

Software Requirements Specification (SRS)
Rate Adjustment by Managing Inflows (RAMI)
Team M

Daniel Connor (connor) Chris Edwards (caedwa) Rod Howard (rihoward)
Maya Muthuswamy (mayadm) Lars Yencken (lljy)

October 30, 2002

Maintained By: Rod Howard (rihoward@students.cs.mu.oz.au)
Version: 1.2

Abstract

A requirements specification for the development of RAMI, a TCP/IP flow control module for the Linux kernel and a suite of network evaluation utilities.

Contents

1	Introduction	4
1.1	Purpose	4
1.2	Scope	4
1.3	Definitions, Acronyms and Abbreviations	4
1.4	Overview	4
1.5	Personnel	5
1.5.1	Development Team	5
1.5.2	Client	5
1.5.3	Supervisor	5
2	Overall Description	6
2.1	User Characteristics and Objectives	6
2.1.1	Researchers	6
2.1.2	General Users	6
2.2	Product Perspective	6
2.2.1	Existing System	6
2.2.2	Proposed System	7
2.3	Product Functions	7
2.4	Requirements Subsets	8
3	Functional Requirements	9
3.1	FCM	9
3.1.1	Core Router Functionality	9
3.1.2	Core Receiver Functionality	10
3.1.3	Non-Core Receiver Functionality	10
3.2	SAM	10
3.2.1	Core SAM Functionality	10
3.2.2	Non-Core SAM Functionality	13
4	Non-functional Requirements	15
4.1	Documentation	15
4.1.1	System Documentation	15
4.1.2	User Documentation	15
4.2	Constraints	15
4.2.1	Performance Constraints	15
4.2.2	Design & Implementation Constraints	15
4.2.3	Hardware Constraints	16
4.3	Quality Requirements	16
4.3.1	Portability Requirements	16
4.3.2	Maintainability Requirements	16
4.3.3	Extendibility Requirements	16
4.3.4	Usability requirements	16
4.3.5	Reliability/Availability Requirements	16
4.4	Error Handling	17
4.5	Licensing	17
5	Interface Requirements	18
5.1	Interface to FCM	18
5.2	SAM User Interface	18
6	Acceptance Criteria	20

7 Deliverables	20
8 Examples of Behaviour	21
8.1 Example: The flow of a TCP packet through the Router to the Receiver	21
8.2 Example: The flow of a TCP packet leaving the Receiver through FCM	21
8.3 Example: Researcher using FCM to create a log file	21
8.4 Example: Researcher using SAM to create a log file	21
8.5 Example: Researcher using SAM	22
8.6 Example: Researcher using SAM	22
8.7 Example: Researcher using SAM	23
9 Client Acceptance	24
10 Glossary	25
11 Appendix	28

List of Figures

1 Shows a typical network with a low-bandwidth link to the outside world	6
2 Shows the path a TCP packet will take through the FCM component of the proposed system	7

1 Introduction

1.1 Purpose

This document formally states the requirements for the project RAMI. Specifically, this document will firstly give an overview of the system to be produced, and then describe the functional and non-functional requirements resulting from the requirements elicitation process. This document will serve as a reference for the design phase, as well as a formal agreement with the Client as to the exact specifications of the product.

1.2 Scope

The project RAMI will produce two systems, FCM (Flow Control Module) and SAM (Statistical Analysis Module).

FCM will contain a set of Linux kernel modules implementing the Client's TCP window sizing algorithm, as detailed in the Client's paper in the Appendix (section 11).

The goal of FCM is to optimise the performance of low bandwidth network links, by changing the window size of TCP connections.

SAM will be a statistics generator and analysis package. The goal of SAM is to enable the Client to evaluate the performance of his algorithm as implemented in FCM.

1.3 Definitions, Acronyms and Abbreviations

This section defines the terms needed to properly interpret this document:

DDD (Detailed Design Document): The document leading on from the SADD that provides a detailed design of all the modules in the project.

FCM (Flow Control Module): The Flow Control Module of this project involving additions to the Linux kernel through the use of kernel modules.

Low Bandwidth Link: A link between two networks with a lower bandwidth across it than between any two hosts in either of the networks it connects; the link is thus a bottleneck, limiting data transfer speed between two networks.

Link: A network communications channel consisting of a transmission path and all related devices between a sender and a receiver.

Receiver: A computer that wishes to receive data from hosts across a low bandwidth network link.

SADD (Software Architecture Design Document): The document specifying the high level design and modular breakdown of this project.

SAM (Statistical Analysis Module): The title of the statistics and analysis package accompanying FCM.

SRS (Software Requirements Specification): This document.

Stream: A series of packets between two hosts, with the same destination and source ports.

RAMI (Rate Adjustment by Managing Inflows): The title of the project comprising of FCM and SAM.

Due to the technical nature of RAMI, the full list of terms (both project-specific and non project-specific) is large, and has been included in the Glossary (section 10).

1.4 Overview

This document consists of ten sections:

- **Section 1: Introduction** gives details of what this document is and how it is organised.

