Medicaid Enterprise Systems (MES) Testing Guidance Framework

Comprehensive and thorough testing of enterprise software throughout the IT investment lifecycle leads to higher quality software products and increased levels of user satisfaction. CMS expects states to effectively test applications and services so that issues are identified and remediated early in the IT investment lifecycle. Early detection of systems issues reduces the number of errors embedded in the software and the cost of rework later in the development cycle, thereby increasing the overall quality of the delivered system and user satisfaction.

This document lists a set of expectations and recommendations for testing which are defined as follows:

- Expectations describe actions and deliverables that states are **required** to demonstrate and/or provide as evidence.
- Recommendations denote industry best practices that have shown to increase the efficiency and quality of products. CMS encourages states to adopt and follow these recommendations to enhance the quality and reduce risk for MES implementations.

Expectations

The expectations are broken down by major project stages: test planning, test execution, and operational monitoring.[1]

Test Planning

Test planning starts with the acquisition process by including testing expectations in Requests for Proposals (RFPs) and contracts and it continues into the early phases of project planning to organize the testing process. Specifying testing requirements in the contract should leave no ambiguity regarding the role of the state and vendors in the testing process. Including testing-related expectations in the contract also ensures that the vendor's test planning processes and level of rigor are documented and available for CMS review.

Expectation 1: CMS expects state contracts to include requirements for system testing. Further, in accordance with 42 CFR 433.112, states must share the contracts with, and obtain prior approval from CMS, to be eligible for Federal Financial Participation (FFP). Depending on the nature of the procured system (e.g., custom developed, Commercial off-the-shelf (COTS), Software as a Service (SaaS)), the following are examples of testing-related requirements that CMS expects states to include in relevant contracts:

- Ensure that test teams have appropriate skills and are independent of the development teams. For example, the use of automated software testing requires the testing team members to have programming experience to develop the automated tests and validate the test results.
- Define the various testing environments (e.g., integration environment, user acceptance environment), the process of managing testing environments, and conditions for promoting software builds from one environment to another. Test environment configuration should be automated to increase efficiency and protect against test environment misconfiguration.

[1] Most systems engineering processes – such as DOD 5000, ISO/IEC 15288, or iterative development – share similar phases of test planning, test execution, and operational monitoring.

- Define defect severity levels (i.e., what qualifies as a defect for each severity level, and expectations about the turnaround time for fixing defects of different severity).
- Define expectations regarding detailed test cases development for each system requirement.
- Define expectations regarding load and performance testing, and the role of automated testing in these types of tests.
- Describe the process for resolving defects when the system is in production, including replicating the defects in non-production environments and conducting root cause analyses (RCAs).
- Describe the process for loading new software builds in the production environment.
- Describe the process for continuously monitoring the production environment's performance and taking the appropriate actions to proactively deal with potential issues. Examples of monitoring include CPU, memory and disk usage, system response time, event logs, and health of processes and services.
- Define expectations regarding measuring and reporting metrics defined in Service Level Agreements (SLA). Examples of SLA metrics include system availability, system recovery objectives in case of system crashes or natural disasters, and system response time.

Expectation 2: CMS expects states or their vendors to develop and share a master test plan that describes the details for how and what testing will occur. The master test plan should cover elements such as:

- List of stakeholders who need to review and approve the master test plans.
- Project scope and summary, which includes a list of features that will be tested in order to remove ambiguity about the testing scope.
- Types of testing, such as integration testing, user acceptance testing, load testing, etc.
- Test entry criteria, which include information such as description of the test environment, specific setup of required test data, review, and approval of test cases, etc.
- Test exit criteria, which specify the quality gates that need to be met before a project can be rolled into production. Most projects have minimum quality gates that are 100% execution of all test cases, no outstanding critical or high severity defects, etc.
- Test data requirements, (e.g., requirement for the generation of vast amounts of random data). In addition, it is sometimes more beneficial to use production data in test environments to test issues found in the production system. The master test plan should describe the method to remove Personally Identifiable Information (PII) and Protected Health Information (PHI) (i.e., data de-identification) when used in lower test environments.
- Testing tools that will be used in the various environments.
- The process for identifying risks, developing mitigations, and tracking the risks.
- Testing schedule.
- Defect management process, which describes the stages a defect will go through before it is closed.
- Test metrics that are used to get reports on the progress of testing. Examples of test metrics include number of test cases executed, defects logged, features with the most bugs, requirements coverage, how long it takes defects before they are closed, etc.

