International Journal for Multidisciplinary Research (IJFMR)
E-ISSN: 2582-2160 ● Website: www.ijfmr.com ● Email: editor@ijfmr.com

DevOps and Continuous Testing in SAP

Sireesha Perabathini

Independent Researcher
Illinois, USA
perabathinisireesha@gmail.com

Abstract
In today's fast-paced enterprise environments, SAP systems are pivotal in managing key business operations. As the demands for faster and more reliable software delivery increase, organizations are turning to DevOps and continuous testing to streamline development and deployment cycles. This paper explores how DevOps practices and continuous testing can be implemented within SAP landscapes, improving code quality, reducing manual errors, and accelerating the delivery of SAP-based solutions. We discuss the various components of SAP (ABAP, HANA, Fiori, PI/PO), tools for automation, and best practices for integrating continuous integration (CI), continuous delivery (CD), and testing in SAP systems.

Keywords: DevOps, Continuous Testing, SAP, CI/CD, ABAP, SAP Fiori, SAP HANA, SAP Cloud Platform, SAP Business Technology Platform (BTP).

1. Introduction

The SAP platform becomes indispensable to businesses undergoing digital transformation through its automation and optimization capabilities for business processes. The complexity of SAP systems' infrastructure requires an agile and robust delivery model to handle frequent updates and changes. DevOps—the practice of combining software development (Dev) and IT operations (Ops)—has gained traction as a means to achieve greater efficiency, collaboration, and speed in software delivery. A central aspect of DevOps is continuous integration (CI), continuous delivery (CD), and continuous testing.

This paper discusses the implementation of DevOps practices and continuous testing within SAP systems, focusing on ABAP (SAP's proprietary programming language), SAP HANA (database), SAP Fiori (UI), and SAP Process Integration (PI) and Process Orchestration (PO). It highlights tools, best practices, and the benefits of integrating CI/CD pipelines and automated testing in SAP environments.

2. DEVOPS in SAP

A. CI/CD Pipelines for SAP

DevOps focuses on automating manual processes such as building, testing, and deployment, resulting in more consistent and faster releases. Implementing CI/CD pipelines in SAP is essential for reducing errors, improving collaboration between teams, and accelerating the release cycle.

1) CI for ABAP: In SAP, ABAP code can be version-controlled using systems like Git or SVN. Tools like Jenkins or GitLab CI can be used to automate the build process, ensuring code quality through automated checks and unit tests (e.g., ABAP Unit tests).

2) CD for SAP: The CD aspect of DevOps in SAP involves automating the deployment of code changes across environments (development, test, and production). SAP Transport Management and tools like SAP CTS+ ensure that code is consistently and reliably moved between systems.

3) SAP Cloud Platform (SCP): For cloud-based SAP applications, SCP and SAP Business Technology Platform (BTP) [2] facilitate the deployment of applications, leveraging cloud-native features to simplify CI/CD integrations.

B. Version Control and Collaboration

Version control in SAP stands as an essential process for overseeing code modifications and monitoring development progress. While SAP traditionally uses the Transport Management System (TMS) for managing ABAP code, the integration of tools like abapGit [1] enables developers to use modern version control systems like Git for managing ABAP code, along with SAP Fiori and SAP HANA scripts. Versioning ensures code stability, traceability, and accountability.

3. Continuous testing in SAP

Continuous testing stands as an essential component that supports DevOps operations and continuous integration/continuous deployment pipelines. This process verifies application quality and checks that changes made to software don't cause regressions or break existing functionality, which helps maintain stable and reliable performance. Within SAP environments continuous testing safeguards against disruptions to core processes by monitoring customizations as well as enhancements and integrations.

The complexity of SAP landscapes—including ABAP (SAP's programming language), SAP HANA (database), SAP Fiori (UI), and SAP S/4HANA—necessitates a robust, automated testing strategy to meet the speed and quality expectations set by DevOps practices. This section explores how continuous testing is integrated into SAP environments and outlines key testing types.

SAP testing encompasses multiple layers that start with unit testing and extend to integration and performance testing. The goal of Continuous testing in SAP is to automate the validation of individual components and End to End business processes through a predefined test sequence that must be passed before deployment. Below are the Key Testing Types in SAP for DevOps Pipelines.

C. Unit Testing:

Unit testing in SAP focuses on validating the functionality of individual software components, such as ABAP classes, methods, HANA database functions, or custom-built logic in SAP Fiori applications.

