SOFTWARE TESTING CAREER PACKAGE

A SOFTWARE TESTER'S JOURNEY FROM GETTING A JOB TO BECOMING A TEST LEADER!

VIJAY SHINDE

Software Testing Career Package

A Software Tester's Journey from Getting a Job to Becoming a Test Leader!

A Complete Manual Testing Course with Practical Tips on Job Hunting, Career Planning and Skill Improvement!

by VIJAY SHINDE & CONTRIBUTING AUTHORS

Copyright & Disclaimer

Contributing Authors

Many thanks to Debasis Pradhan who helped me edit and review this eBook. Debasis is working as a test manager at eG Innovations. He is a software testing consultant by profession and a tester by passion.

Contributing authors:

Gunasekaran Veerapillai (Guna) – The author has worked as a Test Project Manager in Think soft, HCL Technologies and Covansys (CSC). Currently he is working as Competency Head in Wipro Technologies Testing Services. With 30+ years of experience in Banking Industry and IT, Guna is specialized in Test Portfolio assessments, Test process assessments and Automation assessments for many BFSI clients.

J.B. Rajkumar – The author has more than 15 years of experience in both Academics and Software Testing. He has worked as Corporate Trainer, Test Lead, QA Manager and QC Manager. He is a frequent speaker for International Conferences, Colleges, Universities and Software Industries. Presently he is with Automation Practice, in one of the top MNCs.

Suhas R M – The author is working as a software test engineer and having 4 years of manual testing experience.

Tejaswini Patil – She is working with an E-learning organization as a QA Manager.

Adarsh Thampy - Adarsh is a career and marketing expert at Jombay.

Mohit Khatri – The author is specialized in testing Banking Applications, Automation Testing Frameworks and Security Testing.

I hope you enjoy the eBook as much as we enjoyed writing it.

Table of Content

Introduction

There are a large number of testing books out there, some really good while the others pretty mediocre. But one thing is very common among these books -- most of these books are quite narrow in the topics they cover and the level of detail they present. This eBook **presents software testing as a practical engineering activity**, essential to producing high-quality software.

It doesn't matter if you are an undergraduate or graduate student or a fresher looking for a job in software testing or a professional working as a test engineer or a senior QA lead or a test manager, this eBook is designed to be used as the primary textbook and an all-in-one resource for software test engineers and developers.

This **eBook introduces a novel perspective on software testing** by covering the generic models as well as the coverage criteria essential for all software life-cycle processes.

This eBook strives to strike a perfect balance between **theoretical concepts**, which are covered rigorously as well as **practical contexts** thus allowing the readers to build a solid foundation in key methodologies, techniques, tips and tricks in the field of software testing.

This eBook takes an innovative approach to explaining the process of software testing: it describes various aspects of software testing and the process of applying some of the best and well-defined test criteria for software development. The structure of this eBook incorporates the latest innovations in testing, including techniques to test modern trends of software such as SOA, web applications, banking systems, OO, and embedded software.

This eBook is meant to act like a comprehensive, **step-by-step guide** to the most effective tools, techniques, and methods for software testing. Using numerous case studies of successful industry implementations, this eBook presents everything you need to know to successfully carry out software testing in a small to large infrastructure.

This is a great eBook for learning the **art and the science of software testing**. The clear terminology definitions and comprehensive real-life examples provide an easy way to master various software testing techniques.

After reading this eBook you should be able to get started in software testing, learn great tips on how to be an effective tester who finds critical bugs in the application under test, learn how to deal with the developers during uncomfortable project meetings, master the art of how to become a good test team leader/manager and more.

Chapter 1

Software Testing as a Career

What are the Various Career Options for Software Test Professionals?

Testing has become the showstopper for several application/product implementations and business has realized the importance and need of structured testing of applications before a product release.

Testing has created several levels and types. Specialization in these various types of testing has increased the intrinsic value of a software tester. From being a monotonous test case executor, several career options have evolved in front of the testing community. The following paragraphs depict the various options available to a software tester.

Career options for software test professionals:

Of late, testing is looked upon as a good professional career for many of the aspiring youths. As mentioned above, from being a (junior) test engineer one can move up to senior test engineer, test lead to test manager; else can become QA lead, QA Manager, depending on the organizational norms. The options available in the automation testing front are enormous. There are numbers of functional, performance, and security testing tools besides test management tools like Quality Center from HP, CQTM from IBM, Rally, and Asana to name a few.

The demand for niche skills like SOA testers, Security testers are on the increase. There is a dearth of skills in test automation arenas – scripting skills in the tools languages like VB, Java and other scripting languages like Perl, Shell, Python etc., Technical resources with capabilities to evaluate automation tools, create automation framework and reusable components are on demand. Today there is huge demand for good performance testers who can analyze the performance test results, identify the bottlenecks and suggest tuning techniques.

Specialization has come to stay in testing career - The following are some of the key areas where one need to specialize to move ahead in career path in testing apart from good knowledge in software life cycle testing process.

1) Domain Knowledge – Good knowledge in domain area of the application adds value to the testing professionals. There are popular domains like BFSI, telecom, healthcare, manufacturing, embedded etc. Numbers of certifications are available for each of these areas where the tester can get trained and certified.

2) Automation Testing Tools Knowledge – There is great demand for automation and performance testers. A good knowledge/experience in scripting languages of these tools is the basic necessity for succeeding in test automation. Knowledge on creation, validation and enhancement of test automation framework is very much required.

3) Certifications – QAI, ASQ, ISQTB, BrainBench and several other institutes are offering testing specific certifications. These certifications improve the confidence of the clients on the testing professionals. CQTM, PMP are some managerial certifications, which help the testers to scale up in the professional ladder. Certifications on the testing tools offered by vendors like HP and IBM increases the technical competency of the individual.

4) Niche areas in Testing – Experts predict that the niche areas like SOA testing, Security testing and mobile testing are gaining momentum in the testing space. Many tools are emerging in these areas. As testing professionals we should be aware of where the industry is heading and update our knowledge in those areas to be able to stay ahead in the game.

Knowledge enhancement is a continuous process. You should keep on reading all available online and offline testing resources. Spend at least two hours in a week reading software testing articles and news to update yourself to the current happenings and events.

As the saying goes "*you need to run continuously to keep yourself in the same place*", as testing professionals we should always work towards sharpening our testing skills to succeed in this highly competitive environment.

*****

The Definitive Guide to Choosing Software Testing as Your Career

Software testing and quality control are the processes by means of which application quality is controlled in order to yield an improved product. Software testing is done in every phase of product life cycle i.e. from requirement specifications, design, coding, to the user acceptance. Many complex software applications require in-depth analytical and technical skill to test.

Should you select software testing as your career?

Before you start thinking about making a career in software testing you must realize that no one can guide you choosing your career more than you! Start with the self-assessment which will show you more career paths and confidence that you have selected a right career. A self-assessment can reveal your skills, interests, strengths, and weaknesses which are helpful for you to make an informed career choice.

Questions you should ask yourself:

What is your ultimate goal in life?
What could increase your satisfaction at work?
What are your interests?
What job related skills have you developed until now?
What values are important to you in your work?

By answering these critical questions you will not only help yourself identify the passion in your life but also be in a much better position to pick a career that will suit best for your personality. If software testing is the answer to any of these questions then you made your career choice. If you are passionate about finding defects in other's work you can easily survive and progress in your career.

Finding your passion and making a career around it will make you more successful than any other career.

Some people think about this career because according to them getting a QA job is easy compared to other technical jobs. And this is the biggest reason most people fail to get a job. Without the necessary skills getting a job in software testing field is very difficult.

What if you don't have passion in this field?

I'll suggest don't try to get into this field unless you have the burning desire to become one of the finest software test engineers. There are many ways to develop interest about anything. First develop passion for software testing career and then start looking for a job. I'm telling this for two reasons. First, this will help you prepare and face for your interview confidently. Second, you will continuously progress in your career of your interest. No one will force you to learn anything against your passion. You will try to learn everything due to your willingness to learn.

Are you passionate about software testing but still not getting any job?

This happens due to lack of required skills. Let's discuss these required skills in details.

Skills required for a testing job

- Clear understanding of software testing concepts (like requirement analysis, test plan, test cases, test cases execution, bug reporting)
- Understanding of the basics of automation testing and ability to get started quickly on any given automation tool (No one will expect you to be the expert in all automation tools. But at least you should be able to get some automation scripts running on any given tool after having some hands-on training)
- Understanding of Database concepts – You should be able to understand and write basic and complex database queries
- Good communication skill (You will be rejected in interviews if you are not able to express your thoughts and ideas clearly. Interacting with developers, writing effective test cases and writing good bug reports are some essential skills where good communication skills will come in very handy)
- Good analytical and problem solving skill – it is required to understand complex software systems, write test cases that find bugs and test the application
- Time management skill – You must be able to work under pressure and meet a tight project deadline.

Added advantage – Not mandatory skills but won't hurt if you are a tester

- Hold any software testing certifications
- Programming skill

- Knowledge of MS Office suite of tools

Don’t worry if you don’t have some of the skills mentioned above. You can always learn these skills if you are really interested in becoming a good tester.

Now that you know what skills are important in order to secure and prosper in a software testing job let’s explore all career options for software testing professionals.

*****

Software Developer vs. Software Tester – Salary and Career for the Future

Salary Comparison:

Many candidates think that software testers get less salary as compared to development teams. In general, this is not true. This depends on many factors like the company you are working for, the type of work and your overall experience. But most of the companies treat both skills as same and do not differentiate the salary based on the work type. Salary of a developer and tester with same experience is similar in almost all companies. In fact, there are also many companies (like NASA, Google and even Microsoft), which pay more salary to testers than their respective developers. Testers with domain expertise are in more demand than developers of same experience and this might be the reason for this salary difference in these companies.

However, sometimes the salary differs based on skills of the person. Some QA persons get significantly higher salary due their skill, which is in demand for that company. Same things hold well with developers too.

Whatever the situation is, I think you should not compare the salary for selecting your career. If you have the right skills and dedication to work you will always shine in any career.

Career for the Future:

Both development and testing are different careers with different skill sets. In future both these careers will have ample opportunities. Hence you should choose your career based on your interest.

If anyone thinks testing is not much important than development then it’s your responsibility to prove them wrong. Testing is as important as development; it’s as simple as that! Rather let me go ahead and say, it's even more important than development. There are profound examples of software failures due to defects resulting in huge revenue loss as well as human fatalities and no company would want to repeat it by ignoring software testing!

*****

Getting a Job in Software Testing

In last chapter, we have seen the important skills required to get into the field of software testing. You can analyze your abilities and know which are the most important skills required for software testing and see if you have most of them.

I will continue mentioning that "**know your interest before going into any career field**". Just going to software testing career or any other career in demand without giving much thought about the same is wrong and may result in loss of your job interest as well as your job.

Now that you know your abilities, skills, interests you can decide to go for software testing career only if it is your favorite career which suits you and your personality.

Here are few guidelines for how to get a good job in software testing field:

Freshers who have just passed out from college need to understand and comprehend software testing methodologies first. Have hands-on experience on some automation and bug tracking tools like QTP and QC. It is often a good idea to join any software testing institute which will provide you a good start and directions for preparation. Continue your interview preparation while completing this course. This will help you to start appearing for interviews right after completing your course.

If you have some sort of previous IT experience then mention it in your resume while applying for software testing jobs. Having some previous IT experience is always a plus when you are looking for a job in software testing without any testing experience.

Try to find a job with your skills and abilities. Don't ever try to fake experience. This can ruin your career forever. Because if you fake experience, sooner or later you will get caught and lose your job and your career, not to mention the embarrassment and emotional pain that will bring along with it.

Software testing is *NOT 'anyone can do career!'* First remove this misconception from your mind. Testing requires strong analytical, problem solving skills, good knowledge of SDLC methodologies and out of box thinking.

Getting a Software Testing Job as a Fresher

This chapter will cover what you should and should not do to get your first job as a testing professional.

How to Get a Software Testing Job When You Have No Experience?

1) Be Passionate About Testing

Why do you want to do a testing job?

This is probably the most common question across all testing interviews. You should have a clear answer as to why you plan to pursue a career in this field.

If you are a computer science engineer, why did you not choose software development? If you are from some other streams, why not look for a job in your field of study?

People who are passionate about this field and would love to grow as a quality tester often end up getting the best jobs out there. Anyone can be trained. But only the people with real interest in this field can make a name for themselves.

Remove all other career paths that you're currently working on. E.g. you might be searching for oracle, C, C++, Java, Software Testing, etc. If you want to get into software testing, just keep preparing and applying for software testing jobs only. If not today, then tomorrow you will get your dream job. But till that time keep mastering your skills.

2) Have the Right Skills Required For the Job

As a software tester, you'd be spending most of the time trying to "break" the software. You should have excellent aptitude skills coupled with knowledge of testing methodologies and tools.

As a fresher, most of the time, you would not have exposure to any test cases. In such cases, you could join a training institute that offers hands on testing training. You could also take up some freelance work to improve your experience.

Nowadays most job openings require you to have some certifications. This is made mandatory by most companies so that the candidate can be productive from day 1 and no amount of time or money need to be spent on training the candidates on the basics.

If you are serious about testing career, you could go for certifications like ISTQB which will enhance your value in the job market.

3) Choose Your Niche

Have you heard the saying "Jack of all trades but master of none"? Well, these days, especially in IT field, you need to specialize and not generalize. A specialist is always preferred over a generalist by most companies.

Some of the popular testing niches you can choose are:

Manual testing: In this method, a tester takes on the role of the end user- one who will be using the software product. It's a tedious process by which the tester has to use all of the software features to find bugs. In most cases, testers perform these tests based on an already laid out plan.

Automation testing: Using this mode of testing, a tester can evaluate software by writing scripts that automate the tasks. No need of manually doing everything which saves a lot of time. This is often the most cost-effective way of testing software over the long term

Performance testing: In this method of testing, software is tested against a specific workload. Some of the metrics that would be evaluated are responsiveness, maximum load capacity and so on.

Similar to software development, software testing is a vast field. So you should focus on one area of testing. This will make you a hot property in the job market. Even though you should specialize in one area, you should also try to be familiar with other areas. You must also be flexible enough to shift to another track if required.

But don't enter into automation or performance testing career without knowing the ins and outs of manual testing. Remember slow and steady wins the race!

4) Bust All Myths

Sadly, the industry and students have a lot of misconceptions about software testing jobs.

Some of the myths still prevailing are:

Myth #1: *Software testing is a simple job. So anyone can do it:*

Although it's not rocket science, testing still requires a lot of work and intellect from the part of the tester. So don't think this is a walk in the park type of job and jump into it.

Myth #2: *Testing is a second tier job compared to development:*

We had one reader apply for a job who supposedly thought that she was entitled to a testing job since she had already worked as a software developer.

Her argument was that since testing is inferior to development, and she cracked development interview and worked on a project or two, she was entitled to a job as a tester without an interview.

Well, we have news for you. It isn't the case! Software testing is just as important and valued as software development. So experience in one domain doesn't necessarily entitle you to a free pass to another.

Myth #3: *Automation testing means clicking a few buttons and the software will do it for you.*

While it can be true for some existing test cases, most of the time you'd have to create the script for automation testing. So don't think that it will be easy and since you don't know/hate programming, it's the perfect opportunity for you.

5) Write a Perfect Cover Letter and Resume

How many times have you applied to a set of jobs at a stretch using the same resume and not even thinking about including a cover letter?

If you are like most people applying for a job, your reply should be "most of the time" or "always".

There are many candidates who simply attach their fresher resume and blindly apply to just about any job they can without even reading the titles. Do you really think a recruiter will take the pain to evaluate you after such an attempt?

Always customize your resume and make sure that you include a custom cover letter with each application. Instead of trying to apply to 50 different employers at a stretch, apply to just a few relevant ones.

You'll drastically improve your chances of hearing back from the employer.

6) How can a fresher looking for a software testing job get relevant experience?

I've discussed various options in later chapters but here are a couple of options to get started

1) Get some experience by working on dummy projects available on the internet. Search for online dummy projects (e.g. Inventory management software) and download test software and all available documents. Test this software with complete STLC.

- requirement analysis,
- writing test cases,
- executing test cases,
- logging defects and,
- preparing test reports

If possible, get your work (defects, test reports etc.) evaluated by experienced software testing professionals.

2) By adding dummy projects learned from software testing courses: If you have joined any software testing course to learn manual or automation testing then add these dummy projects in your resume. This way you will get some experience to put in your resume and project section in your resume won't be entirely blank. This way you will have an added advantage compared to other freshers.

*****

How to Switch to Software Testing from Other Fields?

For some reasons if you are no longer interested in your current career and want to switch to software testing please read this chapter. Often non-IT candidates find it difficult to switch to software testing career. But don't worry; this guide will help you get better software testing jobs. Many top companies like IBM, Accenture, and Infosys, etc. hire B.Sc. and other non-IT stream candidates also.

First of all, I would like you to take advantage of your current profession. Let's see how this is possible. Assume that you are working for the last 3 years in customer or tech support field in banking domain but want to switch to software testing. If you learn and get control over the some of the testing skills that I mentioned in earlier chapters you will be able to crack the interview easily. You can leverage your expertise from customer support and banking domain along with newly learned testing skills to get a job. After all, being in customer support is always a great way to know how customers think when they use a product. Below are the exact steps you need to follow in such cases:

1) Take a professional course in software testing to learn manual and automation testing. Pay close attention to learn practical things. At the end of the course you must be clear with manual testing concepts. If you learn any automation tools in this period then it would be an added advantage

2) Prepare a good resume mentioning experience of your current profession (IT or Non-IT)

3) Add professional software testing course details in your resume

4) Go and apply for some freelancing software testing opportunities

5) Find online beta testing opportunities. Test beta software for free (or paid whichever is possible)

6) Mention this freelance testing project details and beta testing experience in your resume.

7) Before appearing for interviews make sure to have a strong reason for the question: ***Why do you want to switch to software testing from your current […] profession?***

8) Though you won't be considered as a fresher, there is a chance that in some companies your other profession experience might be treated separately since it might constitute different skills.

9) Find out IT companies having services or projects related to your current profession (E.g. IT companies working on financial projects, in case of our above example). Search all software testing/QA job openings for these companies on their website or job portals. Apply to these openings with a professionally drafted cover letter.

10) Do not use same resume while applying to all openings. Make sure to update the resume to highlight your skills depending on the job requirement.

For IT professionals, it's comparatively easy to get into software testing field than non-IT professionals. Non-IT professionals need to bridge their skill gap by taking part time computer and IT training (E.g. distance learning courses like M.Sc. IT, MCA) along with any professional course related to software testing.

Relevant experience is the biggest hurdle for these candidates. You have following options to get software testing work experience:

- In your professional testing course ask to provide hands-on training on any live project. Work on this project and follow the complete STLC (Software Testing Life Cycle). You can add this experience in your resume.
- After completion of professional course in software testing apply for freelance testing opportunities on sites like elance.com and odesk.com. You can also consider other testing services like utest.com for testing experience on real projects.
- Find beta testing opportunities. Test beta projects and add this as an experience in your resume.

I'll explain all these freelancing opportunities in details in last article.

Make sure to get practical testing experience on these projects. In interviews you will be asked questions on these projects.

*****

Writing a Good Resume for Software Testing Job Application

With a well written resume you should get more interview calls than any other candidate having more experience than you. Little extra efforts to create a resume will make you stand out from the crowd.

After reading this chapter you will be able to write a killer flawless software testing and quality assurance resume that will definitely help you to get more interviews invites.

Your resume is the very first step in any job application process. It's an opportunity to advertise yourself and demonstrate that you are the most suitable person for the available position. Getting an interview call depends on how you present your resume.

How Much Time Do You Get to Impress an Employer?

Finding a job is very hard in this highly competitive job market. Recruiters are getting hundreds, if not thousands, of quality assurance tester's resumes for a single position. You must stand out from the crowd by starting with an effective resume. Recruiters don't have the time to thoroughly read all the resumes. **Your resume will be scanned quickly in a minute.** Yes you read that right, you get hardly one minute to persuade employer to take the interview call decision.

Does that make sense? To make a good first impression on potential employer you must present your skills effectively on first page of your resume, rather the first half page of your resume is very important to make or break it.

Unfortunately, many job seekers do not take this aspect of their job application seriously. They just copy and paste other's resume without even bothering to change the hobbies. Remember, no matter how talented you are, if you don't present your skills properly in your resume, no one is going to see your talent.

How to Make a Great First Impression from Your Resume?

Don't write whole story about yourself without thinking what employers want. Your resume should be focused on the employer's needs. First read the job requirements carefully. Judge yourself based on these requirements. Prepare a list of skills matching the job requirements. Highlight these skills on first page of your resume as well as in job application cover letter.

FAQ's About Software Testing Resume:

How to write software testing project details in your resume?

Job experience section in your resume is the best section to describe the details of projects you worked on. Describe project details with following headings:

- Project name:
- Client name (optional):
- Project description: (brief overview of the project in 2-3 sentences)
- Project environment: (mention software coding language, testing tools etc.)
- Team size:
- On job accomplishments: (mention all your key responsibilities)

Many candidates ask "What should I put in resumes if I've gap in my career?"

Obviously, companies would want to know why you had a career break. There could be many reasons for employment gap. The most common reasons are - enjoying holiday break, relocation, handling family business, skill upgrade, time off to have a baby etc. Don't hesitate to mention the valid reason for any gap in your employment. If the career gap is large then consider mentioning it in your cover letter with a brief reason. You can also handle career gap with positive notes if you have taken off for skill enhancements, certifications, higher education etc. Be honest and I'm sure you will easily convince the interviewer about your career gap. But make sure to keep yourself up-to-date with latest skills and industry standards.

In interview, be prepared to discuss the career break with interviewer. But don't go into much detail. Explain in brief and to the point.

Finally, having career gap for valid reasons is perfectly fine. Just you need to present yourself properly so that employer will not have any concerns about your current skills.

Start career hunt again through people you know. Since these people already know your strengths, they can help you learn how to keep you up-to-date with current technology trends. Also use social networks like LinkedIn, Twitter etc. for making your career search easy.

On-the-job-accomplishments on first page of your resume:

On the job accomplishments are the best ways to convince employers about your abilities and problem solving skills. You can describe these accomplishments in brief in your resume. Describe what the problem was and how you solved it. Prepare some solid examples to support your claims. Also be ready to answer all relevant questions asked by interviewer for your accomplishments.

Here is one example of job accomplishments: "*When I joined […] project in my company I saw the work was ad-hoc and there wasn't any standard software testing process. I took initiative building a standard software testing process that fits our project needs. By this streamlined process we managed our time effectively and started concentrating more on core testing activities*".

Mention relevant modules/subjects you studied

Candidates with computer networking and system administration skills are preferred for software testing positions. If you studied any subject or completed any course related to computer networking and system administration then add it in your resume. If you have Linux/Unix operating system knowledge you can also add that in 'relevant-skills' section of your resume.

Software testing certifications and training:

Software testing certification is an added advantage for all testing and QA positions. Testing certifications like ISTQB and CSTE are preferred in many companies. Always keep learning and equip yourself with necessary tools and skills so that you will never face problem to get a job or switch into a different company. If you have completed any software testing course or diploma after your graduation or post-graduation then put it under "skill enhancements" section of your resume.

How to learn software testing skills to add in your resume?

If you don't have necessary relevant skills to add in your resume then you must learn those skills first. For example, learn how to use defect tracking and test management tools. You can find many open source tools to learn. Download any open source tools and start practicing it at home.

E.g.:
1) You can learn TestLink test management tool online at TestLink online.
You can practice everything on this demo page. Once you get good hands-on experience on TestLink you can put it in your resume.

2) Learn Bugzilla defect management tool online here or download and install it on your local system. Create new account and play with it. Learn how to add and manage defects in Bugzilla. Once you get basic knowledge of this tool you can add this tool under "Defect management tools" section of your resume.

This way you can learn many automation tools online. Everything is available online. You just need to grab and learn it.

Software Testing Resume Essential Sections (Sample):

- Personal details (Name, email and contact) at the top
- Career objective – not more than three lines
- Educational qualification – in reverse chronological order (recent education first)
- Details of skill upgrade – like testing certifications, training, computer networking and system administration skills
- Work experience – details for each project from every employer
- Interests/hobbies and significant achievements

- Additional personal information like marital status, passport details etc. But no more than 3 personal details.

