Configuration Management Report

Hakan Nizamoğlu
Yiğitalp Ertem
Murat Toprak
Saim Güveloğlu

2010
# Table of Contents

1. Introduction .............................................................................................................. 3
   1.1 Identification ................................................................................................. 3
   1.2 Purpose ........................................................................................................... 3
   1.3 Scope ............................................................................................................. 3

2. Configuration Management Organization .......................................................... 3
   2.1 Organization .................................................................................................. 3
      2.1.1 Roles and Responsibilities .................................................................... 3

3. Tools and Technologies ..................................................................................... 5
   3.1 Embedded Board Side ................................................................................ 5
   3.2 Computer Side (Development Environment) ............................................. 5

   4.1 Configuration Identification Procedures .................................................. 6
      4.1.1 Configuration Identification Policies for System-Level Requirements
            Specifications ......................................................................................... 6
      4.1.2 Configuration Identification Policies for Software ................................ 7
      4.1.3 Configuration Identification Policies for Hardware ............................ 7
   4.2 Change Control Procedures .......................................................................... 8
      4.2.1 Change Control Policies for System-Level Requirements Specifications .... 8
      4.2.2 Change Control Policies for Software ................................................ 9
      4.2.3 Change Control Policies for Hardware ................................................ 9
   4.3 Documentation Change Control Procedures ............................................. 10

5. Configuration Audits ............................................................................................ 11

6. Configuration Management Tasks ....................................................................... 12
   6.1 Configuration Identification Tasks ............................................................. 12
   6.2. Change Control Tasks .............................................................................. 14
      6.2.1. Change Requests ............................................................................... 14
      6.2.2. Change Request Report ..................................................................... 14
      6.2.3. Change Request Evaluation .............................................................. 14
      6.2.4. Change Implementation ................................................................. 14
      6.2.5. Defect Tracking ............................................................................... 15
      6.2.6. Development Management and Control ....................................... 15
   6.3 Configuration Status Accounting ............................................................... 15

7. Project Schedules & CM Milestones ................................................................. 16

8. Resources ............................................................................................................. 17
   8.1 Human Resources ...................................................................................... 17
   8.2 Equipment .................................................................................................. 17
   8.3 Facilities ..................................................................................................... 18
   8.4 Budget ........................................................................................................ 18
1 Introduction

1.1 Identification
This is the configuration management plan for our project CUBE which is aimed to lower CPU usage on the server side by taking care of security issues and place it on the embedded board.

1.2 Purpose
This configuration management plan will briefly describe how the technical components are used and configuration process is done and also how the team members contribute what responsibilities they have to the project in the configuration and technologic manner.

1.3 Scope
This Configuration Plan is the first version of our project. Since it is the first there may be another Configuration Plan’s prior to some changes on project’s details.

2 Configuration Management Organization

2.1 Organization
Our project is being developed under group CUBE’s control. CUBE is consisted of 4 people who are responsible for different aspects of the project and development phases. This Configuration Management Plan will also describe those team members’ role in the project. Members are:

- Hakan Nizamoglu
- Murat Toprak
- Yigitalp Ertem
- Saim Guveloglu

Each team member is responsible for criticizing, giving ideas, being a part of brainstorming events whereas social activities in the goodness of the project.

2.1.1 Roles and Responsibilities
2.1.1.1 Software Development
Programming and development works are considered under this topic. Since our program will work on an embedded board which has limited capabilities compared to our computers that we usually use in office or home, we will need to reconsider every piece of code in the name of efficiency. Non-efficient parts in our software will cause hazardous problems for the outcome which may not be tolerable by the users of it. So, programmers should be greedy about the system resources while they allocate system memory for instance. Since our time and workforce are limited in our project every team members should/must contribute their best at this step.

Targeted Members : All of the team members.

2.1.1.2 Hardware Configuration and Hardware Related Software Installation

To make our team members able to write and run and code or helpful code parts (i.e. APIs) development environment must be ready and set to meet the requirements of the development team. There members have to identify pre-assumed components of the projects and apply them to the development board and yet if they cannot make it possible in no way around then they should inform other team members to reconsider the requirements and configuration options of the project.

Targeted Members: Hakan Nizamoglu

2.1.1.3 Testing Team

Another critical point in this project is testing. Unlike the most of projects around the world testing phase is more important in embedded projects because of the lack of system resources. It is the point where outcome of this project give us strong clues and help us to identify what is going wrong. Since we are using a Virtual Machine of Java we have to be careful lot more than usual C programming while coding.

Targeted Members: All of the team members.

