

Tea-time with Testers

AUGUST 2012 | YEAR 2 ISSUE VII

Jerry Weinberg
The Fish-eye Lens

Michael Bolton
Why is Testing Taking so long? – part 2

Joe Demeyer
The Credible Tester

Bernice Ruhland
Importance of Social Networking

Joel Montvelisky
Stop Making Testing Complicated !

T Ashok
Agile Sutra

Anurag Khode
Mobile Application Testing



The KIT Is Koming!

TESTKIT CONFERENCE 2012

Koming!

stKIT

**& Test Automation
Conference**

The Date
October 15 - 17, 2012
BWI Airport Marriott
Linthicum, MD

The KIT is Koming!

TestKIT

Testing & Test Automation

Conference

Save The Date

October 15-17, 2012

BWI Airport Marriott

Linthicum, MD

www.testkitconference.com

Register Now!

OCTOBER 15 - 17, 2012
BWI AIRPORT MARRIOTT
LINTHICUM, MD

- Tutorials ■
- Concurrent Presentations ■
- TABOK Certification Training & Exam ■
- Discussion Forums & Networking ■
- 4th ATI Automation Honors ■
- Keynotes ■



www.testkitconference.com

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.***

Editorial

Wake up Sid !

If you are from India you must have watched this movie, haven't you? Let me brief about it to our non-Indian readers.

'Wake up Sid' is a movie that revolves around *Sid Mehra*, a rich but careless college student. Tired with his careless and irresponsible attitude towards life, his father throws him out of home. With nowhere to go, Sid (who has never been on his own) asks Aisha (his friend) if he can stay with her. Aisha happily accepts it. Eventually, Sid's bad habits revive themselves, as he leaves her place a mess and throws tantrums. Over time, Sid learns that to care for himself, he has to begin cooking and cleaning. He also realizes that he must work, and finally with Aisha's help he joins one magazine as a photography intern.

Well, you must be wondering why am I telling this, don't you? I am telling this because I see a lot of 'Sid's in the field of software testing who are not yet serious about their learning, about acquiring new skills and basically about becoming great at testing. It is very sad that despite of so much of quality stuff available over internet (that too for free) some testers are still unaware of it. Some of them are still writing (detailed) scripted test cases and have no idea of what magic 'mind-mapping techniques' can do. Sadly enough, some are still trying to find the *best* answer for 'Difference between Sanity Test and a Smoke Test'.

Come on guys! It's a high time to think beyond. I know that things won't change overnight but have you at-least given it a start? Well, Scenario is changing fast. Those who have made it, they are already ruling the market. You don't want to get thrown out of competition, do you?

Gear up! You are already late if you haven't started!

On side note, not every Sid gets Aisha's help. That happens only in movies. 😊

Enjoy your issue!

Sincerely Yours,



- **Lalitkumar Bhamare**

editor@teatimewithtesters.com



QuickLook



Editorial

What's making News?

Tea & Testing with Jerry Weinberg

Speaking Tester's Mind

Why is Testing Taking so Long? - 18

The Credible Tester- 26

In the School of Testing

Understanding Browser Compatibility Strategies - 33

Mobile Application Testing-36

Stop Making Testing Complicated-41

T' Talks

Agile Sutra - "Sensitize & Prevent", not "Design & Execute"-46

Testing Puzzle – S.T.O.M. Contest

Our Testimonials

Family de Tea-time with Testers

Testing Puzzles
by Sebi



Crossword
by



What's making News?

- find out the latest happenings in the technology world

TuniTeam launches a FREE trial of the intuitive test management tool : myTestingBank

TuniTeam (www.tuniteam.com) has launched **myTesting Bank**, the test management tool for managing the QA / Validation activities and is offering free trials of this intuitive new test management tool.

myTestingBank enhances the productivity of testers and improves the visibility of the progress of test execution. The cloud based option allows clients, third parties and off shore teams quick and simple access to test scripts, results and reports. myTestingBank offers an alternative to MS Excel / MS Word for more effective management of test plans and test runs.

myTestingBank has been designed for software test teams to efficiently create and update test cases and test runs.

This tool consists of five sections:

1. Projects 2. Requirements 3. Test suites 4. Test run 5. Administration.



Bassem ABID, the product manager says, "We are confident that it's the right time for myTestingBank to be in the market. We have taken more than two years of hard work and we are sure that it'll answers major needs of QA activities. This tool is intuitive, simple and strong."



The test life cycle is as follows: Create project > Enter requirement specifications> Write test suite > Execute test cases > Raise new bugs > Report test result.

myTestingBank is a flexible tool allowing tester to import test suites and requirements from an Excel file to download without having to re-enter them into the application.

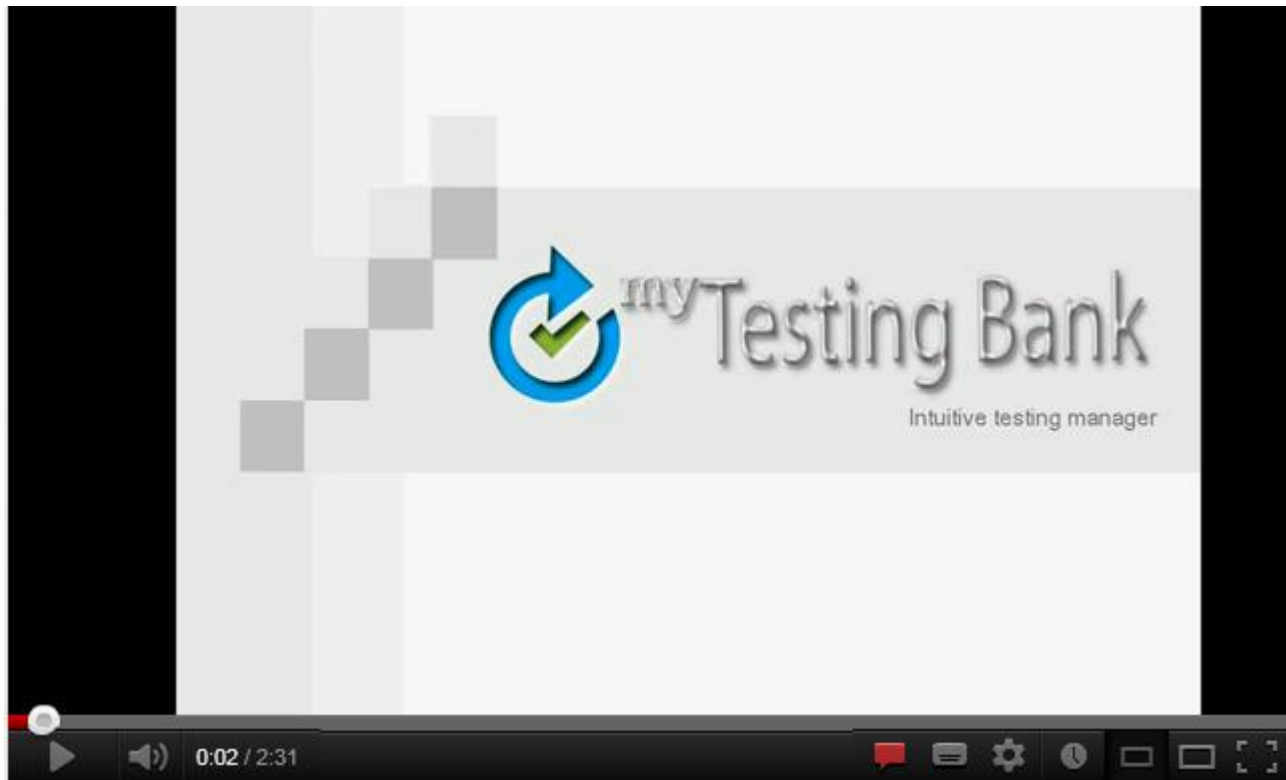
It's also possible to export test suites, requirements and test runs for external use.

The functions "import" and "export" are very useful to retrieve existing test suites and requirements as well as to communicate the test data in MS Excel format.

This tool helps management gain an understanding of the stability of the product under test by providing quick graphical overviews on the progress of testing. With myTestingBank, it's also possible to gather reports and statistics for all the test results.

