

Ten Test Strategy Components to Ensure Successful CMS Upgrade by Leveraging Risk Based Testing Approach

L&T Infotech Center of Excellence



L&T Infotech

Table of Contents

Abstract	3
Concept of Test Strategy Component (TSC)	4
1. 'Requirement-Test Data-Regression Test' TSC for Identifying Testing Boundaries	6
2. 'Discuss-Review-Test' TSC for Verifying Upgrade Strategy	7
3. 'Compliance check' TSC for Development Process	7
4. 'Test Data-Volume Testing' TSC for Ensuring Data & Digital Assets Migration Without Corruption	8
5. 'Smoke Testing' TSC for Verifying Environment Setup	9
6. 'Real Time CMS Users Tests' for Testing UI Navigation and Usability	10
7. 'Parallel Operation' TSC for Testing Globalization	10
8. 'Interface-Regression Testing' TSC for Ensuring System Interfaces	12
9. 'Response Time Monitoring' TSC for Verifying Performance	12
10. 'Formal Review-Parallel Operation' for Verifying Training Material and Documentation .	13

Abstract

A Content Management System (CMS) is a computer software system for organizing and facilitating collaborative creation and distribution of documents and other content. It could be a web application used for managing websites and web content. In many cases, a CMS requires special client software for editing and creating articles.

Global enterprises are increasingly using CMSs to maintain their electronic assets, mainly because it meets their needs:

- Creation, capture, management, security
- Storing/retaining/destroying content
- Publishing and distributing
- Searching, personalizing
- Presenting/viewing/printing digital content

Digital content can be of any type—pictures, images, text, reports, video, audio, transactional data, catalog or even code.

Content management is a critical component of Knowledge Management. Organizations are increasingly investing in the new generation of CMS applications which comes with a comprehensive set of features.

However, these systems often have to undergo major upgrades or changes due to newer technological challenges. The key driving factors for such upgrade exercises are:

- Need to integrate with other products like application servers
- Change in implementation technologies. Eg. Interwoven Team Site was built using Perl
- The newer versions also support Java

In such cases, enterprises have a high level of expectation on:

- Maximizing return on investment with minimum or zero compromise on existing system/application features
- Improved system performance
- Continuing operations with minimal additional efforts within a short time span

These expectations thus drive the need for stringent verification and validation processes during upgrade projects. This paper presents a new concept called the 'Test Strategy Component' framework and its implementation for overcoming critical challenges during CMS upgrade testing.

Concept of Test Strategy Component (TSC)

The test strategy component concept has its roots in the 'Risk Based Testing' paradigm.

All of us know that risk is a major driver for testing. An effective testing approach is one which identifies key risks for an application and helps reduce them early in the project lifecycle. All such risks become the 'Quality Goals' for the application under test.

Testers, make every effort to devise a strong test strategy to ensure that these Quality Goals are achieved in entirety. For every Quality Goal to be tested, we apply various test techniques, use diverse tools and follow different test processes.

TSC is a strategy element, which combines one or more test techniques, tools and processes to test a particular Quality Goal from all dimensions.

For example consider that we need to provide a strategy to test 'Methodology' test factor. We would use 'Compliance' as test technique. A process audit or review will be conducted using a checklist. Thus to test Methodology from all dimensions, we have to use a combination of a testing technique, test process and a test tool. This TSC framework enables the test manager to design a comprehensive test strategy for testing any application thoroughly.

This paper elaborates the framework implementation considering CMS upgrade as a case study. However, the framework can act as a leverage to serve testing needs of any application or any type of activity.

The table below lists the CMS upgrade challenges (Quality Goals) along with recommended TSCs.

