Optimisation and Extension of an Infrastructure Supporting Global Software Engineering Teams

Alan Broady
Thesis for MSc Software Engineering
August 2007




Universiteit van Amsterdam
Hogeschool van Amsterdam
Centrum voor Wiskunde en Informatica
Vrije Universiteit Amsterdam

IBM Global Business Services

Preface

Getting an MSc with a family and a full-time job was always going to be a challenge. After burning the candle at both ends for the last few years, this thesis marks the end point. I've learned a lot. I've got married and had two kids during the whole thing. I've architected, designed or implemented quite a few real-world systems too. Phew!

A few people deserve a lot of thanks for helping me to do this:

Firstly, and definitely most importantly, my wife Caroline for putting up with yet another one of my mad ideas.

Prof. Paul Klint of the Centrum voor Wiskunde en Informatica (Centre for Mathematics and Computer Science) for acting as supervisor for this thesis. Prof. Klint also deserves thanks for accepting my transfer from the UK Open University's Computing for Commerce and Industry Masters programme.

My tutors at the UK Open University: Dr. Anne Lomax, Nigel Kermode, Trevor Nash, Frank Mercer, Dr Veselin Rakocevic, Dr. S.K. Banerjee

The SDLC project team at IBM: Guy Dierx (who also stepped into the breach as thesis supervisor for IBM), Vincent de Beet, Rob Steenbrink.

My employers and colleagues past and present. Thanks to all at IBM (particularly Paul de Wildt), MarketXS and Facing Facts for supporting my studies.

Alan Broady, August 2007

Abstract

This thesis is a network-centric investigation into the IT infrastructure supporting global software engineering teams. A case study is taken as the starting point. A model of such an infrastructure is created using an innovative network simulation package from a university research team. The thesis doubles as an investigation into the efficacy of such tools for general purpose use. Using the model, two problems are addressed. The first problem considers the optimum configuration of tooling. The second problem considers the effects of using of communication and collaboration tools in such a team.

Table of Contents

Part I. Problem Definition

1. Introduction

This thesis is an investigation into the architecture of tools for global software engineering teams based on a case study. It deals with four areas:

- Business processes within global software engineering teams and their relationship to the tools employed.
- Selection of the optimum configuration for tools for global software engineering teams.
- The use of communication and collaboration tools by global software engineering teams.
- The efficacy of network simulation as a means to create a performance model for infrastructure supporting global software engineering teams.

While this thesis was being written, the author acted as infrastructure architect for the case study. In addition, the author worked in a global IT architecture team that made use of communication and collaboration tooling.

Part I of the thesis is concerned with outlining the case study and defining the two problems to be addressed. The following information is presented:

- The context within which this thesis should be placed.
- A description of the case study.
- Definition of the optimisation problem.
- Definition of the extension problem.

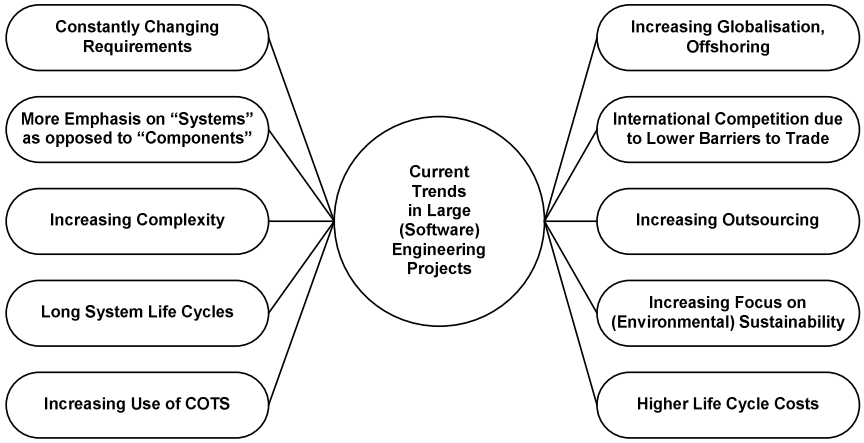
2. Context

2.1. Offshoring and Outsourcing

Over the last ten years software engineering projects have increasingly embraced the practice of offshoring [Carmel 05]. Offshoring is the industry term for relocation of software engineering processes from one country to another, usually with the aim of leveraging (or exploiting, depending upon your political persuasion) lower wages in poor countries in order to reduce costs. In such countries direct labour costs are typically 10% of those in industrialised countries [Carmel 05].

Offshoring in software engineering is a manifestation of the wider phenomenon of globalisation. Some observers have described the trend towards offshoring as a revolution [Meyer 06].

In addition to offshoring, engineering projects are increasingly being carried out by collaborating organisations [Benjamin 04]. This is the trend towards outsourcing.



Current trends in large (software) engineering projects (adapted from [Benjamin 04-I])

2.2. Applicability of the Results

The challenges arising from globalisation are not unique to software engineering teams. Other global teams also need access to IT infrastructure and tools for communication and collaboration. This thesis brings various issues to the fore that will affect such teams.