This month TuniTeam is launching a *Beta Programme* and is looking for a companies and independent testers to collaborate for this event. Please click [here](#) for more details.

You can also see the youtube demo of myTestingBank tool by clicking below image



About TuniTeam:

TuniTeam is a service company specializing in Mobile software development and validation activities. It is operational since 2009 and growing up mainly in European marke



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:

First Prize

Name: Ashwini Sadashivam

Second Prize

Name: Venkatakanthi Datla

Third Prize

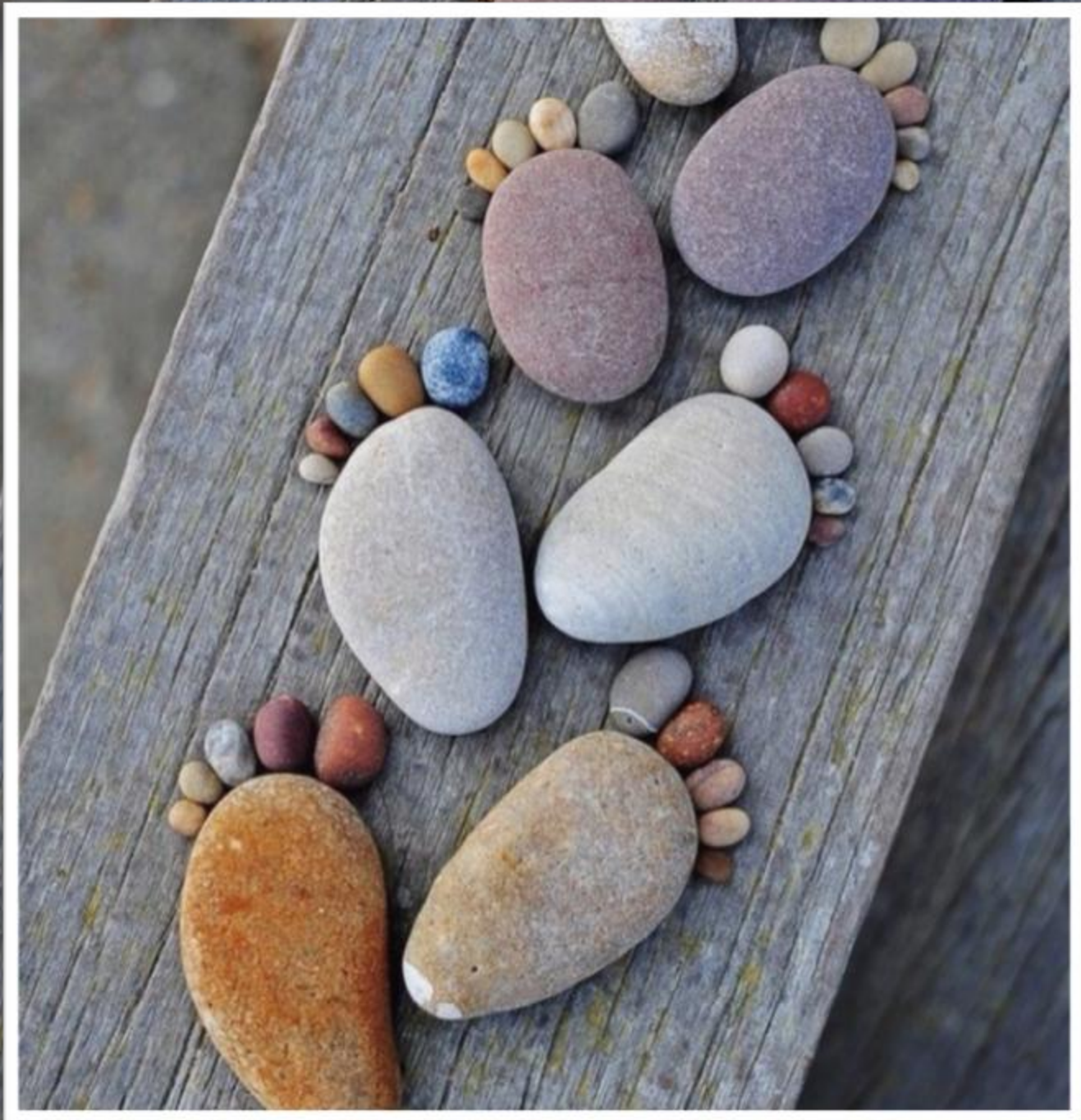
Name : Saritha Prakash

Congratulations !

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

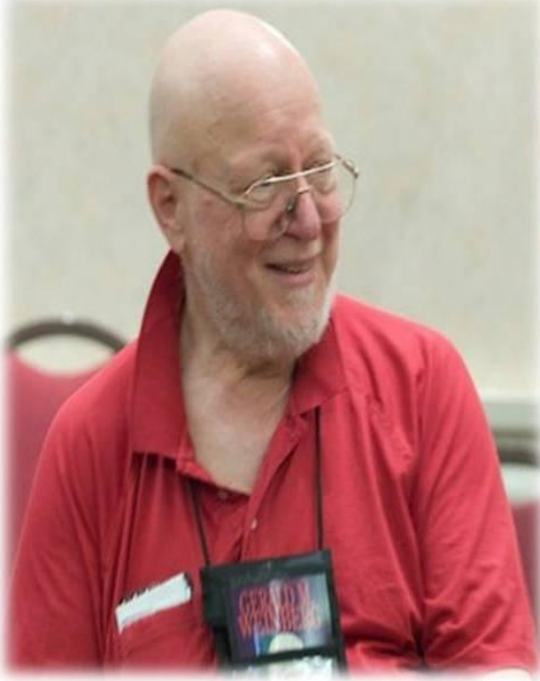
Names of other winners who had sent us correct answers will be declared on our Facebook Page.

Haven't you joined the gang yet? Really?



Join us on Facebook

Tea & Testing



with

Jerry Weinberg

The Fish-Eye Lens (Part 3)

For instance, The Helpful Law:

("No matter how it looks, everyone is trying to be helpful.") reminds me to notice when they classify everyone as either friend and foe, a separation of variables that may not be contributing to solving their problems.

The Background Blindfold

So, since all these tools are available to my clients, too, why is seeing this Giant Hairball such a problem? Why don't they just look around them? One problem is that the context is always there, always impinging on their senses. That should make it obvious to them, except for the psychological phenomenon of habituation—the successive reduction of response to a repetitive stimulus. In common parlance, this is the Background

Blindfold which says, The fish is always the last to see the water.

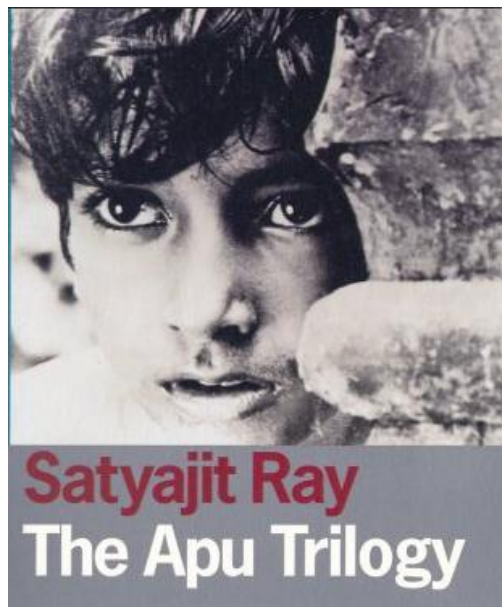
Habituation is a way of canceling out constancies in the environment, and it probably occurs in almost all complex systems, otherwise they would be overloaded with sensory data that most of the time

doesn't carry much information. Habituation is a way of lumping the constancies into a category of "Things that I don't need to pay attention to, because they're always there."

When something new first appears in our environment, we're very stimulated. But after it remains for a while and seems to hold neither threat nor opportunity, we (unconsciously) make it part of our model of the constant environment. It has become habituated out of our awareness, and now its removal will excite our interest.

A stunning example of the removal of an habituated stimulus was shown in Satyajit Ray's movie trilogy, *The World of Apu*. When Apu gets the news of his wife's death, he throws himself on the bed and remains unable to move for days. Ray shows him lying in bed in this state until suddenly the alarm clock stops ticking. Apu is startled out of his lethargy and started on the road to recovery, but the real impact is on the viewer, who has also habituated to the ticking of the clock. Of course the viewer is utterly unaware of the ticking until it stops and the silence strikes like a thunderbolt.

