

# **Software Configuration Management Plan: Smart Video Networking**

Comp190: Senior Design Project  
3/9/10

By:  
Ronald Laidley  
Andre Govier  
Eric Gustavson

**Table of Contents**

- 1. Introduction..... 2**
  - 1.1. Purpose of the Plan..... 2
  - 1.2. Scope of the Plan ..... 2
  - 1.3. Definition of Key Terms ..... 3
  - 1.4. References ..... 3
  - 1.5. Downward Tailoring..... 3
- 2. SCM Management..... 4**
  - 2.1. Organization..... 4
  - 2.2. SCM Responsibilities..... 4
  - 2.3. Applicable Policies, Directives, and Procedures ..... 5
- 3. SCM Activities ..... 5**
  - 3.1. Configuration Identification..... 5
  - 3.2. Configuration Control..... 6
  - 3.3. Status Accounting ..... 7
- 4. SCM Schedules ..... 8**
- 5. SCM Resources..... 9**
- 6. SCM Plan Maintenance ..... 9**

# **1.Introduction**

## **1.1. Purpose of the Plan**

This plan will act as a reference for the members of the Streaming Video Networking development team during the entirety of the Project. It will serve not only to make the deployment of SCM more efficient, but also as a guide to maintain control over the various aspects of SCM during development.

## **1.2. Scope of the Plan**

During this project the team will design and implement server/client video streaming software capable of streaming video using the open source software LCM. The server-side module of the software will provide a library of functions to read video files in either MPEG2 or AVI format and broadcast that video across a local area network using the UDP Multicast protocol. The client-side module will provide a library of functions to setup an LCM listener on a specified channel, and upon receiving packets of video information over that channel, decode the video stream into individual and independent images representing frames of the original video. The SCM plan will apply to the requirements documents, the design documents, the source code, the test plan and the test cases

SCM will be applied on an informal level due to the relatively small scale of this project. However, in order to maintain consistency in units of code being actively developed, the SCM activities dealing with the source code will be more strictly enforced and to a greater depth. SCM will be applied to the design stage, implementation stage, testing stage and deployment stage of this project. The maintenance stage of the project life cycle will not be covered by SCM.

The most crucial limitation on this plan is the time constraint, the total time for development of this project being approximately 15 weeks.

We will assume that the customer will be able to perform certain limited functionality tests for us during the testing phase of the project. This limited testing will be required to ensure compatibility between the hardware being used for development and the hardware on which the system will be deployed.

### 1.3. Definition of Key Terms

**SVN (Smart Video Networking):** Abbreviation of project.

**LCM (lightweight Communication and Masrhalling).** An open source library for message passing and data marshalling.

**MPEG2.** A standard for the generic coding of moving pictures and associated audio information. The standard includes methods for both lossy video and lossy audio compression.

**AVI.** A multimedia container format introduced by Microsoft in November 1992 as part of its Video for Windows technology.

**UDP (Universal Datagram Protocol) Multicast.** A protocol used by computer applications to send messages across local area networks and the internet without requiring prior communications to set up special transmission channels or data paths.

**LCM Listener.** A process that waits to receive messages over a pre determined UDP channel.

**FFMPEG.** An open source library for coding and decoding video and image of various formats.

### 1.4. References

LCM technical report: <http://dspace.mit.edu/bitstream/handle/1721.1/46708/MIT-CSAIL-TR-2009-041.pdf>

MPEG reference website: <http://www.mpeg.org/MPEG/mpeg-systems-resources-and-software/>

AVI reference website: [http://msdn.microsoft.com/en-us/library/dd318187\(VS.85\).aspx](http://msdn.microsoft.com/en-us/library/dd318187(VS.85).aspx)

FFMPEG reference website: <http://ffmpeg.org/>

### 1.5. Downward Tailoring

Due to the short time scale of the project, these sections were deemed unnecessary for inclusion in this plan:

- Configuration Audits and Reviews

- Interface Control

- Subcontractor/Vendor Control

## 2. SCM Management

### 2.1. Organization

Roles	Eric Gustavson	Andre Govier	Ron Laidley
Team Lead			x
Build Manager	x		
Change Control Czar		x	
Documentation Czar		x	
Test Manager			x
Architect		x	
Lead Designer			x
Coders	x	x	x
Deployment Manager	x		
Web Guru	x		

### 2.2. SCM Responsibilities

Tasks	Eric Gustavson	Andre Govier	Ron Laidley
Setup, Manage SVN		x	
Setup, Manage Bugtracker			x
Coding: encoder module		x	
Coding: decoder module			x
Coding: LCM Sender	x		
Coding: LCM Listener	x		
Testing	x	x	x

Final Testing		x	
Demo Preparation			x
Product Packaging, Delivery	x		

### 2.3. Applicable Policies, Directives and Procedures

There are no external constraints placed on the plan by other policies, directives and procedures.