- **Section 2: Overall Description** gives details of the intended users, describes the existing system, and then provides an overview of RAMI.
- **Section 3: Functional Requirements** describes the specific functional requirements, grouped firstly in terms of FCM and SAM, and then further grouped into Core and Non-core requirements.
- **Section 4: Non-functional Requirements** describes the documentation and quality requirements of FCM and SAM, as well as the constraints imposed.
- **Section 5: Interface Requirements** describes the interfaces required for FCM and SAM.
- **Section 6: Acceptance Criteria** lists the criteria that must be fulfilled in order for RAMI to be accepted.
- **Section 7: Deliverables** lists all the deliverables required for RAMI to be accepted.
- **Section 8: Examples of Behaviour** gives typical uses of RAMI.
- **Section 9: Client Acceptance** is a section used for the Client and Team to sign off this document as being the requirements for RAMI.
- **Section 10: Glossary** gives a complete list of definitions of both project-specific and non project-specific terms.
- **Section 11: Appendix** contains the Client's paper on the TCP window sizing algorithm.

1.5 Personnel

This section gives the contact details for all the people involved in the development of RAMI.

1.5.1 Development Team

The developers of RAMI are Team M, consisting of:

Name	Username	Home Ph	Mobile Ph
Maya Muthuswamy	mayadm	9890 3608	0409557919
Daniel Connor	connor	9435 9223	0402143812
Chris Edwards	caedwa	9830 5152	0409411438
Lars Yencken	lljy	9650 7794	0404028880
Rod Howard	rihoward	9853 5853	0409543008

1.5.2 Client

The Client for RAMI is:

Lachlan Andrew
 Dept of Electrical Engineering
 The University of Melbourne

Phone: 8344 3816
email: lha@unimelb.edu.au

1.5.3 Supervisor

The supervisor for RAMI is:

Mark Ng
 Dept of Computer Science & Software Engineering
 University of Melbourne

Phone: 8344 9140
email: markn@cs.mu.oz.au

2 Overall Description

2.1 User Characteristics and Objectives

RAMI has the following two types of users.

2.1.1 Researchers

Researchers refers specifically to those in the field of networking, such as the Client. These researchers are experienced in the use of and development in Linux and other Unix environments. Researchers (specifically, the Client) require RAMI including the SAM extensions to implement their algorithms in a real-world setting and to measure and analyse the behaviour of their algorithms.

2.1.2 General Users

There are two types of General User:

- Experienced Linux users and system administrators, with basic networking knowledge. They are responsible for installing RAMI, and will use only FCM, aiming to increase the performance of low bandwidth links.
- Other users may use the FCM component of RAMI unknowingly after it has been installed by an experienced Linux user.

2.2 Product Perspective

This section aims to describe both the existing and the proposed systems, with respect to why the proposed system is needed.

2.2.1 Existing System

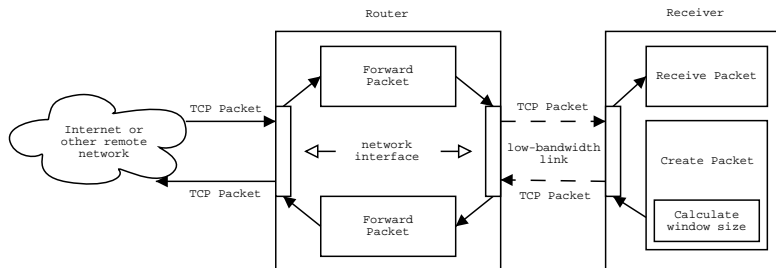


Figure 1: Shows a typical network with a low-bandwidth link to the outside world

The existing system for TCP flow control within the current Linux kernel contains an algorithm for determining the window size of packets sent across a network. The window size is based entirely on information already available to the receiver.

One drawback with the current algorithm is that it is not specifically optimised for low bandwidth links. The window size chosen often becomes too large, causing a packet to be dropped. In response to this, the algorithm then lowers the window size to a sub-optimum level and continues gradually increasing it until it again becomes too large, resulting again in the same response. This cycle results in the low bandwidth link performing beneath its potential. This is particularly a problem because such links are often bottlenecks between networks.

2.2.2 Proposed System

The Client's algorithm for TCP window size takes into account the outgoing queue size on the Router, and in theory causes the window size for the link to stabilise at a level which provides better throughput than the current window size algorithm.

FCM is an implementation of the Client's algorithm which replaces the current algorithm, through the insertion of kernel modules on both the Router and the Receiver computers. SAM has two main functions: firstly, it allows the user to create a network log file containing information relevant to network performance. Secondly, SAM can then use this network log file to perform statistical analysis on it. See Figure 2 for a diagram showing the proposed system.

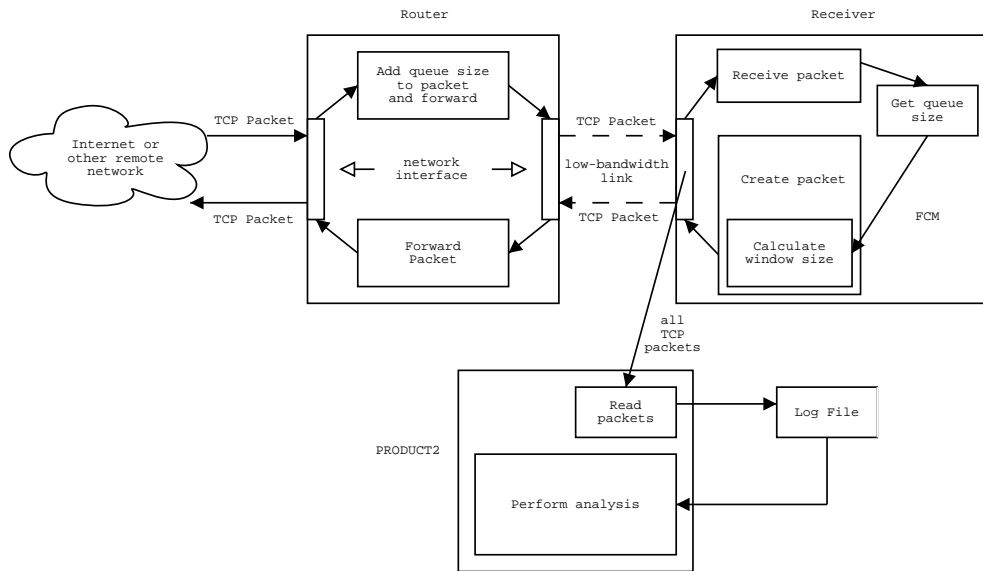


Figure 2: Shows the path a TCP packet will take through the FCM component of the proposed system

