Critical Logic
Integrated Quality Management

Test Management Systems (TMS) and Cause-Effect Modeling (CEM)

Prepared By: Critical Logic

WHITE PAPER

_Test Management Systems (TMS) and Cause-Effect Modeling (CEM)

Many organizations employ commercial "test management" systems as a means to enhance the capability of their testing work. Products such as HP Quality Center, Compuware QA Director, and IBM Rational test manager are among the prominent examples of this technology. These test management systems can play an important role in increasing the productivity of a software testing team.

The role of the test management system changes, however, with the introduction of other testing technology and associated processes. In particular, the introduction of automated test design tools typically changes the processes and application of a test management system. This paper describes important considerations for how to integrate any test management system (TMS) with Critical Logic's Cause-Effect Modeling (CEM) Suite.

Manual Test Authoring with a Test Management System

The key difference between manual test case design and CEM test design is that with manual design, all aspects of the test case must be managed by the tester. This includes all of the following:

_***Coverage provided by the suite of test cases***

_***Coverage provided by an individual test case***

_***Description of the test case***

_***Linkage of test cases to requirements***

_***Level of detail in the test case and consistency of detail across test cases***

_***Identifying test cases that need to change when the application being tested changes***

_***Editing test cases when the application changes***

Not only is each tester responsible for accomplishing all these objectives for their areas of test responsibility, but it falls to each tester and the test organization to ensure that this work is done in a consistent fashion across the entire organization.

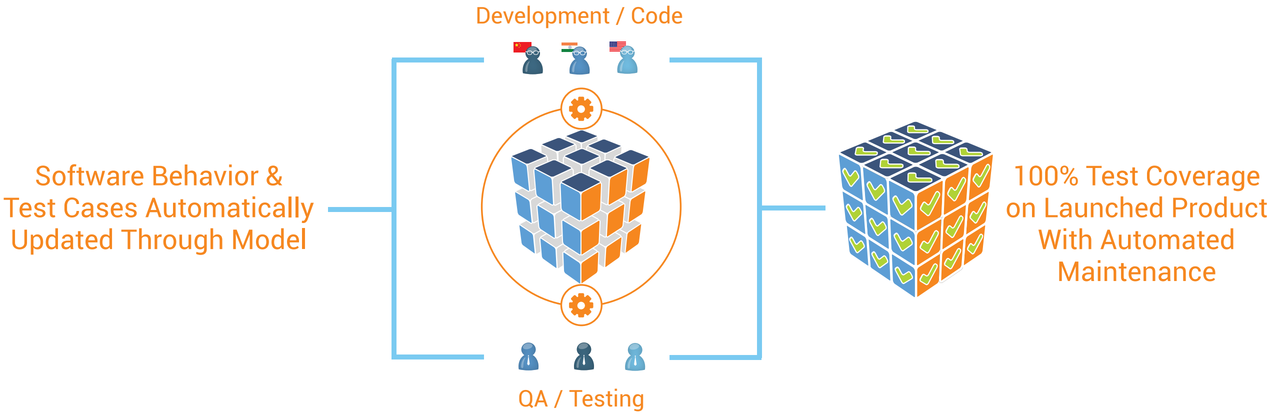
From the viewpoint of test case design, the TMS serves only as a repository for test cases and other test information, not as a test design and documentation tool. The TMS cannot ensure that a test case is correct, that it is defined at a proper level of detail, that it is not redundant, or that it provides the necessary level of coverage.

All of these issues must be implemented through the standards and guidelines created by the test organization; essentially, the TMS is agnostic as to the quality of test cases. The TMS can facilitate management and retrieval of test cases, and tracking test execution, but writing quality test cases is a manual effort.

To enforce a level of consistency and discipline when manually designing test cases, organizations implement processes and standards for writing test cases, including how to format the test cases, how to name and describe test cases and what the coverage of a test case should be. In addition, to track and measure progress in the test design phase, interim deliverables such as test plans and test summaries are required. Importantly, most of these standards and guidelines are oriented to the individual test case, not to test coverage or test suites. For example, test authors are expected to limit what each test case "tests." While this makes it easier to catalogue and select specific tests for specific purposes, it also guarantees that many more tests must be built and maintained compared to authoring tests which provide maximum (and known) coverage with each test case.

Test Authoring with CEM Suite

With CEM, virtually all the issues associated with building, maintaining and managing individual test cases are accomplished in the test model. Modeling standards and model output replace manual authoring standards and guidelines. CEM guarantees full coverage in the minimum number of test cases, a consistent format, and complete linkage between the test cases and the functionality being tested (at multiple levels). With CEM, the role of Quality Center _ or other test management systems _ changes from that of both a repository and a test authoring environment to strictly a repository of testing deliverables.


Development / Code
Software Behavior & Test Cases Automatically Updated Through Model
100% Test Coverage on Launched Product With Automated Maintenance
QA / Testing

CEM also shifts the focus from individual test case maintenance to managing the suite of test cases necessary to cover a defined scope of functionality. When representing software functionality in a model,all necessary test cases are provided and documented. Instead of building, maintaining and managing test cases manually in the TMS, this work is done automatically from the model.

