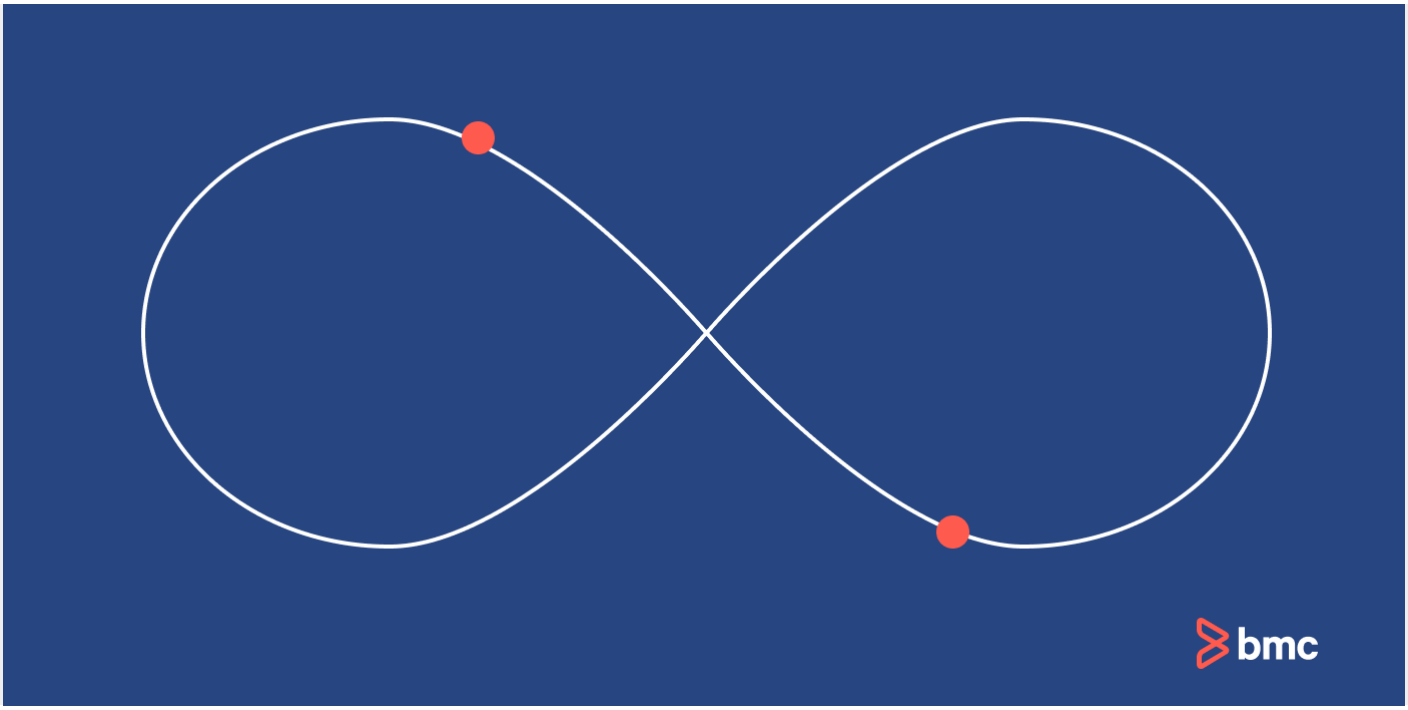
SHIFT LEFT TESTING IN SOFTWARE DEVELOPMENT



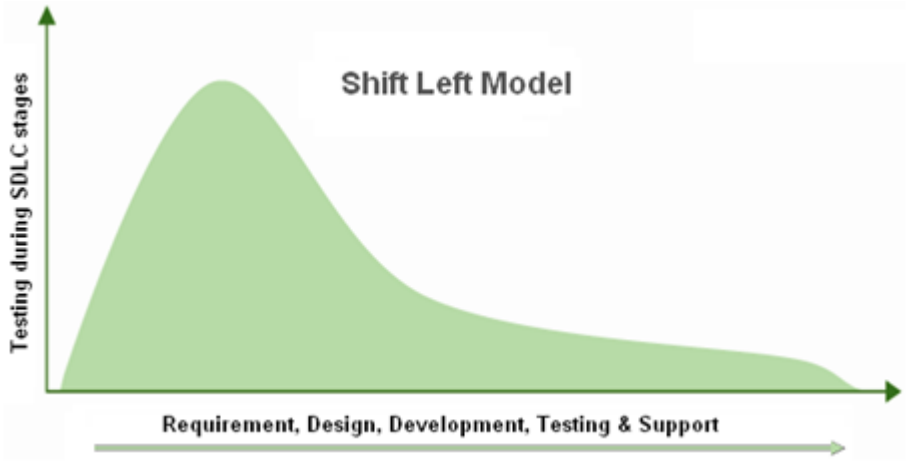
Agile practices, being adopted almost universally, necessitate faster and earlier testing in the software development lifecycle (SDLC). Bringing development and testing together early is commonly referred to as 'shifting left'.

