

## Editorial

We're in the midst of the 'Joy of Giving Week' (JGW), a campaign recently started for all of us. YES! Very typically, a democratic campaign -- For Us, By Us, & Of Us. Running from Sep 27 to Oct 3, 2009, it truly is a celebration of unity. Celebrities, CEOs, organizations, and individuals have joined the effort.

What does it set out to achieve? Simply, a Caring World! A world full of joyful people. When we're joyous, everything seems wonderful, everyone wants to work with us, and everyone wants to be near us. We become a Magnet. Imagine a world full of joyful people. In such a world, humanism will be the way of life.

The JGW will be celebrated every year. To know more, visit the JGW website at <http://www.joyofgivingweek.org/>

*Editorial Team*

## News...

### Conquest 2009 Concludes!

The 12th International "Conference on Quality Engineering in Software Technology" (CONQUEST) concluded recently in Nuremberg, Germany. It was organized by the International Software Quality Institute ([www.isqi.org](http://www.isqi.org)).

A few highlights of the conference were:

#### Keynote lectures

- Prof. Dr.-Ing. habil. Josef Nassauer, Bayern Innovativ GmbH/Technical University Munich, Germany

## What's Inside

News	1
Editorial	1
Upcoming Conferences	2
Article	3
Interview	7

## CONQUEST '09

12th International Conference  
on Quality Engineering in Software Technology

delivered the welcome keynote on Trends in Innovation - Technologies, Networks and Clusters

- Vipul Kocher, Co-President of PT PureTesting Software Pvt. Ltd. and President of the Indian Testing Board presented a keynote on Software Testing – Present Problems and Future Solutions



*Vipul Kocher at Conquest 2009*

#### Awards

- Best Paper: Ferdinand Gramsamer for Useful metrics for managing testing
- Best Presentation: Harry Sneed for Value Driven Testing-The Economics of Software Testing

#### BusinessLink

This year, the conference's Partner country was India. BusinessLink was a unique event for which the ITB led a delegation of Indian companies. The participating companies were: - iVizSecurity, MindTree, NSE IT, PureTesting and Wipro.

The delegation met with various German and other European companies looking for business partners. They had the opportunity to meet Dagmar Wöhrl, Parliamentary State Secretary in the Federal Ministry of Economics and Technology. The delegation feels that the business relations forged there will bring benefits to the Indian as well as German companies.

To mark the event a press release was issued: India meets Bavaria – CONQUEST conference brings India to Nuremberg

A few excerpts from the press release are given below.

Despite the worldwide economical crisis India managed to strengthen its national market. At the same time Bavaria is regarded as one of the economically strongest regions in Europe. Hence it

## News...

News

is no coincidence that the two countries – dissimilar only on the first glance – are meeting at the CONQUEST 2009 conference, the “class reunion” of the international software developer scene, for an exchange of experiences.

Germany – India’s most important trading partner in the EU – and India – the unchallenged market leader as IT-outsourcing partner – meet in the creative atmosphere of an international conference, which continuously gives impulses for innovation. Making new and fostering already existing business contacts is the main focus of the conference, but



2

BusinessLink



it is also about understanding our similarities and differences in order to translate those impressions into new ideas.

Representatives of leading ICT companies from the partner country India will appear at the Georg Simon Ohm University of Applied Sciences. The expert public can look forward to contributions from: Vipul Kocher: PT Pure Testing Software Pvt. Ltd; Raja Anbazhagan: Cognizant Technology Solutions India Pvt Ltd; Rahul Verma: McAfee Software Pvt. Ltd.

## Upcoming Conferences

- **TESTWARES 2009** at Krakow, Poland.  
Conference schedule:  
October 20, 2009 – Workshops  
October 21-22, 2009 – TESTWAREZ conference  
<http://testwarez.pl/>
- **STPCon 2009** (Software Test & Performance Conference)  
Oct. 19-23, 2009  
Hyatt Regency Hotel, Cambridge, MA  
<http://www.stpcon.com/>
- **QA&TEST 2009** (8th International Conference on Software QA and Testing on Embedded Systems)  
Oct. 21-23, 2009  
Bilbao, Spain  
<http://www.qatest.org/en/>

