

**El Paso MPO Management System  
Software Configuration Management Plan**

**Version <1.0>  
2/1/2016**

## Document Control

### Approval

The Guidance Team and the customer shall approve this document.

### Document Change Control

Initial Release:	1/28/2016 V0.1
Current Release:	2/1/2016 V1.0
Indicator of Last Page in Document:	^_^
Date of Last Review:	1/31/2016
Date of Next Review:	TBA
Target Date for Next Update:	TBA

### Distribution List

This following list of people shall receive a copy of this document every time a new version of this document becomes available:

Guidance Team Members:

Dr. Salamah Salamah

Customer:

Mr. Roger Williams

Software Team Members:

Camel Heydarian

Christian Hughes

Chelsey Jurado

Jordan Cox

### Change Summary

The following table details changes made between versions of this document

Version	Date	Modifier	Description
0.1	1/28/2016	ZombieBytes	Initial rough draft of the introduction.
0.2	1/29/2016	Chelsey Jurado & Carmel Heydarian	Completed and edit the introduction
0.3	1/30/2016	ZombieBytes	Initial rough draft of section 2.1
0.4	1/30/2016	Jordan Cox	Initial rough draft of section 3.2
0.5	1/30/2016	ZombieBytes	Complete and edit section 2.1
0.6	1/31/2016	Chelsey Jurado & Carmel Heydarian	Initial rough draft of section 2.2
0.7	1/31/2016	ZombieBytes	Complete and edit section 2.2
0.8	1/31/2016	Christian Hughes & Jordan Cox	Initial rough draft of section 3.1 and 3.3
0.9	1/31/2016	ZombieBytes	Complete and edit section 3.1 and 3.3
0.10	1/31/2016	ZombieBytes	Initial rough draft of section 4
0.11	1/31/2016	ZombieBytes	Complete and edit section 4
1.0	2/1/2016	ZombieBytes	Finalize the document for submission.

SCM	ZombieBytes	Date 2/1/2016 9:05 PM	Page ii
-----	-------------	--------------------------	------------

## TABLE OF CONTENTS

<b>DOCUMENT CONTROL .....</b>	<b>II</b>
<b>APPROVAL .....</b>	<b>II</b>
<b>DOCUMENT CHANGE CONTROL .....</b>	<b>II</b>
<b>DISTRIBUTION LIST .....</b>	<b>II</b>
<b>CHANGE SUMMARY .....</b>	<b>II</b>
<b>1. INTRODUCTION.....</b>	<b>1</b>
<b>1.1. PROJECT DESCRIPTION.....</b>	<b>1</b>
<b>1.2. PURPOSE AND INTENDED AUDIENCE.....</b>	<b>1</b>
<b>1.3. OVERVIEW .....</b>	<b>1</b>
<b>1.4. REFERENCES .....</b>	<b>1</b>
<b>2. SOFTWARE CONFIGURATION IDENTIFICATION.....</b>	<b>2</b>
<b>2.1. SOFTWARE CONFIGURATION ITEM IDENTIFICATION .....</b>	<b>2</b>
2.1.1. <i>Software Requirements Specification.....</i>	<i>2</i>
2.1.2. <i>Software Configuration Management.....</i>	<i>2</i>
2.1.3. <i>Source Code.....</i>	<i>2</i>
2.1.4. <i>Test Suite.....</i>	<i>2</i>
2.1.5. <i>Software Tools.....</i>	<i>2</i>
2.1.6. <i>Change Request Forms.....</i>	<i>2</i>
2.1.7. <i>Change Log.....</i>	<i>2</i>
2.1.8. <i>Other .....</i>	<i>3</i>
<b>2.2. SOFTWARE CONFIGURATION ITEM ORGANIZATION .....</b>	<b>3</b>
2.2.1. <i>Directory Structure and Naming Conventions.....</i>	<i>3</i>
2.2.2. <i>Version Numbers.....</i>	<i>4</i>
2.2.3. <i>Version Control System.....</i>	<i>4</i>
<b>3. SOFTWARE CONFIGURATION CONTROL.....</b>	<b>5</b>
<b>3.1. DOCUMENTATION .....</b>	<b>5</b>
<b>3.2. CONFIGURATION CONTROL BOARD .....</b>	<b>6</b>
<b>3.3. PROCEDURES .....</b>	<b>7</b>
3.3.1. <i>Procedure for Checking-out Artifacts.....</i>	<i>7</i>
3.3.2. <i>Procedure for Committing Artifacts.....</i>	<i>7</i>
3.3.3. <i>Procedure for Creating New Baseline Configuration Artifacts.....</i>	<i>7</i>
3.3.4. <i>Administration of the Software Configuration Repository.....</i>	<i>7</i>
<b>4. SOFTWARE CONFIGURATION AUDITING .....</b>	<b>8</b>