2.1.1.4 Configuration Control Board

Health of the development board is placed under this topic. Any update, any change in hardware and software will be done by these members. These members are responsible to notify team members the changes.

Targeted Members: Hakan Nizamoglu
3 Tools and Technologies

Our whole development process is divided into 2 sub-topics. What we have used in board and regular computer is different due to major differences of those systems.

3.1 Embedded Board Side

We are developing our software in JAVA. Since it is JAVA classes we need JAVA environment for running and testing phases of our project outcome. To list tools and technologies on embedded board:

- JamVM: Open-source JAVA Virtual Machine which can run on Atmel’s NGW100 embedded board.
- Embedded Linux: All of our board side libraries, virtual machines and compilers are for generating executables on embedded Linux. It is also very important to note that we need that Linux kernel for networking purposes and also SD Card slot which is placed on the board.
- Built-in Network Adapter placed on board.
- Analog Sound I/O peripherals
- SD Card slot: Since we are coding in JAVA our virtual machine on NGW100 requires some space for JAVA’s CLASSPATH libraries.
- GNU’s CLASSPATH for JamVM.

3.2 Computer Side (Development Environment)

Atmel’s NGW100 has limited capabilities with its memory and CPU. So we cannot able to use that board for developing since the compiling and debugging is very time consuming on the board. So we are developing our codes on our regular computers, compile them and transfer the executables to the board for running any program on it.

What are we using in our computer?

- NetBeans IDE: NetBeans is Sun’s IDE for specialized for JAVA development. We are using it for our project’s major outcome.
- Filezilla: We are transferring projects executables or libraries to the board using FTP.
Atmel’s GCC Compiler: Since NGW100 has different kind of CPU we needed to differentiate our source codes for different compiling procedures. We are using Atmel’s GCC Compiler for cross-compiling works. Anything other than JAVA class files should be compiled using this compiler in order to run on NGW100.

GNU GCC: To compile system dependent Atmel’s GCC Compiler in any computer we are using GNU’s GCC for compiling the source code of Atmel’s GCC Compiler.

SSH: Any testing and hardware configuration process needs to be done on the terminal. We are using SSH to make it secure.

Network Adapter: To complete our development environment we use board in LAN.

BUILDROOT and TOOLCHAIN which are provided by Atmel for NGW100.

4 Configuration Management Policies and Procedures

4.1 Configuration Identification Procedures

4.1.1 Configuration Identification Policies for System-Level Requirements Specifications

Policies for Configuration Identification of System-Level Requirements Specifications

❖ One and only one system-level specification must exist for the Secure Voice over IP system.

❖ The system-level requirements specification is documented as System Requirement Analysis Report.

❖ The approved system-level requirements specification will be baselined (as the Functional Baseline) and will be subject to formal change control.

❖ The content of the system-level requirements specification must be unambiguous, consistent, complete and testable.

❖ The system-level requirements specification establishes the acceptance
criteria for the Secure Voice over IP system.

4.1.2 Configuration Identification Policies for Software

Policies for Configuration Identification of Software Configuration Items

- Each application software configuration item must have a specification document.
- Application software specifications should be contained in separate documents.
- Each application software specification will be baselined (as part of the Allocated Baseline) and subject to formal change control.
- Application software specifications should be traceable to the system-level requirements specification.
- The content of the application software specification establishes the acceptance criteria for that software configuration item.
- Each application software specification will be numbered with the configuration items identifier as part of the document number, for example if the specification for application software configuration item is SOFT1, will be SOFT1-SPEC.

4.1.3 Configuration Identification Policies for Hardware

Policies for Configuration Identification of Hardware and System Software

- Each hardware and system software configuration item must have a specification document.
- All hardware and system software specifications may be packaged into a single document separately and easily identifiable (the recommendation of the Configuration Manager is that separate documents be used).
- Each hardware and system software specification will be base lined (as part of the Allocated Baseline) and subject to formal change control.
Hardware and system software specifications must be traceable to the system-level requirements specification.

The content of the hardware and system software specification establishes the acceptance criteria for that product.

Each hardware and system software specification will be numbered with the configuration item's identifier as part of the document number, for example the specification for hardware configuration item is HARD1, will be HARD1-SPEC.

Hardware items will not be marked, however a log will be maintained by Configuration Management personnel that links the manufacturer's name, and model number to the hardware or system software configuration item.

4.2 Change Control Procedures

4.2.1 Change Control Policies for System-Level Requirements Specifications

Policies for Controlling Changes to System-Level Requirements Specifications

Require formal (written) requests for all changes.

Review all change requests and limit changes to those approved.

Analyze and evaluate the type and frequency of change requests.