**PNSQC** (Pacific New Software Quality Conference)  
Oct. 26-28, 2009

World Trade Center Portland  
Portland, OR 97204  
<http://www.pnsqc.org/index.php>

### TesTrek Toronto 2009

Sheraton Centre Toronto Hotel  
Toronto, Ontario M5H 2M9 - Canada  
<http://www.qaiworldwide.org/testrek/>

### Software & Systems Quality Conference

October 28-29, 2009  
Hilton Hotel Melbourne, Australia  
<http://www.sqs-conferences.com/au/index.htm>

### STAR West 2009

Oct 5-9, 2009  
Anaheim, CA  
<http://www.sqe.com/starwest/>

## File Fuzzing - Employing File Content Corruption to Test Software Security

by **Rahul Verma**, *QA Technical Lead*,  
McAfee Software (India) Pvt Ltd, Bangalore, India

*With an experience of 7 years in the industry, Rahul has explored the areas of security testing, large scale performance testing and database migration projects.*



*Currently, he is a QA Tech Lead in AVERT Labs at McAfee India. He is a core member of the McAfee Global Performance Testing Team and a Python trainer in the McAfee Automation Club.*

*Rahul has presented at several conferences and organizations including STeP-IN, ISQT,*

*TEST2008, Yahoo!, McAfee, Applabs, BWST-1 and STIG. His recent presentations were on the subjects of Fuzzing, Buffer Overflow Exploitation, Python, Performance Engineering COE, Web Application Security and User Behavior & Performance Perception Analysis (UBPPA). He is a part of the ISTQB working party (2009) as an author for Foundation syllabus. He got the Testing Thought Leadership Award at TEST2008 conference for his website [www.testingperspective.com](http://www.testingperspective.com), along with the Best Innovative Paper Award for his paper on design of Fuzzing Frameworks.*

### Abstract:

Fuzzing is about finding possible security issues with software through data corruption. The software in discussion might be a desktop application, a network daemon, an API or anything you could think of. Fuzzing is extensively used by security researchers and large scale product development companies. It has become an essential part of the Security Development Life Cycle in many organizations and is known to find a high percentage of security issues as compared to other techniques.

This paper focuses on file fuzzing, which is a special class of fuzzing dedicated to corrupting file formats. It is an easy-to-employ form of security testing and can be quickly put to work. Most of the software uses some or other sort of input in the form of files. The paper discusses the general uses and formats of such files and the data corruption strategies that can be employed.

The paper starts with a brief introduction of fuzzing and related concepts and then digs deeper into the area of file fuzzing.

**Audience:**

## Article...

The paper is meant for software testers, leads and managers with fascination for security testing. The paper helps them think about the technical aspects and validate the usefulness of implementing fuzzing.

### Area of Application:

Security Testing, Data Corruption Testing, Fuzzing

### Benefits:

Fuzzing is an automated testing technique for finding security issues and is known to find a high percentage of vulnerabilities reported.

### Issues and Challenges:

Making testers aware of this technique is the first challenge. Understanding it and implementing is the next one!

### Introduction:

Wikipedia defines fuzzing as: *“Fuzz Testing or Fuzzing is a software testing technique that provides random data (“fuzz”) to the inputs of a program. If the program fails (for example, by crashing, or by failing built-in code assertions), the defects can be noted.”*

Let's try to understand fuzzing a little further:

#### • Fuzzing as a security testing technique

As indicated by its definition, fuzzing is all about sending malformed data as input to an application to locate bugs. Such bugs typically result in crashes which after analysis can result in finding a vulnerability which makes the software exploitable in a certain way. A commonly discussed example of this sort is a buffer overflow vulnerability which can allow an attacker to inject shellcode in the application at run time and make it execute malicious code e.g. launching a remote shell.

#### • Fuzzers are anti-parsers

As said in the security world – “All input is malicious”. In terms of fuzzing, we try to generate all sorts of malformed/malicious data. The software employs a lot of parsing routines to interpret the input and take decisions e.g. buffer allocation, making calculations, type conversions, action execution etc. Fuzzing is all about breaking false assumptions or faulty code in such parsers. When malformed data is passed, it can trigger misallocation of memory, unintended interpretation of unsigned data in signed context causing buffer overflows and crashes. In code reviews, a problem may or may not map to a user input. In fuzzing, because such malformed data is directly tied to user input or variables that can be manipulated by a user, any such issues can be directly exploited.

