CEng 491
Senior Design Project and Seminar I

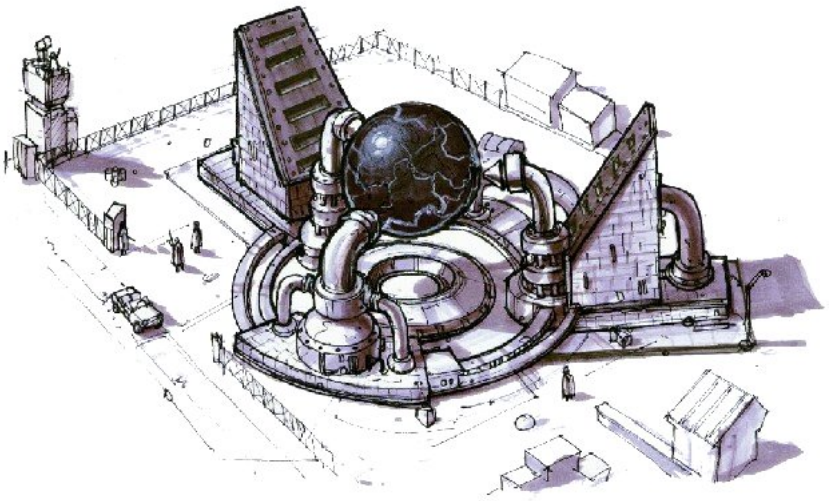
System Requirements Specification and Analysis Report

Project IronCurtain

by **Team WASD**

Gökdeniz Karadağ Doğacan Güney Kaan Kahraman
Furkan Kuru Mustafa Çoşkun

November 7, 2005



Abstract

The Internet has become an indispensable part of our life in conjunction with the growth of technology. People of all age groups and companies in all sectors use the Internet extensively. People spend hours in this enormous network and do most of their job by the help of the Internet. But as a result, this rapid growth of it has brought some important issues of security and controllability. Especially in the recent years, a great number of worms, viruses, Trojan horses have emerged and caused damage of billions of YTLs, not to mention the time and productivity that is lost while coping with these problems. In addition, many companies are faced with the loss of their employees' productivity due to distracting effects of Internet. So, beyond security, access control for their networks is very important to companies. In this project, we will try to address this issue, and find a creative solution that is feasible for a large number of organisations.

Contents

1 INTRODUCTION

1.1 Problem Definition

Most of the corporations reserve a serious amount of money and effort to maintain the security of their network, especially enterprises which deal with large amounts of money and information. In these networks, an important point of vulnerability is users. They can download files that can threaten these fragile networks on purpose or not. In addition, without controlling the network, the World Wide Web may be a factor which can decrease productivity of employees.

Traditional firewalls can be one of the solutions to overcome some of the problems. However commonly used firewall filters network traffic by only evaluating superficial data, like the IP address or the port. An application level gateway analyzes the network traffic in both directions extensively, not only the IP address but also the application level data in the traffic, like which web site is being visited or what words are included on the Web site, is inspected. The gateway then decides about what to do with the packet according to the security and access policies of the organization.

1.2 Project Scope and Goals

Our project aims to solve these issues for all kinds of networks by implementing an application level gateway with the features:

- Complete HTTP/1.1 support
- HTTPS support
- Decomposition of Entire Communication
- Plugin-based architecture for rules
- Multi-threaded (thread-pool) implementation
- Complete configuration over Web
- Secure User Authentication
- Logging and Accounting
- Notification of Administrator on Predefined Conditions
- Content identification

IronCurtain will be able to handle all request methods (like GET, POST, CONNECT) and all responses as defined by the HTTP/1.1 standard.[1] Also, IronCurtain will handle persistent connections, and will be able to compress(gzip) HTML even if the underlying browser (or web server, in a reverse proxy configuration) does not support it. Also, IronCurtain will understand and be able to perform over secure HTTPS connections. IronCurtain will decompose all the communication(url, headers, HTML page) that passes over it, and will make all applicable data available to be used by the plug-ins. Each plug-in will define a mechanism that can be combined to form a rule template.

IronCurtain will log every action of users and system. It will analyze the logs and provide statistical data from it. This data will then be displayed graphically on IronCurtain's web page.

When a predefined condition occurs, IronCurtain will alert the administrator by SMS or e-mail service.

IronCurtain will process the text contained in HTML pages to identify content. But we are not quite sure about how to handle this, currently.

1.2.1 Usage Areas

- **Companies**

  Today nearly all the business sectors need computers and Internet access for maintaining their efficiency and profitability. And companies' employees are using computers and have connection to the Internet. As we know, even though Internet has too much beneficial information, it also has another side, which is distracting and harmful. So, in a company it is essential that a gateway restrict the users access to the some part of Web. And IronCurtain will track the surfing activities of employees and scan the returned web pages for any malicious action. The companies are now also hosting their own Web servers in their LAN. Our proxy will also be a defender for hostile connections to these servers from outside. IronCurtain will also hide the computers behind it. It may look like only one or two machines to the outside world, but in fact there may be several hundred employees surfing the web through it.

[1] http://www.w3.org/Protocols/rfc2616/rfc2616.html

- **Universities**

  Universities have thousands of users and computers. A direct connection to the web is obviously insecure. IronCurtain will be a layer to the outside. The user actions will be logged and in case of any serious problem, the user will be identified easily by using tracks in the logs. The access of the students to some certain sites will be blocked using blacklists. Universities also have too many web servers and there are many connections to these servers. These servers are normally open to attacks, and a layer with our web gateway is a proper defense method for it.

- **Other Areas**

  In fact, IronCurtain will be designed to run in any network. IronCurtain will both be used as a protection for the LAN to hostile actions from outside and control tool for the members on the LAN. The system will protect the web servers being as a layer in front of the server to the outside connections. It will be a controller for the clients inside the LAN.

2 Research

2.1 Comparison of Programming Languages

To decide the programming language that we will use in IronCurtain, we looked at the common languages: C++, JAVA, .NET and Python. The advantages and disadvantages are as follows:

Java is a high level programming language that allows faster development because of its broad libraries and easy to use IDE tools like Eclipse. The Java libraries have a well-arranged structure and documentation. Java also offers operating system independence by running the program code on Java virtual machine. But, there may be some performance problems because of this virtual machine. However, we came across to many gateways written in Java. In fact, CPU bound modules can be written using another programming language. Moreover, IronCurtain will be input output bound as every gateway application; because the network connections are not fast enough and the data transmission part becomes the bottleneck.

Python is a dynamic and very high-level language with lots of useful constructs (such as lists) built in the language. It is a compact and concise programming language. Python is an interpreted language, which may cost both in CPU time and in network latency. However, during our research, we found too many proxies written in Python. As explained earlier, IronCurtain will be I/O bound and the CPU overload will have insignificant effect on the performance of the application.

