testbytes

Making Quality a Habit

Methodology and Practices of Testbytes 2019

Content

The methodologies, tools and process we follow have been devised as a result of years of hard work. Before digging deep, let's go through the process that we follow at first

1. How do we make sure that Test management is effective

- **Initiation** - The first and foremost thing to following the initiation of a project in the testing phase. During this phase, a project manager will be assigned who will be responsible to analyse and review. It will also help in identifying the procedure and process of the data. Apart from this, it helps in having a deep understanding in terms of project objective and communication between the development team.

- **Plan** –The next phase of the development cycle includes the major part of the testing. In this part, if a project planning is not done appropriately then it can be doom to your software or application. This phase has a number of subdivision that much is taken care of in order to gain cent results.

- **Execute** –Once the whole planning process is setup, you need to be careful while executing it properly. A single mistake in execution can cost you a lot. You need to stick with the plan in order to come up with important details. If you feel that it might require a few changed then do it in the plan module.

- **Monitor and Control** –Another of the important phase will be monitoring the whole cycle. It is essential for the management team to keep a track whether the work is done as per the plan or not. Also, the performance of the software is monitored and controlled every time to ensure that you are not making any bigger mistake. It can be due to the communication gap among development and testing team. But this must be resolved on your end only before you handover the whole project.

- **Close** –Then comes the final stage. Once, the whole project is set out then you need to get the documentation done. This phase will be handing over the whole report to your boss and closing it. Then, it will be released to a client or in the market.

2. How do we make sure that nothing goes wrong?

When the project testing is started to execute then there are errors and risks that will pop out of an application.

It becomes easy to identify where a module was wrong or what must be altered. In order to actually be effective during the whole process of software testing, it is essential to follow the basic tricks.

Distribution and allotment of tasks

This part will be part of the planning stage where you need to divide the whole workforce as per the size of an application.

It will depend on various factors including the size of the team, schedule of the members, attitude for project and skill set.

The essential points to share with the team are:

- It is essential to have a chat with the whole team to discuss the objectives. It helps in getting the front-end picture of every single individual involved in the task. Apart from this, it is essential to be clear on what member can be involved in automated work while who must handle the manual task. Giving them a choice will help since they can be available where they are best at.

- List all the essential details of a member as per their experience and skills together to easily distribute the work among them. However, you need to make sure that junior member and interns are also getting a chance under some supervision.

- Once you have a rough sketch of the whole work allocation, you need to rotate it around among member so that they can give in their input. If they feel that they can work better while switching work with others then take their opinions under considerations. But focus more on logic and motive behind this all.

- The reporting for daily work completion. You need to settle on a single thing beforehand. You simply can't check in software for few members while checking e-mails for others. Hence, come up with a mutual decision for the mode of reporting so that none of them is unsatisfied. This way they can communicate the whole progress or issue with you on daily basis.

- It is also essential to know what all the tools are being used by your team so that you are prepared if a need arise.

This will ensure that your whole team is actually working as per the conditions that are given to them.

Resource Management Tips

Resource management is a wider subject then software testing. There is simply so much that the project manager needs to look out to ensure accurate work.

- Software testers work under some serious deadlines that can be difficult to manage. This becomes even more difficult when they have to manage excessive work with limited staff. If at any point of time, a tester feels that timeline can be hampered then get it to touch with your boss immediately so that he/she can work something out.

- Never close up any communication line. Just because your manager has given you a task and you have time doesn't mean that he won't listen to you. Especially when it comes to testing cases, it is better to have a discussion than to give some worst results.

- It is common to have queries, technical issue or defect in the whole application. However, it can be resolved if one hasa direct impact on an application. In such a case, it is better to be on talking terms with every one of your development team. It will be easy to resolve major technical issues.

- It is essential to set up meeting daily or on a particular interval of time. This will help in ensuring that work is under progress and if there are few issues then it can be dealt with.

- When it comes to testers schedule then you can't be sure about it. It will be haywire since emergency can pop up at any time and any place. In such cases, it is better to actually have no expectation that they will stretch the time. Everyone has a personal life and hence you need to make sure that the task is done at the office only with accuracy.

These are the major point that can help in the proper arrangement of your resources for testing team.

Tools and tracking

As mentioned above, communication and reporting are two of the major part of the testing phase. This will actually prevent any raised eyebrow at your end.

- During the testing phase, there are going to be instabilities and issues that are tackled by an individual. In such case, the performance of the task will be effective. If you are regularly updating progress and any sort of issue then management can help a tester to get a possible solution.

- Another factor is regular use of scrum or checkpoint meetings that can be held to ensure that the project is progressing. Also, if there is any sort of dis-balancing then it can be managed at this point.

- The regular contact with management can also help in keeping the lines open. Also, it is possible that you are getting hold of an error that was missed out earlier. Hence, appreciation or might be a promotion came in the future.

3.Our Test strategy

We follow heuristic test strategy which comprises of a project environment with numerous testing techniques that help in identifying the product's quality criteria and product elements. Key factors involved in this particular testing strategy includes,

. Project environment is a set of resources and limitations related to the project that may impact the entire process of testing.

. Product elements are the elements or specific areas of a product that the testers intend to test. The aim behind this is to include all relevant and important areas that require attention.

. Quality criteria are the specific rules, ethics and sources that assists testers in identifying if a product has some issues or not.

. Test techniques are the methods used to create tests for a product. These involve analysis and study of project's environment, product elements and quality criteria.

. Perceived quality is the result attained after conducting tests on a product. This involves applying of various testing methods, which enables one to reach a conclusion about the product's quality.

Standards of Testing

- The entire test should meet the user prerequisites.

- Exhaustive testing isn't conceivable. As we require the ideal quantity of testing in view of the risk evaluation of the application.

- The entire test to be directed ought to be arranged before executing it.

- It follows 80/20 rule which expresses that 80% of defects originates from 20% of program parts.

- Start testing with little parts and extend it to broad components.

Software testers know about the different sorts of Software Testing. In this article, we have incorporated majorly all types of software testing which testers, developers, and QA reams more often use in their everyday testing life. Let's understand them!!!

Common Testing Techniques

Since heuristic is an observation-based model that evolves over time, there are a few other techniques that can be applied universally in every sphere. These include the following:

Function testing

This type of testing is used to identify the functions performed by a particular product. It involves testing each component of the program to validate its functions and sub functions.

Domain testing

To conduct this type of testing, the user decides on the type of data to be tested for different functions along with several other aspects such as typical values, invalid values, convenient values, etc.

Stress testing
This type of testing aims at testing the product's functionality under different stress situations. The conditions selected can be varied such as complex data structure, high load, long test runs and low memory conditions.

Flow testing
Conducted to check the entire flow of the program, flow testing is based on establishing connection between activities.

Scenario testing
This type of testing is done to check the product on the basis of all the possible situations and circumstances. Conducting this helps in identifying the way in which a product would respond in different situations.

Claims testing
This testing is done to verify the various claims made about the product in magazines, advertisements or any other place.

User testing
Conducting this type of testing helps in determining the ways in which a user interacts with the system. The aim behind this type of testing is to be at the users place and test the product from his/her perspective.