Key Differences

All the information requested of the test author when manually authoring scripts is available from the automated test designs; however, in some cases, the format or the representation is slightly different. The following are the key differences:

_***Standard test format.*** *CEM Suite explicitly defines the sequences of causes and effects necessary for each test. Multiple formats are available. If the actual descriptions of causes or effects do not meet standards, they are changed once in the model and all test cases are updated and reloaded into the TMS.*

_Test coverage. *How does CEM Suite answer the question "what does this test case test?" Each test case attempts to test as many specific required variations that it can. At the same time, no test case is redundant; each test case tests unique functional variations. CEM Suite provides two-way traceability. Reports are available that show which specific business rules (and sub-conditions) are tested by each test case. Conversely, reports also can tell the tester which test cases test a specific requirement, business rule, or feature. The tester no longer has to be limited by one test case only testing one "thing."*

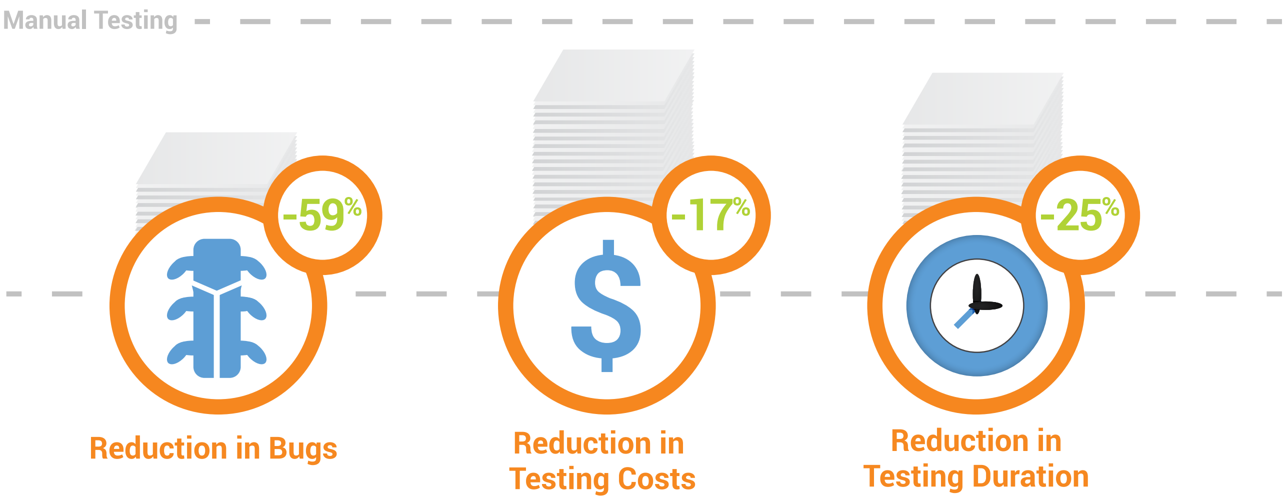
_Test case length. *The size and complexity of CEM test cases are limited and controlled by the model. Each test case represents a single traversal of the model to one or more effects. By managing the functional scope of each model, test case size and length is automatically limited.*

_Test case reusability. *Because maintaining manually authored test cases is time consuming and expensive, test steps used in multiple test scripts are separated and normalized so they only need to be changed in one place. With CEM, this normalization already exists in the models. Test step sequences are modeled once, then integrated into each test case as required during automated test case design. This means that complete test scripts are available immediately in the TMS without the need to maintain linkages and then "assemble" a test script from its reusable components.*

_Test cases and test data. *Properly designed test scripts have two separate components. One component is a description of the required conditions, actions, and expected results (the test script). The other component is test data that will be used during a specific execution of that test script. While the test script usually stays stable, the test data may change from execution to execution. CEM Suite separates these two components so that changes made to one component can be made independent of the other.*

_Test case updates. *When the functional behavior of the software changes, updates are made once in the model. Test scripts are then recomputed and re-optimized. All test documentation is automatically regenerated. Typically, the revised documentation is uploaded to the TMS to replace current documentation. Note that the suite of test scripts is typically uploaded to the TMS, not just an individual test case.*

Comparing Manual Testing with Model-Based Testing



Administrative Processes with CEM Suite

Since CEM test design generates the complete suite of test scripts for a given target set of functionality, while manual test procedures focus on building individual test cases, a few administrative processes may need revision. The following issues may need to be addressed:

Test Coverage. Because CEM test scripts are defined automatically, documenting what each test case covers is done after the test scripts have been generated, not before. This information can be captured in either of two ways:

_Have the test engineer review each CEM test case and write a summary of its coverage. This method then creates the need for manual maintenance of this information whenever the model is changed and test scripts are revised.

_Make available in Quality Center (or other test management systems) the extensive cross-referencing reports provided by CEM Suite. Anyone needing to know what a test case covers, or wishing to find a test case that tests a particular function, can go to these references.

Test script reusability. For manual testing, repeatable steps in multiple test cases are managed in the model. Testers should expect that the test script in their test management system is a complete start-to-finish description of the test. For automated test execution, there may be a desire to build one automated script and leverage it across multiple tests. This is typically done by coordination between the automation engineer and the model manager so that common automation routines are referenced in the model.

Test script selection. There is sometimes a need to select a subset of available test scripts for "smoke tests" or other limited purposes. Tests can be selected by referencing the traceability reports to find the tests meeting the specific criteria. Because both the test scripts and the reports are generated from the model, they remain consistent. The model also can identify which tests should be rerun when a specific change is made to the model. This avoids the need to rerun all the test scripts in the model when only a small change has been made.

Using CEM to Bolster Test Management Systems

Although test management systems such as HP Quality Center, Compuware QA Director and IBM Rational test manager provide a way to enhance an organization's software development's testing capabilities, new technology is creating opportunities for a more comprehensive testing experience. With Critical Logic's Cause-Effect Modeling (CEM) Suite, the need for manual testing is greatly reduced thanks to a comprehensive testing model that manages a suite of test cases needed to cover all desired functionality. When used in conjunction with an organization's existing test management system, CEM facilitates a more complete testing environment, starting with specific requirements and ensuring they are met throughout the process.

Contact Critical Logic to find out how your teams can take advantage of CEM in your agile development environment. Call Critical Logic at 415.814.9524 or visit us at www.Critical-Logic.com.