- ABAP Unit Testing: In SAP, ABAP Unit Testing ensures that individual pieces of ABAP code (such as methods and classes) work correctly. This involves writing test cases using the ABAP Unit Test Framework to validate the functionality of ABAP code. When developers commit code, the ABAP Unit tests are executed automatically within the CI pipeline, ensuring that the code adheres to expected behavior. This is critical to ensure that custom code does not introduce errors or regressions into the system.

- HANA Database Unit Testing: For HANA database objects (stored procedures, views, and tables), unit tests are vital for validating that database changes do not introduce errors or performance issues. Tools like SAP HANA Studio [3] or other SQL-based testing frameworks like SAP Web IDE for HANA are used for testing database functions in an automated manner.

2. Integration Testing:

Integration testing ensures that different modules or systems interact as expected. In SAP, integration tests are essential for confirming that custom-built or third-party applications work seamlessly with core SAP systems.

- API and Web Service Testing: Many SAP applications integrate with external systems or services via APIs (e.g., OData, REST APIs). Integration tests in these cases ensure that data exchanged between systems is valid and follows the required formats and protocols. Tools like Postman or Swagger can be used to automate the validation of APIs in the SAP landscape.
- SAP PI/PO Integration Testing: For SAP Process Integration (PI) or Process Orchestration (PO), integration testing ensures that the communication between SAP and external applications works smoothly. Automated tests can verify the flow of data through adapters, mappings, and transformations. SAP PI/PO testing can be automated with Selenium, SOAP UI, or Tricentis Tosca [4].
- End-to-End Business Process Testing: This involves testing complete business workflows (e.g., order-to-cash, procure-to-pay) to ensure that all components, from front-end SAP Fiori applications to back-end ABAP processing and SAP HANA databases, interact correctly. Worksoft Certify and Tricentis Tosca [4] are tools commonly used for automating end-to-end tests.

3. Regression Testing:

Regression testing in SAP ensures that new features or code changes do not break existing functionality. This is especially important when deploying updates or patches to SAP S/4HANA or when performing system upgrades.

- Automating Regression Tests: In a typical CI/CD pipeline, automated regression tests can be scheduled to run after every code change, patch, or update. These tests verify that the system continues to function as expected after new changes are integrated.

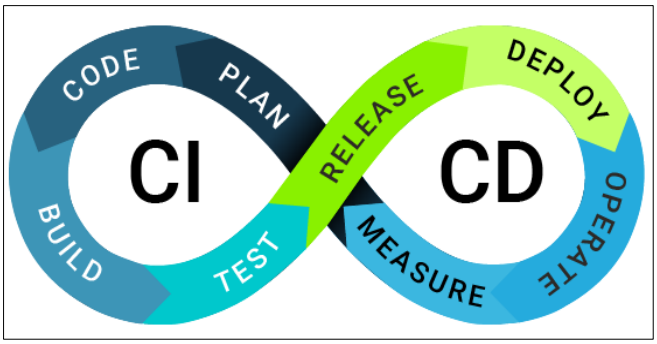

Figure 1. DevOps and Continuous Testing

- Test Suite Management: SAP's Solution Manager manages both test cases and scenarios as well as regression tests. It allows integration with test automation tools, ensuring all critical business processes are validated across multiple versions and environments.

4. Performance Testing:

Performance testing in SAP ensures that the system performs well under varying levels of load and traffic. This is especially important for SAP HANA, as database performance can significantly affect the responsiveness of SAP applications.

- Load and Stress Testing: Automated performance tests can simulate multiple users or high-volume transactions to test how the system behaves under stress. Tools like JMeter, LoadRunner, or SAP Performance Testing Tools help simulate user interactions with SAP systems, measure response times, and assess the scalability of the environment.
- Database Performance Testing: In SAP HANA, performance tests ensure that database queries are optimized, and the system can handle large data volumes without degradation. JMeter or SAP HANA's native performance tools (e.g., HANA Studio, HANA Cockpit, and SQL analyzer [5]) can be used to benchmark SQL queries, stored procedures, and overall database performance.

5.Security Testing:

Security testing ensures that SAP systems are free from vulnerabilities that could potentially compromise sensitive data. Security tests must be integrated into the DevOps pipeline to ensure that vulnerabilities are detected early and addressed before deployment.

Vulnerability Scanning: Automated security tests, including vulnerability scans, penetration testing, and security code analysis, can be incorporated into the pipeline. Tools like Fortify [6], and OWASP ZAP can be used to scan code, APIs, and the entire application for potential security flaws and data breaches [10].