2.3 Product Functions

The functionality of FCM is split between the two separate components - the Router and the Receiver. The FCM component provides the following functionality:

- The TCP window size on a Receiver will be determined by the algorithm specified in section 11 based on the queue length information being gathered from a Router.
- A Router will send information about $p(q)$ (see Glossary, section 10) to all hosts that receive data forwarded by the Router.

The SAM component provides the following functionality:

- A Graphical User Interface (GUI) will allow the user to create a network log file with information relevant to network performance based on network traffic over a period of time.
- A GUI will allow users to perform statistical analysis on network logs.
- Statistics will be able to be produced for a single TCP stream, or multiple streams.
- Graphs will be able to be exported to external files, in Postscript format.
- The following analyses will be available for use:

- Plot Data
- Mean
- Variance
- Spectral Analysis
- Autocorrelation
- CDF
- PDF
- Analysis will be able to be performed on the following raw data:
 - Window size
 - Queue size
 - $p(q)$
 - Throughput
 - Number of packets transferred

For further information see the Functional Requirements for SAM (section 3.2).

2.4 Requirements Subsets

There are two subsets or types of requirements in this document:

- **Core requirements**
In order to meet the Acceptance Criteria (see section 6), all core requirements must be implemented, therefore the design of RAMI must include designing for each of these requirements.
- **Non-core requirements**
These requirements are to be implemented if time permits. RAMI should be designed in such a way to facilitate the addition of these requirements.

3 Functional Requirements

This section details the functional requirements for RAMI, split into subsections for FCM and SAM, each in turn split into core and non-core requirements. Core requirements are not prioritised; they must all be completed in order for RAMI to meet the Acceptance Criteria (see section 6). Non-core requirements are prioritised since they may or may not be done.

3.1 FCM

FCM will be broken down further into two components, which will each run on a different host. Hosts running FCM are either Routers or Receivers.

The following functionality will be provided by FCM:

3.1.1 Core Router Functionality

The component of FCM which runs on a Router will have the following functionality:

1. If an incoming packet is being sent on the low bandwidth network interface and is a valid TCP packet, then a representation of $p(q)$ will be transmitted to the Receiver. However, if this will cause the packet to become larger than the interface's maximum data unit, no modification should occur. The value of q is the queue size after the packet is sent, which is the number of bytes waiting to be transmitted across the low bandwidth network link by the Router. The Receiver in this case is the destination of the original TCP packet.
2. All packets which have been altered by FCM and were originally valid TCP/IP packets shall be valid TCP/IP packets as specified by reference [7], including the extensions in reference [5].
3. Packets which have an invalid TCP or IP checksum should not be modified in such a way that their checksum is made correct.
4. The user must specify the output rate μ of the low bandwidth link. This will be specified in units of kilobits/sec.
5. The user may optionally choose to specify values for the following constants used in the window sizing algorithm; any not selected will use the given default values:
 - (a) b - A constant which determines how sensitive the queue size is to the number of streams passing through it. Default value is 10.
 - (b) q_{\min} - The queue size above which $p(q)$ becomes non-zero, specified in bytes. Default value is 20000 bytes.
6. **Create Network Log**

Researchers must be able to collect data from the Router module to be analysed by SAM afterwards. The following functions are required:

 - (a) **Choose file**

The user must specify a filename to dump records to. The format of the file will be known as NETLOG and is the same as that created and loaded by SAM.
 - (b) **Logged Fields**

The following fields will be recorded in every Network Log:

 - i. Timestamp
 - ii. Queue size, in bytes
 - iii. Queue size, in packets
 - iv. $p(q)$
 - v. Number of packets transferred since last dump

- (c) **Logging Interval**
The user must specify the interval at which data is to be dumped into the network log file. After the user-specified amount of time has passed since the last dump, data about the packets (as specified above) will be logged. Allowed time intervals are 100 ms and any multiple of 100 ms, up to 1 hour.
 - (d) **Duration**
The user must specify how long the log will last. The maximum time for any log is 24 hours.
7. The user must be able to install and uninstall the Router part of FCM to and from the kernel.

3.1.2 Core Receiver Functionality

The component of FCM which runs on a Receiver will have the following core functionality:

1. If a TCP packet containing a representation of $p(q)$ (as produced by the Router functionality) has been received, then any TCP packets for new TCP connections originating at the Receiver shall have their TCP Window Size calculated by the algorithm specified in the Client's paper in section 11. Otherwise the standard window size calculated by the Linux kernel will be used.
2. A record will be kept of the current representation of $p(q)$ on the Router.
3. Upon receiving a packet containing a representation of $p(q)$, the local record of this value will be updated to the value received.
4. The user may optionally choose to specify values for the following constants used in the window sizing algorithm; any not selected will use the given default value:
 - (a) τ - A constant which affects the rate at which the window increases. It has units of Bytes/sec. Default value is 1.
 - (b) α - A smoothing factor, normalised to the interval $(0, 1]$, but given to the module on the interval $[1, 100]$ as an integer. Default value is 25.
 - (c) PS_THRESH - A constant used to determine when to end the prime start phase of the window size updating. Default value is -0.2.
5. All packets which have been altered by FCM and were originally valid TCP/IP packets shall be valid TCP/IP packets as specified by reference [7].
6. The user must be able to install and uninstall the Receiver part of FCM to and from the kernel.

