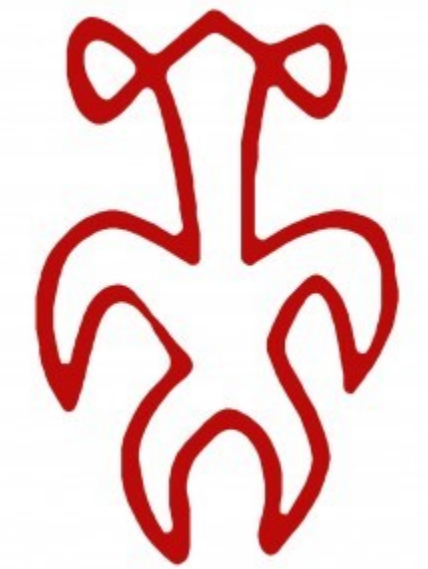
CONFIGURATION MANAGEMENT PLAN



RONGO RONGO

F. Aybike Avşaroğlu
Esra Gariboğlu
Osman Kaya
Önder Kalacı

Contents

6. Plan Optimization

1. Introduction

The GWSN project constructs a greenhouse monitoring system using wireless sensor networks. The major goal of the project is to increase the productivity and quality of the plants in a greenhouse without observing it whole day.

1.1. Purpose of Configuration Management Plan

The absence of coordination and planning makes developers face some unwelcoming consequences. Building the right coordination hierarchy in a software development process seems difficult, however if configuration management report is prepared, coordination problem will be out of question.

Configuration Management Plan (CMP) makes the integration of changes to our project easier and reduces the side effects of this changes. The purpose of this CMP report is to identify and describe a configuration management process for the GWSN project.

1.2. Scope of the Document

The scope of the document is to give a brief description about how the technical components are used, configuration process is done, and the distribution of the responsibilities among the team members.

1.3. Definitions, Acronyms and Abbreviations

CM: Configuration Management
CMP: Configuration Management Plan
GWSN: Greenhouse Wireless Sensor Network
HTML: HyperText Markup Language
JSP: JavaServer Pages
NesC: Network embedded Systems C
GCC: GNU Compiler Collection
IDE: Integrated Development Environment
UML: Unified Modelling Language
CSA: Configuration status accounting

1.4. Document References

- Software Configuration Management Plan (Online) Available:
http://www.nongnu.org/ghosts/developers/contributing/plans/scmp.html
- Sample Configuration Management Plan (Online) Available:
http://it.toolbox.com/blogs/enterprise-solutions/sample-configuration-management-plan-30170
- Cultivating Successful Software Development, Scott Donaldson and Stanley Siegel, Prentice-Hall, 1997.

1.5. Document Overview

In the **Introduction** part, our project, the CMP, and its purpose are explained briefly. The used abbreviations and references are given. In the second part, **The**

Organizations CM Framework,gives information about the organization of our team, responsibilities, and tools used in the project. The identification, configuration management and control practices, configuration status accounting and auditing are be described in the **Configuration Management Process** section. The next section, **Project Schedules and CM Milestones**, important dates and the schedule of our project are given in the light of the syllabus. **Project Resources** part gives information about the resources used. Lastly, **Plan Optimization** part concludes with the methods followed for optimizing CMP.

2. The Organizations CM Framework

2.1. Organization

Our team consists of four people and we are all responsible for various parts of the project. Each member has a contribution to the development phase with ideas, suggestions and interactions. Team members are:

- Aybike Avşaroğlu
- Esra Gariboğlu
- Önder Kalacı
- Osman Kaya

Software Development Team:

Software development for the hardware part: Önder and Osman are responsible for this part of the project, which is based on programming the nodes according to the leach protocol.

Web application development: Aybike and Esra are responsible for creating the interface to access and control the database and to monitor the greenhouse.

Testing Team: The main purpose of this team is to test the software continuously against bugs and unwanted results. All members are responsible.

Configuration Management Team: This team is responsible for preserving and updating configuration management organization.

Release Control Team: This team is responsible for controlling the market and making plans about the release of the project.

2.2. Responsibilities

Software Development Team: Each member has a responsibility in software development part. As mentioned above in the organization section team members partitioned into two groups. One group consists of Önder and Osman; another group consists of Aybike and Esra.

Önder and Osman are responsible for writing the code for our sensor nodes. Their program is based on the leach protocol, which has to be coded separately for nodes and clusters.

Aybike and Esra are concerned with designing the web application interface. JSP, HTML and Javascript are used for creating the web application.

Testing Team: This team tests the software continuously against bugs and unwanted results. Both software development team is responsible for testing their own program.

Configuration Management Team: All members are responsible for preserving and updating configuration management organization and change it if necessary.

Release Control Team: All members are responsible for controlling the market and making plans about the release of the project in control of sponsor firm INNOVA.

2.3. Tools and Infrastructure

NesC (Network Embedded Systems C): NesC is an extension to the C programming language designed to embody the structuring concepts and execution model of TinyOS. Since the sensor nodes we are going to use have very limited resources, we will use nesC to program them.

NesC Compiler: It is nesC compiler for TinyOS. It is Implemented as an extension to GCC and called via TinyOS wrapper ncc. Platform code implements API of macros and functions in C and output C code or object code.

NetBeans: Refers to both a platform framework for Java desktop applications, and an IDE (integrated development environment) for developing with Java, JavaScript, PHP, Python, Ruby, Groovy, C, C++, Scala and Clojure. We use NetBeans for developing the web application of the greenhouse monitoring system.

Mote-View: Mote-View is designed to be an interface between a user and a deployed network of wireless sensors. It makes it easy to connect to a database, to analyze, and to render sensor readings. However, we do not use it now.