#### • Fuzzing is essentially an automated testing technique

Fuzzing is essentially an automated testing technique. As the number of test cases executed

can quickly become very large, it is an art to carry out fuzzing with focus on areas which have the maximum possibility of locating vulnerabilities. It involves prioritization of tests based on analysis of the application, related protocol(s) and past vulnerabilities in similar applications.

### • Fuzzing employed for McAfee Anti-virus Engine

The McAfee Anti-virus engine is subjected to file fuzzing using an in-house built engine specific tool developed by Tony Bartram in C++. Fuzzing is also carried out targeting specific file formats in protocol-aware fashion. For the purpose, custom scripts are developed for samples generation using Python or Perl. Dedicated hardware rigs are used to carry out “data corruption” tests running round the clock for weekly builds.

## 1. Getting Started

Fuzzing has been considered to fit in the category of gray-box testing because of the nature of analysis and automation involved and can be executed as a black-box testing as well. The irony is that despite this fact, the term and the related implementation is mostly unknown to the software testers.

There is good chunk of fuzzing work which can be taken up by a software tester as against the general view of this being suitable only for security researchers. When tester locates a bug as a part of his usual job, he or she is rarely responsible for analyzing what piece of code is actually responsible for the bug. A tester usually logs the defect with test case details and his or her preliminary thoughts and analysis from outside the box. Fuzzing is no different, the only difference being the nature of the data that is submitted.

Fuzzing makes a software tester think beyond BVA and ECP, makes him redefine his view of input to the application and extends the traditional approach to testing, by bringing in a lot of possible test areas.

## 2. Tools of the Trade

There are a lot of tools which are available for carrying out fuzzing of different kinds namely File Fuzzing, Browser Fuzzing, Command Line Fuzzing, Environment Variable Fuzzing, Web Application Fuzzing, ActiveX fuzzing and so on. As the focus of the paper is File Fuzzing, the readers can specifically look at:

- **FileFuzz**, **SpikeFile**, **NotSpikeFile** (<http://labs.iddefense.com/software/fuzzing.php>)
- General-purpose frameworks like **Peach** (<http://peachfuzzer.com/>) and **Sulley** (<http://www.fuzzing.org/fuzzing-software>)

- Last but not the least an upcoming framework for the purpose – **PyRAFT** (<http://pyraft.sourceforge.net>), which is being actively developed by the author of this paper.

Knowledge about what already exists in the area of fuzzing helps one to understand practical implementations of different types of fuzzing. This helps in using or extending existing open source tools or coming up with altogether new tools and frameworks by analyzing the code and execution methodology of the existing ones

If one is specifically looking at file fuzzing and that too for a specific kind of files, one can look for tools rather than frameworks. Even from development perspective, writing a tool for a specific purpose is a lot easier than writing a general-purpose framework because of a lot of design considerations involved.

## 3. Pre-Requisites in terms of knowledge

Following are some of the concepts/technologies that one should be aware of before stepping into fuzzing:

- The essence of security testing and common input based attacks
- Using a Hex Editor
- A programming language of choice. Python is common in the world of fuzzing now. Older fuzzers were developed in C, but one can see some implementations in C#, Java and Perl as well.
- How architecture (Little Endian/Big Endian) impacts binary packing of data
- Programmatically dealing with Binary files (Reading and Writing Data) as per defined data types
- Knowledge of concepts and modules related to hashing and compression.
- Patience...a lot of it. When developing samples or understanding file formats, it's all about hex data and not fancy GUI based testing. One has to be very patient during the file format analysis phase of file fuzzing.

## 4. Purpose of Input files in a software

Software use input files for various reasons. Following are some of the most common uses that can be thought of:

- An Office Productivity Suite like MS Office, OpenOffice is all about creating and publishing files e.g. documents, spreadsheets, presentations etc.
- A Media Player uses media files of different formats to play audio/video
- A Browser users HTML/XML/CSS files to show web content
- An Anti-virus uses virus definition files to detect malware
- License files are used to determine validity/expiry of software
- Configuration files are used by web servers
- Temporary files are written to disk by software to be read at a later stage

### 5. Understanding File Formats

At high level we can classify the file formats into two formats: Text and Binary.