.NET is a new emerging technology developed by Microsoft. It has a great advantage that it supports many languages to work on top of it, such as C#, Visual Basic .NET, Boo (a dialect of python), even Java. However, we are not familiar with .NET platform and we are worried if its libraries are not as sufficient as Java.

C++ is fastest of all of them, which is important for us because we do not want to introduce any additional network latency in our program. However, programming in C++ is the hardest among all and the libraries are not as extensive as the others.

2.2 Similar Programs

For our research, we examined three programs.

2.2.1 WebCleaner

http://webcleaner.sf.net

WebCleaner is a filtering HTTP proxy written in Python for the most part. It is still under active development with its latest release on 10/11/2005.

WebCleaner is at its core an asynchronous program. It uses Python's asyncore module for its operation. Since, we are also concerning deployment of our gateway on the server-side (in the form of a reverse proxy), this approach may not be really suitable.

WebCleaner has all the standard features that most other filtering proxies have out there. It allows you to arbitrarily modify(add, remove, replace, etc.) both HTTP request and HTTP response headers. For example, you can remove the browser identification header, thus hiding your browser and underlying OS.

WebCleaner has complete HTTP/1.1(so it has persistent connections and can compress and decompress documents on-the-fly, even if the underlying browser does not support it) and HTTPS support. When a request for a

HTTPS web site comes, WebCleaner spawns an SSL gateway for communication. Unfortunately, WebCleaner replaces the web site's SSL certificate with its own, so it is doubtful how secure this method really is.

By using PIL (Python Imaging Library) for image manipulation, WebCleaner can analyze and make changes on the images. For example, it can 'stop' animated GIFs, only showing its first frame. WebCleaner can process high-quality images, reducing their quality in order to save bandwidth.

The features that we have mentioned so far are more or less available to many other proxies. One of the features that sets WebCleaner apart is that has an HTML parser (written in C) which parses the web page completely. With this, WebCleaner allows you to make modifications of the HTML document itself. For example, one can remove the annoying ¡blink¿ tags with simple rules. Or, you can identify and remove tags with known security problems.

Another peculiar feature of WebCleaner is that, it has an JavaScript engine embedded in. It uses the Mozilla SpiderMonkey JavaScript engine. Since, JavaScript is executed in the proxy and not in the browser, with a fine-grained security policy(for example, one can use SELinux on Linux to block WebCleaner from writing to any file but its logs and configuration files), this can block out most of the potential security flaws of JavaScript.

WebCleaner can integrate with the ClamAV anti-virus module and also has NTLM proxy authentication. NTLM (NT Lan Manager) is a authentication protocol originally designed for Microsoft Windows-based operating systems but Mozilla Firefox and Apache Web Server also has support for it.

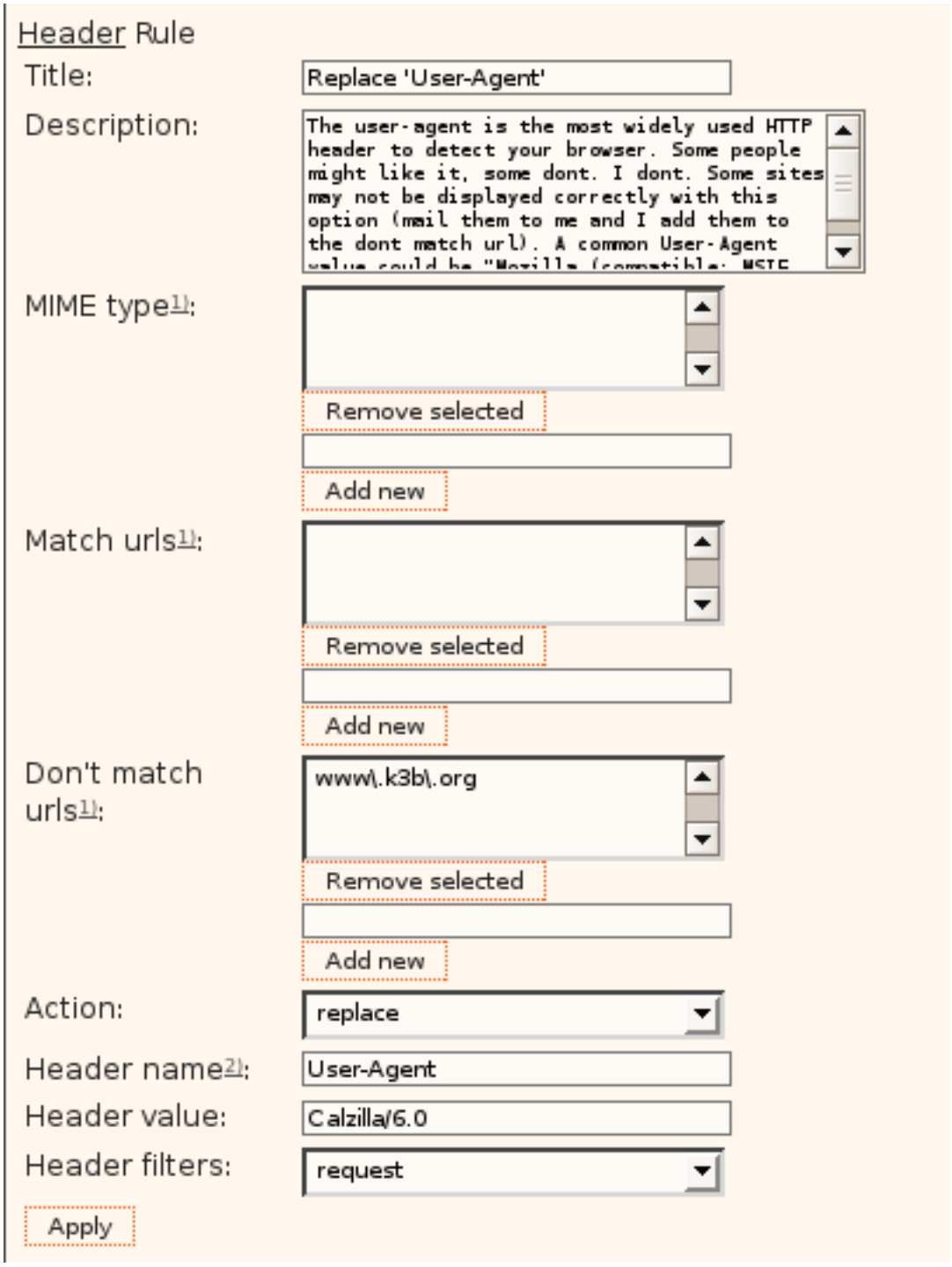
WebCleaner can also use SquidGuard blacklists for blocking web pages. It also allows the administrator to rate content per-host, and allows creating policies for blocking the site if its rating is too low.