In the same way, my clients are habituated to the context in which they spend their working days, and they generally notice it only when it changes. Therefore, they are not naturally good sources of information about that environment. They have habituated the context away. They are the fish, and the organization is their water.



The Foreground Fantasy

Another kind of bias arises directly from the Background Blindfold. If people are habituated to the ordinary things in their environment, they automatically are biased towards noticing things that are not habitual. I call this the Foreground Fantasy:

The fish is always the first to notice air.

Indeed, some exceptional event is often the triggering reason for me to be called in—a system crashes, a project fails to produce a viable product, an employee (or two, or ten) quits, Chicken Little runs around crying that the sky is falling. Regardless of how real the threat is, the "fantasy is experienced as reality," and they begin to flop around like fish out of water. Usually, that's when they call me.

Fortunately, when I'm summoned as an external consultant, I'm the real fish out of my usual water, so a Fish-Eye Lens is the perfect tool by which I can put both background and foreground in perspective. Once I have this perspective, perhaps I can calm the excited ones and arouse the cool ones.

The Five-Minute Rule

Even when my clients become aware of some of their surroundings, both they and I are subject to a variety of observational biases.

Here's an example involving Time Magazine's Man of the Century, Albert Einstein:

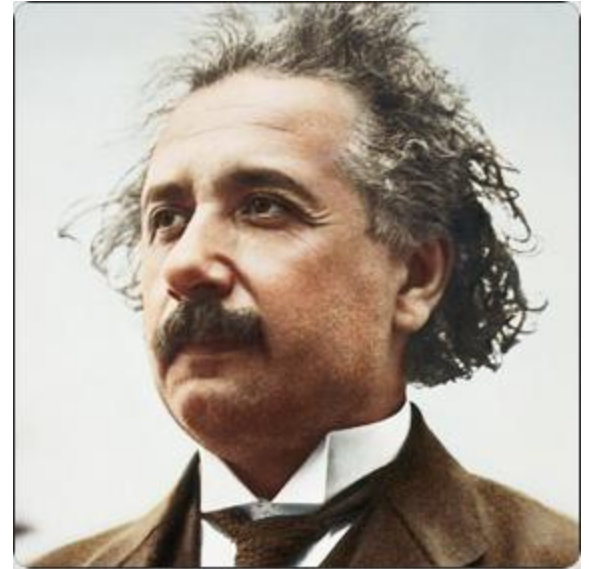
Ernst Strauss, Einstein's assistant from 1944 to 1948, tells the story of searching for a paper clip with Einstein. They found a bent one, so they needed a tool to straighten it. When he found a box of new

paper clips, Einstein took one out and bent it into a tool to straighten the other one. When Strauss pointed out how ridiculous this was, Einstein said, "Once I am set on a goal, it becomes difficult to deflect me."

I love Einstein stories like this; they make me feel so smart. As a consultant, I'm generally called in when there's a problem, and generally that means that my clients, like Einstein, are overly focused on a solution. Thus, they fail to see relevant things in their environment. Often, as in this story, they fail to see solutions they are holding in their hands.

That's where the Five-Minute Rule comes from. I wrote of this rule in *The Secrets of Consulting*, but it's worth repeating here:

Clients always know how to solve their problems, and always tell the solution in the first five minutes. It's worth repeating because inexperienced consultants have a hard time believing it could be true. When they enter a new school of flopping fish, they're so overwhelmed with their own issues that they forget to focus their Fish-Eye Lens, so they miss the solution that their clients, like Einstein, are holding but not seeing.



Satir's Three Universal Questions

How do I get all this information about context to begin with, so I can strip out all these biases? One way is to use Virginia Satir's Three Universal Questions, modified slightly to take into account the entire organization and all its people:

- How did they get here? (Past)
- How do they feel about being here? (Present)
- What would they like to have happen? (Future)

If you read *The Secrets of Consulting*, you'll recognize that "How did they get here?" is the general question underlying Boulding's Backward Basis:

Things are the way they are because they got that way.

You'll also recognize "How do they feel about being here?" as embodied in Brown's Brilliant Bequest:

Words are often useful, but it always pays to listen to the music (especially your own internal music).

"What would they like to have happen?" is often harder to get at, because each organization has "official" aspirations, but these seldom give an accurate picture of what people are really hoping will happen.

Here are some examples of Big Picture questions that help me see the larger context, and which once again make an enormous difference in how I approach an assignment.

How did they get here?

- Can I get a history of the organization? Is it an official history, and if so, how does it differ from the unofficial oral history that I hear from people?
- What's their past experience with consultants? What was the process by which they chose me? Is that typical in this organization?
- How long do people typically stay in this organization? How long do they stay in the same jobs? Where did they come from before they got here?
- Is this a profit-making company, and if so, what is its history of profitability? If it's a non-profit, what is the vision that created them?

How do they feel about being here?

I can get the best impression of this part of the context by simply walking around and meeting people, being friendly and inquisitive about their work and the things I see around their workplaces. Of course, if the people who invited me discourage me from walking around and doing this, that tells me more than a thousand pictures.

- Are the people I meet eager? Happy? Friendly?
- Do they like their surroundings? What kind of evidence do I find in the way they organize their workspaces? Do they feel free to show evidence of their personal life, and if so, what do they show and not show?
- What evidence do they show of professionalism and how they measure it? Do I see certificates as evidence of training, and if so, what training are they showing off?

Do they keep things tidy? Do they keep things so tidy it looks as if they are afraid of citations for disorderly workplace?

- Are people puzzled about what's expected of them in this organization? Do their titles have meaning? Do they use their titles, and if so, how do they use them? As badges of honor? As attempts to force respect?
- Are they sure of themselves? Do they feel empowered to make decisions and be backed up by the organization?
- Is this the right mood for succeeding in this job? If not, what would have to happen to get them in the right mood?

What would they like to have happen?

- Why did they look for a consultant? For reassurance about what they've already decided? The expertise? As a scapegoat? To hold their hands?
- What will success look like, to them?
- How long do they want me here?

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

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 light-colored sand surface. In the sand, there is a faint, circular mandala-like pattern. 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

Why is testing taking so long ?

- part 2



- By Michael Bolton

Last time (read Part 1 of this series) I set up a thought experiment in which we divided our day of testing into three 90-minute sessions. I also made a simplifying assumption that bursts of testing activity representing some equivalent amount of test coverage (I called it a micro-session, or just a “test”) take two minutes. Investigating and reporting a bug that we find costs an additional eight minutes, so a test on its own would take two minutes, and a test that found a problem would take ten. Yesterday we tested three modules. We found some problems. Today the fixes showed up, so we’ll have to verify them.

Let’s assume that a fix verification takes six minutes. (That’s yet another gross oversimplification, but it sets things up for our little thought experiment.) We don’t just perform the original microsession again; we have to do more than that. We want to make sure that the problem is fixed, but we also want to do a little exploration around the specific case and make sure that the general case is fixed too.

Well, at least we’ll have to do that for Modules B and C. Module A didn’t have any fixes, since nothing was broken. And Team A is up to its usual stellar work, so today we can keep testing Team A’s module, uninterrupted by either fix verifications or by bugs. We get 45 more micro-sessions in today, for a two-day total of 90.

Module	Fix Verifications	Bug Investigation and Reporting (time spent on tests that find bugs)	Test Design and Execution (time spent on tests that <i>don't</i> find bugs)	New Tests Today	Two-Day Total
A	0 minutes (no bugs yesterday)	0 minutes (no bugs found)	90 minutes (45 tests)	45	90

Team B stayed an hour or so after work yesterday. They fixed the bug that we found, tested the fix, and checked it in. They asked us to verify the fix this afternoon. That costs us six minutes off the top of the session, leaving us 84 more minutes. Yesterday's trends continue; although Team B is very good, they're human, and we find another bug this time. The test costs two minutes, and bug investigation and reporting costs eight more, for a total of ten. In the remaining 74 minutes, we have time for 37 micro-sessions. That means a total of 38 new tests this time—one that found a problem and 37 that didn't. Our two-day today for Module B is 79 micro-sessions.