#### • Text Formats

These can take two common forms. One form can be typically seen in configuration files wherein a plain text file is used where each line corresponds to a configuration setting and has a key-value pair separated by “:” or “=” or “=>” etc. This format can also be seen in log readers where each line in a log is a comma separated content of various parameters. These days, XML is more popular in defining configuration files. It gives freedom of creating a much more complex structure e.g. nested definitions. Other text formats are HTML files which are again mark-up language based files with tags and attributes.

#### • Binary Formats

These formats are more complex to analyze and are not human readable. They can simply consist of binary packed data as per a set protocol or can be compiled data as well using proprietary compilers. One common format that binary files follow is a TLV format – Type-Length-Value, wherein type is based on identifiers recognized by the software, length gives the length of data that follows and then the value i.e. data. Typically, the type and length fields have fixed number of bytes allocated to them in terms of number of bytes and the data part is flexible and is dependent on the length field. Such records are put in sequence as shown in the below snapshot:

Type	Length	Value	Type	Length	Value
------	--------	-------	------	--------	-------

A very complex format of this nature is the SWF format which has the tag identifiers based on a tag record header that takes 2 bytes. Instead of consuming full 2 bytes, the format takes the first 10 bits as the tag identifier and the next 6 bits as tag length. Amazing, isn't it? This is true for short tag formats, and for long formats the approach is changed.

Till we know the format of a file, it is like a black box and the data corruption is also black-box corruption. As soon as you start understanding the file format and start data corruption taking care of dependencies of the file formats into consideration, it becomes a gray-box fuzzing. You need not know the code that deals with it, the high level logic of how the data is interpreted will suffice.

(To be Continued...)

*To be concluded in the next issue of this newsletter.*

### ITB introduces QAMP certification in India. ITB is the exclusive partner in India (National Authority) for QAMP.

QAMP, a new certificate for advanced vocational training of software engineers, supported by iSQI has successfully been introduced in co-operation with and according to the needs of the IT-Industry.

QAMP is targeted towards quality assurance managers, because they require competence in the entire development process. It tests the practical knowledge and the project experience of the certified employee. Theoretical and practical knowledge both have to be updated annually. In addition to a broad theoretical knowledge, there's a need to successfully pass three certification exams:

- Software Test (FL)
- Specified Module (AL)
- Requirement Engineering

#### Four Steps To QAMP

- Software Testing. (ISTQB Foundation Level)
- Specified Module .ISTQB Advance Level Test Manager/Configuration Manager/Project Manager/Secure Software manager/Software Architecture
- Requirement Engineering (IREB)
- 2 years of practical experience

For more information, please contact **Vidhi Baid** at [vidhi@indiantestingboard.com](mailto:vidhi@indiantestingboard.com)

#### Upcoming ISTQB Examinations

- Oct. 25, 2009 at Cochin
- Oct. 31, 2009 at Ahmadabad
- Nov. 15, 2009 at Pune, Hyderabad, Chennai, Trivandrum, Bangalore, Mumbai, Kolkata, Noida
- Nov. 22, 2009 at Coimbatore
- Nov. 27, 2009 at Madurai

For Online registrations, please visit [http://www.istqb.in/enrollment\\_form.php](http://www.istqb.in/enrollment_form.php)

