CATALYST
CONSULTING GROUP, INC.

Deliverable: 1.4 Software Version Control and System Configuration Management Plan

VoteCal Statewide Voter Registration System Project

State of California, Secretary of State (SOS)

October 19, 2009
Version: 2.1

Work Product Acceptance Form

Catalyst Consulting Group is pleased to present the following VoteCal Project work product/deliverable. This work product is now complete and is ready for the Secretary of State to review and approve.

Work Product:	Deliverable 1.4 Software Version Control and System Configuration Management Plan
SOW Reference #:	Attachment 1 Statement of Work Exhibit 2 VoteCal System Task and Deliverables
Delivery Date:	October 19, 2009

Secretary of State

By: Mary Winkley

Date: 10/21/09

Name: Mary Winkley (SOS Project Director)

Authors

This document was prepared by:

Contributors		
Don Westfall Catalyst Consulting Group 211 West Wacker Drive Suite 450 Chicago, IL 60606	Dave Puplava Catalyst Consulting Group 211 West Wacker Drive Suite 450 Chicago, IL 60606	Tim Smith Catalyst Consulting Group 211 West Wacker Drive Suite 450 Chicago, IL 60606

Date	Document Version	Document Revision Description	Revision Author
September 28, 2009	0.1	Initial Draft	Don Westfall
September 28, 2009	0.2	Draft Revisions	Tim Smith
September 29, 2009	0.3	Draft Revisions	Dave Puplava
September 29, 2009	1.0	Draft Submitted to SOS	Don Westfall
October 6, 2009	*2.0	Submitted to SOS (*Reversioned)	Don Westfall
October 6, 2009	*3.0	Submitted with Deliverable to SOS (*Reversioned)	Don Westfall
October 19, 2009	*3.1	Made wording corrections (*Reversioned)	Don Westfall
October 19, 2009	2.1	Submitted to SOS	Don Westfall

Table of Contents

1 Introduction

This document is Deliverable 1.4, the VoteCal Software Version Control and Configuration Management Plan. It has been developed to the specifications presented in Deliverable Expectation Document (DED) 1.4, VoteCal Software Version Control and Configuration Management Plan and as reviewed by the Secretary of State (SOS).

For the purposes of this document, the VoteCal Software Version Control and Configuration Management Plan (Deliverable 1.4) will be referenced as the Configuration Management Plan.

The following is a list of some commonly used terms to provide a consistent definition of terms used throughout this document.

A list of commonly used terms and acronyms are found in the VoteCal Project Glossary. The Glossary is maintained on the VoteCal SharePoint repository (see the Document Management Plan in this document for more information on the project SharePoint repository maintained by Catalyst).

Configuration Items are managed through the following processes:

- Configuration identification
- Configuration control
- Configuration status accounting
- Configuration audits

Each of these processes is described in this document. This document actually identifies the configuration items for the VoteCal Project.

The work products placed under configuration management include work products delivered to the customer (i.e., project deliverables), designated internal work products, acquired products, tools, and other items used to create and describe these work products.

Configuration Management includes a variety of tools, depending on the type of configuration item. These tools for the VoteCal project include: SharePoint, RequisitePro, JIRA, and Subversion.

This Plan addresses the processes of how Catalyst will maintain software version control and approach release management. It also provides the Catalyst approach to hardware configuration management.

1.1 Purpose and Objectives

The purpose of the VoteCal Configuration Management Plan is to establish and maintain the integrity of work products (documents and software) using configuration identification, configuration control, configuration status accounting, and configuration audits.

Configuration Management is a discipline applying technical and administrative surveillance to:

- Identify and document the functional and physical characteristics of a configuration item
- Control changes to those characteristics
- Record and report change processing and implementation status
- Verify compliance with specific requirements

It is important to maintain the integrity and traceability of work products produced throughout the lifecycle of a project. The configuration management process involves:

- Identifying the configuration of selected work products that compose the baselines at any given time.
- Controlling changes to configuration items
- Maintaining the integrity of the baselines
- Providing accurate status and current configuration data to project stakeholders

1.2 Scope

The VoteCal Configuration Management Plan addresses all configuration items: schedule, deliverables, other document-based work products, software, hardware, and release management.

All configuration items that have a baseline are generally covered under this Configuration Management Plan; however the detailed processes for configuration items other than VoteCal software and hardware infrastructure are presented in other project management plans as described in Section 3.1 of this document.