4. CI/CD and testing tools for SAP

D. Tools for CI/CD in SAP

Several tools and platforms are available to help implement CI/CD pipelines for SAP systems:

- Jenkins: Jenkins is a widely used automation server for building and deploying software. It integrates with SAP systems to automate the build, test, and deployment processes for ABAP, HANA, and Fiori applications.
- GitLab CI: GitLab CI is a robust tool that supports both version control and automated pipelines. It can be integrated with SAP systems to enable seamless CI/CD operations.
- Azure DevOps: Azure DevOps provides a comprehensive suite for managing projects, version control, build automation, and CI/CD. It supports SAP systems through custom pipelines for deployment and testing.

E. Test Automation Tools

Automation is a key element of continuous testing in SAP. The following tools are widely used for test automation within SAP landscapes.

1. SAP CBTA (Component-Based Test Automation): SAP CBTA is a specialized automation tool developed to automate business processes and workflows within SAP systems. The tool connects seamlessly with SAP Solution Manager to manage test cases and tracking the execution of test scripts. The tool supports testing of SAP systems, making it suitable for testing of integrated business processes across the SAP landscape [7].

2. Tricentis Tosca: Tricentis Tosca enables organizations to automate functional testing and regression testing along with API testing throughout the complete SAP system environment. The solution enables seamless integration with SAP S/4HANA and other SAP platforms including SAP Fiori and SAP BW to automate comprehensive business processes using minimal scripting [4].

3. UIVeri5: UIVeri5 is a testing tool specifically created to test UI5-based applications. It allows you to perform end-to-end testing for applications built using the UI5 framework. UIVeri5 enables automated functional testing of SAP Fiori and other SAP UI5 apps in a way that's similar to tools like Selenium but is tailored for UI5 applications [8].

4. Postman/ReadyAPI: Postman and ReadyAPI are tools for automating API testing. In SAP, they are used to test integrations between SAP and external applications via web services and APIs (e.g., OData services in SAP S/4HANA). They allow you to define API test cases, automate them, and integrate them into the CI/CD pipeline.

5. Best practices and challenges in devops and continuous testing in SAP

F. Best Practices for Continuous Testing in SAP

1. Shift Left Testing: Start testing earlier in the development cycle to catch issues sooner. Shift-left testing works best for the unit-level tests that have shorter execution times. All tests within the CI/CD pipeline which developers activate by triggering builds must operate swiftly to prevent any build process delays. End-to-end tests, transaction testing, static code analysis, and security testing benefit from execution outside CI/CD pipelines with daily or more frequent run schedules. Developers receive early quality feedback from these tests when they are automated outside the CI/CD pipeline to prevent build delays [9].

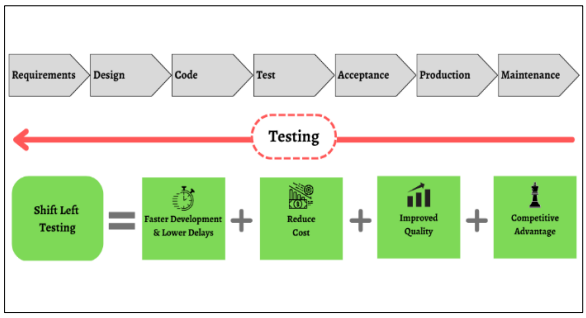

Figure 2. Shift Left Testing

2. Automate the Entire Testing Lifecycle: Automation is key for continuous testing. Automate unit tests, regression tests, integration tests, and performance tests to ensure that no step is missed and to achieve greater efficiency in the testing process.

3. Use a Test Data Management Strategy: Implement a strategy to manage test data effectively. Use data anonymization tools or create synthetic data sets for testing purposes to ensure that sensitive information is protected.

4. Maintain Test Coverage and Quality Metrics: Continuously monitor the effectiveness of tests and maintain comprehensive test coverage across all critical areas, such as user interfaces, back-end logic, performance, and security.

5. Adopt a Test-Driven Development (TDD) Approach: Encourage developers to write unit tests before writing business logic. This ensures that every feature is covered by automated tests.

6. Integrate with SAP Solution Manager: Use SAP Solution Manager for managing test cases and tracking changes to maintain control over the entire testing process.

7. Automate Performance Testing: Integrate performance testing tools like LoadRunner or JMeter into the CI/CD pipeline to continuously validate system performance.

G. Challenges:

1. Complexity of SAP Landscapes: SAP systems are often large and complex, consisting of numerous modules, custom developments, and third-party integrations. This complexity can make testing difficult, as there are many interdependencies between systems and components. The complexity of SAP systems, with various modules, landscapes (on-premise, cloud, hybrid), and technologies, makes implementing CI/CD challenging.