***Last date for payment & registration is 10 days before the exam date***

## From the ITB Desk

From ITB Desk

6

New Affiliates

### Resources

**Advanced Software Testing - Vol. 1:  
Guide to the ISTQB Advanced Certification as an  
Advanced Test Analyst**

by Rex Black

Rockynook Computing, Paperback, Oct 2008

ISBN: 1933952199

**Advanced Software Testing - Vol. 2:  
Guide to the ISTQB Advanced Certification as an  
Advanced Test Manager**

by Rex Black

Rockynook Computing, Paperback, Dec 2008

ISBN: 1933952369

**Foundations of Software Testing: ISTQB  
Certification**

by Dorothy Graham, Isabel Evans, Erik Van  
Veenendaal & Rex Black

Cengage, Paperback, Feb 2008

ISBN: 1844809897

**Foundations of Software Testing: Fundamental  
Algorithms and Techniques**

by Aditya P Mathur

Pearson Education, 2008

Addison-Wesley Professional, 2007

ISBN-10: 81-317-1660-0

**Software testing Foundations: A study Guide for the  
Certified tester Exam**

- Foundation Level
- ISTQB Compliant

by Andreas Spillner, Tilo Linz, Hans Schaefer

Rocky Nook, 2007

ISBN: 9781933952086

### New Affiliates

**iEnergizer, Inc.**

Website: <http://www.iEnergizer.com>



**Qutesys Technologies Pvt. Ltd**

Website: <http://www.qutesys.com/>



**Safaltek Software Pvt. Ltd**

Website: <http://www.safaltek.com>



### Random Collection: Blogs on Testing

**Tester Tested**

<http://testertested.blogspot.com/>

**Thinking Tester**

<http://shrinik.blogspot.com/>

**Testing Perspective**

<http://www.testingperspective.com/>

**I.M. Testy**

<http://blogs.msdn.com/imtesty/>

**James Bach's Blog**

<http://www.satisfice.com/blog/>

**Quality Frog - Questioning Software**

<http://www.questioningsoftware.com/>



## William (Bj) Rollison

Test Architect

Microsoft, USA

[Bj.Rollison@microsoft.com](mailto:Bj.Rollison@microsoft.com)

*Bj Rollison is a Test Architect with Microsoft's Engineering Excellence group where he develops technical training curriculum, and teaches testers and developers at Microsoft various testing techniques and methodologies.*

*Bj started his professional career building custom solutions for small and medium sized businesses for an OEM company in Japan in 1991. In 1994 he joined Microsoft's Windows 95 international team. In 1996 he became a test manager in the internet division responsible for Internet Explorer 4.0 and several web-client products. He moved to Microsoft's technical training group in 1999 as the Director of Testing responsible for planning and organizing training for more than 6000 Microsoft testers. In 2003 Bj decided his passion was teaching and mentoring testers and became a Test Architect.*

*Bj also teaches software testing courses at the University of Washington, and sits on the advisory boards for testing certificate programs at the University of Washington, the University of California Extension Santa Cruz, and Lake Washington Technical College, and is a frequent speaker at international software testing conferences.*

**WIT** -- BJ, please tell us something about yourself, your interests, your hobbies and what do you do in your free time?

**BJ** -- I am a relatively private person regarding my personal life, but regarding my interests or hobbies I think most people know that I am an avid sailor. I taught scuba diving for 20 years, but now only dive occasionally; mostly to change the zincs on my sailboat. I also enjoy surfing and

windsurfing, and even raced windsurfers in Japan for 2 years. A few years ago I took up the sport of fly-fishing, and during the winter months I enjoy downhill skiing and the challenge of not breaking my legs. When I am not in or on the water, I enjoy organic vegetable gardening with my lovely daughter.

**WIT** -- Please tell us something about your new book "How we test software in Microsoft".

**BJ** -- We wrote the book because people outside of Microsoft kept asking - How do we test software at Microsoft?

This is not really an easy question to answer because Microsoft has several divisions with multiple product lines, and each product line pretty much has autonomy over how it ships its product to our customers.

The book attempts to illustrate the career path of SDETs at Microsoft, some professional skills and techniques that we teach to new SDETs at the company to help improve our testing efficiency and effectiveness, and also discusses some of our best practices that many of our groups have found useful. We tried to differ from most other books on testing by sprinkling real-life stories and experiences throughout the book to make it more enjoyable to read. Overall we are pretty pleased with the many positive reviews and feedback we have gotten, and we certainly hope that more testers read the book and find value in our story.

**WIT** -- Do you have plans to write another book?

**BJ** -- I thought about this for a long time. Writing a book is very time consuming. But, when I put together a program at the University of Washington for test automation I realized there are few books on practical application of well-designed, robust test automation. Most books on programming focus on developing applications, and the few books on test automation mostly focus on strategy. So, I am currently writing a series of books on automation practices using the C# programming language because it is easy to learn and very effective for everything from API testing to Win32 clients, web clients, web services, etc.. The goal is to make these books for practitioners filled with practical code samples, and suggestions for designing and developing robust automated test cases. The first book will focus on automated API testing, followed by GUI automation of Win32 clients, managed code clients, etc. I am trying to keep them down to about 150 pages and I hope to complete the first book in the series by the end of the year.