1.3 Standards

The configuration management processes and procedures are based on the following standards:

- Institute of Electrical and Electronics Engineers (IEEE) Standard for Software Configuration Management Plans, 2005 Revision (IEEE Standard 828-2005)
- Capability Maturity Model Integration for Development Version 1.2 (CMMI for Development v1.2), Configuration Management Process Area

SOS has adopted the state's (previously Department of Finance's) Project Management Methodology as its standard, as was described in Section 200 of the Statewide Information Management Manual in March 2006 when the project was approved. The methodology also reflects industry-standard processes described in the Project Management Body of Knowledge (PMBOK).

1.4 Assumptions, Dependencies, and Constraints

The Software Version Control and System Configuration Management Plan is based on the following assumptions, dependencies, or constraints:

- This document has identified all the items that must be placed under configuration management. If subsequent configuration items are identified, they will be placed under appropriate configuration
- Configuration items will have a baseline requiring that changes involve release of the baseline, making changes, and creating a new baseline, and that these changes will be approved through the processes defined in the Change Control Plan
- Project team members will adhere to the configuration management processes; in particular, that VoteCal developers will adhere to the software version control process.

1.5 Document Control

Configuration management is a dynamic process that occurs throughout a project's life cycle. Accordingly, at a minimum, the Configuration Management process will be reviewed at the end of each project phase, and the Configuration Management plan will be updated as required.

This document contains a revision history log. When changes occur, the version number will be incremented and the date, name of the person authoring the change, and a description of the change will be recorded in the revision history log of the document.

As with other work products of the VoteCal project, the approved Configuration Management Plan will be placed under configuration management in accordance with the Document Management Plan (a subset of the Project Management Plan). Also, in accordance with the Document Management Plan, the Configuration Management Plan will be stored on the VoteCal Project SharePoint repository maintained by Catalyst and available to the Project Team, the Independent Project Oversight Consultant (IPOC), Independent Verification and Validation (IV&V) vendor, and SOS senior management.

2 Roles and Responsibilities

The following roles and responsibilities have been identified for the configuration management process.

2.1 Catalyst Quality Manager

The Catalyst Quality Manager will be responsible for conducting monthly process reviews on all VoteCal processes, including Configuration Management, to verify adherence to the processes by the Catalyst Project Team. In addition, the Catalyst Quality Manager will be responsible for conducting quarterly configuration audits. The Quality Manager has the responsibility to report discrepancies found during the audits to the Catalyst Project Manager and creating and assigning an action item.

2.2 Catalyst Document Manager

The Catalyst Document Manager will be responsible for acquiring document-based configuration items and storing those items in the SharePoint repository. In coordination with the Catalyst Project Manager, the Catalyst Document Manager will also create and release (if needed) the baseline versions of these configuration items. Although the Catalyst Document Manager will not necessarily be the author of a document-based configuration item, this individual will be responsible for creating the physical baseline (e.g., PDF version).

2.3 Catalyst Requirements Manager

The Catalyst Requirements Manager will be responsible for maintaining the integrity of requirements in the RequisitePro repository. The Catalyst Requirements Manager will also be responsible for maintaining the integrity of the requirements traceability throughout the VoteCal project lifecycle. The Catalyst Requirements Manager will have the responsibility for creating and releasing (if needed) baselines.

2.4 Catalyst Infrastructure Lead

The Catalyst Infrastructure Lead will have the responsibility for maintaining the systems and network hardware configuration list.

2.5 Catalyst Team Leads

The Catalyst Team Leads and particularly the development leads will be responsible for maintaining the discipline of team members in adhering to the software versions control requirements.

2.6 Catalyst Project Manager

The Catalyst Project Manager will maintain overall responsibility for adherence to configuration management policies for the project. The Catalyst Project Manager will be responsible for the identification of configuration items and will work with the SOS Project Manager to determine the baseline for each configuration item.

2.7 SOS Project Manager

The SOS Project Manager will be responsible for acquiring approval for configuration items which establishes the baseline.

3 Configuration Management

The configuration management process involves configuration identification, configuration control, configuration status accounting, and configuration audits.

3.1 Configuration Identification

Configuration identification is the selection, creation, and specification of items that need to be placed under configuration control. In the VoteCal project, the following table presents the items that will be placed under configuration control, the baseline for the configuration item, the repository in which it will be stored, and the configuration management process for the item. Please note that the repositories listed here are those VoteCal Project repositories that will be maintained by Catalyst.

