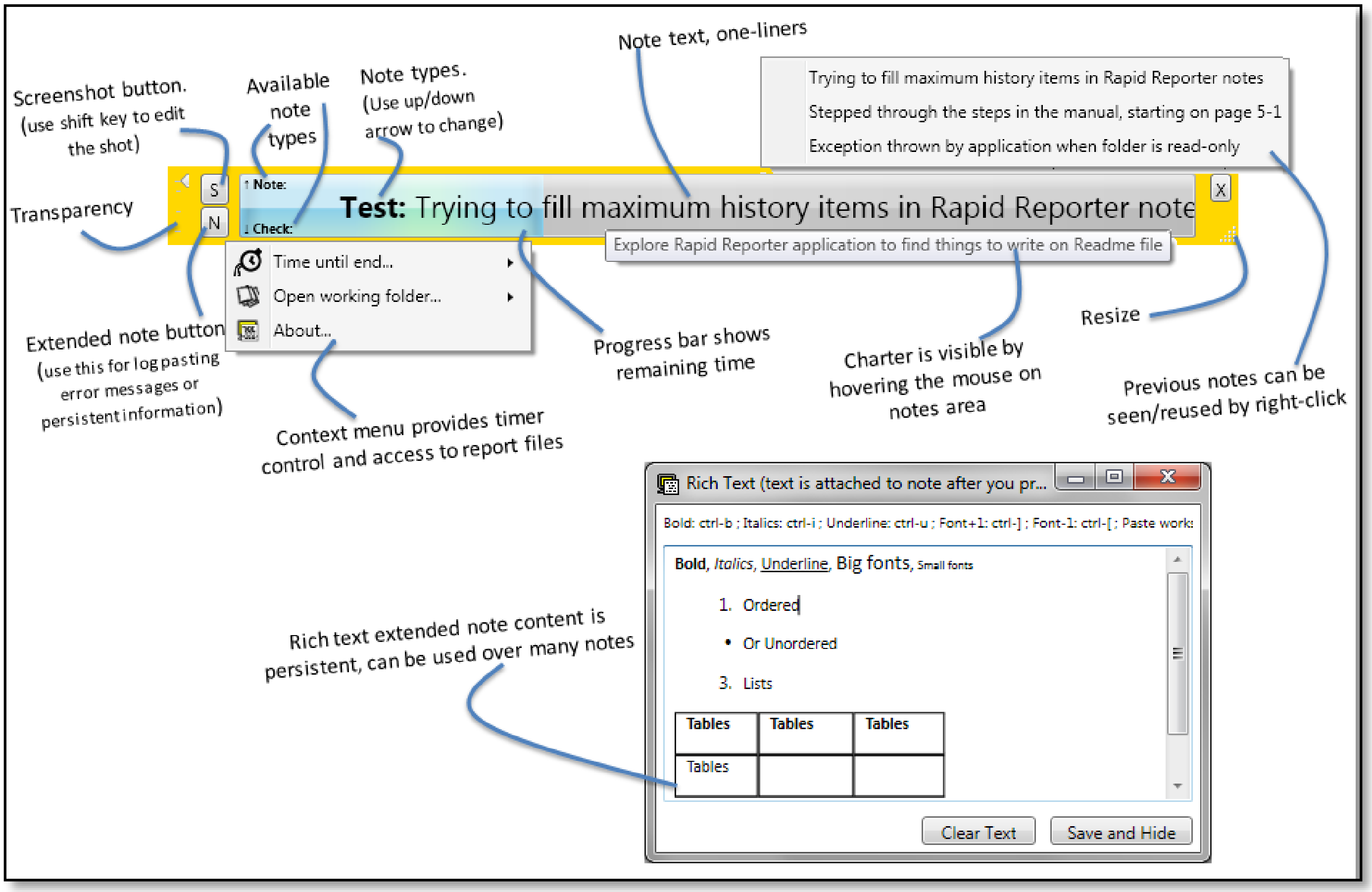
In order to generate an HTML report, you have to run Rapid Reporter from command line:

```
C:\> RapidReporter.exe –tohtml <nameofsessionfile.csv>
```

Visual Guide

Some people prefer learning with visual clues.

