Table of Contents

1. Introduction

This section presents the benefits of a Configuration Management System and information supporting the development of this document, including the audience, related documents, and a glossary of terminology used.

1.1 Executive Summary

The primary benefits of Software Configuration Management are Quality Assurance, Change Tracking, and Business Continuity. These are achieved by associating software changes with new or updated requirements or application problems, maintained in a robust system, with procedural tracking and authorization.

Quality Assurance is provided by controlling access to the application, data, source code, and other configurable items and allowing change only through the processes defined in configuration management. No work is performed on an application without approval and no changes are exposed to the client without rigorous testing and approvals.

Traceability is recorded in the Configuration Management (CM) System that maintains the Requirements, Change Requests, and Problem Reports, allowing the review of the What, When, Why, and Who of application changes. The CM system is linked to the Version Control System (VCS) and provides the association between the change documentation and the modifications in the source code. This linkage can be through systems integration or manual association.

Business Continuity, related to application availability, is the feature of the VCS that maintains the version history of the applications, data, source code, and other configurable items and the VCS associated data in the configuration management system. Should one, some, or all systems associated with an application fail, the information necessary to restore those systems to full functionality is available and allows them to be quickly and completely restored.

This system, called Configuration Management, allows the clients, management, budget, and developers to interact in a controlled fashion to produce positive results for all involved.

1.2 Purpose of This Document

This document translates the project requirements for the Control of Application Development and Release into the Configuration Management Plan. It presents the processes, environments, and responsibilities necessary for Software Configuration Management.

Note: This is a living document. During this time at The Company, the people and systems in place are in a state of transition, forcing the need for changes in this Configuration Management Plan to consider possible changes and to present the best fit for the environment.

1.3 Intended Audience for this Document

The audience for this document and their respective uses of the document include:

2. Configuration Management Approach

This Approach presents How Configuration Management fits into the software development/test/release process, Who is involved and How they are involved, and What is required in the application environments and supporting systems to take advantage of the benefits of Configuration Management.

2.1 Organization

The image on the right side of this page depicts the involvement of Configuration Management in the Development process.

As Requirements are collected, they are re-written as discretely testable requirements and stored in a CM System as objects or Tickets.

The various design documents are stored in a CM System and revised as necessary.

The Version Control System (VCS), Serena VM, is configured for the development structure as designed and the users and groups are authorized for proper access to the Configuration Items (CIs).

Developers, DBAs, and Administrators are assigned their requirements or change Tickets, generate the code, scripts, and other CIs, and build the product system. The CIs are then added into the VCS.

The CIs are Unit Tested in the Development Environment. The results of the testing are provided to Project Management for determination. If successful, their status is updated to "FIXED". If unsuccessful, their status is updated to "RE-OPENED" and returned to development for correction.

The Test Build Team prepares a build for the Test Environment and installs it. Functional, Regression, and System Integration Testing is performed by SQA and the results provided to the CAB. If the testing is successful, the Tickets are updated to "VERIFIED" status and approved for the UAT Environment. If unsuccessful, the Tickets are changed to "RE-OPENED" status then returned to development for correction.