Consider the degree to which a change is needed and its anticipated use.

Evaluate changes to ensure that they are not incompatible with the original design and intent. No change should be implemented without careful consideration of its ramifications.

Emphasize the need to determine whether a proposed change will enhance or degrade the system.

Approve changes only if the benefits outweigh the costs.
Schedule all changes.

Enforce documentation standards.

Plan for change.

4.2.2 Change Control Policies for Software

Policies for Controlling Changes to Application Software

- Require formal (written) requests for all changes.
- Review all change requests and limit changes to those approved.
- Analyze and evaluate the type and frequency of change requests.
- Consider the degree to which a change is needed and its anticipated use.
- Evaluate changes to ensure that they are not incompatible with the original design and intent. No change should be implemented without careful consideration of its ramifications.
- Emphasize the need to determine whether a proposed change will enhance or degrade the system.
- Approve changes only if the benefits outweigh the costs.
- Schedule all changes.
- Enforce documentation and coding standards.
- Require that all changes be implemented using sound software engineering practices.
- Plan for change.

4.2.3 Change Control Policies for Hardware

Policies for Controlling Changes to Hardware and System Software

- Require formal (written) requests for all changes.
Review all change requests and limit changes to those approved.

Analyze and evaluate the type and frequency of change requests.

Consider the degree to which a change is needed and its anticipated use.

Evaluate changes to ensure that they are not incompatible with the original design and intent. No change should be implemented without careful consideration of its ramifications.

Emphasize the need to determine whether a proposed change will enhance or degrade the system.

Approve changes only if the benefits outweigh the costs.

Schedule all changes.

Enforce documentation standards.

Plan for change.

4.3 Documentation Change Control Procedures

Policies for Controlling Changes to Deliverable Documentation Items

After the initial acceptance of a document, further changes will be subject to formal change control.

If only certain pages (less than 30% of the document) are changed, then change pages shall be issued; the document number will not change.

If the document is reissued in its entirety (more than 30% of the pages are changed), the document number will have the revision identifier appended; for example, document XYZ001 will become document XYZ001A.

In all cases, change bars shall be used to indicate changed material.

The date of issue will be recorded on the footer of each page (centred between the document number and the page number).
All changes made to previously approved documents must also be approved before release (via a Change Request).

5. Configuration Audits

In the auditing part, the evaluation of the project is done and depending on the results of the evaluation, the actions are taken. In the project, auditing is done after important changes. Moreover, any member can ask an audit in the group meetings.

Another important point to complete the project is auditing. Auditing provides us to foresee the problems that can occur. Auditing is consisting of three main parts which are functional audit, physical audit and process audit.

Functional audit will be for the performance analysis and the verification of the product. Physical audit will be performed for the design issues. Both will have a checklist to be approved by CCB. Also both audits will be performed after each module is built and before each demo.

The three types of auditing are:

**Functional audits**: These are done to see the performance by testing the software.
- Analysis of related CR history
- Test readiness review
- Documentation Control

**Physical audits**: These are done to be sure that the software consists all of the required components, documents and data.
- Schedule and action progress evaluation
- Design concept verifications for each design element.

**Process audits**: These are done to compare the manner in which the end product is produced to the written procedures.
6. Configuration Management Tasks

6.1 Configuration Identification Tasks

Following table identifies the configuration identification tasks that will be performed by CM personnel. Estimated effort in person days is provided as a guideline; however, CM is a Level of Effort function on the SecVoIP project.

<table>
<thead>
<tr>
<th>Configuration Identification Task</th>
<th>Task Description</th>
<th>Estimate (days)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Staff CM position(s)</td>
<td>Identify personnel with the needed skills to perform all CM tasks (not just configuration identification tasks). Arrange for that person to be assigned to the SecVoIP project.</td>
<td>2 days</td>
</tr>
<tr>
<td>Task</td>
<td>Description</td>
<td>Duration</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-----------</td>
</tr>
<tr>
<td>Develop CI numbering scheme</td>
<td>Develop a numbering scheme that is simple to use but allows each CI to be identified uniquely.</td>
<td>2 days</td>
</tr>
<tr>
<td>Establish Functional Baseline</td>
<td>When the Requirements Report is approved, write a memo to the CUBE members establishing the Functional Baseline and the need for change control of the document.</td>
<td>1 day</td>
</tr>
<tr>
<td>Monitor the Developmental Configuration</td>
<td>As software development progresses and documents are approved make sure that the software development team maintains a clear, unambiguous environment to prevent loss or duplication of software designs or modules.</td>
<td>60 days</td>
</tr>
</tbody>
</table>
6.2. Change Control Tasks

We have divided our work into parts as independent as possible. The development of each part is under different person’s responsibility.

