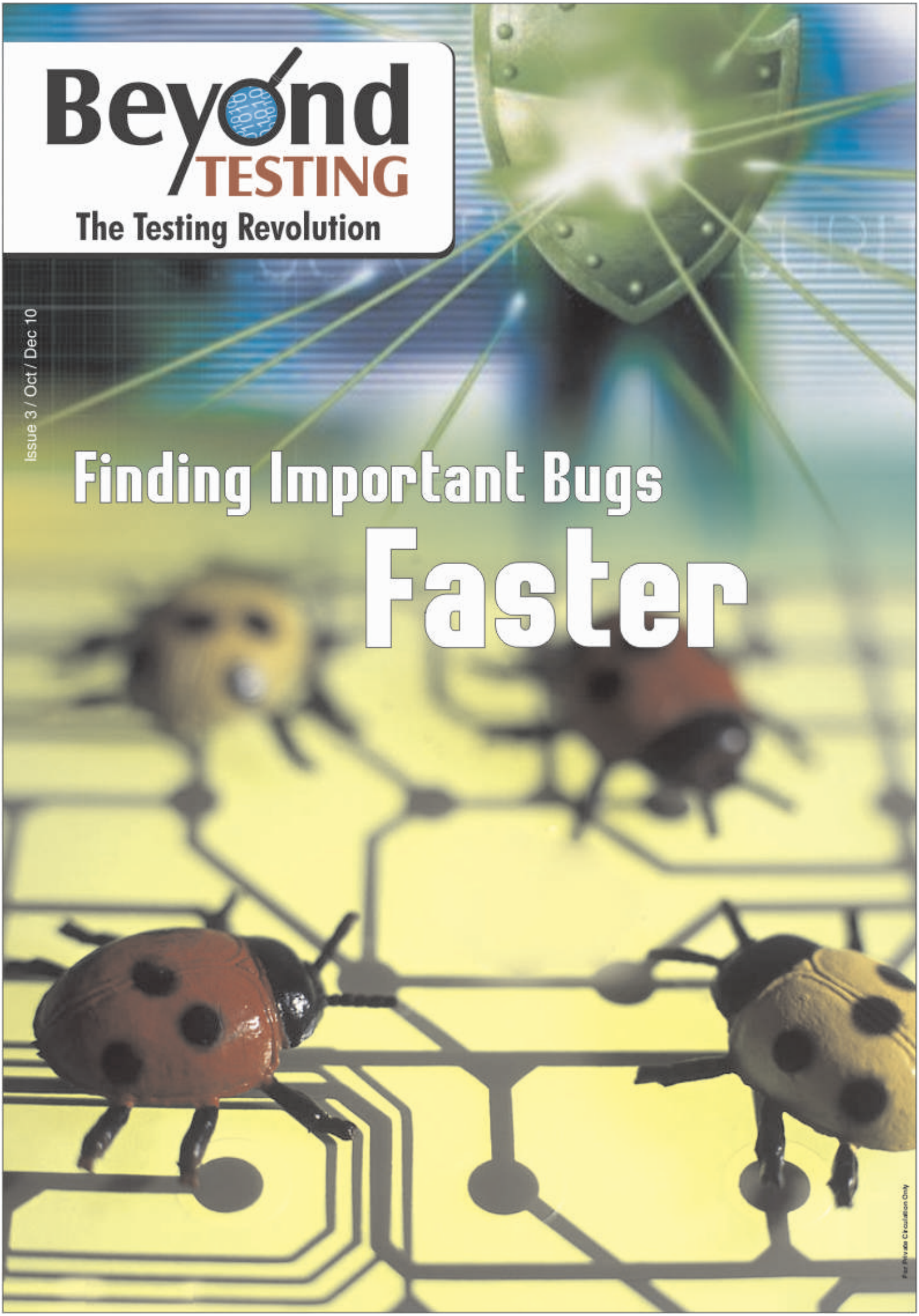
Beyond TESTING

The Testing Revolution

Issue 3 / Oct / Dec 10

Finding Important Bugs Faster



The Journey of Thousand miles starts with a single step

Take the Step Today

SEED Infotech Ltd. is Pune's leading Training, Consulting and Staffing Company, offering a wide array of solutions customized for a range of key verticals and horizontals in the IT industry. From customized corporate training in IT, Project Management and Soft-skills to Strategy Consulting for the implementation of the right tools for enterprise wide planning, software development and testing, SEED Infotech's service portfolio encompasses the entire range of solutions required by the IT industry. We have 33+ branches all across India.

- 16+ years of Experience in Technology Training, Staffing and Consulting
- 600 dedicated and highly skilled IT Professional with 200+ in house Trainers
- 1,80,000+ Students and Professional trained so far
- 1,00,000+ Sq. ft. of Training Infrastructure
- Courses mapped with Industry Requirement
- State-of-the-art IT laboratories and communication set-up
- 33+ Training Location across India
- Strategic tie-ups with Global Technology leaders

Short Term IT Courses

- C, C++
- .NET
- Java/J2EE
- Oracle
- Linux
- PERL
- PHP
- IT for 10th & 12th

Hardware & Networking Courses

- A+
- N+
- MCSA
- CCNA
- SCHANSA

English & Foreign Languages

- Spoken English
- Japanese
- Chinese
- German
- French
- Maths & English Aptitude
- Personality Development

Job Oriented Diploma Courses

- IBM SEED - Diploma in Software Testing
- IBM Tivoli Storage Manager
- SCTS - SEED Certified Technology Specialist (Java / .NET Track)
- Technical Writing
- SPIC - Software Professionals' Incubation Center
  - **Java / .NET - Software Testing - Hardware & Networking - Tivoli Admin**

SEED Tutorails

- English
- Maths
- Science
- Sanskrit
- German
- French
- Marathi
- Hindi
- Computer Science

SEED Infotech Ltd.

'Panchasheel', 42/16, Erandawana, SEED Infotech Lane,
Off Karve Road, Pune - 411004. **Tel.:** 020-25467484 / 9850885179
Email: info@seedinfotech.com | **www.seedinfotech.com**

seed™
beyond the obvious

- **Aundh :** Tel.:020-25898486 / 9881204114
- **Camp :** Tel.:020-26138983 / 9881198738
- **Chinchwad :** Tel.: 020-27468307 / 99229 33327
- **Hadapsar :** Tel.: 020-32533550 / 9921555200
- **Satara Road :** Tel.: 020-24226393 / 9225645641
- **Sinhgad Road :** Tel.: 020-24353384 / 9960636220

Chief Editor
Jayesh Ingale

Board of Advisors
- Narendra Barhate
- Jayesh Ingale
- Milind Limaye
- Gireendra Kasmalkar
- Rajesh Vartak
- Rajesh Agrawal
- Subodh Hawa

Production Team
Art & Graphic Designer : Smita Khade
Printing & Publishing : Kedar Kakade

Feedback
feedback@seedinfotech.com

For Article
article@seedinfotech.com

Facebook
http://facebook.com/beyondtestingmag

Twitter
http://twitter.com/beyondtesting

Printed & Published by
Avani Publication

36 pages including cover

Column

Security Testing using SQL Injection

A Primer on Usability Testing

Cover Story

Scenario Testing - Tests that matter Most to Customers

Finding BUGS Faster - Tips and Tricks

Contents

Chief Editor

We are approaching towards last couple of days of the year 2010. So it is time to make resolutions for the New Year. By making resolutions, we show our commitment towards a particular individual or a project.

Along with our personal commitments, let us make a commitment to our customers. We, as testers will make our customer life better, by providing our management vital information of customer pain areas. By providing such information, we are helping management to take appropriate decisions. How do we do that? Simple, by identifying and reporting important defects faster. Important for whom? Obviously, the customer.

This issue of Beyond Testing focuses exactly on this subject. We discuss Scenario Testing, which focuses on developing user centric test scenarios. We also offer you tips and tricks to find bugs faster. Apart from that we have columns on usability testing and security testing. For aspiring testers and newbies, we have student corner which covers topics like Black Box techniques, Quality Assurance and Quality Control. Also, take our final word on importance of domain knowledge in software testing.

And do not forget to enjoy "Testing Humour", where bugs are secretly talking about us (I mean testers).

We also encourage testers to write an article for "Beyond Testing" magazine. So join the "Testing Revolution" by e-mailing your article at article@seedinfotech.com and make yourself proud.