Risk testing
This type of testing is used to check the way in which a product responds in a particular circumstance or situation. Designing of appropriates test cases based on the issues identified is an important part of this type of testing. The best test cases can be

prepared after seeking help from past test reports, design documentation, etc.

Automatic Checking
This type of testing enables one to conduct automated testing of a product. It is important to ensure that the tool selected for automated testing enables one to partially automate test coverage, use automatic test data generators, etc.

Things we consider before conduction of heuristic testing

It is important to keep certain key factors in mind before conduction this form of testing. Some of these include:

- The purpose of the project as interpreted by the user as well as the tester
- Information needed to conduct the test should be precise and concise
- Relationship between tester and developer
- The team members who will be conducting or supporting the test
- The sequence and duration of product events

Selecting product elements
While performing a test, it is important to ensure that all the unique and important aspects of the product are taken in focus so that there is no bug that is missed. Some of the product elements that are important while conducting a test are:

- Structure of the final product
- Functionality delivered by the product
- The data used by the product and to be used while testing the product
- Interfaces that are used to access the product/system
- Identifying the ways in which product will be used
- Defining quality criteria such as reliability, usability and scalability
- Models/ approaches are essential for any testing company. Unlike most of the companies, we follow multiple testing models in accordance with the product.

4. The models we basically follow

These models ensure maximum productivity and minimal flaws when it comes to software testing. And, Again, which model to choose will solely depend on the nature of the project.

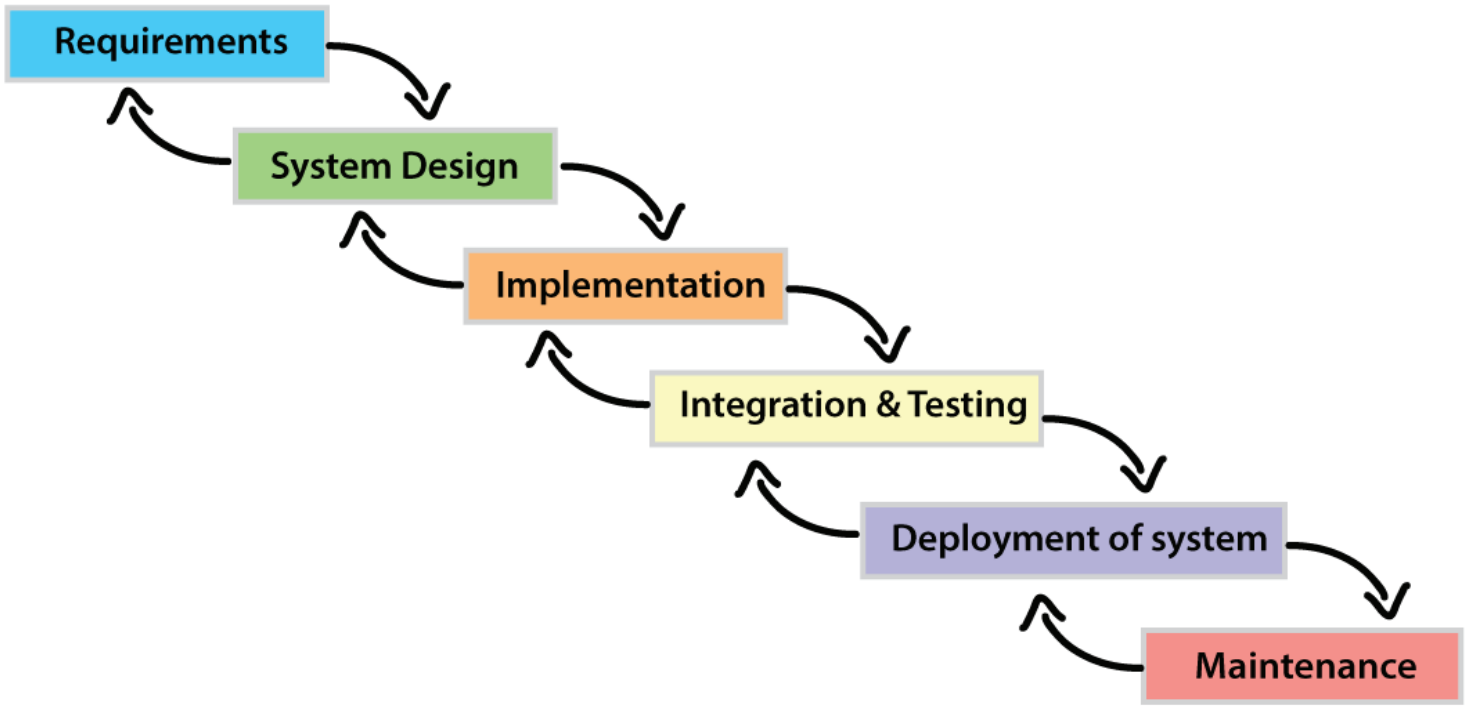
1. Waterfall Model

This is the most basic software development life cycle process which is followed broadly in the industry. Here the developers follow a sequence of processes where the processes flow progressively downwards towards the ultimate goal. It is like a waterfall where there are a number of phases.

These phases have their own unique functions and goals. There are, in fact, four phases – requirement gathering and analysis phase, software design, programmed implementation and testing, maintenance. All these four phases come one after another in the given order.

In the first phase all the possible system requirements for developing particular software are noted and analyzed. This in turn depends on the software requirement specifications which includes detailed information about expectations of the end user. Based on this a Requirement Specification.

Document is created which acts an input to the next phase, i.e. software design phase. What needs to be emphasized here is that once you move into the next phase it won't be possible to update the requirements. So you must be very thorough and careful about the end-user requirements.



Have a look at the description of sequential phases in waterfall model

Requirement gathering: All possible requirements that can be gathered from the client will be done in this phase. Gathered requirements will also be documents for reference.

System design: Requirements of software will be studied well and a system for testing will be designed accordingly. By designing the architecture hardware and system requirements can be found out for any project

Implementation: with proper input from the system design, The system is divided into small units and will be tested

Integration and designing: After testing each unit the entire system as a whole will be tested

Deployment of system: After the completion of functional and non-functional testing the software will be deployed in customer environment and will be closely monitored.

Maintenance: There is chance that issue might surface in a customer environment to ensure maximum user experience new version of the software will be released with required patches.

Advantages

- Easy to implement and maintain.
- The initial phase of rigorous scrutiny of requirements and systems helps in saving time later in the developmental phase

- The requirement of resources is minimal and testing is done after completion of each phase.