6.2.1. Change Requests

During the development of the SecVoIP any member of the team can request any change any time. Since each person deals with a different portion, change request reports should be prepared only if the change is major or it can affect others work. Minor changes can be stated informally; change request report is not necessary for minor changes. But, the purpose of the change and the change should be stated.

6.2.2. Change Request Report

Here is the format of the change request report:

· Name of the member making the request
· Date of the request
· File name of the source code
· Reason for change
· Proposed changes

6.2.3. Change Request Evaluation

In weekly meetings, change requests belonging to the previous week are discussed. The reasons for the changes, their priorities and effects are taken into account during the evaluation. Since the member developing the part of the code which is wanted to be changed will make the desired change, his opinion will be taken especially.

6.2.4. Change Implementation

If the change request is approved, the member developing the part of the code which is subject to change will make the necessary changes.
6.2.5. Defect Tracking

All members are responsible for defect tracking for a reliable software. They should report all defects detected in the software. For removing defects, change request should be done and evaluated to determine the responsible person to eliminate the defect.

6.2.6. Development Management and Control

We use Java Coding Conventions in the development of our project for understandability. And every class has declarations including:

- Name of the file
- Name of the contributors
- Creation date
- Description of the class
- Change history (including the person who made the change, date of the change, reasons to change and changes made)

6.3 Configuration Status Accounting

Configuration status accounting involves the recording and reporting of the change process.

The goal of configuration status accounting is to maintain a status record of all items in the system baseline, thus providing traceability of all changes to the system.

Since this is a relatively small project compared to other commercial projects, and we are 4 people sharing the code, we think that it is enough to use our mail group to state our current status, changes and bugs. The important thing here is that the explanations should be made for every change in case of the risk for forgetting it. Also, the decisions and bugs should be documented with reasons.

Our major activities of configuration status accounting are:

- Identifying the configuration status information to be recorded
- Maintaining a record of configuration changes
- Reporting the status of configuration management
7. Project Schedules & CM Milestones

Because of the lack of time and source (four member project), configurations are managed by person who is responsible for the configured part. Configurations will be reported at schedule and all members can see that who change which part of project. Also the configurations should be accepted by other members. When the first alpha version is completed, members will focus other parts and if they have different ideas or improvements, these are debated by group members. At the end, all group members have rights about configuration of the project.

We will inform our assistant about our progress in weekly meetings and consider the given feedbacks. We will make short demos in every 2 weeks after the first snapshot demo. We will continue to hold our own weekly meetings as a group. We have prepared a Living Schedule or our development process. The tests and debugging process has made us to make the scheduling in this way. After a module or a function has been designed and implemented, the preliminary tests and benchmarks will be performed. After this performs, new road map will be arranged. By this method, the schedule will always be up-to-date and meaningful. But we have strict deadlines for parts of the project, which will be described in following section.

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
<th>Due Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Software</td>
<td>Firmware prototype Completion</td>
<td>Completed</td>
</tr>
<tr>
<td>Hardware</td>
<td>Hardware prototype Completion</td>
<td>Completed</td>
</tr>
<tr>
<td>Software</td>
<td>Establishing Data Transmission</td>
<td>19.02.2009</td>
</tr>
<tr>
<td>Hardware</td>
<td>Sound Engine Integration</td>
<td>19.03.2009</td>
</tr>
<tr>
<td>Software</td>
<td>Encryption Completion</td>
<td>26.04.2009</td>
</tr>
</tbody>
</table>
8 Resources

8.1 Human Resources
Since we are a small group, Configuration Management will be handled by all members. This task will not take much time of us considering development process of all system. Yet members have the following workload when configuring the project is needed.

Hakan Nizamoğlu: Redesign Focus, Optimization
Murat Toprak: Implementation, Hardware Module Redesign
Yiğitalp Ertem: Implementation, Technology Redesign
Saim Güveloğlu: Testing, Version Tracking, Documentation

8.2 Equipment
No special equipment is anticipated. PCs available to all team members should be sufficient to support the CM needs. This includes everything except for software version control which will use Trac. Trac accounts are supplied by the department. Trac will help us coordinating our work online and version control will be much easier by this way.
8.3 Facilities
We will not have any special facility for Configuration Management. By the way Configured Report will be edited by a PC and printed and saved for future use. We are not going to use any facility other than the software and hardware which we are using for the project now.

8.4 Budget
Configuration Management is a Level of Effort activity. Since we are not working on a budget for this project, the same situation applies for the Configuration Management too.