## 3. SCM Activities

### 3.1. Configuration Identification

Requirements Document:

The requirements set forth by the customer are as follows:

1. The SVN must recognize and use both MPEG and AVI files.
2. LCM must be used to transfer information (video frames).
3. There must be a noticeable decrease in bandwidth use. Right now, all images from the video are being sent across the network without alteration. Although there is no exact limit, the bandwidth usage of the SVN system must be less than this case.

Since these have been set by the customer, they must be adhered to. Thus, control of the system must be maintained to ensure these requirements are met.

Design Document:

The design document is based on different views of the SVN system used to describe how it operates. As such, future updates to the system will affect the design document to varying degrees.

Source Code:

The source code for both the server-side module and the client side module of the system will be controlled for this project. This source code will consist of methods which will utilize both the open source LCM libraries and FFMPEG libraries in order to process, send and receive video and image data of various formats.

Test Plan:

The SVN system does not lend itself to a large automated test suite. Instead, a base video will be run through the system and will be verified as correct by the management team. From here, any changes to the system must have the same video run through it to determine if the new result differs from the initial, accepted result.

Test Cases:

1. Send MPEG file with no compression.
2. Send MPEG file with some nominal compression.

3. Send MPEG file with maximum compression.
4. Send AVI file with no compression.
5. Send AVI file with some nominal compression.
6. Send AVI file with maximum compression.

### **3.2. Configuration Control**

#### Requirements Document:

1. Request: If there is a request for a change in the requirements document, the developer must formally state his or her reasoning (the request is impossible, there is a better choice, etc...) to be presented to the customer.
2. Evaluate: All members of the project must evaluate the request. If a majority of the participants feel the request is justified, it will be submitted to the customer.
3. Approve/Disapprove: The customer has the right to approve or disapprove of any changes to the requirements of the system they want. Developers can attempt to reason with them how they see fit, but the customer has the final say.
4. Implementation: If the customer approves the change, the document must be altered to show this. All work must also reflect this change. If a previous requirement was removed, it shall no longer be a concern. If a new requirement was added, all future actions must be made to ensure it is met.

#### Design Document:

1. Request: Should a developer find that the current design is insufficient, he or she can simply bring it to the attention to the team along with his or her proposed solution.
2. Evaluate: Since the design team is so small, they can evaluate the request together. The developers must determine if the proposed solution will benefit the project or negatively impact it.
3. Approve/Disapprove: Should a majority of the developers agree that the new design will be beneficial to the project, it will be approved. Otherwise, it will not.
4. Implementation: The design document must be altered to depict the new change. All work that has been done and that will be done must be altered to

#### Source Code:

1. Request: If a developer is requesting a change to source code that only he or she has touched, he or she can simply submit it to the repository via Subversion. To request a change to code of another developer, he or she must describe the change and why it is necessary to said developer.
2. Evaluate: The two developers should be able to evaluate the benefit of the proposed change. If not, the rest of the team can be brought in to discuss it's merits. The change should provide some enhancement without having any negative impact on the rest of the system.
3. Approve/Disapprove: If the two developers involved or a majority of the team determine that the proposed change to the source code will improve the product without inconveniencing the rest of the code, then the change will be approved.
4. Implementation: The change will be made to the code and it will be resubmitted using Subversion. Tests will have to be run to ensure that the change has not introduced any bugs.

#### Test Plan:

1. Request: Should a developer feel the current test plan is insufficient or unnecessary, they may present a proposed solution to the rest of the team.
2. Evaluate: The team should determine if the proposed change will still produce the necessary results offered by the original test plan. Even if the new test plan is easier to manipulate, if it does not gain at least the same results of the initial test plan it is not worth it.
3. Approve/Disapprove: Should the team find that the new test plan is easier create or provides better results, it will be approved.
4. Implementation: The test will be created in accordance with the plan and run on the system. The results should be better than the previous test plan.

#### Test Cases:

1. Request: A developer may informally submit a request to add a new test case or remove an old one whenever they want. Along with it the must also submit a reason for adding or removing said case.
2. Evaluate: The team must determine if removing a test case will leave the product open to bugs that could go unchecked, or if any new test case added provides some coverage against other bugs that could have been overlooked.
3. Approve/Disapprove: If the team finds that adding the test case will detect a reasonable problem in the system or that removing a test case will have no consequences, then the request will be approved.
4. Implementation: The case will be added or removed from the test suite. The next time the system is changed and the tests are run the new case will be p[resent or the removed case will be absent.