Here's a summary of the different areas in Rapid Reporter's graphic user interface:



Screenshot button. (use shift key to edit the shot)

Available note types

Note types. (Use up/down arrow to change)

Note text, one-liners

Transparency

Extended note button (use this for log pasting error messages or persistent information)

Context menu provides timer control and access to report files

Progress bar shows remaining time

Charter is visible by hovering the mouse on notes area

Resize

Previous notes can be seen/reused by right-click

Rich text extended note content is persistent, can be used over many notes

Trying to fill maximum history items in Rapid Reporter notes

Stepped through the steps in the manual, starting on page 5-1

Exception thrown by application when folder is read-only

↑ Note:

Test: Trying to fill maximum history items in Rapid Reporter note

↓ Check:

Explore Rapid Reporter application to find things to write on Readme file

Rich Text (text is attached to note after you pr...

Bold: ctrl-b ; Italics: ctrl-i ; Underline: ctrl-u ; Font+1: ctrl-] ; Font-1: ctrl-[; Paste work:

Bold, *Italics*, Underline, Big fonts, Small fonts

1. Ordered

- Or Unordered

3. Lists

Tables	**Tables**	**Tables**
Tables		

Detailed Usage: How to

❖ Enter a note

1. Write your note in the note area (*the big blank space in the middle of the app*)

2. The note is saved into the session when the enter key is pressed.

3. Press the up or down arrow to iterate between the note types. This can be done before the actual note is written or while it is being written. A note type cannot be changed from within Rapid Reporter after the enter key is pressed.

❖ Attach screenshots or extended notes

1. Attachments can be added to a note before the note they relate to is written or while it is being written. A note attachment cannot be changed from within Rapid Reporter after the enter key is pressed.

2. Press any of the two attachment buttons to take a screenshot or to add an extended note.

Screenshots:

a. By pressing the button, the whole screen is saved and marked as attachment for the next note.
b. Pressing the button while holding the SHIFT key will open the screenshot for editing in MS Paint. This is useful when you are interested in adding a note or cropping part of the screenshot.

Extended notes:

a. These notes can be used to add logs info, error messages, pictures and etc. Use it also for taking notes that persist for a while in the session (*the content is not deleted when the note attachment is saved*).
b. Use the keyboard shortcuts for formatting: Ctrl-B, ctrl-I, ctrl-U… Advanced formats (*like tables*) are supported but need to be copied from a different text editor at this moment .

❖ Edit a note

Notes cannot be edited from within Rapid Reporter's interface.

My personal experience shows that after a note is entered, there is no much reason to edit it or fix it. In fact, if there is a reason to fix the note, this would be important material for the session debrief, so mistakes and confusions in notes are worth being kept.

Nevertheless, notes can be edited by following these steps:

1. Pressing the "Open working folder" in the context menu will show all the session files

2. Locate the appropriate *.CSV file and edit it on notepad or a spreadsheet (*note that spreadsheet apps may lock the file from Rapid Reporter's use*).

❖ Change the working directory or folder

By default, Rapid Reporter will use the directory from which it was executed for all his file manipulations (*saving the session and the attachment, or making consolidated reports or HTML docs*).

This directory can be modified in one of two ways, allowing you to have one copy of Rapid Reporter yet work on different session paths:

1. Graphic interface

a. Before the session starts (*i.e. before the charter is entered*) Rapid Reporter context menu allows you to modify the working directory

b. After the session starts, all that can be done from the context menu is opening the directory. No path changes are allowed during the session (*after the charter is entered*).

2. Command line interface

a. Rapid Reporter has a command line argument called –directory that let's you change the directory used for a session:

```
C:\> RapidReporter.exe –directory <path>
```

b. This will work for testing session, for HTML docs, for consolidation reports…

```
C:\> RapidReporter.exe –directory <path> -report
C:\> RapidReporter.exe –directory <path> –tohtml <file.csv>
```

❖ Use Rapid Reporter with different terminology or note types

Rapid Reporter starts with the following note types by default:

Setup / Note / Test / Check / Bug / Question / Next Time

Some people would like less, or more, types.

Other people would like to use the terminology used at their workplace (*Defect instead of Bug, or Segment instead of Setup*).

Other people use other languages and would like to translate the types (*Pergunta instead of Question, Notitie instead of Note*)

1. To change the note types, start Rapid Reporter from command line. Any word added after the executable name will be used as a note type. For example:

```
C:\> RapidReporter.exe Nota Prueba Pregunta Proxima
C:\> RapidReporter.exe Information Question Lookup
```

to be continued in next issue…

Back To Index

Do you think that even your own tool should be part of this unique section?*

Feel free to write us. Let the world know what your Tool can do !

To know more write to us at **teatimewithtesters@gmail.com**

* Conditions Apply

Chris Philip has been working with his first employer TCS since 2 years, passionately in area of Automation Testing. The passion and interest in automation was revealed during his college projects in robotics and successful completion of project work from India Space Research Organization, automating the microscopic and sensitive calculations regarding the minute changes in accelerometer data in launch vehicles, rockets and spacecrafts. The project was done in microprocessor programming language.