Conclusion:

Resume Formatting Tips

1) Keep resume brief but comprehensive in expression.
2) Make sure you have a clearly defined career objective placed at the top of the first page. Keep it short and include your professional goals in two or three lines.
3) Remember a single spelling error is sufficient to reject your resume. Spell check your resume twice before hitting the send button.
4) Highlight relevant skills.
5) Do not fake anything outside your experience.
6) Focus on employer's need. Think about recruiters expectations.
7) Avoid use of too many tables. Use it for mentioning your qualification only.
8) Restrict resume to 2-3 pages unless you are applying for a team lead or managerial positions.
9) Don't add irrelevant personal details like age, height, weight, father's details etc. unless requested.
10) No need to write 'Curriculum Vitae' or 'Resume' word at the top of your resume.
11) Do not use word "I" while describing project responsibilities. E.g.: Instead of "*I wrote test cases…*" use "*Wrote test cases…*"
12) Make sure you write your name, email address and phone number on top left corner of the resume.
13) Write relevant skills and on-job-accomplishments on first page of your resume.

I've detailed each and every aspect to write a killer software testing resume. Now you should not face any difficulties writing an effective software testing resume.

With only a minute to capture attention of employer you should showcase your skills effectively to make a first impression. Once you are successful in that be ready to explain everything you mention in your resume. You should be able to produce example stories to support your skills. Writing a good resume is the first step towards getting a dream job. Let's see how to prepare for interview in next chapter.

*****

Software Testing Interview Preparation

As a software tester, we keep performing testing activities in various phases of a project. When it comes to testing our own skills, we may not end up choosing an appropriate approach. I am talking about different interview rounds and how to face them. Let's discuss about the challenges that a tester has to face in an interview.

You should be very clear about manual testing methodologies. The next and most important thing is to know which type/technique/concept of testing should be applied at what stage of STLC. Knowing what to test, when to test and what not to test is very important.

Preparation before the job interview:

Before attending the interview, check the job profile in detail. Understand if the requirement is purely in manual testing, automation testing or on both. Check if your job profile experience match with what is expected.

The interviewer will mostly stick to questions around the given job profile and what you have mentioned in your resume. Make sure you answer the questions confidently which are based on the skills mentioned in your resume. It depends on how the discussion goes between the candidate and the interviewer, which leads to discussion in other areas.

Appear confidently at the time of interview:

In most cases, the interview starts with a question – tell me about yourself. One can answer to this question by following a sequence like starting with your name, qualifications, and interests and how you started your career as a software tester. Some interviewers do not like to hear about personal family details. So do not proceed with these details unless the interviewer asks for it.

Give clear and concise answers. Do not try to explain about ideal cases. Interviewers are interested in practical approach, rather than ideal cases. Explain how you usually go about solving a problem and your way of tackling things. Don't talk anything negative about any person especially about developers. If you do so, then it shows that you are not mature enough. Nowadays, most of the interview scenarios are cited rather than direct question and answer. If the scenario is new to you, take few seconds to think on it and then answer. Do not hurry up to answer the question without thinking.

The way you present yourself in the interviews is very important. Right attitude is very important too. Many managers can judge it easily, if you have really worked on projects or it's just a fake experience. The confidence level with which you answer makes a strong impression. For any question if you are not sure about the correct answer, just make an attempt. Don't just give up. You can also talk about things that you explored in free time or with your interest. This shows that you take initiative and are a continuous learner as well.

As many of you must have experienced that the interviewers keep asking about the processes that you have followed or are familiar with. One does not need to worry if they have never followed any processes. Following the processes is up to the company and a tester cannot do much regarding that. But of course one can follow some processes for his/her own task (I mean the modules that you own or are in charge of). This will not only help to manage things but also inspire others to follow some processes. Any process that has proven results can be followed. Instead of blaming others for not following any processes, you should take an initiative to do it. Do not forget that initiative is one of the qualities that a tester should possess.

One more important point: It's not necessary that the person who is taking your interview is from QA background. A person from developing background can also take software testing job interviews. In such case it becomes very important to answer the questions very carefully. It may sound illogical when a person from non-QA background interviews a tester. But remember it will be a very good experience as you will know how testing is perceived by others.

Think out of the box (more on this in later chapter). Don't just follow the traditional testing methods. Learn from your mistakes. Automate your daily routine tasks. Think from the user's perspective. Think how user will use your application. This way you will get insight of an application which will help you answer the questions clearly.

Besides being curious to learn you should upgrade your skill in following areas -
- Database concepts and SQL queries (basic and advanced)
- Learn scripting languages (AWK, Bash, Perl, Python, Ruby, Tcl, VBScript, JavaScript etc.) for automation testing
- Network and system administration concepts (troubleshooting skills)

Finally, you should make the interviewer feel that the application you are testing in current organization is a complex application and it has lot many challenges for testers.

Also, if you don't know answer for a question, say so. Don't fool around and get into trouble.

*****

Why You May Not Be Getting a Job in Software Testing?

If you have applied for many job openings and also attended many interview calls but still if you don't have the job you are looking for then there must be something wrong.

Be honest and think of some of the possible reasons mentioned below. And probably you might get the answer as to why you couldn't crack any software testing interview yet!

Reasons why you are not getting hired for a testing job -

1) You have a badly formatted resume. No one cares to read it and even you don't care to change it.

2) You are not passionate about software testing. You are just running behind the testing job as others do. You are looking for software testing job just for the sake of being employed somewhere.

3) You are not applying for the jobs relevant to your profile.

4) You don't have strong and valid reason for the question – Why do you want to join or switch to software testing?

5) You are not reading company details like their services, locations, current news etc. before appearing for the interview.

6) You are faking work experience.

7) You pretend to have all the testing skills.

8) You think "testing is no-brainer task" and anyone can do it.

9) You think no training is required to work in software testing field and you will learn everything on job.

10) You don't have sufficient examples to support your answers.

11) You are from non-IT background and now want to switch to software testing, but you don't have the skills required for software testing jobs.

Remember, you are the only person responsible for your career. No one will reject you the job in software testing without a reason. Find out what that reason is. After all, employer will always select the best fit for their positions. You just need to think – how can you best fit your skills to match the job!

Selecting a Good Institute for Learning Software Testing Skills

Selecting a good software testing institute is always a tough task. Many institutes fake about their staff skills, education and experience. Also they make false commitments of assisting for 100% placement. Beware of such institutes which give you 100% job guarantee after course completion. If any institute is claiming this then get all details about their claims and hidden terms if any. Also you shouldn’t expect 100% placement guarantee from any institute. Focus on course content and quality of teaching. Institutes can only assist you getting an interview calls. It’s up to you, how you convert those opportunities into job offers by presenting your skills at the time of interview.

Here are some criteria for selecting the best software testing institute.

You need to ask below questions:

1) What is the course syllabus?
2) What are the qualifications of the instructors?
3) Is this a course for both manual and automation testing?
4) Which automation tools are covered?
5) Do you have automation testing lab setup with necessary tools?
6) Are you going to provide testing tools for home practice?
7) Do you use live projects for teaching manual and automation testing tools?
8) What are the course prerequisites?
9) What are the course fees? Can we pay the fees in installments?
10) Can we attend some demo classes before joining or paying the course fees?
11) Will you provide course notes?

Based on the answers to these questions your decision to join the testing institute will be easier.

*****

Switching from Manual to Automation Testing

Clear understanding of manual testing concepts is a prerequisite for advancing to automation testing career. In fact most of the automation testers who are doing a great job in automation career are very good at manual testing concepts.

Though there are no hard and fast rules about when to switch from manual to automation, but below are some guidelines which will help you kick start your automation testing career.

How to start with Automation Testing career?

Many candidates complain about not having automation testing requirements on their current project. But you can create this opportunity by discussing with your manager about the benefits of automating regression tests. You can start with automating simple repetitive test cases. If you are working on the project for a long time and good at writing test cases you can easily identify repetitive test cases that need to be automated.

Do some research online about the best suitable automation tool for your project needs and start using it. If you have no budget for commercial tools you can start with open source automation tools.

E.g. start using JMeter automation tool for performance testing of web application. JMeter is an easy to learn open source performance testing tool. You can also use other open source record and playback tools like Selenium to automate web application regression tests.

The other opportunity I can think of is daily build deployment process. Often the QA team is required to deploy the daily builds manually before starting to test. But you can use simple scripts (even a batch script) to automate this task.

When you want to advance automation career you can put this experience in your resume. Also join some professional automation courses that provide hands-on training with tools like QTP, LoadRunner, Selenium etc. This hands-on dummy project experience can be mentioned in your resume. Do not fake automation testing experience at any stage of your career. This won't help you get any job as you cannot answer questions on practical automation work.

These are few small things to get started in automation testing. Once you get hold of the process and tools you can start automating complex tests. But don't expect to switch to automation career overnight. You need to practice more and more automating simple and complex tests before you can become an expert automation tester.

As I said earlier don't wait for automation work opportunity. Create it and start working right away. Don't be afraid of scripting languages. Many record and playback tools create complete script for your tests. You need some basic coding knowledge to understand that and tweak it for achieving your purpose. Also there are many tools (e.g. HP LoadRunner) that do not require much scripting knowledge.

Skills required for automation testing:

- Clear understanding of manual testing concepts
- Programming and scripting language skills
- Experience with database connection, concepts and queries
- Knowledge of any automation tool
- Knowledge of test automation frameworks (data driven, keyword driven, etc.)
- Knowledge of performance and regression testing
- Unit testing (mostly done by developer)
- Analytical and problem solving skills

Is manual testing underrated?

Certainly not! Some people have a tendency to call manual testing as "*anyone can do job*". These people are either not familiar with SDLC or they are egoistic. Both these testing types need specialized skill sets to accomplish task. Since manual testing is the foundation to start automation, it cannot be ignored for automation testing.

Which is better to get a Job?

Both the manual and automation testing types have bright career paths ahead. Both the skills will help you advance in your career. I would say only manual or only automation opportunities are less compared to mix of both. The chance of getting a job increases if you are an expert in manual testing with some automation experience.

You should be open to try and learn new things. Don't dismiss the opportunity because you don't understand it. Get some experience and then make up your mind. Initially some technologies may sound difficult to learn but after some hands-on experience you may feel comfortable working on it. But after trying hard if you find automation testing difficult to understand and implement, you can always choose to stick to manual testing and be a master at it.

Finally, select the career where you are really good at and enjoy doing it. When you become expert in that field start growing by learning other relevant skills.

****

Chapter 2

Getting Started in Software Testing - Testing Skills Improvement!

How is Actual Testing Process Carried Out in a Company Environment?

Those who get just out of college and start searching for jobs usually have this curiosity as to how the actual working environment in the companies would be.

Here in this chapter I'll focus on actual working process involving software testing in the companies. I will try to share some practical experiences here for those who have only theoretical knowledge in testing and curious to know how a production level real testing lab operates.

Whenever we get any new project there is initial project familiarity meeting. In this meeting we basically discuss things like who is the client, what is the project duration and when is the delivery, who are involved in the project i.e. manager, tech leads, QA leads, developers, testers etc.

And then from the SRS (software requirement specification), project plan is developed. The responsibility of testers is to create software test plan from this SRS and project plan. Developers start coding from the design document. The project work is divided into different modules and these project modules are distributed among the developers.

In the meantime, the tester's responsibility is to create test scenarios and write test cases according to assigned modules. We try to cover almost all functional test cases from SRS document. The test cases are maintained in test management tool.

When developers finish individual modules, those modules are assigned to testers for verification. Smoke testing is performed on these modules and if test fails, modules are re-assigned to respective developers for fixing the defect. For modules which pass smoke test, manual testing is carried out from the written test cases. All bugs are logged in bug tracking system and triage team assigns these bugs to respective developers for fix. Upon bug fixing, testers do bug verification and regression testing of all related modules. If a bug passes the verification it is marked as verified and is closed. Otherwise, above mentioned bug cycle gets repeated.

Different tests are performed on individual modules and integration testing is performed during module integration. These tests include compatibility testing i.e. testing application on different hardware, OS versions, software platforms, browsers etc. Load and stress testing is also carried out according to SRS.

Finally, system testing is performed by creating a virtual production like environment. On successful test completion, test report is prepared and decision is taken to release the product!

This was a brief outline of project life cycle process.

Testing as it is carried out in each SQA life cycle as per IEEE and ISO Standards:

1. Review of the software requirement specifications.
2. Objective is set for the major releases.
3. Target date planned for the releases.
4. Detailed Project Plan is build. This includes the decision on Design Specifications.
5. Develop Test Plan based on Design Specifications
6. Test Plan: This includes objectives, methodology adopted while testing, features to be tested and not to be tested, risk criteria, testing schedule, cross-platform support and the resource allocation for testing.
7. Test Specifications: This document includes technical details (software requirements) required prior to the testing.

8. Types of Test Cases:
    - Smoke (BVT) test cases
    - Sanity test cases
    - Functional test cases
    - Regression test cases
    - Negative test cases
    - Extended test cases
9. Development: Modules are developed one by one based on their dependency
10. Installers Binding: Installers are built around the individual product.
11. Build procedure: A build includes Installers of the available products – multiple platforms.
12. Testing: Smoke Test, (BVT) Basic application test to take decision on further testing
13. Bug fixing started
14. Testing of new features
15. Cross-platform testing
16. Stress testing and memory leakage testing.
17. Bug Reporting: Bug report is created
18. Development: Code freezing. No more new features are added at this point.
19. Testing: Builds and regression testing.
20. Decision to release the product
21. Post-release Scenario for further objectives.

*****

What Testing/QA Process Should Be Introduced On a New Project?

Software testing should not be an uncontrolled process. Uncontrolled processes often result in high cost and time with low quality output or even situations where the output cannot be measured. To avoid this, testing should be done in a highly controlled environment and it should be a planned process in every single stage of STLC (Software Testing Life Cycle). At the end of the testing cycle, there should not be any uncertainties regarding any aspects that were tested.

Who is responsible for setting up a testing process?

Generally the test manager set up testing process on a project. But in some companies, which have very small test teams this can be set by the test lead or even by the team members after extensive discussion within team.

The purpose of controlled testing process is – improved quality, testing cost and testing time.

Here are some **QA best practices and methodologies** with set of rules to setup a testing process or a new testing department in an organization:

- Set process to create and maintain test plan and test strategy documents.
- Get testers involved in earlier stages i.e. right from requirement gathering phase. This will help to find the defects in early stage of development, thus bringing down the cost of fixing them in the product life cycle.
- Setup proper communication channel with project management and stakeholders. This will ensure you and stakeholders are on same page in terms of expectations and test output. This will also ensure that testing is in alignment with your organizational priorities.
- If your test starts with a new build then set-up processes and rules to get new build when required. E.g. the rule can be – get new build every day in the morning or weekly on a pre-decided day and time.
- Set-up a process for build deployment. Tasks like who should deploy the build, where the build should be deployed, how to deploy it, what should be done if build deployment fails etc. should be decided in advance.
- Set process for BVT (Build Verification Test). Identify some smoke tests to be executed on every new build. Automate these smoke tests as far as possible and run this BVT on every new build. Build should be rejected if smoke test fails. Test process should define whom to inform and how to inform (by email, by logging a defect etc.) upon BVT failure.
- Set process to write and manage test cases. Type of test cases to be covered, how to write test cases (any specific format to be followed), and test management tool to be used (e.g. HP Quality Center, TestLink, TestRail, Rally etc.)
- Process to use test management tool – how to manage test cases, test cases priorities, test cases keywords, requirements, test case assignments, how to update test cases with results, how to manage failed test cases, how and when to mark test cases as blocked etc.
- Define the defect management process – using defect management tools (like JIRA, Bugzilla etc.), adding new defects (defect summary, steps to reproduce, expected result, actual result, assumptions, notes etc.), assigning defects, setting defect severity and priority, process to update defects (status, assignee etc.) after verification.
- Set process for internal team reporting and format of reporting – who will report to whom, reporting frequency (daily, weekly etc.), what to report (issues, obstacles, daily status etc.)
- Process for test effort estimation – how to estimate, estimate considerations
- Using automation tools – which automation tools to use, how to use those tools, how to write and maintain test scripts, defining manual and automation tests
- Building and using test environment – build test environment that is an exact replica of the production environment. Test environment should be periodically updated from production

environment. Set process for updating and maintaining test environment including test infrastructure, test data, DB backups etc.
- Team skill assessment and improvement – QA team skill assessment needs to be performed periodically in order to identify strengths and weaknesses so training can be offered if needed. Conduct QA performance reviews and appraisals.
- Team induction – Process to train new members on board (who will train new members, what topics should be covered, duration, training evaluation etc.). Process to train new team member should be efficient enough to have less training time and mentor's involvement.

It's important to review the test process at regular intervals to identify and work on key improvement areas. You must also define the maturity levels (initial, controlled, efficient, and optimized) of all the key areas so that at any given point you can identify which key areas need further improvement and processes which are in optimization phase.

Testing process is a continuous process to improve tester's efficiency, effectiveness and most importantly quality of the end product.

How to Start Working On a Test Project When You Are New in QA?

If testing process is already established and followed by a team then getting on board on such projects is not much difficult. If you are a newbie on the project then start with reading, learning and understanding the project requirements. Identify all major components of the project. List these components and get those reviewed from someone who knows more about the project before diving into the details of those components. Set-up meetings with senior members/mentors and explain your understanding about the project. Discuss your doubts, questions and queries with them and get their clarifications from time to time.

Understanding the project business rules and processes are also very important. Discuss daily tasks with your team lead/manager and communicate status at the end of the day. Once you get familiar with the project start taking small functional or UI testing assignments. Learn and follow the test design, execution and defect management process.

Pay attention to the small details. This is very critical in initial phase of your career on a new project. Talk more with team members and developers. Earn respect in team by adding value to the project.

The above described process is the most easiest and efficient process for any newbie to get started on the QA project.

*****

Importance of Domain Knowledge

FAQ: *"Looking at the current scenario from the industry it is seen that the testers are expected to have both technical testing skills as well as need to be from the domain background. How and when this domain knowledge is imparted to the tester during the testing cycle"*

First of all I would like to introduce three dimensional testing career approach as coined by Danny R. Faught**.** There are three categories of skills that need to be judged before hiring any software tester.

Here are those three skill categories:

1) Testing skills
2) Domain knowledge
3) Technical expertise.

No doubt that any tester should have the basic manual and automation testing skills. With these skills and testers curiosity they can find most obvious software bugs. Then would you say that it is sufficient testing? Would you release the product on the basis of this testing? Certainly not. You will want to have a product looked by the domain experts before the product goes into the market.

While testing any application you should think like an end-user. But every human being has some limitations and one can't be the expert in all of the three skill categories mentioned above. So you can't assure that you can think 100% like how the end-user is going to use your application. User who is going to use your application may have good understanding of the domain. You need to balance all these skill activities so that all product aspects will get addressed.

Nowadays you can see professionals who are being hired in different companies have more domain expertise than technical skills. Current software industry is also seeing a promising trend where many professional developers and domain experts are moving into software testing.

Let's take an example to show importance of domain knowledge. When companies hire entry level candidates they don't expect them to compete with the experienced professionals right away. Why? Because experienced professional certainly have the advantage of domain knowledge and testing experience. They have better understandings of different scenarios that can arise in a real-life production environment and thus they can deliver the application better and faster.

Here are few more examples where you can see the **distinct edge of domain knowledge**:

1) Mobile application testing
2) Wireless application testing
3) VoIP applications testing
4) Protocol testing
5) Banking applications testing

6) Network testing

How will you test such applications without knowledge of the specific domain? Why companies with banking applications prefer candidates with BFSI (Banking, Financial Services and Insurance) domain knowledge? To test BFSI applications candidates should have detailed knowledge of banking processes i.e. the candidates who are banking and finance domain experts. Without this knowledge do you think any tester can test the application effectively? They may test the quite obvious scenarios known to every tester but they will certainly struggle to test core features and complex scenarios unless they get familiar with all financial terminologies and processes.

When the testers know the functional domain better they can write and execute test cases much better. They can effectively simulate the end user actions which is distinctly a big advantage.

What if you are not a domain expert?

Most companies can't afford to have all domain experts on every project. You may be posted on any domain you might not be familiar with. In such cases here is how to get yourself equipped with all necessary domain skills in quick time:

- Read the product documents/user manuals and explore the product as if you are the customer and think of scenarios what customer will do with application.
- If possible, visit the customer site and try to understand how they use the product.
- Read all possible online resources about the domain
- Understand competitor's product and their features. Compare those with your product.
- Participate in events addressing on such domain and meet domain experts.
- Ask domain experts from your team/company to take knowledge sharing sessions
- Ask company to provide training from the subject matter experts

*****

Writing Effective Test Cases

Writing good test cases that have highest possibility of finding defect is a skill. You can achieve this skill with some experience and in-depth study of application for which you are writing test cases.

Here, I will share some tips on how to write test cases and test case procedures.

What is a test case?

"A test case has components that describe an input, action or event and an expected response, to determine if a feature of an application is working correctly."

Levels of writing test cases:

1. Level 1: You will write basic test cases from the available specification and user documentation.
2. Level 2: This is practical stage in which writing test cases depend on actual functional and system flow of the application.
3. Level 3: You will group some test cases and write a test procedure. Test procedure is nothing but a group of small test cases, usually comprising of maximum of 10.
4. Level 4: Test cases automation. This stage will help testers to focus testing efforts on new functionality whereas test automation will take care of regression testing.

Test cases writing process: How you will start?

When you get software requirement specifications (SRS) document you would first go through the requirements to understand the functionality and scope of the application. Once you get the complete overview of the functionality that is being developed you will start writing test cases for the modules assigned to you. First you would start with functional test cases covering all the business rules mentioned in the SRS document.

While writing test cases, it's good idea to start with all requirements on a specific page or screen. So you would write:

- Functional tests,
- Business rule tests,
- Navigation tests (links, buttons etc.),
- Integration tests (integration with other modules),
- Security tests,
- End to end tests,
- Negative tests,
- Load and performance tests

For maximum test coverage you should refer all available requirement documents, tech designs, wire frames, and HTML page mockups. Most of these documents are available while development is in process. You should also capture the requirements discussed in design meetings and cover these requirements in test cases.

How to minimize test case writing and execution time?

Sometime you may notice that writing and executing test cases is taking too much time than required. Test cases writing time can be minimized by following the following techniques:

- Use standard test case templates to write all test cases
- Understand the requirements clearly before starting to write test cases.
- Follow the approach of writing test cases based on testing type sequence. E.g. start writing functional test cases first and then move on to UI, Compatibility, Security and so on.
- If you are not using any test management tool then use MS Excel spreadsheet to write test cases. MS Excel functionalities are so powerful that you can write many similar test cases in very less time. E.g. To write test cases for a long form with many fields, most of these test cases are repetitive with just the field name changed. You can use spreadsheet functions to write these test cases within few minutes. You can even import these test cases if you start using test management tool. I'll suggest starting to use it as soon as possible.
- For writing test cases for various permutations and combinations first use truth table formats and then describe the test cases.
- Use flow diagrams while writing test cases. This will ensure maximum test coverage. Also you will be able to write test cases faster covering all use cases.
- Group your test cases based on application pages/screens or use cases. Both approaches are useful and easy for maintaining test cases.

Test execution time depends on whether you are automating or executing those manually. Manual test execution time varies on complexity of test cases and it has nothing to do with time required to write the test cases. A test case may be written in one minute but it may take hours or even days to execute the same.

Execution time can also be saved by other techniques like using browser auto-fill feature for web based applications. Using this feature webpage data can be filled in one mouse click thus eliminating the need to enter data each time you run a test. Also you may consider executing tests in sequence so that all test cases related to one page (excluding integration and end to end tests) will be executed before moving on to the next page.

Why you should use test management tool?

If you are still using MS word doc files or MS excel spreadsheet to write test cases it's time to switch to test management tool. There are many open source and easy to use test management tools that can ease your test case writing and maintenance task. Test management tools can be used to write test cases, manage requirements, log and link defects to test cases and generate test reports.

Benefits of using test management tool:

- Organizing all your test cases in one place will help you reduce your test case management time.
- Test management tools can provide you with the accurate status of the test progress at any time helping to take some business decisions about the software release.
- You can verify test cases based on priority in case of tight deadlines.
- Duplicate test cases can be avoided.
- QA status can be tracked effectively.
- Easy to phase out test cases execution based on release versions.
- Good test coverage reporting.