Wishing all the readers a New Year filled with new hope, new joy and new beginnings…

Jayesh !ngale

Security Testing using SQL Injection

Vishal Shah

We are living in an era where technology is being widely used and people are becoming more tech-savvy. Internet and online services are playing major role in today's world. Online banking, online shopping, social networking, telemedicine are now a part of day to day life, and we are happy to have everything just a click away.

Wait a minute… have we ever thought that how much secure our applications are? Are we really sure that the applications we develop and deliver; keep data secure? How confident are we that our information or private data is secure? Have we really verified that the defense mechanisms (if any) employed is working correctly? If not, it is never too late.

Everything in this world has some pros and cons, technology is not an exception. Some notorious people are using technology for destructive purpose, for example hacking is a hobby for some people. Let's take some known real-world examples.

On November 1, 2005, a high school student breaks into the site of a Taiwanese information security magazine from the Tech Target group and steal customers' information

On January 13, 2006, Russian computer criminals broke into a Rhode Island government web site and allegedly stole credit card data from individuals who have done business online with state agencies

On March 29, 2006, Susam Pal discovered an SQL injection flaw in an official Indian government tourism site

Security attacks may be noticed only after the unwanted parties break in, at which time it may be too late, and that is why security testing becomes top priority.

Formal Definition of Security Testing

Security testing is a process to determine that an information system protects data and maintains functionality as intended (Ref. Wikipedia).

Security is one of the most crucial forms of testing and may be most complex. Practically it is not possible to carry out extensive security testing but the testers can analyze all possible risks and major areas where a security attacks could be successful.

This is where OWASP* Top 10 attacks come as handy reference. OWASP lists SQL injection has one of the most popular and deadliest attacks among hackers.

What is SQL Injection?

SQL Injection is a code injection technique that exploits a security vulnerability occurring in the database layer of an application. Attackers take advantage of the fact that programmers often chain together SQL commands with user provided parameters, and can therefore embed SQL command inside these parameters. The result is that attacker can execute those SQL queries on the backend database server through web application.

Most of the applications use some type of database and an application under test might have a user interface that accepts user inputs to retrieve some data or to save the data in database. Some of the user inputs might be used for framing SQL statements that are executed on the database. Some notorious user could provide unexpected inputs to the application that are used to frame and execute SQL statements on the database. If this happens then the attacker could log in to your application, could take control of the database, and could access confidential information of other user.

Let's take a simple example of login screen.

Application accepts username and password from the user and passes that to the SQL statement

```
SELECT * FROM users WHERE u_name = '"&struname&"' AND password =
'"&strpassword&"';
```

If tester enters username and password as admin & admin123 respectively then the query will be like this

```
SELECT * FROM users WHERE u_name = '"&admin&"' AND password =
'"&admin123&"';
```

The malicious user can enter admin'-- as struname & no strpassword, then the same SQL statement will be like

```
SELECT * FROM users WHERE u_name = '"&admin&"'-- AND password =
'"&admin&"';
```

Now the part of the SQL statement after admin will be treated as comment. Please refer to table 1.0 for more SQL injection Characters. If there is any user with username as admin then the application could allow the user to log in as admin.

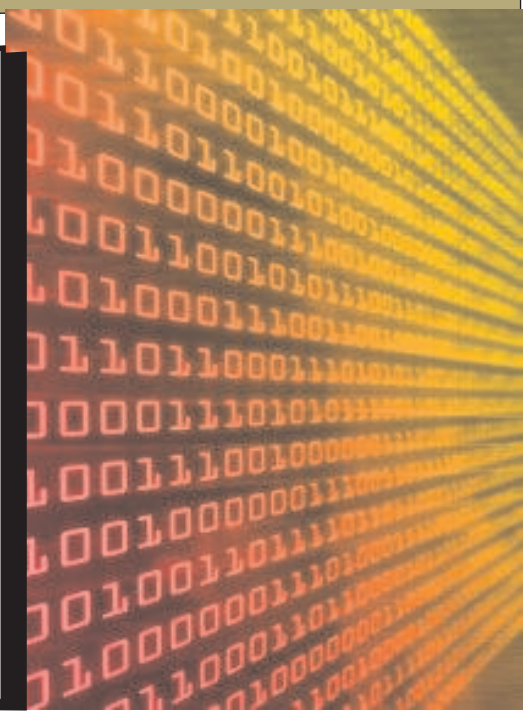
Attacker can try other common user names like, administrator, sysadmin, demo, default etc. If none of these exist then attacker can enter "admin" or 'a'='a' as username then the query would be like

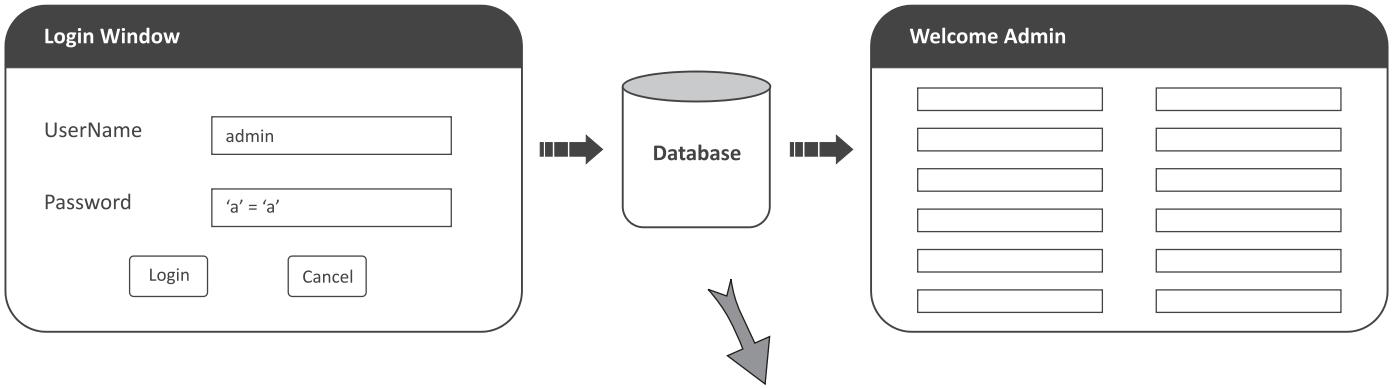
```
SELECT * FROM users WHERE u_name = 'admin' or 'a'='a' AND password =
'admin123' or 'b'='b';
As 'a'='a' will always be true, query will return all rows in the table.
```

Isn't it serious?

But you can prevent this kind of attacks by using following prevention techniques.

- Parameterized statement – User input must not directly be embedded in SQL statement, instead parameterized statement must be used
- Use of Escape character – Many attacks can be thwarted by simply using the SQL string escaping mechanism '\' and "\"






```
SELECT * FROM users WHERE u_name = 'admin AND
password = 'b'='b';
```

Few things to be consider before using SQL injection

1. SQL injection problem should be tested only in test environment
2. Database or table's backup should be taken before doing SQL injection.

Security testing is too vast to be covered in a single article but I have tried to share some essence of it. In next issue we will see what is Cross site scripting.

Till then test for SQL injection attacks, and deliver secure applications to customers.

SQL Injection Characters

• 'or"	character String Indicators
• -- or #	single-line comment
• /*...*/	multiple-line comment
• +	addition, concatenate (or space in url)
• ||	(double pipe) concatenate
• %	Wildcard attribute indicator
• ? Param1 =foo& Param2=bar	URL Parameters
• PRINT	useful as non transactional command
• @variable	local variable
• @@variable	global variable
• waitfor delay '0:0:10'	time delay

References

- Security testing of web applications against SQL Injection, explained with simple examples – By Inder P Singh.
- Wikipedia
- OWASP, Open Web Application Security Project

A Primer on Usability Testing

Preeti Sarawade

What makes a product/service "Usable" ?

More and more customers are getting "addicted" to a plethora of products/services. It is very difficult for us to imagine our lives without them because our life has become very easy and comfortable because of them, we owe them!!

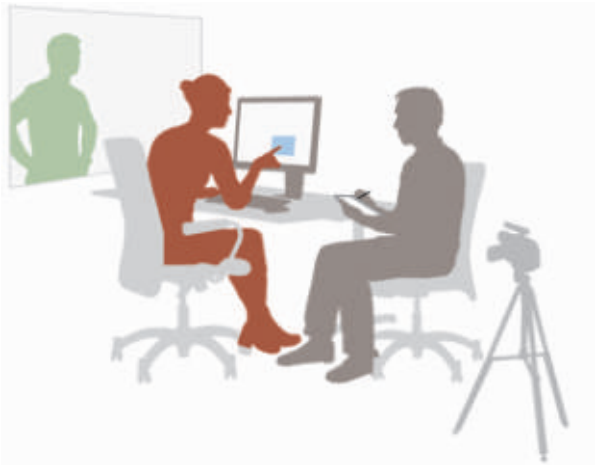
A software product is of no use if it remains on the shelf and does not fulfil the needs of a customer.