Overall, WebCleaner is a good HTTP filtering proxy. It is very feature-rich and is under active development. Unfortunately, it has some bad sides too. First of all, there are a lot of web sites on world wide web, that has a broken HTML page. If you ever look at the source code of any web site (for example, www.osnews.com) you will probably see lots of errors, like unclosed tags, nonstandard tags, wrong document type, etc. WebCleaner's HTML parses is too naive, it simply ignores tags that it can't parse. With its default policy set to REMOVE HTML tags that it doesn't understand, most web sites become unnavigable. Also, WebCleaner does not have an automatic content filtering mechanism, instead, it relies on the administrator to rate content per host, a task which may prove difficult for large companies, where each employee has his own set of preferred web sites. And finally, it seems that WebCleaner is designed for personal home use, not for deployment on large companies, schools, or as reverse proxies in front of web servers, all of

which is part of our goal.

	Name	Short description
☐	BinaryCharFilter?	Replace binary characters from HTML documents with their ASCII equivalent
☐	Blocker?	Block URL requests and show a replacement URL instead.
☐	Compress?	Compress HTML pages.
☐	GifImage?	Deanimate GIF images.
☐	Header?	Add and replace HTTP header values.
☐	HtmlRewriter?	Modify HTML pages.
☐	ImageReducer?	Replace images with small, low-quality JPEGs.
☐	ImageSize?	Replace images having a certain width and height.
☐	MimeRecognizer?	Guess a missing content type.
☐	RatingHeader?	Evaluate content rating HTTP headers.
☐	Replacer?	Replace regular expressions in HTML and text data.
☐	VirusFilter?	Scan content data for viruses.
☐	XmlRewriter?	Modify XML pages.

WebCleaner has a set of predefined plug-ins. Choosing one(for example, Header), activates it. Then we can configure what headers we wish to add or remove.


Header Rule
Title:
Replace 'User-Agent'
Description:
The user-agent is the most widely used HTTP header to detect your browser. Some people might like it, some dont. I dont. Some sites may not be displayed correctly with this option (mail them to me and I add them to the dont match url). A common User-Agent
MIME type[1]:
Remove selected
Add new
Match urls[1]:
Remove selected
Add new
Don't match urls[1]:
www\.k3b\.org
Remove selected
Add new
Action:
replace
Header name[2]:
User-Agent
Header value:
Calzilla/6.0
Header filters:
request
Apply

2.2.2 Willow

http://www.digitallumber.net/software/willow/

Willow is basically a asynchronous response-caching HTTP proxy server. Willow is also written in Python. Its features do not match up to other HTTP proxies like WebCleaner, but Willow has a very unique feature that sets it apart from other proxies. It has a smart content filtering mechanism.

The rationale behind Willow is simple. Keeping blacklists and whitelists for content management on World Wide Web is obsolete. The number of web sites are measures in billions (and counting), so it is simply impossible to rate content manually. Instead, Willow keeps a (reasonably small) list of good and bad sites, and when a request for a new web page comes, the web page is classified on the fly based on information gathered from the previous web pages. To this end, Willow employs a Bayesian algorithm (which, from its looks, looks similar to those that are used for intelligent email spam filtering). Since, each new web site is also added to the good list or the bad list, an administrator can start with an empty good and bad list, surfing the web and rating contents, and Willow will slowly learn the pattern of the allowed and blocked sites.

Willow is under development with its latest release on 9/29/2005, but the pace of development seems slow compared to WebCleaner.

Both Willow and WebCleaner are configurable over the web.

2.2.3 Aladdin eSafe

http://www.esafe.com

Unlike Willow and WebCleaner, eSafe is closed source. Unlike Willow and WebCleaner, eSafe is a true application level gateway, not just an HTTP proxy. It is both commercial and proprietary, there is no free download for it (neither binary nor open-source), so the following information is mostly a compilation of their white papers and information on their web sites.

eSafe does not include a content filtering mechanism in the program, instead it has a simple whitelist/blacklist database. But, on their web site, it is claimed that this database is updated regularly by an intelligent crawler which classifies about 100.000 page per day.

Among features that are interesting to our project, eSafe blocks HTTP tunnelling attacks, so that you can't bypass its HTTP filtering mechanism.

eSafe can block known malicious code attacks, such as buffer overruns and SQL slamming. There is no clear information on how eSafe updates its

malicious code database, but it is probably kept in some database on web.

eSafe can recognise most P2P programs, like Kazaa, Imesh. eSafe can stop P2P programs running in the network from functioning completely(so that, a program like Kazaa can not even connect to its network) or from sharing or downloading some files(like, trying to share an important document for company or downloading an mp3 file without DRM protection). eSafe also recognises popular IM networks, like MSN Messenger or ICQ, and can block access according to a security policy. Also, it stops spyware and adware programs. Spyware programs simply can't establish connection to their network, thus guaranteeing the animosity of the company.

eSafe seems to be an excellent application level gateway. But, of course, all these features may also be just marketing talk. Since we never had the chance to download and test it for ourselves, we can't say how useful it really is.

2.3 Survey

We interviewed Selçuk Tuna(ST), system administrator in *Aydın Yazılım* and Serhat Akçabayırlı(SA), system administrator in *Cybersoft.*

Q. Do you use an application level gateway to control your network?

ST. Yes we are using one.
SA. Yes, we are using *WebSense.*

Q. What are the features that you like and dislike about your current system?

ST. We are using the gateway to inspect and report users' web activities, like sites visited, bandwidth used, web usage duration. These information are stored in the logs, and if a previously defined condition happens, we are alerted. This is satisfactory for us, but I am not happy with the reporting capabilities of the system.
SA. I am very happy with the current system, it has a large database of good & bad addresses that is constantly being updated. It controls all of the network, blocks instant messaging and file sharing applications and it allows per user settings.

Q. Mr. Tuna, do you take proactive action against the alerts detected. And how do you think the reporting capabilities of the system can be improved?

ST. No we only log the actions of users and have alerts sent to us, we do not block user's access or put limitations on them. Reports of the system can be extensively detailed, and it should be presented in an easy to understand way. Also, the reports are generated at predefined intervals using the logs. It would be better if they were updated real-time.

Q. Is it easy to create new rules in your system?

ST. It is not easy, it requires detailed technical knowledge of TCP, HTTP, XML. But I have no problem in dealing with these.
SA. It is easy to add new URLs or keywords to the existing ones.

Q. Do you use a reverse proxy in front of company servers?

ST. Our servers are in a DMZ. Traffic to the servers are routed through a firewall and intrusion detection system. We are not filtering the traffic at application level.
SA. I am not authorized to answer this question.

Q. Are you enforcing rules based on the user, ip or group. Is this an important feature for you?

ST. We are not enforcing rules. Our logs record ip information as a reference.
SA. Absolutely, this is a must-have feature for us. We apply different rules for different users.