1	Identifying Testing Boundaries (CMS Key Features Vs Tailored Features)	Correctness	Requirements	Test Data, Test Cases	Regression Testing
2	Verifying Update Strategy	Maintainable, Portable	Compliance, Operations	Checklist, Parallel Operations	Design(Upgrade strategy document) review, Discussions with development and content managers team, Parallel Testing
3	Compliance check for development	Methodology	Compliance	Checklists	Audits, Process reviews

	process				
4	Ensuring data & digital assets migration without corruption	Portability, File Integrity	Requirements, Control	Test Data, Checklist	Test Data Identification, Volume Testing
5	Verifying Environment Setup	Reliability, Security	Requirements, Manual Support	Fact finding, Checklist	Audit, Smoke Testing
6	Testing UI Navigation and Usability	Ease of Use	Manual Support, Requirements	Fact finding, Checklist	Audit, Smoke Testing
7	Testing Globalization	Ease of Operation	Operations, Compliance	Checklist, Parallel Operation, Fact Finding	Test Data Identification, Inspection, Regression Testing
8	Ensuring system interfaces	Coupling	Functional and Regression Testing (Inter system, Operations, Control)	Test Data, Regression Test cases	Regression Testing
9	Verifying Performance	Performance	Stress, Execution	Test Data, Fact finding	Response Time Monitoring
10	Verifying Training Material and Documentation	Ease of operation	Operations, Compliance	Check list, Parallel Operation, Fact Finding	Document review, Discussion with content manager, Parallel Testing

The following sections describe each TSC in detail.

1. 'Requirement-Test Data-Regression Test' TSC for Identifying Testing Boundaries

Identifying the demarcation between CMS features, the application accessing the CMS, and the tailored features is crucial in ensuring the upgrade 'correctness'.

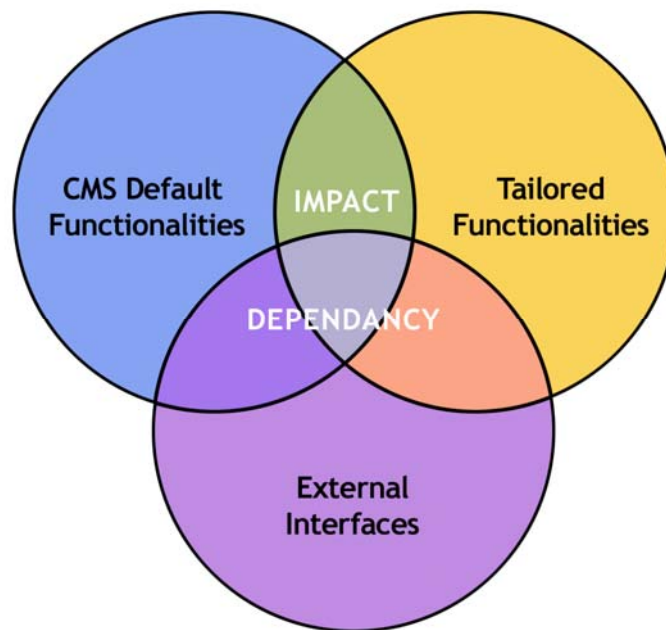


Figure 1: Content Management System Functionalities

The QA team should review and understand requirements and should try finding answers for the following questions:

SN	Check Points
1	What are the business objectives behind this upgrade activity?
2	What are the CMS-specific features and what are the application-specific features?
3	What are the limitations of the current CMS version?
4	Which are the best and most frequently used CMS features in the current implementation?
5	Which are the best and most frequently used application features?
6	Which are the best features or improvements promised by the newer CMS version?
7	What are the improvements that the newer CMS version has promised?
8	Are there any limitations, change in usage or known issues in the newer version?
9	Is there a mechanism for having a roll-back to the prior version, or other contingency plan, in case the new upgrade does not work after installation?

The answers to these questions would help in three ways:

- To identify which areas come under regression testing
- To build effective and realistic test data for the tests
- To prioritize the features to be tested thoroughly

Along with requirement documents, we also need to refer to existing test-ware and defects logs. This analysis helps during the process of defect tracking for subsequent upgrades.

2. 'Discuss-Review-Test' TSC for Verifying Upgrade Strategy

Upgrade Strategy is the most important document which needs to be complied with. Verifying the Upgrade Strategy thus becomes very important. During the upgrade document review, the test factors that we need to evaluate are 'Maintainability' and 'Portability'.