Customer will be happy only when he finds his requirements and expectations completely satisfied when using the product.

Just imagine some situations of poor user experiences with applications/softwares –

1. Windows Live -
   Imagine a situation where one has to send a large page text content to his counterpart on Windows Live but he/she just cannot send all the text content in one go, and has to break it down in set of 400 characters and send it one after another in sequence.
2. Error handling issues in a software -
   A software or a program doesn't allow a user to restore the data that has been lost due to some mistake and the data was highly confidential ?
3. Error messages -
   What if a software doesn't prompt appropriate warning message before a critical operation ?

All these situations ultimately lead to human frustrations.



Products/Services which are usable have reduced level of such frustrations.

According to Jacob Nielson, to be usable, a product or service must consider, at a minimum, these five basic dimensions:

- Learnability: The system should be easy to learn. The system should be designed in such a way that it encourages new users to get started easily. Should invite exploration, not frustration.
  Typical frustration – "I never understand the questions asked in the registration form"
- Efficiency: Experienced users would now expect improved productivity.
  Typical frustration – "I need to enter the same information so many times"
- Memorability: It should be easy for users to remember the preconditions, steps, data/information required for executing a said feature.
  Typical frustration – "I need to type in the details in a particular format, I need to remember the format every time I use this feature"
- Errors: The system should be designed in such a way that users make few errors and in case if they do, system can easily recover from that error.
  Typical frustration - "I have done wrong settings, can I revert back to default"
- Satisfaction: The system should be pleasant to use so that users are subjectively satisfied when using it; they like it. Typical frustration – "I am colour blind and I am not able to change the colours of the user interface"

The relevance and importance of these dimensions depends on the context and the type of users. For example, in applications such as Gmail, Facebook, learnability should be the focus, while for complex systems like medical, aerospace, stock trading applications error tolerance, efficiency should be the centre of attention. Similarly learnability may be focus for first time users, efficiency may be important for experienced users.

What is Usability Testing -

Usability testing is a research tool, with its roots in classical experimental methodology. It tests the degree of user satisfaction and compliance to user's requirements with respect to a software product.

The range of tests one can conduct is considerable, from true classical experiments with large sample sizes and complex test designs to very informal qualitative studies with only a single participant.

Each testing approach has different objectives, as well as different time and resource

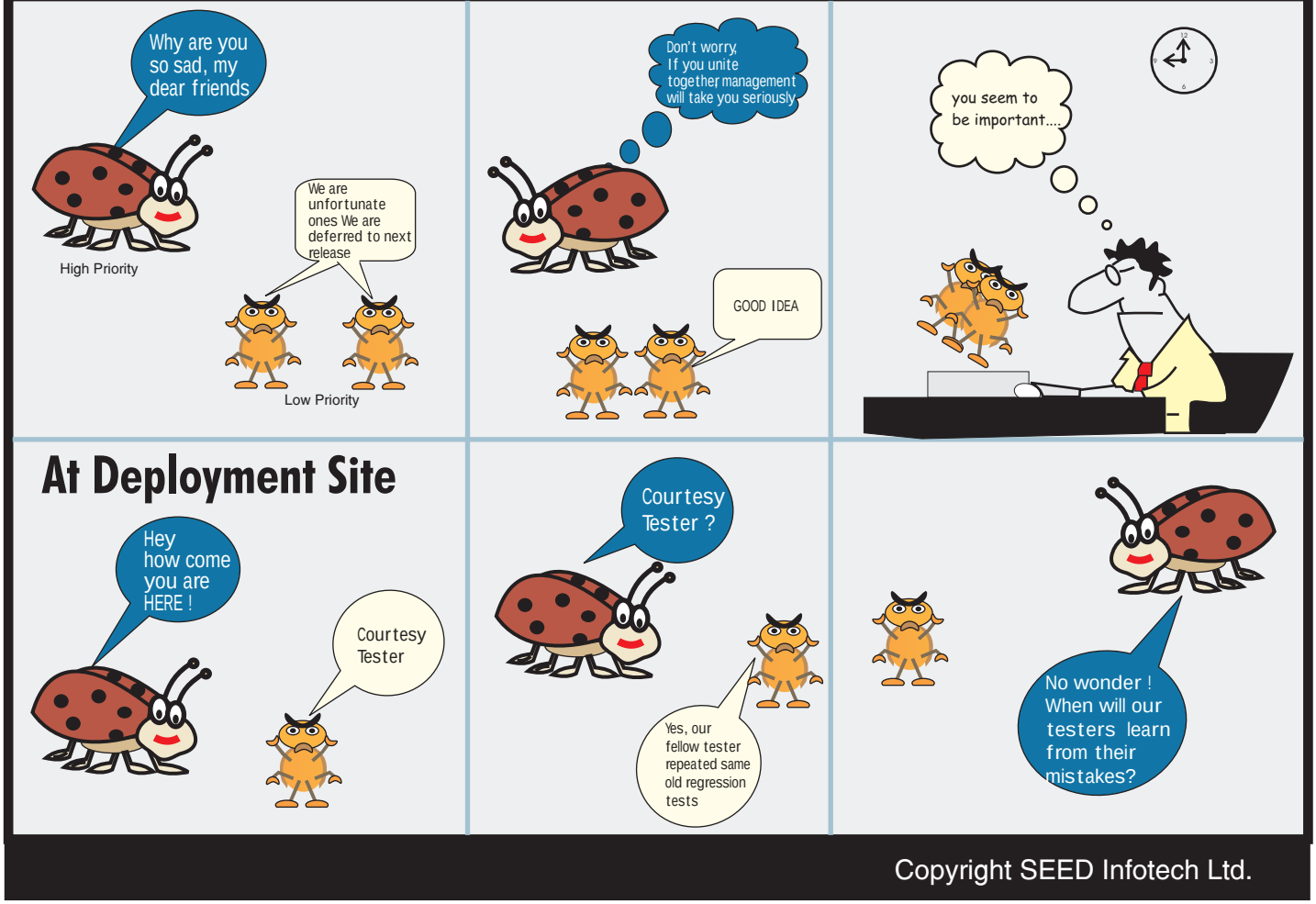
requirements.

Challenges in Usability Testing -

True usability is invisible for an end user.

When something is missing or if there is a problem in using the system, it can be judged by the user. But when the product is doing well, that will go unnoticed. That means, usability becomes an issue only when something is lacking or missing.

Testing Humor



Defined Usability Goals and Objectives -

Designing a product to be useful must be a structured and systematic process, starting with high level goals and moving to specific objectives.Objectives should not be too simplistic and at the same time they should not be unrealistic. The objectives may be

- Effective to use
- Efficient to use
- Safe to use
- Easy to learn
- Easy to remember how to use

Now, lets move on to the methods and techniques used for Usability testing.

Participatory Design –

Participatory design employs one or more representative users on the design team itself. This approach involves the end user into the heart of the design process from the very commencement of the project.

Even the emotional reactions of the end user are tapped.

A variation on this technique is to arrange short, individual workshops where users, designers, and developers work together on an aspect of design.

For example, users, designers, and engineers using workable models, work together to determine the best size and shape for the product.

Surveys –

One can understand the preferences of a broad base of users about an existing product using this technique

Surveys can be used at any time in the lifecycle but are most often used in the early stages to better understand the potential user.

Again, asking people about what they do or have done is no substitute for observing them do it in a usability test.

Expert or Heuristic Evaluations :

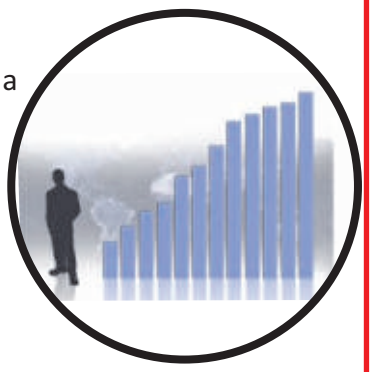
Expert evaluations involve a review of a product or system, usually by a usability specialist or human factors specialist who has little or no involvement in the project.

The specialist performs his or her review according to accepted usability principles (heuristics) from the body of research, human factors literature, and previous professional experience.
The viewpoint is that of the specific target population that will use the product.

A ''double'' specialist, that is, someone who is an expert in usability principles or human factors as well as an expert in the domain area (such as healthcare, financial services, and so on, depending on the application), or in the particular technology employed by the product, can be more effective than one without such knowledge.