Expectation 3: CMS expects states to develop and share an incident response handling plan and a contingency plan for sustaining operation of the legacy system if the testing process demonstrates that the new system exhibits untenable performance behaviors either during the system development phase or after production. The plans should include elements such as:

- A process for conducting quality audits during system development to examine qualitative and quantitative metrics that assess the system health and quality and to assess risks of missing important milestones.
- A process for conducting quality audits during system operation to examine qualitative and quantitative metrics to assess the operational system health and quality and to assess risks of continued quality issues.
- A process for triggering an escalation procedure to make decisions to either sustain or revert back to a legacy system if issues cannot be fixed or addressed with appropriate workarounds.

Test Execution

Regardless of the software development methodology used (e.g., agile, waterfall), the type of system (e.g., custom development, COTS, SaaS), or the responsibilities of the state and vendor, the state is ultimately responsible for ensuring that robust testing has been performed and the delivered system is of high quality. Disconnects between activities executed by different stakeholders can create schedule, cost, quality, and maintenance problems.

Expectation 1: CMS expects states to exercise their responsibility to assess the system functionality via testing and to develop and share clear documentation of state and vendor responsibilities regarding testing and quality. The documented process is expected to also include processes and remedies to address system defects. To meet the expectations, states should ensure certain steps are performed, such as:

- System requirements are documented in sufficient detail to allow the development and/or use of test scenarios and for the testing team to develop comprehensive test cases that handle normal and exception data, (e.g., data incorrectly formatted), or data that falls outside specified ranges.
- Test cases are linked to, and provide full coverage of, functional system requirements, including testing across multiple browsers and devices.
- Every step in a test case should have a clear indication of the expected results and pass/fail conditions.
- A thorough independent state review of the test results is performed at various stages in the development cycle.
- Pre-production system builds are available to system and business users to conduct frequent user acceptance testing.
- RCAs are conducted for defects, promoting effective re-use of lessons learned.
- Defects are consistently tracked and reported, using defined defect severity levels.
- Appropriate turnaround times for error/defect resolution are maintained.
- Test cases should include non-functional requirements, such as conducting test cases to test high availability, failover, disaster recovery, response time, data migration, and system performance under simulated peak load.
- The testing team has a clear escalation path to the appropriate stakeholders if high-severity defects are not fixed within a reasonable timeframe or if system quality issues persist throughout system development.

Expectation 2: CMS expects states to ensure the testing team includes experienced testing and quality assurance team members:

- The state and vendor test assessment team should include team members who understand the nature and purpose of the system and the kind of software involved.

- The state testing team should include members with appropriate experience and expertise to review test plans and procedures and to evaluate the functionality.
- The quality assurance team members should ensure correct system quality procedures are followed, (e.g., entry and exit criteria are followed for code to be promoted from the development environment to the system testing environment).

Expectation 3: CMS expects states to provide evidence and test results to CMS for different types of tests. Different types of testing include:

