1. INTRODUCTION

1.1 Purpose

Software quality reveals itself most when a change in the software is needed. Modern day software is generally developed by large teams. And this large team develops a large project. Identifying, implementing and managing the changes is a task that is a challenge in itself.

1.2 Scope

This document is the Software Configuration Management Plan of WASD for the IronCurtain project. We try to establish an integrated process for identification documentation, implementation, auditing and approving of all changes, throughout the life cycle of our project IronCurtain.

1.3 Benefits

This Software Configuration Management Plan will contribute to the development of IronCurtain in the following ways:

- The integrity of the system will be maintained by the dependency tracking of the changes made on the configuration items.
- Coordination of access to configuration items will be managed, so that concurrent changes will not conflict.
- Changes will be controlled and managed effectively, so that only necessary and sufficient changes will be realized.
- Communication throughout the developers will be improved by reporting and auditing.
- As the consequence of all these, overall project quality will be improved by effective configuration management.

1.4 Reference Documents

The documents referenced in this report are:

- Software configuration management plan IEEE Std 828-1998
- WASD Final Design Report

1.5 Acronyms and Abbreviations

CVS: Concurrent Versions System

CM: Configuration Management

CCB: Configuration Control Board

CI: Configuration Item

RA: Requirements Analysis

SCR: System Change Request

2. TEAM'S CM FRAMEWORK

This section explains the responsibilities related to the CM tasks for the project.

2.1 Roles & Responsibilities

Our project team consists of 5 people, that number is few enough that change management will not require extensive number of meetings and/or communications. Each member of our team is also a member of the CCB.

Responsibilities of the CCB are:

- Identification of Configuration Items(CI)
- Establish baselines
- Approve or Reject SCR
- Locate any bugs caused by SCRs
- Track and Manage the SCRs
- Test the code produced by SCRs
- Inform the team members of a SCR
- Update the code in the CVS to make it include the result of SCR
- Make sure that change is well commented
- To track and update schedules accordingly

2.2 Resources

Resources can be classified as follows;

- The human resources for the CM are the members of our project. Configuration management experience of our team members are obtained by all members of our staff through Ceng350 - Software Engineering course. Also Kaan Kahraman, Mustafa Çoşkun and Furkan Kuru have taken Ceng451 - Information Systems Analysis and Design course.
- The necessary software resources are provided by our department. CVS will keep track of the changes made in the project history and will be useful in version management. Check-in and check-out facilities will make it possible to

return to a previous point in project history undoing changes. We will mostly be using integrated CVS function of Eclipse IDE. Also, we have PHProjekt project management software available for our use. It will help us in managing our project and resources.

2.3 Policies, Directives and Procedures

When a change is required to be implemented in the project, team members will obey to the team's rules about configuration management.

After a needed change is identified, it will be discussed during our frequent meetings, or on on-line resources, for example PHProjekt. For every change that is accepted, a team member will be assigned to implement the change. Then the change will go live on CVS, and a changelog entry will be added to our ChangeLog file. Our ChangeLog file will be in the following format;

```
[change_date] [Person implementing the change]  [his email]
      * [file(s) changed]: [description of the change]
```

3. CONFIGURATION MANAGEMENT TASKS

This section identifies the configuration management tasks and items, establishes the baseline management procedure, identifies the main baselines, and describes the repositories that are used.

These tasks mainly include the control of the configuration, identifying and resolving dependencies, notification of all members and documentation of the configurations to ensure the coordination between group members.

3.1 Configuration Identification

This section lists the categorization of the configuration items to examine them systematically.

Communication Items

We will use PHProjekt as a main communication medium. It will be used for binary file sharing (such as documents), for TODO list, for instant chatting and for talking over forums.

Documentation Items

Documentation is used to archive the configuration details to serve up to the group members and to support the configuration management process.

Software Items

We can divide the software items as follows:

- Database (SQlite and python bindings for it)
- Application Code (Python code plus web interface)

3.2 Configuration Items

This section shows our agreed configuration items and their identification. Every subsection defines description, usage, identification and processes for the items when configured.

3.2.1 Database

Database is an important component of our project because we keep our logs and user information in the database. There are two aspects in this issue as structure management and the test data. We use a file that includes SQL operations (which is util/createDB.py) to construct the database for the project. We don't keep the database itself (which as this is a SQLite database is a binary file) in CVS but we keep the database creation file in CVS. Configuration for this file must be discussed and agreed by the all members because of the possible inconsistencies. After the configuration, the configuration must be documented and send to the documentation archive.

3.2.2 Data Files

Data files that will be used in IronCurtain are rule and plug-in files. Plug-in files are also source files and they will be versioned in CVS. Rules are stored in a binary format so instead of storing these binary files in CVS, we will write a small Python script(file: util/createRules.py) which generates the rules.

3.2.3 Software Configuration

Our project consists of these parts:

- Core
- Logging
- Admin Web Panel
- User Management and Authentication
- Plug-in Engine
- Plug-ins and Rules

3.3 Baseline Management

Baselines are composed of CIs at a specific point in time. The baselines are used to control changes to the CIs throughout the life cycle of IronCurtain. We work on the creation of new baselines and approve any changes to a baseline together as a team. Baselines will be tracked, audited, retained and version controlled in the Concurrent Versions System (CVS) version control tool and PHProjekt project management tool.

4. CONFIGURATION CONTROL

This section describes the evaluation procedure of all change requests and change proposals and their subsequent approval or disapproval.

4.1 Change Control

Any minor change (like fixing small bugs, making changes to interfaces that do not cause inconsistencies in the program) can be done by an individual member without consulting the rest of the project team. Of course, such changes must still be properly explained in our ChangeLog file. If a team member wants to make a major change in the code, then this must be either discussed in PHProjekt forum, in our weekly meeting hour, or via email.