Conclusion :

To conclude, usability testing is a systematic process with defined goals and objectives, methods and techniques which when combined with functionality testing provides us excellent idea about usefulness of applications.



When systems match user needs, satisfaction often improves dramatically. In a 1992 Gartner Group study, usability methods raised user satisfaction ratings for a system by 40%. So what are you waiting for... ?

* Jakob Nielsen has been a leading figure in the usability field since the 1980s and he is author of bestselling book, Usability Engineering (Nielsen, 1993).

References :

Handbook of Usability Testing, by Jeffrey Rubin and Dana Chisnell

Some of our visionaries failed to see the future. Hard to believe !!!. Check some of these...

While theoretically and technically television may be feasible, commercially and financially I consider it an impossibility, a development of which we need waste little time dreaming.

- Lee DeForest, American radio pioneer, 1926

Television won't be able to hold on to any market it captures after the first six months. People will soon get tired of staring at a plywood box every night.

- Darryl F. Zanuck, Head of 20th Century-Fox, 1946.

This `telephone' has too many shortcomings to be seriously considered as a practical form of communication. The device is inherently of no value to us.

- Western Union internal memo, 1878

Airplanes are interesting toys but of no military value.

- Marshal Ferdinand Foch, French military strategist and World War I commander.

There is no reason for any individual to have a computer in their home.

- Ken Olson, President of Digital Corporation, 1977

The Internet will catastrophically collapse in 1996.'

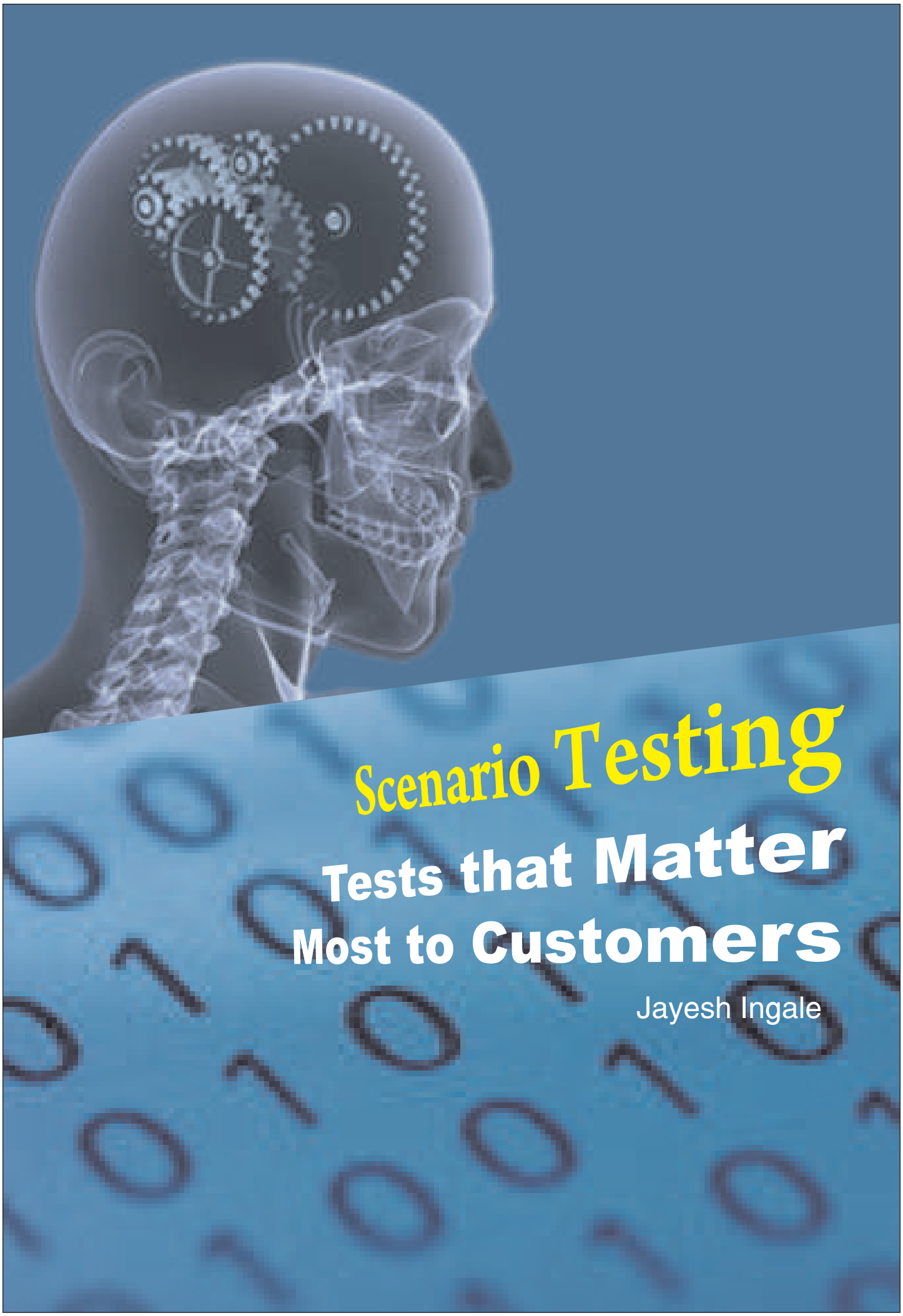
- Robert Metcalfe, internet inventor

I think there is a world market for maybe five computers.

- Thomas J. Watson Snr., IBM Chairman, 1943

There is no hope for the fanciful idea of reaching the Moon because of insurmountable barriers to escaping the Earth's gravity.

- Dr. Forest Ray Moulton, University of Chicago astronomer, 1932.


Scenario Testing
Tests that Matter
Most to Customers
Jayesh Ingale

Last month, my friend met with an accident. He suffered minor injuries. He was rushed to a nearby hospital. The doctor examined my friend and concluded that he needs to be hospitalised for one day. During his period, he would undergo a couple of tests.

I completed all the formalities like registration and payment of advance fees. Since the duration of hospitalization was less than 24 hours, my friend was not eligible for insurance claims. We selected a private room. So far, things were looking fine.

While the nurses did the tests, my friend looked relaxed. Finally the tests were done and we were waiting for the results of the tests. By evening, we had the reports in our hands. And what!!! The tests were positive. Doctors explained me that my friend was suffering from a rare disease. We were immediately asked to shift to Speciality Care Centre belonging to the same hospital. The clerk made appropriate transfer from private room to special room in the Specialty Care Centre.

Now since hospitalization was extended, we were now eligible for insurance claims. I explained the clerk regarding this change to which he replied "Sir, do not worry, I will change the status". Relieved with his response, we proceeded to Speciality Care Centre.

The relief was short lived. The moment we entered the room, there was another surprise in store for us. It was already occupied by another patient. Investigations revealed that the same room was allocated to two patients. We finally managed to get things right and settled ourselves in another room.

After a couple of days, the attended dropped a bill for 1 day treatment. I was shocked. Since we were insured under cashless mechanism, there was no reason for us to pay. Also it seemed that the previous transaction was not completed properly.

I requested the clerk to rectify the problem. The clerk as usual, politely accepted my request and the bill was cancelled.

What do you notice in this short version of my narration? Aren't there lot of parameters in this scenario? Aren't these situations inter-related to each other? How many transactions are there? What can you say about these transactions? Does this scenario reflect complexities? Is it motivating? Do you feel that such scenario can really happen?

Let me put it in order for you,

1. Patient registration for 1 day private room
2. Advance payment
3. Change the private room to room in Specialty care centre
4. Check of room availability
5. Change number of days of hospitalization
6. Cancel the current monetary transaction
7. Generate new transaction
8. Update details of insurance with cashless mechanism
9. Return advance payment

Getting You Started

The above brief illustrated a real life scenario wherein I was a stake holder of the hospital management system. So can you think of hypothetical situations for the software you are testing?

Let us take up couple of scenarios for a mortgage system...

Scenario 1 - Change EMI

Let us first define objective. The objective is to verify that system performs the transactions correctly for multiple EMI variations. We then design a scenario keeping this objective in mind.