- *Unit Testing:* The developer conducts the unit test, typically on the individual modules under development. The unit test often requires simulating interfaces to other modules or systems which are the source of input data or receive the output of the module being tested but that are not yet ready to test.
- *System Integration Testing:* When a new module or a significant new layer of functionality is added to the system, a series of tests are performed to ensure that the new module or functionality operates correctly in conjunction with all pre-existing modules/functions. This type of testing is most prevalent when a new function is developed or an application programming interface (API) has been added or modified. In addition, states should include testing federal reporting requirements, such as T-MSIS, as part of this type of testing.
- *Regression Testing:* Once changes are made to a system, such as new functionality or significant modifications due to defect resolution, a series of tests must be performed to ensure that all pre-existing functionality is still operational and still passes previously executed tests.
- *User Acceptance Testing:* Once development and system testing have been completed, a “final” series of tests must be performed to ensure that the new/updated system functionality satisfies the needs of the business and system users and helps them perform their daily activities in a more streamlined manner. Accessibility testing (i.e., 508 compliance) could be incorporated as a form of end-user testing. In addition, states should include testing federal reporting requirements, such as Transformed Medicaid Statistical Information System (T-MSIS), as part of this type of testing.
- *Performance Testing:* The system must be subjected to tests that ensure that SLAs are attainable and sustainable. These tests primarily focus on ensuring that system response times and infrastructure parameters (e.g., CPU, storage, network) are within acceptable tolerances.
- *Load Testing:* These tests include employing specific tools and techniques (often automated) to apply simulated high – yet realistic – volumes of user traffic and thus stress on the underlying infrastructure components. The primary goal of this type of testing is to determine at what point the system “breaks” or response times are degraded to a point where they become intolerable. These tests provide an insight into the scalability of the system.
- *Parallel Testing:* This important type of testing is typically used when transitioning from a legacy system to a new system and it aims to find out whether the legacy system and the new system are behaving the same or differently and to ensure that the two systems produce consistent results. This type of testing should include testing of federal reporting requirements, such as T-MSIS.
- *Data Migration Testing:* Migration Testing is a verification process of migration of the legacy system to the new system with minimal disruption/downtime, with data integrity, and no loss of data, while ensuring that all the specified functional and non-functional aspects of the application are met post-migration. Migrating data from legacy systems is often difficult and risky as it requires combining data from multiple sources and databases and handling data quality and inconsistency from disparate sources.

- *Security Testing:* Security testing ensures that sensitive information, such as PII and PHI, is protected. Security testing should be performed prior to production and on an ongoing basis when the system is operational. For more details on the scope and requirements of security and privacy testing, please refer to the Streamlined Modular Certification for Medicaid Electronic Systems Guidance Document, Appendix D: Framework for the Independent Third-Party Security and Privacy Assessment Guidelines for Medicaid Enterprise Systems.
- *Usability Testing:* Usability testing is a method of testing the functionality of system and involves observing real users as they attempt to complete tasks in the system. The goal is to reveal areas of confusion and uncover opportunities to improve the overall user experience. Conducting usability testing helps improve the user experience.

Expectation 4: CMS expects states to share testing and quality metrics with CMS. Metrics include, but are not limited to:

- Percentage of requirements covered by and traced to test cases.
- Percentage of software code covered by test cases.
- Current list of defects with data, such as defect title, description, test case reference, requirements reference, severity, open date, status, etc.
- Charts or graphs showing the distribution of defects by severity level.
- Graphs showing the rate of opening and closing defects.
- Statistics showing defect age.
- Statistics for defect reopening.

Expectation 5: CMS expects states to develop a thorough deployment plan. The deployment plan should include, but not be limited to, the following components. In addition, some components below may not be needed for SaaS solutions:

- A release plan that describes the activities for a phased implementation or roll-out. The release plan may include the following activities as appropriate: preparation of the environment, conversion information, product installation information, and data migration.
- Production readiness checklist that describes a list of steps necessary to ensure the product deployment requirements are met.
- Communication plan to ensure that everyone who needs to be informed about deployment activities and results gets the needed information in a timely manner.
- Issue/change request tracking method that will be used to record project issues and decisions.
- Business continuity and disaster recovery plan that describes a business resumption plan when a catastrophic event occurs.

Expectation 6: CMS expects states to use and share collected data in an actionable manner and to hold vendors accountable for unacceptable system quality that jeopardizes important milestones and/or system users buy-in:

- A triage process to understand and analyze data collected, (e.g., defect statistics), to properly assess system quality issues.
- A process for developing RCAs with a plan to resolve severe issues.
- A process for officially informing vendors of unacceptable quality issues and requesting a plan to remedy such issues, (e.g., a cure notice).
- A process for escalating consistent issues to an identified body of decision-makers and CMS to determine the future of the system and the system development with the current vendor(s).