Shift Left and traditional testing

Shift Left testing is a change from what is typical in a waterfall software development project. Traditional approaches start testing immediately prior to releasing into production. Testing this late in the process means that when bugs or usability issues are inevitably found, the release is delayed until these are fixed.

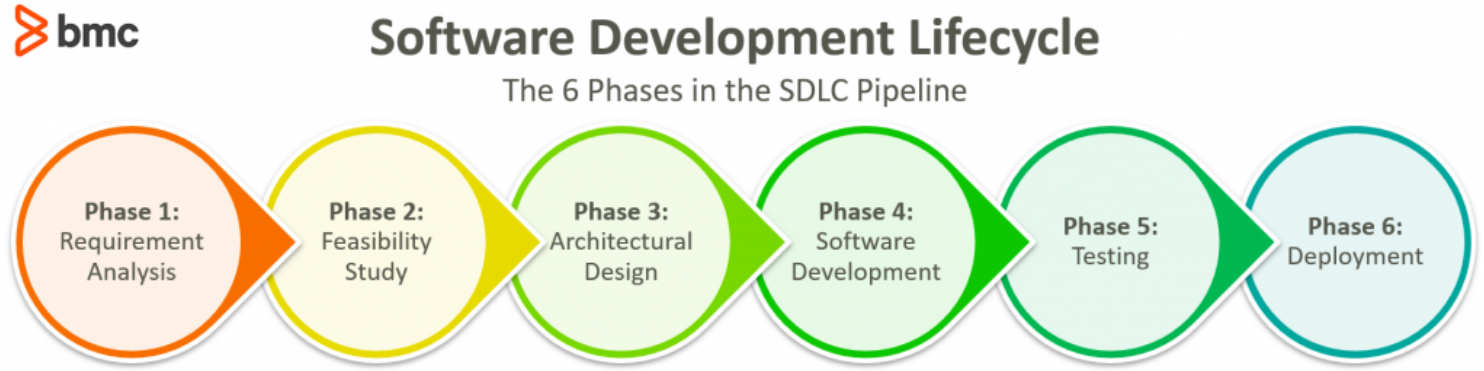
Prior to adopting agile practices like Shift Left development, testing has been a bottleneck that seriously impedes on-time project delivery. Eliminating this bottleneck is one of the key benefits of Shift Left in DevOps.

What does Shift Left mean in software development?