Disadvantages

- It is not possible to alter or update requirements
- You cannot make changes once you are into the next phase.
- Cannot start the next phase until the previous phase is completed

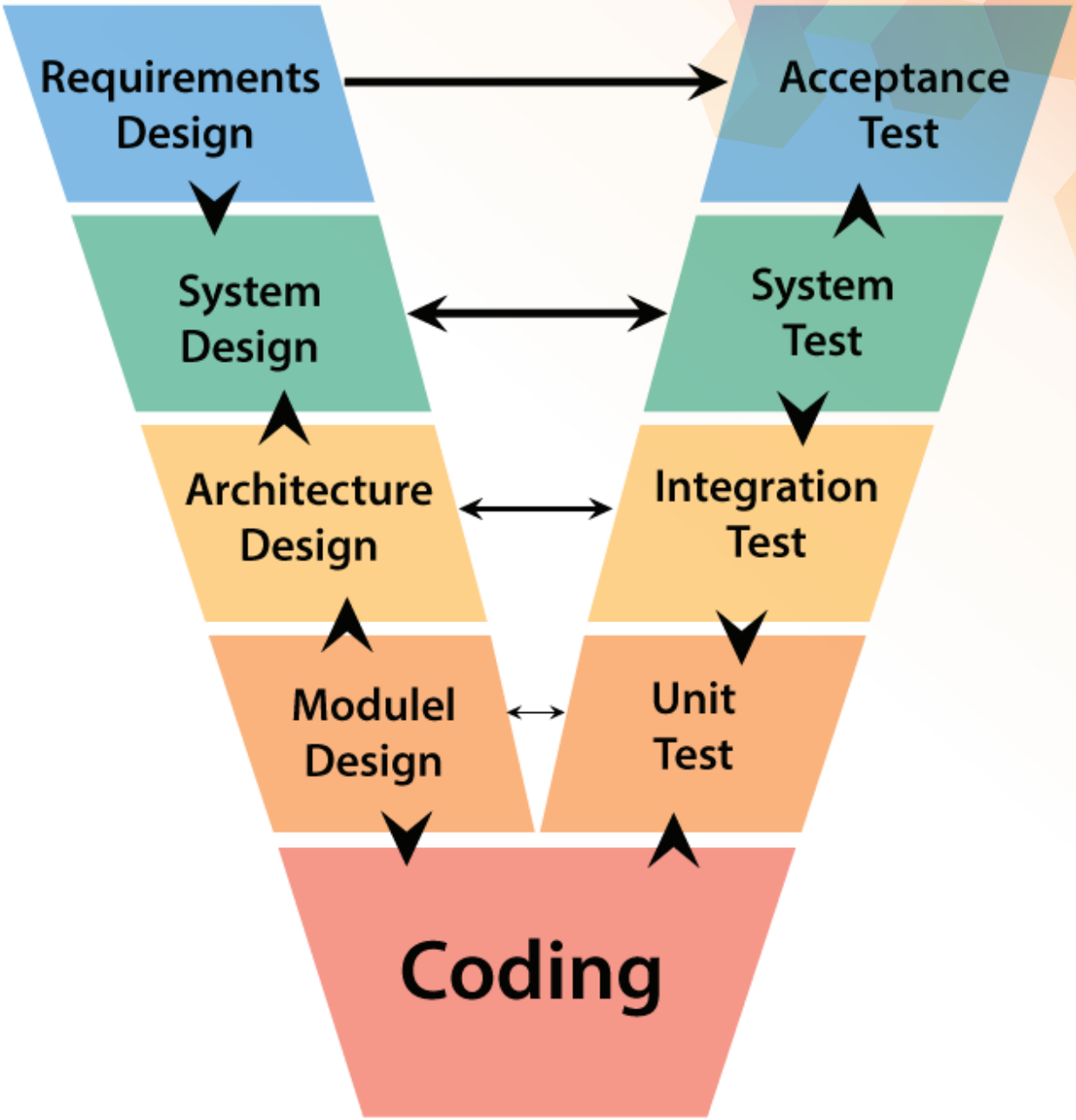
When to use?

- Can be used when the requirements are crystal clear
- Migration projects in which requirements remains the same only there is a change in platform
- Projects where sponsor will do the testing activities

V Model

This model is widely recognized as superior to waterfall model. Here the development and test execution activities are carried on side by side in the downhill and uphill shape. In this model, testing starts at the unit level and spreads towards integration of the entire system.

So, SDLC is divided into five phases – unit testing, integration testing, regression testing, system testing and acceptance testing.


Requirements Design
Acceptance Test
System Design
System Test
Architecture Design
Integration Test
Modulel Design
Unit Test
Coding

Phase s of V-Model

The phases of V-model includes verification, coding and validation that are further divided into different stages.

1) Verification phase:
The verification phase of V-model includes business requirement analysis, system design, architectural design and module design.

Business requirement analysis is the stage of having a detailed communication with the customer so that it gets easier to understand and comprehend his/her exact requirements.
It is beneficial to complete acceptance test design planning at this stage.

System design stage involves understanding and detailing out the entire hardware and communication setup for the product being developed. System test design can also be planned at this stage.

Architectural design stage involves understanding the technical and financial feasibility of the product before it is actually developed. The focus is to understand the data transfer that will take place between internal and external modules.

Module design stage focuses on designing a detailed plan for the internal modules of the system. Also known as low-level design (LLD), it is important to ensure that the design is compatible with other modules in system architecture and other external systems.

2) Coding Phase:

During this phase, the actual coding of the system modules is taken up. On the basis of system and architectural requirements of the program, the best suitable programming language is selected using which the coding is done at par with the coding guidelines and standards. The code is then reviewed and optimized to ensure the delivery of best performing product.

3) Validation phase:

During this phase, the product undergoes various forms of testing.

Unit testing is conducted at an early stage so that the bugs are eliminated at the starting stages of product development. Integration testing is done to check whether there is a valid and proper communication within the internal modules of the system.

System testing enables the testing of the entire system and to ensure if the internal modules communicate effectively with the external systems.

Acceptance testing is done to test a product in user's environment and to check if it's compatible with the other systems available in the environment.

Advantages

Easy to use the model since testing activities like planning and test designing are done before coding

Saves time and enhances chances of success.

Defects are mostly found at an early stage and downward flow of defects is generally avoided

Disadvantages

It is a rigid model

Early prototypes of the product are not available since the software is developed during the implementation phase

If there are changes in the midway, then the test document needs to be updated

Agile model

In this SDLC model requirements and solutions evolve through collaboration between various cross functional teams. This is known as an iterative and incremental model.

Agile Test Plan

The Agile Test plan includes all the types of testing performed in a particular iteration. It includes the following:

- It defines the test scope, sprint goals, test and extent to which the test is to be performed
- It specifies the testing tools to be used, data and configurations for the test and the environment in which the test will be performed.
- It schedules the test tasks and frequency of tests, i.e. how many time will they be performed.
- It defines the testing methods and techniques
- It also determines the expertise and training required to carry out tests.
- It sets the priority of the tests according to the customer's point of view.

Agile Testing Quadrants

The Agile testing quadrant divides the entire testing process into 4 parts. This makes the Agile testing process easy to understand.

Among the 4 quadrants, the left 2 tell the testers which code to write and the right 2 quadrants help them understand the code better with the help of feedback to the left quadrants.

Quadrant 1

This quadrant focuses on the quality of code. It includes test cases and test components which are implemented by the testers. These test cases are for automation testing to help to improve the code.

Quadrant 2

This quadrant contains business driven test cases which are also implemented by the testing team.
The main focus of this quadrant is on the customer requirements. It improves the business outcomes of the software being created.