1	Customer applies for a mortgage loan amounting for Rs.50 lacs, 20 years, 12% interest with EMI of Rs 30,000
2	Bank approves the loan
3	Bank deducts first instalment (EMI) of Rs 30,000
4	Customer increases EMI to Rs 39,000
5	Customer makes Rs 1 lac prepayment and request for change in EMI
6	Customer decreases EMI to 35,000
7	Customer pays entire balance amount after 5 years and closes the loan

Scenario 2- Change loan scheme

In this case, objective of stakeholder is different. The objective is to verify that application allows change from existing loan scheme to a teaser loan (offering lower interest by the same bank) after a certain period and allows reverting back to previous loan scheme

1	Customer applies for a mortgage loan amounting for Rs.50 lacs, 20 years, 12% interest with EMI of Rs 30,000
2	Bank approves the loan
3	Bank deducts first instalment (EMI) of Rs 30,000
4	Customer changes his scheme to teaser loan scheme offering 8% interest
5	Bank charges customer penalty of 1% on loan outstanding
6	Bank keeps EMI same, but reduces tenure by 5 years
7	Customer after 5 years reverts back to previous loan scheme
8	Bank charges customer penalty of 1% on loan outstanding
9	Bank keeps EMI and decreases loan tenure by 1 year

A scenario is thus a story that describes a hypothetical situation. It helps a person to understand a complex situation. In testing, you check how the program copes with this hypothetical situation.

How Scenario Testing is different from other types of Testing :

1. In other types of testing, the tester will meticulously follow the manual or specs (Software Requirement Specification / Functional Requirement Specification). Scenario testing requires a different approach. You not only explore the application by interacting with it but perform own market research in order to learn more about the application you are testing.
2. In other testing types, tester performs exhaustive testing while in scenario testing, tester performs useful tests.
3. The tester requires superior imagination and creativity skills for performing scenario testing as compared to other types of testing
4. Other testing types focus on requirements to build up tests, while in scenario testing, the focus goes beyond requirements. The tests (scenarios) are built on objective / interest of stake holder

Characteristics of Good Scenario :

According to Cem Kaner, a test scenario has the following characteristics

a) a story based
b) motivating,
c) credible,
d) complex,
e) easy to evaluate.

As we have seen in examples above, the story depicts different situations and their inter-relationships. The scenario becomes motivating if it is important for the stakeholder and, if a defect does get detected, has significant impact on the stakeholder. By credible, we mean that the stakeholders believe that scenario can happen in real life and we are not taking of a mere possibility. Typically scenarios have lot of inter linked transactions and hence are complex. So it is equally important for the tester to make sure that it is easy to evaluate the test results.

Benefits :

Majority of critical/high priority defects are discovered in scenario testing

One of the most important benefit of scenario testing is it helps us to find defects which are really important for the customer. This type of testing focuses on stakeholder objective and interests.

Test Scenarios also ensure better coverage and can be very efficient as a single scenario can capture several specs.

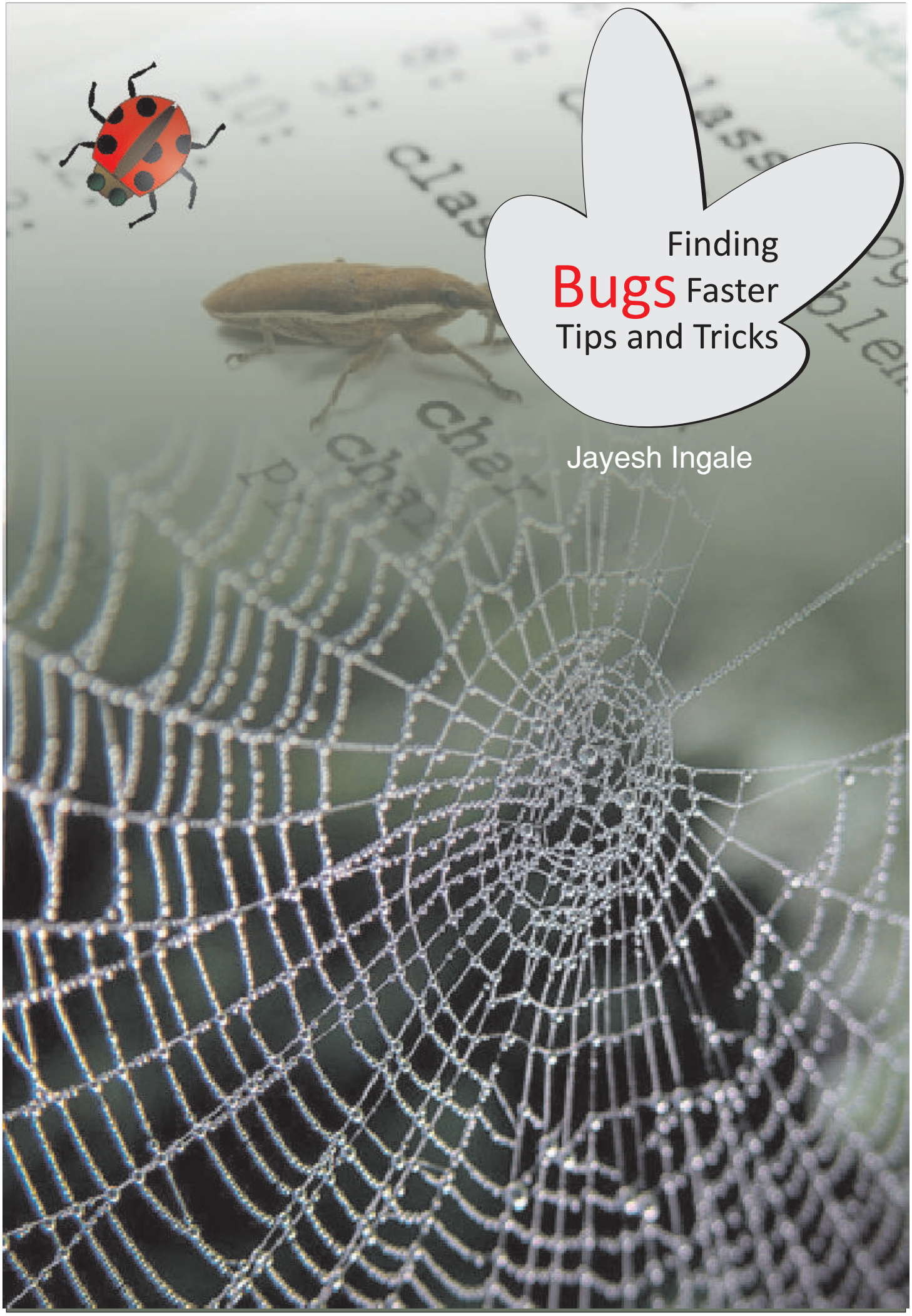
A scenario models real use of the application

The scenarios also help in discovering requirement problems. Although these issues may irritate internal stakeholders, their resolution in early stages of software development can reduce lot of frustrations later on in the project.

References :

An Introduction to Scenario Testing, Cem Kaner, Florida Tech, June 2003

Wikipedia – Scenario Testing


Finding
Bugs Faster
Tips and Tricks
Jayesh Ingale

Requirement understanding is key for software testing. But a good tester not only tries to understand the system by going through the requirement but tries to see gaps between them.

The gaps can be identified by asking questions. Remember the concept of 5 W's. Keep asking Why? Where, Who, When, What, How and you will see many gaps in the requirements.

Consider a typical requirement: System shall allow customer to change their banking information. Possible questions can be: Who can change the information other than customer? When can the customer change information? What information can the customer change? How can the customer change the information (different alternatives, steps)

If you find a bug in a module, try more tests on the same module, as probability of finding more defects is very high. Remember Pareto Principle also known as 80-20 rule which states that roughly 80% of the effects come from 20% of the causes. Similarly applying this principle to testing, 80% of the defects are found/present in 20% of the project code base. This does not give you liberty to forget 80% of the code base; it simply means that the tester must focus more on the 20% code base.

Testers who analyse bugs recognize interesting patterns. Some of the patterns can be

- Bug rate increases in the beginning of testing, reach a limit and fall steadily.
- Bugs detected at boundaries or interfaces.
-Bugs detected for a specific business domain or technology

This is one of the most important aspects testers miss out. While finding out all the stakeholders is not so easy job, independent research carried out by tester can provide valuable information. Once you know the stakeholders of the project, think what would be their interest in using the application and how they would use the application.

There are some default settings for application. Most of the tests are designed and executed with these default settings. Change the settings and execute same tests. Since the number of test cases increases dramatically, tester needs to prioritise them and use judgement.

6 Change Configuration

Run the tests on lower memory, different servers, and network with different bandwidth or different video resolution. Even if you have thoroughly tested the software on appropriate test bed, there would still be issues when the application is deployed on production environment.