# 1. Introduction

The purpose of this section is to describe the project, state the purpose of this document, state the intended audience of this document, provide an overview of the sections within this document, and list any references that were used to write this document.

## 1.1. Project Description

The system we are developing is for the El Paso Metropolitan Planning Organization (MPO). The El Paso MPO currently uses Microsoft Access and Microsoft Excel to manage projects, create projects, and generate reports. The El Paso MPO Management System will be web-based. The system will improve the current process by having one workspace in which the users can create, update, and retrieve information on various projects and reports.

## 1.2. Purpose and Intended Audience

The purpose of the Software Configuration Management (SCM) plan, is to provide a schema for the software team to ensure consistency throughout the development of this project. The SCM will contain guidelines for establishing naming conventions and managing changes in the project. The SCM must be robust so that the changes and updates made to the system are consistent and the project is managed in an organized fashion. The intended audience for this document is the software development team, guidance team, and the representatives from the El Paso MPO.

## 1.3. Overview

This document will contain:

Software Configuration Item Identification:

This section identifies a list of elements from the project that are likely to change.

Software Configuration Item Organization:

This section identifies a method for managing versions via a version control policy, and the organization of the configuration items.

Documentation:

This section provides documentation for formal change requests.

Configuration Control Board:

This section provides information regarding the Software Configuration Board that will review proposed changes.

Procedure:

This section describes the process of managing items and explains the process for making changes to the various software configuration items.

Auditing:

This section serves as a tool for validating and verifying changes against the baseline version.

## 1.4. References

Preston-Werner, T. (n.d.). *Semantic Versioning 2.0.0*. Retrieved January 29, 2016, from <http://semver.org/>  
Research National Institute of Dental and Craniofacial. (2014, March 6). *Version Control Guidelines*. Retrieved January 30, 2016, from <http://www.nidcr.nih.gov/Research/ToolsforResearchers/Toolkit/VersionControlGuidelines.htm>  
Ritchie, S. D. (2012, March 5). *Build Numbering and Versioning*. Retrieved January 30, 2016, from <http://ruthlesslyhelpful.net/2012/03/05/build-numbering-and-versioning/>

SCM	ZombieBytes	Date 2/1/2016 9:05 PM	Page 1
-----	-------------	--------------------------	-----------

## 2. Software Configuration Identification

In this section we have identified elements and documents that are part of developing the El Paso MPO management system that are likely to change. We have also described the structure of the repository that we will use for version control. Additionally, we have described the naming conventions we will be using to organize updates to our documents and source code.

### 2.1. Software Configuration Item Identification

The focus of this document is managing change. In order to create a systematic process to manage and implement changes, we must first identify what items are most likely to change throughout the development process. The items that we have identified that will change are the following:

- Software Requirements Specification (SRS)
- Software Configuration Management (SCM) Plan
- Source Code
- Test suites
- Software Tools
- Change Request Forms
- Change Log
- Other documents

#### 2.1.1. Software Requirements Specification

The SRS contains all the requirements required to create the system. Any time a change is made to the requirements the SRS must be updated to reflect these changes.