It contains the following tests:

- Testing scenarios which may occur and workflow
- Testing the user experience
- Performing pair testing

Quadrant 3

This phase provides feedback for the previous two phases. There are many iterations of reviews and feedbacks carried out in this quadrant which helps to strengthen the code.

Usability tests, exploratory tests, user acceptance tests, and collaborative tests are performed in this quadrant.

Quadrant 4

The non-functional requirements of the code, such as performance, security, scalability, etc. are taken care of in this quadrant.

Testing for stress and performance are carried out in this phase. Security and infrastructure test, data migration and load testing.

This quadrant makes sure that the code satisfies all the non-functional requirements.

Advantages

Ensure customer satisfaction with rapid and continuous development of deliverables.

It is a flexible model as customers, developers and testers continuously interact with each other

Working software can be developed quickly and product can be adapted to changing requirements regularly

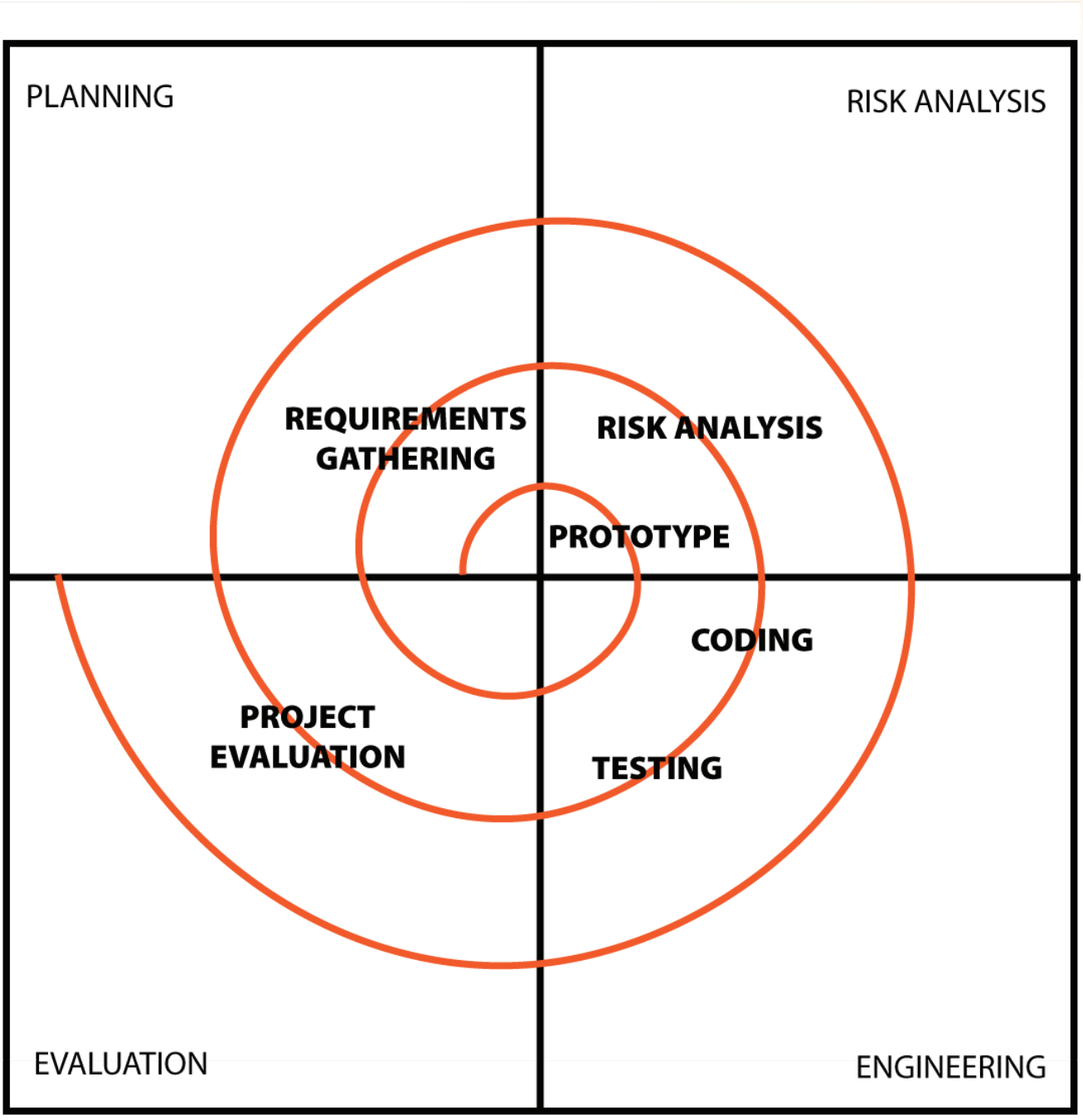
Disadvantages

In large and complex software development cases it becomes difficult to assess the effort required at the beginning of the cycle

Due to continuous interaction with the customer, the project can go off track if the customer is not clear about the goals

Spiral model

It is more like the Agile model, but with more emphasis on risk analysis. It has four phases – planning, risk analysis, engineering and evaluation. Here gathering of requirements and risk assessment is done at the base level and every upper spiral builds on it.



Requirement Gathering: Requirements are gathered from customers and objectives will be defined, identified, elaborated and analysed

Identify and Resolve risk: All the solution required for the test will be analysed and risk associated along with it will be calculated and will be resolved with effective strategies.
At the end a prototype will be developed for solution which will be used for current and future issues

Coding and testing phase: Testcase preparation and test execution happens in this phase

Evaluation Phase: Evaluation test by us and the customers will happen in this phase

Advantages

- Risk avoidance chance is enhanced due to the importance on risk analysis.
- Its a good model for complex and large systems.
- Depending on the changed circumstances additional functionalities can be added later on
- Software is produced early in the cycle

Disadvantages

- It's a costly model and requires highly specialized expertise in risk analysis
- It does not work well in simpler projects

Rational Unified Process

RUP Model in Software Testing

The RUP methodology uses the object orientation approach in its design and the use of UML (Unified Modeling Language) notation is designed and documented to illustrate the processes in action. It uses commercially proven techniques and practices.

It is a process considered heavy and preferably applicable to large development teams and large projects, but the fact that it is extensively customizable allows it to be adapted to projects of any scale.

Specifics

For project management, the RUP model provides a disciplined solution such as the tasks and responsibilities outlined within a software development organization.

RUP is, in itself, a software product. It is modular and automated, and its entire methodology is supported by several development tools integrated and sold by IBM through its "Rational Suites."

The methods of competition in the field of software engineering include "clean rooms" (considered heavy) and agile (light) such as Extreme Programming (XP-Extreme Programming), Scrum, FDD and others.

There are certain guidelines and templates that are defined, for the staff members of a production cycle, by RUP: part of the client and an evaluation of the progress of the project by its management. It helps developers to stay focused on the project.

Management Requirements

Proper documentation is essential for any large-scale project; note that RUP describes how to document functionality, system limitations, design restrictions, and business requirements.The use cases and the scenarios are examples of dependent process artifacts, which have been considered much moreeffective in capturing functional requirements.