The different architectures considered for the version control system in section 4 are typical of the options available to any application deployed over a WAN. As such, the comparison of the various architectural patterns has wider general applicability.

The network simulation method has broad applicability. Many applications feature client-server pairs. Responsiveness is an important quality to establish, and one that frequently proves problematic. The thesis shows one way to create a model that can predict responsiveness.

3. Case Study

3.1. Overview

This thesis is partially based on a case study of a project carried out by IBM. The project in question delivered a global software engineering infrastructure to a large corporation. Such projects are referred to within IBM as Software Development Life Cycle (SDLC) projects. The thesis was written between project inception and deployment of the first iteration into production. As such it was not possible to draw conclusions about how the actual system fared in practice.

The corporation in question is migrating application development and maintenance activities to an outsourced and largely offshored model. A new methodology is being adopted – the IBM Rational Unified Process [Gibbs 07]. The SDLC project implemented the methodology through deployment of the IBM Rational suite of software engineering tools [Rational].

This combined transition to a new methodology, a new commercial model and to working in global teams is radical. Different workflows (to use the RUP terminology[1]) will be carried out by specialised teams working at different locations. The software engineering tooling will be used to coordinate activities between these teams.

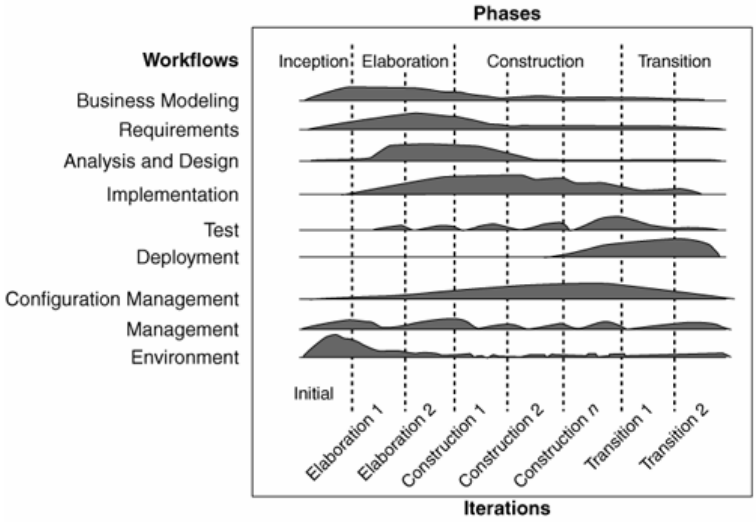
The main drivers behind the corporation's move to this new model were as follows:

- Reduce costs through offshoring to developing countries.
- Increase flexibility and leverage expertise through outsourcing.
- Improve quality through standardisation of processes.
- Improve traceability[2] through the use of suitable tooling.

Many changes to business processes were required in order to realise this new strategy. This thesis is only concerned with changes to the software engineering methodology, tooling and network infrastructure.

3.2. Software Engineering Methodology

Software engineering processes were modelled according to the IBM RUP methodology. This methodology views a project as being divided into phases and iterations. Activities are grouped into workflows.



RUP workflows (sometimes referred to as disciplines), phases and iterations (from [Gibbs 07])

[1] In earlier RUP documentation the term "disciplines" was used. In recent IBM documentation the term "workflows" has been used.
[2] This is in turn related to increasing pressure on large organisations to be able to justify their information systems in the light of legislation such as the US Sarbanes-Oxley Act [Vance 07].

The RUP methodology was adapted for the corporation in question using a tool using known as Rational Method Composer. The adapted methodology specified numerous roles, work products and processes. It is beyond the scope of this thesis to describe the methodology in depth.

The intention is to assign workflows to different teams. The teams are to be provided by "vendors" of software engineering "services". Of the four phases, it is most likely that the construction phase will be outsourced and offshored. Of the nine workflows, it is most likely that the implementation and test workflows will be outsourced and offshored. Some workflows, particularly the requirements workflow, will continue to be carried out by employees of the corporation.

The teams are to use the new infrastructure to perform most of their daily activities. They are also to use the infrastructure to exchange work products as and when necessary.

3.3. Software Engineering Tooling

The infrastructure is based on the IBM Rational toolset, which provides end-to-end traceability. The toolset consists of three products for team collaboration specialised for software engineering:

- ClearCase, a version control system.
- ClearQuest, an issue tracking system.
- RequisitePro, a requirements management system.

Software engineers use an IBM Rational IDE (based on Eclipse) in order to perform activities such as component design, implementation and test. The IDE is integrated with the other three products.