#### 2.1.2. Software Configuration Management

The SCM document contains the process for maintaining control of the software configuration items. The process may change during the life cycle of this project, therefore this document must be updated to correspond to the changes made.

#### 2.1.3. Source Code

The source code is the implementation of the system. If there is a change in the requirements the source code is likely to change. The source code will also require change when an error is detected.

#### 2.1.4. Test Suite

Test suites are meant to test the system to make sure it adheres to the requirements. Therefore any time a change is made to a part of the system, the test suites must be updated to reflect the changes made.

#### 2.1.5. Software Tools

Software tools include artifacts such as compilers, IDEs, as well as the operating system used to create and maintain the system. The software tools used must be documented in order to serve as future reference for individuals that continue to manage this project.

#### 2.1.6. Change Request Forms

Change Request Forms must be filled out whenever a change is requested. There must be a record of all requests made for future reference.

#### 2.1.7. Change Log

All changes made must be logged with a reference to the original request form.

SCM	ZombieBytes	Date 2/1/2016 9:05 PM	Page 2
-----	-------------	--------------------------	-----------

### 2.1.8. Other

Other documents will be stored in the repository as needed.

## 2.2. Software Configuration Item Organization

This section will specify how the items in our system are to be organized. The structure will be easily searchable and will include all items created as part of the development process.

### 2.2.1. Directory Structure and Naming Conventions

The directory structure is shown in Figure 1 below.



Figure 1. Directory structure of the system repository.

Naming conventions are important to maintain organization. It will allow for quick access to documents associated with the project. The naming convention that the software team has decided on is as follows :

**ZombieBytes\_<Doc Name>\_<Version>.<ext>**

SCM	ZombieBytes	Date 2/1/2016 9:05 PM	Page 3
-----	-------------	--------------------------	-----------

The convention can be broken down like so: the name of the software team followed by the document name followed by the version and finally the file extension. Each of these elements is separated by an underscore.

### 2.2.2. Version Numbers

In order to maintain item versions, we created version number schema. For source code the software team has decided on the following format:

**X.Y.Z**

X represents a major change in the system and is the main version number. A major change signifies a new feature that significantly impacts the system. Y represents a minor change in the system and the addition of a new feature. Z represents revisions made to the code and bug fixes. The board will decide what will be considered a major change, a minor change, or a revision/bug fix.

For documents the software team has decided on the following format:

**X.Y**

X will represent when a document reaches its final version, e.g. the final draft (1.0), final paper (2.0). Y will represent any changes made such as adding/removing/editing sections, rewording, grammatical errors, etc.

### 2.2.3. Version Control System

The software team will be using Git for the primary version control system, and will be using GitHub to publish the Git repositories online. This will allow all members of the software team to have a working copy of the project on their system at all times. Members will be required to update their repository before working on the project to ensure that everyone is working on the latest version of the project. The project, being located on all team members' systems, and on GitHub, will serve as the main form of backup.

SCM	ZombieBytes	Date 2/1/2016 9:05 PM	Page 4
-----	-------------	--------------------------	-----------

### 3. Software Configuration Control

This section contains the change request forms a team member must submit in order to notify the other team members that a change is needed. This section contains the change approval form that the team will need to approve in order for the change to be implemented. This section also establishes the structure of the configuration control board. Lastly, this section also provides the procedures the team must adhere to while making changes to the software system.

#### 3.1. Documentation

The software development team needs to maintain a record of changes made to the system. Two forms were created to assist in this process. The first form required in this process is the Change Request Form. The Change Request Form must be completed by the person requesting a change to initiate the process of making changes to any of the SCM items. This form and the information that must be submitted can be found below in figure 2.

**Wufoo**

### Change Request Form

This form is to be completed by the person making the request for this change.