3.1.3 Non-Core Receiver Functionality

1. Window size must be correctly calculated for connections making use of the window scaling extension specified by reference [5].

3.2 SAM

This section lists the functional requirements for SAM split into Core and Non-Core.

3.2.1 Core SAM Functionality

This section describes the core functional requirements for SAM.

1. **Load a file**
SAM will allow the user to specify a log file of format NETLOG to be loaded. If the file is not in format NETLOG, as judged by the file header, SAM will give the user an error message and prompt the user to load a different file.

2. Select Data for Analysis

Once a log file has been successfully loaded, the user must be able to select a single form of input data to be analysed from the information available from the log file. This input data will include:

(a) **Queue size**

The size of the outgoing queue of the Router (in bytes) at individual points in time (analysis available either per stream or as an aggregate by sum).

(b) **Window size**

The window size of packets received by the Receiver at the time of each packet's arrival (analysis available either per stream or as an aggregate by sum).

(c) $p(q)$

A linear function of the queue size of the Router (analysis available either per stream or as an aggregate by sum). Explained further in the Client's paper in section 11.

(d) **Throughput**

The number of bytes passing through the Router for a particular stream over a one second period (analysis available either per stream or as an aggregate by sum).

(e) **Number of packets transferred**

The number of packets transferred since the last time data was dumped (analysis available either per stream or as an aggregate by sum).

All these data have time as the independent variable.

3. Select Analysis Type

Once a log file has been successfully loaded, the user must be able to select a single analysis type to be performed. All the analysis types are applicable to all the above data types. The analysis types are:

(a) **Plot data**

Perform no analysis, plot raw data vs. time.

(b) **Mean**

Calculate the mean of the data:

$$\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i$$

where \bar{x} is the mean and the x_i are the n data values (as on page 41 of [6]).

(c) **Variance**

Calculate the variance of the data:

$$variance = \frac{1}{n-1} \sum_{i=1}^n (x_i - \bar{x})^2$$

where \bar{x} is the mean and the x_i are the n data values (as on page 51 of [6]).

(d) **Spectral analysis**

Perform spectral analysis (a Discrete Fourier Transform - see Glossary, section 10) on the data.

(e) **Autocorrelation**

Plot the autocorrelation coefficients for the data against the time lag (where the time lag k has a range of 0 to the last time the data was dumped):

$$r_k = \frac{n}{n-k} \frac{\sum_{i=1}^{n-k} (x_i - \bar{x})(x_{i+k} - \bar{x})}{\sum_{i=1}^n (x_i - \bar{x})^2}$$

where r_k is the lag- k autocorrelation coefficient and the x_i are the n data values (as in [2], with the scaling factor of $\frac{n}{n-k}$ added in).

- (f) **CDF**
Calculate the cumulative distribution function of the data [3]:

$$cdf(x) = Pr[X \leq x]$$

where $cdf(x)$ is the probability that the variable takes a value less than or equal to the data value x , displaying it as a histogram, with the values of the data along the X-axis, and the probability along the Y-axis.

- (g) **PDF**
Calculate the probability distribution function of the data [3]:

$$pdf(x) = Pr[X = x]$$

where $pdf(x)$ is the probability that the variate takes the data value x , displaying it as a histogram, with the values of the data along the X-axis, and the probability along the Y-axis (as on page 314 of [6]).

The output of performing an analysis is either a single graph or a single table (see 'Select Output Type' below).

4. **Select Analysis Options**

Once the analysis type is known by SAM, if necessary, SAM will ask the user for any options or variables that are needed by the chosen analysis. Each variable will already have a default value that can be changed by the user.

5. **Select Stream Combination**

Once a log file has been successfully loaded, SAM will provide the user with a list of possible streams and aggregate streams contained in the log file from which to perform the chosen analysis on. The user must select at least one stream or stream aggregate. All stream combinations are applicable to all analysis types. For applicability of stream combinations to data types, see section 2 above for details.

All of the following stream combinations will be available for selection:

- (a) **Aggregate**
Analysis is performed on the sum of data values of all streams at any point in time and a single overlay is produced.
- (b) **Per stream**
Separate analysis is performed on one or more individual streams and overlaid on the same graph.

6. **Select Output Type**

Once a log file has been successfully loaded, SAM will provide the user with a list of possible output types for the analysis. All of the following output types are applicable to all the stream combinations, analysis types and data types chosen. The user must select one of the following:

- (a) **Graph to screen**
View on screen the single graph produced by the analysis.
- (b) **Graph to file**
Export the single graph produced by the analysis to Adobe Postscript Version 2.0.
- (c) **View Table**
Display the table showing the analysed data on screen.
- (d) **Export to File**
Save the table showing the analysed data to a text file.

When more than one stream or stream aggregate is chosen and the output is a graph, the graph will have multiple overlays, where each overlay corresponds to performing the chosen analysis on a particular stream or stream aggregate.