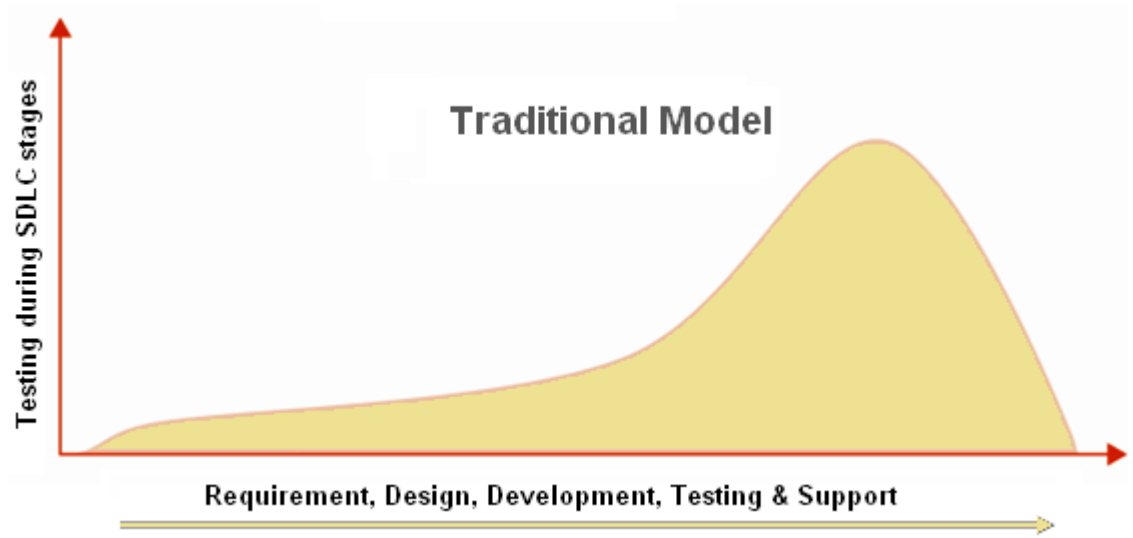


Shift Left is a practice intended to find and prevent defects early in the software development process. The idea is to improve quality by moving tasks to the left as early in the lifecycle as possible. Shift Left testing means testing earlier in the software development process.

The easiest way to explain the meaning of Shift-Left software testing is to think of the development cycle as a line running from left to right.



In the old model, testing only came into play on the far right of the line. Recognizing the bottleneck here, we now want to move the initiation of testing as far to the left as possible.



Why is Shift Left important in DevOps?

In the traditional software development model, requirements are kept on the left side of the plan, and the delivery and testing requirements on the right. The problem is that these practices can't

handle changing expectations and requirements, resulting in negative outcomes for the business such as:

- Increased costs
- Increased time to market
- Unexpected errors

6 benefits of Shift Left testing

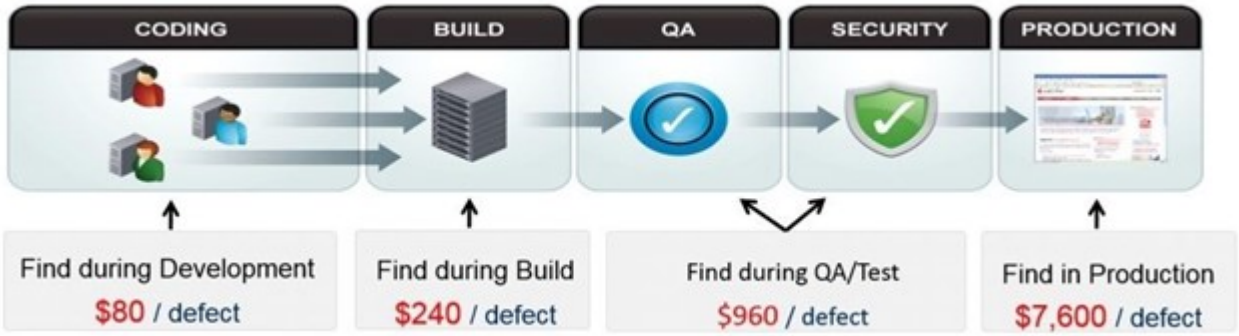
There are several benefits that can be obtained by adopting a Shift Left strategy in DevOps. Here are some of the most important:

1. Reduced costs

Cost alone is a very strong benefit of the Shift Left approach to testing. Estimates indicate that over half of all software defects could be identified during the requirements phase, with less than 10% emerging during the development phase of the lifecycle. The cost of resolving these defects works in reverse:

A defect that is removed after the product has gone into product will cost around 100 times more than one that is identified and removed during the requirements phase.

Research from the Ponemon Institute, in 2017, found that if vulnerabilities get detected in the early development process, they may cost around $80 on an average. But the same vulnerabilities may cost around $7,600 to fix if detected after they have moved into production.



2. Enhanced testing automation

Shifting left gives a greater ability to automate testing. Test automation provides some critical benefits:

- Much fewer human errors
- Increased test coverage (multiple tests can be conducted at the same time)
- Ability for testers to focus on more interesting and fulfilling tasks
- Fewer production issues

3. Increased software delivery speed

Earlier means faster. When you find defects earlier in the production cycle, you can also fix them a lot faster. As a result:

- The time between releases can reduce significantly.

- The quality of software improves.

4. Improved product quality

The Shift Left approach emphasizes the need for developers to concentrate on quality from their earliest stage of a software build, rather than waiting for errors and bugs to be found late in the SDLC.