Smart Draw: Smart Draw is a visual processor used to create flowcharts, organization charts, mind maps, project charts, and other visuals. It integrates with Microsoft Word, Excel, PowerPoint and Microsoft Project. We used SmartDraw for drawing data flow diagrams, UML diagrams and ER diagrams.

SVN (Subversion): SVN is a software versioning and a revision control system to maintain the data and allow editing and sharing if necessary. It is easier with SVN to update the data and track the status of the applications.

3. Configuration Management Process

3.1. Identification

The following Configuration Items are determined to be identified: source code, data and documentation.

3.1.1.Source Code

3.1.1.1 Source Code of Sensor Nodes

The coding of the star topology based protocol and the leach protocol on the sensor nodes are already finished successfully. We took many backups with proper names, identifying the current state, while writing the codes. Thus, we always have the chance to go back and look what new features are added at each increment. Moreover, at each backup the changes are commented.

3.1.1.2 Source Code of Gateway

The coding of the application running on the gateway is already finished successfully. It has only one version . We compiled with the object oriented design principles, especially with the open-closed principle, since, in the future we may need to change the code running on the sensor nodes, thus, we may need to change how the gateway reads the data. Except for that, there is no need to change the code on the gateway by itself.

3.1.1.3 Source Code of Web Application

There are always changes in the design of the user interface and the functionality provided to the users. Thus, we need to change the codes. For each new functionality added or for each new file added we took backups with proper names, identifying the current state. The process of improving the user interface and the functionality provided to the users are going on for the time being.

3.1.2.Data

The only database of the system is on a MySQL server on a laptop. The data tables in the database are stable. Except for the user information on the database,

there would not be any change provided that there is no change on the protocol running on the sensor nodes.

3.1.3.Documentation

Most of the documents are in our Google group(RongoRongo). But some of these documents are accessible from the project's web page. These documents are:

- Project proposal
- Requirements analysis report
- Initial design report
- Final design report
- Configuration management plan

3.2 Configuration Management and Control

Before started to use Trac and SVN, we had already finished the implementations of the protocols on the sensor nodes and the application on the gateway. Thus, what we are going to explain should be attributed to the web application.

3.2.1 Request for Change

At any time, any of the group member can request a change in any part of the application. Most of the time, the team gathers as whole. However, if we do not get that chance, since we are a small team consisting of four people, the requests are evaluated almost instantly via e-mail or phone. If the changes are approved by all the members, they are assigned in the Trac system with a ticket. The changes can be traced by using SVN.

3.2.2 Acceptance Or Rejection Changes

We did not explicitly assign a project manager or team leader. However, for each part of the project, there is a person who has more responsibility for that part. Although that person has more responsibility for that part, the decisions could be accepted or rejected by the all members of the team.

3.2.3 Implementing the Changes

When the change is accepted by all the members, the followings are done. Firstly, the source code is changed as required. Then, unit testing is done. Lastly, the changes in the source code is updated in SVN.

3.3 Configuration Status Accounting

Configuration status accounting (CSA) is the process of creating and organizing the knowledge base necessary for the performance of configuration management. In addition to facilitating CM, the purpose of CSA is to provide a highly reliable source of configuration information to support all project activities including program management, systems engineering, software development, modification, and maintenance.

Our accounting activities are as follows:

- Configuration status information reporting
- Keeping the configuration change history
- Milestone configuration reports
- Audit reports
- SVN and Trac activities

3.4 Auditing

Auditing plays an important part in our project. Auditing is regularly done by weekly meetings. At the start of weekly meetings, each member demonstrates his/her

assigned job to others. If these jobs are found deficient or lacking by other members, fix jobs are filed to that member.

Change requests are mostly handled with the Trac system. However, if there is a disputed change request, this request is discussed thoroughly. Justification of these change requests are required to get the approval of other members.

According to the project schedule, weekly plan for the next week is decided. Each member or group is assigned a part of the project. If there is a job that requires collaboration of members, meeting times are decided.

A weekly report is written together and uploaded to the group page. This report is used as checklist at the next week's audit.

Apart from the weekly audits, when the project has reached a milestone, a bigger audit is held. In these audits, general progress evaluation is done. Consistency check between the design report and the implementation is reviewed. Evaluation report is written for later use. A detailed schedule is written for the part which will be completed until the next milestone. Group discusses on difficulties that might arise on the next phase. Actions and solutions are produced for these possible difficulties. Members share their thoughts on things like how this version should be demonstrated for the best result, which parts would look good on the presentation and which parts would not look so good.

4. Project Schedules and CM Milestones

We partitioned our milestones into tasks to organize our project according to the deadlines.

- Prototype implementation (December 28, 2010)
- Prototype testing (February 20, 2011)
- Web application testing (March 4, 2011)
- User Interface design (March 15, 2011)

- ◯ Prototype release (March 30, 2011)
- ◯ Final test (April 30, 2011)
- ◯ Team presentation (May 10, 2011)
- ◯ Final release (May 30, 2011)

5. Project Resources

The resources of the GWSN project can be listed as follows:

- RongoRongo team members
- Mote-View
- Moteworks
- Netbeans
- Dreamweaver
- SVN : Revision Control System
- MySQL
- Crossbow Processor/radio board: XM2110CA IRIS 2.4 GHz
- Crossbow Sensor board: MDA100CB IRIS / MICA
- Crossbow Gateway/programming interface: MIB520CB IRIS / MICA USB PC Interface
- RongoRongo's Web Site

6. Plan Optimization

Development of the project will be done in accordance with the CMP. Everyone in the group has to join weekly audits, follow the mail group and inform the group if there is a change on the plan. We think that obeying the rules which we decided together is a

must. Anyone who does not follow the rules will get a warning. Also, weekly meetings with our assistants guarantees that we move according to our plan.