When more than one stream or stream aggregate is chosen and the output is a table, the table will have multiple columns, where each column corresponds to performing the chosen analysis on a particular stream or stream aggregate. The first column in this table shall hold the X-coordinate values.

7. **Run**

SAM will give the user the option of running the analysis and output only after a data type, an analysis function, a set of streams and/or stream aggregates and an output type have been chosen.

8. **Select Output Options**

Once the output type is known by SAM, if necessary, SAM will ask the user for any further options needed by the chosen output.

9. **Create Network Log**

The user must be able to create a log file of format NETLOG, which can be analysed afterwards. A log file will consist of one or more records, where a record consists of one or more of the following fields. If a record is not being produced for every packet, then one record will be produced for each TCP connection.

- (a) The user must specify at least one of the following fields to be logged:
 - i. Timestamp
 - ii. Average packet size in Bytes
 - iii. Average TCP window in Bytes
 - iv. Destination address and port
 - v. Source address and port
 - vi. Number of packets transferred since last dump
- (b) The user must specify a local network interface to extract the data from.
- (c) The user must specify a filename to dump the data to.
- (d) The user must specify the interval at which data is dumped into the log file, as either:
 - i. **Number of Packets**
After a certain number of packets have been collected since the last dump, data about the packets may be logged.
 - ii. **Elapsed Time**
After a certain amount of time has passed since the last dump, data about the packets may be logged. Allowed time intervals are 100 ms, and any multiple of 100 ms, up to 1 hour.
- (e) The user may specify the how long the log will last. The maximum time for any log is 24 hours.

10. **Exit Program**

The user must be able to exit the program at any time.

3.2.2 Non-Core SAM Functionality

This section lists the non-core requirements for SAM, listed from most desirable to least desirable:

1. **Round-trip time as data to be analysed**

The round-trip time from a source to a destination at individual points in time must be another type of data available for analysis from the log file, with analysis available per stream only.

2. **Output a Gnuplot file**

SAM must be able to output a file containing Gnuplot commands, and any data files necessary to run these commands. The file containing the commands must be accepted as input by Gnuplot.

3. **Output a NETLOG file**
SAM must be able to output new NETLOG and STREAMSLOG files containing the data after processing.
4. **Line Styles**
SAM must provide the facility to globally set all line styles to a particular style which will be used as a default from then on. SAM must still provide the option to change each line style individually.
5. **Estimated error on analyses**
SAM must be able to calculate the estimated error on all the analyses (using the Central Limit Theorem) and output it as error bars (if the output type was a graph) or values (if the output type was a table).
6. **Aggregate by median stream combination**
SAM must allow a stream combination of aggregate by median data value, available for the same data types as aggregate by sum, and available for all analysis and output types.
7. **Analysis of mean by stream**
SAM must allow the mean of each stream to be plotted against the stream (instead of time).
8. **Multiple analysis selection**
SAM must allow more than one type of analysis to be performed simultaneously. SAM must also provide functionality for a separate graph for each analysis type chosen for display.
9. **Multiple data selection**
SAM must allow more than one data type to be analysed and graphed simultaneously, and overlaid on the same graph despite the use of different axes and scaling.
10. **Real-time data logging**
SAM must be able to perform real-time graphing of raw data or data analysis.

4 Non-functional Requirements

4.1 Documentation

This section specifies the documentation that must be produced for RAMI to be accepted:

4.1.1 System Documentation

FCM must include a text file describing the structure of all the kernel modules, and the functionality in the kernel it relies upon. Any module options must also be described here. If any defined constants are used in the source code, these must be described. Since the source code of FCM is likely to be modified, the DDD and TP will be provided as extra system documentation.

4.1.2 User Documentation

The following user documentation must be provided:

- An installation manual for FCM describing all the different installation options and their functions.
- An installation and user manual for SAM which provides instructions on how to install SAM and use all of the different features available.

All user documentation must be:

- Written in English.
- Provided in HTML format.
- Provided to the Client in hard copy.

4.2 Constraints

4.2.1 Performance Constraints

Since the aim of FCM is to ascertain whether the Client's algorithm is feasible, it is more of an experimental system, where performance is less critical. Hence, the only performance constraints on FCM are as follows:

- The Router kernel module of FCM must be able to modify packets (as described in Functional Requirements, section 3.1) at a rate of one megabyte per second for one minute.
- The Receiver kernel module of FCM must be able to receive packets (as described in Functional Requirements, section 3.1) at a rate of one megabyte per second for one minute.
- These two performance constraints must be measured on an Intel Pentium 4 processor, with a speed of two gigahertz, or an older Intel processor of lower speed.

4.2.2 Design & Implementation Constraints

FCM must:

- Contain modules for the standard Linux kernel version 2.4.18
- FCM must require no modifications to existing Linux 2.4.18 kernel code.
- Implement the TCP window size algorithm on the Receiver, as specified in the Client's paper in section 11.
- Be implemented using the C programming language.

SAM must:

- Run under Debian 3.0 (woody)

- Be provided as a Debian package, with any dependencies listed in the package
- Not depend on packages outside of Debian 3.0 unless provided to the user along with the package for SAM

4.2.3 Hardware Constraints

- The user must have a processor capable of running the Linux kernel 2.4.18.
- This PC must also have access to a network.

4.3 Quality Requirements

4.3.1 Portability Requirements

- SAM and FCM must run under Linux kernel version 2.4.18.
- It is a non-core requirement that SAM and FCM run on Linux with no specific kernel version requirement.