## Interview...

**WIT** -- You were in India last year for the [test2008](http://www.test2008.in/) in [http://www.test2008.in/] conference. Please tell us your experiences about the conference and your interaction with Indian testers?

**BJ** -- The Test2008 conference in India was pretty remarkable, and I was quite impressed with how the organizers planned and conducted workshops throughout the major cities and then gathered folks for the technical presentations at Delhi. I have travelled to India to work at Microsoft's facilities in Hyderabad several times in the past. I am always impressed by many of the testers I meet in India. Although many Indians initially appear to be rather reserved, I find them to be quite passionate and willing to share their perspectives and viewpoints after building a relationship of mutual respect.

**WIT** -- Please tell us something about your blog - **I. M. Testy** [http://blogs.msdn.com/imtesty/] and your website **TestingMentor** [http://www.testingmentor.com/]

**BJ** -- My blog is a venue to share some of my thoughts about testing, and the testing discipline. I thank the many folks who visit my blog, and I am sure they read several other blogs as I do. So, while there are probably a few folks who would love to read about my battles with slugs in my vegetable gardens I think most readers go to my blog to read my thoughts about our profession, and hopefully learn a trick or two they might find useful in their jobs.

The TestingMentor website is simply a way for me to distribute some job aids and tools that I have developed over the years. I am a big proponent of stochastic test data, and when I first began talking about it outside of Microsoft many people asked me if there were any available tools. There were a few tools, but few had the capabilities that are necessary in today's world. For example, there are a plethora of string generators available, but most of them are only capable of generating ASCII or ANSI characters, and the ones that can generate Unicode characters are incapable of generating strings with surrogate pair characters. So, I decided to design and develop to test data generation tools and a few other utilities and make them freely available to testers to use in either manual or automated testing. Right now I am working on an automation library TestDataGenerator that will wrap many of my existing and new data generators into a single DLL for use in automated test cases.

**WIT** -- From your work, it appears that you are promoting different perspectives of testing. What are these different perspectives?

**BJ** -- TI promote software testing as a professional discipline. The practice of software testing is difficult and requires a variety of skills and knowledge. Software testing is not simply about finding bugs, and many businesses realize that bug quantity does not equal product quality. Personally, I view our profession similar to medical doctors. Doctors spend years studying physiology, chemistry, biology, anatomy, and a myriad of other subjects, and they continue to learn new ideas and skills throughout their careers. Likewise I believe that professional software testers should have in-depth knowledge of the systems they are working on. Of course we know the really great doctors are not only experts in the science of their profession, they also have a great bed-side manner with their patients. Similarly I think great testers should also understand customer values but from a different perspective. To paraphrase Bill Gates, 'testers should understand how our customer's use the software and understand the bugs the customers run into, but the role of the tester is fundamentally different than that of the project/program manager or UX designer.' In my opinion, many people in the industry have bought into the notion that behavioural testing of software from a black box only approach in an attempt to emulate the end-user customer needs is good enough. What I have come to realize over the years, and many companies also understand, is that relying primarily on behavioural testing approaches is very costly to the business, and is no longer adequately meeting customer's quality expectations as the complexity of the systems and expectations of the customers increase. Essentially, I try to promote a pragmatic perspective that encourages testers to constantly improve their technical understanding of the systems they are working on, and to enrich their professional knowledge by continuing to study and learn about the testing discipline.

**WIT** -- What is your perspective on Open Source Testing tools?

**BJ** -- I recently read "Implementing Automated Software Testing" by Elfriede Dustin, Thom Garrett, and Bernie Gauf. I highly recommend this book, and I whole-heartedly agree with their statement "Taking advantage of freely available open source components or free/shareware that provides the required features allows for reuse and ease of additional feature integration, saving time, and money." Microsoft is also starting to adopt a more open policy and many previously internal tools as well as external contributions have begun to populate the <http://www.codeplex.com/> [http://www.codeplex.com/] open source project community. However, I also realize that many companies have strong policies about using open source tools and since some of my test data generation tools are used by such companies I don't add them to an open source community, but make them freely available for everyone to use. However, I am also currently collaborating with colleagues on two different projects that are on CodePlex, and I expect we will see a proliferation of testing tools and professionals from across the world collaborating on those tools in the open source communities in the near future.