How to write test cases?

Fields used in writing test cases:

Test case ID: Unique ID for each test case. Follow some convention to indicate types of test. E.g. 'TC_UI_1' indicating 'user interface test case #1'.

Test priority (Low/Medium/High): This is useful while test execution. Test priority for business rules and functional test cases can be medium or higher whereas minor user interface cases can be low priority. Test priority should be set by reviewer.

Module Name – Mention name of main module or sub module.

Test Designed By: Name of tester

Test Designed Date: Date when wrote

Test Executed By: Name of tester who executed this test. To be filled after test execution.

Test Execution Date: Date when test executed.

Pre-condition: Any prerequisite that must be fulfilled before execution of this test case. List all pre-conditions in order to successfully execute this test case.

Dependencies: Mention any dependencies on other test cases or test requirement.

Test Title/Name: Test case title. E.g. verify login page with valid username and password.

Test Summary/Description: Describe test objective in brief.

Test Steps: List all test execution steps in detail. Write test steps in the order in which these should be executed. Make sure to provide as much details as you can. Tip – to efficiently manage test case with lesser number of fields use this field to describe test conditions, test data and user roles for running test.

Test Data: Use of test data as an input for this test case. You can provide different data sets with exact values to be used as an input.

Expected Result: What should be the system output after test execution? Describe the expected result in detail including message/error that should be displayed on screen.

Post-condition: What should be the state of the system after executing this test case?

Actual result: Actual test result should be filled after test execution. Describe system behavior after test execution.

Status (Pass/Fail): If actual result is not as per the expected result mark this test as failed. Otherwise update as passed.

Notes/Comments/Questions: To support above fields if there are some special conditions which can't be described in any of the above fields or there are questions related to expected or actual results mention those here.

Add following fields if necessary:

Defect ID/Link: If test status is fail, then include the link to defect log or mention the defect number.

Test Type/Keywords: This field can be used to classify tests based on test types. E.g. functional, usability, business rules etc.

Requirements: Requirements for which this test case is being written. Preferably the exact section number of the requirement doc.

Attachments/References: This field is useful for complex test scenarios. To explain test steps or expected result using a visio diagram as a reference. Provide the link or location to the actual path of the diagram or document.

Automation? (Yes/No): Whether this test case is automated or not. Useful to track automation status when test cases are automated.

Download sample test case template: Doc file and XLS file.

Basic format of test case statement:

Verify
Using [tool name, tag name, dialog, etc.]
With [conditions]
To [what is returned, shown, demonstrated]

Verify: Used as the first word of the test case statement.
Using: To identify what is being tested. Here you can use 'entering' or 'selecting' instead of using 'depending on the situation'.

All your test cases should be simple and easy to understand. Don't write explanations that run long like essays. Stick to the point and make the steps easy to understand.

In one of the later chapters we will see why it is important to write clear and easy to understand test cases.

*****

Tips to Design Test Data

Reporting a new defect without proper troubleshooting is a bad testing practice and it is even worse if you file a defect which is due to incorrect or corrupted test data.

In this chapter, I'll provide tips on how to prepare test environment so that any important test case will not be missed by improper test data and incomplete test environment setup.

What is test data?

You need to mention test data while writing test cases. The testers may provide this input data at the time of test execution or in case of automation application may fetch the required input data from predefined data locations. Test data could be any kind of input for the application, file loaded by the application or entries read from the database tables. It may be in any format like XML, system test data, SQL test data or stress test data etc.

Preparing proper test data is essential for effective testing and it is part of test setup process. Generally, testers refer to it as testbed preparation. In testbed all software and hardware requirements are set up using predefined data values.

If you don't have the systematic approach for building test data while writing and executing test cases then there are chances of missing important test cases. The testers can't justify any defect by saying that test data was unavailable or incomplete. It's every tester's responsibility to create their test data according to the test case requirements. Don't rely on test data that is created by other testers or standard production test data, which might not have been updated for months. Always use latest data set according to your testing requirements. Note – If you have standard process to create and update test data periodically you can rely on it instead of creating new data every time.

How to keep your test data intact for any test environment?

Often more than one tester is responsible for testing same build. In this case, more than one tester will be having access to common test data and each tester will (knowingly or inadvertently) manipulate that common data every time build is tested. The best way to keep your valuable input data collection intact is to keep personal copies of the same data so you can restore it as and when needed. It may be of any format like inputs to be provided to the application, input files such as word file, excel file or other photo files etc.

Remember there should be standard process to be followed before restoring your data copies. This process should include informing all the testers about this data change and getting their formal permission before uploading your data set.

How to prepare test data for performance test cases?

Performance tests require very large data sets. Particularly if application is fetching or updating data from database tables then large data volume play important role while testing such application for performance. Sometime creating test data manually will not detect some subtle bugs that may only be caught by actual very large volume of data created by application when it runs in production. If you want real time data, which may be difficult to create manually then ask your manager to make it available from live environment.

This data is usually more useful to ensure you are testing a real-time functioning of application thus improving your chances of finding more defects.

For example when you are testing statistics reports with date range spread over multiple months or even years it is very difficult to create data manually for such a large period, so there is no other easier option than using live server data backup for testing (of course you should mask that data for privacy reasons, but first make sure your client doesn't mind you using this data).

Ideal test data

Test data can be said to be ideal if for the minimum size of data set most of the application errors get identified. Try to prepare test data that will incorporate all application functionality, while not pushing the cost and time constraint for preparing test data and running tests.

How to prepare test data that will ensure maximum test coverage?

Design your test data considering following categories:

Test data set examples:

1) No data: Run your test cases on blank or default data. See if proper error messages are displayed.
2) Valid data set: Create it to check if application is functioning as per requirements and valid input data is properly saved in database or files.
3) Invalid data set: Prepare invalid data set to check application behavior for negative values, alphanumeric string inputs etc.
4) Illegal data format: Make one data set of illegal data format. System should not accept data in invalid or illegal format. Also verify that proper error messages are generated.
5) Boundary condition data set: Data set containing out of range data. Identify application boundary cases and prepare data set that will cover lower as well as upper boundary conditions.
6) Data set for performance, load and stress testing: This data set should be large in volume.

Conclusion:

- Creating separate data sets for each test condition will ensure close to complete test coverage.
- Preparing proper test data is the core part of "project test environment setup".
- The testers cannot pass the bug responsibility saying that complete data was not available for testing.
- The testers should create their test data in addition to the existing standard production data.
- Your test data set should be ideal in terms of cost and time.

Use tips provided in this chapter to categorize test data to ensure complete functional test cases coverage. Be creative; use your own skills and judgments to create different data sets instead of relying on standard production data while testing.

*****

Sample Test Plan Template

Test plan reflects your entire project's testing schedule, approach, methodologies and procedures. This chapter includes a sample test plan outline which can be used for preparing test plan for your projects.

This test plan includes the purpose (i.e. scope, approach, resources, and schedule of the testing activities), to identify the items being tested, the features to be tested, the testing tasks to be performed, the personnel responsible for each task, and the risks associated with this plan.

Find what actually you need to include in each index point.

Test Plan Template:

Name of the Project:

Prepared by:

Date:

Test Plan Index

1.0 Introduction

A brief summary of product being tested. Outline all the functions at a high level.

2.0 Objective and Tasks

2.1 Objectives: Describe the objectives supported by the Master Test Plan, E.g., defining tasks and responsibilities, vehicle for communication, document to be used as a service level agreement, etc.

2.2 Tasks: List all tasks identified by this Test Plan, i.e., testing, post-testing, problem reporting, etc.

3.0 Scope

General
This section describes what is being tested, such as all the functions of a specific product, its existing interfaces, integration of all functions.

Tactics
List here how you will accomplish the items that you have listed in the “Scope” section. For example, if you have mentioned that you will be testing the existing interfaces, what would be the procedures you would follow to notify the key people to represent their respective areas, as well as allotting time in their schedule for assisting you in accomplishing your activity?

4.0 Testing Strategy

Describe the overall approach to testing. For each major group of features or feature combinations, specify the testing approach, which will ensure that these feature groups are adequately tested. Specify the major activities, techniques, and tools which are used to test the designated groups of features.

The approach should be described in sufficient detail to permit identification of the major testing tasks and estimation of the time required to do each one.

4.1 Unit Testing

Definition:
Specify the minimum degree of comprehensiveness desired. Identify the techniques which will be used to judge the comprehensiveness of the testing effort (for example, determining which statements have been executed at least once). Specify any additional completion criteria (for example, error frequency). The techniques to be used to trace requirements should also be specified.

Participants:
List the names of individuals/departments responsible for Unit Testing.

Methodology:
Describe how unit testing will be conducted. Who will write the test scripts for the unit testing, what would be the sequence of events of Unit Testing and how will the unit testing activities take place?

4.2 System and Integration Testing

Definition:
List your understanding of System and Integration Testing for your project.

Participants:
Identify who will be conducting System and Integration Testing on your project. List the individuals who will be responsible for this activity.

Methodology:
Describe how System & Integration testing will be conducted. Who will write the test scripts for the unit testing, what would be sequence of events of System & Integration Testing, and how the testing activity will take place.

4.3 Performance and Stress Testing

Definition:
List your understanding of Stress Testing for your project.

Participants:
Who will be conducting Stress Testing on your project? List the individuals that will be responsible for this activity.

Methodology:
Describe how Performance & Stress testing will be conducted. Who will write the test scripts for the testing, what would be sequence of events of Performance & Stress Testing, and how will the testing activity take place?

4.4 User Acceptance Testing

Definition:
The purpose of acceptance test is to confirm that the system is ready for operational use. During acceptance test, end-users (customers) of the system compare the system to its initial requirements.

Participants:
Who will be responsible for User Acceptance Testing? List the individuals' names and responsibility.

Methodology:
Describe how the User Acceptance testing will be conducted. Who will write the test scripts for the testing, what would be sequence of events of User Acceptance Testing, and how will the testing activity take place?

4.5 Batch Testing

4.6 Automated Regression Testing

Definition:
Regression testing is the selective retesting of a system or component to verify that modifications have not caused unintended effects and that the system or component still works as specified in the requirements.

Participants:
Methodology:

4.7 Beta Testing
Participants:

Methodology:

5.0 Hardware Requirements
Computers
Modems

6.0 Environment Requirements

6.1 Main Frame
Specify both the necessary as well as the desired properties of the test environment. The specification should contain the physical characteristics of the facilities, including the hardware, the communications and system software, the mode of usage (for example, stand-alone), and any other software or supplies needed to support the test. Also specify the level of security which must be provided for the test facility, system software, and proprietary components such as software, data, and hardware.

Identify special test tools needed. Identify any other testing needs (for example, publications or office space). Identify the source of all needs which are not currently available to your group.

6.2 Workstation

7.0 Test Schedule

Include test milestones identified in the Software Project Schedule as well as all item transmittal events.

Define any additional test milestones needed. Estimate the time required to do each testing task. Specify the schedule for each testing task and test milestone. For each testing resource (that is, facilities, tools, and staff), specify its periods of use.

8.0 Control procedures

Problem Reporting:
Document the procedures to follow when an incident is encountered during the testing process. If a standard form is going to be used, attach a blank copy as an “Appendix” to the Test Plan. In the event you are using an automated incident logging system, write down those procedures in this section.

Change Requests:
Document the process of modifications to the software. Identify who will sign off on the changes and what would be the criteria for including the changes to the current product. If the changes will affect existing programs, these modules need to be identified.

9.0 Features to Be Tested

Identify all software features and combinations of software features that will be tested.

10.0 Features Not to Be Tested

Identify all features and significant combinations of features which will not be tested and the reasons.

11.0 Resources/Roles & Responsibilities

Specify the staff members who are involved in the test project and what their roles are going to be (for example, Mary Brown (user) compiles Test Cases for Acceptance Testing). Identify groups responsible for managing, designing, preparing, executing, and resolving the test activities as well as related issues. Also identify groups responsible for providing the test environment. These groups may include developers, testers, operations staff, testing services, etc.

12.0 Schedules

Major Deliverables: identify the deliverable documents. You can list the following documents:
- Test Plan
- Test Cases
- Test Incident Reports
- Test Summary Reports

13.0 Dependencies

Identify significant constraints on testing, such as test-item availability, testing-resource availability, and deadlines.

14.0 Risks/Assumptions

Identify the high-risk assumptions of the test plan. Specify contingency plans for each (for example, delay in delivery of test items might require increased night shift scheduling to meet the delivery date).

15.0 Tools
List the Automation tools you are going to use. List also the Bug tracking tool here.

16.0 Approvals

Specify the names and titles of all persons who must approve this plan. Provide space for the signatures and dates.

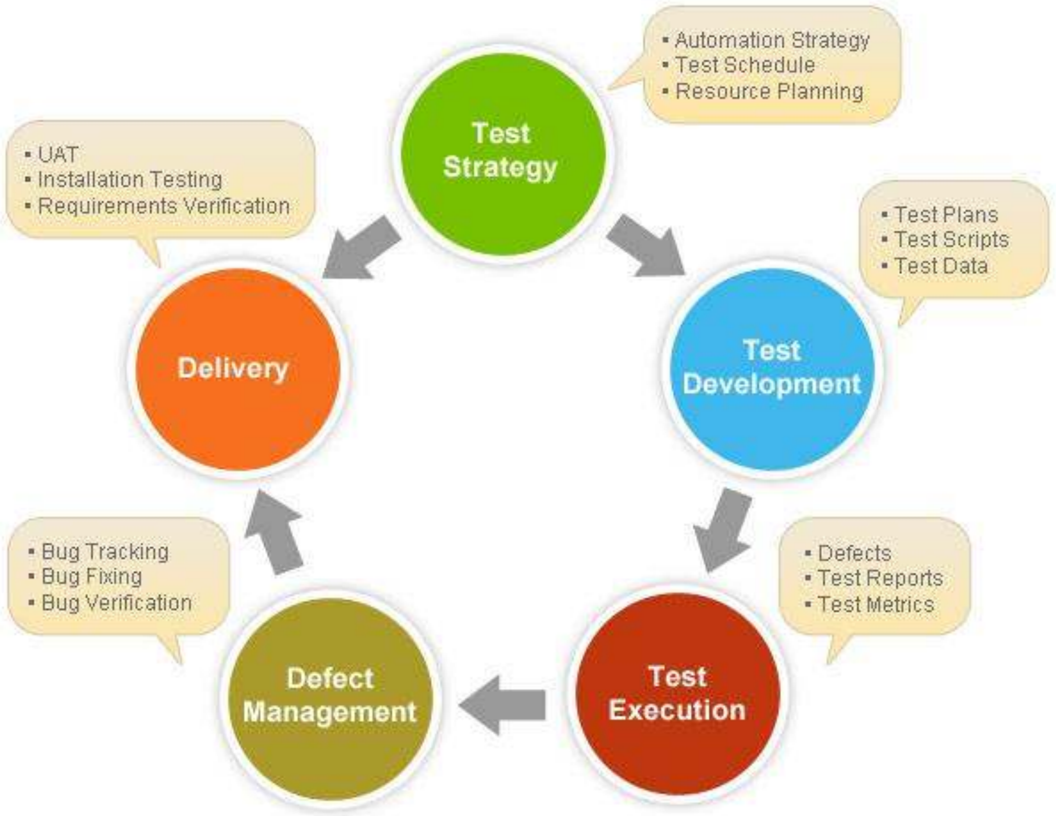
Name (In Capital Letters) Signature Date

1.

2.

Note - You can also download the sample master test plan with example from here.

Please note that we have included the test strategy section in test plan itself. You can even create a different test strategy document with detailed test approach.



[Image credit iconma.com]

Test Strategy vs. Test Plan:

There is lot of confusion between these two documents. Let's start with basic definitions. Generally it doesn't matter which comes first. Test planning document is a combination of strategy plugged with overall project plan. According to IEEE Standard 829-2008, strategy plan is a sub item of test plan (as we covered in above test plan template).

Every organization has their own standards and processes to maintain these documents. Some organizations include strategy details in test plan. Some organizations list test strategy as a subsection in test plan and details are separated out in different document.

Project scope and test focus is defined in test plan. Basically, it deals with test coverage, features to be tested, features not to be tested, estimation, scheduling, and resource management.

Whereas test strategy defines guidelines for test approach to be followed in order to achieve the test objectives and execution of test types defined in testing plan. It deals with test objective, approach, test environment, automation strategy and tools, and risk analysis with contingency plan.

To summarize test plan is a vision of what you want to achieve and test strategy is an action plan designed to achieve this vision!

*****

Testing Software Requirements Specifications (SRS)

Most of the bugs in software are due to incomplete or inaccurate functional requirements. The software code, doesn't matter how well it's written, can't do anything purposeful if there are ambiguities in the requirements.

Cost of fixing the bug after completion of development or product release is too high. Thus it's better to have requirement analysis and catch the requirement ambiguities and fix them in early development life cycle.

How to measure and freeze requirements in design phase?

We need to define some standard tests to measure the requirements. Once each requirement is passed through these tests you can evaluate and freeze the functional requirements.

Here is one example to measure requirements. Let's say you are working on a web based application and the requirement is as follows:

"Web application should respond to user queries as early as possible"

How will you freeze the requirement in this case?
What will be your requirement satisfaction criteria? To get the answer, ask this question to stakeholders: How much response time is OK for you? If the answer is, we will accept the response if it's within 2 seconds, then this is your requirement benchmark. Freeze this requirement and carry the same procedure for next requirement.

Now let's take another example. For one of our company projects we got SRS document for first phase of the application from the client. When we were discussing these requirements, everyone in the team was having their own understanding of the requirements. We quickly realized that there are lots of ambiguities in this document. We decided to first fix these ambiguities before writing test cases or starting design phase. All ambiguous requirements were sent to client with list of questions to get those clarified. Later everyone understood the overall project scope and we were clear to start project design and writing test cases.

Requirements should be clear and consistent - Make sure overall project scope and requirements are clear to everyone before starting to write test cases.

Generally we use our own method to uncover the unspecified requirements. When we read the SRS document, we note down our own understanding of the requirements that are specified, in addition to other requirements that SRS document is supposed to cover. This helps us to ask the right questions about unspecified requirements.

Next criteria for testing the requirements specification is – "discover the missing requirements".

Sometimes SRS writer(s) simply assume some requirements. But requirements should not be based on assumptions. Requirements should be clear and complete covering each and every aspect of the system to be developed. Requirement specifications should mention both types of requirements i.e. what system should do and what should not.

For checking the requirements completeness, divide requirements in three sections, 'must implement' requirements, requirements those are not specified but are 'assumed' and third type is 'imagination' type of requirements. Check if all requirement types are addressed before design phase.

Check if the requirements are related to the project goal - Sometimes SRS writers have their vision about the functionality of later phases of the application. For that they may introduce some requirements in current phase of the application which they may just want to implement but not use until the later parts are completed. We need to understand such requirements carefully. If you thing any requirement is not relevant to scope of the current phase being developed you can ask question about the purpose of that requirement. This will then describe the particular requirement in detail making it easier for designing and testing the application considering the future scope.

How to decide if the requirements are relevant or not?
Use this simple method - Set the project goal and ask this question: If we do not implement this requirement will it cause any problem achieving our specific goal? If not, then this is an irrelevant requirement. Knowing these requirements is useful for project managers. They can ask client to differ these requirements in case of tight deadlines.

Requirements specification document should address following:

- Project functionality (What should be done and what should not)
- Software, Hardware interfaces and user interface
- System Correctness, Security and performance criteria
- Implementation issues (risks) if any

Conclusion:

- **Requirements should be clear and specific with no uncertainty**
- **Requirements should be measurable in terms of specific values**
- **Requirements should be testable having some evaluation criteria for each requirement**
- **Requirements should be complete without any contradictions.**

*****

What is Build Verification Test?

Build verification testing comprises of set of tests that are run on every new build to verify that the build is testable before it is delivered to test team for further testing. These are core functionality test cases to ensure that the application is stable and doesn't have any obvious issues. Typically BVT process is automated. If BVT fails then that build is again assigned to developers for fix.

BVT is also called as smoke testing or build acceptance testing (BAT)

New build is checked mainly for two things:

- Build validation
- Build acceptance

Note – Smoke testing could be subset of BVT tests. Smoke testing does not focus on finer details unlike BVT.

BVT basics:

- It is a subset of regression tests run on every new build to verify main functionality.
- Typically build verification tests should be executed on daily builds. Build should be rejected if BVT fails. A new build is released after the fixes are done.
- BVT saves test team efforts to deploy and test the build when major functionality is broken.
- Build verification tests should be designed carefully enough to cover basic functionality.
- Typically BVT should not run for more than 30 minutes.

BVT is primarily important for checking for the project integrity and whether all the modules are integrated properly or not. Module integration testing is very important, as project development is generally divided into modules for different teams. I have seen many cases of application failure due to improper module integration.

Main task in build compile process **–** Code check-in i.e. including all new and modified project files associated with respective builds. BVT was primarily introduced to check initial build health i.e. to check whether all new and modified files are included in release, all file formats are correct, every file version and language, and flags associated with each file. These basic checks are worth running before build is released to test team for further testing. You will save time and money by discovering the build flaws at the very beginning using BVT.

Which test cases should be included in BVT?

This is very tricky decision to take before automating the BVT task. BVT success mainly depends on which test cases you include in it.

Here are some simple tips to include test cases in your BVT automation suite:

- Include only critical test cases in BVT.
- All test cases included in BVT should be stable.
- All test cases should have known expected result.
- All included critical functionality test cases are sufficient for maximum application test coverage.

Do not include test cases in BVT for unstable modules. For some under-development features, you can’t predict expected behavior as these modules are unstable and you might have list of known failures for these modules. There is no point including test cases in BVT for such modules.

How to include critical functional test cases in BVT?

You can make this task simpler by communicating with all parties involved in project development and testing life cycle. Test team should list all critical test cases and get those reviewed from business team. Once those are approved by business team you can start automating those tests for BVT. While listing critical test cases some quality standards should be set and followed. These standards can be set by analyzing project scope and major features.

BVT example: BVT test cases for text editor application (some sample tests only):

1) Test case for creating a text file.
2) Test cases for writing something into the text editor.
3) Test cases for copy, cut, paste functionality of text editor.
4) Test cases for opening, saving, deleting text file.

These are some sample test cases, which can be marked as 'critical' and for every minor or major changes in application these basic critical test cases should be executed as a part of BVT process.

What happens after executing the BVT suite?

1) The result of BVT execution is sent to project email distribution list (DL).
2) The BVT owner (person executing and maintaining the BVT suite) inspects the BVT result.
3) If BVT fails then BVT owner has to diagnose the cause of failure.
4) If the cause of BVT failure is some defect(s) in the build, all the relevant information with failure logs is sent in the BVT failure notification.
5) Developer on his initial diagnostic replies to this DL with cause of the failure, mentioning whether this is really a bug and what will be the bug-fixing scenario.
6) Upon fixing the defect(s), again BVT test suite is executed. If BVT pass, the build is sent to test team for further testing.

This process is repeated for every new build.

Reasons for BVT failure:
BVT failure doesn't always mean that there are defects in the build. There could be other reasons like error in BVT automation suite, infrastructure error, hardware failures etc. Troubleshooting the cause of BVT failure is very important to take quick and proper actions.

Tips for BVT success

- Do proper analysis and spend time understanding the project goals while writing BVT test scripts.
- While reporting BVT failure, include all possible logs, screenshots and steps to reproduce the issue. This will help developers to fix the issue quickly.
- Including stable functionality test cases in BVT suite ensures that BVT does not fail frequently for known issues.

- Automate BVT process as much as possible. Right from the build release process to BVT result – automate everything that you can.
- BVT automation suite should be maintained and modified from time-to-time. E.g. include test cases in BVT when there are new stable project modules available.
- Have some small penalties when build is failed in BVT. Some candies or a small team coffee party from developer who breaks the build will do. :)

Conclusion

Build should not be assigned to test team unless and until all the tests in BVT have passed. BVT can be executed by developers or testers and BVT result is communicated throughout the team. Immediate action should be taken to fix the bug in case of BVT failure. BVT process should be automated. Only critical test cases should be included in BVT. These test cases should ensure maximum application test coverage.

