Quality Assurance Standards & Processes

Version 1.06

Last Update: 09/22/2019

For Internal Use Only

Contents

Quality Assurance

Introduction

The written Quality Assurance and Quality Engineering policies and standards are pertinent to DevCare Solutions and is to be considered as a master guide for following and implementing the practice at any client engagement or as part of product development.

Purpose

The purpose of the Quality Assurance Standards and Processes is to:

- Maintain industry standard and relevant QA practices.
- Maintain conformity across QA implementations within DevCare and its clients.
- Adapt to a uniform practice to learn and contribute within the QA ecosystem at DevCare.
- Assist clients with QA implementations, frameworks and standardizations.

The formation of these QA policies is driven by many factors, with the key factor being a risk. These policies set the ground rules under which DEVCARE operates and ensures that the clients are provided with best-in-class QA services

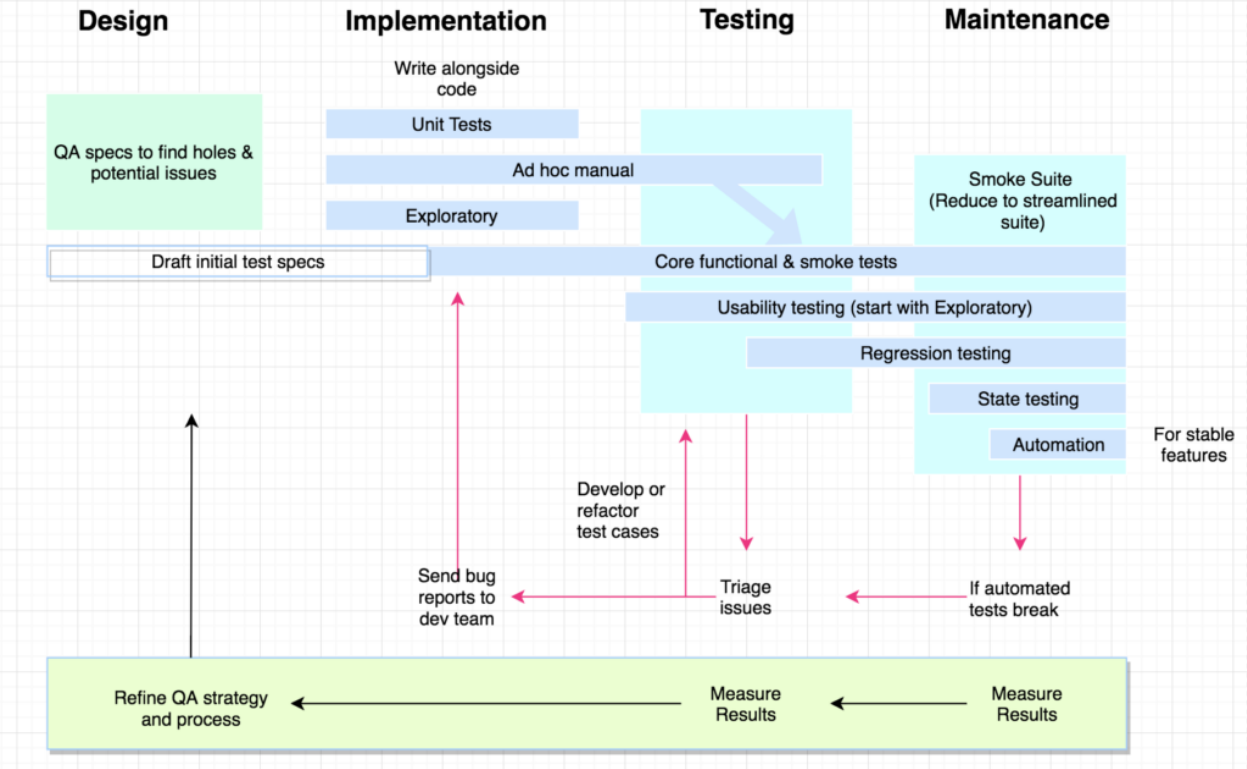
Exceptions

While every exception to a standard potentially weakens protection mechanisms for DEVCARE practices and standards, occasionally exceptions will exist. When requesting an exception, users are required to submit a business justification for deviation from the standard in question.