## Interview...

**WIT** -- What is your opinion about the “Schools of Testing”? To which School of testing such as Process oriented, exploratory testing etc. do you belong? Or are you unhappy with this classification?

**BJ** -- Well I have blogged about this in the past at [End the segregation of the four schools of testing](http://blogs.msdn.com/imtesty/archive/2006/10/20/end-segregation.aspx) [http://blogs.msdn.com/imtesty/archive/2006/10/20/end-segregation.aspx] and [Schools of Testing Revisited](http://blogs.msdn.com/imtesty/archive/2007/06/01/schools-of-testing-revisited.aspx) [http://blogs.msdn.com/imtesty/archive/2007/06/01/schools-of-testing-revisited.aspx]

These days, I try not to dwell on such trivialities or meaningless debates that have absolutely no impact on the vast majority of professional testers in the industry or the advancement of our profession.

Although I do suspect Bret Pettichord intended the concept of ‘schools’ to provide insight into different approaches to testing that co-exist within any project (much like successful teams include people with different personality types), in my opinion the whole “schools of testing” notion has become a ridiculous attempt to segregate the discipline by some fictitious philosophical doctrine.

**WIT** -- What do you feel about Software Testing Certifications? How do you see them being used or abused, especially in relation to testing?

**BJ** - I think many people misunderstand certifications. People can “game the system” for many different types of certifications issued in various professional disciplines, and software testing certifications are no different. But, I think most people are overlooking personal responsibility for one’s own education and personal development. Simply showing up for some class and passing an exam is simply a formality of many certifications. It is how the individual applies themselves to the learning experience, and acquires new skills and knowledge that differentiates a person who is truly committed to a discipline versus a hobbyist or someone who simply wants to collect wallpaper.

Currently, I think many companies are using software testing certifications for 2 purposes. First, similar to a college degree is illustrates to the company that you took time and passed the exams to become certified. For some companies a certification is a flag that “puts the foot in the door,” and could be the difference between a resume that is reviewed more carefully versus one that becomes circular file (trash bin) fodder. All companies use various types’ mechanisms to flag resumes, and software certifications are simply one way a company might choose to identify potential candidates. Secondly, the certification process provides a common professional jargon within the workplace. For example, when I speak to someone who is certified about a particular testing technique such as equivalence class partitioning I don’t have to spend the first 30 minutes of the conversation explaining how the principles of mathematical set theory is applied in software testing which makes the overall conversation much more productive.

**WIT** – Can you tell us something about Microsoft’s Engineering Excellence group? What work do you do there?

**BJ** - The Engineering Excellence group was established in 2003 by Bill Gates with a mission to revitalize our engineering teams through strategic leadership and community building, technical career development, and identification of best practices around the company to improve our internal practices and processes. The team is mostly staffed with senior engineers who have been at the company for several years working on several different projects, and who are recognized as subject matter experts in one or more technical areas by their peer group at Microsoft.

As a Test Architect in the group I am primarily responsible for designing, developing and teaching technical courses for the 6000+ testers at Microsoft. I and others on our team also collaborate with divisional leadership teams, and provide internal consulting on the adaptation of best practices into project cycles. More recently, my role includes building community both inside and outside of Microsoft, consulting with premier customers on our internal practices, and partnering with universities to incorporate adoption of software testing curriculum in computer science programs. Although I don’t directly work on any specific product, I think I have one of the best jobs at Microsoft because I get to meet so many brilliant and talented people from across the company every day, and I get to work on challenging projects that will ultimately impact how we design, build and ship world class software.