5. Increased client satisfaction

Faster delivery of software with less defects is a major benefit of the Shift-Left approach.

If nothing else convinces you that this is a good move, then the smiles on the faces of your business partners should be all you need.

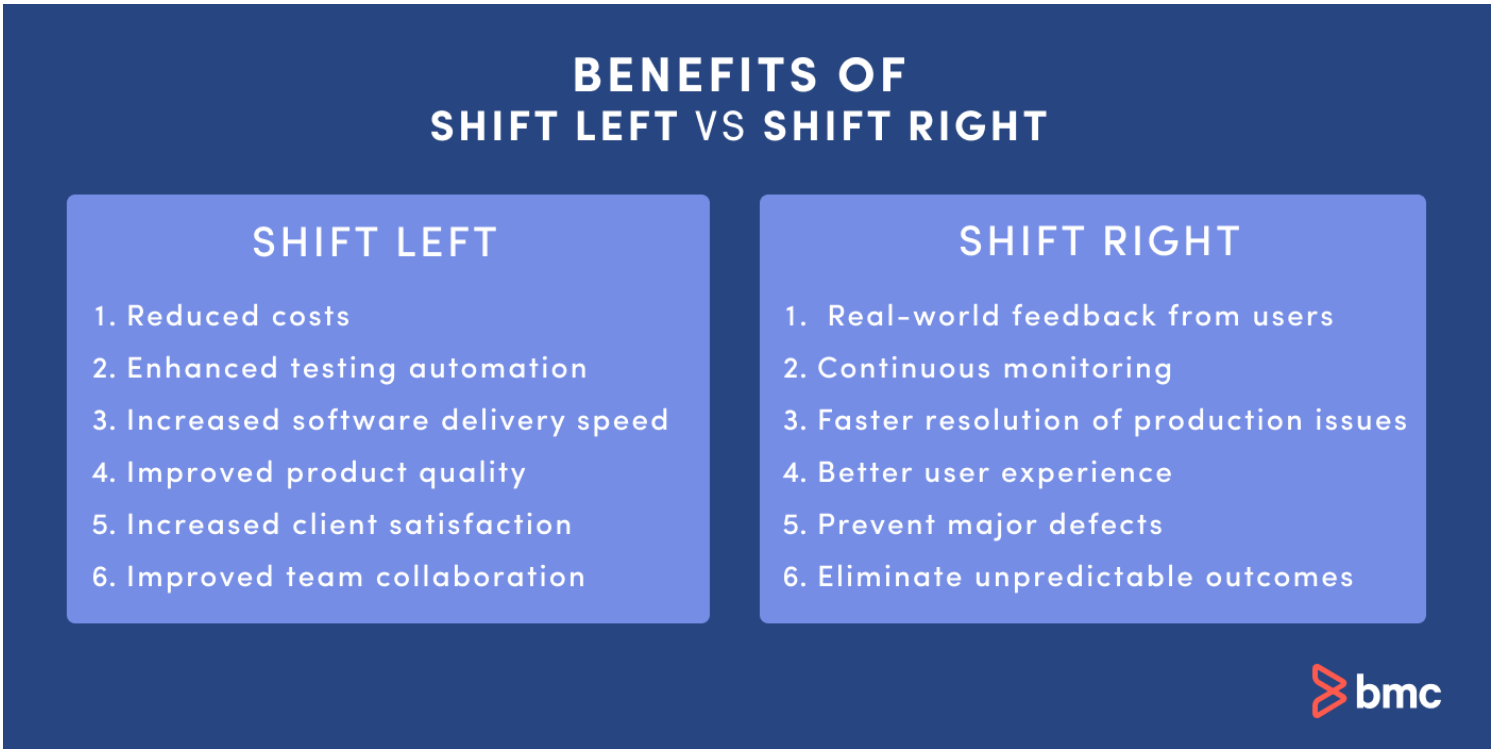
Learn from the choices Humana made when selecting a modern mainframe development environment for editing and debugging code to improve their velocity, quality and efficiency.

6. Improved team collaboration

Shifting left enables product teams perform daily tasks like:

- Testing
- Providing feedback
- Reviewing changes and progress

Shift Left vs Shift Right in DevOps



A Shift Left testing approach may not always be able to deliver optimal performance and functioning in a real-world environment. In such situations, a Shift Right testing strategy may help to:

- Enhance customer experience

- Provide scope for implementation of test automation
- Ensure better test coverage

What is Shift Right in software development?