Chris is active member of Linux club, IEEE and Computer Society of India (CSI).

His special interests are software automation, having extensive hands-on experience in QTP, Sahi, Selenium and RFT.

Actively participates in blogs and discussions related to automation practices. He has presented 3 white papers till date about the automation practices, short cuts and interesting logics applicable in Quick Test Professional.

Chris is reachable at **christhomsonphilip@gmail.com** & on twitter **@chris_cruizer**

by

QUALITY TESTING
Quality is delighting customers

Testing PUZZLES by Sebi

Claim your Smart Tester of The Month Award. Send us an answer for the Puzzle and Crossword bellow b4 15th Feb 2012 & grab your Title.

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

Gear up guys.......

It's Time To Tease your Testing Bone

Puzzle “Find the Prime”

"What is the biggest number that is validated in http://www.testalways.com/1/ as being prime?"

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

Back To Index

TESTING CROSSWORD

1		2		3		4			5
	■		■		■		■	■	
6							7		
	■		■		■		■	■	
8					■		■	■	
	■		■	■	9				
10				■		■	11	■	
	■		■	■		■		■	
■	12						13		
■		■	■	■		■		■	

Horizontal:

1. It is an open source (tool) project aiming to structure and industrialize functional testing activities (6)

4. It is a testing performed outside an organisation at client place (4)

6. It is tool uses a highly graphic interface making Load Testing Fun and Fast (6)

7. It is a testing in which all branches in the program source code are tested at least once (3)

8. It is a tool to perform repeatable tasks that help managers, architects, developers and testers to test an application against its performance (5)

9. It is a interface between a host machine and a Controller in Loadrunner Tool (5)

10. Testing of individual software components is called ______testing (4)

12. Use of symbols, rather than numbers, combined with rules-of-thumb, in order to process information and solve problems. It is called _______Processing (8)

Vertical:

1. It is a tool for browser-based testing of web applications (8)

2. Testing the ease with which users can learn and use a product (9)

3. A special-purpose implementation of a software module, used to develop or test a component that calls or is otherwise (4)

4. It is the actual group of code that will be deployed to a test environment (5)

5. It is a final stage of testing that is performed on a system prior to the system being delivered to a live environment (10)

9. It is a testing phase where the tester tries to break the system by randomly executing the system's functionality (5)

11. Short form of Application Programming Interface (3)

12. A test case design technique for a component or system in which test case design is based upon the syntax of the input. It is called ____________ (in short form) (2)

13. Testing whether the software or system installation being tested meets predefined installation requirements. It is called ______ testing (in short form) (2)

Answers for Last Month’s Crossword:

T	E	S	T	I	N	G	B	O	T
■	■	E	■	T	■	T	■	■	E
B	R	E	A	D	T	H	■	■	S
L	■	T	■	T	■	A	R	C	T
E	■	E	■	■	■	R	■	■	D
R	E	S	U	L	T	N	1	■	A
B	■	T	■	C	■	E	■	■	T
Y	E	T	I	S	■	S	J	E	A
■	L	■	■	A	■	S	■	■	■
U	N	I	T	J	U	B	U	L	A

We appreciate that you

“LIKE” US !

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 ?

Image : vernhart

Our Testimonials

I got to know about your magazine from my colleague and to my surprise it's really wonderful.

I love solving your testing puzzle and crossword as well.

- Tara Sharma

Hi,

I am a software programmer working in some MNC. Test Manager at my workplace is your big fan and he recommended your magazine to us as well. I was little doubtful earlier but as I went through your issues I realized that our Test Manager was RIGHT!

"Tea-time with Testers" is the magazine that every tester, programmer and manager should read.

- Benjamin D.

I LOVE "Tea-time with Testers" !

- Yamuna Kashyap

www.TeaTimewithTesters.com is and awesome magazine and so do your Teach-Testing Campaign.!