The Use of a Component-Based Architecture

The component-based architecture creates a system that can be easily extensible, promoting reuse and software an intuitive understanding. A component usually refers to an object in object-oriented programming.

RUP provides a systematic way to build this type of system, focusing on the production of an executable architecture in the early stages of the project, that is, before committing resources on a large scale. The Components referred to here are generally included in the infrastructures already existing in place. These infrastructures include CORBA as well as Component Object Model (COM).

The Use of Visual Software Models in the Rup Model

By abstracting the programming of your code and representing it using graphical building blocks, RUP can be an effective way to get an overview of a solution. The use of visual models can also allow individuals with less technical profile (as clients) to have a better understanding of a given problem, and thus be more involved in the project as a whole.

The UML modeling language has become an industry standard for representing projects, and is widely used by RUP!

Check Software Quality

It does not ensure software quality is the most common failure in all computer systems projects. Usually, one thinks about the quality of the software after the completion of the projects or the quality is the responsibility of a different team development team.

Management and Control Change Software

In all software projects the existence of change is inevitable. RUP defines methods to control and monitor changes. As a small change can affect applications in totally unpredictable ways, change control is essential to the success of a project.

RUP also defines the areas of work and security, which guarantees a programmer that changes in another system will not affect your system.

Phases of the RUP Methodology

So far these guidelines are general, to be adhered to go through the life of a project cycle. The phases (see figure below) indicate the emphasis given in the project at a given moment. To capture the temporal dimension of a project, RUP divides the project into four different phases:

Initiation or Design: emphasis on the scope of the system;

Preparation: emphasis on architecture;

Construction: emphasis on development;

Transition: emphasis on the application.

RUP is also based on the 4 Ps:

- **People**
- **Design**
- **Product**
- **Process**

The layers are composed of iterations. Iterations are windows of time; iterations have defined term as the phases are objective.

All phases generate artifacts. These will be used in the next phase and document the project and allows a better follow-up.

Design Phase

The design or initiation phase contains the workflows necessary for the agreement of interested parties - stakeholders - with the objectives, architecture and planning of the project. If these actors have a good knowledge, it will not be necessary to analyze. Otherwise, a more elaborate analysis is required.

In this stage, the essential requirements of the system are transformed into use cases. The objective is not to close them at all, but only those that are necessary to shape the opinion.

The step is usually short and is used to define if it is feasible to continue with the project and define the risks and the cost of the last one. A prototype can be made for the client to approve. As the RUP cites, the ideal is to perform iterations, which must be well defined in terms of their amount and objectives.

Elaboration Phase

The preparation will be for the design of the system, as a complement to the survey and / or documentation of use cases, in front of the architecture of the system, to review the business model for the project and to start the version of the user manual. One must accept: Product description (increase + integration) is s table; the project plan is reliable? The costs are eligible?

Construction Phase

In the construction phase, the physical development of the software starts, production codes, alpha tests. Beta tests were carried out at the beginning of the transition phase.

You must accept the tests, stable and test processes, and the system code are "baseline".

Transition Phase

In this phase is the delivery ("deployment") of software, which carries out the deployment and delivery plan, the monitoring and the quality of the software. Products (releases, versions) are going to be delivered, and place customer satisfaction. This stage also takes place the training of the users.

Disciplines of the RUP Methodology

The Business Modeling Discipline

Organizations are increasingly dependent on IT systems, so it is imperative that information systems engineers know how applications are integrated into the development of the organization. Companies invest in IT, which understands the competitive advantage of value added by technology.

The goal of business modeling is to first establish a better understanding and communication between business engineering and software engineering.

Understanding the business means that software engineers must understand the structure and dynamics of the target company (the client), the current problems that the organization is facing and potential methods and strategies for making amends. Another important aspect that must not be undermined is that the relevant parties such as the developers as well as the customers and also the

end-users must have clear understanding about the organization, and an important feature of this understanding is that it must be common among all the parties involved. Business modeling explains how to describe the vision of an organization in which the system will be implemented and how to use this vision as a basis to describe the processes, functions and responsibilities.

Course Requirements

This course explains how to get requests from interested parties ("interested parties") and convert them into a set of requirements that the products work within the system to be built and provide the detailed requirements for what is necessary for the system.

Analysis and Design of the Discipline ("Design")

The purpose of the analysis and design is to show how the system will be carried out. The objective is to build a system that:

Execute in a specific execution environment, tasks and functions specified in the descriptions of use cases

Satisfy all your needs

It is easy to maintain when there are no changes in the functional requirements, the results of the project in an analysis and design model optionally has an analysis model. The design model is utilized as a conceptual version of the source code, displaying only the bare minimum. This allows the user of any one inspecting to ascertain the style in which the source code has been rendered.

The design model is rendered in such a way that it contains different divisions of designs. These divisions are stored within definite subsystems. Every subsystem has a distinct interface that is precisely designed. It also contains descriptions of how the objects in these classes collaborate to carry out the design of use cases.

The Discipline Implementation

The effects of the application are:

- With reference to the layered subsystems organized for an application, the organization code is configured.

- The different classes or divisions of components are carried out. These components include

    1. **Source Files**

    2. **Executables and**

    3. **Binaries**

- Components developed as units are tested

Incorporate the results produced by the individual executors (or teams), in an executable system. The systems are achieved through the components of the application. The process aims at performing an important function, which is to define the exact procedure to be utilized, in order to re utilize components which are either; already existing or have been freshly introduced. This allows for a hassle system maintenance possibility and a substantial improvement in chances of reutilization of components.