4.3.2 Maintainability Requirements

It is likely that Researchers will need to modify FCM to change the algorithm used to calculate window size. Thus all code documentation has to follow the coding standard set out in appendix 2 of the SQAP to make code understandable and maintainable.

4.3.3 Extendibility Requirements

- SAM may be extended to incorporate a traffic generator as part of the analysis package, and therefore should be constructed in a way that facilitates this future extension. This extendibility requirement is a non-core requirement.
- SAM must feature an interface which allows plug-in statistics packages to allow the easy extension of additional analysis functions.
- SAM must feature an interface which allows plug-in output packages thus allowing the easy addition of further output types.

4.3.4 Usability requirements

The interface for FCM will use standard Linux module conventions. That is:

- It will be loadable using the `insmod` and `modprobe` tools.
- Any runtime options will be passed as symbol values to `insmod`.

SAM must:

- Use standard X-window tools as provided in the GTK+ version 1.2 toolkit or in the gnome.ui version 1.4 toolkit.
- Have some form of “tool-tips” to simplify usage.
- Feature in-program help.

See section 5.2 for Interface requirements.

4.3.5 Reliability/Availability Requirements

- FCM must be able to transfer one gigabyte of data between Router and Receiver.

4.4 Error Handling

- **FCM**
FCM will handle and recover from errors according to the standard Linux kernel conventions.
- **SAM**
Error handling in SAM is less critical. Where possible, the user should be notified by a dialog box if an error has occurred, and of the nature of the error.

4.5 Licensing

Both SAM and FCM will be licensed under the GPL (GNU General Public License) version 2 as listed at <http://www.gnu.org/copyleft/gpl.html>. As such, both SAM and FCM may make use of packages and portions of code which are licensed under the GPL or LGPL (GNU Lesser General Public License) or other compatible licenses.

5 Interface Requirements

This section describes the interface requirements for FCM and SAM. However, since there are no requirements for what SAM's user interface must look like on screen, the layout and screenshots of the SAM user interface have been left to design.

5.1 Interface to FCM

The two kernel modules of FCM will be loaded by the user using the `insmod` or `modprobe` tools provided with the Debian linux distribution. If the user wishes to unload one of the modules, the `rmmmod` tool, again provided with Debian, will be used to remove it. Additionally, when logging is required, the user will run the separate logging component, which will be a command line interface.

The following options will be accepted by the Router component:

- **maxrate**
Type: `int`
The rate of the low bandwidth link from the Router, in kilobits/sec. There is no default value, so this must be specified.
- **bval**
Type: `int`
The value of b , used to determine the sensitivity of the queue to the number of streams passing through the Router. Default value is 10.
- **q_min**
Type: `int`
The queue size above which $p(q)$ becomes non-zero, specified in packets. Default value of 30 packets.

The following options will be accepted by the Receiver component:

- **tauval**
Type: `int`
The value of τ in [4], which affects the rate at which the window increases. It has units of Bytes/sec and a default value of 1.
- **alphaval**
Type: `int`
The value corresponding to α in [4], which is a smoothing factor. The value here is divided by 100 to give the value of α . The interval of **alphaval** is [1,100], specified as an integer. The default value is 25.
- **ps_thresh**
Type: `int`
The value corresponding to PS.THRESH, specified as a positive integer, but corresponding to a real negative number. The value here is an integer, which has the following relationship to PS.THRESH: $PS_THRESH = -\frac{ps_thresh}{100}$. The default value is 20.

5.2 SAM User Interface

Due to the graphical nature of SAM, the requirements for the GUI of SAM are closely linked with the functional requirements of SAM, which are detailed in section 3.2. This section is intended to extend those requirements, rather than repeat them.

The GUI for SAM shall:

1. Include four lists (Data, Analysis Type, Output Type and Stream Combination) enabling the user to select which data, analysis, output and stream combination(s) are to be analysed and displayed (see section 3.2 for more details).
2. Include a facility for displaying the graph or tables produced from analysis.
3. Adhere to the GNOME 2.0 Human Interface Guidelines [1] for desktop integration, windows, menus, toolbars, controls, layout and appearance, icons and user input.

6 Acceptance Criteria

The final product will be accepted and deemed a success if:

1. All core Functional and core Non-Functional Requirements as stated in sections 3 and 4 of this document have been implemented.
2. A working version of RAMI has been provided to the Client, where working is defined as functioning as per the requirements detailed in this document.
3. All deliverables (see section 7) have been delivered.

7 Deliverables

This section lists all the required deliverables in order for RAMI to be accepted.

1. A CD-ROM containing a .tar.gz file for each component. These files must contain the full source of the components. Additionally, precompiled binary Debian packages will be provided for the Pentium architecture, and the files necessary to create such packages.
2. All documentation (both System Documentation and User Documentation) for RAMI in the forms specified in section 4.1.
3. The SRS (this document) as a text document.

8 Examples of Behaviour

This section gives examples of the use of RAMI. The examples of use for SAM have been chosen since they illustrate typical uses of SAM.

8.1 Example: The flow of a TCP packet through the Router to the Receiver

1. A valid TCP packet enters the Router, to be forwarded.
2. The Router places it in the queue of the Router part of FCM while it waits to be sent.
3. When the packet is dequeued to be sent, a representation of $p(q)$ is added to the packet by the Router FCM.
4. The packet travels across the low-bandwidth link.
5. The packet (including the representation of $p(q)$) is received by a Receiver.
6. The Receiver part of FCM records the representation of $p(q)$ found in the packet.
7. The packet is processed by existing kernel modules and sent to its destination application/host.