BVT is very effective for nightly, weekly as well as long term builds. This saves significant time, cost, and resources and after all test team will not frustrate spending time on incomplete build!

*****

Smoke and Sanity Testing Difference

Points to differentiate smoke and sanity testing:

Smoke Testing: This is done to check if critical functionality of the application which was working earlier is also working on new build before taking that build for rigorous testing.

- Smoke testing originated in the hardware testing practice of turning on a new piece of hardware for the first time and considering it a success if it does not catch fire and smoke. In software industry, smoke testing is a shallow and wide approach whereby all areas of the application are tested without getting into too deep.
- A smoke test is scripted, either using a written set of tests or an automated test.
- A smoke test is designed to touch every part of the application in a cursory way. It's shallow and wide.
- Smoke testing is conducted to ensure whether the most crucial functions of a program are working, but not bothering with finer details (such as build verification tests).
- Smoke testing is normal health checkup of a new build of an application before taking it to test in-depth.

Sanity Testing: This is done to check if the small code changes or bug fixes related to a specific functionality are working as expected before taking that build for detailed verification of that specific functionality or bug fix.

- A sanity test is a narrow regression test that focuses on one or few areas of functionality. Sanity testing is usually narrow and deep.
- A sanity test is usually unscripted.
- A sanity test is used to determine a small section of the application is still working after a minor change.
- Sanity testing is done to prove the application is functioning according to specifications. This level of testing is a subset of acceptance testing.
- Sanity testing is to verify whether requirements are met or not, checking all features breadth-first.

*****

Testing Banking Applications

Banking applications are considered to be one of the most complex applications in today's software development and testing industry. Are you wondering what makes Banking application so complex and what approach should be followed in order to test the complex workflows involved? In this chapter we will be highlighting various stages and techniques involved in testing banking applications.

The characteristics of a Banking application:

- Multi-tier functionality to support thousands of concurrent user sessions
- Large scale integration, typically a banking application integrates with numerous other applications such as bill pay utilities, trading accounts, online fixed deposits etc.
- Complex business workflows
- Real time and batch processing
- High rate of transactions per seconds
- Secure transactions
- Robust reporting section to keep track of day to day transactions
- Strong auditing to troubleshoot customer issues
- Massive storage system
- Disaster management.

The above listed ten points are the most important characteristics of a banking application.

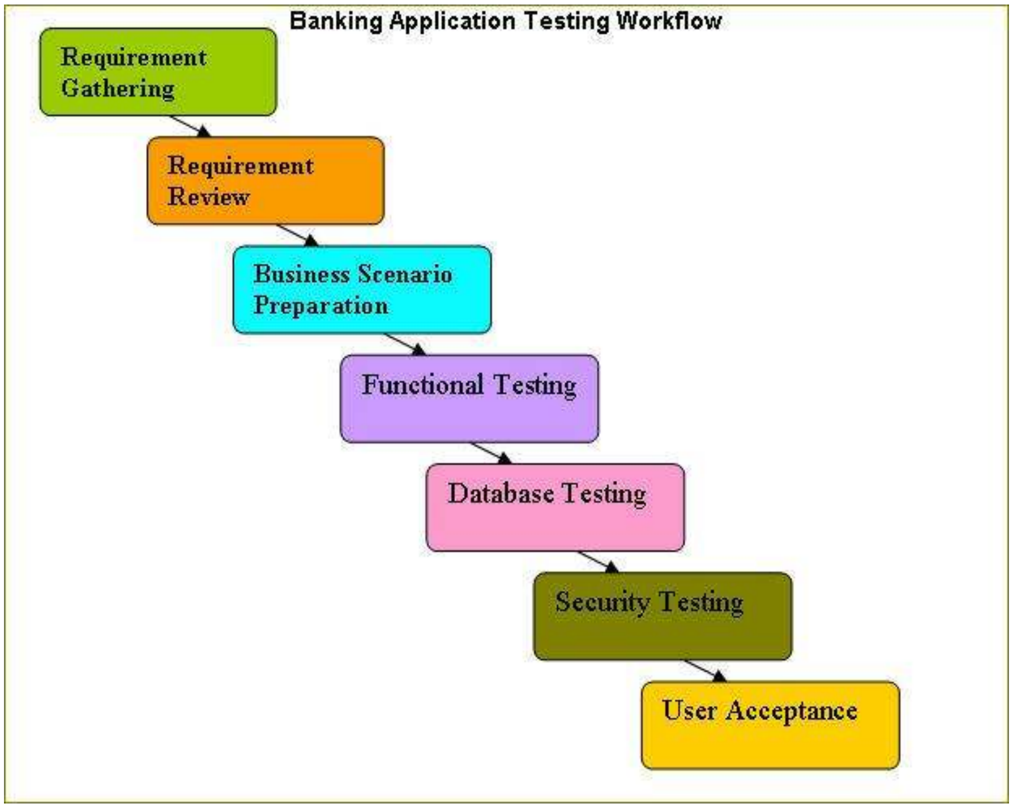
Banking applications have multiple tiers involved in performing an operation. For Example, a typical banking application may have:

1. Web server to interact with end users via web browsers
2. Middle tier to validate the input and output for the web server
3. Database to store data and procedures
4. Transaction processor which could be a large capacity Mainframe or any other legacy system to carry out trillions of transactions per second.

If we talk about testing banking applications it requires an end to end testing methodology involving multiple software testing techniques to ensure:

- Total coverage of all banking workflows and business requirements
- Functional aspect of the application
- Security aspect of the application
- Data integrity
- Concurrency
- User experience

Typical stages involved in testing banking applications are described below, which we will be discussing individually.



1) Requirement Gathering:

Requirement gathering phase involves documentation of requirements either as functional specifications or use cases. Requirements are gathered as per customer needs and documented by Banking Experts or Business Analyst. To write requirements more than one subject experts are involved as banking itself has multiple sub-domains and one full-fledged banking application will be the integration of all. For example, a banking application may have separate modules for Money Transfers, Credit Cards, Reports, Loan Accounts, Bill Payments, eTrading etc.

2) Requirement Review:

The deliverable of requirement gathering is reviewed by all the stakeholders such as QA engineers, development leads and peer Business Analysts. They cross check that neither existing business workflows nor new workflows are violated.

3) Business Scenario Preparations:

In this stage QA engineers derive business scenarios from the requirement documents (Functions Specs or Use Cases). Business scenarios are derived in such a way that all business requirements are covered. Business scenarios are high level scenarios without any detailed steps. Further, these business scenarios are reviewed by Business Analyst to ensure all of business requirements are met and it's easier for the BAs to review high level scenarios than reviewing low level detailed test cases.

4) Functional Testing:

In this stage functional testing is performed and the usual software testing activities are performed such as:

- Test case preparation: In this stage test cases are derived from business scenarios, one business scenario leads to several positive and negative test cases. Usually, tools used during this stage are Microsoft Excel, Test Director or Quality Center.
- Test case review: Reviews by peer QA engineers.
- Test case execution: Test case execution could be either manual or automatic involving tools like QC, QTP, Selenium or any other.

5) Database Testing:

Banking application involves complex transaction which are performed both at UI level and Database level, Therefore Database testing is as important as functional testing. Database in itself is an entirely separate layer and thus it is usually carried out by database specialists and the testing of the same uses techniques like

- Data Loading

- Database Migration
- Testing DB Schema and Data types
- Rules Testing
- Testing Stored Procedures and Functions
- Testing Triggers
- Data Integrity

6) Security Testing:

Security Testing is usually the last stage in the testing cycle as completing functional and nonfunctional are entry criteria to commence security testing. Security testing is one of the major stages in the entire application testing cycle as this stage ensures that application complies with Federal and Industry standards. Security testing cycle makes sure the application does not have any web vulnerability which may expose sensitive data to an intruder or an attacker and complies with standards like OWASP.

In this stage the major task involves the whole application scan, which is carried out using tools like IBM Appscan or HP WebInspect (most popular tools).

Once the scan is complete the Scan Report is published and false positives are filtered out. The rest of the vulnerabilities are reported to development team for fixing depending on the severity.

Other tools for security testing used are: Paros Proxy, Http Watch, Burp Suite, Fortify tools Etc.

Apart from the above stages there might be different stages involved like Integration Testing and Performance Testing.

In today's scenario majority of Banking Projects are using: Agile/Scrum, RUP and Continuous Integration methodologies, and Tools packages like Microsoft's VSTS and Rational Tools.

RUP stands for Rational Unified Process, which is an iterative software development methodology introduced by IBM and it comprises of four phases in which development and testing activities are carried out.

Four phases in RUP are:
i) Inception
ii) Collaboration
iii) Construction and
iv) Transition
RUP widely involves IBM Rational tools.

These are the typical phases involved in testing complex banking application.

*****

Getting Started with Agile Testing

What is agile scrum (sprint) process?

Scrum is a software development process. In today's rapid world, project stakeholders want immediate return on their investments (ROI). They don't want to wait longer to get the full featured product. As a result, nowadays new software development and testing frameworks are catching momentum i.e. Scrum approach.

In scrum, projects are divided into small features that are in turn developed and tested in specific time-frames called as sprint (small cycles). The main idea being -- features should be developed and tested in specified small time-frames. This agile scrum team is handled by a scrum master.

Scrum is an iterative, incremental framework for projects and products or application development. Scrum has become more and more popular software development and testing framework among organizations in the recent past. Many small to large sized IT companies have started to embrace Scrum framework, as this can create excellent quality products in shorter and more quantifiable timeframes than any other traditional software development methodologies. This framework can save companies both time and money.

Soft Skills for a Scrum Team:

What Soft Skills are required to be a Successful Scrum Team?

When we start our regular (Agile) sprints (Cycles of work), we usually find some of the challenges with our team members. These challenges are not part of technical difficulties. It usually occurs with team member's mindset or their soft skills. Many successful Scrum projects have taught us that the success (or the failure) of scrum depends on if the team members support wholeheartedly towards the Sprint.

- **Team Spirit** - On a Scrum team we find only "Our work", "we have completed our Sprint".
- **Communication** - There must be highly collaborative interaction between client and the delivery teams.
- **Commitment** - Ensure that the team embraces the concept of whole-team responsibility and whole-team commitment to deliver working software at the end of each sprint.
- **Transparency** - Transparency among team members and management gives a real momentum to the scrum team.

Scrum Results

If scrum team follows some of above said soft skills, the team's overall velocity will increase significantly. In turn, customers will appreciate the results or updates and also can react quickly to any potential problems. Team can deliver higher value software features in a short time period, keeping everyone on top of changing business conditions.

Agile Testing Challenges

Agile Testers may face a lot of challenges when they are especially new to working with Agile development teams. A tester should be able to apply Root-Cause Analysis when finding severe bugs so that they are unlikely to recur. While Agile has different flavors, Scrum is one of the processes for implementing Agile. Some of the challenging scrum rules to be followed by every individual are:

- Obtain Number of Hours Commitment Up Front
- Gather Requirements / Estimates Up Front
- Enter and update the actual hours and estimated hours daily.
- Build Daily (Nightly)
- Keep the Daily Scrum meetings short
- Code Inspections are Paramount

So, in order to meet the above challenges, an agile tester needs to be innovative with the tools that they have. Great ideas happen when what you have (tangible and intangible) meets the world's deepest hunger to excel.

Automated Regression Testing Challenges in Agile Environment

Agile Projects present their own challenges to the Automation team; Unclear project scope, Multiple iterations, Minimal documentation, early and frequent Automation needs and active stakeholder involvement all demand lot of challenges from the Automation Team. Some of these challenges are:

Requirement Phase:

Customers/software folks take liberty to make too many changes to the requirements. Sometimes, these changes are so volatile that the iterations are bumped off. These changes are one of the greater challenges in implementing Agile Automation testing process.

Selecting the Right Tools:

Automation in the early stages of an agile project is usually very tough, but as the system grows and evolves, some aspects settle in and it becomes appropriate to deploy automation. So the choice of testing tools becomes critical for reaping the efficiency and quality benefits of agile.

Script Development Phase:

The scope and scale of regression testing grows with each sprint and ensures that this remains a manageable task. The testing team usually use test automation for the regression suite.

Resource Management:

The challenging part in the Resource management is to find out test resources with multiple skills and allocating them.

Communication:

In traditional testing, developers and testers are like oil and water, but in agile environment, the challenging task is that they both must work together in tandem to achieve the target.

Daily Scrum Meeting:

Few questions that you must keep asking yourself are -- What is the effectiveness of daily 15 minutes stand up these meetings? How far these meetings help Automation practice Developers?

Release Phase:

The aim of Agile project is to deliver a basic working product as quickly as possible and then to go through a process of continual improvement. This means that there is no single release phase for a product. The challenging part lies in integration testing and acceptance testing of the product.

If we can meet these challenges in a well optimized manner, then Automated Regression Testing in Agile environment is an excellent opportunity for QA to take leadership of the agile processes.

****

Chapter 3

How to Take Your Software Testing Career to New Heights!

By now you should have learned how to get a testing job and improve your testing skills. Now what? Let's discuss more on how to excel in your career and take your software testing career to new heights!

Becoming a Good Tester

What are special abilities and skills that make a tester as a good tester? Well, I was reading Dave Whalen's article "Ugly Baby Syndrome!" in StickyMinds.com and found it very interesting.

According to Dave, “Nobody wants to tell someone they have an ugly baby”, but unfortunately, it's testers job to tell developers that their baby (software) is ugly (full of defects).

It’s a tester’s skill how he/she conveys that message to developers without hurting them.

Below are few tips to handle this situation:

- Be honest and responsive: Tell developers about your plans to test the application.
- Be open and available: If developer asks you to have a look at the application before releasing to QA team, then politely give feedback on it and discuss extra efforts needed if any. Don’t log defects at this stage.
- Let them review your tests: Give them access to view test cases written by you.
- Use of Bug tracker: Do not target developers by making public comments about issues in their work. Use defect tracker to log all defects.
- Don’t take it personally: As a tester, it’s common that you could be a close target during most conflicts. So try to develop a thick skin!

To be a great software tester, you need to develop following characteristics within you:

1. **Be Skeptical** Don’t believe that the build given by developers is bug free or quality outcome. Question everything. Accept the build only if you test and find it defect free. Don’t believe anyone whatever be the designation they hold, just apply your knowledge and try to find errors. You need to follow this till the last testing cycle.
2. **Ensure End User Satisfaction:** Always think what can make end user happy. How they can use the product with ease. Don’t stop by testing the standard requirements. End user can be happy only when you provide an error free product.
3. **Prioritize Tests:** First identify important tests and then prioritize execution based on test importance. Never ever execute test cases sequentially without deciding priority. This will ensure all your important test cases get executed early and you won’t cut down on these at the last stage of release cycle due to time pressure.
4. **Be Open to Suggestions:** Listen to everyone even though you are an authority on the project having in depth project knowledge. There is always scope for improvements and getting suggestions from fellow software testers is a good idea.
5. **Identify and Manage Risks:** Risks are associated with every project. Risk management is a three step process. Risk identification, analysis and mitigation. Incorporate risk driven testing process. Priorities software testing based on risk evaluation.
6. **Focus on Negative Side as Well:** Testers should have test to break attitude. Concentrating on only positive side will almost certainly create many security issues in your application. You should be hacker of your project to keep other hackers away from it. Negative testing is equally important. So cover a good chunk of your test cases based on negative scenarios.
7. **Learn to Negotiate:** Testers have to negotiate with everyone in all stages of project life cycle. Especially negotiation with developers is more important. Developers can do anything to

prove that their code is correct and the defect logged by testers is not valid. It requires great skills to convince developers about the defect and get it resolved.

8. **Stop the Blame Game:** It's common to blame others for any defects which are not caught in testing. This is even more common when the tester's responsibilities are not defined concretely. But in any situation never blame anyone. If an error occurs, first try to resolve it rather than finding someone to blame. As a human everybody makes mistake, so try to avoid blaming others. Work as a team to build team spirit.

*****

What Makes a Good Software Test Leader?

Leadership Qualities

- Test lead should be able to communicate ideas and information effectively – This is the most important leadership skill. Test lead should be able to convey accurate information to team members as well as other teams. After all, he/she is the information bridge between test team, management and development team.
- Leader should be able to set positive examples with his/her behavior.
- Technical and Problem solving skills – Leader who understands the testing process, automation tools, and systems can guide the test team effectively.
- Ability to manage, represent and keep the team together.
- Leader should be confident in his/her thoughts and opinions – This is crucial to gain respect within your team and organization.
- Deep understanding of the business – It's important that leader should have deeper understanding of project objectives, stakeholder needs, and risks.
- Ability to work under pressure, tight schedules and limited resources.
- Ability to prioritize and assign the resources accordingly.

*****

Is Software Testing a Boring Job?

Some testers think that software testing is a repetitive task and one is bound to get bored sooner or later. However, software testing is not a boring job; not unless you are doing it wrong!

Though at times, it will stress you to your limits, it is up to you to let it bore you or not. As quoted by Michael Bolton, testing is a continuous learning process by exploring, discovering and investigating the information you have.

Sometimes testing can seem like a monotonous job especially if you have been doing same repetitive test cases execution work for years. But you should learn to accept certain level of repetition. You can think of automating those repetitive tasks. But don't directly jump to automation because automation is not the solution to everything. Also you may start getting bored executing same tests even when you automate. Hence you need to find different ways to bring some variety in your daily routine job.

If you ask any experienced developer around you he will tell you how boring his job is. Same goes for just about any other profession on the earth. You can't stop doing things just because those are repetitive. Our whole life is just a sequence of repetitive tasks. You need to find innovative ways to do the same things in new way.

Again if you are passionate about software testing you will never find this as a boring job. Believe me testing is fun and challenging job, if you know how to spice it up.

*****

Creative Thinking as a Software Tester!

This is a phrase that you must have come across dozens of times, 'Creative Thinking' or 'Out of the Box Thinking'.

Do we know what it actually means when we say 'Thinking out of the Box'?

As per Wikipedia: "*Thinking outside the box is to think differently, unconventionally or from a new perspective. This phrase often refers to novel or creative thinking*"

But the above definition could be extended when we relate it to our field, Software Testing. When we step into the field of software testing the first thing we are taught or we learn are the Two Boxes, a white box and a black box. Since then all we do is either black box or white box testing. Perhaps this is one reason; we tend to limit our mind from thinking beyond the boxes. Did we ever think that going beyond these could help us in gaining a higher pace towards a solid career in software testing?

Below we will be discussing few techniques which I follow and many of my Mentors follow as well,

Rapid Fire Test Case Creation:

This technique, as the name suggests is about rapidly creating test cases. The first thing that comes to our mind when we talk about test case creation is Requirement Document, an Excel Sheet and some guidelines provided by the organization. For once keep aside all the things, get an idea about what you think you are about to test. Pick up a Pen and a Paper and write as many scenarios as you can within let's say 60 seconds. Repeat the process till you are not able to think of any more scenarios or ideas. And finally review them.

Definitely you will be surprised by the number of ideas you already have without even looking into the requirement document.

Cross Testing Ideas (Analogy):

While testing an application treat it like an entirely different application, which you have used before and then start testing. By doing this you might come across issues, which are not part of the requirements but just some common/generic features, which should be present and often overlooked.

E.g. If you are testing a portal, use it like you use your Email program or any application, which you worked on before and see how the application behaves.

I remember exploring a critical defect using this technique. I was testing secured login feature of a finance application and tried altering the URL and navigating to a different page (which was a defect in my last tested application). By doing this I was able to bypass the login mechanism using Secure ID and this was neither a test case nor any other team member thought that this should be one of the scenarios.

Reverse or Backward Testing Ideas:

What is the normal workflow that you follow while testing? Aren't these the exact same steps, which were used while developing the application, Requirements >> Unit Cases >> Integration Testing >> System Testing or any other approach?

The minds of people working on the development of an application are bound to think in the direction which covers positive scenarios. But the end user might not do the same every time. That is the reason why production defects or UAT defects exist even after extensive rounds of Unit Tests, Integration test and System Tests.

E.g. Requirement says you can upload a file which does not exceed file size of 10 MB. Most testers will try uploading a 1MB, 2MB, 3 MB and so on till 10 MB is reached or error message is displayed. Why don’t you start with 10MB and then try 11MB and then 9 MB? This example is nothing but a BVA but how many of us have tried using BVA in scenarios other than an input box?

Questioning:

Ideally every QA engineer should know the purpose of any requirement. Putting up questions will help a QA engineer to refine his purpose of testing. If a QA engineer is good at questioning (s)he will be good at testing as well. You need to make sure none of the question, how so ever small or silly, is ignored.

And in turn questioning will also enhance the domain knowledge of the person performing the testing.

Researching:

Researching proves to be very beneficial before starting testing. Just be aware of the issues, which other people faced while doing similar assignment. Say, you have to start a cross browser testing as one of your assignments. Before starting the tests researching the issues, which other people encountered while using the same browser will help you find defects even before starting the actual testing.

Pause: an Ice breaker

Sometimes testing could be a monotonous process and when the ideas begin to saturate, you might start feeling that none of the solutions are working out. Worst still, you might even run out of ideas. In such cases, an effective ‘pause’ can do a lot of wonders and could help you kick-start from where you left.

A pause could be a cup of coffee or simply gazing out of the window. Apart from being creative, timing, speed of implementation of ideas and their execution are of high importance. You might get an excellent idea, but what if it is too late to implement it? These are just few ideas which will help you generate more ideas in turn.

*****

Want to Start Automation Testing on Your Project? Here is How to Start!

Why Automation Testing?

1) How do you ensure that new bug fixes have not introduced any new bugs in previously working functionality? Do you manually test complete functionality for previously tested modules every time when you have bug fixes or functionality additions? Well, you might do it manually but then you are not doing it efficiently in terms of company cost, resources, and time.

- Automate your testing task when you have lot of regression work.

2) Do you have manual testing option to load test your application with 1000 users at a time? Of course not - unless you have 1000 testers working on a single test case. Load and performance testing can be done very efficiently using automation tools.

- Automate performance, load and stress testing work.

3) You are testing application where code is changing frequently. You have almost same GUI but functional changes are more so testing rework is more.
- Automate your GUI testing work when application GUI is almost frozen but you have lot of frequently functional changes.

Automation Testing Risks and Challenges

There are some distinct situations where you can think of automating your testing work. Here are few risks associated with automation testing you must read before starting with automation testing:

1) Do you have the skilled resources?
The testers working on automation work should possess programming or scripting knowledge. Do you have such resources on your team? Do they have technical capabilities or programming background to easily adapt to new technologies? Are you ready to invest time and money training your resources for skills required for automation?

2) Initial cost for Automation is very high:
Initial cost associated with automation tool purchase, resources training, and test script maintenance is very high. If you are spending too much and getting merely a GUI rich automation tool with very basic automation scripts then what is the use of automation?

3) Do not think of automating GUI if it is changing frequently:
Cost to maintaining GUI automation scripts could be very high if GUI is changing frequently.

4) Is your application stable enough to automate?
It would be a bad idea to automate test cases in early development cycle (except for agile environment).

5) Are you thinking of 100% automation?
Complete test automation is practically impossible. You have chance of maximum automation in areas like performance testing, regression testing, load and stress testing. But manual intervention is necessary for application areas like user interface, documentation, installation, compatibility and recovery testing.

6) Do not automate tests that run only once:
Identify application areas and test cases that might run only once. Avoid automating such modules or test cases.

7) What is your test automation ROI?
One of the key reasons for test automation is to reduce investment on manual testing efforts and utilize the power of automation. But have you ever thought of calculating real ROI of test automation? What is the guarantee that test automation ROI is greater than manual testing ROI?

Conclusion

Before making decision to select any automation tool make sure it is the best fit for your needs. Both manual and automation testing have their own limitations. Try to find a middle way to benefit from the best of both worlds. Remember automation testing cannot find all the bugs and it's not a replacement for skilled human testers.

*****

Software Testing Best Practices and Tips to Progress in Your Career

Software Testing Best Practices:

Mistakes make man perfect. But don't wait until you make mistakes. Read these testing best practices I've learned from my experience:

1) Analyze test results thoroughly. If any test case fails while logging defect provide as much details as possible. Also provide the possible solution for the problem. Troubleshooting or root cause failure analysis will help you to provide these solutions.

2) Maximize test coverage by covering almost all types of tests. You will not miss any important test scenario when you start preparing and maintaining testing checklist for every release.