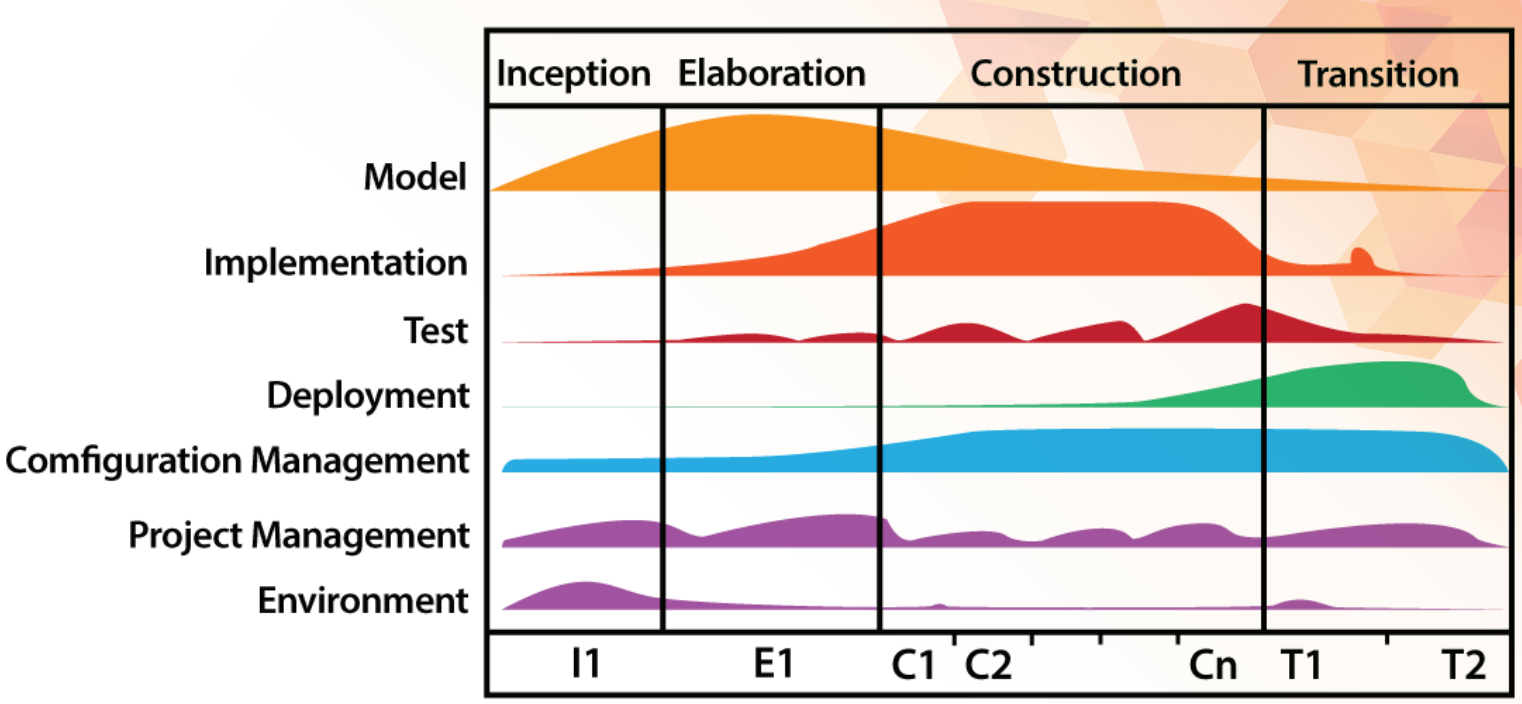
Discipline Test

The purposes of discipline testing are:

- Check the interaction between objects
- Check the correct integration of all software components
- Check that all requirements have been executed correctly
- Identify and ensure that defects are addressed before the software implementation
- Make sure that all defects are corrected, reviewed and closed

In case there are defects in the project, their correction may take up unnecessary costs due to the defects not being brought to light within due time. If the project however, is tested in its entirety, this would be beneficial as any defects which might be creeping into the projects can be identified and ascertained at the earliest. This will in turn have a massive reduction in the costs involved with the rectification of the defects. This is the iterative approach proposed by the Rational Unified process.

In order for the test to bear fruits and have the best possible outcomes, the tests need to be conducted on four parameters of quality and also there must be set standards which need to be met for the project to be considered as have passed the test.



Phases in RUP

Inception Phase – A vision document, initial business case, project plan, showcasing phases and iterations, initial risk assessment etc. happens in this pase

Elaboration Phase – 80% of the use case model will be completed in this section. Non functional supplementary requirements, executable architectural prototype, revised risk list etc will also be completed in this phase

Construction phase – The designed software is integration tested in this phase

Transition – The end product will be delivered to users their review will be recorded so that bugs can be removed.

Advantages

With an emphasis on accurate documentation this model is able to resolve risks associated with changing requirements of the client

Integration takes less time as the process goes on throughout the SDLC.

Disadvantages

The biggest disadvantage is that the team members need to be experts in their niche.

In big projects such continuous integration, it might give rise to confusions

Rapid application development

This is another incremental model like the Agile model. Here the components are developed parallelly to each other. The developments are then assembled into a product.

Advantages

The development time is reduced due to simultaneous development of components and the components can be reused

A lot of integration issues are resolved due to integration from the initial stage

Disadvantages

It requires a strong team of highly capable developers with individual efficacy in identifying business requirements

It is a module based model, so systems that can be modularized can only be developed in this model

As the cost is high, the model is not suitable for cheaper projects

5.Testing strategy and techniques we follow

Beginning of Planning

- To help figure out which system is appropriate for your project, begin by putting up front the accompanying helpful inquiries.
- Consider the technology and application included. How was the application developed? How is the user experience?
- Consider testing necessities. Does the application have an extremely complex work process?
- Do you have to run the test cases in a foreordained order?
- What are the license costs connected with every tool?
- Does the test case have to be updated always?
- What aptitudes does your staff as of now have? Is there a team that could connect to one of the systems?

How do we determine the size of the Project

The size of the project relies upon the functional size of the framework under test. The functional size indicates the quantity of functionality that is significant to the client.

A substantial undertaking that is produced over a more drawn out time period may be firmly connected to and even subject to an abundance of documentation at each stage of the venture. This may likewise be ordered by the higher administration levels, and in such a case the Waterfall or V-model may bode well.

The Agile technique depends on negligible documentation and won't be proper for such a vast venture, as it may likewise make it harder to judge endeavors required for testing.

Thus, for a smaller undertaking, that requires the stages to advance rapidly, and depends on a dynamic improvement technique, the Agile testing model would be the approach.

Budget and Time of the Project

At last how much time is accessible will dependably influence the determination of testing procedures. At the point when additional time is accessible we stand a chance to choose more strategies and when the time is seriously constrained we will be restricted to those that we know, have a decent possibility of helping us find only the most imperative errors.

In case that the deadline is near for the testing phase then automation testing is the best decision.

Test Objective

In case the test objective is basically to pick up certainty that the product will adapt to ordinary operational activities at that point use cases would be a sensible approach. When the goal is for extremely exhaustive testing then more thorough and detailed methods (incorporating structure-based methods) ought to be selected.

Interpretation of Resources

Another vital thing is to assign the right resources to the test execution practice otherwise you can't accomplish the desired test scope and results. Plan your product testing process based on the accessibility of your testing assets. This incorporates test environment, testing tools, testing devices, and human resources.

You must interpret the range of abilities of resources and their availability for the undertaking. For instance, an expert tester will take two days to finish an errand; though, an apprentice may take four days for a similar undertaking.

Also, in case that you are utilizing some new testing toolsor devices, the duration of the learning time ought to be incorporated into the evaluations.

You may have particular programming or hardware prerequisites to comply with the framework in progress. Thus, ensure that you have or you will place the required test environment when required.

Considering the Testing tools

The advantage of utilizing appropriate testing tools can't be discredited. So, the way to deal with selecting them ought to be exceptionally insightful.

Kind of Framework Utilized

The kind of framework (e.g. graphical, embedded, financial, and so on.) will impact the selection of methods. For instance, a money-related application including numerous calculations would profit by boundary value analysis.

Don't Disregard the Significance of Documentation

Documentation is an important element of manual testing, particularly with regards to hunting issues through reports. An adequate report is fundamental to helping other team members (developers and testers alike) comprehend what problems exist and how to discover them.

To write a substantial report, you require a simple title that obviously determines the issue, an arranged list of steps to reproduce the issue that is straightforward and easy to track, specifications on the priority and seriousness of the issue, and knowledge on what ought to happen if the issue is settled.