Q. Would you rate the importance following features, 5 being the most important and 1 being the least?

Feature	Serhat Akçabayırlı	Selçuk Tuna
Configurable over a web page	2	2
User or group based rules	4	4
User or group based logs	3	5
Easy to create new rules	1	3
Content Filtering	5	1

3 Requirements

This section gives details about functional, non-functional and minimal software / hardware requirements.

3.1 Functional Requirements

Functional requirements can be listed as:

- **Enable or Disable IronCurtain**

  In some cases, it may be necessary to suspend IronCurtain completely. There will be an option to disable the gateway actions. Then all connections will have direct access without any inspection. Anytime with the enable option, system will continue to control and secure the traffic

- **Web Content Control**

  In Web pages, apart from text, there exist images, multimedia objects such as Flash, ActiveX, Java frames. These objects may contain insecure code fragments. To avoid them there will be options to filter the incoming pages. Ads, animated GIFs, Flash and ActiveX objects can be disabled by authorized user. Incoming or outgoing web content will be analyzed based on HTTP request and response headers, the HTML page, images, references to other sites. Basically IronCurtain will track and parse all the communication between the server and the client. Then it will expose the data to our plug-in environment.

  In IronCurtain plug-in environment the proper module will perform on the parsed data, it will apply the pre-defined rules.

- **Add/Remove Users or Groups**

  IronCurtain will allow adding and removing of users or groups. It will be possible to configure access and accounting per-user and per-group.

- **Control User Accounts**

  Every user in this system will have an account saved in the database. There may be alternatives for identifying users; It may be IP address or MAC based authentication, this would be more useful for a transparent proxy. It may use an existing authentication scheme for proxy authentication, Kerberos or NTLM.

- **Secure Login**

  Administrator is required to enter his/her ID and password before using our proxy server. The entered login information is checked against the stored information: if they match, the user is logged on to the system; if they do not match, the system should print an error message and again present the login screen.

- **Add/Remove Rules**

  Through the web interface, the administrator will be able to add arbitrary new rules, such as modifying HTTP headers or blocking images. The administrator will also be able to remove them from the system. When there is any change in the rules, the system will take action immediately. IronCurtain will have a plug-in based architecture. Rule plug-ins, which will be written in our programming language or a simple script language, will define the mechanism. Rules, which will be defined over the web interface, will use plug-ins as their underlying template and will define the policy.

- **Check the bandwidth usage**

  The bandwidths of the networks and Internet access are limited. So, too many unnecessary downloads or uploads of some users lead to slow connections, delays, denial of service problems for other users and programs. In order to avoid this, a proper logging of bandwidth usage should exist in this system. For every user this logged bandwidth usage data will be displayed and there will be an option to limit the upload or download quota of a specific user. This is necessary for preventing users from exploiting the bandwidth and suffering other users on the net.

- **Check user actions**

  There will be restrictions and some controls to prevent users from being distracted. In order to do this control all the user actions will be logged. Then the visited web sites and durations on these sites will be displayed. And it will be possible for authorized user to block any users access to some sites. There will be also black lists. The sites in these lists will be unreachable by any user. The administrator will be able to change this list manually by adding or removing sites.

- **Check statistical data**

  All of the user actions, or a part of it if the administrator configures as such, will be logged extensively. Statistical data will be generated

from the logs of users actions, and it will be updated in real-time. A table will be displayed showing the web sites visited, a counter and a duration field. This information can be used for a general view of the Internet usage. The general problems such as bandwidth bottleneck can be detected. Moreover, it can provide critical information in case of hostile actions, viruses and worms. The links that nearly all the computers connect and a counter with a high value for that link may indicate a worm. It will be possible to take action by adding that link to black list, before its unwanted consequences. Statistical data for connections will also be displayed as diagrams.

3.2 Non-functional Requirements

Non-functional requirements can be listed as:

- **Scalability**

  Since Web Access Control is an important issue for large companies too, our software must scale equally well to any number of users. Even, in large corporal networks, users should face minimal or no latency in their regular Web usage.

- **Maintenance**

  Maintainability is required to be able to solve issues encountered before and after enabling the system. Later on it may be requested to add new functionality or enhance the system. Therefore, the source code must be maintainable.

- **User Interface**

  The user interface will mainly be a website for the administrator to configure the system. The interface must be robust. Since, administrators will want to have a fine-grained control over their network, the web interface must enable the administrator to configure both macro details like the size of the thread pool or micro details like blocking a specific user from accessing a specific web site.

- **Security**

  There are two aspects of security. First, the internal network which clients connect to our software may not be secure. Therefore, we must provide methods (such as, NT Lan Manager[2] or Kerberos[3] to enable se-

[2] http://en.wikipedia.org/wiki/NTLM
[3] http://web.mit.edu/kerberos/www/

cure user authentication over insecure networks. Secondly, the Internet is, unfortunately, filled with people with malicious intents. IronCurtain must be safe from malicious code attacks like buffer overruns(which, we think that coding in a high-level language like Python will help enormously) itself, and also it must provide methods that allow administrators to define rules to block such known attacks.

Also the people that are being protected and controlled by the IronCurtain may try to bypass IronCurtain. The system must be resistant to such attempts, and should log and alert the administrator also. To prevent bypassing IronCurtain, there are things an administrator must do; like blocking all outgoing HTTP connection attempts, an allowing only the IronCurtain's access to outside.

- **Platform Independent Use**

  IronCurtain should be operational and functional across different platforms like UNIX, Windows, etc.

- **Reliability**

  The system should be as bug free as possible. All sub components should work asynchronously, so that any delay caused by one of the components should not block other components work flow.

3.3 Minimal Hardware Requirements

Minimal hardware requirements for our project are: A PC with the following configuration will be needed:

3.3.1 Development

A PC with the following configuration will be needed:

- Intel Pentium IV 1 GHz
- 512 MB DDR RAM
- 40GB Hard Disk Space
- Internet Connection

3.3.2 End User

- Intel Pentium IV 2 GHz
- 1GB DDR RAM
- 100MB Hard Disk Space
- Local or Wide Area Network

3.4 Minimal Software Requirements

Software requirements for the project are divided into categories:

3.4.1 Development

- Recent Linux Distribution such as Ubuntu 5.10, or Windows XP or newer
- Full Installation of Python 2.4
- SPE Python IDE or Eclipse Installation with PyDev Plug-in
- Navicat (MySQL Manager)
- LaTeX Suite

3.4.2 End user

- Recent Linux Distribution such as Ubuntu 5.10, or Windows XP or newer
- Full Installation of Python 2.4
- MySQL Client
- Web Browser (Mozilla Firefox, Microsoft Internet Explorer)

3.5 Data Flow Diagram

Our data flow diagram describing the working mechanism of IronCurtain can be found in the appendix. Here is a more detailed explanation of the diagram.

- **DFD Level 0**

  As seen from the diagram,we have three users: server,client and administrator. The main functionalities of the users are shown on the diagram. According to the requests IronCurtain writes some information to the database or process it and give a response to the users.