Module	Fix Verifications	Bug Investigation and Reporting (time spent on tests that find bugs)	Test Design and Execution (time spent on tests that <i>don't</i> find bugs)	New Tests Today	Two-Day Total
A	0 minutes (no bugs yesterday)	0 minutes (no bugs found)	90 minutes (45 tests)	45	90
B	6 minutes (1 bug yesterday)	10 minutes (1 test, 1 bug)	74 minutes (37 tests)	38	79

Team C stayed late last night. Very late. They felt they had to. Yesterday we found eight bugs, and they decided to stay at work and fix them. (Perhaps this is why their code has so many problems; they don't get enough sleep, and produce more bugs, which means they have to stay late again, which means even less sleep...) In any case, they've delivered us all eight fixes, and we start our session this afternoon by verifying them. Eight fix verifications at six minutes each amounts to 48 minutes. So far as obtaining new coverage goes, today's 90-minute session with Module C is pretty much hosed before it even starts; 48 minutes—more than half of the session—is taken up by fix verifications, right from the get-go. We have 42 minutes left in which to run new micro-sessions, those little two-minute slabs of test time that give us some equivalent measure of coverage. Yesterday's trends continue for Team C too, and we discover four problems that require investigation and reporting. That takes 40 of the remaining 42 minutes. Somewhere in there, we spend two minutes of testing that doesn't find a bug.

So today's results look like this:

Module	Fix Verifications	Bug Investigation and Reporting (time spent on tests that find bugs)	Test Design and Execution (time spent on tests that <i>don't</i> find bugs)	New Tests Today	Two-Day Total
A	0 minutes (no bugs yesterday)	0 minutes (no bugs found)	90 minutes (45 tests)	45	90
B	6 minutes (1 bug yesterday)	10 minutes (1 test, 1 bug)	74 minutes (37 tests)	38	79
C	48 minutes (8 bugs yesterday)	40 minutes (4 tests, 4 bugs)	2 minutes (1 test)	5	18

Over two days, we've been able to obtain only 20% of the test coverage for Module C that we've been able to obtain for Module A. We're still at less than 1/4 of the coverage that we've been able to obtain for Module B.

Yesterday, we learned one lesson:

Lots of bugs means reduced coverage, or slower testing, or both.

From today's results, here's a second:

Finding bugs today means verifying fixes later, which means even less coverage or even slower testing, or both.

So why is testing taking so long? One of the biggest reasons might be this:

Testing is taking longer than we might have expected or hoped because, although we've budgeted time for testing, we lumped into it the time for investigating and reporting problems that we didn't expect to find.

Or, more generally,

Testing is taking longer than we might have expected or hoped because we have a faulty model of what testing is and how it proceeds.

For managers who ask "Why is testing taking so long?", it's often the case that their model of testing doesn't incorporate the influence of things outside the testers' control. Over two days of testing, the difference between the quality of Team A's code and Team C's code has a *profound* impact on the amount of uninterrupted test design and execution work we're able to do. The bugs in Module C present interruptions to coverage, such that (in this very simplified model) we're able to spend only *one-fifth* of our test time designing and executing tests. After the first day, we were already way behind; after two days, we're even further behind. And even here, we're being optimistic. With a team like Team C, how many of those fixes will be perfect, revealing no further problems and taking no further investigation and reporting time?

And again, those faulty management models will lead to distortion or dysfunction. If the quality of testing is measured by bugs found, then anyone testing Module C will look great, and people testing Module A will look terrible. But if the quality of testing is evaluated by coverage, then the Module A people will look sensational and the Module C people will be on the firing line. But remember, the differences in results here have *nothing* to do with the quality of the testing, and *everything* to do with *the quality of what is being tested*.

There's a psychological factor at work, too. If our approach to testing is confirmatory, with steps to follow and expected, predicted results, we'll design our testing around the idea that the product should do this, and that it should behave thus and so, and that testing will proceed in a predictable fashion. If that's the case, it's possible—probable, in my view—that we will bias ourselves towards the expected and away from the unexpected. If our approach to testing is exploratory, perhaps we'll start from the presumption that, to a great degree, we don't know what we're going to find. As much as managers, hack statisticians, and process enthusiasts would like to make testing and bug-finding predictable, people don't know how to do that such that the predictions stand up to human variability and the complexity of the world we live in. Plus, if you can predict a problem, why wait for testing to find it? If you can really predict it, do something about it *now*. If you don't have the ability to do that, you're just playing with numbers.

Now: note again that this has been a thought experiment. For simplicity's sake, I've made some significant distortions and left out an enormous amount of what testing is really like in practice.

- I've treated testing activities as compartmentalized chunks of two minutes apiece, treading dangerously close to **the unhelpful and misleading model of testing as development and execution of test cases**.
- I haven't looked at the role of setup time and its impact on test design and execution.
- I haven't looked at the messy reality of having to wait for a product that isn't building properly.
- I haven't included the time that testers spend waiting for fixes.
- I haven't included the delays associated with bugs that block our ability to test and obtain coverage of the code behind them.
- I've deliberately ignored the complexity of the code.
- I've left out difficulties in learning about the business domain.
- I've made a highly simplistic assumptions about the quality and relevance of the testing and the quality and relevance of the bug reports, the skill of the testers in finding and reporting bugs, and so forth.
- And I've left out the fact that, as important as skill is, luck always plays a role in finding problems.

My goal was simply to show this:

Problems in a product have a huge impact on our ability to obtain test coverage of that product.

The trouble is that even this fairly simple observation is below the level of visibility of many managers. Why is it that so many managers fail to notice it?

One reason, I think, is that they're used to seeing linear processes instead of organic ones, a problem that Jerry Weinberg describes in **Becoming a Technical Leader**. Linear models "assume that observers have a perfect understanding of the task," as Jerry says. But software development isn't like that at all, and it can't be. By its nature, software development is about dealing with things that we haven't dealt with before (otherwise there would be no need to develop a new product; we'd just reuse the one we had). We're always dealing with the novel, the uncertain, the untried, and the untested, so our observation is bound to be imperfect. If we fail to recognize that, we won't be able to improve the quality and value of our work.

What's worse about managers with a linear model of development and testing is that "they filter our innovations that the observer hasn't seen before or doesn't understand" (again, from *Becoming a Technical Leader*.) As an antidote for such managers, I'd recommend **Perfect Software, and Other Illusions About Testing** and **Lessons Learned in Software Testing** as primers. But mostly I'd suggest that they *observe the work of testing*. In order to do that well, they may need some help from us, and that means that we need to observe the work of testing too. So over the next little while, I'll be talking more than usual about **Session-Based Test Management**, developed initially by James and Jon Bach, which is a powerful set of ideas, tools and processes that aid in observing and managing testing.

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](#)

**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.

Hi Guiding Star,

My name is Ankit and I am working as a manual tester for testing mobile applications and games.

I have 3 years of experience in the same domain, but at this point of time I am so confused about my future as in how do I enhance my career graph? Currently it's getting very difficult for me to find a nice paying decent job as a mobile app/game tester.

Which all tool, skills should I learn to enhance myself as mobile app tester? Seeking your expert advice.

With Regards,

Ankit

Hello Ankit,

I understand your worry. At some point anyone can get concerned about the current situation and become suspicious about personal growth.

It also depends on kind of company you are working in. If you are working in an organization where there is no much scope for your growth, you may think in the same way for complete domain you are working in. In addition you are looking forward for a job and somehow it is not working for you, is making you more frustrated.

I think Mobile Testing domain is still emerging and you should not be disappointed at all. Due to overall international scenario, many companies have controlled their hiring and it is impacting all sectors. You can use this time in improving your skill sets, working on mobile automation testing tools and at the same time keep on searching for better opportunity.

You can try exploring some open source mobile automation tools like Monkeytalk and Robotium. There are some good cross platform paid tools also which you can explore e.g. SeeTest, Zap-Fix etc.

Testers having good knowledge and experience on Mobile Automation Testing are what employers are looking for. Since you have 3 years of experience, employers will certainly expect some advanced skills from you.