Table 3-1 VoteCal Configuration Items

Configuration Item	Baseline	Repository	Process Description
VoteCal Project Schedule	Approved project activity list and schedule. File type: Microsoft Project 2007 with baseline.	SharePoint 2007	Schedule Management Plan (part of the Project Management Plan and Schedule)
VoteCal Project Deliverable Products (Document-based Deliverables; numbered deliverables in the SOW)	Approved final deliverables and approved updates to such deliverables. File type: Portable Document Format (PDF)	SharePoint 2007	Document Management Plan (part of the Project Management Plan and Schedule)
VoteCal Document-Based Work Products – all document based work products not considered deliverables (e.g., use cases, test cases, test scenarios, etc.)	Approved final versions of work products that are determined to be configuration items by agreement between Catalyst and SOS. File type: PDF	SharePoint 2007	Document Management Plan.

Configuration Item	Baseline	Repository	Process Description
VoteCal System Requirements (business and technical requirements)	Validated requirements and traceability updates to requirements.	RequisitePro 7.1	Requirements Traceability and Gap Analysis Plan.
Software Code (VoteCal Source Code; test scripts, etc.)	All project source code	Subversion 1.6.5	Software Version Control and System Configuration Management Plan
Hardware	Approved final versions of system and network device configurations both logical and physical. File Type: Visio diagrams, Excel spreadsheets, device configuration text files, etc.	SharePoint 2007	System Configuration Management Plan

A configuration item that has a baseline can only be changed or updated in response to an approved change request. The process for submitting, analyzing, reviewing, and approving change requests is presented in the VoteCal System Change Control Plan.

The baselines will be established based on the following events:

- Project Schedule – Completion of the agreed upon integrated schedule between Catalyst and SOS.
- Deliverables – Approval of deliverable products by SOS.
- Requirements – Approval of the Requirements Validation

There may be work products without a baseline that the Catalyst and SOS Project Managers may determine to be configuration items. For example, interim work products supporting a VoteCal Project deliverable may be considered a configuration item. While Table 3-1 is considered an exhaustive list, there may be items that SOS and Catalyst determine should be configuration items, in which case Table 3-1 would be updated and an updated version of this plan would be created.

3.2 Configuration Control

Changes to configuration items that have a baseline will only be made in accordance with the approved change control process as defined in the VoteCal System Change Control Plan (Deliverable 1.6).

3.3 Configuration Status Accounting

The nature of configuration status accounting depends on the configuration repository being used. A Catalyst team member has been assigned to each configuration management repository as described in Section 2 (Roles and Responsibilities) of this document. That team member will have the responsibility to serve as a “gatekeeper” for that repository, making sure that only appropriate

configuration items are included in the repository. The Catalyst team member will also have the responsibility for creating and releasing baselines within that repository.

3.4 Configuration Audits

A key element of configuration status accounting is the performance of quarterly configuration audits. Configuration management audits are conducted to confirm that configuration management records and configuration items are complete, consistent, and accurate. The Catalyst Quality Manager will be responsible for conducting periodic audits on each of the configuration management repositories. Any discrepancies will be reported to the Catalyst Project Manager and an action item will be created and assigned.

4 Software Version Control

Catalyst will use Subversion as the software version control tool. This section describes the approach to using Subversion for source code version control. It describes a developer's basic work cycle, conflict resolution, and versioning process.

4.1 Developer's Work Cycle

Most software version control systems follow a *lock-modify-unlock* model where a developer checks-out/checks-in source code file edits. This model sometimes leads to problems such as unnecessary serialization of work or circular dependence. In contrast, Subversion uses a *copy-modify-merge* where a developer updates/commits source code file edits. This model seeks to solve some of the problems associated with the former model by changing the basic work cycle for developers.

A developer's basic work cycle begins with the *working copy*. A working copy is a copy of the versioned source code files from the Subversion repository on the developer's local machine.

The developer then edits the source files as needed and *commits* his or her changes to the repository; i.e. makes those changes available for other developers to update their local working copy.

Other developers on the team can then periodically *update* their working copies to *merge* these committed changes with their local changes

This process generally works very well in practice. There are cases, however, when two developers try to commit edits to the same line of code, in which case Subversion is unable to automatically merge changes. This condition is called a *conflict*.

4.2 Conflict Resolution

In the event of a conflict, it is up to the developer to resolve the conflict by collaborating with the other developer or developers to determine which changes should merged with his or her working copy and then committed to the Subversion repository.