- **DFD Level 1**

  This is a more detailed diagram of IronCurtain. The main parts of IronCurtain is filtering, control mechanism, configuration and logging process. These parts communicate with users and eachother.

- **DFD Level 2:Filtering**

  This is a more detailed diagram of the filtering part in IronCurtain. This is the most important part because all the requests and responses are processed here.First the requests and responses are parsed and then it is analyzed. In analyze part the validity of the request and response is checked by the control mechanism. After analyzing the processed request or response are sent by dispatcher to the users.

- **DFD Level 2:Control Mechanism**

  Control mechanism briefly checks the access rights and user authentication. It is connected to all the other modules. It gets the info from the other parts and checks whether this user have the right to do that action or not. According to that it sends an answer of approval/disapproval.

- **Configuration**

  Configuration part is used for changing the properties of users, rules and main configuration of the proxy. It checks the access rights of the user by using control mechanism.It writes the information to the database. Only administrator user can use that part of the IronCurtain.

- **Logging**

  Logging process is used for logging all the activities of the user and also alerting if there is a violation of a predefined condition. All the request and response info is provided to logging process by filtering part. The logging process sends log and statistical info to the database and administrator user.

4 PROJECT SCHEDULE

Project scheduling is one of the most important sub-tasks of project management. For this project, scheduling activities and tasks are given below.

4.1 Process Model

We are developing this application for our senior project; thus our project will follow the linear(waterfall) model of development. We first analyze the problem, and then design our solution. In the development phase we will have a two pass implementation strategy. The first will be a prototype and concept implementation. The second will be the final product. During the development plan, we will use a more spiral like process model, to accommodate for conditions that arise during the implementation phase.

4.2 List of Deliverables

Documentation

- Proposal Report
- System Requirements Specification and Analysis Report
- Design Report
- Help for Web Site
- Help for Proxy Usage
- Full Tutorial Construction

Code

- First Pass Implementation of a HTTP Proxy

  This proxy will be a simple single-threaded, synchronous proxy that does not do any sort of filtering or access control. Still, it will support most of the HTTP/1.1 standard, such as persistent connections, and will be able to handle most HTTP responses (like 200 OK, 404 NOT FOUND, 302 FOUND). Also, it will parse all the occurring communication. This will be mainly useful as a base implementation for the later versions.

- Database

- Web Site
- More Robust Implementation
  A new implementation, capable of handling much more requests.
- Initial Plug-ins
- Initial Rules for Normal Proxy Usage
- Secure Authentication
- HTTPS Support
- Completed Plug-ins
- Completed Rules for Normal Proxy Usage
- Rules for Reverse Proxy Approach

4.3 Functional Decomposition

Requirements Specification

- Interviews with Customers
- Internet Search
- Questioning Potential Users

Learning the Languages and Tools

- Determining the Proper Languages
- Determining the Tutorials and Manuals
- Studying the Tutorials

Database Construction

- Creating Tables for Users, Groups, Hosts, etc.
- Working Out Inter-relations Between Tables
- Complete Database Implementation

Interface Construction

- Initial Web Site Design

- Administrator Console Design
- Web Site Implementation

Testing

- Database testing
- Web Site Testing
- Proxy Testing
- Full Package Testing

5 USAGE SCENARIOS

5.1 Scenario 1: Access Control

Murtaza, a computer science student, installs IronCurtain in his home system. He visits SourceForge.net regularly in order to check the latest news of his favourite projects, and also to learn about new projects. He downloads many new projects to his computer to read the source code. He gets very annoyed of that fact that each time he clicks on a file, SourceForge.net first directs him to a site to choose the mirrors and then waits a few seconds before the download starts. There is an option to select a default mirror, but this option uses a cookie, and also does not eliminate the wait time. To solve this, Murtaza writes a rule. All SourceForge.net downloads' addresses have a similar pattern. With the rule, whenever the proxy sees the GET request for an address with such a pattern, it automatically makes a GET request to a defined mirror to start the download immediately. The mirror selection page, need for downloading cookies and wait time are eliminated by using only IronCurtain.

5.2 Scenario 2: Accounting

Thomas Anderson, who is one of the administrator of the Metacortex company, installs IronCurtain to guard their network. He sets up rules accordingly so that all the known game sites are automatically blocked to ensure that everyone in the company spends his working hours actually working and not playing games. Some time later, when Thomas Anderson is going over the logs, he notices a peculiarity in the system. One of the employees spends a considerable amount of his time in one web site. Visiting the web

site, Thomas Anderson sees that this is a new game site. Thomas Anderson adds this site to the list of blocked hosts, also warns the executives for the misbehaviour of that employee.

5.3 Scenario 3: Reverse Proxy

IronCurtain is used for guarding a web server, which Ahmet publishes his blog from. In order to be able read people's responses to his latest blog, Ahmet sets up a comment section. The blog software he uses reads the comment from a location and shows a single HTML page which contains both his own writings and the comments. While a spambot crawls the web, it finds this page, and decides to post a spam comment. Ahmet, who gets bored of these type of comments, writes a rule. With the rule, IronCurtain is made to recognize the web page used to post comments and blocks comments with special patterns in them.

5.4 Scenario 4: New Rule Definition

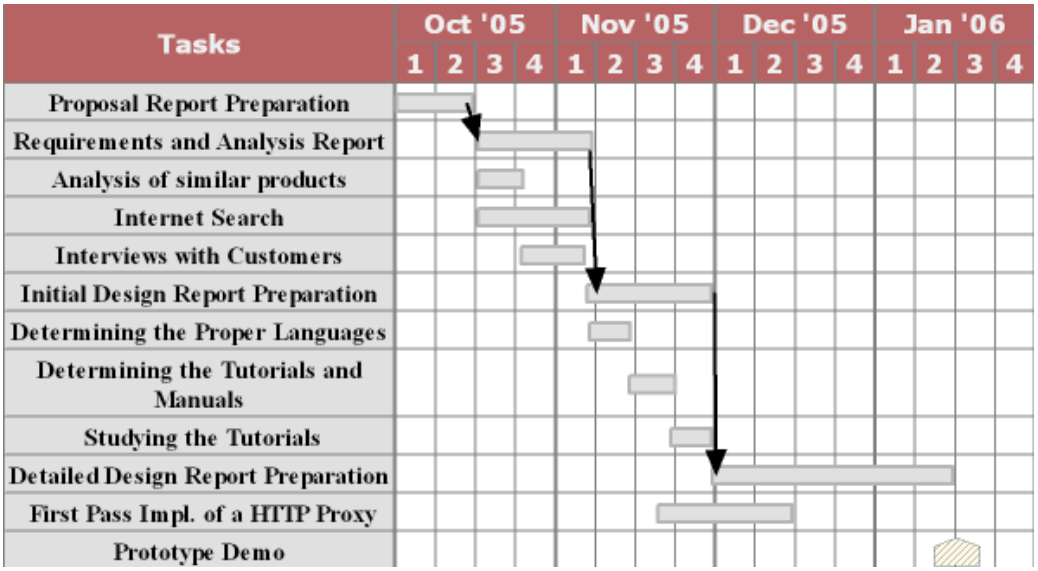
Nurettin wants to write a new rule that replaces Flash animations with a placeholder. He only wants the Flash animation to be viewable after he clicks on the placeholder. So, he logs on to IronCurtain's web interface. He clicks on the link to set up the new rule. Unable to find any appropriate templates, he decides to write his own template. The template goes like this: "If the OBJECT tag contains a PARAM tag with VALUE field with a string that ends in .swf then download the flash to a file and replace the flash link with the link to this file."