Here are some quick points of action for you:-

1. Analyze the skill sets that employers are looking for. You can take a look on job descriptions for that.
2. Start Exploring some Automation Tool, gain scripting knowledge if possible.
3. Try for some (good) software testing certification. I am advising you not just to get just certified but for gaining some confidence and to gain awareness about some skills, practices which you might not have learned till now.
4. You can actively participate some good testing communicates to learn different things, keep yourself updated and of course to find out some good job opportunities.

I hope this helps you to get prepared for better opportunities. Feel free to reach me for any queries around mobile app testing. Wish you all the best.

- Anurag Khode



The Credible Tester

- PART 1

- By Joe DeMeyer

A few years ago, when I joined a new project, many things started to happen. I attended introductory meetings, and project information was emailed to me. Also, I met with other team members assigned to the same project. We talked about the project, its purpose, strategies and similarities to previous projects, and how we might fulfill the business intent. Some of us dabbled in prototyping solutions.

Joining that project as a *developer* has some interesting contrasts to my experiences starting a project as a Test Lead. During my transition from developer to tester, there were challenges I did not experience or expect.

As a Test Lead, I did not expect skepticism around my ability to understand the products under development, or concern around starting on test cases, or subtle clues I may not be qualified to lead let alone test. In short, I felt no perceived credibility. A developer has some built-in credibility with a title, the college education associated with it, and possibly a reputation with other developers. Even with a new Project Manager or colleagues, the credibility of a developer seems to start with some level of confidence.

The Credible Tester

Establishing credibility (and eventually a reputation) in testing and technology is a challenging and worthwhile but necessary effort on every project. It's challenging not only because of the technology domain but because a tester's perceived value is mostly in test execution. While you can use that to your advantage, credibility is important since many subsequent testing activities depend on it. Your status reports are taken more seriously, more of your bugs are fixed, your contribution is valued, and your ability to influence change improves.

While your skill and qualifications may place you on a project in a test lead role or land you a great test lead job, try some of these ideas to grow your credibility and your testing career.

Be Available, Visible, and Vocal

During my first Test Lead assignments, I noticed Project Managers tended to communicate more with developers. I hadn't noticed it as a developer but as a Test Lead it was very evident. A tester has work comparable to any team member and their results help determine direction and course of the project. Much of that work should be discussed with the PM early in a project to set expectations.

Not only did I learn to schedule a meeting with the PM and other team members soon after starting a project, I learned to be available, visible, and vocal during those first weeks: available to meet project sponsors and team members, visible at kick off and other introductory meetings, and vocal about testing's role in the project. I'm not recommending a testing-is-the-center-of-the-universe attitude but you want to demonstrate strength and confidence without being argumentative or arrogant.

MEET THE PROJECT TEAM

When I am assigned to a project, I want to meet all the team members during the first week (if you want to establish your credibility, you can't be in the background). I especially want to meet with the Project Manager, Lead Developer, and even project sponsors. These meetings provide you an opportunity to not only help your new team members understand more about testing, but learn more about the project from many points of view.

When I meet with the Project Manager, I discuss project goals at a high level to get an understanding of the project and to establish rapport. If you have not previously worked with the PM, some easy ice breakers are asking about tenure in the industry or with the company. These can open other topics including technology, career arc, or previous work in other industries.

In general, I believe that project management still has opportunities to learn more about testing and how to best use testing. By opening these topics early, you begin to explore their understanding and expectations. Where their understanding may be incomplete or biased, take that opportunity to establish an academic foundation for exploring new meanings to testing concepts.

While I convey new testing ideas, I don't want to start a heated discussion. Rather, I want to provoke a conversation that both lays a foundation for working together and for experimenting with testing ideas that may be unknown to the Project Manager (see Table 1 for topics to discuss with them and other team members).

Table 1. Discussion Topics

Project Manager	When I meet with a Project Manager, I discuss their expectations for testing automation, testing completion, test status reporting, and test execution.	
	Possible Expectation	Possible Testing Strategy
	Use automation wherever possible.	We will review products and determine where automation makes sense, and where automation can be maintained.

	Testing will be complete when all test cases have executed, and you sign off on the product.	Test cases will be created to address risk, provide coverage, and evaluate business intent. Let's work together to prioritize the test cases. As we execute them, we will provide our results and behavioral observations about the products often. When you and the business sponsor believe you've realized the value and quality you expected, you can decide to stop testing.
	Provide a weekly status report.	We will schedule periodic meetings with you and the business sponsor to review information we have learned from our tests.
	Test execution can start after the Code Complete date.	We prefer to begin exploration and execution immediately. In this manner, we can provide information sooner to you and developers to help prevent defects after the Code Complete date.
	<p>I also discuss some basic information about testing: The nature of testing is a search for information and to report that information. Testing and testers do not provide decisions. In my experience, this occasionally causes enough cognitive dissonance to begin a deeper discussion of testing's role. Embrace that conversation and use it to lay a foundation and build rapport.</p> <p>Lastly, I might describe some new testing techniques I want to experiment with during the project. Describe the techniques and how they provide value to the project.</p>	
Lead Developer	In discussing the project with the Lead Developer, a good understanding of the domain and technology can assist in establishing rapport and technical credibility. I want to understand where they think risks are, and where they believe the most challenging development occurs.	
Project/Business Sponsor	When I discuss the project with project sponsors, I want to understand what they expect from the project, what they think the benefits are, and where they think it will help meet enterprise goals. Ideas from this conversation may help focus development of risk-based test cases and scenarios.	

Testers	While discussing the project with a tester, I want to convey a sense of my understanding of the project, and where I think there are testing opportunities. I also want to talk about testing experiences, and assess the tester for strengths, opportunities, and their expectations for the project.
---------	--

Meeting the Team (sidebar/anecdote)

In one project, my first team meeting occurred when I could not be in the office. I dialed into the meeting and was introduced as the Test Lead. Occasionally, participating in a meeting by phone can be challenging however attending the first meeting of all team members on a new project was interesting. But I think it had an unintended positive affect.

The PM started going through the project schedule and mentioned a few meetings would occur to provide details. When asked who wanted to attend, several people responded and I asked to attend also. There was a pause and a clarifying question. "Yes", I replied, "I would like to attend that meeting, please." When more meetings were discussed, clarifying questions became invitations. When finally I met other people on the project, they knew me as the tester who wanted to participate in many meetings.

I want to participate in the project at the same level as most other team members – at least at the start of the project.

to be continued in next issue...

Joe DeMeyer has been working in IT for over 10 years in both development and testing roles. He enjoys sharing his experience with his peers and the FIRST Lego League team he coaches.

To maintain his skills, Joe designs, builds, and most importantly, tests small payloads and flies them using a weather balloon.

Contact Joe at Dev_To_Test@yahoo.com.



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 young 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



By Bernice Niel Ruhland

The Importance of Social Networking – Part 3



Since November 2011, I have published a series of articles on how testers can develop their skills and knowledge. I would like to take a different approach to discuss how I apply my own advice. Recently, I had to learn how to effectively test a web-based application across multiple browsers. To accomplish this task I needed to understand how other testers approach this problem and the tools they are using. I am fortunate to have a large testing network through social media and that many of these testers are bloggers.

This series does not discuss how I perform cross-browser testing. Instead it provides a wealth of information shared from the Testing Community and information from my personal research.

I would like to thank everyone who contributed information. Without their willingness to share their experiences and recommendations, I would not have been able to understand potential approaches in such a short time. A heart-felt thank you to: Ajay Balamurugadas, Mike Talks, Lisa Crispin, Martijn de Vrieze, Anne-Marie Charrett, Karen Johnson, Gagneet Singh, Dave McNulla, Moise Stedte, Akshay Thakkar, and Dorothy Graham.

What tools are available to assist in testing?

Automated Tools Research

I attended a Dorothy Graham session called "What Managers Think They Know about Test Automation – But Don't" through the STAREAST 2012 Virtual Conference. I found the session to be informative plus it provided resources to help plan and implement automation. I emailed Dorothy and she kindly provided a risk assessment spreadsheet, containing multiple tabs to understand the return on investment from test automation. Below are a few automation and management tips from the virtual session.