For verifying the Upgrade Strategy, 'Discuss-Review-Test' approach is highly recommended. The brainstorming sessions with the technical team along with QA and Business Users team help a lot in articulating a foolproof Upgrade Strategy. During this phase a technical team may take up a data restructuring exercise. We need to ensure that this restructuring does not have any negative impact on system usage from a business user perspective.

3. 'Compliance check' TSC for Development Process

It is important to form a basic methodology of going about the entire testing process. The QA cycle should run parallel to the development cycle. Participation of the technical team and business users in the QA activities is crucial for maximizing testing coverage.

The upgrade requirements need to be reviewed not only by the QA team but also by the technical team and the users of the system to uncover flaws in gathering and understanding the upgrade requirements. During the design process it is important to involve both the QA team and the technical team to review the Upgrade Strategy.

During the implementation phase of the CMS Upgrade, it is important that the end users of the CMS are also invited for periodic reviews. This helps uncover major usability and navigation defects early in the lifecycle. Similarly users should also be brought in to provide inputs for the training manual, to judge the clarity of the most frequently used workflows in the training material. Figure 2 below depicts the involvement of various groups during

different phases of the CMS Upgrade Process and their roles.

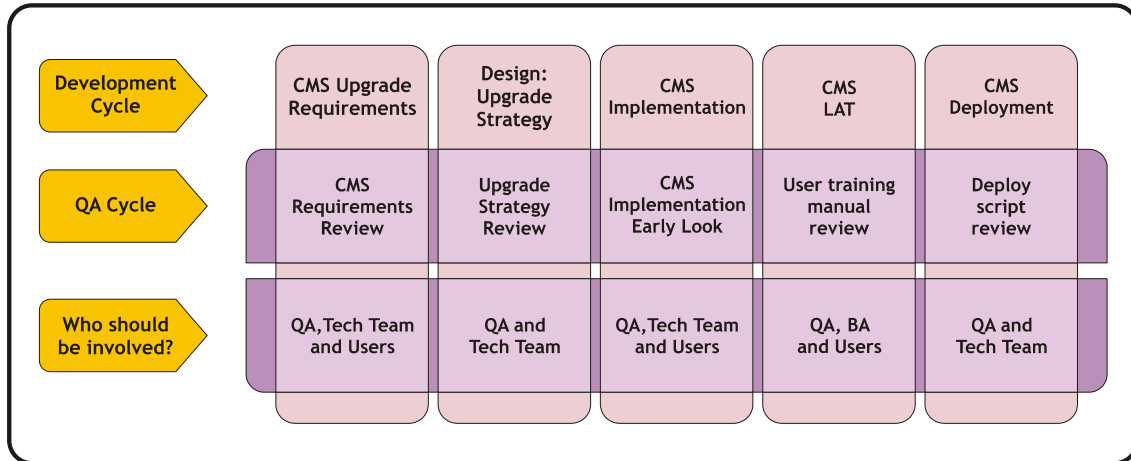


Figure 2: Involvement of various groups during different phases of CMS Upgrade process

4. 'Test Data-Volume Testing' TSC for Ensuring Data & Digital Assets Migration Without Corruption

For a CMS, especially during system upgrades, ensuring data and digital assets migration without corruption is a major challenge. We need to test the 'portability' and 'integrity' of the entire system. As illustrated in the figure below, majority of the defects identified during the upgrade process are related to data corruption.

