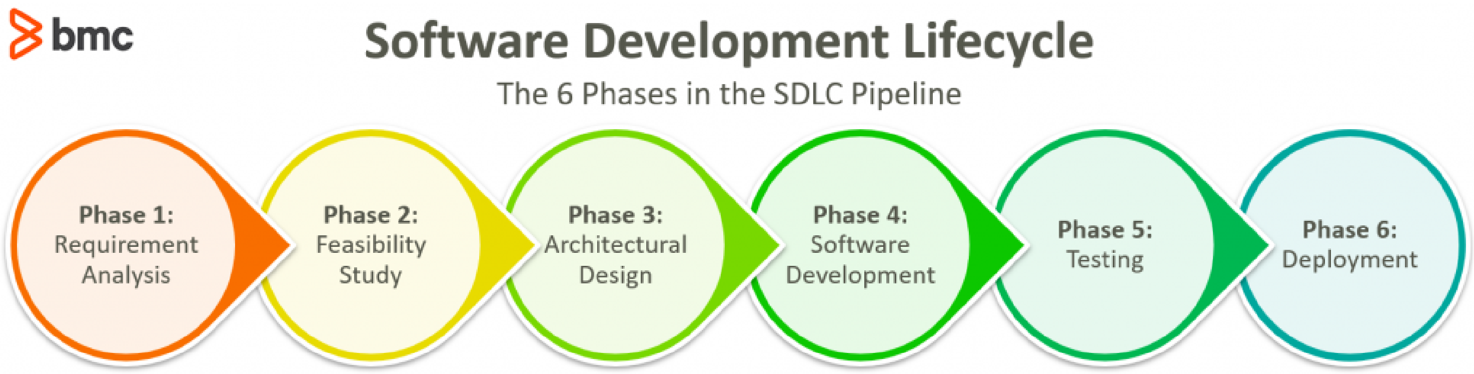
TESTING AUTOMATION EXPLAINED: WHY & HOW TO AUTOMATE TESTING

The purpose of testing in Quality Assurance (QA) is to understand the risks associated with software builds. As software applications become increasingly distributed and complex, QA must perform a variety of tests repeatedly. Across the Software Development Lifecycle (SDLC), these tests are automated, shifted left, and conducted continuously by the collaborating teams of Devs and QA.


bmc
Software Development Lifecycle
The 6 Phases in the SDLC Pipeline
Phase 1: Requirement Analysis
Phase 2: Feasibility Study
Phase 3: Architectural Design
Phase 4: Software Development
Phase 5: Testing
Phase 6: Deployment

The increasing scale and scope of testing in DevOps, however, requires automating test. This is achieved in two key ways:

- Using advanced test automation tools
- Writing custom test scripts that subject new pieces of code to real-world use cases

In this article, we will discuss what testing automation is and how to get started with testing automation in your IT organization.

What is test automation?

Testing automation is the practice of automating the test procedure of comparing the actual performance of software with the expected outcomes, particularly when subjected to specific evaluation criteria such as:

- User stories
- Application requirements

The procedure for test automation involves:

1. Defining test requirements.
2. Building a test environment.
3. Managing tests and data points.
4. Utilizing the results to drive key decisions in the SDLC process.

The testing procedure is repeatable and often scalable to multiple test use cases. These tasks are usually difficult to perform manually, at least at the frequency and scale that is required to evaluate new builds to a large and distributed code base.

Popularity of test automation

Automated testing is increasingly the norm in DevOps and Agile driven organizations. Surprisingly, most IT organizations already use some form of it. According to recent research, only 11% of IT shops use no test automation. But, and there's always a but, with 25% of new startups already prioritizing in-house development, the challenge is to produce high quality software in terms of:

- Performance
- Security
- Cost-value propositions

So, even if you're already doing some test automation, you'll likely want—need—to implement it more widely to continue to stay ahead.

Why automate testing?