Automation Points

- Automation should free up mundane testing to allow more time for intelligent investigation.
- Automation should support the overall testing strategy.
- The scripts do not think; the scripts do what they are told.
- Automation is not a magic tool. It will not find bugs - good tests find the bugs.
- It is a good idea to check some of the green tests. Did they really pass? Correctness of your expected results is more important in automated tests since a tester will not see the mistake.
- Building good test automation takes time and effort ... it doesn't just come out of the box.
- DRY = don't repeat yourself. If the test is repeated, remove it.

Management Points

- It is important to keep your management team informed. Provide them with quick, small reports. Tell them the important things they need to know.
- Establish common ground to educate management. What are their goals? Why are they interested in automation? Are their goals realistic? Remember unrealistically high goals can result in failure.
- Automation ROI = (benefit - cost) / cost (cost could be effort such as salary cost)
- Relationships are key for success in test automation. ie., developers, managers.

Dorothy has a new book in collaboration with Mark Fewster called "Experiences of Test Automation". The book is written from a case study perspective with each case study as a standalone account allowing the chapters to be read in any order. The book covers automation projects that succeeded and failed to provide a variety of learning points. To contact Dorothy and for more information refer to her website at: <http://www.dorothygraham.co.uk/>.

Tools

Tools like WebUI Test Studio QA are recording the "elements" in use. You can select the browser type when you play the recorded tests. Refer to their website for more details and pricing information. The following link takes you to their "Get to know WebUI Test Studio QA Edition Webinar" <http://www.telerik.com/automated-testing-tools/support/videos/overview/get-to-know-webui-test-studio-qa-edition-webinar.aspx>

WebAii Free framework records and generates corresponding code. The following link takes you to a short video: <http://www.telerik.com/automated-testing-tools/support/videos/overview/webaii-testing-framework-overview.aspx> .

If you want to run on different operating systems and browsers, refer to sikuli project where you create a script once. The following link takes you to their website where a demo is provided. <http://sikuli.org/>.

Consider a service such as SauceLabs' cloud testing that lets you run automated cross-browser regression tests. Refer to their website for more information and to run a test. <http://saucelabs.com/>

Cross-Browser Testing Tools

- SauceLabs: <http://saucelabs.com/>
- TestApi: <http://testapi.codeplex.com/>
- Lunascape: <http://www.lunascape.tv/>
- IE Tester: <http://www.my-debugbar.com/wiki/IETester/HomePage>

- Gomez: <http://www.gomez.com/google-adwords/gomez-platform-free-trial/?gclid=CPSJpKypj7ACFUOo4AodJJJeqA>
- Spoon.net: <http://spoon.net/browsers/>
- Selenium: <http://seleniumhq.org/>
- Watir-WebDriver: <http://watirwebdriver.com/>
- CrossBrowserTesting: <http://crossbrowsertesting.com/>
- BrowseEmAll: <http://www.browseemall.com>
- Adobe BrowserLab: <https://browserlab.adobe.com/en-us/index.html>
- Browsershots: <http://browsershots.org/>

Articles

- Review of Cross-Browser Testing Tools. <http://www.smashingmagazine.com/2011/08/07/a-dozen-cross-browser-testing-tools/>
- 12 Best Cross Browser Testing Tools to Ease Your Browser Compatibility Testing Efforts. <http://www.softwaretestinghelp.com/best-cross-browser-testing-tools-to-ease-your-browser-compatibility-testing-efforts/>
- The Importance of Cross Browser Compatibility: Tips and Resources. <http://www.noupe.com/tools/the-importance-of-cross-browser-compatibility-tips-and-resources.html>
- 10 Useful Tools for Cross-Browser Compatibility Check. <http://www.1stwebdesigner.com/design/tools-browser-compatibility-check/>
- Browser Compatibility. <http://www.htmlite.com/SD011.php>
- Cross-browser Web application testing made easy. <http://www.ibm.com/developerworks/web/library/wa-crossbrowser/>
- Risk-Based, Cross-Browser Testing with Scrum. <http://mattarcherblog.wordpress.com/2011/11/08/risk-based-cross-browser-testing-with-scrum/>



Bernice Niel Ruhland is a Software Testing Manager for ValueCentric, LLC a software development company located in Orchard Park, New York. She has more than 20-years experience in testing strategies and execution; performing data validation; and financial programming.

To complete her Masters in Strategic Leadership, she conducted a peer- review research project on career development and on-boarding strategies. She uses social media to connect with other testers to learn more about their testing approaches to challenge her own testing skills.

The opinions of this article are her own and not reflective of the company she is employed with. If you have any questions / comments on this article or if you would like to connect, Bernice can be reached at:

LinkedIn:
<http://www.linkedin.com/in/bernicenielruhland>

Twitter: bruhland2000

G+ and Facebook: Bernice Niel Ruhland

[illegible]

with Anurag

Cloud Based Mobile Software Testing Services and Tools

With the boom of various technologies in the market, Software Testing Services has entered in a new era. Mobile Software Testing is no exception for this. I am closely monitoring the Mobile Testing Market since 5-6 years and believe me it is growing very rapidly. The mobile testing market is becoming mature and lots of tools, processes, services are coming in to picture for Mobile Apps Testing. Cloud based mobile testing services/tool is a link in the same.

Cloud computing is one of the hottest topic now days. By using cloud technology, there are some companies which are providing the user with a Cloud based Mobile Testing solution. It may be Mobile Test Automation via devices over cloud, accessing the devices over cloud for UI and Functional Testing, Performance testing with devices over cloud, setting up the private cloud in the enterprises and so on.

As I am talking of various cloud based mobile Testing solution, we need to understand the basic cloud concept.

What is Cloud?

The cloud lets users to run applications and store data online. In more sophisticated language and as per NIST definition of Cloud, "Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction."

What are the types of Cloud?

Cloud computing can be classified in to three types on the basis of service provided as **SaaS, PaaS, IaaS**.

- **SaaS**-It Stands for Software as a Service. It allows users to run an existing online application. For e.g. Gmail, Google docs, Sales force where software's are out there in the cloud. All applications based on SaaS are free or Charged on a Subscription basis. You can access it by any computer via internet.
- **PaaS**-It Stands for Platform as a Service. PaaS allows user to create their own cloud application using supplier-specific tool and languages. For e.g. Google has a Google App Engine which allows user to create their own cloud based application on Google's Infrastructure. Microsoft also has a PaaS as Azure where it allows to create application.
- **IaaS**-It Stands for Infrastructure as a Service. Disk storage and virtual servers are key things here. Amazon EC2, Amazon S3, Rackspace Cloud Servers are some of the leading vendors.

Also on the **basis of location** where the cloud is hosted, cloud can be classified as **Private, Public, Hybrid** and **Community** Cloud.

- **Public Cloud**- In this type of cloud the resources like apps are available publically over internet. Windows Azure Services Platform ,Amazon Elastic Compute Cloud (EC2), IBM's **Blue Cloud**, Sun Cloud, Google AppEngine are example of it
- **Private Cloud**- Private Cloud provides access to the limited people mainly in any enterprise environment. It is also called as internal cloud or corporate cloud. Organization that wants control over their data explicitly goes for this.
- **Community Cloud**: - It shares infrastructure between several organizations from a specific community
- **Hybrid Cloud**-The hybrid cloud is a combination of Public, Private and Community Cloud.

As I am not a cloud computing expert, I have just put forward some basics of cloud computing which may help you understand how cloud works. However you can check about the cloud computing in details over [here](#) and also you can just research over the net. Our main objective is to take a look on how this Cloud computing is entering in the Mobile Testing arena.

Cloud based Mobile Application Testing Services:-

The cloud based solution enabled developers around the world to connect to and control mobile devices over the Internet. For all application testing needs Mobile handsets are connected to the system from which user can access the devices over internet with a web-based UI. Here are some players in Mobile Testing Arena which are using different Cloud based platforms to serve the Mobile Testing Industry better.



- SeeTest Mobile Cloud
- Zap-Fix
- Perfecto Mobile's MobileCloud™ Platform
- Keynote Deviceanywhere
- SOASTA
- Testdroid Cloud
- mAutomate
- SILK
- Neotys
- QaaS(Gorrilalogic)
- MuDynamics

Before selecting such services for your testing needs please ask yourself few questions:-

1. If these services have **sufficient devices** to fulfill your testing needs?
2. **Device Diversity:-** If devices are available on various platforms, of various screen sizes and OS versions?
3. **Carriers:-** If the services let you access different carriers all over the world.?
4. **Automation Capabilities:-** Some services are having Automation Capabilities also.