Types of Testing and When They are Required

Automated Testing

You may automate the testing of an application or software by running scripts which open up a website page, push a few buttons, input any data, and afterward check for a few results.

You could likewise automate the API testing by writing scripts that call to the API with different information and afterward check the outcomes that are returned.

More testing strategies are moving towards automated testing in light of the fact that manually going through test cases, again and again, can be blunder inclined, tedious, and expensive. It is done particularly in an agile environment where a similar arrangement of tests should be run like every two weeks or so to verify nothing has broken.

Why Manual Testing?

Manual testing is performed based on human judgment and experience. By utilizing this procedure, those zones get tested that might not have been tested or needed as through the manual testing you can test all aspects of the product.

Let's have a look at the Video Representation by Coolgrad about software testing best practices

Manual testing is just reasonable when the test cases are run just once or twice, and regular repetition isn't wanted.

Performance Testing

Performance testing is the testing to evaluate the speed and viability of the framework and to ensure it is creating results inside a predefined time as in performance necessities. It falls under the category of black box testing.

Load Testing

Load testing strategy is utilized to test an application by frequently and relentlessly expanding the load on the application until the point when it leads threshold objective.

For the most part, this product testing strategy recognizes the greatest working limit of an application and to figure out which component is causing degradation.

Estimate the Time for Load Testing

Since load testing is a different exercise out and out, the time and resources required for it are not the same as that required in the normal testing process. Subsequently, it is imperative to estimate time for manual load testing and evaluate the venture result and due date as needed to be.

Analyze to Test Parallel or Sequentially

If you've to run the test simultaneously yet on various machines then automation testing is conducted since on account of manual testing you need to conduct the test sequentially.

GUI Testing

For testing, GUI shows automation testing is executed. There are numerous devices utilized for recording user activities and afterward replay them any number of times. This is useful for looking at genuine and expected outcomes.

Acceptance Testing

Acceptance testing is known by several distinct names. Sometimes it is named user acceptance testing. Other times it's called system testing.

The basic concept of acceptance testing is that you have a few tests which test the real expectations or requirements of the client, and other tests which run against the system as a whole.

This type of testing could be used to test the usability or to test the system functionality or both.

Regression Testing

The idea behind regression testing is to ensure the product doesn't relapse in functionality.

This is critical with Agile development techniques where programming is produced incrementally and there is a steady potential that including new features could hamper present ones.

Most automated tests are regression tests.

Exploratory Testing

Manual testing is best for Exploratory testing in which testers are allocated an approximately characterized assignment to accomplish utilizing the product being tested.

One of the advantages of exploratory testing is that anybody can participate to help test since they should simply meander about the software in a free-form way.

Exploratory testing isn't irregular, yet they aren't scripted like manual tests too.

System Testing

The system testing is normally done by a team that is free of the development group to assess the property of an application.

In an application development, system testing is executed as the initial level of testing where the framework is tested all in all.

End-to-End Testing

End-to-end testing is a type of product testing to test if the flow of an application is carrying on as assumed from beginning to end.

In this testing strategy, the whole application is checked for basic functionalities like interfaces, communicating with different frameworks, network, database, and different applications.

The fundamental purpose of utilizing end-to-end testing technique is to decide the different conditions of an application not withstanding ensuring that the precise data is transmitted between different components of the system.

Unit Testing

The foundational phase of testing is unit testing. Unit testing is the act of instrumenting input and output accuracy checks for singular units of code. The measurement unit, for this situation, is independent code techniques or functions.

Unit tests are an awesome method to approve determined data functions.

Security Testing

Security testing is a class of testing which is performed to recognize vulnerabilities in a framework and related foundation, with a specific end goal to ensure client and organization information, and also the rational attribute.

There are various distinctive sub-classes to security testing, for example, penetration testing, which strives to recognize vulnerabilities that an attacker could damage from internal or external access.

Ad-hoc Testing

In this situation, there is no particular approach. It is an absolutely impromptu technique for testing where the comprehension and knowledge of the software tester is the main essential factor.

Other key factors to consider for finalizing testing techniques–

Who will Test?

The test purpose must definitely appoint duties regarding the different phases of testing to project staff. The self-governing tester facilitates a fresh view of how well the application meets the necessities. Utilizing such an individual for the component test requires a great knowledge which may not be feasible in a very iterative environment.

The engineer brings an information about the aspects of the program yet, in addition, an inclination concerning his/her own particular work.

Considerations About Adopting for Automation

In spite of the fact that you shouldn't automate the majority of your testing, you ought to constantly search for chances to grasp automation so as to spare time and make it less difficult to run some tests all the more regularly (particularly as testing turns out to be more integrated in the whole software developing process).

Make sure to adopt an automated as the first strategy while considering which tests are best off performed manually and which tests are great nominees for automation.

You must likewise keep unpredictability as a top priority. While profoundly complex test cases ought to stay manual, automating straightforward smoke tests can include critical value.

Direct Conceptualizing

It is likewise prudent to save some time for conceptualizing the real test outcomes and objects. While conceptualizing for the test objects, it is critical to retain the numbers genuine as opposed to expecting some unachievable numbers.

Conceptualizing will furthermore enable you to distinguish and cover some unexpected deferrals amid the testing stage while it encourages you to use from the greatest test coverage.

Build up Quality Assurance Process

Distinct projects can have an alternate structure of project groups, assignments, and roles. Subsequently, you may have the option to adherence to the organization quality assurance process or change it according to the undertaking needs. You may likewise need to characterize the states in the bug life cycle for your venture.

Quality assurance process incorporates the total quality cycle for the venture beginning from the comprehension of prerequisites, making of test scenarios, performing of test cases, analysis and reporting of bugs, checking the fixes, and lastly working out the smoke test of the ultimate software.

Delays and Possibilities

You can utilize a few estimation systems to give practical evaluations for testing exercises. In any case, there is as yet a chance that actual work will go astray from the assessed effort. Sometimes, it happens that development required more exertion and the release does not end up accessible for testing until the due date comes close.

So, it is prescribed to consider such postponements and also include a few possibilities in the estimations.

Changes in the Code

In the event that the changes made in the code are every much frequent then automation testing ought to be performed. In some cases, it happens in making out variations in the one module that can damage the other module.

Hence, it's essential to test the entire modules after rolling out changes to one of the modules and this can without any difficulty be tested through the automation process.

Try not to Neglect Bug Cycle

Bug cycle is a necessary element of test execution. Sometimes, it happens that the real test cycle keeps going a few more days than what was regulated before.

In this order, it is critical for the testing personals to recognize the way that the test cycle intensely relies upon the resistance of the build. In case that the site or application isn't steady or when it is loaded with defects, it will need more time to settle them, which toward the end extends the testing cycle.

Recognize the Gaps

Now and again, there're gaps between the project skill demand and the member's capabilities. Its obligation of Test Manager to recognize which abilities the individuals need in order to make a proper training plan for them.

Capability

Automation testing is more productive when there are vast quantities of system users.