8.2 Example: The flow of a TCP packet leaving the Receiver through FCM

1. An application on the Receiver sends a packet to a host which is to be reached via the low-bandwidth link.
2. Receiver FCM calculates the window size in the following way:
 - (a) If Receiver FCM has a recorded representation of $p(q)$ and the TCP connection was established after receiving this, then the Client's algorithm is used to calculate window size.
 - (b) Otherwise, the default kernel window sizing algorithm is used.
3. The packet is created using the window size calculated, and is passed to an existing kernel module to be sent to its destination as would occur without FCM.

8.3 Example: Researcher using FCM to create a log file

In this example, it is assumed that the Researcher has installed and is running the Router part of FCM. The Researcher wishes to log network data once a second, for one minute.

1. The user types the command to run the logger, giving two command line arguments: one specifying the interval at which data is dumped, and the other specifying how long the log will last.
2. The user also gives an argument which is the filename to dump the data to.
3. The user enters this command, and logging starts.
4. One minute later, the logging completes.

8.4 Example: Researcher using SAM to create a log file

In this example, a Researcher uses SAM to log the window size of outgoing packets over a 15 minute time period to a NETLOG file.

1. The user opens SAM.
2. The main GUI window is displayed to the user, which contains at least a 'File' menu.

3. The user selects 'Create log' from the 'File' menu.
4. SAM displays a list of TCP and IP packet fields to potentially log.
5. The user selects the field 'Window Size'.
6. SAM displays options for the logging interval and for the duration of the logging run, and for the local network interface to log from.
7. The user specifies a duration of 15 minutes and a logging interval of 100ms.
8. SAM displays a directory listing and a field in which a filename can be entered.
9. The user enters a filename, and clicks 'Log'.
10. SAM displays the current status of logging: 'Logging Packets'.
11. 15 minutes later, the status changes to display 'Finished'.
12. The user exits SAM.

8.5 Example: Researcher using SAM

In this example, a Researcher has run SAM and has created a log file. The Researcher now wishes to perform Spectral Analysis on the Window Size, for all streams available. The Researcher also wishes to save the graph produced.

1. The user opens SAM.
2. The main GUI window is displayed to the user, which contains at least a 'File' menu.
3. The user selects 'Open file' from the 'File' menu.
4. A list of files for the user to choose from appears.
5. The user selects the log file (created earlier) from the list.
6. A list of Data for Analysis become available to the user.
7. A list of Stream Combinations becomes available to the user.
8. A list of Analysis Types becomes available to the user.
9. A list of Output Types becomes available to the user.
10. The user selects 'Window Size' as the data to be analysed from the Data for Analysis list.
11. The user selects all the streams from the Stream Combination list.
12. The user selects 'Spectral Analysis' as the analysis to be performed from the Analysis Type list.
13. The user selects 'Graph' as the output type.
14. The user is now able to perform the analysis and output, and clicks on 'Run'.
15. A list of options to be set for the 'Graph' output type is displayed, and the user changes several options, including the option of saving to a file.
16. A single graph is written to the chosen file, with the Spectral Analysis of the Window Size overlaid for each stream.
17. The user exits the program.

8.6 Example: Researcher using SAM

In this example, a Researcher has already created a log file, this time using FCM on the Router machine. The Researcher now wishes to perform CDF on the Queue size, summed over all the streams (aggregate). The Researcher wishes to view the graph produced, but not save it. Several steps performed by SAM which are identical to the previous example have been omitted.

1. The user has performed the first 9 steps of the above example successfully.

2. The user selects 'Queue size' as the data to be analysed from the Data for Analysis list.
3. The user selects 'Aggregate' from the Stream Combination list.
4. The user selects 'CDF' from the Analysis Type list.
5. The user selects 'Graph' from the Output Type list.
6. The user enters the analysis-specific variables (number of bins) and then any output-specific variables (if they wish).
7. A single graph is produced, with a single plot of the CDF of the Queue size for the aggregate (by sum) of all the streams.
8. The user exits the program.

8.7 Example: Researcher using SAM

In this example, a Researcher tries to open a file that is not in the correct format (format NETLOG).

1. The user opens SAM.
2. The user selects 'Open file' from the 'File' menu.
3. A list of files for the user to choose from appears.
4. The user selects a file from the list (of an incorrect format).
5. An error message to the user is displayed notifying them that the file they have chosen is not of format NETLOG, and the user is prompted to choose another file.
6. The user selects another file from the list, this time of the correct format.
7. Since the file is of the correct format, the user may proceed with analysis. (as per examples 8.5 and 8.6).

9 Client Acceptance

I hereby agree that I have read and understood this document and accept it as listing the complete requirements for the product RAMI. I agree that on completion of the Acceptance Criteria, Team M will have satisfactorily delivered the product.

Any changes to the content of the requirements described in this document (that are not spelling, grammar or formatting changes) are to be negotiated between myself and Team M.

Lachlan Andrew
(Client) Date

Maya Muthuswamy
(Project Manager) Date

Rod Howard Date

Lars Yencken Date

Chris Edwards Date

Daniel Connor Date

10 Glossary

bandwidth

The upper limit on the amount of data, typically in Kilobits per second (Kbps), that can pass through a network connection. Greater bandwidth indicates faster data transfer capability.

CDF (Cumulative Distribution Function)

The probability that a variable takes a value less than or equal to a data value x .

datagram protocol

Delivers data in packets the same size as those that were sent, an example being **IP**. For example, if one host sends another host two 50-byte datagrams, that host will receive two discrete 50-byte datagrams. Compare with **stream protocol**.