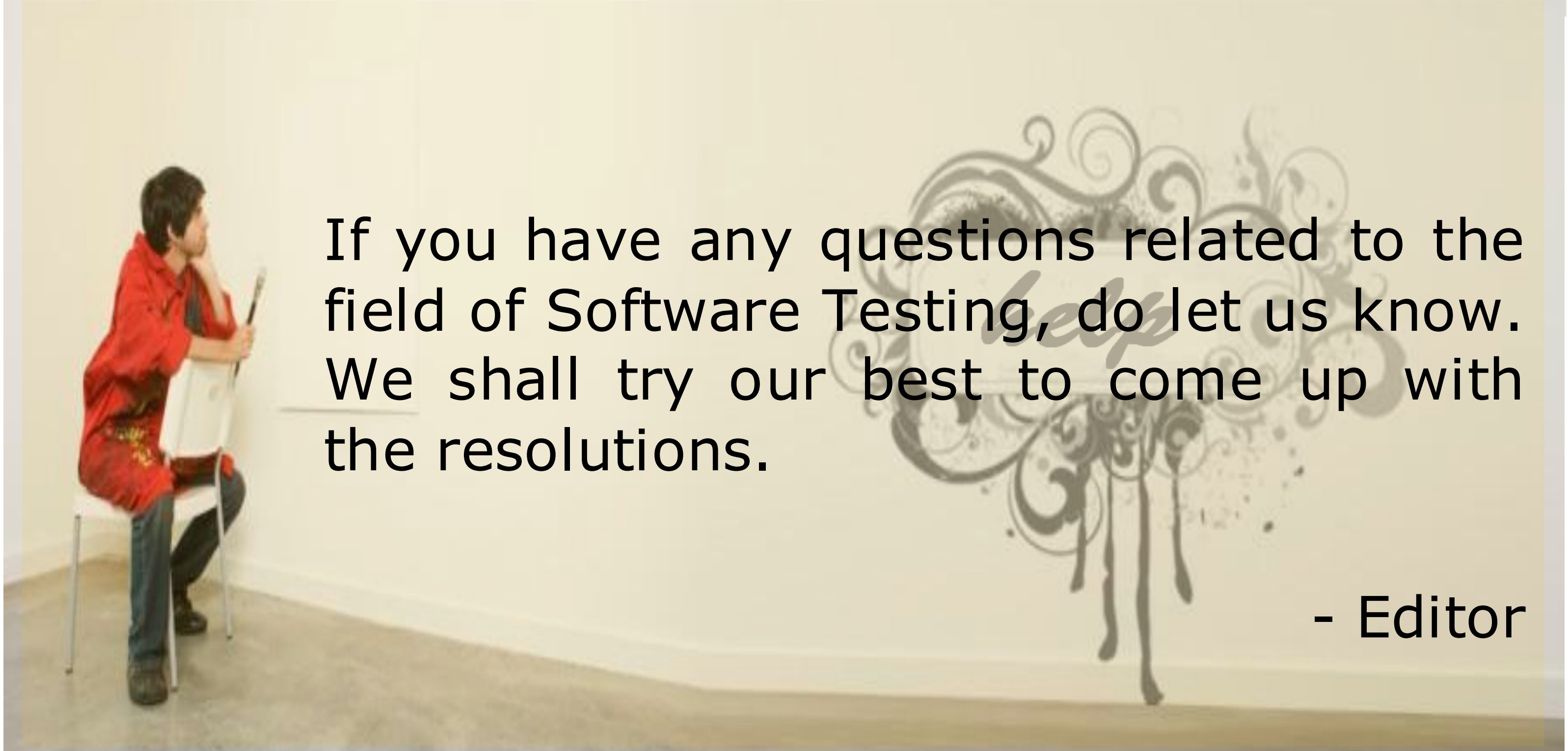
- Manjula Ghuge

Mindblowing Magazine !

– Jonathan

You ask...

We'll help...!



If you have any questions related to the field of Software Testing, do let us know. We shall try our best to come up with the resolutions.

- Editor

Feel free to write us your expectations.
Help us to help you better.

We are just a mail away: teatimewithtesters@gmail.com

in nexxt issue

articles by -

Jerry Weinberg

T Ashok

Joel Montvelisky

Anurag Khode

Mike Talks

Bernice Ruhland

Anne-Marie Charrett

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

Cover Page Image- Roses and Teacups

Core Team:

Kavitha Deepak (Bristol, United Kingdom)
Debjani Roy (Didcot, United Kingdom)
Anurag Khode (Nagpur, India)

Anurag

Kavitha

Debjani

Testing Puzzle & Online Collaboration:

Eusebiu Blindu (Brno , Czech Republic)

Romil Gupta (Pune, India)

Eusebiu

Romil

Tech -Team:

Subhodip Biswas (Mumbai, India)
Chris Philip (Mumbai, India)
Gautam Das (Mumbai, India)

Subhodip

Chris

Gautam

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

To get **FREE** copy ,

Subscribe to our group at

Google

Join our community on

facebook

Follow us on

www.teatimewithtesters.com

Give Feedback