3) Know your application and domain – Understand all requirements and project scope before starting to write test cases. Get domain knowledge of your project. The best option to get domain knowledge is Google and other option is your friend circle from the same domain.

4) You can't guarantee a 100% bug free application as well as 100% test coverage.

5) Follow 'divide and conquer' approach for writing test cases. Break application functionality into smaller modules or even units and write test cases for these smaller application parts. This method is again helpful for completing test cases quickly without missing any important tests.

E.g. list all smaller functional modules for your web application. Assume that 'accepting user information' is one of these modules. You can break this 'user information' screen functionality into even smaller parts (like UI testing, security testing, functional testing etc.) for writing test cases.

6) Do not assume anything in testing. You should have test to break attitude. Don't think beforehand that there will not be any defects in the application. Start testing the application with the intent of finding defects.

7) Write your test cases in requirement analysis and design phase itself. This way you can ensure all the requirements are testable.

8) Make your test cases available for developers for review. Let them see your test conditions so that they can take care of the most obvious defects saving time and rework.

9) Go beyond requirement testing. Test application for what it is not supposed to do.

10) Refer defects history report for regression testing. This will help to analyze the most defect-prone modules in the application and prioritize your testing accordingly.

11) Often testers or developers make config changes in code files to test the application in lower test environments. It is expected to change values of these config files before uploading files to production servers. You should note down all such code changes done for testing purpose and at the time of final release make sure that the tester or developer has removed all these before deploying code to live servers.

12) Don't allow developers to make any changes on test environment. For deploying new build get all deployment steps in release notes and deploy code on test servers. This way missing configuration steps can be captured at right place.

13) The testers should discuss and share best testing practices and experience with other testing teams in the organization.

14) Whenever possible make face-to-face communication with developers for resolving your disputes quickly and to avoid any misunderstandings. Once the dispute is resolved send the summary of the agreed upon solution over email. This will keep track of all your communication. This process works well in small agile teams.

15) Make sure you are not running out of time for high priority tasks. Prioritize your testing work from high to low and plan to work accordingly. This will ensure you will never run out of time for important work.

These are some basis tips that will help you doing your day to day testing activities.

Tips to Progress in Your Software Testing Career:

These tips will not only help you survive but also help you advance faster in your software testing career. Make sure you follow them:

Tip #1) 360 degree testing approach – For bug hunting success think from all perspectives. Find all possible information related to application under test apart from SRS documents. Use this information to understand the project better and apply this knowledge while testing.

E.g. If you are testing a partner's website integration with your own application (e.g. through APIs), make sure to understand partner's business fully before starting to test.

Tip #2) Written communication – Keep all communication in written format. Even if you discuss things in meeting circulate meeting minutes over email. No matter how friendly your lead or manager is but keep things in emails or documents.

Tip #3) Try to automate daily routine tasks – Save time and energy by automating daily routine task no matter how small those tasks are.
E.g. If you deploy daily project builds manually, write a batch script to perform the task in one click.

Tip #4) Continuous learning – Never stop learning. Learn and introduce new testing processes that work best for your project. Explore better ways to test application. Learn new automation tools like Selenium, QTP or any performance testing tool. Nowadays performance testing is a hot job destination for software testers! Have this skill under your belt.

Tip #5) Admit mistakes but be confident about whatever tasks you did. Avoid making same mistakes again. This is the best method to learn and adapt to new technologies. Finally learn from your own mistakes. If you don't make mistakes you are not working hard enough.

Tip #6) Get involved from the beginning – Ask your lead or manager to get you (and other QAs) involved in design discussions/meetings from the beginning. This is more applicable for small teams without QA lead or manager.

Tip #7) Keep notes on everything – Keep notes of daily new things learned on the project. This could be just simple commands to be executed for certain task to complete or complex testing steps. By doing this you won't need to ask same things again and again to fellow testers or developers.

Tip #8) Improve your communication and interpersonal skill – Most important to progress in your software testing career at all stages.

Tip #9) Make sure you get noticed at work – Sometimes your lead may not present the true picture of you to your manager or company management. In such cases you should look for opportunities where you can show your performance to top management.
Warning – Don't play politics at work if you think your lead or manager is kind enough to communicate your skill/progress to your manager or top management. In that case no need to follow this tip.

Tip #10) Software testing is fun, enjoy it – Stay calm, be focused, follow all processes and enjoy testing. See how interesting software testing is. I must say it's addictive for some people.

Tip #11) Read, read and read – Keep on reading books, white papers, case studies related to software testing and quality assurance. Always stay on top of the news in software testing and QA industry. Also keep reading testing blogs to keep yourself updated.

*****

Best Certifications in Software Testing

It's not mandatory to have any software testing certifications but if your employer prefers having one then you can go for it. However, to increase your understanding of testing methodologies and processes you can take any of the available certifications. Some companies prefer candidates with certifications like ISTQB, CSTE and HP QTP. Getting these certifications can also increase your confidence while working on live projects.

Taking any of the testing certification can help you improve in your career but it depends how you take this certification. Don't just memorize answers for the sake of exam. Take those seriously and work on how you can use those testing practices in your daily activities.

Below are some of the well-known testing certifications you may want to take:

1) ISTQB
2) CSTE
3) HP QTP Certification

1) ISTQB

Tips to solve ISTQB questions:
ISTQB questions are formatted in such a way that the answers look very much similar at the first look. People often choose the one, which they are more familiar with and end up choosing the wrong answer. Instead, you should carefully read the question twice or thrice, till you are clear about what is being asked in the question. Sometimes more than one or none of the answers could be correct and hence you should be careful with the 'both a and b', 'all of the above' or the 'none of the above' type answers.

Most of the times options are tricky and candidates get confused. To choose the correct answer, you should start eliminating the answers one by one. Go through each option and check whether it is appropriate or not. If you end up selecting more than one option, repeat the above logic for the answers that you selected. This should definitely work.

Before you start with the question papers, please read the material thoroughly. Practice as many papers as possible. This will help a lot because, when you actually solve the papers, you apply the logic that you know.

Following are the levels at which both of these certification are available to suit the needs of every professional:

Foundation Level – This is aimed at professionals who need to demonstrate practical knowledge of fundamental software testing concepts. Certification at this level provides the broad coverage about a specific area. This is specifically designed to enhance the knowledge set and future scope of managers.

Entry criteria – at least six months practical experience.

Advanced Level – This is for the people who have achieved advanced point in their software testing career. This certification is appropriate for anyone having deeper understanding of software testing. This includes the testers, test analyst, test managers, test consultants, quality managers, and business analysts.

Entry criteria – must hold foundation level certification and sufficient practical experience.

Expert Level - This extends advanced level knowledge by providing in-depth practical knowledge in software testing and related subjects. This aims for professionals who have mastery in particular testing subject. There are different expert level modules available depending on the skills. Currently exam is being considered for two modules i.e. improving the test process and test management. There are a couple of other modules in design phase by ISTQB org.

Entry criteria – holds foundation level and advanced level certifications, at least 5 years experience out of which 2 years on the specific expert level topic.

How to apply

To apply for any of this certification you need to go through this list of global as well as country specific exam providers listed on this page.

2) CSTE - Certified Software Tester

CSTE is a commonly used benchmark certification program for all Testers and Managers that emphasize on professional competency and best practices in quality control in IT industry. Requirement or Eligibility criteria to take CSTE certification include and one of the following:

4 year degree from a recognized institution + 2 year experience

3 year degree from a recognized institution + 3 year experience

2 year degree from a recognized institution + 4 year experience

6 years of experience in IT industry

And worked at any time within the prior 18 months in the field covered by certification designation.

Exam Pattern:
The exam is divided into following parts:

Part 1 – Multiple choice 100 questions – objective / 75 minutes time limit

Part 2 – Short answers / essay 12 questions – subjective / 75 minutes time limit

Exam duration is 2 ½ hours and the passing mark is 70% average score of both the parts.

How to Apply:
The applicant should submit the request on online certification candidacy application in the customer Portal with the payment of $350 (PDF + initial exam) or $420 (PDF + Book + CD + initial exam).

See more details about eligibility and fees on this page. You can apply for this certification through Customer Portal at Software Certifications website.

3) HP QTP Certification

QTP certification by HP is meant for QuickTest Professional. This is for the professionals who wish to gain mastery in testing tools by HP called QuickTest Professional (QTP). Since 2012, HP has come up with the HP QTP Certification v11.0 (Exam HPO-M47 – HP QTP 11.0 Software).

How to apply:

HP manages all their certification programs through Pearson VUE org. Listed below are the steps to apply for HP0-M47 (or any other HP) certifications:

- Get you HP learner ID on this page. This is a unique id required for all certifications. You can create your account on above page to get this learner ID.
- Create new Pearson VUE profile at this page using HP's Learner ID you received in above step.
- Once you create your account you can apply for the QTP certification by making direct payment or purchasing a payment voucher at this page.

*****

How to Ask for Promotion and Salary Increase in Your Next QA Performance Review (Appraisal)

How to face QA performance appraisal confidently?

Many companies conduct periodic reviews to give feedback to employees on their performance and to assist them in developing their career. The performance appraisal period vary in different companies and it is typically six months or one year in most of the companies. Performance appraisal is the right time to ask for your promotion as well as salary raise. ***Remember you won't get anything unless you ask for it!***

Why performance appraisal?

To reward employees for their good work, appraisals are meant to assist employees to develop their career and enable them to reach their full potential. Performance appraisal process involves discussion of an employee's achievements for the last review period (typically one year) and identifying areas for improvements for the next review period. It helps employees to develop clear performance objectives for the next review period.

In this chapter, I will concentrate more on "QA performance appraisal", the essential skills and parameters used to judge and rate the QA performance.

This chapter will help you in following ways:

- If you are a fresher and not yet faced any performance appraisal, you will get better idea of what is performance review and how to face it.
- If you are an experienced tester / quality assurance engineer then you will know "how to ask for promotion and salary increase in your performance review".
- How to effectively summarize your hard work and responsibilities to garner good impression in the eyes of the management.

In companies having yearly appraisal policy, performance appraisal process begins one month before the end of financial year (usually March). Performance review forms are circulated to every eligible employee with detailed instructions on how to fill the form and process to send completed forms. After that, face-to-face review meetings are scheduled with reviewers.

Following major activities get discussed in review meeting:

- Project you worked on in previous year
- Employee's overall performance
- Comments on performance ratings given by employee and reviewer
- Employee feedback
- Areas for improvements
- Performance planning for the next year.

What are the criteria used to rate the employee performance?

We are specifically discussing about QA performance appraisal, so here are the main parameters considered while rating software testers/QA persons.

Software testing skills:

1. Ability to find critical bugs
2. Bug reporting skill
3. Ability to automate work
4. Test case design ability
5. Testing completeness and coverage

Management skills:

1. Effective role model
2. Team motivation skill
3. Estimation and scheduling ability
4. Ability to anticipate and address issues
5. Mentoring ability
6. Planning and time management skill

Personal skills:

1. Can work independently?
2. Team player
3. Self-learning abilities
4. Discipline
5. Willing to learn?

6 Takes initiative?
7 Admit mistakes?
8 Grasping skill

Other skills:

1 Communication (Written and verbal)
2 Documentation skill
3 Interviewing skill (If applicable)
4 Training and presentation skill

Based on these parameters employee can provide self-rating from 1 to 10 for individual parameter and the overall rating is calculated as the average of all these ratings. Reviewer's rating is also present for each column and so is the reviewer's final rating.

Ratings are classified as:
Rating from 1-5: Poor performance
6: Need improvements
7: Meet position requirements
Ratings from 8-9: Exceed position requirements
10: Exceptional! Exceed all requirements all the time.

In review form, employee provides feedback on his/her work, company culture, work process, and management style for the review period.

Employee's feedback section is the best section to ask for promotion or salary hike. Mention your overall and relevant QA experience and your ability to handle more challenging work assignments. This will make management to think on your expectations about promotion and salary increment.

Reviewer will fill the "Employee performance planning for the next appraisal" section. In this section reviewer addresses the improvement areas like technical and non-technical skills or other personal improvements. Reviewer needs to mention some specific goals that the employee should meet in the next appraisal period. This will become the base objective for your next appraisal.

This is the overall appraisal process. Now the key part is how and when to ask for promotion and salary raise?

Key points you need to study before asking for promotion and pay raise:

1) What were your major achievements in the past year?
You should be ready with a list of key projects you completed or worked on in past year. How was the overall quality of work in this period? Note down some examples to illustrate your contribution to company growth.

2) Positive attitude:
Companies want employees with positive attitude. Management thinks on employee's leadership qualities and capabilities before promoting to senior or lead position.

3) Your relationship with your boss and co-workers:
This is a crucial point. Make sure you don't have any disputes between you and your boss or co-workers. You should be a fair team player.

4) Any major work issue in previous year?
You should be aware of project issues raised due to your mistakes. If these issues were major then think twice before asking for promotion or pay raise. If the issues are minor and you were not directly responsible for those issues then you should have explanation ready for these issues when management raise these negative points in appraisal meeting. Make sure you don't blame any co-workers for any issues that were a direct result of your own mistakes.

5) Explain why you deserve promotion:
You need solid work portfolio to explain this. Put forward your contribution to company and how this helped to improve the company.

6) Are you prepared to handle senior level position challenges?
A senior level position means more responsibilities. You need to have both technical as well as management skills to handle such positions. Explain how you are the best fit for the new position.

7) Be prepared to have your exact salary expectations:
If management is ready to promote you then you might get to this question: How much pay raise do you expect this year? You should be ready with exact figure or a range based on current market salary range for your new position. Before presenting any amount you should also consider your current salary, company's previous salary hike records and your accomplishments for the appraisal period.

8) Know the exact time for getting promotion and pay raise:
If you got promotion in last performance appraisal then ask for promotion in current appraisal only if you did some outstanding work. If company is in financial problems then wait till company comes out of this situation (but don't wait too much longer).

Conclusion

Be professional and specific about your expectations during your appraisal. Learn to ask for promotion and salary raise, otherwise you may not get anything. Provide examples to illustrate your contribution to company growth. Be prepared for any outcome. It may be positive or negative. You should be calm in your response and don't forget to thank your boss and handshake at the end of the appraisal meeting.

*****

A Guide to Surviving the Economic Downturn

This chapter is for those who are already into this troubling world and wondering how to get stabilized here or want tips to survive when a recession strikes.

Though it is hard to realize at first, it is important to know that recession could be a blessing in disguise and often gives us an opportunity to adapt ourselves to the new environment to survive.

Here are my top three tips to survive in a recession

1) Upgrade your skills – make a stronger profile

When I was young my teachers used to emphasize on garnering specialized skills. Similarly in our testing world, specialization is getting much attention and many organizations are able to charge differential rates to the clients for a performance architect or automation developer etc. Of late, requirements are coming from the clients differently. Clients need a test automation architect, who is good in QTP, LoadRunner, Perl, UNIX, SQL, Java etc.

How you can deal with this change? Get yourself specialized in one stream and then move on to learning other skills. Unless you are multi-skilled, it is going to be very difficult to survive in this emerging IT industry. When you are out of a project, take that as a boon period to upgrade your skills. Try some testing certifications like ISTQB, CSTE, CSQA or tools related certifications like AIS, ASE or domain related certifications in Insurance, Banking or telecom. It'll help you to improve your profile.

2) Learn to manage stress

One more critical skill, which we need to learn, is stress management. As we move on in our career, we will get more work related pressure. I know several of our colleagues who never see the sunlight during the project execution days and this will only "get intensified" further due to the cost cutting and operating margin pressure on the IT companies.

We need to learn to live with these pressures in life. A class on Yoga or meditation will help to stabilize your mind.

3) Be always ready to face challenges

In recession, every other day you will see news of IT firms downsizing employees. This is the hard reality in most of the countries where job security is associated with the work life. It is not the question of you being a performer or non-performer, but the question of available business and required resources.

One should not get demotivated or depressed by these events in life. If you happen to face this, take it bravely and face the challenges. Cross-platform skills will help you survive during this time. I know many examples like one of our earlier delivery managers in insurance vertical is now the CEO in a hospitality company. Life gives you enough opportunity if you are ready to take challenges and use every opportunity to upgrade your skills.

As we grow, not only our skills should grow with us, but also the certifications and memberships, which give an enhanced look to our profiles. Also, our confidence to face the challenges should also improve.

Any recession is the right time for all of us to introspect ourselves on the current skills and move forward. Nothing is impossible if you have the determination to WIN.

****

Chapter 4

Defect Management Skills - How to Manage Defects Like a Pro!

What is a Bug/Defect/Issue?

By definition, a software bug is an error, flaw, mistake, failure, or fault in a computer program or system that produces an incorrect or unexpected result, or causes it to behave in unintended ways. Most bugs arise from mistakes and errors made by people in either a program's source code or its design, and a few are caused by compilers producing incorrect code. I've seen people using the words bugs, defects and issues interchangeably to define something wrong in the product. From my testing world: All these words (bugs, defects and issues) mean the same and one of below four rules apply:

1) Application does something, what system documentation says it should not.

2) Application does not do something, what system documentation says it should.

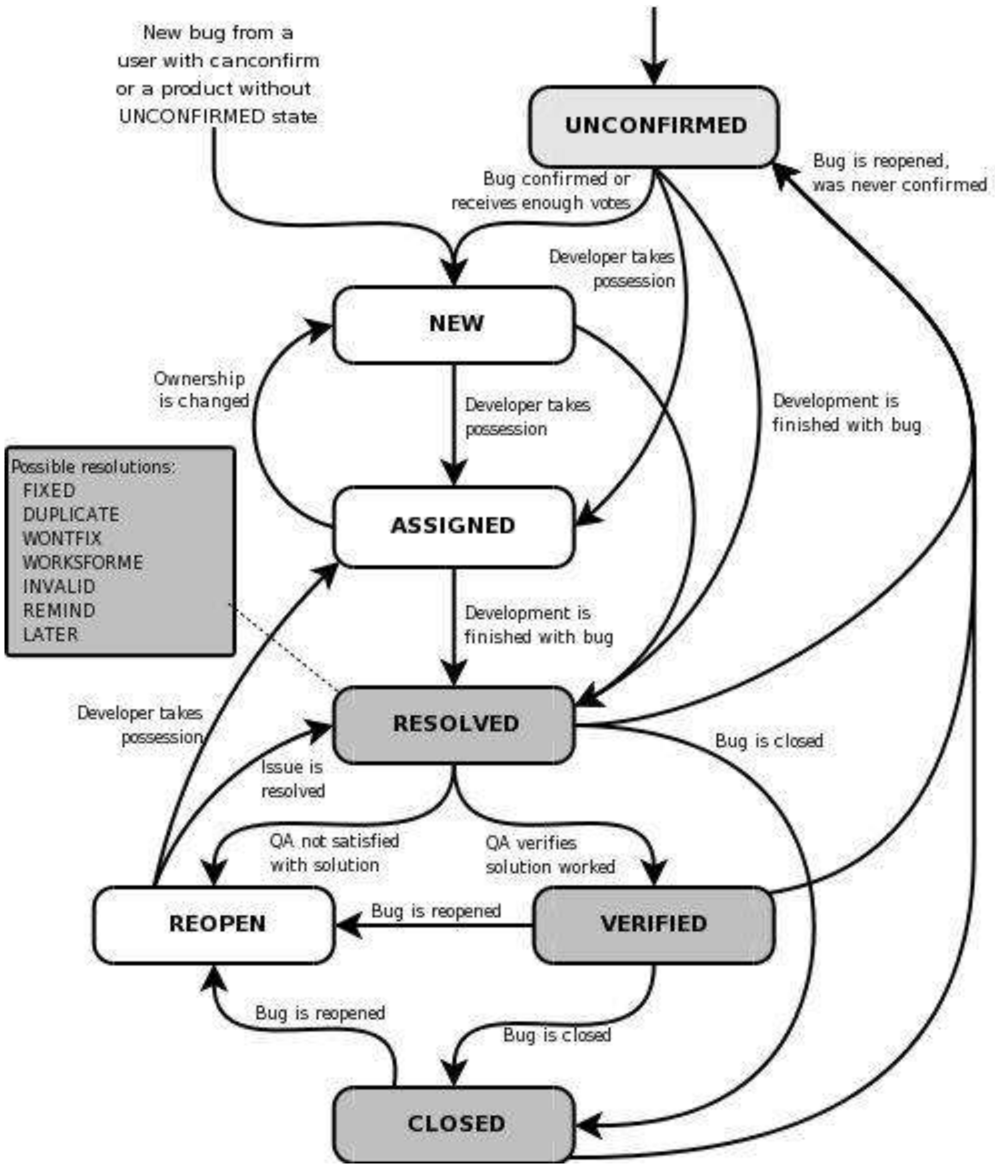
3) Application does something what system documentation does not mention.

4) Application does not do something what system documentation does not mention, but it should do.

*****

Defect Life Cycle

Refer below image for defect life cycle:


New bug from a user with canconfirm or a product without UNCONFIRMED state
UNCONFIRMED
Bug confirmed or receives enough votes
Bug is reopened, was never confirmed
Developer takes possession
NEW
Ownership is changed
Developer takes possession
Development is finished with bug
Possible resolutions: FIXED DUPLICATE WONTFIX WORKSFORME INVALID REMIND LATER
ASSIGNED
Development is finished with bug
Developer takes possession
RESOLVED
Issue is resolved
Bug is closed
QA not satisfied with solution
QA verifies solution worked
REOPEN
Bug is reopened
VERIFIED
Bug is reopened
Bug is closed
CLOSED

[Reference – Bugzilla defect life cycle]

Let's dig deeper into the defect status from its discovery to resolution:

1. When the tester or QA engineer files a defect its status is **NEW**.
2. If the defect is not related to current build or can't be fixed in this release or it is not important to fix immediately then the project manager can set the defect status as resolved with proper resolution.

   Note - There can be a triage team to update/assign new defects.

3. When the project manager assigns defect to a developer it is marked as **ASSIGNED**.

Note – the project manager can assign defect to himself/herself if there is no further action needed from the developer (mostly in case of invalid or duplicate defects).

4. When the developer makes code change and verifies the fix then it is marked as **RESOLVED** and assigned to test team for verification.

   Following are the various defect resolutions (set only when defect status is RESOLVED):

   - If defect is fixed it is marked as FIXED.
   - If similar defect is already logged it is marked as DUPLICATE.
   - If defect can't be fixed or it is working as intended then developer can mark it as WONTFIX.
   - If the actual and expected results are same then defect is marked as WORKSFORME.
   - If the functionality is working as per the specifications or it is just due to some misinterpretation or caused by incorrect testing configuration then it is marked as INVALID.
   - If defect is not related to current release or it is not important to fix in this release it can be marked as LATER.

5. Defects with resolved-fixed status are assigned to test team for verification. If the defect resolution is not satisfactory and issue is still open then the tester can mark it as **REOPEN**. Defects with reopen status are again assigned to developers for further action.
6. The tester or the defect reporter will re-test the test case and if the solution is satisfactory then the defect is marked as **VERIFIED**.
7. When the defect is verified and there is no pending action it can be marked as **CLOSED**.

*****

Sample Defect Report

Below sample defect report will give you an exact idea of how to report a defect in defect tracking tool.

Here is the example scenario:

Let's assume a scenario where admin wants to create new user account in an application with following steps:

- *Log into the application*
- *Navigate to 'Users' menu and click on the 'New User' drop down menu option*
- *Enter data for all required fields on the page (e.g. First Name, Last Name, Age, Address, Phone etc.).*
- *Click on 'Create User' button in order to create new user.*

Now you should see a message saying, "New user has been created successfully".

But while executing this scenario manually assume the application crashes showing an error page on screen. (Capture this error message window and save as an image file).

How will you report this defect effectively?

Here is a sample defect report for the above scenario:
(Note - some 'defect report' fields might differ depending on your defect tracking system)

Defect report example:

Below defect report information is sufficient to log a defect in Bugzilla

Title: Application crash upon click of 'Create User' button while creating a new user.
Defect ID: (automatically created by the defect tracking tool when you save the defect report)
Area/Path: Users > New User
Build Number: Version Number 1.0
Severity: High (High/Medium/Low) or 1
Priority: High (High/Medium/Low) or 1
Assigned to: Developer-X
Reported By: Your Name
Reported On: Date
Reason: Defect
Status: New/Open/Active (depends on the defect tracking tool)
Environment: Windows 2003/SQL Server 2005

Description: Application crash upon click of the 'Create User' button on 'New User' page, making it impossible to create a new user in the application.