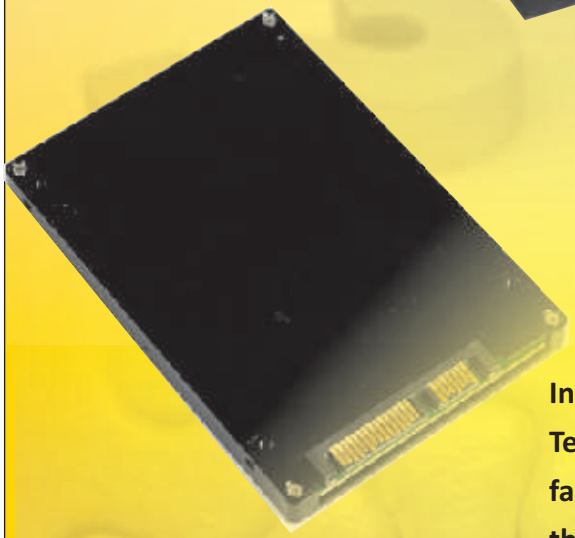
7 Exploratory Testing

Supplement your scripted tests with exploratory tests. Scripted test cases have their own boundaries; they focus on certain behaviour of the application. For example, if you are doing scripted testing, you will verify only the expected results stated in your test case. You wouldn't bother to observe other changes in the state of the application, like some fields getting blank, or change in colour of some controls, or some fields getting disabled and so on. Exploratory testing encourages continuous learning from the application behaviour and thus helping tester developing powerful tests and finding those hidden bugs.

Tips & tricks

Testing Technique

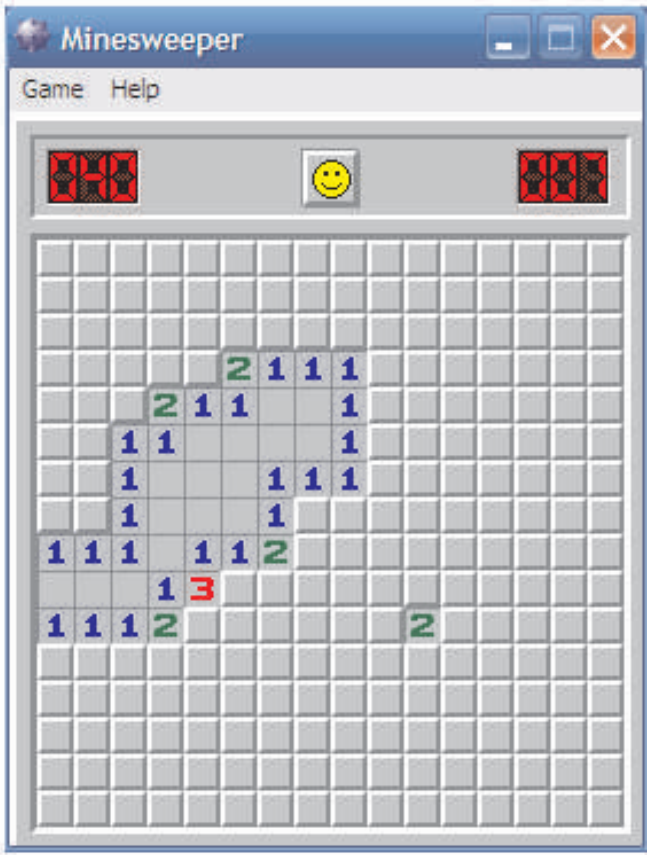
Akanksha Agrawal



In this article, we are going to explore the various Black Box Testing Techniques and their application in real life scenarios. We are all familiar with black box approach of testing. In this type of testing, the system is treated as "black box". The term black box means, consider the input and output specifications and not the internal structure of the application.

For eg, if you are asked to test minesweeper game running on Windows OS using the black box approach. How will you do??? Do something like this:

1. Click on any one of cell.
2. Now suppose the number behind it is 2, so this means that there should be two mines out of eight cells covering it.

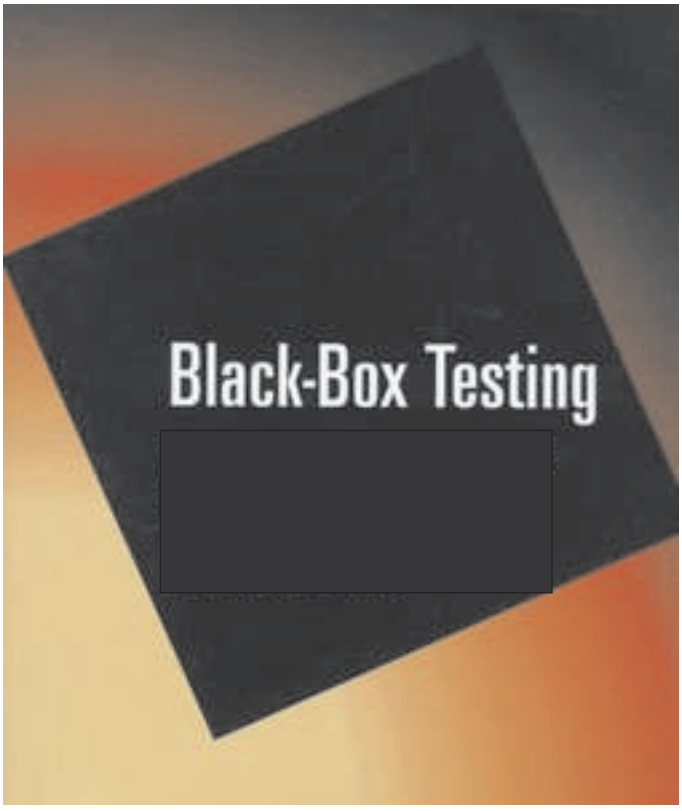


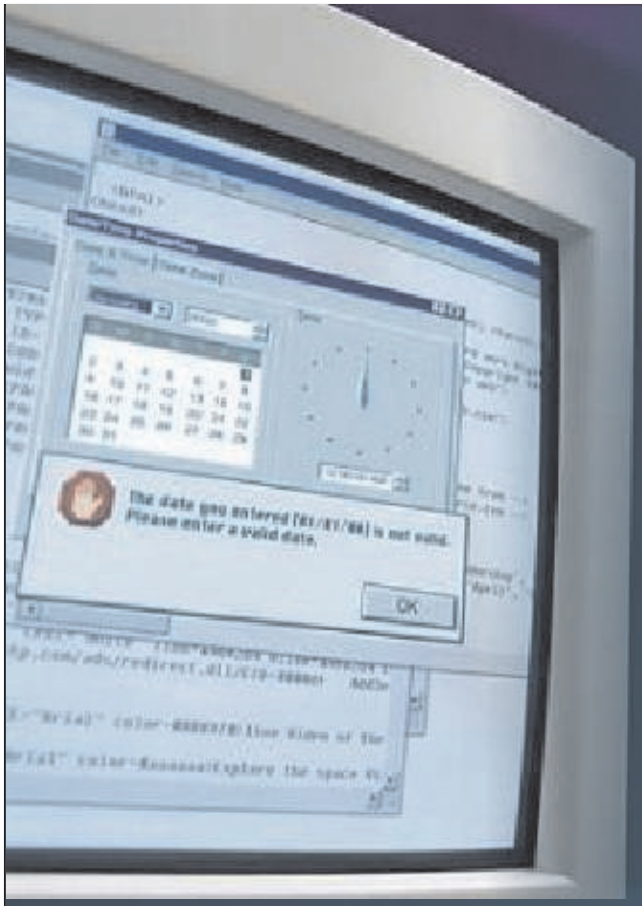
But now to test this, we have 'n' number of combinations to go with. Now the question arises: How can we perform Black box testing to cover such a huge set of data? How do we choose a suitable set of inputs from the set of all possible valid and invalid inputs so that effective testing can be done???

Keeping in mind that exhaustive testing is impossible, even if we try, it will be very expensive for a simple software unit. As an example, suppose you wish to test a single procedure that calculates the square root of a number and if you want to test it exhaustively, you have to try all positive input values. Isn't it daunting!!! Hey, wait!!! Don't panic!!! There are some more set to go. What about all negative numbers??? And what about fractions??? These are also possible inputs.

The number of test cases would rise rapidly to the point of infeasibility. In such cases, the smart tester has to effectively use the resources available by developing a set of test cases that gives the maximum number of defects in the given time and all effort spent should be effective. To help achieve this goal using the black box approach we have several methods. Very often combinations of the methods are used to detect different types of defects. These various methods are:

1. Equivalence Class Partitioning (ECP)
2. Boundary Value Analysis (BVA)
3. Cause Effect Graph
4. Error Guessing
5. State Transition Analysis, etc





Equivalence Partitioning Method

This is one of the most widely used Black Box Technique used to reduce the number of test cases to a manageable level along with effective coverage. An equivalence class consists of a set of data that is treated the same by the module as they produce the same result. Any data value within a class is equivalent, in terms of testing, to any other value. This means that if any one of the member, detects a defect from a specified class, then all other members of that class will also reveal the same defect.