**Name of person requesting changes**

First Last

**Date**

/  /

MM DD YYYY

**What document are these changes are in reference to? (If not applicable leave blank)**

**Provide a brief description of the changes you are requesting**

**Provide a brief justification for what precipitated these changes**

**Select a priority level**

☒ Low

☐ Moderate

☐ High

**Indicate Desired Delivery Date**

/  /

MM DD YYYY

[Report Abuse](#)

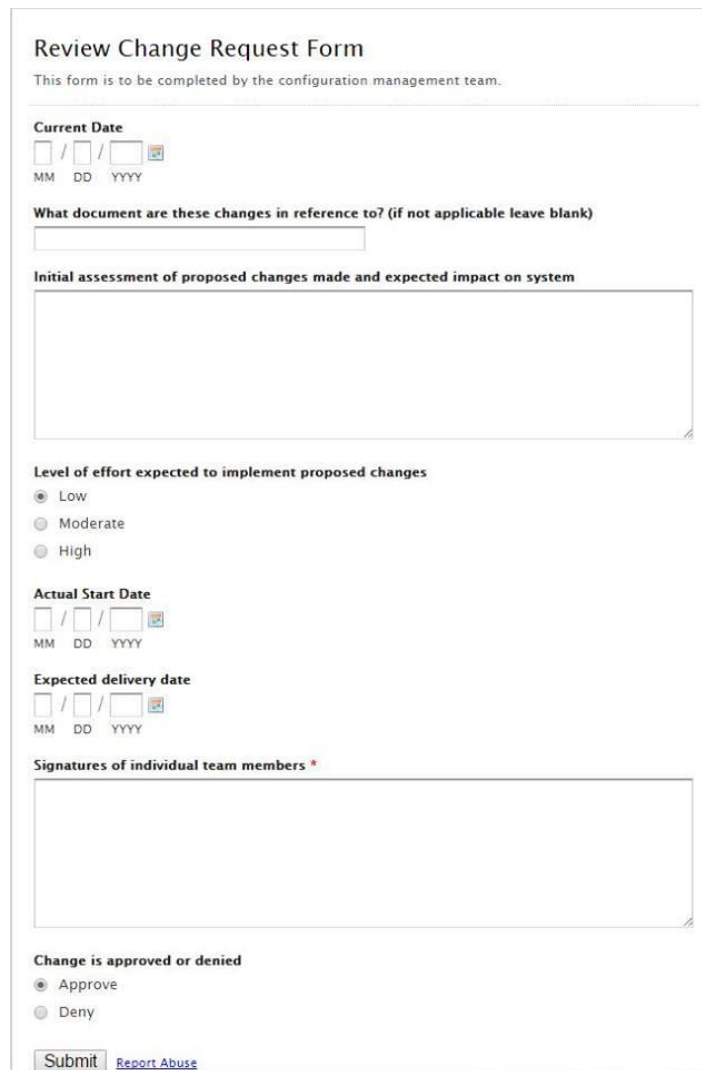
Google Hangouts is sharing your screen with plus.google.com.

Figure 2. Change Request Form

SCM	ZombieBytes	Date 2/1/2016 9:05 PM	Page 5
-----	-------------	--------------------------	-----------




The second form is the Review Change Request Form filled out by the Configuration Control Board when reviewing the original change request. It is in this form that the original request is either approved or disapproved. This form and the information that must be submitted can be found below in figure 3.



**Review Change Request Form**


This form is to be completed by the configuration management team.


**Current Date**  
 /  /    
MM DD YYYY

**What document are these changes in reference to? (if not applicable leave blank)**

**Initial assessment of proposed changes made and expected impact on system**

**Level of effort expected to implement proposed changes**  
☒ Low  
☐ Moderate  
☐ High

**Actual Start Date**  
 /  /    
MM DD YYYY

**Expected delivery date**  
 /  /    
MM DD YYYY

**Signatures of individual team members \***

**Change is approved or denied**  
☒ Approve  
☐ Deny

[Report Abuse](#)

Figure 3. Review Change Request Form

Once both forms are filled out, they will be entered into an Excel document that will serve as the Change Log. The links to the Change Request Form and the Review Change Request Form will be found in the repository, along with the Change Log.