**WIT** -- What, in your experience are the areas of improvement for most testers and how can they improve?

**BJ** – Design! Many testers I’ve met are very good at breaking software or finding bugs in software. But really, finding bugs in software is really not that hard! My 7 year old daughter finds bugs in software...and let me tell you...she lets me know when she does. However, software testing is much more than bug hunting. Software testing is a creative and intellectually challenging discipline that requires a myriad of skills. The value of a software tester is not in his/her knack to find bugs, but in his/her aptitude to understand the systems they are working on and his/her ability to design effective test cases that evaluate tenet and non-tenet attributes of the product to provide important information to the project’s management team. In my experience, many of the test cases I’ve reviewed have no clearly defined purpose, are too restrictive in their execution, or are so vague they cannot be handed off for another party to execute. Test cases are records that may be required for several reasons such legal or governmental requirements. For software that has a protracted shelf-life, software maintenance requires repeatability to prevent regressions in service environments and effective test cases that are reused in the

## Interview...

Interview

10

Contact Us

maintenance cycle and the next release can reduce overall production costs. Whatever the reason for test cases, designing effective tests is a critical skill that professional testers should master as they progress in their career. But, one thing is for certain; the less a tester understands the inner workings of the systems they are evaluating, and the external needs or values that are driving the project the less the tester's ability to design effective tests from a variety of perspectives. Unfortunately, there is no simple solution to learning how to design effective test cases. I personally study various testing methodologies, I study how bugs can manifest themselves in different ways and look for common root causes, and I study various design patterns. I also review a lot of test cases from across the company, provide feedback, and help redesign test cases so they are more comprehensive.

**WIT** -- What big changes do you see in the industry and what will be the potential impact on software testers?

**BJ** -- The proliferation of Agile concepts in software development lifecycles shouldn't be a surprise to anyone who is paying attention in the industry, and many testing teams are feverishly trying to figure out how to wedge their existing testing strategies into an agile lifecycle. If the primary testing strategy has relied on exploratory or behavioural testing to find bugs many teams are discovering that this testing approach is simply a bottleneck in the process. This is especially true on projects with close connections to the customer where the customer is providing constant feedback during the process, and the iterations between releases to the customer is relatively short. The agile environment requires testers to work much more closely with the developers. Rather than an adversarial relationship testers and developers will work in partnership and find way

to identify problems sooner in the lifecycle. Testers can partner with developers to strengthen unit tests, participate in code reviews and inspections, and better understand where to focus their 'testing attention' based on a more in-depth analysis of the system. There is also a tremendous amount of work throughout the industry focusing on static and dynamic analysis tools. We will see an increase in tools designed to identify defects sooner, and also help developers prevent certain classes of defects. I also suspect we will see greater use of coverage analysis and path analysis tools by testers to identify weak or under-tested areas of the code and design tests to help reduce risk in those areas. I also think we will probably start seeing tools that reveal properties and class details without the source code similar to reflection in managed code that can help testers conceive of new or different tests. Essentially, the testers of the future will require a much more technical skill set and greater understanding of the systems they are working on. Of course, another impetus of change is the downturn in the economy is the fact that many companies are demanding more with less; meaning those companies are demanding testers who are able to engage in upstream defect detection and prevention practices, design and develop effective automation, collect and analyze important information (including customer feedback) and incorporate that information to improve processes and practices, and who are perceived as valuable assets to the company.

*We thank [WhatIsTesting.com](http://WhatIsTesting.com) for allowing us to publish this interview, which is originally available at:*

*[www.whatistesting.com/forum/phpBB3/viewtopic.php?f=35&t=229&start=0](http://www.whatistesting.com/forum/phpBB3/viewtopic.php?f=35&t=229&start=0)*



Head Office:

INDIAN TESTING BOARD  
A-24, First Floor, Sector-59  
NOIDA - 201301(U.P)

email: [info@indiantestingboard.com](mailto:info@indiantestingboard.com)

Phone: 0120-4355270

Branch Office :

INDIAN TESTING BOARD  
1/2-1, 1st MAIN ROAD, 5th CROSS JUNCTION  
PALACE LOOP ROAD,  
VASANTH NAGAR  
BANGALORE - 560052  
Mobile : 09343494617

Accreditation Managers :

**Vinay Baid**

Cell: +91 9310373004

e-mail: [vinay@indiantestingboard.com](mailto:vinay@indiantestingboard.com)

**Pravesh Pugalia**

Cell: +91 9343494617

e-mail: [pravesh@indiantestingboard.com](mailto:pravesh@indiantestingboard.com)