Defect Categories

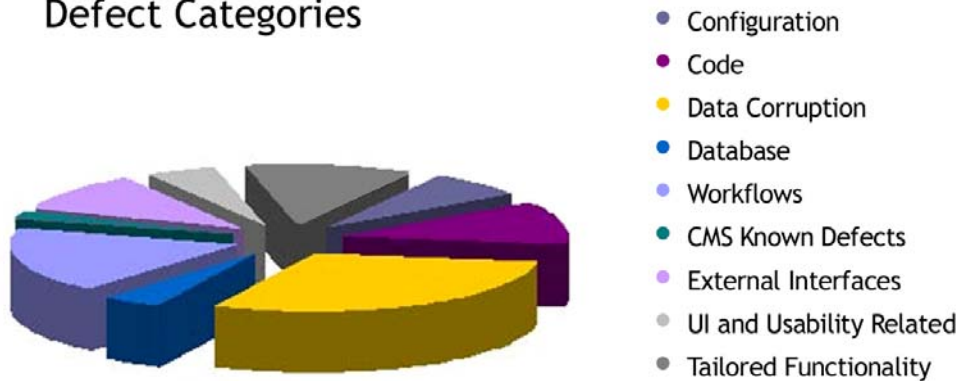


Figure 3: Break-up of defects by defects category

This high number of data corruption defects necessitates complete data integrity testing for the entire system. Hence creation of test data that covers majority of the content types in the CMS is extremely important. The Upgrade requirements document acts as a vital input for this activity. Checklists play a vital role in ensuring that all types of content are included in the test data.

However, only test data creation is not sufficient. It is also necessary to run a huge number of tests to ensure complete test coverage. Unless the entire content already supported by the previous version of the CMS is tested, the data integrity test remains incomplete. Volume testing backed up by efficient test automation scripts is the only solution to this challenge.

5. 'Smoke Testing' TSC for Verifying Environment Setup

Verifying adequate environment setup is a key challenge for testing the upgrades. If different environments are used for different phases like QA, User Acceptance testing and testing in production environments, it becomes necessary to ensure that appropriate access rights are set up on every environment.

Frequent environment configuration changes may lead to an unstable environment in terms of access to the CMS through various roles and to various content types. This may also prove to be a security threat in cases when the roles and their access rights are not well-defined. Hence, 'Security' and 'Reliability' factors need to be thoroughly tested before initiating the detailed testing process for the new version.

Smoke testing proves to be an effective method to make an 'environment readiness check'.

The table below lists sample Smoke test checkpoints for CMS upgrades:

SN	Check Points
Level 1	
1	Verify log-in access as "Author"
2	Verify log-in access as "Editor"
3	Verify log-in access as "Content Manager"
4	Verify access to all Work areas
5	Verify Visual format fields
6	Verify Custom unlock DCR workflow
7	Verify Search DCR
Level 2	
1	Verify simple content publishing
2	Verify that editor is able to publish newly created content using custom workflow
3	Verify Workflow e-mail sent to content manager
4	Verify e-mail contains valid link for content
5	Verify content is reflected on the site

Having well-defined configuration processes also helps reducing the risk of an unstable environment setup.

6. 'Real Time CMS Users Tests' for Testing UI Navigation and Usability

The challenge here is not only to present a satisfactory user interface for the new version of the CMS, but also to make sure that the new version supports all the crucial usability features of the previous version. Comparisons are bound to be made, as the same users using the older version would now be using this new version of CMS. In such a scenario, Backward Compatibility testing plays a vital role. It is important to ensure that even if the usability is a little different from the previous version of CMS, the workflows are not difficult to understand and access. In other words, 'Ease of Use' is to be ensured while testing for UI Navigation and Usability. For instance, the way individual task items are accessed, or the way user-help appears on the CMS User Interface should be similar to the previous version, to ensure customer satisfaction. Following seven pointers act as important checkpoints to ensure that the Usability and Navigation Testing is on the correct track:

SN	Usability Factor	Examples
1	Appropriate feedback	Error messages
2	User Interaction	Help and Instructions
3	Multiple Input Paths	Various Navigations
4	Constraints	File paths, Validations on each field
5	Mappings	Shortcut keys to access the activity
6	Consistency	Layout, design of interface and workflows, colors and fonts
7	User interface standards	Following Style guides, Navigations and layouts

7. 'Parallel Operation' TSC for Testing Globalization

In case of enterprises having a global presence, the external interface may support multiple locales and multiple languages. Customized content is created in CMS for each language. It is quite possible that there are different ways the users would want to use CMS. The QA team in this case should concentrate on testing 'Ease of Operation' for various diverse users of the CMS.

A key challenge in testing multi-lingual CMS implementations is the language understanding constraints of testing resources. It is important to create a test environment for this content, to understand the impact of change in content on the external interface (application) and to understand the business need for maximum scope and coverage for testing.