How it is performed?

1. The input domain is divided into various classes on the basis of their input conditions that system should handle equivalently.
   a. If an input condition is a range, there will be one valid and two invalid class.
      For instance, Password text field should accept min 4 and max 8 characters. So, 4-8 will be a valid class, less than 4 will be one invalid class and greater than 8 will be another invalid class.
   b. If an input condition is a specific value set, there will be again one valid and two invalid class.
      For instance, the text field should accept mobile number only, ie 10 digit number, hence, 1000000000-9999999999 is valid class, less than 1000000000 is invalid class and greater than 9999999999 is another invalid class.
   c. If an input condition specifies a member of a set, one valid and one invalid class will be defined.
      For instance, the primary colors for a paint application should be Red, Green and Blue. So, these colors will come in valid class and all other colors will come in invalid class.
   d. If an input condition is a 'must do' condition then, one valid and invalid class is defined.
      For instance, first letter of Name field should always be in capital letter [A-Z].
2. A representative from each class is chosen and test cases are designed with the then specified data set.

   Representatives for above examples:

   Eg 1.a Test with any password 2 characters long, with 6 characters long and with 13 characters long.

   Eg 1.b Test with 765437484, with 9465348576 and with 23456789234567.

   Eg 1.c Test with Red and also with Black.

   Eg 1.d Test with Akanksha and also with akanksha.

   So ECP is a technique where we "divide" the set of tests and gain control over test coverage ie we "rule" it.

Advantages of ECP

1. It reduces the number of test cases without losing on coverage.
2. It helps a tester in deciding a data set which can reveal defects with high probability.

Limitation of ECP

1. The implementer of this technique has to ensure mutual exclusiveness of the equivalent classes
2. Determining result/output for respective input must be predicted.

There are number of other black box techniques like Boundary Value Analysis, which helps us to do software testing with manageable number of test cases and also overcoming the limitations of ECP. I will discuss this topic in my next post.

Till then, keep applying Equivalent Class Partitioning (Divide and Rule concept) in your domain of testing and have a nice testing time!!!

Demystifying Quality control & Quality assurance

Muskan Jaisinghania

Everybody wants to use quality product. Quality is a word which we frequently use in our day to day life, for example whenever we go for shopping, we ask the shopkeeper to show good quality product.

"The quality of something can be determined by comparing a set of inherent characteristics with a set of requirements. If those inherent characteristics meet all requirements, high or excellent quality is achieved. If those characteristics do not meet all requirements, a low or poor level of quality is achieved". (Ref. ISO Definition) "Quality is value for money; we expect a return on investment."

Quality is determined by the product users, clients or customers, not by society in general. Quality is measured by looking at the attributes of the product and is fitness for use. Quality can have subjective aspects and not just quantitative aspects.

In context of software "High quality software meets the needs of users while being reliable, well supported, maintainable, portable, and easily integrated with other tools". According to Tom Maccabe "Low complexity and high degree of user satisfaction is often associated with good quality".

To ensure good quality of any product (IT/ Non IT), often some standards and certifications are adopted by organizations. The prominent one are ANSI, AIAA, IEEE, EIA, ISO.

Quality can't be achieved without any effort but it is based on various good manufacturing and improvement processes; such set of processes is collectively known as Quality Management System.

Quality Control

Control: An evaluation to indicate needed corrective responses; the act of guiding a process in which variability is attributable to a constant system of chance causes.

"Quality control is the observation techniques and activities used to fulfill requirements for quality. It provides routine and consistent checks to ensure data integrity, correctness, and completeness." Also Quality control is a set of activities intended to ensure that quality requirements are actually being met.

To deliver quality correctness and completeness of developed product is tested; it includes Testing of work product. Hence Quality Control is often known for its Corrective actions. Corrective actions are steps that are taken to remove the causes of an existing nonconformity or undesirable situation. The corrective action process is designed to prevent the recurrence of nonconformities or undesirable situations. It tries to make sure that existing nonconformities and situations. Corrective actions address actual problems. Because of this, the corrective action process can be thought of as a problem solving process.

One of my friend who was working on testing of Human Resource Management based application shared with me an incident; while testing, he found that application was accepting future dates as a birth date of an employee which he reported as a defect. He asked me whether this activity comes under quality control or quality assurance? What do you think?

Yes, your guess is correct.

Since it deals with finding and fixing of defects it is a corrective action hence is control.

Now, Let's interpret Quality Assurance

Quality Assurance

The act of giving confidence, the state of being certain or the act of making certain.

"Quality assurance is a set of activities intended to establish confidence that quality requirements will be met. QA is one part of quality management. Quality assurance is the planned and systematic activities implemented in a quality system so that quality requirements for a product or service will be fulfilled. Two principles included in Quality Assurance are: "Fit for purpose" - the product mistakes should be eliminated. "

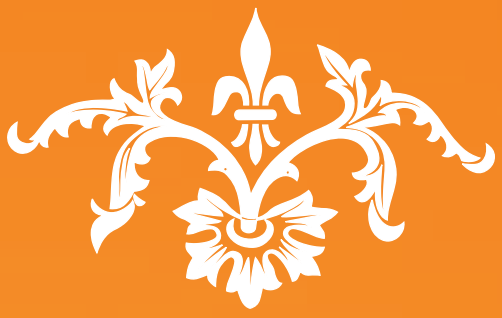
It often includes Root cause analysis, Postmortem reviews and Audit. Quality Assurance practices advocate preventive actions. Preventive actions are steps that are taken to remove the causes of potential nonconformities or potential situations that are undesirable. The preventive action process is designed to prevent the occurrence of nonconformities or situations that do not yet exist. It tries to prevent occurrence by eliminating causes.

To illustrate Quality Assurance, let me take you to "Early days" of software development where approach of development was

- Requirements
- Design
- Coding
- Testing
- Deployment

As testing was introduced late in process of software development, defects were detected late in the software development life cycle. This not only made it costly to fix, but had major impact on delivery schedules. The problem was in process and not in the product being developed. Hence there was need of improving process of software development. Instead of testing the product towards the end of the life cycle, testing now became integrated part of all phases viz, requirement, design, coding. This was all possible because of "IMPROVEMENT IN PROCESS" of Development.

To deliver quality products, correction and prevention of defects both are equally important.



Testing Quotes

"We have as many testers as we have developers. And testers spend all their time testing, and developers spend half their time testing. We're more of a testing, a quality software organization than we're a software organization" - Bill Gates in InformationWeek

"A program which perfectly meets a lousy specification is a lousy program. " - Cem Kaner - Kaner, Falk, and Nguyen