2. Transport Management: Automating the transport process can be difficult, as SAP relies heavily on the Change and Transport System (CTS) to move objects between systems.

3. Tool Integration: Integrating third-party tools into the SAP ecosystem can be complex, especially for legacy SAP systems that may not support modern CI/CD practices.

4. Limited Support for Traditional CI/CD Tools: SAP's traditional transport management system, CTS (Change and Transport System), does not always integrate well with standard CI/CD tools. This can lead to challenges when attempting to automate the deployment and testing of custom SAP developments.

5. Test Data Management: Managing test data across different environments can be a challenge. SAP systems often require large and complex datasets to accurately simulate real-world usage. Managing and refreshing test data without compromising sensitive information is essential but challenging.

6. Performance Testing at Scale: While tools like JMeter and LoadRunner can simulate load and stress on SAP systems, it is often difficult to replicate real-world conditions in highly customized SAP environments. Performance tests must account for the unique configurations, integrations, and data volumes in the SAP landscape.

7. Security: Ensuring that sensitive data (e.g., credentials, certificates) is protected during automated deployments and tests is a critical concern.

6. Conclusion

Organizations must incorporate DevOps practices and continuous testing within SAP environments to enhance software quality and shorten release cycles while promoting teamwork between different departments. Organizations can build effective CI/CD pipelines for ABAP, Fiori, HANA, and other SAP components by utilizing tools such as Jenkins, GitLab, SAP CBTA [7] along with automated testing frameworks. Despite implementation challenges within complex SAP systems DevOps practices deliver superior software delivery speed and reliability which makes overcoming these obstacles worthwhile. To maintain high-quality applications and fulfill business requirements organizations must embrace DevOps and continuous testing alongside the ongoing development of SAP systems.

References

1. SAP Blogs, "Versioning your local objects using abapgit,"2019[Online]. Available: https://community.sap.com/t5/application-development-blog-posts/versioning-your-local-objects-using-abapgit/ba-p/13405479
2. SAP Blogs, "Beginner-friendly introduction to SAP BTP,"2023[Online]. Available: https://community.sap.com/t5/enterprise-resource-planning-blogs-by-members/beginner-friendly-introduction-to-sap-btp/ba-p/13550560
3. SAP Blogs, "Unit Testing SAP HANA Views – A Beginners Guide,"2015[Online]. Available: https://community.sap.com/t5/technology-blogs-by-members/unit-testing-sap-hana-views-a-beginners-guide/ba-p/13143606
4. SAP Blogs, "SAP Application Testing Solutions by Tricentis - Overview,"2021[Online]. Available: https://community.sap.com/t5/technology-blogs-by-sap/sap-application-testing-solutions-by-tricentis-overview/ba-p/13510493#:~:text=SAP%20Enterprise%20Performance%20Testing%20by%20Tricentis%20covers,answering%20the%20question:%20%E2%80%9C%20Does%20it%20scale?%E2%80%9D
5. SAP Blogs, "Spotlight: Using the SQL Analyzer with SAP HANA Cloud,"2021[Online]. Available: https://community.sap.com/t5/technology-blogs-by-sap/spotlight-using-the-sql-analyzer-with-sap-hana-cloud/ba-p/13481042
6. PeterBarker, "Integration between “SAP Code Vulnerability Analyzer” and “SAP Fortify by Micro Focus,"2019[Online]. Available: https://community.sap.com/t5/application-development-blog-posts/integration-between-sap-code-vulnerability-analyzer-and-sap-fortify-by/ba-p/13401595
7. SAP Blogs, "Composite test script creation - Component Based Test Automation(CBTA),"2023[Online]. Available: https://community.sap.com/t5/technology-blogs-by-members/composite-test-script-creation-component-based-test-automation-cbta/ba-p/13562346

8. SAP Blogs, "Choose the best Test Automation for your SAP Fiori App,"2021[Online]. Available: https://community.sap.com/t5/technology-blogs-by-members/choose-the-best-test-automation-for-your-sap-fiori-app/ba-p/13511103
9. Isaac Sacolick, "How to improve CI/CD with shift-left testing,"2020[Online]. Available: https://www.infoworld.com/article/3516013/how-to-improve-cicd-with-shift-left-testing.html?upd=1657725549072
10. OWASP, "OWASP Benchmark," OWASP Foundation, 2019. [Online]. Available: https://owasp.org/www-project-benchmark/