Subversion allows the developer to decide which change is the most current or in what order both changes should be merged manually together.

Once resolved, the developer commits the manually merged changes back to the subversion repository for the other members of the development team to update their working copies.

4.3 Defect Resolution across Environments

It is possible to identify a defect in staging or production environments, but that the development environment has changed and cannot be synchronized with the staging or production environment. Through the use of release tagging and the branching capabilities of Subversion, the code can be reconciled across environments to resolve defects across those environments.

4.4 Versioning Process

Subversion is a complete versioning file system that versions directories as well as individual files. This allows copying of entire directory structures. Most Subversion projects follow the pattern of *branching* and *tagging* particular copies the source tree for various release versions. Branching and tagging is a shallow copy of the primary source code tree (referred to as the *trunk*). This shallow copying allows the development team to make as many copies as necessary without needing to worry about consuming substantial disk space.

The VoteCal Development Team shall version VoteCal software components by tagging copies of the source with a particular release number. By maintaining release copies of the source code, the VoteCal Development Team can continue development on the main copy of the source while supporting the release copy of the source (e.g. bug fixes).

Catalyst uses a *Major/Minor* version-numbering scheme, a simplified form of the *Major/Minor/Build/Revision* scheme recommended by Microsoft for software that targets the .NET Framework.

5 Release Management

Releasing of the VoteCal system to production will require efficient release management processes to establish a successful deployment. The release management process involves release planning, release staging, “Day 1” testing, and go-live.

5.1 Release Planning and Requirements Gathering

Prior to each of the VoteCal deployment phases (Pilot Deployment and Testing Phase and the Deployment and Cutover Phase), the Catalyst Implementation Team in coordination with SOS counterparts will identify the release plan schedule for all components contained in the release. The source code is then tagged with the release number in the Subversion repository and built to produce the deployable modules of the system.

5.2 Release Staging and Testing

Once all of the requirements, dependencies and planning specifics are gathered for a particular release, the code is packaged, compiled into working subsections and the release “build” is created. The code associated with that release “build” is placed in the pre-production staging environment for acceptance testing. The Catalyst test team will conduct a regression test on the release to make sure it is ready for production. The Catalyst Implementation Team will also conduct a “Day 1” simulation in which the VoteCal system is tested through the expected “Day 1” production activities. County user activities are simulated through the Model EMS.

5.3 Go Live (Deployment)

The Catalyst Implementation Team will conduct a Go/No Go meeting with their SOS counterparts. During this meeting the results of the final regression test and the "Day 1" simulation are reviewed. If the combined implementation team determines that the VoteCal application is ready for production, the implementation team makes a formal "Go" decision. On the Go decision, the VoteCal code is migrated to the production environment and the implementation team begins to convert counties to the live system.

If the Go/No Go meeting results in a No Go decision, the rational for that decision is identified. If the reason is external to the project (e.g., lack of readiness by EMS vendors or other state agencies), the project team develops a new deployment plan and schedule. If the No Go decision is based on defects in the VoteCal solution, the defects are documented in JIRA and the defects are corrected, along with a new deployment plan and schedule.

6 Hardware Configuration Management

The interdependent processes of hardware configuration management will allow for the creation and maintenance of an up-to-date record of all VoteCal system infrastructure components, including related documentation.

Items specific to the hardware configuration management strategy will be recorded via the use of configuration items (CI).

All VoteCal specific CI's that are inventoried via the hardware configuration management plan will be documented to account for the following attributes:

- Technical - Data will be recorded that describes the CI's capabilities which include software version and model numbers, hardware and manufacturer specifications and other technical details like networking speeds, IP addresses and storage specifics such as size or speed.
- Ownership – Data will be recorded to describe ownership attributes such as purchase date, warranty specifics, installed location and party responsible for support of the item.
- Relationship - The relationships between hardware items, software items, and users/departments.

The associated detail recorded via the hardware configuration management plan will be implemented through the use of various document types/formats such as Visio diagrams, Excel spreadsheets, device configuration text files, etc. These documents will be stored in the VoteCal SharePoint document repository for reference by VoteCal Project Team members.

SOS will review and approve the initial inventory and will review and approve any changes to it.

Data recorded through the hardware configuration management processes will not only be used to document business and technical details of the VoteCal infrastructure components, but will also be used to assist in managing VoteCal infrastructure components. Information recorded will be used to assist with the following infrastructure tasks:

1. Planning
2. Identification
3. Control (Change)

4. Monitoring
5. Verification