A APPENDIX

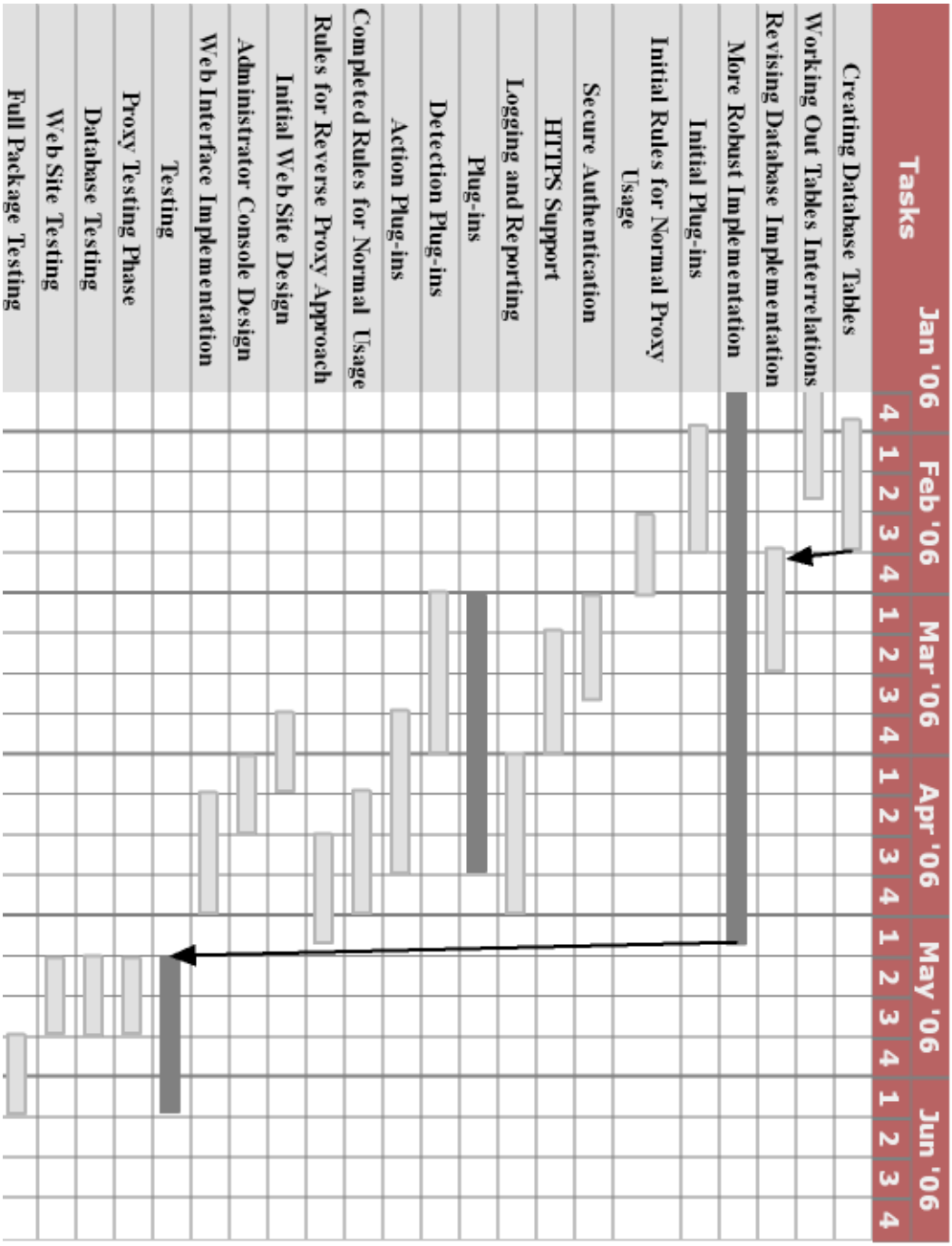
A.1 What's with the name?

Our project's name IronCurtain, contrary to what one may initially think, is not a reference to the boundary dividing Europe ideologically and physically. It is a reference to the game Command & Conquer: Red Alert 2. In the game the Soviets could build the Iron Curtain which would make vehicles invulnerable. The painting on the cover is a concept drawing of the Iron Curtain, taken from http://firingsquad.com