Steps To Reproduce:

1) Login to application under test using admin credentials
2) Navigate to the Users > New User menu
3) Fill all the user information fields
4) Click on 'Create User' button
5) Notice the error page with error [include error summary here]

Expected result: On click of the 'Create User' button, user should be prompted with a message "New user has been created successfully" and the user account should get created in the database.

Attach following files with this defect report:

- Error logs if any from server log files
- Screenshot of the error page

Save this defect in defect tracking tool and note down the defect id for future reference. Respective developer and module owner will get notified by email for further actions.

*****

Writing a Good Defect Report

"***The point of writing problem report is to get bugs fixed***" – Cem Kaner.

If a tester is not reporting a defect with proper steps, the developer will more likely reject this defect as invalid.

Wiring a good defect report is a skill. Let's discuss more in this chapter on how to improve this defect writing skill.

Defect report template

Defect Reporter: Your name.

Product: Product name.

Version: Product version number if any.

Component: Major sub module of the product.

Platform: System environment where you found this defect. E.g. 'PC', 'MAC', 'HP', 'Sun' etc.

Operating system: OS where you found this defect. E.g. Windows, Linux, Unix, SunOS, or Mac OS. Also mention the OS versions if applicable. E.g. Windows 2k, Windows XP, Win 7, Win 8 etc.

Priority:
When should the defect be fixed? Generally set from P1 to P5. P1 is with highest priority and P5 is "fixing when time permits".

Severity:
This describes the defect impact.
Types of Severity:

- Blocker: No further testing work can be done.
- Critical: Application crash or loss of data.
- Major: Major loss of function.
- Minor: minor loss of function.
- Trivial: Some UI enhancements.
- Enhancement: Request for new feature or some enhancement in existing one.

Status:
First status is 'New' when defect is logged. Later on, the defect status is changed to Fixed, Verified, Reopened, Won't Fix etc.

Assigned To:
If you know the developer who is responsible for the defect then assign it to him/her. Otherwise, keep it blank so that the module owner or test lead will assign the defect to responsible developer. Also add the manager's email address in CC list. All this can be controlled by test management tool.

URL:
The page URL if any where the defect is observed.

Summary:
A brief summary of the defect; mostly less than 60 words. Summary should reflect the problem.

Description:
Detailed description of the defect. Use following fields for description:

- Reproduce steps: Clearly mention the steps to reproduce the defect.
- Expected result: How the application should rather behave when above mentioned steps are carried out.
- Actual result: What the actual result is by running above steps i.e. the defect behavior.

These are the important steps in defect report. You can also add the "Report type" as one more field which will describe the defect type.

Types of defect reports:

- Coding error
- Design error
- New suggestion
- Documentation issue
- Hardware problem

How to write a good defect report?

If the defect logged by you is not reproducible it will never get fixed. Clearly mention the steps to reproduce the defect. Don't assume or skip anything. Also do not write an essay about the problem. Defects with clear and concise steps to reproduce will help developers to reproduce and fix it quickly. Do not combine multiple problems even they seem to be similar. Write different defect report for each problem.

Below are some defect writing guidelines:

1) Report the problem immediately:

If you found any defect while testing, do not wait to write detail defect report later. Instead write the defect report immediately. This will ensure that all the defect steps are covered. If you decide to write the defect report later then you may forget and miss the important steps.

2) Reproduce the defect two or three times before writing the defect report:

Make sure your steps are clear enough to reproduce the defect without any ambiguity. If your defect is not reproducible every time you can still file a defect mentioning the sporadic nature of the defect.

3) Test the same defect occurrence on other similar modules:
Sometimes developer use same code for different similar modules. So there is a fair chance that the defect found in one module can occur in other similar modules as well. You can even try to find more severe version of the defect by following this approach.

4) Write a good defect summary:
Defect summary will help developers to quickly analyze the defect nature. Poor quality report will unnecessarily increase the development and testing time. Remember defect summary is used as a reference to search defects in defect inventory.

5) Read the defect report once again before hitting submit button:
This will help to understand if overall defect problem is stated clearly or not. Check if any sentence is having any sort of ambiguity that can lead to misinterpretations. Misleading words or sentences should be avoided in order to have a clear defect report.

6) Do not use abusive language:
It's nice that you did a good work and found a defect but do not use this as a chance to criticize the developer or to attack any individual.

No doubt that your defect report should be of high quality if you wish the developer to be able to understand and replicate the defect to be able to fix the same. Focus on writing good defect reports; spend some time on this task as this is the main communication point between tester, developer and manager. Managers should inform their team that writing a good defect report is the primary responsibility of any tester. Your efforts towards writing a good defect report will not only save time and resources but also create a good relationship between you and developers.

*****

Tips and Tricks to Quickly Find Critical Bugs

If you are a software tester or a QA engineer then you must be thinking of new ways to find bugs in an application.

Many people think that finding a blocker bug (such as a system crash) should be considered rewarding. While there is some truth to such assumption, you should also try to find out the bugs that are most difficult to find and those that always mislead the users.

Finding such subtle bugs is the most challenging work and it gives a tester much higher level of satisfaction. Here is my experience of one such subtle bug that was not only difficult to catch but was difficult to reproduce also.

I was testing one module from my search engine project. We were doing only around 20% automation so most of the things were manual. The module under test was related to traffic and revenue stats of different affiliates and advertisers. Testing such reports having millions of records is always a painful task. When I tested this report it was showing the data processed accurately for some time but incorrect after some time. It was strange and confusing to see the results. I was even not able to predict the exact nature of this problem.

As it turned out, there was a cron job (cron job is an automated script that runs after specified time or condition) to process the log files and update the database. But there were multiple cron jobs that were running on log files and database to synchronize the total data. There were two cron jobs running on same table within some time intervals. Thus there was a column in the table that was getting overwritten by other cron job, causing data inconsistency. It took us a long time to figure out the problem due to the vast database processes and different cron jobs.

My point is - try to find out hidden bugs in the system that might occur for special conditions and cause severe impact on the system. You need some tips and tricks to find such defects.

What are those tips and tricks?

1) Make sure to have in-depth understanding of the complete application before starting to test.
2) Prepare good test cases (test cases that have high probability of finding defects) before starting to test. Functional test cases around the critical functionality of the application should be covered in detail.
3) Create sufficient test data before test execution. This data set should include all test conditions and also the database records if you are going to test database related application.
4) Repeat the tests on different test environments (systems, browsers, OS, hardware etc.).
5) Try to find out the result patterns and then compare your results with those patterns.
6) When you think that you have completed most of the test conditions and when you think you are tired take a break and start afresh.
7) Use your previous test data patterns to analyze the current set of tests.
8) Try some common test cases for which you found the bugs in some different application. If you are testing input text box try inserting some special characters or HTML tags as an input.
9) Last but not least - try hard to find the bug as if you are testing only to break the application!

Few more negative testing tips to find defects quickly -

- Provide special characters as input to the text fields, characters as the input to number fields and the other way around.
- For web application, submit the form without entering any value.
- Press Enter button on any web application form. Page should get submitted for valid data.
- Test input fields for boundary value conditions.
- Try pressing submit, add to cart or continue button multiple times in short time interval.

*****

Proper Defect Troubleshooting to Avoid Invalid Bugs

I hate "Invalid bug" label from developers for the bugs reported by me, don't you? I think every tester should try to get all bugs resolved. This requires a great level of troubleshooting and bug reporting skills.

The main reason for bugs being marked as invalid is "insufficient troubleshooting" by the tester before reporting the bug. In this chapter, I will focus only on troubleshooting to find main cause of the bug. Troubleshooting will help you decide whether the ambiguity you found in your application under test is indeed a bug or any test setup mistake.

Out of total invalid bugs, 50% bugs get marked as "invalid" only due to the tester's incomplete testing setup. Let's say you found an ambiguity in the application under test and you are preparing the steps to report this ambiguity as a bug. But wait! Have you done enough troubleshooting before reporting this bug? Have you confirmed if it is really a bug?

What troubleshooting steps do you need to perform before reporting any bug?

Troubleshooting of:

- What's not working?
- Why it's not working?
- How can you make it work?
- What are the possible reasons for the failure?

Answer for the first question "what's not working?" is sufficient for you to report the defect steps in defect tracking system. Then why is the need to answer the remaining three questions? Think beyond your responsibilities.

A good tester finds bugs and reports them; but a great tester finds where it is broken and what needs to be done to fix the same.

You should be able to suggest all possible solutions to resolve the bug including efficiency and drawbacks for each solution. This will increase your credibility within your team and will also reduce the possibility of getting your bugs rejected, not due to this credibility but due to your troubleshooting skills.

Before reporting any bug, make sure it isn't your mistake while testing, you haven't missed any important test environment flag to set or you might have not configured your test setup properly.

Troubleshoot the reasons for failure in the application. Report the bug only after proper troubleshooting. I have compiled a troubleshooting list of different reasons for failure.

Common reasons for failure:

1. Often, some global configuration file is used to pick or set some application flags. If you are using any such file make sure it is up-to-date as per the application requirements. Failure to maintain this file with proper config values will certainly lead to malfunctioning of your application under test. You can't report this as a bug.
2. Check if your database is properly configured: Missing table is often the main reason why your application will not work properly. I have a classic example for this: One of my projects was querying many monthly user database tables for showing the user reports. First table's existence was checked in master table (this table was maintaining only monthly table names) and then data was queried from different individual monthly tables. Many testers were selecting wider date range to see the user reports. But often it was crashing the application as those tables were not present in database of the test machine server, giving SQL query error and they were reporting it as bug, which subsequently was getting marked as invalid by developers.

   Missing entry in master table is certainly an issue but as you are reporting another issue instead of this, it will likely get marked as invalid. So proper troubleshooting will help to find the root cause and log it in your defect clearly.

3. If you are working on automation testing project then debug your script twice before coming to conclusion that the application failure is a bug.
4. You are not using valid access credentials for authentication.
5. There is any other hardware issue that may not be related to your application.
6. You are testing on a setup which is not recommended. You can do failover testing but not the functional testing.
7. You are testing on an unsuccessful or incomplete deployment with missing or corrupt files and registry entries.
8. Before logging a defect you are not checking 'system event viewer' for details/logs. You can trace out many failure reasons from system event log file.

These are all common troubleshooting steps. Failure to follow these can cause great impact on your credibility within your team.

****

Chapter 5

Web Testing Guide!

Testing Client Server and Web Based Applications

Projects are broadly divided into two types of architecture:

- 2 tier applications
- 3 tier applications

Client-Server testing

This type of testing is usually done for 2 tier applications (usually developed for LAN).
Here we will be having a front-end and back-end.

The application launched on front-end often have forms and reports, which are used to input, monitor and manipulate data

E.g. applications developed in VB, VC++, Core Java, C, C++, D2K, PowerBuilder etc.,

The back-end database for these applications could be MS Access, SQL Server, Oracle, Sybase, MySQL or Quadbase.

The tests performed on these types of applications are
- User interface testing
- Manual support testing
- Functionality testing
- Compatibility testing & configuration testing
- Intersystem testing

Web application testing

This is done for 3 tier applications (developed for Internet/Intranet/Extranet).
Here we will be having browser, web server and DB server.

These applications are accessible in browser and are developed in languages like HTML, DHTML, XML, JavaScript etc.

Applications for the web server are developed in Java, ASP, JSP, VBScript, JavaScript, Perl, Cold Fusion, PHP etc. (All the manipulations are done on the web server with the help of these programs developed)

The DB server often has oracle, SQL server, Sybase, MySQL etc. (All data is stored in the database available on the DB server)

The tests performed on these types of applications are
- User interface testing
- Functionality testing
- Security testing
- Browser compatibility testing
- Load/stress testing
- Interoperability testing/intersystem testing
- Storage and data volume testing

The types of tests performed on these applications are:

1) User interface testing for validations & user friendliness

2) Functionality testing to validate behavior, input/output, error handling, data manipulations, service levels, order of functionality, links, content of web page & back-end coverage
3) Security testing
4) Browser compatibility
5) Load / stress testing
6) Interoperability testing
7) Storage & data volume testing

Few more points to summarize the difference between client server and web applications:

Client/Server application:

- Application runs in two or more machines
- Application is menu-driven
- Connected mode (connection exists until logout)
- Limited number of users
- Less number of network issues when compared to web app.

Web application:

- Application runs in two or more machines
- URL-driven
- Disconnected mode (state less)
- Unlimited number of users
- Many issues like hardware compatibility, browser compatibility, version compatibility, security issues, performance issues etc.

The main difference in client/server and web applications is accessing the resources. Communication between client and server maintains state, whereas in case of web application connection is stateless (http) and hence cookies are used in web applications but not in client server. There are always issues of security and compatibility for the web applications available on Internet for all users.

In next few chapters we will see how to test web applications for functionality, security and performance.

*****

Sample Test Cases for Testing Web Application Cookies

Let's first focus on what are cookies and how they work. It would be easy for you to understand the test cases for testing cookies once you have the clear understanding of functionality and storage of cookies.

What is a Cookie?

Cookie is a small set of information stored in a text file on user's hard drive by the web server. This information is later used by the web browser to retrieve information. Generally the cookie contains personalized user data or information that is used to communicate between different web pages.

Why websites use cookies?

Cookies are used to identify various web users of a particular website and track user's navigation trends throughout the website pages. The communication between web browser and web server is stateless (http protocol).

E.g. if you are accessing domain http://www.example.com/page1.html then web browser will simply query to example.com web server for page1.html. Next time when you access http://www.example.com/page2.html then new request is sent to example.com web server for sending page2.html page. Web server wouldn't usually know anything about to whom the previous page1.html was served.

What if you want to store the previous history of user communication with the web server? For this somewhere you need to maintain user's state and interaction between web browser and web server. This is where web cookies come into picture. Cookies serve the purpose of maintaining user interactions with web server.

How does cookie work?

The HTTP protocol used to exchange information files on the web is used to maintain the cookies. There are two types of HTTP protocol - Stateless HTTP and stateful HTTP protocol. Stateless HTTP protocol does not keep any record of previously accessed web page history. While stateful HTTP protocols keep some history of previous web browser and web server interactions and this protocol is used by cookies to maintain the user interactions.

Whenever user visits a site or page that is using cookie, a small code inside that HTML page (generally a call to some language script to write the cookie; JavaScript, PHP, Perl etc.) writes a text file on user's machine called as cookie.

Here is one example of the code that is used to write a web cookie and can be placed inside any HTML page:

Set-Cookie: NAME=VALUE; expires=DATE; path=PATH; domain=DOMAIN_NAME;

When user visits the same page or domain later, this cookie is used to identify the second visit of the same user on that domain. Usually cookie expiration time is set while writing it. This time is decided by the application that is going to use the cookie.

Generally two types of cookies are written on user's machine.

1) **Session cookies:** This cookie is active till the browser that invoked the cookie is open. When we close the browser this session cookie gets deleted. Sometime duration for the session, let's say 20 minutes, can be set to expire the cookie.
2) **Persistent cookies:** These cookies are written permanently on user's machine and lasts for a much longer duration; usually, months or even years unless deleted.

Where are cookies stored on my computer?

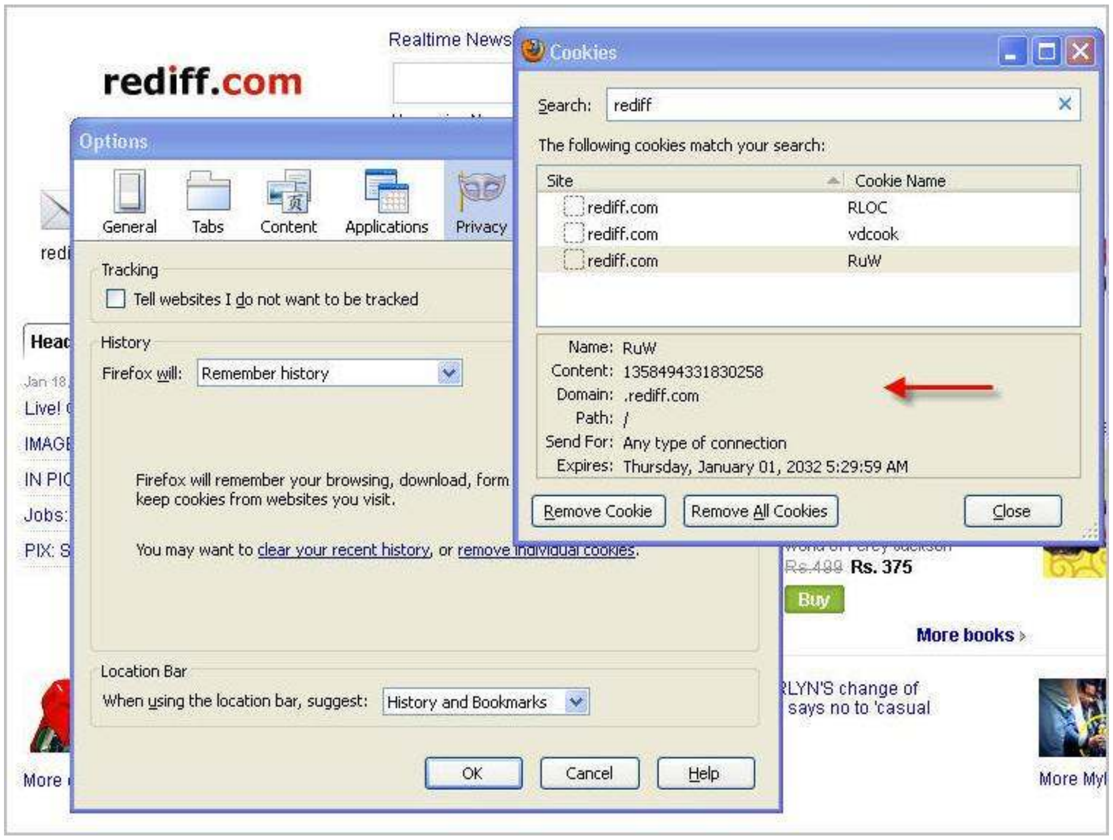
When cookie is written on user's machine by any web application, it gets saved in a text file on user's hard-drive. The path where the cookies get stored depends on the browser used to access the web page. Different browsers store cookie in different paths. E.g. in IE cookies are stored at location "C:\Documents and Settings\Default User\Cookies"
Here the "Default User" is current logged in user like "Administrator" or user name like "Vijay" etc.

The cookie path can be easily found by navigating through the browser options. In Mozilla Firefox browser you can see the cookies in browser options itself. Open the Mozilla browser and navigate to Tools->Options->Privacy and then "Show cookies" button (or "remove individual cookies" link).

How are cookies stored?

Let's consider example of a cookie written by rediff.com website on Mozilla Firefox browser:

When you open the rediff.com page or login to your Rediffmail email account, a cookie will be written on your hard-drive. To view it simply click on "Show cookies" button or "remove individual cookies" link mentioned in last paragraphs. Click or search for Rediff.com site under this list. You can see different cookies written by Rediff.com domain with different names.



Tracking
Tell websites I do not want to be tracked

History
Firefox will: Remember history

Firefox will remember your browsing, download, form
keep cookies from websites you visit.

You may want to clear your recent history, or remove individual cookies.

Location Bar
When using the location bar, suggest: History and Bookmarks

Search: rediff

The following cookies match your search:

Site	Cookie Name
rediff.com	RLOC
rediff.com	vdcook
rediff.com	RuW

Name: RuW
Content: 1358494331830258
Domain: .rediff.com
Path: /
Send For: Any type of connection
Expires: Thursday, January 01, 2032 5:29:59 AM

Site: Rediff.com Cookie name: RMID
Name: RMID (Name of the cookie)
Content: 1d11c8ec44bf49e0… (encrypted content)
Domain: .rediff.com
Path: / (Any path after the domain name)
Send for: Any type of connection
Expires: Thursday, December 31, 2020 11:59:59 PM or at end of session

Applications where cookies can be used:

1) To implement shopping cart:
Cookies are used for maintaining online ordering system. Cookies remember what user wants to buy. What if user adds some products in their shopping cart and if due to some reason user don't want to buy those products at this time and closes the browser window? When next time the same user visits the purchase page he/she can see all the products added in shopping cart during his/her last visit.

2) Personalized sites:
When user visits certain pages they are asked which pages they don't want to see or visit. User selections are stored in cookies and maintained till the user is online.

3) User tracking:
To track number of unique visitors online at that particular time.

4) Marketing:
Some companies use cookies to show advertisements to users. These advertisements are controlled with the help of cookies. When and which advertisement should be shown, what is user interest, which keywords user is searching on a site -- this is all maintained using cookies.

5) User sessions:
Cookies can track user sessions to particular domain using user ID and password.

Drawbacks of cookies:

1) Though writing Cookie is a great way to maintain user interaction, if user has set browser options to warn before writing any cookie or disabled the cookies completely then sites containing cookie will be completely disabled, thus resulting in loss of site traffic and revenue.

2) Too many Cookies:
If web application is writing too many cookies on every page visit and if user has turned on the option to warn before writing cookie, this could turn users away from your site.

3) Security issues:
Sometimes user's personal information is stored in cookies and if someone is able to hack the cookie information then the hacker can get access to other personal information of the user. Even corrupted cookies can be read by other domains resulting in security threats.

4) Sensitive information:
Some sites store sensitive user information in cookies, which may be against privacy laws.

Sample test cases for testing web application cookies:

The first obvious test case is to test if your application is writing cookies properly on hard-disk or not. You can use the Cookie Tester application to check whether cookies are accepted or rejected on your machine.

Test cases:

1) Due to privacy issues personal or sensitive data should not be stored in the cookie. Check design documents to see what information is stored in cookies.

2) If you have no option than saving sensitive data in cookie, make sure it is stored in encrypted format.

3) Check if there is no overuse of cookies. It will annoy users if browser is prompting for cookies more often and this could result in loss of site traffic and eventually loss of business.

4) Disable the cookies from your browser settings and test the application. Most of the site functionality may not work in this case. Check if appropriate messages are displayed to user like "For smooth functioning of this site make sure that cookies are enabled on your browser". There should not be any page crash due to disabling the cookies on browser (note - to perform this test clear all cookies and restart the browse).

5) Accepts/reject some cookies: If web application is writing 10 cookies then randomly accept or reject some. To test this you can set browser options to prompt whenever cookie is being written to hard-disk. Then you can either accept or reject cookies when promoted. After this, try to access important website functionality to check page crash or data loss.

6) Delete cookies: Allow site to write the cookies and then clear all cookies from the browser. Check site behavior after deleting the cookies.

7) Corrupt the cookies: Corrupting cookie is easy. You know where cookies are stored. Manually edit the cookie in notepad and change the parameter values e.g. change the cookie content, cookie name or expiry date and then check website functionality. Sometime if cookie is corrupted other domain can access the cookie data. This is an issue as the cookies written by one domain say rediff.com can't be accessed by another domain say yahoo.com.

8) Check the functionality of overwriting cookies (if applicable) - check the functionality where one page is allowed to delete the cookie set by another page under same domain. This is a common case in case of pixel/action/conversion tracking. Conversion tacking cookie is deleted when conversion happens. This is done to avoid logging multiple conversions for same user. Check if visiting website conversion page deletes the conversion tacking cookie properly and no more invalid conversions are logged for the same user.

9) Test cookies on multiple browsers: Checking cookies on different browsers is the most important case as you need to confirm if application is writing and reading cookies correctly on different browser and website functionality is working as expected. Perform this test on different browser versions like IE, Firefox, Chrome, Safari, Netscape, and Opera.

10) If cookies are used to maintain user logins verify that changing the user id (or any other similar parameter) in browser address bar is giving proper access denied warning message.

These are just few sample test scenarios for testing web application cookies. You can write multiple test cases from these scenarios.

*****

How to Test Web Applications?

Web application testing checklist:

1) Functionality Testing
2) Usability testing
3) Interface testing
4) Compatibility testing
5) Performance testing
6) Security testing

1) Functionality Testing:

Test for – all links under a website, database connections, forms used for submitting or getting information from user, and cookies.