Operational Monitoring

Once a system goes into production, operational monitoring tracks the system “health” on an ongoing basis. Monitoring the production environment ensures that the system responds well to peak loads, users are satisfied with the system response, the system is intuitive and well-liked by users, and the system handles component failure in a graceful manner.

Expectation 1: CMS expects states to perform ongoing testing after production to validate changes to the system and to share the testing results with CMS. This is effectively a form of regression testing. Testing may include:

- A set of tests to ensure that newly added functionality works as expected.
- A process for initiating and monitoring RCAs for production defects, and for using the RCA results to enhance the quality of the software and the quality of testing in order to avoid detecting defects witnessed only in the production environment.
- A suite of regression tests to ensure newly added functionality does not break existing functionality that was already working in the production system.
- A signoff process to allow the new software to be loaded in the production system.
- A documented process to back out certain changes to the production code, if necessary.

Expectation 2: CMS expects states to develop and share a set of metrics to monitor the health of the system and escalation procedures to notify the appropriate stakeholders for certain detected conditions. Metrics may include, but are not limited to, the following items:

- Statistics regarding usage of important computing capacity, such as CPU, memory, and storage.
- System availability.
- Statistics regarding system response time for various user transactions.
- Statistics regarding a help desk or call center, if applicable.
- Statistics regarding corrective action plans and RCAs.
- Statistics regarding turnaround times to resolve production defects.

Expectation 3: CMS expects states to use and share collected data in an actionable manner and to hold vendors accountable for unacceptable system quality that jeopardizes important milestones and/or system users buy-in:

- A triage process to understand and analyze data collected from the operational system, (e.g., SLAs and defect statistics), to properly assess system quality issues.
- A process for developing RCA with a plan to resolve severe issues.
- A process for informing vendors officially of unacceptable quality issues and request for a plan to remedy such issues, (e.g., a cure notice).
- A process for escalating consistent issues to an identified body of decision-makers and CMS to determine the future of the system and make decisions regarding the operational system and the current vendor(s).

List of Recommendations

CMS recommends states also adopt the following best practices to assist in meeting CMS MES testing expectations in an efficient and effective manner.

Recommendation 1: States should define formal testing team structures that may include a combination of state and vendor staff.

- If the first level of testing is performed by the vendor testing team and the state team functions more as user acceptance and oversight team, the state should produce clear documentation of the division of responsibility and the handoff procedures between the state and vendor testing teams.

Recommendation 2: States should consider pooling test resources and knowledge with other states to increase efficiency. This may take several forms, such as:

- Making test scenarios and test cases available for sharing between cooperating states, which allows states to learn new practices that enable them to become better generators of their own new test cases.
- Making testing tool licenses available for sharing between cooperating states whenever possible to allow greater access to valuable tools that might otherwise be too expensive for them to acquire individually.
- Launching and actively participating in a states' learning consortium that consolidates lessons learned, serves as a central knowledge custodian, and provides a forum for iterative self-improvement. CMS could work with states, if interested, to test the effectiveness of this recommendation and to develop a mechanism to operationalize it.
- Exploring the use of a common testing environment to enhance states' overall testing regimens.

Recommendation 3: States should use synthetic datasets whenever possible and to the extent practical. Synthetic data is artificially generated to replicate real-world data but does not contain any identifiable information. This lowers the barrier to deploy data by removing the need for vast volumes of real data and adds security by eliminating PII/PHI.

Recommendation 4: CMS recommends the use of automated testing tools. The recommended breadth and depth of unit, functional, and regression testing cannot be effectively performed solely by manual testing. Automated testing should be implemented in a cost-effective manner, and both states and their vendor(s) should be participants in this effort. Iterative development and regression testing benefit from the speed and consistency provided by automation tools.