The Quality Imperative: SAS Institute's Commitment to Quality

A corporate statement of SAS' commitment to product quality, service quality, and customer satisfaction

Testing

Testing

Overview

SAS R&D staff embrace both the challenges and benefits of measuring software quality throughout its life cycle. From development engineers applying continuous integration (CI) techniques to test engineers verifying and validating SAS products through continuous testing (CT), the effort to ensure that our software meets or exceeds customer expectations never ends.

Testing software requires skilled software professionals, effective testing processes, and robust test automation tools. Given our CEO's core philosophy that staff are the company's greatest asset, SAS actively recruits, trains, and retains highly qualified development and test engineers with domain and product expertise. Early in the development life cycle, test and development engineers engage with product management to understand and refine software requirements and to increase their testability. Team members jointly contribute to enhancing test strategies, including how to best automate and implement a cohesive testing program for their product.

Test engineers meet regularly to discuss process improvement opportunities and test automation innovations. A central quality portal and internal testing site provides guidance and best practices. Managers participate in formal quality reviews and provide a key signature for the product sign-off before the software can ship.

Test engineers work closely with software engineers to verify the successful implementation of new features and to validate the continued baseline of existing features. Teams choose from an evolving set of testing methodologies, including requirements-based testing, use-case testing, exploratory testing, consumer-driven contract testing, and systems testing. Product development teams also collaborate with specialized teams within SAS that focus on key areas such as software security, performance, accessibility, localization, and other dimensions of quality.

When defining a test strategy for their products, product teams perform these tasks:

- Provide feedback on software requirements and design
- Design test plans or strategies
- Write and support test procedures, automation tools, and reusable test libraries
- Perform early exploratory testing
- Write and execute automated tests and manual test scripts
- Report defects
- Verify fixes
- Report test results
- Monitor quality metrics
- Analyze and improve test coverage
- Review customer documentation
- Test nonfunctional requirements, such as performance and security
- Test for adherence to R&D policies and standards
- Write and support qualification tests and samples

Directors and managers are accountable for ensuring that product development teams follow SAS' quality procedures.

Test Documentation

Each product team is responsible for planning the overall testing strategy for their projects. Teams also use agile approaches for maintaining project-specific test plans and documentation. The entire R&D community collaborates on a due diligence checklist to ensure consistency for testing sign-off. This checklist serves as a source document when planning the test strategy for a software release.

Teams may use a variety of approved test tools and processes—both internal and third party—to document and manage test artifacts and test results. Group test plans document the testing strategy for an entire group or department. Teams may develop individual test plans for specific projects, products, solutions, or features. In addition to test plans, teams may also create test inventories, matrices, or design specifications.

Where appropriate, teams may perform early exploratory testing during the planning and documentation process. This technique enables teams to get acquainted with the software and to understand changes in the current release. Little or no formal documentation is created for this type of testing, although defects can be entered if anomalies are uncovered. As the software matures, teams generate more robust test documentation. Test documentation covers areas such as the functionality of the product that is being tested, error-handling capabilities, and stress or performance testing.

Test Cases

Teams maintain baseline suites of legacy tests to verify product functionality delivered in previous releases. Test engineers continuously improve test suites by designing and writing tests to validate new functionality and store the versioned tests in a source code repository.

Test engineers design test cases to determine if specific components of the software perform correctly. Correctness of a test result is evaluated against existing benchmarks, knowledge of past performance, expected behavior as identified in documented requirements, known results as published, and an understanding of the software's design.

Development teams employ a variety of testing methodologies to verify and validate software, based on the specific needs of the product under test, such as the following:

- Analytical tests, including statistical and numerical validation. See **Appendix 2: Validating an Analytical Component** for details on how SAS handles this critical testing step.
- API and unit tests to verify the correct behavior of components before system integration.
- Compatibility testing to assess the ability of software or web applications to function across different browsers, configurations, or cloud platforms.
- Error testing examines how syntax and run-time error conditions are handled.
- Functional testing determines whether the software functions as expected.
- Internationalization testing checks the software readiness for any required localization development. Localization testing is done by native language users.

- Migration testing checks that customers can move to current versions of the software without problems.
- Performance testing evaluates whether the software performs as well or better than previous releases.
- Regression testing identifies whether software changes have introduced errors or unintended behavior.
- Security testing can include exploitation of OWASP Top 10 weaknesses, Common Weakness Enumeration (CWEs), Common Vulnerabilities and Exposures (CVEs), encryption mechanisms, error handling, input handling and application programming interface security.
- Stress testing creates an overload situation to determine how the software product, procedure, or module functions under the stressed condition. Stress testing also evaluates the ability of the software to recover from overloaded conditions, to measure how the project performs under peaks and ebbs in use.
- System testing validates the complete and fully integrated product.
- Usability testing and accessibility testing evaluate how easy it is to use a particular feature for all customers, regardless of ability.

Testing Tools

Teams employ a broad range of testing tools:

- Multiple industry-standard third-party and open-source test automation frameworks, such as Katalon, Robot, and other industry-standard third-party (including open source) test automation frameworks.
- Coverage analysis tools for C, Java, Go, and JavaScript source code to highlight potential areas for improved coverage and to optimize regression testing.
- Test drivers that execute command-line-based tests on multiple platforms (cloud platforms, operating systems, and programming environments).
- Internal and third-party continuous testing tools to leverage unit, integration, and acceptance test suites for continuous quality.
- Test management and tracking tools to record test cases and their results history for all platforms.
- Internal and third-party tools for continuous integration and validating software fixes.
- Security testing tools that scan for common software security vulnerabilities under both static and dynamic conditions. For more detail, see the Security Testing section.
- Performance monitoring tools
- Problem tracking tools such as Jira for tracking defects, enhancements, issues, and suggestions found during and after software development and testing.