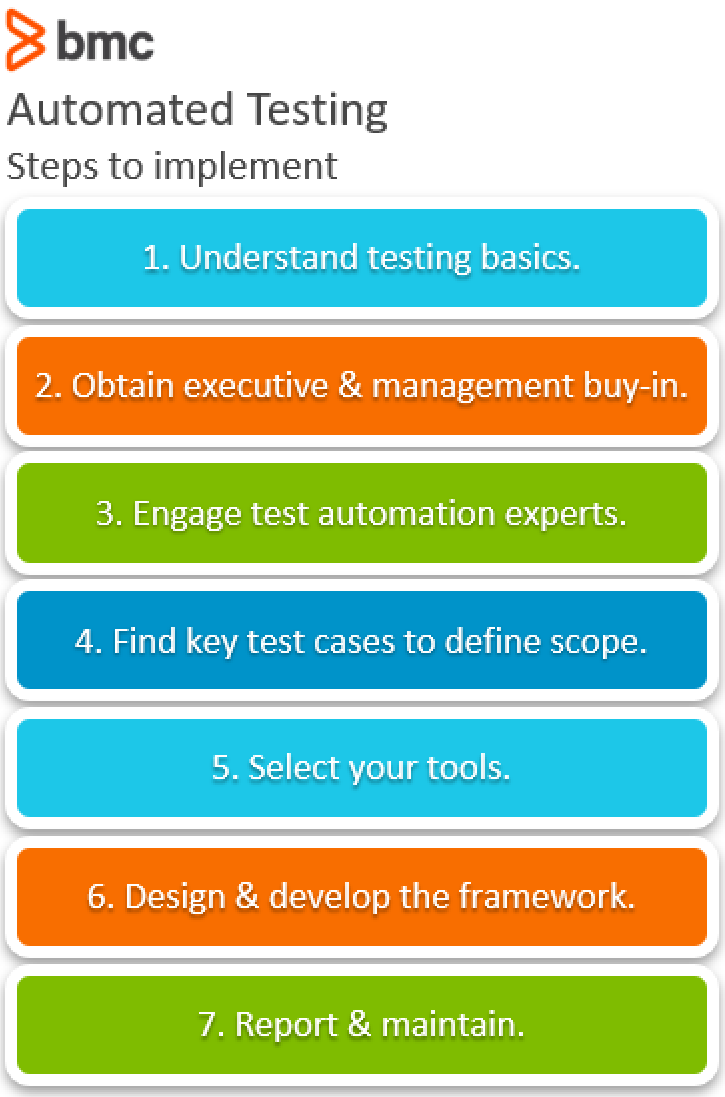
These characteristics make testing automation useful in the following ways:

- Saves time, money, and human resources.
- Navigates software architecture that are distributed and complex.
- Modern SDLC frameworks such as Agile and DevOps require automation to work successfully.
- Follows test frameworks and workflows without the risk of human error.
- Time-consuming tests don't require human supervision and can be conducted for prolonged durations.
- Test coverage can be readily enhanced. (Manual testing may not suffice in testing for an exhaustive set of requirements.)
- Provides accurate results and can be calibrated to mitigate noisy results (from non-deterministic and random parameter performance) and inaccuracies.

- Frees up employees for meaningful, interesting, and non-repetitive tasks, encouraging employee engagement.
- Code is reusable and scalable across multiple applications.

How to automate testing

This following is a high-level overview of the steps involved in getting started with testing automation. By following this approach, you can introduce a seamless testing automation procedure and mitigate the associated risks.


bmc
Automated Testing
Steps to implement
1. Understand testing basics.
2. Obtain executive & management buy-in.
3. Engage test automation experts.
4. Find key test cases to define scope.
5. Select your tools.
6. Design & develop the framework.
7. Report & maintain.

1. Understand testing basics

Make sure you know essential testing practices, like:

- Writing test cases for software
- Selecting test automation frameworks
- Following DevOps testing best practices

2. Obtain executive and management buy-in

Organizational change is hard. If the manual testing procedure works with acceptable results, the management team may not be inclined to invest in expensive automation tools and transform their

SDLC practices. A thorough cost-benefit analysis will be required.

3. Engage test automation experts

Test automation architects and engineers are rare but necessary to develop an optimal testing automation strategy. Automation requires a strategic approach to identifying capabilities, frameworks, and processes to take advantage of technology in the testing procedure.

4. Find key test cases to automate and define scope

Agile and DevOps follow rapid build and release cycles. Automated tests with large coverage and application requirements are time-consuming. Therefore, these tests must be based on the technical and business priorities. Knowing how to simplify the technical complexity and testing only what's necessary is key to a successful testing automation strategy.

5. Select your tools

With a vast selection of tooling available, finding the right one is challenging. It's important to fully understand the testing requirements and make the selection from business, technical, and user perspectives. The tool should conveniently support the testing strategy. The tool should not introduce significant technical challenges or a learning curve for developers.

6. Design and develop the framework

Once the scope of testing is defined and the tooling is available, the next step is to design the framework, prepare automation tests, schedule execution, and finalize the test deliverables. Tests can be executed continuously, in parallel, locally within in-house servers, or using cloud-based IT environments.

7. Report and maintain

Make finalized results available in easy-to-tread reports that make it easier for developers to proactively spot issues and fix them early during the development stage. The testing procedure should be improved on a continuous basis, requiring maintenance and update of scripts over the course of release cycles.

Related reading

- BMC DevOps Blog
- DevOps Guide, a series of articles and tutorials
- Testing in DevOps: Introduction & Best Practices
- What's Testing as a Service? TaaS Explained
- Deploying vs Releasing Software: What's The Difference?