5. **Support:-** Just look in to company profile. If they can provide you proper support if you encounter any issue?
6. **Security:-** If your sensitive data is secured over this cloud?
7. **Pricing and Plans:-** If it is really going to be cost-effective solution for you?

These are some very basic questions that may help you selecting the right service/Company for you for your Mobile Apps Testing Needs. In our upcoming articles, we will try to cover some more generic topics in Mobile Application Testing and will take a look in some of the player in this service. Till then stay connected and don't forget to drop me a line at anurag.khode@hotmail.com .

You can follow @mobileapptest @anuraagrules @mobileqazone for latest updates in this arena.



Anurag Khode is a Passionate Mobile Application test engineer working for Mobile Apps Quality since more than 4 years.

He is the writer of the famous blog **Mobile Application Testing** and founder of dedicated mobile software testing community **Mobile QA Zone**.

His work in Mobile Application testing has been well appreciated by **Software testing professionals** and **UTI** (Unified Testing Initiative, nonprofit organization working for Quality Standards for Mobile Application with members as Nokia, Oracle, Orange, AT & T, LG Samsung, and Motorola). Having started with this column he is also a Core Team Member of **Tea-time with Testers**.

Contact Anurag at anurag.khode@hotmail.com

I am not able to replicate this issue. This is working fine on my machine so close this bug.

Developer



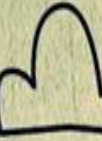
I don't care if it is working fine on your machine. We are not going to deliver your machine to client.

Tester





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**

Stop making testing complicated!

An expert makes things look easy.

One of the things that differentiates between a professional and a non-professional is the way in which the first makes the things she does seem elegant and easy to do (I like to think of it as flowing trivially).

You can see this in the way an acrobat naturally flows in the stage (I saw a performance of "Cirque du Soleil" a couple of weeks ago that made feel this), or in the way painter applies his brush in a soft and almost imprecise manner (although by the way the painting ends up looking, there was nothing imprecise about it!), and even by watching how a gifted programmer comes up with the lines of code that will elegantly and correctly fulfill the goal of his program.



The common thread to all of these is how they make a complicated task seem simple and at times even trivial.

Can testing be elegant?

The answer is YES.

I remember a session during a QA conference in the US some years ago. A very experienced tester was giving a presentation on a particular testing method, explaining how by following some basic rules and with the helps of heuristics you can find most bugs in almost any application.

To prove his point he challenged the audience to name any application that can be brought up on his computer and he would quickly find at least 5 (or was it 10?) bugs on it, on stage and live.

I don't recall the application they choose, it might have been notepad or even the windows calculator, but the fact is that this tester took the application and very quickly, making it seem almost trivial, he found a number of pretty amazing bugs. He even managed to crash the application twice on two unrelated bugs.

What this tester did was make testing seem easy and very elegant.

Why do some testers make things seem so complicated?

This is a hard question to answer, basically because I think there are a number of answers to it.

1. Everything is complicated when you don't know how to do it right

Think about learning to ride a bike. When you tried it first it seemed like an impossible task, most probably you fell hundreds of times, and many of us still have the scars on our knees to prove it 😊

Whenever you are learning something it will be complicated at first. Until you learn how to do it right you will feel and look clumsy, insecure, and definitely-not-flowing.

So, a reason some tester make testing seem complicated is because they haven't learned how to test.

The problem here is that, unlike riding a bike where most people end up riding nicely, there are many testers that even after a number of years in the profession (and some of them with nice diplomas and certifications hanging from their walls) they have still not learned how to test right. These guys will never make testing seem easy.

2. Some types of testing are really complicated

Another reason is that, no matter what you think or say, some types of testing are really complicated.

There are a number of examples:

- Complex embedded systems, where the number of components and integrations is extremely large.
- Working on a system that is still under development, where you need to improvise a lot of the environment (e.g. stubs or drivers) only to make things respond. And many more.

Sometimes you simply cannot expect to make something that is too complicated look simple...

3. Testing itself is not complicated, but the changing environment makes it look chaotic

An additional cause can be related not to the testing itself but to testing project environment, especially when it changes so much that it makes any type of plans or preparations useless.

You can see this in Start-Up companies where most of the things are not 100% defined, and changes in plans and products are not only a reality but an advantaged used to succeed in forming a winning enterprise.

You can also see this on companies of any size going over troubled and turbulent times, when there is too much pressure, sometimes even layoffs, and this reflects on the way the priorities and projects are constantly changing. This environment makes the work of the testers, as well of almost all employees not only seem but also be complicated.

4. Some testers think making their jobs seem complex will help them to justify their work

Finally there are the testers who may already have the experience and knowledge to be good testers, who also work on systems that are not really complex, and in companies that are not undergoing any kind of turbulent times, but still *choose* to make their work seem complicated.

Why would they want to do such a thing? I am not sure, to me this seems not only unnecessary but in a sense also dumb.

One of my guesses is that they feel that if they don't make testing seem complicated, and if they don't make the whole team believe that they are doing something hard, they will end up losing their jobs.

The problem with this is that by making their work seem complicated, and by been all the time "busy" with their complex preparations and planning, they become bottlenecks.

So instead of helping the whole team to save time and to be more productive they end up been a burden and a waste. Something that in my mind has a bigger chance of getting them replaced or simply fired.

Slackers are a burden to all the testing community!

It would still be somewhat OK if this behaviour would harm only them... But since there are too many testers who make their work seem complex only because they are afraid to make testing seem simple, and also because a large number of the testers who claim to have experience in practice don't really know how to test, this ends up reflecting negatively in all the testing community.

I am not sure what can be done about it other than making sure you and your team work professionally and provide real testing value to the organization.

Once, a long time ago, I though that certifications and standardized training could provide an answer, but not I am a lot more skeptic about this...

What do you think?

Do you make testing look simple?

What can be done with testers who think that by making testing seem complex they are justifying their work, or with the testers that knowingly or un-knowingly refuse to learn how to test? I am curious to know your opinion and hear about your experience on these points.

[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

Agile Sutra - "Sensitize & Prevent", not "Design & Execute"

As a consultant deploying HBT (Hypothesis Based Testing) in Agile environment, I made an interesting discovery. That the focus of testing should be on "sensitize & prevent" defects rather than "design & execute" test cases.

In the Agile environment, the focus is on decomposing the problem into small user stories and delivering it. This implies means that we are decomposing complexity and thereby demystifying it. By deduction therefore, therefore the code should be more correct than wrong. Also because the element of delivery 'an user story' is small, it should be easy to test and therefore easy to convert the test into script. And therefore testing is interwoven naturally with coding. Does it mean that the code is cleaner?

In my interactions with the team, I discovered that despite the code being delivered faster, quality challenges exist. Customer reported defects still keep the team busy. Challenges because of extreme focus on the 'small' and not on the larger picture??

Let me illustrate ...

Situation #1

The user story in point is a logging system. This creates detailed logs to enable better supporting. My focus was on testing it. The objective of this is to add entries in the log. As you might surmise, the functionality is not very complex and therefore functional test cases are indeed easy to generate. Therefore the functional test cases were kinda simple... Hmmmm this does not right.

I proceeded to question beyond the typical behaviour of the user story- why are we implementing this, who is going to benefit from this, what they might expect from this and so on. The answers that I got via probing were interesting. The intent stated as the prime reason for this user story was to 'enable better supportability by giving detailed information in the detailed logs'. Yeah, seems the typical reason.

On questioning on how it may look in real life, I discovered that this log file could be a pretty long (a few thousand lines) and not exactly machine analysable. Ouch- this means that the poor support guy would be glued to the monitor in a kinda "edit-search mode" looking for potentially interesting information. Hmmmm.. seems a onerous task that will consume a non-trivial effort/time to diagnose the problem.

On laying out this potential situation after interrogating the user story team, the team understood that this requires serious rework as the "usability" of this is deeply flawed. The supportability is not getting any better. That is when a light bulb started to glow in me - I found an interesting bug, not in the code yet (as this is yet to be coded) but in the design itself.