Various reports such as the following are available to teams for evaluating software quality and testing progress:

- Queries of the problem tracking databases about number, age, type, and severity of defects, by internal group structure or product
- Verification status of individual defects as well as responsible individual or department
- Stability of code by tracking the number of test and source files pushed within a given period

Test Execution

Test engineers execute automated tests using both internal and commercially available tools. Test engineers also execute manual test scripts to verify more complex aspects of the software that do not lend themselves easily to automated testing. Such testing might be repeated on several test configurations. Test results are available, typically in graphical or dashboard format, and saved to a repository for traceability and diagnostic purposes.

Performance Testing

Performance regression testing is done across releases and relative to other versions of SAS. For new releases and redeveloped solutions, the group tests against performance requirements that are provided by product management. Each major hardware platform and public cloud provider is also tested for performance characteristics by many of the product groups. Performance testing results are kept for future release comparison.

The testing groups use internal and third-party tools to test compiler effectiveness, Java code performance, C code performance, I/O performance, big data scalability, algorithm effectiveness against third-party databases and SAS internal data sets. Much of the work is automated, and parameters are set so that performance bottlenecks are flagged for analysis. We also check how computer resources are being used (that is, memory, I/O, and CPU). For deep analysis, monitoring and profiling tools such as HP Diagnostics and Datadog are used.

Performance, load, system, and endurance testing are conducted on web-based, Java applications, and rich clients. This testing is based on multiuser scenarios driven under load conditions using application load testing tools such as LoadRunner, Performance Center, and Ranorex. This ensures software quality by identifying performance bottlenecks, memory leaks, and scalability problems.

Performance test engineers provide advice to product development teams and product management on code changes, data architecture changes, application architecture changes, and technical architecture or hardware. They also supply recommendations for cloud provider infrastructures that are most appropriate for SAS' software offerings.

Software Security Testing

Overview

Product teams are required to perform security tests in accordance with internal software security policies, standards, and processes. Security testing includes security function testing, application vulnerability testing, dynamic scanning of applications, and static source code scanning. All software components, including third-party components, are required to be scanned against Common Vulnerabilities and Exposures (CVE) published in the National Vulnerability Database (NVD), maintained by the U.S. government's National Institute of Standards and Technology (NIST).

In addition to scanning web applications and the web application server environment, SAS uses a suite of tests that are specific to SAS technology. Depending on the software type, these tests can include:

- Industry-recognized security scanning approaches to flag common security issues, such as those identified by the Open Web Application Security Project (OWASP), Common Weakness Enumeration (CWE™), Common Architectural Weakness Enumeration (CAWE), and Common Attack Pattern Enumeration and Classification (CAPEC™).
- Testing with users of different role-based security access to make sure that each user has the appropriate access levels.
- Data access, based on row-level permissions, to confirm that data authorization is applied appropriately for each user.
- Password and encryption security.
- Correct behavior with Transport Layer Security enabled protocol (HTTPS).
- Validated credential protection when using SAS/ACCESS engines to connect to data sources (for example, user ID and password).
- Product-specific security tests for appropriate user authorization and error testing.
- Integration testing of security features and controls.
- Penetration tests for some configured deployments.

Application Vulnerabilities Testing

Software resilience to external threats is important to our customers, and SAS software security testing tools are focused on eliminating known application vulnerabilities such as those described in OWASP, CWE™, CAWE, and CAPEC™. Issues that are detected during security testing are entered into the problem reporting system and evaluated promptly for appropriate fixes and resolutions.

SAS has taken the following steps to deliver secure applications:

- Education and training: SAS provides ongoing developer training in techniques to mitigate development errors and vulnerabilities. SAS licenses tools that are designed to generate test cases for security vulnerabilities, including those described on OWASP and CWE lists as well as other lists.
- Deliver shared security components across SAS products: SAS develops shared components and coding guidelines for common issues and an input sanitation filter to provide strong security protection across SAS products.
- Monitor and analyze industry issues: SAS monitors and analyzes industry issues regularly, and draws on the evolving information from OWASP, CWE™, CAWE, and CAPEC™ to evaluate and remediate identified security weakness and vulnerabilities.
- Frequently update security analysis tools and techniques: SAS performs vulnerability testing using the most current tools and techniques for feature and maintenance releases.

Results of vulnerability tests and scans that are conducted by SAS are company confidential. By policy, SAS does not share the tests or the individual results.

Customer Notification

SAS provides several forums that customers can use to get information about updates to SAS products, including security fixes. The SAS security bulletins page (**http://support.sas.com/security/alerts.html**) provides updates about security issues. Security fixes for released products are highlighted through the standard technical support process for hot fixes, including the SAS support community. Customers can subscribe to the community or sign up for support newsletters (or both) to receive regular updates about hot fixes and other important news from SAS.

- To subscribe to SAS Technical Support News, go to **http://support.sas.com/techsup/**.
- To subscribe to the SAS support community, see **https://communities.sas.com/t5/Getting-Started/How-to-learn-about-hot-fixes-to-SAS-software/ta-p/283553**. Note that results can be filtered using the keyword SECURITY.

Release Information
The version of this paper is January 2022.

Unless otherwise indicated, this document relates only to SAS 9.4, SAS Viya, and the products that are available with SAS 9.4 and SAS Viya. It also relates to services from the date of this paper forward. Quality processes are continually evolving. Therefore, SAS reserves the right to modify the processes described in this document at any time. If you are using SAS 9.4 and SAS Viya and have questions about processes in those releases, send email to **qualitypaper@sas.com.**

Learn more about SAS Solutions at sas.com.