### 3.2. Configuration Control Board

The Configuration Control Board will be made up of all the members of the software team. The team is made up of: Carmel Heydarian the Systems Analyst , Chelsey Jurado the Lead Programmer , Jordan Cox the V&V, and Christian Hughes the Designer. All the members of this team will be responsible for evaluating and approving or disapproving a proposed change to the system. All the team members will have access to modifying different parts of the system. Team members will only be able to modify any part of the system after the change is approved. Chelsey Jurado will be responsible for distributing the changes. Approving or disapproving changes will be decided with a vote from all the members of the team. In the event of a tie, the individual whose position in the team deals with the change will have their vote count as two votes instead of one. An example of this would be: “The team has voted to change something in the source code. The lead

SCM	ZombieBytes	Date	Page
		2/1/2016 9:05 PM	6

programmer's vote will count for two votes instead of one." Changes will be approved using the documentation provided in section 3.1 of this document. The factors that will be evaluated to approve or disapprove a change will also be included in the document provided in section 3.1. Errors in the code will be reported by the V&V through the change request form indicating the file that is to be changed, the line number where the change will take place, and a description of the change that will take place. The artifacts that are changed will be placed in a version directory, and a previous version of the artifact will also be saved as a baseline.

### **3.3. Procedures**

This section provides an overview for the various procedures involving managing changes to the Software Configuration Items.

#### **3.3.1. Procedure for Checking-out Artifacts**

The procedure for checking out artifacts from the team repository first involves filling out a Change Request Form. After the Change Request Form is completed a Review Change Request Form will need to be completed. Once the team member who wishes to check out the artifact from the repository receives approval, they will be able to do so.

#### **3.3.2. Procedure for Committing Artifacts**

Before an artifact is committed to the repository the Configuration Control Board must approve the changes that were made. The rules for the Configuration Control Board can be found in section 3.2. When the artifact is approved, the team member who made the changes will be responsible for committing the artifact to the repository.

#### **3.3.3. Procedure for Creating New Baseline Configuration Artifacts**

The procedure for creating a new baseline configuration artifact will involve two steps. The first step is the team discussing the changes made to the artifact and the Configuration Board approving all the changes. The second step is the team discussing the changes with a member of the guidance team and receiving their approval on the changes made to the artifact. After these two requirements have been met, the artifact in question can replace the previously established baseline artifact.

#### **3.3.4. Administration of the Software Configuration Repository**

Administration of the Software Configuration Repository will be managed by the administrators Chelsey Jurado the Lead Programmer of the team and Christian Hughes the Designer. Every team member will have access to the repository, but the administrators will be responsible for ensuring the repository is up to date, and that the naming conventions, and guidelines established in this document are being followed.

SCM	ZombieBytes	Date 2/1/2016 9:05 PM	Page 7
-----	-------------	--------------------------	-----------

## 4. Software Configuration Auditing

It is important to follow the procedures that we have established in this document to ensure the thorough completion of the intended system. In order to accomplish this we have set in place an auditing process to do so. The auditing process is necessary because it produces documentation that eliminates confusion as to who proposed what change, how it was handled, and how long the changes took to implement.

It is important that when managing changes the team always takes the requirements into consideration. Any change made to a Software Configuration Item must be verified and validated against the requirements specified in the SRS. This verification and validation will be ensured by Jordan Cox the team's V&V. The method that will be used to document this verification and validation will be a traceability matrix. The requirements will be mapped to the various Software Configuration Items that fulfill them using forward tracing. Each Software Configuration Item will also be mapped to the requirement it fulfills using backward tracing. This will allow the team to easily determine which Software Configuration Items will most likely change whenever a requirement changes or vice-versa.

^\_^