The following steps should be taken to initiate the tests for Globalization:

- Creating a structure of all the content types in CMS and also the languages available for them
- Identifying business critical content that is frequently created, updated or accessed

- Deciding the type and method of testing that is possible on globalized content
- Reusing the existing methods and contents for testing

For testing multi-lingual content the following steps should be followed:

- Conducting tests with the users having domain and language experience
- Taking sessions with the users to understand the importance of the content and the expected results
- Finding out the similarities in functionalities and workflows for English and other languages and then deciding the test strategy
- Adding the English identification text in each of the fields for clear understanding and identification
- Preparing test data for target languages from the existing content
- Mapping the contents to the English contents in terms of location on the page

Language Contents

SN	Content Type	Language	Path	Content copied from	New Content nomenclatures	Changes in the fields with specific Nomenclatures	Status
							Saved
							Deployed
							Submitted
							Verified

For testing multi-location content or diverse workflows the following steps should be considered:

- Map all existing functionalities and flows that are possible by the QA tester and assign the remaining for UAT real users of the system to test. Convey all these in detail.
- Usability is one of the major factors to be considered while testing the globalized content.
- Creating the same environment for testing.

8. 'Interface-Regression Testing' TSC for Ensuring System Interfaces

During an upgrade, the underlying system architecture may undergo changes. Ensure that the upgrade does not have impact on other system components such as mail interface to workflows, data publishing mechanisms, etc. A complete regression cycle is important to ensure perfect 'coupling' between various systems with which the CMS communicates. The test cases for Interface-Regression testing should be built during Upgrade Strategy design and review phase.