THIS PART OF THE PAGE IS LEFT BLANK INTENTIONALLY

Quality Assurance Process

While there is no one-size-fits-all, the below process depicts the high level of framework of QA process DevCare adheres to. It may be customized from Organization to Organization and client to client depending on the project needs:



THIS PART OF THE PAGE IS LEFT BLANK INTENTIONALLY

Characteristics of Quality Assurance

DevCare will adhere to below characteristics to define its Quality Assurance practice maturity:

Cross-functional

The quality of your product impacts every facet of the business. As a result, every team — from product and dev to marketing and sales — all have a stake in your QA process. Cross-functional interests requires cross-functional input. Good QA processes are designed to help all stakeholders have input and insight into product quality

Right-sized

More QA doesn't mean better quality, and what's the right level for a startup of 10 won't look like the right level for an enterprise of 1000. Good QA can happen at orgs of any shape and size, but it always has to fit the company's needs

Automated & Integrated

QA happens across the application's lifecycle, so it needs to weave in and out of the many platforms and tools that different teams use. QA is traditionally somewhat manual, from test execution to results triage. Good QA is well-integrated into the tools and processes that are essential to the team's workflow, and whatever can be effectively automated is automated.

Flexible & Adaptive

Needs and processes change over time; QA shouldn't have to be entirely reinvented every time the org structure changes or the company's needs change. Good QA anticipates growth and is designed for scalability, as well as flexibility. At the high-level (overall process) as well as the test case level (not over-investing in things like brittle Selenium test scripts)

Measured

You can't have good insights without data from practice. Traditional QA metrics often take a "spray and pray" approach to test writing, or follow edge cases down rabbit holes. Good QA takes a step back to think strategically about overall product quality and process performance, measures key metrics and uses that data to drive decisions about the team, product and process

Types of Quality Assurance

Manual Testing

Manual testing is the process of using the functions and features of an application as an end-user would in order to verify the software is working as required. With manual testing, a tester manually conducts tests on the software by following a set of pre-defined test cases. Let's take a close look at how this is done.

Steps for Manual Testing

Understand the Requirements

In order to successfully conduct manual tests, you first need to understand the requirements of the software. By understanding the requirements, you'll know what needs to be tested and what classifies a defect. This is a crucial part of manual testing as the main goal is to make sure the software is bug-free.

Write Test Cases

Once you understand the requirements, you can write test cases – another crucial part to manual testing. Test cases guide the tester through a sequence of steps to test functions and different scenarios within the software application. Writing good test cases is important as it makes executing the tests go smoothly and ensures good test coverage. Good test cases should also be repeatable, which allows future testers to go in and conduct the tests without having to ask additional questions.

Conduct The Tests
Once the test cases are written and the testing environment is prepared, it's time to begin testing. Follow the test cases and mark each test as "passed", "failed, or "skipped". When doing manual testing, it's important to keep notes on what happens when a test fails.

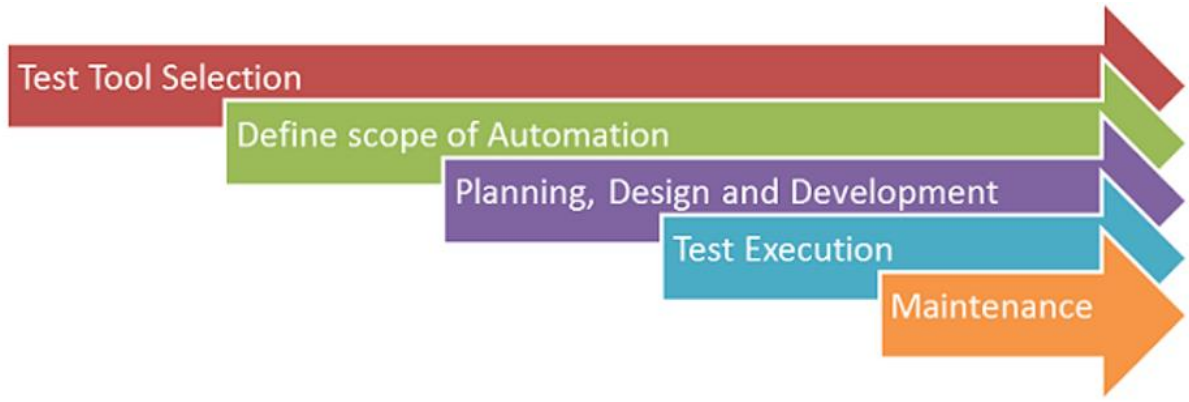
Log Good Bug Reports
In addition to actually testing, the tester is responsible for logging bugs. As you encounter bugs, you need to log information for the development team about the defect. Writing good bug reports helps you and your team. You'll save yourself time when answering questions about the bug later.