A.2 Gannt Chart

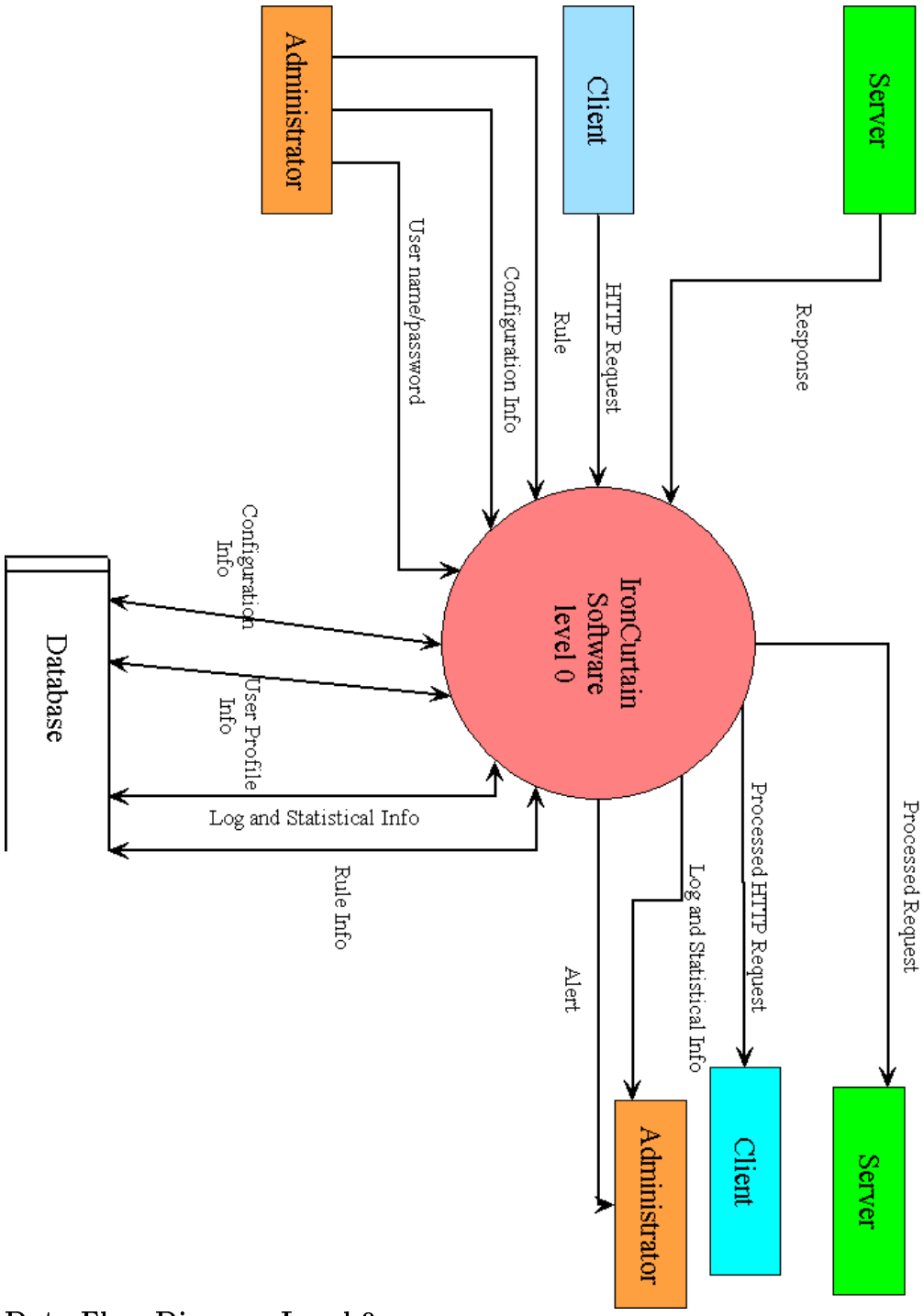


Gannt Chart Part 1

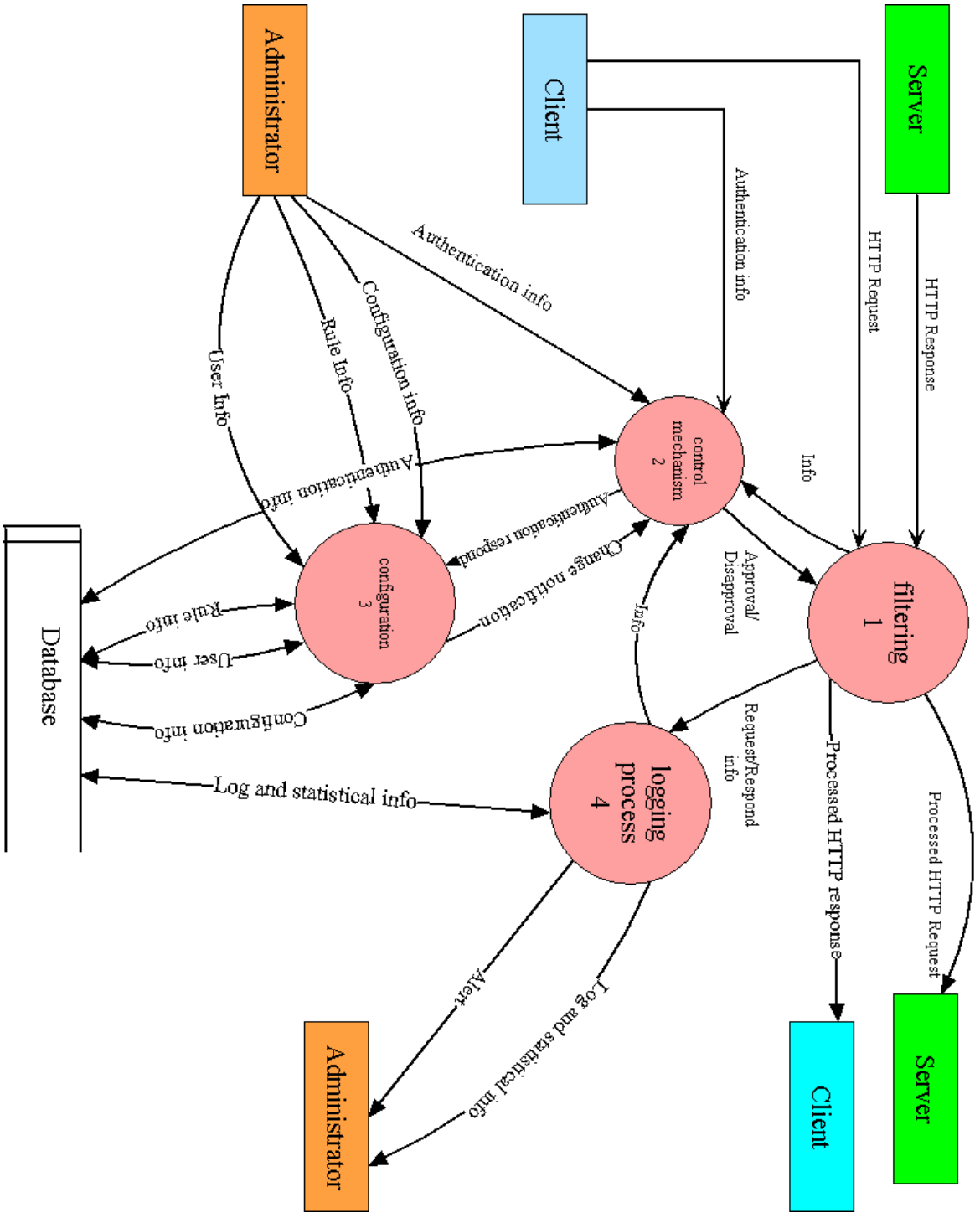


Gannt Chart Part 2

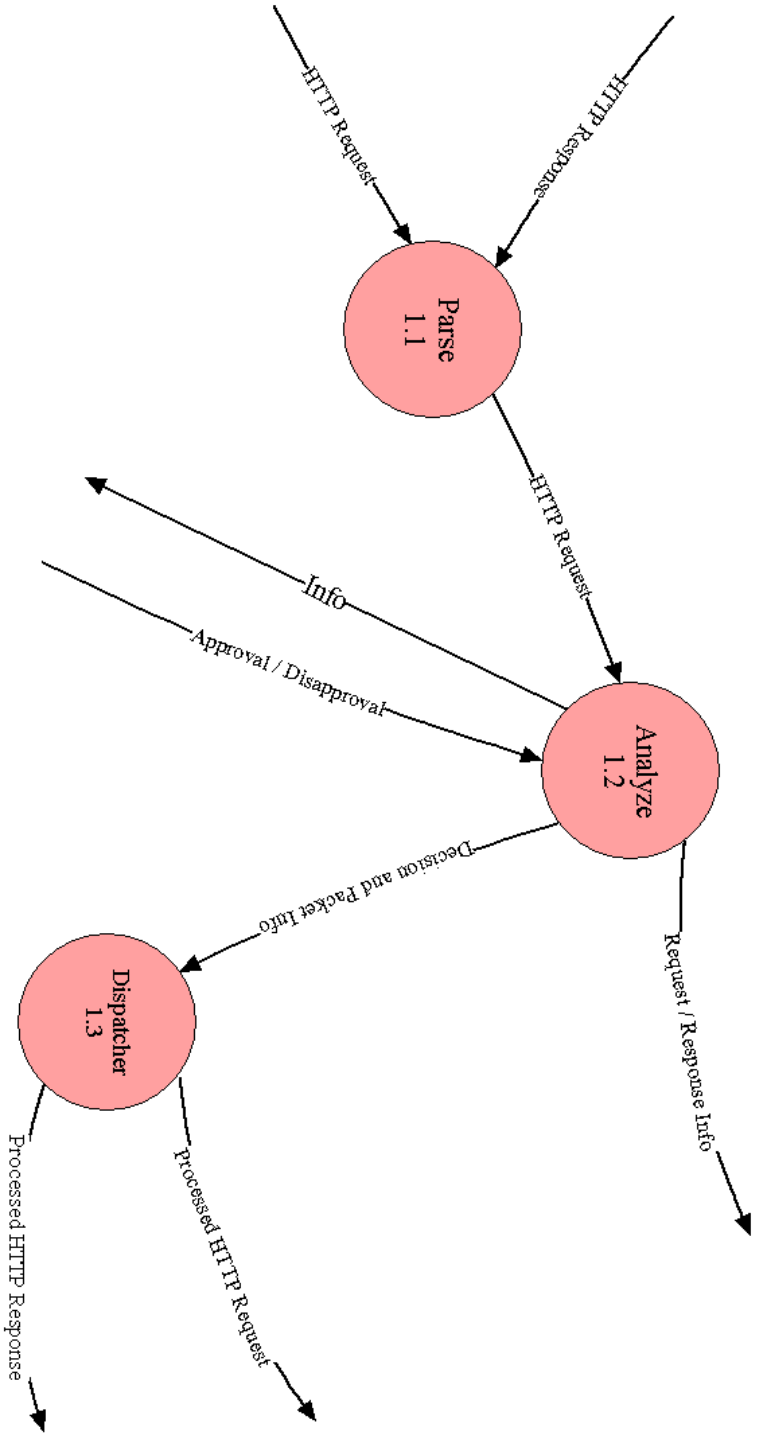
A.3 Data Flow Diagrams