DDD (Detailed Design Document)

The document leading on from the SADD that provides a detailed design of all the modules in the project.

Debian

A Linux distribution produced by the Debian Project. It is a collection of programs and system utilities, together with a Linux kernel, which make up a working Linux system.

Discrete Fourier Transform

A function used to analyse data that transforms a function of time into a function of frequency, thus enabling frequency (or spectral) analysis. Computational methods entitled 'Fast Fourier Transform' (FFT) are generally used to calculate this.

down state

A state FCM is in whenever it is not loaded and thus not currently running.

FCM (Flow Control Module)

The Flow Control Module of this project involving modifications or additions to the Linux kernel.

General Users

Users as described in section 2.1.2.

GTK+ (The GIMP Tool-Kit)

A multi-platform toolkit for creating graphical user interfaces.

GUI (Graphical User Interface)

A graphical interface provided for the user. Runs on a windows graphical environment such as **X**.

host

A computer or other device connected to a network.

IP (Internet Protocol)

The fundamental Internet protocol, used atop almost any physical network. The network layer (Layer 3) in the **TCP/IP** stack that enables a connectionless internetwork service.

IP address

A binary value used by the **IP** protocol to determine how to deliver **packets** to their destination hosts.

kernel

The part of an operating system which works at the lowest level, controlling access to hardware. It provides a set of system calls and other interfaces which allow programs to access networks, files etc. The kernel referred to in this document is that of Linux, distributed by Linus Torvalds.

kernel module

A module which can be loaded by the kernel after booting. Such a module can implement new functionality or modify existing functionality.

low bandwidth link

A link between two networks with a lower bandwidth across it than between any two hosts in either of the networks it connects; the link is thus a bottleneck, limiting data transfer speed between two networks.

link

A network communications channel consisting of a transmission path and all related devices between a sender and a receiver.

NETLOG

The format of the log file produced, to be specified exactly in the SDD.

occupancy The number of items in the outgoing queue of a network node at any given point in time.

packet

A unit of information transmitted, as a whole, from one device to another on a network.

PDF (Probability Distribution Function)

The probability that a variable takes the data value x .

 $p(q)$

A piecewise linear function of the **queue size** that is used in the Client's window sizing algorithm. Explained further in the Client's paper in section 11.

queue size

See **occupancy**.

Receiver

A computer that wishes to receive data from hosts across a low **bandwidth network link**. The Receiver FCM contains a window sizing algorithm based on the **occupancy** of the closet **Router** providing it data.

Researchers

Users as described in section 2.1.1.

round-trip time

The time required for a network communication to travel from the source to the destination and back. Round-trip time therefore includes time required for the destination to process the message from the source and generate a reply.

Router

A network host providing data to one or more **Receivers** across a low-bandwidth **network link**. It will route **packets** over this link as they arrive from hosts on its local network, and send packets to local hosts as they arrive from the low-bandwidth network link.

The Router **FCM** will modify all packets it routes to include its **occupancy**.

SADD (Software Architecture Design Document)

The document specifying the high level design and modular breakdown of this project.

SAM (Statistical Analysis Module)

The title of the statistics and analysis package accompanying **FCM**.

SRS (Software Requirements Specification)

This document.

stream

A series of packets between two hosts.

stream protocol

Delivers data in variable-length **packets** whose size have no necessary relationship with the size of the packets that were sent, an example being **TCP**. For example, if one host sends two 50-byte packets to the other host, that host may receive them as 100 single-byte packets, as a single 100-byte packet, or as a handful of smaller packets. Further, those 100 bytes might be sandwiched between data from previous and successive sends.

Compare with **datagram protocol**.

TCP (Transmission Control Protocol)

The error correcting Transport layer (Layer 4) in the **TCP/IP** protocol suite. A **stream protocol** that allows efficient data transfer across networks of different kinds with different communication parameters, as well as different underlying communication protocols.

TCP/IP

The protocol suite used in the Internet, intranets, and extranets.

TCP window

The amount of outstanding (unacknowledged by the recipient) data a sender can send on a particular connection before it gets an acknowledgement back from the receiver that it has gotten some of it.

RAMI (Rate Adjustment by Managing Inflows)

The title of the project comprising of **FCM** and **SAM**.

tool-tips

Small pop-up windows in a **GUI** that aid usability by appearing and giving help/instructions when the user leaves the mouse idle over a button for a few seconds.

up state

A state **FCM** is in whenever it is currently loaded and running.

window

See **TCP window**.

X

Also known as X-Windows. A graphical environment for Unix which serves as a platform for other applications.

11 Appendix

References

- [1] Calum Benson, Adam Elman, Gregory Merchan, Seth Nickell, and Colin Robertson. Gnome 2.0 human interface guidelines, 2001.
- [2] C. Chatfield. *The Analysis of Time Series: Thoery and Practice*. Chapman and Hall Ltd, first edition, 1975.
- [3] James J. Filliben. Nist/sematech e-handbook of statistical methods: Related distributions.
- [4] S. Hanly, R. Mukhtar, and L. Andrew. A novel algorithm for receiver-based management of low bandwidth access links.
- [5] V. Jacobson, R. Braden, and D. Borman. RFC 1323: TCP extensions for high performance, May 1992.
- [6] David S. Moore and George C. McCabe. *Introduction to the Practice of Statistics*. W. H. Freeman and Company, third edition, 1999.
- [7] J. Postel. RFC 793: Transmission control protocol, September 1981.