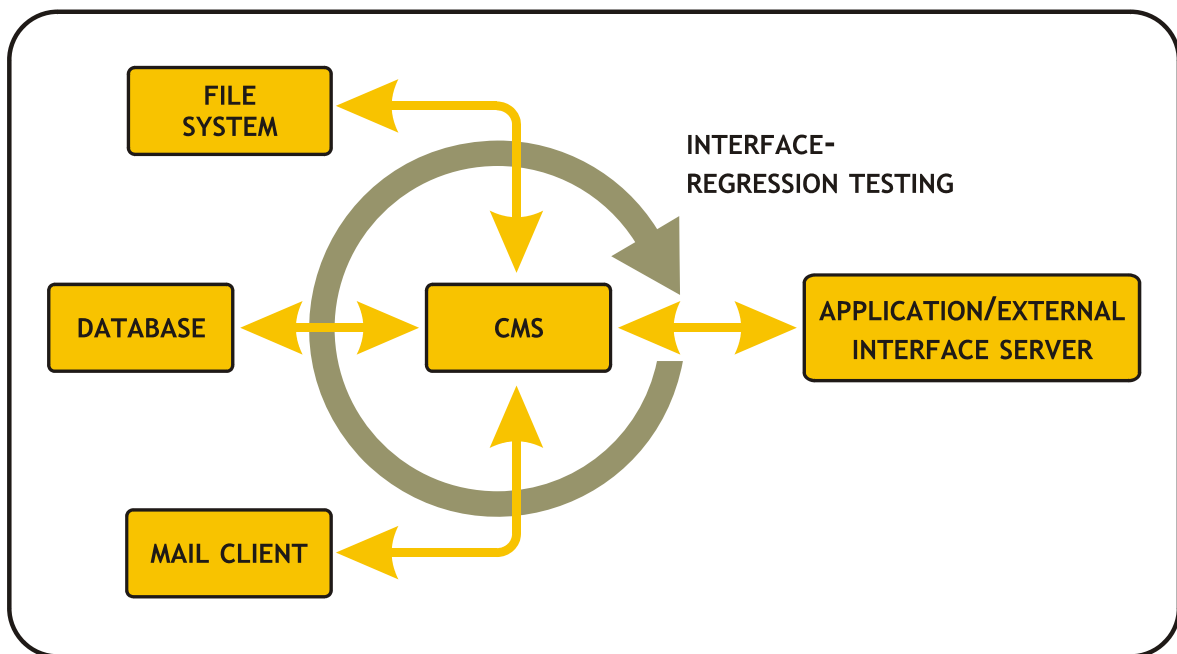


Figure 4: Interface Regression Testing for CMS Upgrades

9. 'Response Time Monitoring' TSC for Verifying Performance

The newer version of CMS may have some performance issues due to implementation limitations. This normally hampers the performance of applications or CMS features that are frequently used. During testing, identify the most frequently used CMS features like creating, saving or previewing contents. These are the main features with which the user interacts. For example while creating new content, if application takes a lot of time to load the initial controls, then it can in turn hamper content creation activity. Hence, it becomes important to test the performance of the new version of CMS thoroughly.

The test strategy component to be used here is 'Execution-Fact Finding-Response time monitoring'.

Identify the most frequently used CMS tasks like opening a new content data record or an existing content file or saving content. Identify the most frequently used content work areas and content types. Execute the same task on old CMS version as well as on the new version. Note down all the responses in a response time monitor log. Use the same log to monitor response time for other environments as well. Once these readings are taken, analyze the log and come up with any major areas where response time is comparatively very high for the newer version. Development team then should analyze this and take corrective actions to reduce the response time.

Response Time for: Opening a DCR							
SN	Work Area	DCR Name	Type of DCR (New/ Existing)	CMS Old Version		CMS New Version	
				On Staging Server	On Production Server	On Staging	On UAT
1	Work Area 1	DCR1	Existing				
2	Work Area 2	New DCR	New				
3							
4							
5							
6							

10. 'Formal Review-Parallel Operation' for Verifying Training Material and Documentation

The QA team can contribute heavily in designing the user training material. The users normally will refer to this user guide to get an exact picture of differences in the old and new version of CMS. A checklist should be built up for validating 'Ease of Operation' of the training material and documentation.

The following checklist can be used as an "Acceptance" checklist while reviewing user-training material.

SN	Check Points
1	Does document include both old and new ways of creating, editing, saving and deploying content?
2	If CMS log in process has changed, then does documentation elaborate changed process?
3	If there are any changes in the process of saving or previewing contents, then does the user manual describe it along with appropriate images/screen shots?

4	If there is any content or data restructuring, does documentation list both the old and new location?
---	---

We also recommend an operational test. During this test, members of the QA team should act as 'actual CMS users who will follow instructions given in the user guide to complete the routine CMS tasks.

References

Copyright for this white paper belongs to QAI.

http://www.steptwo.com.au/papers/kmc_evaluate/

http://www.steptwo.com.au/papers/cmb_usability/index.html

<http://www.aurigalogic.com/auriga/avs/home/solutions/ecm/architecture.html>

Author's Biography

Ashwini Barve

Current Position / Core Responsibilities

Working as Test Manager

Responsible for managing an independent QA and Testing practice within SBU

Previous Positions / Core Responsibilities

Worked as Project Leader for application development and maintenance projects

College Attended, Degree

Walchand College of Engineering, Sangli, B.E. (Computer Science)

Professional Associations

Member of SPIN, Pune; Registered member of PMA, Delhi

Certifications

Certified Software Test Engineer (CSTE #6320), Qualified Project Management Professional (QPMP)

Total years of experience

Total 9 years of experience in IT. 3+ years in QA and Testing

Amit R. Patil

Current Position / Core Responsibilities

Working as Test Lead responsible for QA activities in a project

Previous Positions / Core Responsibilities

Worked as Tester for application maintenance projects

College Attended, Degree

Bharati Vidyapith College of Engineering, Pune, B.E. (Chemical)

Total years of experience

Total 5 years of experience in IT. 1+ years in QA and Testing



About L&T Infotech

L&T Infotech offers domain-specific IT services and solutions. We are sharply focused on a set of industries that include manufacturing, banking and financial services, insurance, energy and petrochemicals and telecom (product engineering services). We are part of Larsen & Toubro, a multi-billion dollar engineering and construction conglomerate. We leverage this unique business-to-IT connect to offer the winning edge to our clients.

For more information, visit us at www.lntinfotech.com or email us at info@lntinfotech.com.