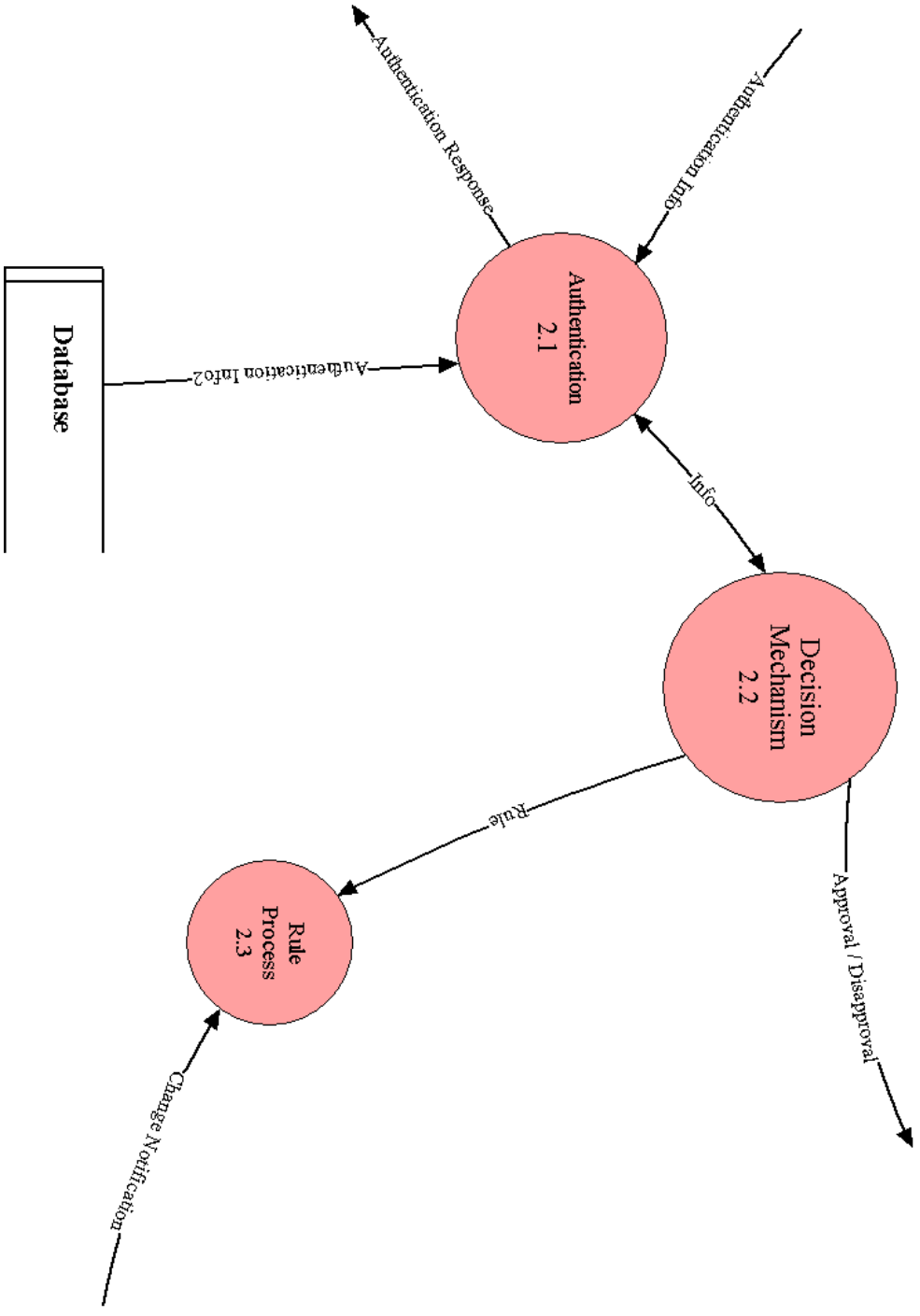
Data Flow Diagram Level 0



Data Flow Diagram Level 1



Data Flow Diagram Level 2: Filtering



Data Flow Diagram Level 2: Control Mechanism

A.4 State Transition Diagram