Check all the links:

- Test all outgoing links for a domain under test.
- Test all internal links.
- Test navigation links on the same page.
- Test links used to send emails to admin or other users.
- Check if there are any orphan pages.
- Check for broken links in all above-mentioned links.

Test forms in all pages:
Input/output forms are integral part of any website. How to test web forms?

- Check validations implemented on each field.
- Check for the default values of fields.
- Check forms with incorrect inputs values.
- Check for options to create, delete, view or modify the forms.

Form fields should be validated against correct and incorrect values. Incorrect values often include negative tests. E.g. email id field validation, credit card type and CVV field validations, amount field validations etc.

Cookie testing:
Test all important test cases discussed in previous chapter.

Validate HTML/CSS:
If you are optimizing your website for search engines then you should validate website HTML and

CSS. Validate website for HTML syntax errors, Meta tags and also check if it is crawlable to different search engines.

Database testing:
Check for data integrity and errors while performing DB operations like edit, delete, modify web forms. Check if all the database queries are executed correctly, data is retrieved and updated correctly. You can also test database for load conditions. We will address this in detail in next performance testing chapter.

2) Usability Testing:

Usability means the ease at which an average application user can use the website to achieve specific goals.

This test includes:
Check for ease of use and navigation:

Website should be easy to use and clear instructions should be provided to user whenever necessary. Check if provided instructions satisfy the purpose. Home menu should be provided on each page and it should be consistent.

Content checking:
Content should be logical and easy to understand. Check for spelling errors. Dark colors are annoying and should not be used excessively. Check if website and content developed industry standards are followed. There are commonly accepted standards for using website colors, fonts, frames etc. Content should be meaningful. Images should be placed properly with proper sizes and alignments. Other optional resources for user's help should be validated if present e.g. search option, sitemap, help files etc. should be present and functioning as expected. Html sitemap should list all website links with proper hierarchical view of navigation.

3) Interface Testing:
The main interfaces are:

- Web server and application server interface
- Application server and Database server interface.

Check if all the interactions between these servers are executed properly and errors are handled properly. If database or web server returns any error message for any query by application server then application server should catch and display these error messages appropriately to users. Check what happens if user interrupts any transaction in-between. Check what happens if connection to web server is reset in between.

4) Compatibility Testing:
Website compatibility is very important for the overall testing aspect. Below compatibility tests should be executed for any web application:

- Browser compatibility
- Operating system compatibility
- Mobile browsing
- Printing options

Browser compatibility:
I have experienced this as most influential part of website testing. Some applications are very dependent on browsers. Different browsers have different configurations and settings that your web page should be compatible with. Website coding should be cross-browser platform compatible. If you are using java scripts or AJAX calls for UI functionality, perform security checks or validations and then give more stress on browser compatibility testing of your web application. Check compatibility of web application on different browser versions.

OS compatibility:
Some functionality in your web application may not be compatible with all operating systems. All new technologies used in web development like graphics designs and interface calls like different APIs may not be available in all Operating Systems. To check OS compatibility, test web application on different operating systems like Windows, UNIX, MAC, Linux, and Solaris with different OS flavors.

Mobile browsing:
With a steady stream of new mobile devices and mobile browsers it's equally important to test web application on mobile browsers. Using your site analytics you can easily identify most used mobile devices and browsers for your website. Test web application on these browsers for appearance and mobile browser compatibility tests. There are device specific browsers and compatibility tools to perform these tests.

Printing options:
If website is having page-printing options then make sure fonts, page alignment, page graphics etc. are printed properly. Web pages should fit to paper size or as per the size mentioned in printing option.

5) Performance testing:
Below are some performance testing test cases:

1. Check if page load time is within acceptable range
2. Check page load on slow connections
3. Check response time for any action under light, normal, moderate and heavy load conditions
4. Check performance of database stored procedures and triggers

5. Check database query execution time
6. Check for load testing of application
7. Check for stress testing of application
8. Check CPU and memory usage under peak load condition

More on:

- Web Load Testing
- Web Stress Testing

Load testing – This is the test to ensure if application sustain real-life load condition. Load testing is conducted to determine how many users can be handled without slowing the user experience. Load testing also measures the response time when number of users are increased gradually. In heavy load condition website should handle peak requests, large input data and connection to DB without any performance issues.

Stress testing – In this type of performance testing web server behavior is tested by increasing load to the point of breaking the application. This allows to determine the peak load where application crash and how to failover the application.

Various load testing tools are used for performance testing of web application.

6) Security Testing:

Security testing of e-commerce websites is very important. Website security testing activities includes:

- Testing website for unauthorized access
- Testing if sensitive data is encrypted before transmitting
- Testing if web application is protected from hackers

Following are some test cases for testing website security:

1. Check for SQL injection attacks.
2. Secure pages should use HTTPS protocol.
3. Page crash should not reveal application or server info. Error page should be displayed for this.
4. Escape special characters in input.
5. Error messages should not reveal any sensitive information.
6. All credentials should be transferred over an encrypted channel.
7. Test password security and password policy enforcement.
8. Check application logout functionality.
9. Check for Brute Force Attacks.

10. Cookie information should be stored in encrypted format only.
11. Check session cookie duration and session termination after timeout or logout.
12. Session tokens should be transmitted over secured channel.
13. Password should not be stored in cookies.
14. Test for Denial of Service attacks.
15. Test unauthorized application access by manipulating variable values in browser address bar.
16. Test uploaded files so that exe files are not uploaded and executed on server.
17. Sensitive fields like passwords and credit card information should not have auto complete enabled.
18. File upload functionality should use file type restrictions and also anti-virus for scanning uploaded files.
19. Check if directory listing is prohibited.
20. Password and other sensitive fields should be masked while typing.
21. Check if forgot password functionality is secured with features like temporary password expiry after specified hours and security question is asked before changing or requesting new password.
22. Verify CAPTCHA functionality.
23. Check if important events are logged in log files.
24. Check if access privileges are implemented correctly.

In next chapter we'll discuss more on various website security testing techniques.

Tips to setup test environment for web application testing:

It's good practice to ask for deployment process in release notes prepared by development team. Release notes document should include the functionality developed and process to deploy the code/build on test servers. This document should be complete enough to be referred while deployment the code on production. You can prepare a template for this release document so that the things expected to be covered will be included in it.

Here are few broad steps to deploy website code on test servers:

- Deploying web servers on test machines (Apache, IIS etc.)
- Deploying DB servers (MSSQL/MySQL/Oracle/Sybase etc.)
- Creating test database and tables (from the DB script provided in the deployment build)
- Create test data for the application. Test data can be imported from the production server by masking sensitive information.
- Deploy code, websites, and security certificate and test the deployment with test data.

Note that this test environment setup may vary based on testing types like load testing, functionality testing, load balancing test etc.

*****

Web and Desktop Application Security Testing Techniques

Need of Security Testing

Software industry has achieved a solid recognition in this age. In the recent decade, however, cyber-world seems to be even more dominating and driving force which is shaping up the new forms of almost every business. Web based ERP systems used today are the best evidence that IT has revolutionized our beloved global village.

These days, websites are not only meant for publicity or marketing but these have also evolved into stronger tools to cater complete business needs. Web based Payroll systems, Shopping Malls, Banking, Stock Trade application are not only being used by organizations but are also being sold as products today.

This means that online applications have gained the trust of customers and users regarding their vital feature named as SECURITY. No doubt, the security factor is of primary value for desktop applications too. However, when we talk about web, importance of security increases exponentially. If an online system cannot protect the transaction data, no one will ever think of using it. Security is neither a word in search of its definition yet, nor is it a subtle concept. However, I would like to list some complements of security.

Examples of security flaws in an application:

1) A Student Management System is insecure if 'Admission' branch can edit the data of 'Exam' branch
2) An ERP system is not secure if DEO (data entry operator) can generate 'Reports'
3) An online Shopping Mall has no security if customer's Credit Card Detail is not encrypted and can be viewed by anyone administering the website.
4) A custom software possess inadequate security if an SQL query retrieves actual passwords of its users

Security Testing Definition:

Now, I present you the simplest definition of Security in my own words.

"*Security means that authorized access is granted to protected data and unauthorized access is restricted*".

So, it has two major aspects; first is protection of data and second one is access to that data. Moreover, whether the application is desktop or web based, security revolves around the two aforementioned aspects. Let us have an overview of security aspects for both desktop and web based software applications.

Desktop and Web Security Testing:
A desktop application should be secure not only regarding its access but also with respect to organization and storage of its data. Similarly, a web application demands even more security with respect to its access, along with data protection. Web developers should make the application immune to SQL Injections, brute force attacks and XSS (cross site scripting) vulnerabilities.

Similarly, if the web application facilitates remote access points then these must be secure too. Moreover, keep in mind that brute force attack is not only related to web applications, desktop software is vulnerable to this as well.

Some key terms used in security testing

Before we go further, it will be useful to be aware of a few terms that are frequently used in web application security testing:

What is "Vulnerability"?
This is a weakness in the web application. The cause of such a "weakness" can be bugs in the application, an injection (SQL/ script code) or the presence of viruses or malware.

What is "URL manipulation"?
Some web applications communicate additional information between the client (browser) and the server in the URL. Changing some information in the URL may sometimes lead to unintended behavior by the server.

What is "SQL injection"?
This is the process of inserting SQL statements through the web application user interface into some query that is then executed by the server.

What is "XSS (Cross Site Scripting)"?
When a user inserts HTML/ client-side script in the user interface of a web application and this insertion is visible to other users, it is called XSS.

What is "Spoofing"?
The creation of hoax look-alike websites or emails is called spoofing.

Security testing approach: In order to perform a useful security test of a web application, the security tester should have sufficient knowledge of the HTTP protocol. It is important to have an understanding of how the client (browser) and the server communicate using HTTP. Additionally, the tester should at least know the basics of SQL injection and XSS. Hopefully, the number of security defects present in the web application will not be high. However, being able to accurately describe the security defects with all the required details to all concerned will definitely help.

I will now explain how the features of security are implemented in software application and how these needs to be tested. My focus will be on Whats and Hows of security testing, not of security.

Security Testing Techniques:

1) Access to Application:

Whether it is a desktop application or a website, access security is implemented by 'Roles and Rights Management'. It is often done implicitly while covering functionality, E.g. in a Hospital Management System a receptionist is least concerned about the laboratory tests as her job is to just register the patients and schedule their appointments with doctors. So, all the menus, forms and screen related to lab tests will not be available to the Role of 'Receptionist'. Hence, the proper implementation of roles and rights will guarantee the security of access.

How to Test: In order to test this, thorough testing of all roles and rights should be performed. Tester should create several user accounts with different as well as multiple roles. Then he should use the application with the help of these accounts and should verify that every role has access to its own modules, screens, forms and menus only. If the tester finds any conflict, he should log a security issue so it can be fixed.

2. Password cracking:

The security testing on a web application can be started with "password cracking". In order to log in to protected areas of the application, one can either guess a username/ password or use some password cracking tools for the same. Lists of common usernames and passwords are available in many open-source password cracker tools. If web application does not enforce a complex password it may not take very long to crack the username and password.

If username or password is stored in cookies without encryption, attacker can use different methods to steal the cookie information.

3. Data Protection:

There are further three aspects of data security. First one is that a user should be able to view or utilize only the data, which he is supposed to use. This is also ensured by roles and rights E.g. a TSR (telesales representative) of a company can view the data of available stock, but cannot see how much raw material was purchased for production.

Testing of this aspect is already explained above. The second aspect of data protection is related to how that data is stored in the DB. All the sensitive data must be encrypted to make it secure. Encryption should be strong especially for sensitive data like passwords of user accounts, credit card numbers or other business critical information. Third and last aspect is extension of this second aspect. Proper security measures must be adopted when flow of sensitive or business critical data occurs. Whether this data floats between different modules of same application or is transmitted to different applications, it must be encrypted to make it safe.

How to Test Data Protection:

The tester should query the database for 'passwords' of user account, billing information of clients, other business critical and sensitive data and should verify that all such data is saved in encrypted form in the DB. Similarly, (s)he must verify that between different forms or screens, data is transmitted after proper encryption. Moreover, the tester should ensure that the encrypted data is properly decrypted at the destination. Special attention should be paid to different 'submit' actions. The tester must verify that when the information is being transmitted between client and server, it is not displayed in the address bar of web browser in understandable format. If any of these verifications fail, the application definitely has security flaws that need to be plugged immediately.

4. URL manipulation through HTTP GET methods:

The tester should check if the application passes important information in the query string. This happens when the application uses the HTTP GET method to pass information between the client and the server. The information is passed via different parameters in the query string. The tester can modify a parameter value in the query string to check if the server accepts it.

Via HTTP GET request user information is passed to server for authentication or fetching data. Attacker can manipulate every input variable passed from this GET request to server in order to get the required information or to corrupt the data. In such conditions, any unusual behavior by application or web server is the doorway for the attacker to get into the application.

5. Brute-Force Attack:

Brute Force Attack is mostly done by some software tools. Using a valid user ID, the software attempts to guess the associated password by trying to login again and again. A simple example of security against such attack is account suspension for a short period of time as all the mailing applications like 'Yahoo' and 'Hotmail' do. If, a specific number of consecutive attempts (mostly 3) fail to login successfully, then that account is blocked for some time (30 minutes to 24 hours).

How to test Brute-Force Attack:

The tester must verify that some mechanism of account suspension is available and is working accurately. (S)He must attempt to login with invalid user IDs and Passwords alternatively to make sure that software application blocks the accounts that continuously attempt login with invalid information. If the application is doing so, it is secure against brute-force attack. Otherwise, this security vulnerability must be reported by the tester.

Above security aspects except URL manipulation should be taken into account for both web and desktop applications while, the following points are related with web based applications only.

6. SQL Injection and XSS (cross site scripting):

If user input data is crafted in SQL queries to query the database, attacker can inject SQL statements or part of SQL statements as user inputs to extract vital information from database. Sometimes, even if attacker is not successful to crash the application, from the SQL query error shown on browser, attacker can get the information they are looking for. Special characters from user inputs should be handled/escaped properly in such cases.

Conceptually speaking, the theme of both of these hacking attempts is similar, so these are discussed together. In this approach, malicious script is used by the hackers in order to manipulate a website. There are several ways to provide security blanket against such attempts. For all input fields of the website, field lengths should be defined small enough to restrict input of any script E.g. Last Name should have field length 30 instead of 255. There may be some input fields where large data input is necessary. For such fields proper validation of input should be performed prior to saving that data in the application. Moreover, in such fields any HTML tags or script tag input must be prohibited. In order to provoke XSS attacks, the application should discard script redirects from unknown or untrusted applications.

How to test SQL Injection and XSS:

The tester must ensure that maximum lengths of all input fields are defined and implemented. (S)He should also ensure that the defined length of input fields does not accommodate any script input as well as tag input. Both these can be easily tested E.g. if 20 is the maximum length specified for 'Name' field; and input string "<p>thequickbrownfoxjumpsoverthelazydog" can verify both these constraints. It should also be verified by the tester that application does not support anonymous access methods. In case any of these vulnerabilities exists, the application is in danger.

Cross Site Scripting (XSS):

The tester should additionally check the web application for XSS (Cross site scripting). Any HTML e.g. <HTML> or any script e.g. <SCRIPT> tag should not be accepted by the application. If it is, the application can be prone to an attack by Cross Site Scripting.

Attacker can use this method to execute malicious script or URL on victim's browser. Using cross-site scripting, attacker can use scripts like JavaScript to steal user cookies and information stored in the cookies.

Many web applications get user information and pass it to some variables from different pages.

E.g.: http://www.examplesite.com/index.php?userid=123&query=xyz

Attacker can easily pass some malicious input or <script> as a '&query' parameter which can explore important user/server data on browser.

7. Service Access Points (Sealed and Secure Open)

Today, businesses depend and collaborate with each other. Same holds good for applications; especially websites. In such case, both the collaborators should define and publish some access points for each other. So far the scenario seems quite simple and straightforward but, for some web based product like stock trading, things are not so simple and easy. When there is large number of target audience, the access points should be open enough to facilitate all users, accommodating enough to fulfill all users' requests and secure enough to cope with any security-trial.

How to Test Service Access Points:

Let me explain it with the example of a stock trading web application; an investor (who wants to purchase the shares) should have access to current and historical data of stock prices. User should be given the facility to download this historical data. This demands that application should be open enough. By accommodating and secure, I mean that application should facilitate investors to trade freely (under the legislative regulations). They may purchase or sale 24/7 and the data of transactions must be immune to any hacking attack. Moreover, a large number of users will be interacting with application simultaneously. So the application should provide enough number of access points to entertain all the users.

In some cases, these access points can be sealed for unwanted applications or people. This depends upon the business domain of application and its users, e.g. a custom web based Office Management System may recognize its users on the basis of IP Addresses and denies to establish a connection with all other systems (applications) that do not lie in the range of valid IPs for that application.

Testers must ensure that all the inter-network and intra-network access to the application is from trusted applications, machines (IPs) and users. In order to verify that an open access point is secure enough, testers must try to access it from different machines having both trusted and untrusted IP addresses. Different sort of real-time transactions should be tried in a bulk to have a good confidence of application's performance. By doing so, the capacity of access points of the application will also be observed more accurately.

The testers must ensure that the application entertains all the communication requests from trusted IPs and applications only while all the other request are rejected. Similarly, if the application has some open access point, then tester should ensure that it allows (if required) uploading of data by users in a secure way. By this secure way I mean, the file size limit, file type restriction and scanning of uploaded file for viruses or other security threats. This is all how a tester can verify the security of an application with respect to its access points.

Additionally, a security test should be avoided on a production system. The purpose of the security test is to discover the vulnerabilities of the desktop or web application so that the developers can then remove these vulnerabilities from the application and make the application and data safe from unauthorized actions.

****

Chapter 6

Test Team and Project Management!

Importance of Documentation in Software Testing

People do not talk much about software testing documentation. The general opinion about testing documentation is that, anyone who has free time can do the documentation.

Here is one story shared by one of our readers on importance of documentation: “We had delivered a project (with an unknown issue in that) to one of our clients (one of those difficult clients). And they found that issue at client side, which was really bad situation for us. As usual all blame was on QAs. The issue was something related to compatibility of one website. When it came to us, we were having proof that we didn’t get any such requirement to check compatibility of the website also. That was the lesson for us and we realized importance of documentation and from that day we started to work on creating documents like requirement analysis summary, requirement to test case mapping, testing deliverables, test plan, test cases, sanity testing checklist, bug reports etc.”

I’m not saying that testers should stick to requirements without having to worry about other test cases. Certainly there are some obvious test cases which may not be described in SRS documents. A clear understanding of roles and responsibilities to everyone can definitely help to achieve desired results.

“***Ink is better than the best memory***” – Chinese proverb

Software Testing Documentation: What’s that?

We all read various articles on testing technologies and methods, but how many of us have seen articles on documentation? No doubt there are very few of them out there. Is it that documents are not important? No, but it’s because we have not yet realized the importance of documentation.

But if we observe closely it’s the fact that the projects with all the documents have high level of maturity. Most companies do not give even a little importance to the documentation as much they give to software development process. If we search on web then we can find various templates on how to create various types of documents. But how many of them are really used by organizations or individuals?

Fact is that, careful documentation can save an organization’s time, efforts and money. While going for any type of certification, this is precisely why there is lot of stress given on documentation. It’s because it shows importance of client and processes to individual and organization. Unless you are able to produce document that is comfortable to user no matter how good your product is, no one is going to accept it.

In my workplace we own one product, which was having a bit confusing functionality. When I started working on that I asked for some help documents to Manager and I got the following answer “No, we don’t have any documents”. Then I made an issue of that, because as a QA I knew no one can understand how to use the product without documents or training. And if user is not satisfied, how we are going to make money out of that product?

“***Lack of documentation is becoming a problem for acceptance***” – Wietse Venema

Even same thing is applicable for User manuals. Take an example of Microsoft. They launch every product with proper documents. Even for Office 2010 we have such documents, which are very explanatory and easy to understand for any user. That’s one of the reasons that all their products are successful.

In small companies, we always hear things like “project X was rejected in proposal or kickoff phase”. It’s just because proposal documentation often lacks concise and expressive language and thus fails to show the capability of the organization. It’s not that small companies can’t deliver good quality projects but it’s their inability to express their capability.

I personally feel Quality is the only department that can make it possible. We are the only department, which can argue on this and can provide successful future to our organizations.

Let's organize all discussion in few points in quality perspective:
- Clarify quality objective and methods
- Ensure clarity about tasks and consistency of performance
- Ensure internal co-ordination in client's work
- Provide feedback for preventive actions
- Provide feedback for your planning cycle
- Create objective evidence of your quality management system's performance

There are hundreds of documents used in software development and testing life cycle. Here I am listing few important software testing documents that we need to use/maintain regularly:

1) Test plan
2) Test design and Test case specification
3) Test Strategy
4) Test summary reports
5) Weekly Status Report
6) User Documents/ manuals
7) User Acceptance Report
8) Risk Assessment
9) Test Log
10) Bug reports
11) Test data
12) Test analysis

Also Software testers regularly need refer following documents:
1) Software requirement specifications
2) Functional documents

Software Testing Documents always play an important role in Project development/testing phase. So always keep things documented whenever possible. Don't rely on verbal communication.

Documentation will not only save you but also help organization in long run, thus saving thousands of dollars on training and more importantly on fixing issues caused due to lack of development and testing documents. Don't document just or the sake of avoiding finger pointing at you, but the habit of documentation will also bring a systematic approach in your testing process, leaving behind an ad-hoc testing.

*****

Software Testing Weekly Status Report Template

Writing effective status report is as important as the actual work. Weekly report is an effective way to track major project issues, accomplishments, pending tasks and milestones. By using these reports you can prepare next week's actionable items based on the task priorities.

Use below simple template to write weekly status report

Status report prepared By:
Project name:
Date:
Status:

A) Issues:
Issues holding the QA team from delivering on schedule:

(You can mark these issues in red color. These issues may require management help in resolving)

Issue description:
Possible solution:
Issue resolution date:

Issues that management should be aware of:

(These are the issues that do not hold the QA team from delivering on time but management should be aware of them. Mark these issues in yellow color. To report these you can same template as above)

Project accomplishments:
(You can mark these in green color)
Accomplishment:
Accomplishment date:

B) Next week's Priorities:
(Actionable items for the next week. List them in two categories)

1) Pending deliverables:

(*Mark them in blue color: These are previous week's deliverables which should be released as soon as possible in this week)*
Work update:
Scheduled date:
Reason for extending:

2) New tasks:
(List next week's new tasks here. You can use black color for this)
Scheduled Task:
Date of release:

C) Defect status:

Active defects:
(List all active defects with details like defect reporter name, module name, severity, priority, assigned to etc.)

Closed Defects:
(List all closed defects with details like defect reporter name, module name, severity, priority, assigned to etc.)

D) Test cases:
(*List total number of test cases written, test cases passed, failed and pending to execute)*

This template should give you overall idea of the status report. Don't ignore these reports even if your managers are not forcing you to write. These are most important for your work assessment in future.

*****

Testing Under Tight Deadlines

How to test when there is not enough time for thorough testing?

Sometimes it may not be possible to test the whole application within schedule. In such situations it's better to find risk areas in the project and concentrate more on them.

Here are some questions you should ask to find important functional areas in the application to minimize risk:

Find:

1) important functionality of your project
2) high-risk modules of the project
3) which functionality is most visible to the user
4) which functionality has the largest safety impact

5) which functionality has the largest financial impact on users
6) which aspects of the application are most important to the customer
7) which parts of the code are most complex, and thus most subject to errors
8) which parts of the application were developed in rush or panic mode
9) what the developers think are the highest-risk aspects of the application
10) what kinds of problems would cause the worst publicity
11) what kinds of problems would cause the most customer service complaints
12) what kinds of tests could easily cover multiple functionalities

Focus your test efforts on these areas to greatly minimize the project risks and achieve the quality goal in crunch situations.

*****

Building a Great QA Team

What do we mean by a great testing/QA team?

"A team with a star player is a good team, but a team without one is a great team." – Author unknown.

This chapter stems from experience gained while working for different teams and observation of team member's behavior under time pressure coupled with complex nature of project. This holds good for Software Testing teams which find prominence in project activities and requires right mix of people for performing these activities.

Why do some software testing teams fail and other succeed? Is there any solution for this problem? The answer is "Yes"/"No" – and depends on how the team member aligns himself towards the common goal of the team, working together with common understanding of problem at hand.