4.2 Version Control

Version control of source files is handled by CVS. Binary files are kept in PHProjekt's file management part with explicit version info (like, if we are working on a file named "UserManual.doc", first version will be stored as "UserManualv1.doc, second as "UserManualv2.doc", etc).

4.3 Release Management

Release management, which is the process through that software is made available to its users, is an important task for any maintenance project. Our project will have two releases, one initial and one final. Depending on the feedback we receive from our instructors and assistant after the first release, we will finalize the features for our final release.

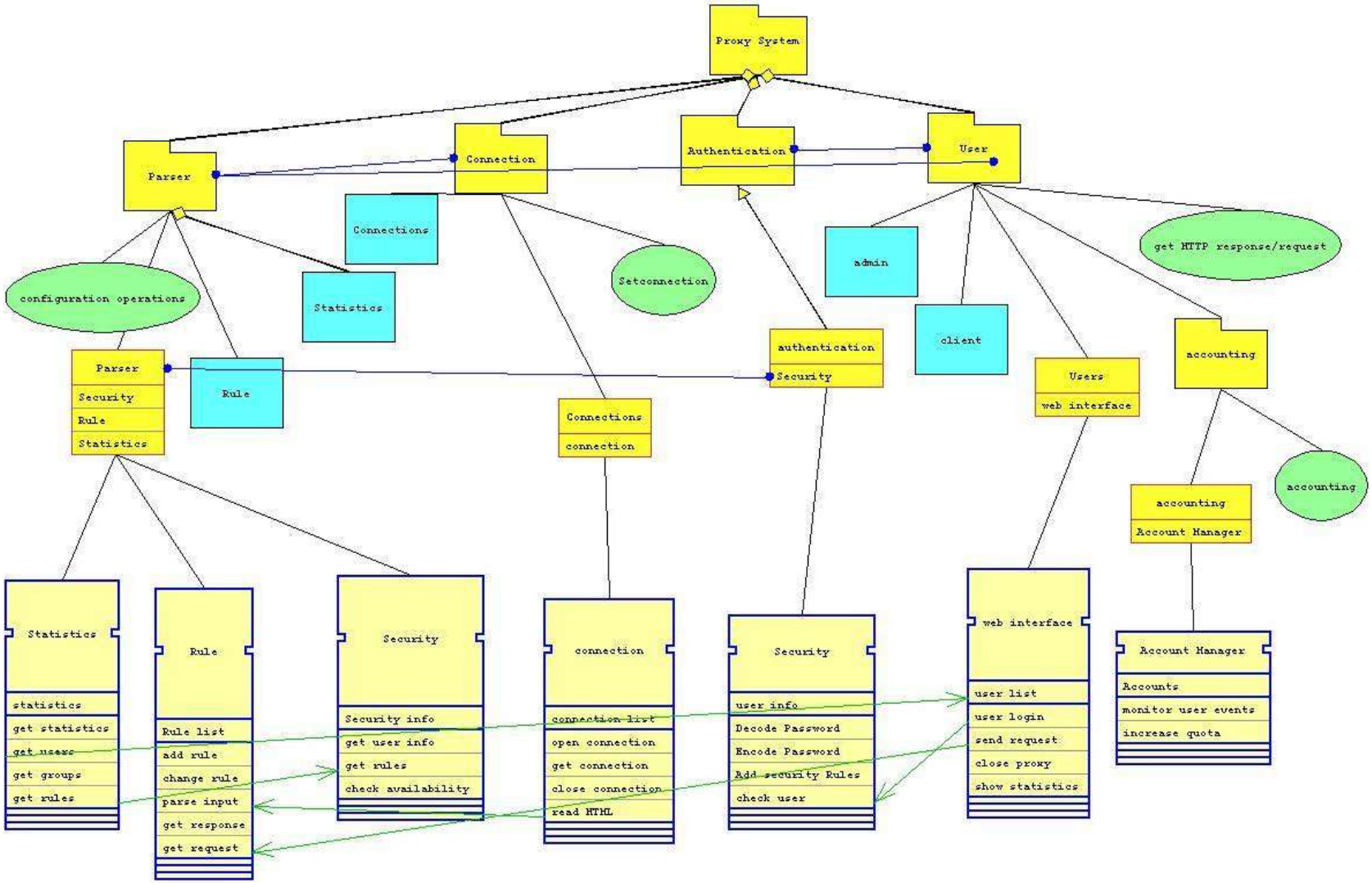
5. CONFIGURATION AUDITS AND REVIEWS

This section describes the audit and review procedure following the changes, milestones; so that every member of the development team will be aware of these changes, can comment on them and can contribute to their approval or disapproval.

The person responsible for the implementation of a change will audit the correct functioning of the recently added code. Furthermore, during our meetings we will check the results of the changes and apply peer review on them. If an inappropriate change is made, we will return to a previous version on the CVS.

6 APPENDIX

On the next page our project’s component diagram is provided.


Proxy System
Parser
Connection
Authentication
User
Connections
Statistics
Setconnection
admin
client
get HTTP response/request
configuration operations
Rule
authentication
Security
Users
web interface
accounting
Parser
Security
Rule
Statistics
Connections
connection
accounting
accounting
Account Manager
Statistics
statistics
get statistics
get users
get groups
get rules
Rule
Rule list
add rule
change rule
parse input
get response
get request
Security
Security info
get user info
get rules
check availability
connection
connection list
open connection
get connection
close connection
read HTML
Security
user info
Decode Password
Encode Password
Add security Rules
check user
web interface
user list
user login
send request
close proxy
show statistics
Account Manager
Accounts
monitor user events
increase quota