Situation #2

The user story in this case was "checkpointing", a set of APIs that allows developer to implement "application level transactions" that are needed in the system. This enables take a checkpoint i.e. snapshot of the system to be taken before updating the system with new assets. Post deployment of the assets, in case of any issue, the system can be rolled back to the prior checkpoint.

Similar to Situation #1 the functional behaviour did not complex and therefore the functional test cases were simple. Again my nose wrinkled in suspicion, as the test cases designed were too simple. I embarked on the detailed probe and discovered "a critical situation" where the prior checkpoint would be deleted before the current one is completed resulting in a unrecoverable system. The light bulb in me glowed brightly, a serious flaw uncovered, once again not in the code but the "would-be" code. This happened when we dug into the design of the code, assumptions made (note that we were looking for bug related to environment) and questioning led to this potential flawed situation.

The discovery...

An user story is like a "sutra" - an aphorism, that needs to be delved into detail to understand its entirety. And this is needed if you want to test well. Questioning is a key activity to dig into the details as the typical documentation of user story is condensed. Most often the functional complexity of user

story is low, the challenge is understanding the behaviour of interactions with other stories, environment and the non-functional aspects.

What I discovered is that the act of breaking the "big" into "small" (user stories) makes one forget about who the end user is and what they value. Hence it is necessary to think from the end user's perspective as what they do and how the user story fits in the end user flow and how non-functional attributes of the larger flow matter to the user story.

"Sutras" are powerful, as they communicate deep stuff in a few words. To understand the deep stuff, intense questioning is key. Therefore in the Agile context, testing is therefore not anymore an act of evaluation post coding, it is about intense questioning to "sensitise and & prevent defects" rather than "design & execute test cases".

Write less. Communicate more.

Think deeply and may the light flow into you.

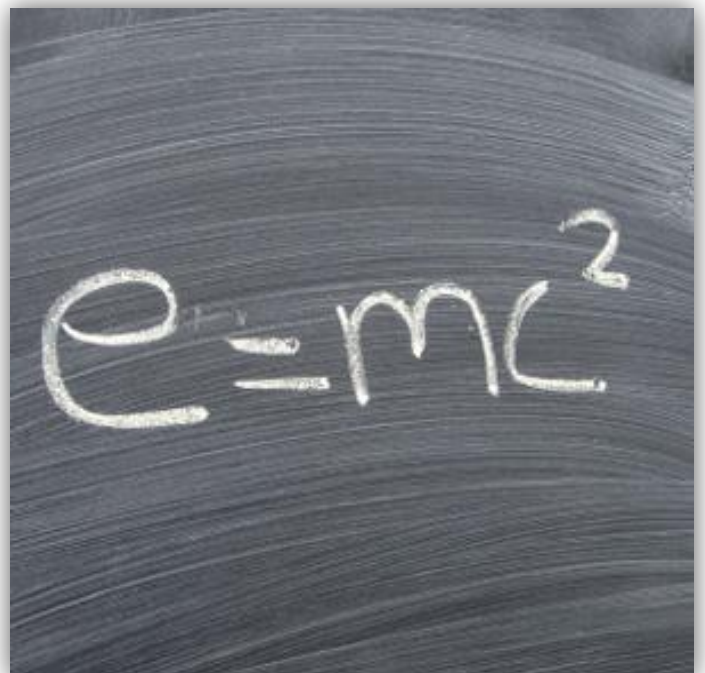
[Back To Index](#) 



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.

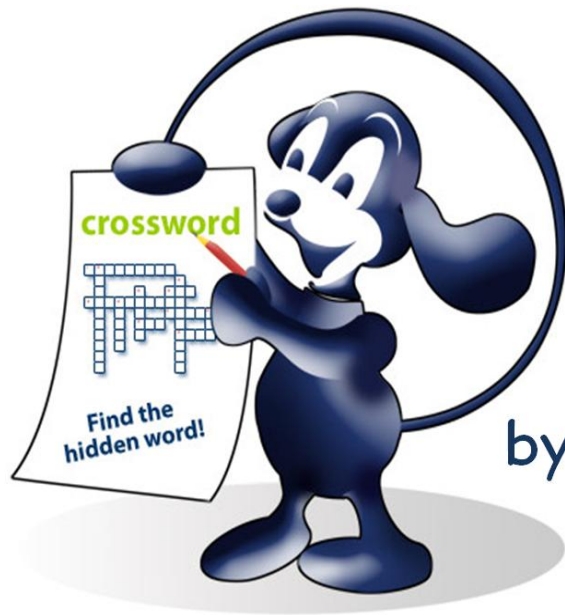
Looking at the scope and future of mobile apps, Mobiles, Smartphones and even Tablets, Mobile QA Zone has been emerging as a Next generation software testing community for all QA Professionals. The community focuses on testing of mobile apps on Android, iPhone, RIM (Blackberry), BREW, Symbian and other 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 18th Sept. 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.

“The bad box”

In the image you have an application that takes 5 inputs and calculates the sum of it. Let's assume one of the input values, corresponding to one of the input box, it is badly processed internally by the application. And this causes for that input value to be multiplied with 0.1 before being added to the sum. For example four of the inputs are let's say 7,5,7,9 and for the malfunctioned one is 3. The result returned is the equivalent of $7+5+7+9+(3*0.1)$ instead $7+5+7+9+3$.

The question is: can you determine by a single test which is bad input box? You only have one try to enter any inputs you want, see the result returned, and based on that point to the malfunctioning one.

Inputs

+ + + +

=

Result

[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
10							
				11			
12							

Horizontal:

1. A mobile app quality tool from uTest (8)
5. It is an automated testing framework for "C" language (5)
7. It is a tool that generates data, tables, views, procedures, etc for database testing purposes, in short form (3)
10. A factor that could result in future negative consequences; usually expressed as impact and likelihood (4)
11. Continuously raising an input signal until the system breaks down, is called _____ testing (4)
12. It is done to evaluate the application's behavior beyond normal or peak load conditions, is called _____ testing (6)

Vertical:

1. VersaTest Certifier belongs to _____ (6)
2. A white box test design technique in which test cases are designed to execute paths is called _____, in short form (2)
3. After the designing and coding phase in software development life cycle, the application comes for testing then at that time the application is stated as _____ in short form (3)
4. It is a black-box GUI test automation tool. Its first 3 words (3)
6. It is a tool for mutation testing of your C# code in order to measure the adequacy of your unit tests (6)
8. The Complete Quality Assurance Test Management tool, its first 3 words (3)
9. Testing after code is mostly complete or contains most of the functionality and prior to reaching customers, is called _____ testing (5)
11. The testing activities that must be repeated when testing is re-started after a suspension, in short form (2)

Answers for last month's Crossword:

S	Y	M	A	N	T	E	C
E		A			L		T
C	U	C	U	M	B	E	R
U		A		A		G	
R	I	S	K	X		G	
I		T		Q			M
T	E	S	T	P	L	A	N
Y		T			T		C



*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 ?

Our Testimonials

When I read your issue for the first time, I realized how much is there happening in the field of testing which I was unaware about. I have started practicing mind maps. Hey and your each issue is worth reading. Thanks for being there and opening up my eyes.

- Deepti Jakatdar

'Tea-time with Testers' is kind of magazine through which software testing has found its voice. Many thanks for bringing it and focusing on real time issues. I must say that you are contributing a lot the evolution of testing. Keep up the great job.

- Vindy K.

Great issue, great website and Smart Tester of the Month competition is cool. I hope to win exciting prize soon.

- KV Burns

in ne>xt issue

articles by -

Jerry Weinberg

T Ashok

Joel Montvelisky

Samarjeet Mohanti

Joe Demeyer

Mike Talks

our family

Founder & Editor:

Lalitkumar Bhamare (Mumbai, India)

Pratikkumar Patel (Mumbai, India)



Lalitkumar



Pratikkumar

Contribution and Guidance:

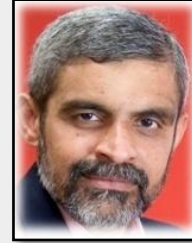
Jerry Weinberg (U.S.A.)

T Ashok (India)

Joel Montvelisky (Israel)



Jerry



T Ashok



Joel

Editorial | Magazine Design | Logo Design | Web Design:

Lalitkumar Bhamare

Image Credits-weipphoto.com

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