The success also depends on leadership attributes possessed by the Test Lead –"Captain of the ship".

The objective of this chapter is to help software test engineers or any person who believes in teamwork, to understand characteristics of high performance teams and how to cultivate such practices in their own teams.

Success of team in long run doesn't depend on individual who is considered the "STAR" but does depend on all who form clusters of stars that make a great team.

Characteristics of Great Software Testing Team

Ask yourself the following question: ***Does your new team member know the reason he has been selected for the team?***

New members of the team are often puzzled about their presence in team. Although you may argue that he/she need not know the purpose and just work on task assigned to him/her. This is an assumption made by many higher management people. By clearing the roles and responsibilities helps individuals to understand the project in bigger context. That includes the relevance of his/her job, skills of individuals that could be contributed towards the projects, sharing common team goal, which was defined earlier. This does bring great commitment towards the work and hence contributes towards its quality.

Ownership:

When project complexity increases in terms of tasks and team size, it would not be possible to keep track of individual's tasks by a single leader. Hence the solution to this would be assigning ownership to individuals. However this virtual leadership often acts as an impediment rather than solution if not considered appropriately. Mere appointment of individual as owner without considering whether he/she could manage their team would not bring desired result.

Individuals acting as owners should have a mindset that matches the leader's mindset and should have the pride on their part to act as future leaders. These are people who could make difference by carrying along with them their team members. By showing indifferent attitudes towards their team will disintegrate the team.

The task of owners is not merely restricted to assigning task to team members but to understand the importance of tasks at hand, situation at much broader perspective and to bring common level of understanding among their team members. Supporting their team member at the time of difficulty of handling task, providing word of encouragement, correcting their mistakes by not acting as lead but as a peer, acting up on ideas or taking advice for appropriate situation from experienced members would certainly benefit towards the shared goal of the team. Collaboration and a solid sense of interdependency in a team will defuse blaming behavior and stimulate opportunities for learning and improvement.

Knowledge of seasoned players in the team

The term seasoned players indicates the person who has spent considerable amount of time in the same project or similar kind of work. They are resources who have vast knowledge about the project. By channeling their knowledge in proper way, the entire team could be benefited. These individual should show an act of diligence towards each other's work rather than arrogance. It is commonly said "*past success breeds arrogance*". They are higher performers, whose absence could be felt in a team but it should be not the sole criteria as there are equal chance for others who has similar caliber to act at this position.

Motivation – Key Factor

Motivation is not all about giving speech when members of team are assembled but rather every effort should be made to tailor these speeches to address each individual. This means each team member has unique qualities and unique working style. This task is rather complex than said for the Test Lead since it will need extra effort on the leader's part to sense the team member's feeling; not only the task assigned to members but also on project as whole.

Positive attitude of lead will energies the team – This is quoted from experience working for one of the great test teams. If as the leader, you complain about long working hours or insist the team members to work at schedule, which is impossible to meet, your team will soon reflect your attitude. He/she is true leader who in spite of unreasonable schedule, instills confidence among team members to believe in their abilities and at the same time working at the background on his part to justify his team member's effort working on unreasonable schedule. And bring an extension to these schedules to make his team member's job simple.

Recognition

Everyone likes to be recognized for his/her work. When an individual is awarded for his/her work, the responsibility of team lead is to bring individual recognition in front of others. The team lead's decision for this kind of task should be impartial. This does bring great respect for the awarded individual by members in the team. They would be acting on similar grounds and ultimately the team would benefit from their collective response.

Very often members working for the virtual leader are often not recognized due to zero visibility to the leader of team. It is the virtual leader who has to bring on table the accomplishment, contribution done by team member towards their task. This would indicate that the virtual leader has every potential to become the future leader who does take care of members of his team.

One-One basis Meeting

It is often seen that roles and responsibilities for the members are defined and assessment is done at the end of project. Agreed that it is formal process, but informal talk like One – One basis adds to this formal process as well. These informal meeting should address issues at present whom members won't feel like conveying during group meeting, future opportunities for members, identifying future leaders/owners of the team and equally acting on issues at hand after feedback from team members. Timely and appropriately delivered feedback can make the difference between a team that hides mistakes and a team that sees mistakes as opportunities.

The responsibility for poor performance is usually a function of the team structure rather than individual incompetence; yet, it is individuals who are sent to training programs for fixing such issues. If team members feel that they are pitted against one another to compete for rewards and recognition, they will tend to withhold information that might be useful to the team's greater goal.

When a team has problems, the effective team leader should focus on the team's structure before focusing on individuals.

*"**Don't tell people how to do things, tell them what to do and let them surprise you with their results.**" – George Patton*

Conclusion

There are plenty of things to be considered while building a successful team. The key words – Unity, Trust, Respect for others' opinion and acting without fear are ingredients for a great and successful test team. After reading this chapter look at your team and question yourself "Are you working in a great test team" or "Will you make every effort to build a great test team"? Then don't wait, try next second to build "Great Software Testing Team".

*"**Coming together is a beginning, Keeping together is progress, Working together is success**". – Henry Ford*

****

Bonus Chapter

Freelance Testing Opportunities to Earn Extra Money!

Freelance Software Testing Opportunities – Work from Home Jobs for the Testers

With recession still wreaking havoc in many countries and jobs getting tougher, more and more testing professionals are looking for opportunities outside of regular day jobs and freelancing is one such avenue that has gained a lot of traction, of late.

Working as freelance tester always provides you financial support and moreover it also help to keep your technical skills sharp.

Learn how you can make money from home working as a freelance software tester!

There has never been a better time to break into freelance software testing. With thousands of small businesses and fortune 500 companies looking for CrowdSourcing solutions, there is a high demand for freelance testers.

This book will show you how many amazing opportunities you have to make money working as a freelance software tester and how quickly your life can change if you are serious to work for yourself! Most importantly, you can get started as a freelance tester with very little knowledge of software testing. And if you are an expert software tester then there are endless opportunities for you.

In the beginning you could be working very hard to get your first few clients. But once you get to know the process it will be very easy for you to get more freelance testing work with your work experience and strong testing portfolio on various freelancing sites, which we are going to see shortly.

There are lots of books on learning software testing but no one will teach you how to make money working as a freelance software tester. You will notice that this book is shorter than many other testing books available online. Because I don’t believe in providing an eBook of 100 pages which has only 10 pages of useful information. You will find this book useful providing the exact actionable information needed for you to get started earning your first dollar as quickly as possible.

The only thing you need to do is start taking action right away. Test all the resources mentioned in this book and see what works best for you.

How to Maximize Your Chances of Getting Freelance Testing Work?

Here are few tips before you start working for any of the freelancing websites offering work from home job opportunity:

Note: These tips are applicable for only first 4 freelancing sites mentioned in chapter 2.

1) Initially apply for small projects with lowest possible bid. This is necessary for you to get started with at least one positive feedback. You can increase your hourly rate at any time later.

2) There are limitations on number of free bids you can place on projects. To utilize this quote effectively make sure to apply for relevant jobs where you are the best fit. Also apply for at least one relevant job per day. This will keep your profile active which will help you get steady work flow. Even considering a 10% success ratio you will have to apply for at least 10 jobs to get your first contract.

3) Look for profiles of other testers/QAs working on software testing projects. Study their profiles and see how they present their skills and apply for new projects. Prepare your detailed profile with your expertise, experience and other soft skills.

4) Application cover letter should convince how you are the best fit for their requirements and how you can complete this job efficiently. Also do not use same cover letter while applying for multiple jobs.

5) Avoid looking desperate for getting a job. Looking desperate shows that you don't have professional experience.

6) Take all tests related to software testing skills. Be well prepared for these tests and score in top 10%.

7) Communicate constantly with clients. Make sure you understand the project requirements clearly. Also contact them whenever you have queries while working on the project.

8) Be willing to provide extra services until the client is 100% satisfied.

9) Stick to deadlines. This is important for getting long term quality contract.

Where to Find Freelance Testing Jobs?

Work from Home opportunities for Software testing professionals

One of the most frequent questions I get is "where to find freelance testing opportunities?" Here are some great resources for you to earn some extra income working from home as a software tester:

Note – I've prepared this list from my own experience and partly with the help of our readers who are doing great working from home. This does not mean it will work for everyone. You need to try different opportunities and see what works best for you. To avoid scams, you need to be very careful to make sure that work from home job is legitimate before joining or providing any personal information.

oDesk.com

There are hundreds of opportunities for software testers on this freelancing site. Also chances of getting a full time long term contract are higher on oDesk compared to other freelancing sites. You need to remember the tips we provided in first chapter in order to get a job from oDesk. Getting a freelancing work on oDesk is comparatively easy but you need to be consistent with your work quality.

You can search for "software testing" work on oDesk or here is the category under which you will find almost all software testing QA opportunities.

Freelancer.com

You need to create freelancer account in order to bid on any of the projects.

Tips - Bidders with positive reviews get more projects. If you are good at writing you can find technical writing opportunities as well.

Search for testing jobs on this site at:

1. Testing QA
2. Software Testing
3. Testing

Elance.com

How it works?

Create your freelancer profile here on Elance. Search for the jobs and submit your proposal. Once you are awarded the project you can work in a shared workroom on Elance. Elance will automatically track your time for hourly based projects. Your payments are sent after the work completion.

Tips: Make sure you add all your key skills, job history and a nice photo to get noticed when clients search for freelance testers for the work.

Find testing QA opportunities on Elance here.

PeoplePerHour.com

This is one of the most popular freelance services with hundreds of new job postings each day.

How it works?

As a tester you need to build up your profile on PPH. The profiles are ranked based on the economic activity per month so that newbie get chance to rise up.

Once you have created your profile and provided all the required information you can start searching for jobs. You can even get daily notifications for your search term so that you won't miss any good testing opportunity. You can send your proposal when you find a relevant testing job.

Tips: You need to submit a competitive proposal in order to increase your chance of winning. Your proposal should mention the information like what you will deliver and by when. When you complete the work make sure to ask for reviews. Freelancers with highest and positive reviews have greater chance of winning the bid.

Once you win the bid, communicate with the client and complete the work. You will get paid on completion of the project. All payments are handled through Escrow so there are no chances of any fraud.

Where to find testing jobs on PPH:

Click here to search all current testing opportunities available with PPH.

QAonRequest.com

How does it work?

You need to register to use this service. While creating account mention all your testing skills. Once your application is accepted you will be notified when there is a new testing project with skills relevant to your profile. Tester who accepts the job and the work deadline first is selected for that project.

Services provided by QAonRequest – Functional testing, beta testing, test plan design, exploratory testing.

Apply to become a tester at QAonRequest.

uTest.com

uTest is the world's largest marketplace for software testing services.

This is by far the most popular site for finding software testing opportunities for almost all manual and automation testing skills. Using uTest you will get opportunity to work on top quality products before they are released to the market. Also there is a huge learning opportunity for testers on uTest. You get chance to view and learn from the bug reports submitted by top rated testers. It's truly amazing experience working with top companies which you won't get by working with one company on a single project as a full time tester.

How does it work?

Companies list their projects with uTest. uTest notify the testers about this work via email. It's up to you whether you want to accept the offer or not. uTest pays base fee per project. This base pay is also displayed on your account dashboard before accepting or declining the offer. Apart from the base pay you will also get fixed amount for exceptional and valuable defects found. E.g. $10 for exceptional bug and $5 for very valuable bug. Also the pay structure differs based on the testers ranking on uTest. High ranking testers get more incentives compared to new testers.

Overall, this is a good opportunity for experienced testing professional.

Though uTest pays a handsome amount of money for quality bugs filled, they are very choosy when it comes to selecting good testers. As a selection process you will have to work on an unpaid test cycle on their sandbox environment. During this test cycle you need to log 4 defects for the test application. You are rated based on the quality of these defects. To get selected on their paid team you need to score in top 30% testers for a given test cycle.

Tips – In selection process, be careful while submitting your 4 defects for the test cycle. Make sure you follow all their instructions and log only exceptional or high quality defects.

Cons – Though this is a huge opportunity for testers, you need to invest a lot of time initially. Working on unpaid or contest projects suck your valuable time without getting any significant pay.

QAInfoTech.com

How it works?

The testers need to register at CrowdSource testing community at QA InfoTech. Based on your profile, QA InfoTech admin team will shortlist projects that match to your skills. You have the option to accept or reject the offer.

You will be paid for every deliverable that you submit including valid bugs reported, test cases written or any test artifacts that have been assigned to you. Remuneration depends on project factors such as time, complexity of testing requirements, the devices you use etc. All these details are provided to the testers before they accept the offer.

Payments are made using PayPal.

Tips: The testers are rated on some criteria like number of valid bugs reported, number of projects you have worked upon etc. Experienced testers working with QA InfoTech for longer period will have higher ratings which make them eligible for more critical projects that have a higher remuneration.

Click here to be a part of QA InfoTech cloud sourced testers.

Passbrains.com

Create your tester profile with all profile details, test skills, and test environment. It works very similar to uTest. Once you create account you will be notified for relevant projects available for testing. The pay details ($ amount per defect) are provided on project dashboard. Once you accept the project you need to strictly follow the test deadline for completing the test. At the end of the test cycle you need to submit the test cases and defect reports.

Follow this link to create your tester profile on Passbrains.

Userfeel.com

This is a usability testing service where companies list their website for usability tests. The testers use website and report usability issues and provide feedback about the site navigation. You must be good at expressing your thoughts into the microphone as you need to explain what problems you faced while using the site, answer some questions along with your feedback about website usability.

How it works?

You need to take one sample test. This decides how often you will get usability test requests. The amount earned is not huge but can be a good addition to your monthly income. You get $10 per test.

Read their FAQ and apply to become a tester.

Similar usability testing sites:

WhatUsersDo.com

UserTesting.com

99tests.com

How it works?

Customers create their account with 99tests and submit product requirements to test. A testing contest is announced and during this period, interested testers can join the contest and test the product.

All bugs logged by the testers are submitted to client for validation. For every validated bug the testers get reputation points. At the end of the contest, typically a week, top three testers are announced.

The prize money is sent to these top testers via PayPal payment option.

Click here to create your tester account with 99tests.

Pay4Bugs.com

How it works?

Create your tester account with Pay4Bugs. In your account area you will see recommended assignments to test. You can select these recommended assignments or even pick from the list of available assignments. Read the assignment instructions and start testing. Log the bugs you find while testing. Make sure you check the existing bugs before logging a new bug report.

Each assignment has a different price paid for bugs. You can view the price paid per bug while selecting the assignment. You will be paid for all approved bugs. Payments are made through PayPal.

Click here to become a Pay4Bugs tester and join the bug hunt.

Mob4Hire.com

This is one of the oldest mobile application testing services. Developers can set a flat fee for each tester to make per test, or request testers bid on the job. Customers can pay for Mob4Hire's Managed Services team that offers testing/development expertise and project management.

Testing requirements are posted on mob4hire and testers who meet the criteria are invited to bid on the project. The winning bidder gets the application URL to test on the specified devices.

Payment model - On average, Mobsters (mobile application testers) make between $20 and $50 an hour for their time, expertise, and effort.

Create the tester account here.

Donanza.com

This is a freelance jobs aggregator website which notifies you for all jobs posted on the Internet for your selected categories or keywords. Follow this site if you want to get notifications for all freelance testing jobs from almost all popular freelancing sites including oDesk, Elance and Freelancer. But this is just a job aggregator site meaning you will have to visit the original job posting site for the job application.

Tips – Create account and select job category as software testing/QA. This will show you all freelancing jobs from this category on your account dashboard and you will be also notified by email. You don’t need to search testing jobs on different freelancing sites once you start using this service.

Check out some latest freelancing jobs for testing here.

Similar freelance testing job aggregator site:

- WhyDoWork

Beta Testing Opportunities

What is Beta Testing?

The simple definition of beta testing – testing carried out by real users in real environment.

Though companies do rigorous in-house quality assurance from dedicated test teams, it's practically impossible to test application for each and every combination of the test environment. Beta releases make it easier to test application on thousands of test machines and fix the issues before releasing the application publicly. The selection of beta test groups can be done based on company needs. Company can either invite few users to test the preview version of the application or they can release it openly to try by every user.

Many companies pay the testers for testing their beta applications. You can find beta testing job openings on job portals like Indeed.com, careers.org and simplyhired.com. There are many part time beta testing opportunities available on these job portals.

E.g. Beta Tester Jobs, Beta Tester.

Some companies also look for volunteer beta testers. But these beta testing opportunities can land you your next full time job from the company offering these opportunities. In fact, I know one example where my friend was very enthusiastic testing beta antivirus software. One antivirus company was so impressed with his work that they offered him a full time job.

How to get started as a beta tester

Once your application as a beta tester is accepted by a company, follow these steps:
- Download and read the software requirement specifications, known defects and modules to test
- Download and install the beta software
- Start testing
- Prepare the bug report for the issues found in the application
- Also note down your suggestions/feedback about the application to improve the user experience
- Submit the bug report and feedback to the company.

Game Testing Opportunities

There are many game testing opportunities for serious game enthusiasts.

Game testing companies pay on hourly basis for testing their new game releases. You can search game testing jobs on job portals like indeed.com and simplyhired.com

Game Testing Industry Introduction:

The video game testing industry is set to become the largest industry. In spite of the recession, there was no dearth in the sales of the game titles, although the game console sales were hit and the game testing companies had to revamp their strategies.

Gaming had its ups and downs over the years but it continues to grow leaps and bounds. Facebook application games are really path breaking with budding developers experimenting their knowledge. Episodic games are the new thing. Games for the iPhone are the new frontier.

So, no one in the game industry knows where games will be even two or three years from now. The only thing they know is that everything is changing and that the games that are released in a few years will be different from what we have now.

Following Jobs are made available by Gaming Industry:

1) Video game programming Jobs (designing video games)
2) Video game testing jobs

Designing Video games requires skilled and experienced video game designers. Testing video games is equally challenging as game tester needs to have a solid writing skill, very good communication skill and habit to keep attention for details.

Video game testers play critical role in game development industry. As video game programmers spend years deigning video games and video game tester needs to make sure it's ready for release in very short time span.

What is a Typical Game Testing Process?

Computer games take from one to three years to develop (depending on scale). Testing begins late in the development process, sometimes from halfway to 75% into development (it starts so late because, until then, there is little to play or test).

Once the testers get a version, they begin playing the game. Testers must carefully note any errors they uncover. These may range from bugs to art issues to logic errors. Some bugs are easy to document but many are hard to describe and may take several steps to describe so a developer can replicate or find the bug. On a large-scale game with numerous testers, a tester must first determine whether the bug has already been reported before they can log the bugs themselves. Once a bug has been reported as fixed, the tester has to go back and verify that the fix works – and occasionally return to verify that is has not reappeared.

Game Testing Strategy:

Evaluation of game rules:
Game rules adequately explain operation of all components of the game including features, free games etc. Game functions as defined by rules.

UI, Functional, Performance and Compatibility test:
Verify Games outcome and data are correctly shown when games are played. Verify Game Functionality such as Game Progress, game outcomes, handling of incomplete and re-started games, multi player games.

Verification the Integration points:
Check if game win determination aligns with game rules.

Reviewing gaming procedures:
Procedures will be reviewed by System management, player account management, tournaments and promotions.

Infrastructure and security review:
Require to verify all equipment and network implementation. Secure and reliable operation e.g. time synchronization, OS reliability and security.

How to Test Games?

This process is almost similar to product or web application testing. Here is the typical game testing process:

Identification: First analyze and identify the game rules and behavior

Functional Testing: Ensure game works as intended. This also includes integration testing with third party tools used if any.

OS and Browser compatibility: Most critical game testing part is to ensure game works on required Operating systems. For online games check functionality on all intended browsers.

Performance testing: This becomes critical for online games if gaming site handles betting on game. Game testers must verify if Game Testing site smoothly handles customer load.

Multi player testing: For multi player games you need to verify the game functionality to handle all players and functionality with fair distribution of game resources to all players.

Reporting: Bug reporting to developers. Bug evidence need to produced and submitted through bug reporting system.

Analysis: Developers hold the responsibility to fix the bugs.

Verification: After the fix, bug need to be verified by the testers to confirm that it shouldn't reappear.

Game Testing Tips:

1) Understand Random Number Generator evaluation (RNG): This is very important to add unpredictability in game. In most games this RNG system is used to map game outcomes.
2) First identify the "game algorithm" from Source code to identify issues in game application.
3) Verify the source code for appropriate use of random numbers and error handling. (Only if you know the source code)
4) Validate and evaluate the game predefined rules.
5) Verify consistency of game rules.
6) Make sure offensive content or material is never displayed.
7) Regularly Check Game history and system event logs.
8) Make sure Games outcome are displayed for a reasonable time.
9) Irrespective of Single/Multi player games we need to validate bandwidth and client software.
10) Verify Minimum/maximum limits of bets, deposits and other critical game symbols.
11) Verify correct game and system operation after game fail over and recovery.
12) Always verify all reports for data accuracy. Verify reports for date, time, number of wins, money etc.
13) Test System requirements. This is very important in game testing. Verify all the infrastructure and security requirements, Game equipments, network and game synchronization with OS.
14) Make sure sufficient information is always available to users to protect game players.

Game Testing Jobs:

Gaming field is getting much better day by day and Game Career as a Game designer or tester is very bright. There are many game testing professionals making decent amount of money as a video game testers, working from home. Present Internet generation bringing massive innovations and scope to grow. IT and Non-IT people are willing to spend their free time to play online and video games. "Game testing from home" is now a new trend to earn money. We can clearly see that it's getting into our daily activities.

If anyone of you is trying hard getting into gaming industry then you need to have interest and passion that drives you to success. Due to addition of vast and complex new games, Game QA is no longer less technical than general software QA. Game testing was widely considered as a "stepping stone" position but now it's becoming full-time job opportunities for experienced testers.

If you have passion for games and good understanding of testing methodologies, becoming a successful game tester is not difficult for you!

I've listed some job portals where game testing job openings are posted regularly. Read the job requirements carefully and apply with a good cover letter and resume with game testing experience.

E.g.:

- Simplyhired.com Game Tester Jobs
- Careerbuilder.com Game Tester Jobs
- Indeed.com Game Tester Jobs
- Indeed.com Video game Tester Jobs
- Naukri.com Game tester Job (India)

Tips –You can also search for more opportunities by identifying game design companies e.g. Electronic Arts (EA) (complete list here) and apply for their testing jobs.

One More Great Income Resource for the Testers

Create a blog/website on a topic of your passion!

Creating and monetizing a blog or website around your passion is also a proven way to support your income. E.g. If you are passionate about cooking you can start a recipe blog. This is just an example. There are endless opportunities based on individual's interest and expertise.

You can start your website on any topic with these simple 3 steps:

1) Choosing a topic for your website/blog

2) Searching for available domain name for your website

3) Purchasing a hosting plan and installing Wordpress CMS on your site.

All these steps take no more than 30 minutes to complete if you are a tech savvy person.

You may not earn a living but definitely earn some extra money to pay the bills. If you need any help to get started please contact me and I'll guide you more on that separately!

Our Recommendations

We've taken care to include almost all TOP RATED freelance testing/Crowd sourcing services in this book. If you think any good service is missing from this list please contact us and we'll consider it for inclusion.

Below are our few recommended services you can try first:

1) uTest.com

2) oDesk.com

3) Freelancer.com

4) Elance.com

5) PeoplePerHour.com

6) QAonRequest.com

7) QAInfoTech.com

Bottom Line: There are many freelancing opportunities for the testers and they are now choosing this path to support their earnings. Some testers are so successful on oDesk, Freelancer and Elance that they have quit there day job and working full time from home on these freelancing opportunities. It takes hard work and patience to see some real money. Many testers quit before seeing the real benefits of these opportunities. If you follow good work practices it's not hard for you to move to a better paying freelancing job.

****

About the Author

Author is a software tester by passion and profession who resides in Pune, India with his wife and a son. He spent over 9 years of his life testing various projects in different MNCs and also helping thousands of testing professionals around the world through his blog.

If you are interested in checking latest happenings in software testing please visit author's blog

Tutorials - http://SoftwareTestingHelp.com
Online Training - http://SoftwareTestingHelp.org

Best Regards,
Vijay Shinde
Founder and Author - SoftwareTestingHelp