Shift Right testing is initiated from the right, i.e., post-production. In this Shift Right practice, you'll test a completely built and functioning application to ensure performance and usability traits. Reviews and feedback from targeted users further help in enhancing the quality of the software.

An important characteristic of the Shift Right approach is a willingness to:

- Validate a hypothesis by trying out new solutions
- Collaborate with customers to determine what is working (instead of working from assumptions)

Continuous feedback from users may help in responding better to software failures.

6 benefits of Shift Right testing

When you adopt Shift Right testing in your development processes, you can gain significant benefits.

1. Real-world feedback from users

Testing in production allows you to see how software performs in the real world by actual users in actual scenarios. While you can try to replicate such environments in earlier testing, use in real situations may turn up unexpected issues.

2. Continuous monitoring

The DevOps culture of continuous delivery of software improvements requires continuous testing. Shift Right testing and monitoring provides the feedback loop needed to make frequent releases with confidence.

3. Faster resolution of production issues

With Shift Right testing, you will find issues faster so you can address them with speed. With efficient issue resolution, your users suffer less downtime and can depend on more reliable releases.

4. Better user experience

When users are engaged with your team, they enjoy fast and responsive issue resolution, but also fewer issues over the long run. Eliminating bottlenecks and user interface issues, and employing continuous improvement approaches, leads to higher user satisfaction.

5. Prevent major defects

The approach allows you to identify performance issues, problems with functionality and glitches in the user experience so you can respond quickly, before these problems bloom into major setbacks.

6. Eliminate unpredictable outcomes

Shift Right testing puts you in a proactive position with continuous monitoring that dramatically lowers the possibility of unpleasant surprises.

How to implement Shift Left testing

There are some key Shift Left strategies that will help you adopt this approach to software testing.

Demand planning

Test analysts will engage with business and operational stakeholders, providing a forward view of demand. Having this view enables you to—ahead of time—plan and finalize:

- The budget
- Resourcing
- Test strategies

Demand planning is an integral part of the Shift Left approach and provides a starting point for all other activities in the test lifecycle.

Static testing

Static testing is carried out in the early cycles of the project, and includes validation of requirements and design. The purpose of static testing is to find defects early in the life cycle that could prove to be very expensive to remove in the later phases of the project.

Use appropriate checklists to verify and validate requirements and design. Log defects into a defect management tool.

Unified test strategy

This is an overall, high level strategy for testing end-to-end—from unit testing through user acceptance testing (UAT), operational readiness testing (ORT), and post-deployment testing. The strategy will cover all phases of quality control, defining clear responsibilities.

A unified test strategy allows you to analyze dependencies on environments, stubs, automation, and test data—ensuring that the respective teams can fulfill the needs.

Risk-based analysis

Risk-based analysis is carried out to determine the impact and likelihood of failure for each test scenario. This approach is used for functional, non-functional and regression types of testing.

Once the test cases are established, decide the priority for the test cases based on the finished analysis. Discuss the impact of failure with the business analyst or designer. Determine the likelihood of failure from the development team.

Challenges in adopting Shift Left testing

Challenges to Shift Left testing can impede adoption. Some of these include:

- **Insufficient leadership:** Leaders need to foster new habits and incentivize the right behaviors. A focus on meeting a launch date, for example, might mean there's no time for Shift Left testing.
- **Inability to change processes:** Teams can get set in their ways and rely on familiar tools and approaches. Cultures can be tough to change.
- **Limited resources:** Shift Left testing may be perceived as requiring more from teams without supporting the heavier workload.
- **Failing to engage testers early:** Your team may delay bringing in testers during the earliest phases, where they can make the greatest contribution.
- **Unaligned testing scope**: Testing must match usage scenarios and changes in the software may not get reflected in the testing protocols.

Related reading

- BMC DevOps Blog
- What Is Continuous Testing?
- Load Testing, Performance Testing & Stress Testing Explained
- Testing Frameworks: Unit Tests, Functional Tests, TDD & BDD Explained
- What is Parallel Testing?
- Patch vs Hotfix vs Coldfix vs Bugfix: Differences Explained