6. What makes us stand apart?

1. Experience

It is one of the most important aspects that should be considered while hiring a software testing company. you should check whether the company has any experience in software testing or not. To do so, you can check or read their portfolio and by going through their past experiences. Some parameters on which you can rate the company's experience are Parameters of coding, usability, the performance delivered, design and marketing, load balancing capabilities etc.

2. Qualification

However qualification plays a vital role in selecting the software testing company, but it is not necessary that a more qualified outsourcing company will give you the better result. The things that matter is the innovative ideas, the capability of the team and of course the qualification as well. So, before hiring a company the criteria of qualification is also need to be considered. You should give your project to the company who have a team of well-qualified professionals.

3. Coding Standards & Framework:-

Before hiring a software testing company you should assure that the company and the team should have a sound knowledge of programming language and tools related to software testing. The company having the more structured framework and an organized coding system will have the better code maintainability compared to others.

4. The Extent of Service:-

It is another important point that needs to be considered before hiring a software testing company. It is very important to see the extension or the scope of the services rendered by the company as it will help you in the long run. You should assure that the company is providing you with all the steps of software development lifecycle.

5. Team Location:-

It is another important question that comes to the mind while selecting a software testing company. The question is whether you should choose an offshore company or a company in the same country?

The overseas company can provide you the high-quality service but at the same time, there can be an issue of communication barriers, language issue, cultural differences etc. So, if you are choosing an international company, make sure that they are able to beat these barriers.

6. Service Level Agreement:-

It is an agreement or a contract signed between both the parties i.e. the service provider and the client. the SLAs defines the output expected from the service provider. It is very important to define the SLAs between both the parties to ensure 100% alignment of goals settled between both the parties.

Some key points that your SLA must have are:-

- Knowledge transfer
- core business know-how
- Process compliance
- Timelines of reporting and project management
- Quality measures etc.

7. Flexibility and Scalability:-

There should be flexibility in the services provided by the software testing company or an outsourcing company. It should be decided by examining the factors such as modifiability, ease of exit, robustness, new capability etc. Outsourcing contracts demand a degree of mouldability to ensure that the timescale fluctuations are met.

8. Quality Improvement:-

It is one of the primary objective of the client to achieve a remarkable quality improvement through outsourcing a software testing company. As the agreement comes to an end, the working method and process tried to improve continually. Eventually, the target should be the overall improvement of the end product.

9. Intellectual Property Protection:-

It is one of the important aspects to be taken care while outsourcing the services. IP refers to the creation of mind like inventions, designs, artistic work, and symbol etc. used in commerce. It is one of the biggest challenges to protect the IP of business when it is outsourced.

10. Security:-

When you hire a software testing company, security is the most important aspect that needs to take care. The software must be having the information about the company that should not be disclosed to everyone. So, a business should choose the company which provides security to the software.

11. Testing Infrastructure:-

It refers to the tools and techniques that are required for software testing. Before hiring a software testing company you should make sure the service provider must have all the required infrastructure to support your software or product. The testing infrastructure includes software, hardware, operating system, backend database systems, testing tools, platforms etc.

12. Management Style:-

Management plays an important role in software testing. So before hiring a software testing company, you should make sure that the managerial style of your company is compatible with the service provider. It is important that both should have a same managerial language which will help them work together.

13. Responsibility and Accountability:-

Responsibility and accountability should go together. The software testing company you are hiring must be responsible and should be one who can take the accountability also. You would love to work with the company who is ready to take the responsibility and the accountability as well.

14. Cost of Working:-

After accessing the company on the above-mentioned parameters, you should decide the true cost of working with them. You should choose the outsourcing company which provides you the maximum ROI in terms of quality, overall value addition, and timely delivery.

15. Documentation Standards:-

Before hiring a software testing company, you should make sure that the company should have all the required documentation standard you need. Some of the documents are; test plans, scripts, test plans, test scenarios and test results etc. You should make sure that the company you are hiring should be well documented and you have easy access to the documents.

7.How do we find testers and testing allies

Often, we can't always rely on in-house testers, Testing strategies such as beta and usability testing requires a testing community at disposal. And complexity is yet another factor that drives to trust on allies.

Evaluate our Stakeholders

Before we make any moves, it is important to learn our office culture comprehensively.

Office culture is not just the rules you follow on a daily basis.

It is the people in your office, especially the stakeholders of your business.

We usually have a meeting with your project managers and other leaders who understand the requirement of your company.

Ask their point of view on hiring a team of software testers.

Their recommendations will help in creating a basic idea of what sort of software testers we should hire.

Such meetings will help us understand questions regarding manual testing, automated testing, deadlines followed generally, time management requirements, communication and other requirements.

Also, we will ask our stakeholders and managers about the preferred coding language utilized.

Evaluate our Company's Methodology

We always re-evaluate the methodologies utilized in your company. It has always given us a clear picture about what we need

Target Community Events And Testing Conferences

After evaluating our office culture, we will find the right testers in order to create a software testing team. However, there are many potential testers who don't fit your criteria for hire.

Hence, we always make sure that we avoid the clutter and reach the right software testers quickly.

One way of doing that is getting involved in community events and testing conferences. These scenarios provide a meeting ground with group defined testers.

We already know what kind of testers are coming to attend a community event or testing conference.

Hence, obtaining potential candidates from these events is a smart move to save time and efforts without compromising in any manner.

We also announce our interest in hiring a software testing team and allow the potential firms or testers reach you.

Gathering Potential Options Online

We have always promoted your requirements via social media, job portals, and other ways. Attract with competitions focused towards testing needs and office culture you desire.

Right candidates have always been available in platforms such as LinkedIn and Twitter.

Hence, we have utilized the online presence of your company to gather candidates faster without too much investment.

Ask Questions About The Procedure They Follow

After finding the right candidates to build a team, we make sure that we have understood their approach.

We will them scenarios and ask questions regarding the testing procedure, bug finding, and other associated concepts.

This way, we can understand their thinking ability and recognize problem-solving capacity, which is a much-needed trait.

Also, we will include questions associated with the office culture we have. Here we will ask them about the coding approach and knowledge, their ability to provide flexible timings and more.

And most importantly, we will ensure that the testers have understanding of manual and automated testing approach of software testers.

Test The Testers

An actual execution test is necessary before making a decision. Hence, we should invest in creating an assignment that aligns with your work culture.

Put some limitations that you face on a daily basis. For instance, we will let the testers decide the ratio of manual and automated testing.

This way, we can test the testers in terms of their approach, skills, techniques, and methodologies.

Evaluate Teamwork of the Testers

It is essential for software testers to work in coordination with each other and other programmers in your company.

The ability to communicate matters, so that, the testers and enhance the productivity of your team with seamless collaboration with programmers and designers.

After going through each and every step mentioned above, we will find the right software testers to hire.

They blend into our office culture and become an inherent part of our company. However, it is still important to allow them some time tvtto adjust to the work culture.

Contact Us

65, Brodway Suite, Newyork NY, 10006
PH: +1 (212) 744-1256

Kalas road, Vishrantwadi, Pune,
Maharashtra-411015
PH: +918113865000