### **3.3. Status Accounting**

#### Requirements Document:

The document itself is the sole maintainer of all requirements for the SVN system. Whenever a change is made by the customer or developers that alter the requirements, this document needs to be updated accordingly. As such, this document will only be updated as necessary, not at scheduled intervals. The document will be available to all developers and be displayed on the project website.

#### Design Document:

The design document will be maintained in the same manner as the requirements document.

#### Source Code:

All source code files will be stored in a repository via Subversion. It will maintain an up to date repository of all files. Updated files will be checked in via Subversion and all access to these files will be available to all developers.

#### Test Plan:

The test plan will be displayed on the project website. No reports will be necessary for changes to the plan, but all changes must be documented explicitly stating why they were made. Anyone can gain access to the test plan, but it can only be altered by developers.

Test Cases:

Because of the nature of the SVN system, an automated test may not be possible without excessive and unnecessary work. As such, there is no code to maintain for this. however, all test cases will be available on the project website. Any updates will have to be clearly documented.

### 4. SCM Schedules

ID	Task Name	Start	Finish	Duration	Feb 2010				Mar 2010				Apr 2010				May 2010			
					1/31	2/7	2/14	2/21	2/28	3/7	3/14	3/21	3/28	4/4	4/11	4/18	4/25	5/2	5/9	
1	Initial Meetings, Preliminary Requirements	1/26/2010	2/2/2010	6d																
2	Refined Requirements	2/2/2010	2/9/2010	6d																
3	Architecture Design	2/10/2010	2/17/2010	6d																
4	Video Technology Review	2/18/2010	2/25/2010	6d																
5	Select 3rd party sw libraries for video encoding, SCM Planning	2/18/2010	3/5/2010	12d																
6	Design Documents, begin development	3/3/2010	3/17/2010	11d																
7	Development: Integrate & test Video Encoder	3/3/2010	3/23/2010	15d																
8	Development: Integrate & test Video Decoder (to create image files from video)	3/3/2010	3/30/2010	20d																
9	(Spring Break)	3/23/2010	3/30/2010	6d																
10	Development: Send video stream over LCM. Integrate lcm with encoder & decoder.	3/24/2010	4/7/2010	11d																
11	Testing.	4/8/2010	4/22/2010	11d																
12	Final Testing, demo preparation.	4/13/2010	4/20/2010	6d																
13	Product packaging and delivery.	4/21/2010	4/28/2010	6d																
14	Final Presentations, Wrap-up.	4/27/2010	5/4/2010	6d																

Color Key:

Blue: Planning

Green: Development

Red: (spring break)

Orange: Testing

Purple: Shipping

Critical Path Link:



## **5. SCM Resources**

### Requirements Document:

There are no tools necessary to complete this activity aside from a text editor. However, requirement analysis and refinement skills are necessary to ensure that all requirements are recorded and understood so that they can be completed.

### Design Document:

The only tool necessary is Visio (or some other modeling program) for creating the views that define the project. Again, design analysis skills will be needed to help ensure the project is defined accordingly.

### Source Code:

The software tools necessary for this project are LCM and ffmpeg. The use of LCM is a requirement made by the client, since it is what BAE currently uses to send information across their network. ffmpeg was chosen as a video manipulator since the product must be able to compress and send video information over a network via LCM, and ffmpeg is easy to use and offers all the functionality needed for this project. This also requires rese for all personnel involved, since none have used either tools before and cannot be avoided.

### Test Plan/ Test Cases:

Again, no specific tools are necessary, but the ability to to create an extensive test suite is. The tests that are run must be able to determine if the system is working properly, and only by using equivalence partitioning over all inputs can it be determined if the system is working as desired.

## **6. SCM Plan Maintenance**

Although the team lead (Ron), change control czar (Andre), and documentation czar(Andre) are in charge of maintaining the document as well as recording and evaluating changes to the plan, all team members will have input towards the development of the plan and the project. Weekly meetings will ensure that the plan will continue to evolve as the project moves forward. However, changes can occur at anytime, so although consideration and updates will be made at least once a week, changes will be made continuously by the team as matters develop. All members will have a say in any new ideas that may affect the plan, and the group will decide as a whole if these changes are necessary for the project. Should the team not be able to meet physically for each request, Google Wave and email will be used to communicate various ideas for change so that every member of the team can have a say in any new developments. Once a consensus has been reached ot change the plan, this document must be updated by the documentation czar to reflect said change.