At its core, a good bug report should have a strong title, steps to replicate the bug (often the test case steps), an expected & actual result, and any relevant attachments that will help the development team understand the issue (screenshots, screen recordings, export files, etc).

Report On The Test Results
After running tests, it's good to know the results of the tests at a high level. How many tests were run? How many tests failed? How many tests were skipped?

Automation Testing

Automation testing is a Software testing technique to test and compare the actual outcome with the expected outcome. This can be achieved by writing test scripts or using any automation testing tool. Test automation is used to automate repetitive tasks and other testing tasks which are difficult to perform manually.



Steps for Automation Testing

Test tool selection

Test Tool selection largely depends on the technology the Application Under Test is built on. For instance, QTP does not support Informatica. So QTP cannot be used for testing Informatica applications. **It's a good idea to conduct a Proof of Concept of Tool on AUT.**

Define the scope of Automation

The scope of automation is the area of your Application Under Test which will be automated.
Following points help determine scope:

- The features that are important for the business
- Scenarios which have **a large amount of data**
- **Common functionalities** across applications
- Technical feasibility
- The extent to which business components are reused
- **The complexity** of test cases
- Ability to use the same test cases for cross-browser testing

Planning, Design, and Development

During this phase, you create an Automation strategy & plan, which contains the following details:

- Automation tools selected
- Framework design and its features
- In-Scope and Out-of-scope items of automation
- Automation testbed preparation
- Schedule and Timeline of scripting and execution
- Deliverables of Automation Testing

Test Execution

Automation Scripts are executed during this phase. The scripts need input test data before there are set to run. Once executed they provide detailed test reports.
Execution can be performed using the automation tool directly or through the Test Management tool which will invoke the automation tool.

Example: Quality center is the Test Management tool which in turn it will invoke QTP for execution of automation scripts. Scripts can be executed in a single machine or a group of machines. The execution can be done during the night, to save time.

Maintenance

As new functionalities are added to the System Under Test with successive cycles, Automation Scripts need to be added, reviewed and maintained for each release cycle. **Maintenance becomes necessary to improve the effectiveness of Automation Scripts.**

Framework for Automation

A framework is set of automation guidelines which help in

- Maintaining consistency of Testing
- Improves test structuring
- Minimum usage of code
- Less Maintenance of code
- Improve re-usability
- Non-Technical testers can be involved in code
- The training period of using the tool can be reduced
- Involves Data wherever appropriate

There are four types of frameworks used in automation software testing:

1. Data Driven Automation Framework
2. Keyword Driven Automation Framework
3. Modular Automation Framework
4. Hybrid Automation Framework

Automation Tool Best Practices

To get maximum ROI of automation, observe the following

- The scope of Automation needs to be determined in detail before the start of the project. This sets expectations from Automation right.
- Select the right automation tool: A tool must not be selected based on its popularity, but it's fit to the automation requirements.
- Choose an appropriate framework
- Scripting Standards- Standards have to be followed while writing the scripts for Automation. Some of them are-
    - Create uniform scripts, comments, and indentation of the code
    - Adequate Exception handling - How error is handled on system failure or unexpected behavior of the application.

    - User-defined messages should be coded or standardized for Error Logging for testers to understand.
- Measure metrics- Success of automation cannot be determined by comparing the manual effort with the automation effort but by also capturing the following metrics.
    - Percent of defects found
    - The time required for automation testing for each and every release cycle
    - Minimal Time is taken for release
    - Customer Satisfaction Index
    - Productivity improvement

The above guidelines if observed can greatly help in making your automation successful.

Benefits of Automation Testing

- 70% faster than the manual testing
- Wider test coverage of application features
- Reliable in results
- Ensure Consistency
- Saves Time and Cost
- Improves accuracy
- Human Intervention is not required while execution
- Increases Efficiency
- Better speed in executing tests
- Re-usable test scripts
- Test Frequently and thoroughly
- More cycle of execution can be achieved through automation
- Early time to market

Software Testing/s that can be automated

- Smoke Testing
- Unit Testing
- Integration Testing
- Functional Testing
- Keyword Testing
- Regression Testing
- Data Driven Testing
- Black Box Testing

How to choose a Automation Testing Tool

Selecting the right tool can be a tricky task. Following criterion will help you select the best tool for your requirement-

- Environment Support
- Ease of use
- Testing of Database
- Object identification
- Image Testing
- Error Recovery Testing
- Object Mapping
- Scripting Language Used
- Support for various types of test - including functional, test management, mobile, etc...
- Support for multiple testing frameworks
- Easy to debug the automation software scripts
- Ability to recognize objects in any environment
- Extensive test reports and results
- Minimize training cost of selected tools

Tool selection is one of biggest challenges to be tackled before going for automation. First, Identify the requirements, explore various tools and its capabilities, set the expectation from the tool and go for a Proof Of Concept.