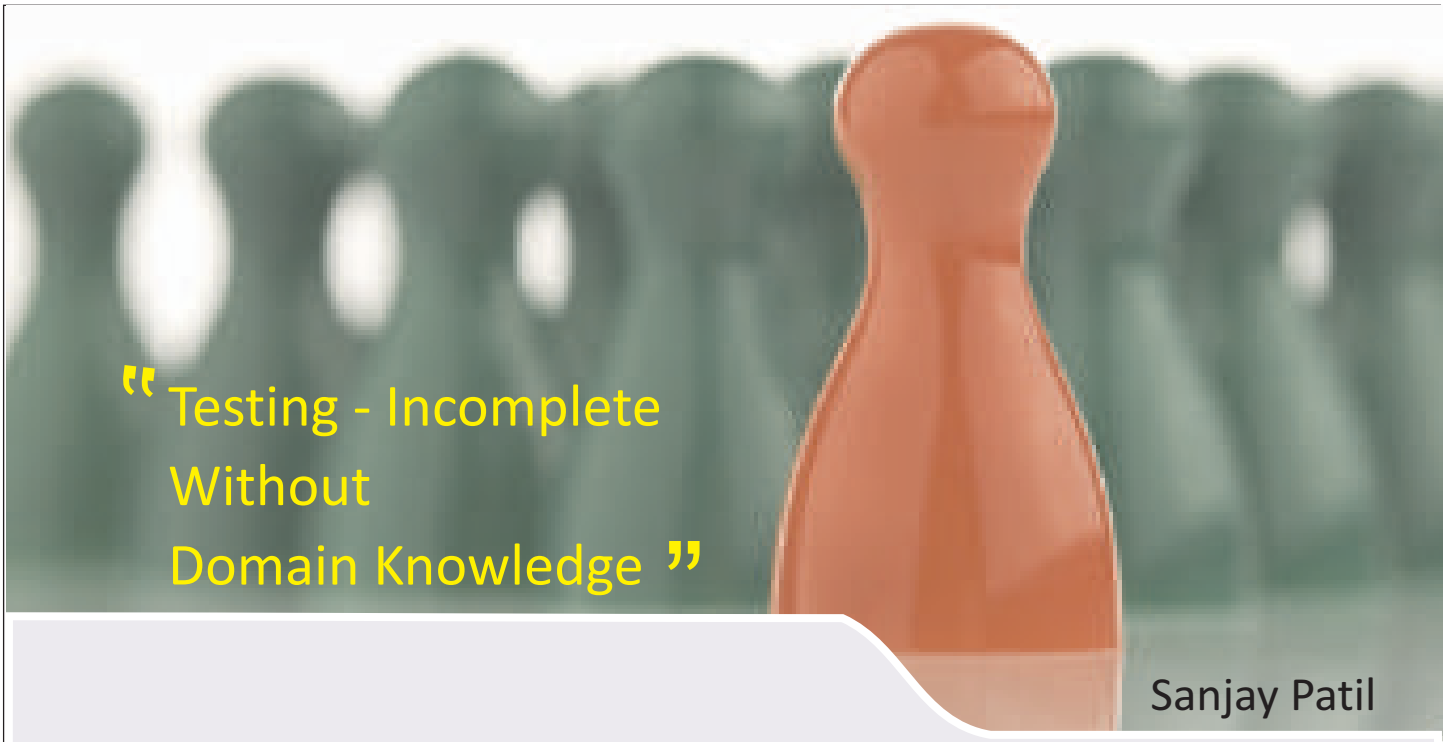
"To find the bugs that customers see - that are important to customers - you need to write tests that cross functional areas by mimicking typical user tasks. This type of testing is called scenario testing, task-based testing, or use-case testing." - Brian Marick

Before software can be reusable it first has to be usable." – Ralph Johnson.

"The test doesn't find the bug. A human finds the bug, and the test plays a role in helping the human find it." - Pradeep Soundarajan

Testers do not like to break things; they like to dispel the illusion that things work. -Lessons Learned in Software Testing

Some distinctions don't make difference - Principle of ECP
(James Bach)



"Testing - Incomplete Without Domain Knowledge"

Sanjay Patil

You have been asked to fly a Boeing airplane. What will you do? Have you ever thought of doing something and you don't have any knowledge about it? Could be anything, but have you dared to do so, I guess very few probably 1 out of 1000, might have dared to do so. But what about others; they would think twice before doing such thing. Similarly can anybody think of doing reasonable testing without Domain knowledge? It's is simply impossible. How can we test an application without knowing how the particular business work, pain areas of customers and possible solutions?

No doubt tester should have the basic testing skills which include technical and non –technical skills. Common sense can help you find most of the obvious bugs in the software. Then would you conclude that this much testing is sufficient? Would you release the product on the basis of this testing work? Certainly not. You will certainly have the product reviewed by the domain expert before the product is shipped.

What could be the different business domains?

1) Mobile application testing.
2) Wireless application testing
3) VoIP applications
4) Protocol testing
5) Banking applications
6) Network testing
7) ERP / SAP Applications
8) Healthcare Applications and many more...

Let's suppose you are testing BFSI applications (Banking, Financial Services and Insurance) You should know what are the user requirements in banking, insurance domain, working procedures, commercial environment, exposure to brokerage and so on. Armed with this knowledge the tester will have reasonable confidence to test the application. Here comes the need of subject-matter experts.

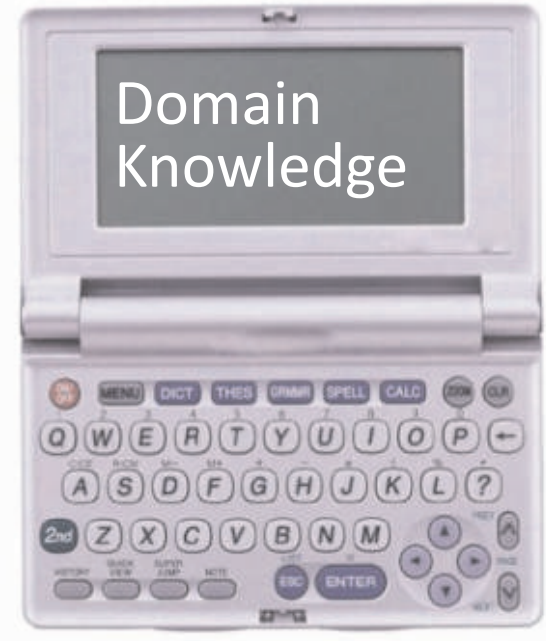
Let's take example my current project. I have been assigned a project related to Investment banking which is mainly in prime brokerage. I don't have any idea about what is that and many more terminologies related to that e.g. hedging, trade life cycle, and settlement systems. So do you think that I am in position to test this application thoroughly, No not at all. I must have knowledge of Trade life cycle, different classes of Investment assets and how do they function in the market. I am good in Testing and its technical aspects but still I can't test this application completely so what is the solution?? I must have to have acquire knowledge related to Trading, how customer trades, how the trade gets processed, how the Settlement and Clearing take place, how to do cross border trading, what is the role of Broker , what all securities I can I trade and many more questions. If you have answer to all these questions, only then can you test the application completely.

When we know the functional domain better, we can write and execute quality test cases and can effectively simulate the end user actions. I am referring to the depth of knowledge. The depth of knowledge helps the tester to design test cases with a comprehensive set of data. You can use equivalence class partitioning to categorize data in equivalent classes and optimize the number of test cases.

It is not only the depth of the knowledge, but the breadth of knowledge is equally important. This is where the real difference lies. Most of team members are focused on modules or couple of features. Testers need to see and understand the application as whole. This helps the tester to test impact or dependencies between one business rule/process with other business rules/processes.

What if you don't have enough domain knowledge of that project? You need to quickly grasp as many concepts as you can. Try to understand the product as if you are the customer and what customer will do with application. Visit the customer site if possible, find out how they work with the product, read online resources about the domain of the application, participate in events addressing such domain, meet the domain experts.

So get into the breadth and depth of domain knowledge and you will see positive results!!!



Visit us :
www.beyondtesting.co.in

1245+ Students celebrating their success.

Next could be you

Be a part of Software Testing Program

In today's IT industry, there is a serious lack of trained professionals with the desired skill sets to take on the ever increasing demands of an industry which has been marked by sustained double digit revenue growth.

Our program is aimed at providing focused hands on training in Software testing by the practicing Testing Professionals from the industry.

Highlights:

- World Class Curriculum designed by SEED
- State-of-the-Art Training Infrastructure
- Certified & Experienced Trainers
- 25,000+ Interview Calls in Testing with our Clients
- Campus placement interviews every week
- Certificate Awarded Jointly by SEED
- **100% Placement Asst. with more than 200 leading IT companies**

Our Programs:

- Functional Testing - Performance Testing - Security Testing - Usability Testing - Test Automation Tool

1245+ Placed in last 7 months

GENUINE • 100% • JOB ASSISTANCE

For registrations & more details contact :

SEED Infotech Ltd.: 'Panchasheel', 42/16, Erandawana, SEED Infotech Lane, Off Karve Road, Pune - 411004.

Tel.: 020-25467484, 25467484 | **Cell :** 98508 85179, 99229 33317

Email: sqa@seedinfotech.com | **Web.:** www.seedinfotech.com

seed™

beyond the obvious

For Campus Placements & Recruitment Solutions, Call : + 91 9822671630

Beyond TESTING

find us on facebook & twitter

http://facebook.com/beyondtestingmag

http://twitter.com/beyondtesting

The Right Time

Now is the Right time to
Get the rewards and respect that you deserve

Become a Software Tester

- 100% Job Assistance with MNC's
- Daily Campus Interviews
- 25000+ Interview Calls in S/w Testing till date

1245+ Placed in last 7 months

Karve Road : 25467484, 99229 33317
Aundh : 25898486 | Camp : 26138983 | Chinchwad : 27468308
Hadapsar : 32533550 | Satara Road : 9225645641 | Sinhgad Road : 24353384

IBM SEED ACADEMY

seed™
beyond the obvious

ibm@seedinfotech.com | www.seedinfotech.com