Pros and Cons of Manual Testing vs Automated Testing

	Manual Testing	Automated Testing
Pros	• Anyone can test • Easiest way to improve quality • Great for people who may not have formal testing experience • Focused on the customer's work flow primarily	• Faster test cycles • Able to identify more defects in shorter time frame • Saves companies money after the third time the test cases are run • Is an excellent way to meet the testing needs of agile development • Easy to focus on all possible work flows • Higher product quality over manual testing
Cons	• May not identify all test cases • May not identify all defects • Lower product quality because of higher defect count	• Test scripts typically written by automation testing scripter • Requires manual test cases to be written first that are then automated • Requires automation testing platform • Not cost effective for less than 3 test cycles

QA Standards and Processes

Standard QA Process and Businesses

At some stage or other, a business enterprise feels the need to implement QA processes to standardize their operations, product delivery, and empower business solutions. However, if the QA process implementation is neglected for long or not created at the right time, it affects the overall ability of an enterprise to deliver effective business solutions. The existing process fails to detect shortcomings in offered solutions leading to slackening market reputation. Recall of products not only gives a beating to the brand image but also affects the profitability. Applying critical patches can upset customer confidence and may lead to lower marketability. Again, the bigger a project, the more is the chance of critical defect due to extensive processes and procedures involved. A standard QA process, on the other hand, assure structured mechanism that controls the process by establishing different toll gates and identify and report the defect slippage, if any.

Standard QA Process: Two Principles

A standard QA process follows two basic principles.

- Fit for purpose principle: It means the process would look into various issues to validate that the product developed is perfect for the intended purpose.
- Right first time principle: Any mistake must be identified and eliminated at the first instance. The product should not be released without being certified free from defects.

Different Models

- **Waterfall Model:**
  A sequential software development model, it views the QA process akin to a waterfall, a steady flow of the process through various phases. Requirements gathering, designing a base system, implementation to a system, verification, and maintenance constitute different phases product development process goes through. Initially, requirement review and test planning for a given product is carried out. Then, the test case designing is completed based on review of test cases performed earlier. All types of tests execution for functional, usability, cross browser, and security testing are carried out simultaneously to log all bugs in each cycle. This is continued with the resolution of bugs accessed through a proper regression testing.
- **Rapid Action Model:**
  With the objective of encouraging collaborative ambiance and facilitating dynamic gathering of requirements, this model favors rapid prototyping, which requires minimal planning. The absence of comprehensive pre-planning activities saves the time and allows the software writing process to proceed faster. Interspersing planning with the software writing also make it convenient to change development according to newly emerged requirements. This model also permits active

participation of business owners in prototyping, test case writing, and unit testing of products.

- **Agile Test Framework:**
  In an agile framework, the QA process is carried out in close cooperation with both developers and testers. In fact, the QA process plays an extensive and vital role in the development phase only. The development is carried out in a break up released cycle followed by effective software testing process. While the product development is divided into smaller chunks, the entire team takes the responsibility for maintaining the quality. The process involves just the right amount of documentation and the response is very fast and quick. Offshore QA team always plays a very vital role in agile framework. Analysis of requirements and discussion on queries related to business flow and scope through mails/scrum call add to the overall quality.
- **V-Model:**
  A hybrid form of the waterfall model, this entails phase-wise product development. A graphic representation of the entire process takes shape of V, as it bents upward following the coding phase. One phase is completed thoroughly before the development process enters the next phase. Contrary to the linear process prescribed by the waterfall model, it favors a structured testing approach. The key benefits of this process include superior quality and prevention of downward fault multiplication. The phase-wise testing, verification, and validation assure development process thoroughly checked at various stages. This contributes to improved quality and reliability and reduces the need to rework to the minimum possible.

Goals of Quality Assurance

The main objective of the QA process is to ensure robust products with improved end-user experience and greater market acceptance. Like any other product development process, software development has its share of complexities and risks. Each activity contributes to the entire process and a flaw at one stage may impact the overall quality. Again the risks are multiple, both technical and nontechnical. We may often confronted with problems, such as below-par performance, operational difficulties, browser stubbornness, maintenance problems, high cost, schedule overruns, etc. While we can overcome some of these using pragmatic ways, a number of risks demand thorough and standard QA process throughout the product development.

The QA process offers a reliable and effective way to reduce both programmatic and technical risks through multipronged strategy and efficient use of man, material, and money. It sets standards for each stage, defines tasks, monitors the endeavors to achieve them, and streamline the process with an eye on the objective. For example, the product coding standards. If there are no codes, it is likely that the product developed falls short of set usability requirements. If codes are set but the process lacks an explicit assurance that all parameters are fully met, there is the risk that coding does not meet the quality standard. Likewise, absence of error management in the workflow may enhance the risk that defects are not given adequate attention and if left unsolved, they may impact the end-user experience. An effective QA process helps overcome all the risks and improve the

quality during the workflow and at the final delivery. It assures full control over quality stan-dards and effective mechanism to clear bugs guaranteeing superior life cycle and market acceptability.

Different Types of Testing

- **Unit Testing:**
  The key objective of unit testing is to verify and validate software by examining all possible individual software components. It entails dividing an application into the smallest possible testable units and testing them separately to validate their behavior. The main advantage of this approach lies in its ability to identify maximum possible bugs.
- **Integration Testing:**
  Once all modules are tested individually, they are again tested in an integrated mode. The purpose is to ensure that distinct components of the application work in accordance with customer requirements. Test sets are developed with the express purpose of examining the interfaces between various unit components. Integration test is considered successful only when the actual result meets the expected objectives.
- **System Testing:**
  This Black-box type testing requires testing the system as a whole and takes into consideration the overall specifications, ranging from developers expectations to end user requirements. Covering all combined parts of a system, it ensures that the system must meet all functional as well as business requirements. It focuses and verifies the specifications while viewing the entire system as one unit. It validates the system that can be used for the intended purpose. The system test design is derived from the system design documents and is used in this particular phase. To make the approach more result-oriented, many specialized tests for performance, stress, function, automation, etc, are also carried out.
- **Acceptance Test:**
  It prescribes verification and validation of an application against all given user acceptance and business requirements. Though this approach is similar to system testing, where the whole system is verified, it varies considerably in its focus. The testing is always carried out by the real users in a business environment. Known as end user testing, it allows the customer to check the acceptability of the system. Customers can perform any test based on their business processes.

Business Users Involvement

All test cases should be approved by the business team. Test plans are also shared with the business team and the strategy is finalized after its approval.

- The business users are allowed a say in the bug identification process, as the application is are directly related to the requirements.

- A thread should be opened to the business user for any requirement clarification not cleared by the test team.
- Testers are required to provide a weekly status to a business team to make sure that the requirements are being met and all pending issues can be discussed. Release notes must be business users centric.
- Before signing off, the business team should be invited for final approval. One common test environment should be arranged where testing and business experts can test and build together on the same database. Any change requested must be converted to use test cases is approved by the business team.

Bug Regression will be the main concern throughout all given testing phases. All bugs that are resolved as “fixed, needs retesting” should be regressed hen testing team is notified of the new drop containing the fixes. The test lead is responsible for all activities, from tracking and reporting to development and product management at each step. Testing is suspended on the affected software module when various levels of test case bugs are discovered. Testing is also suspended if there is any important scope change leading to impact on the Critical Path. A bug report is filed by development team. After fixing the bug, development team follows the “drop criteria” to provide its latest drop for additional testing.

Conclusion

There is always a question, what to review in a proper assigned manner to ensure that a given QA standard process is followed. There are many tools and techniques that are available for use. A proper bug-free testing always requires an expert tester’s creativity, experience, and intuition combined with right techniques. Testing is more than just debugging. Testing is not only used to locate defects and correct them, but also used to verify and validate the development process and reliability measurement. A good tester must always conduct a proper gap analysis to evaluate the project results and match the original objectives. He must review all delivered documents and ensure these have been delivered with an acceptable level of quality or an acceptable substitute is in place. If still there is gap between stated and found results, the given project goals must be achieved with minimum place for errors. The necessary controls and systems in place must work properly. A proper established QA standard process assures a bug-free, well established software product within the stipulated budget.