3.4. Communication and Collaboration Tooling

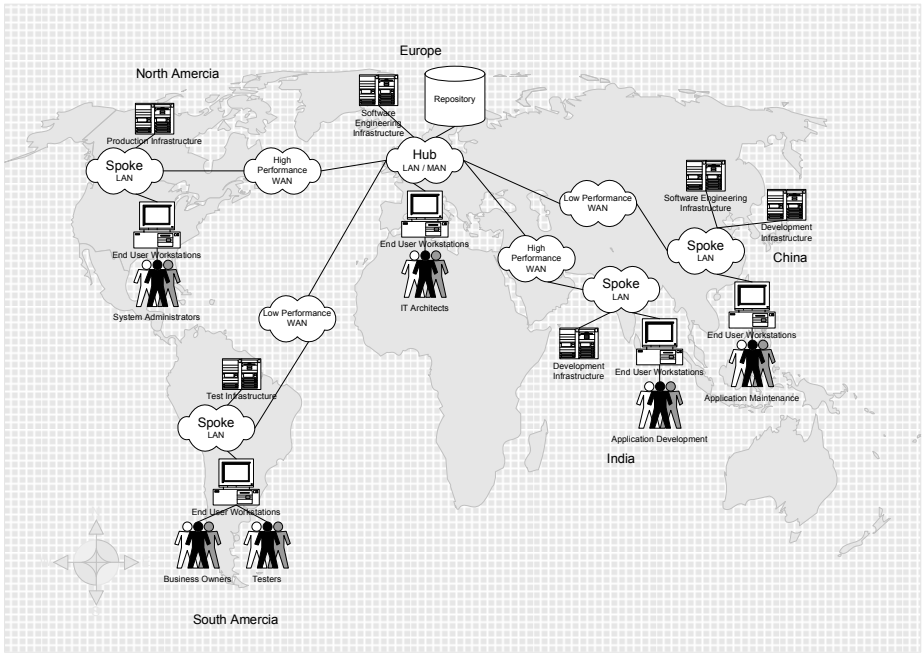
The SDLC project did not involve deployment of any new, general purpose communication and collaboration tooling. It was assumed that the teams will use e-mail and conventional telephony for this purpose.

3.5. Network Infrastructure

Workflows are to be assigned to teams clustered at geographic locations referred to as spokes. For example, testing might be carried out in South America, analysis in Europe and implementation in India. The exact location will vary according to the project at hand. Network connectivity to the various locations differs in cost, availability and lead time. As such, not all spokes are to be connected to the hub by high performance, highly available networks.

Software engineering artefacts, such as source code, test reports and requirements documents are to be stored in a central repository. This must be located at the corporation's main data centre in Europe, referred to as the hub. This centralisation is thought to simplify management of intellectual property in the new environment. However, it makes the system sensitive to the characteristics of network connectivity to the various locations.

The finished software products must be deployed on production infrastructure. This production infrastructure could be stationed at any of the data centres operated worldwide by the corporation.



Typical distribution of user groups and infrastructure

3.6. Key Requirements

The SDLC project had many requirements. It is beyond the scope of this thesis to describe them in depth. A number of key requirements directly affect the problems addressed in this thesis:

- There must be a single repository for all artefacts of the engineering process.
- Direct end-user remote connectivity must feature encryption and two factor authentication.
- If possible, source code and other artefacts should remain within the corporate network.
- If possible, thin client architecture should be used.

3.7. Relationship between Business Processes and Technical Components

Each workflow from RUP will be performed using certain technical components. There are additional components for implicit (but still primary[3]) processes such as communication and collaboration.

Business Process	Generic Technical Component	Case Study Technical Component
Business Modelling	UML Modelling Tool	IBM Rational Software Architect IDE
Requirements	Requirements Management Tool	IBM Rational RequisitePro
Analysis and Design	UML Modelling Tool	IBM Rational Software Architect IDE
Implementation	Integrated Development Environment Issue Tracking Tools Build Tools	IBM Rational Software Modeler (and variants) IDE IBM Rational ClearQuest Build tools specific to the project in question.
Test	Testing tools Issue Tracking Tools	TestDirector IBM Rational ClearQuest
Deployment	Build Tools Configuration Management Systems IT Asset Management Systems	Build tools specific to the project in question. Configuration management of production systems is based on PVCS. Peregrine AssetCenter.
Configuration Management	Version Control System	IBM Rational ClearCase
Management	Project Management Tools	Microsoft Project
Environment	System Management Tools Operating Systems Middleware etc.	Specific to the project in question
Communication	Non-real-time Communication Tools. Real-time Communication Tools.	E-mail. Conventional telephony.
Collaboration	Collaboration Tools.	ClearCase, ClearQuest and RequisitePro could be classified as specialised collaboration tools. Tools specific to the project in question.

[3] Many business process modelling methodologies divide processes into primary (creating the product), secondary (supporting primary processes, e.g. HR) and tertiary (strategic). For an example see [Cleland 06].

4. Optimisation Problem: ClearCase / IDE Configuration

4.1. Background

Users will be based at different locations worldwide, known as spokes. The location of the spoke will determine the available network bandwidth, latency and availability to the hub where the central repository is located. The IBM Rational suite can be deployed according to a number of different architectural patterns [Wahli 04]. Certain architectural patterns may be appropriate for different types of connections. The optimisation problem is selection of the appropriate architectural pattern in each case. In addition, limits on team size are to be computed.

The selection of the appropriate architectural pattern will be based primarily upon the quality of the resulting user experience. However, other issues such as infrastructure cost, availability, scalability and security should also be taken into account.

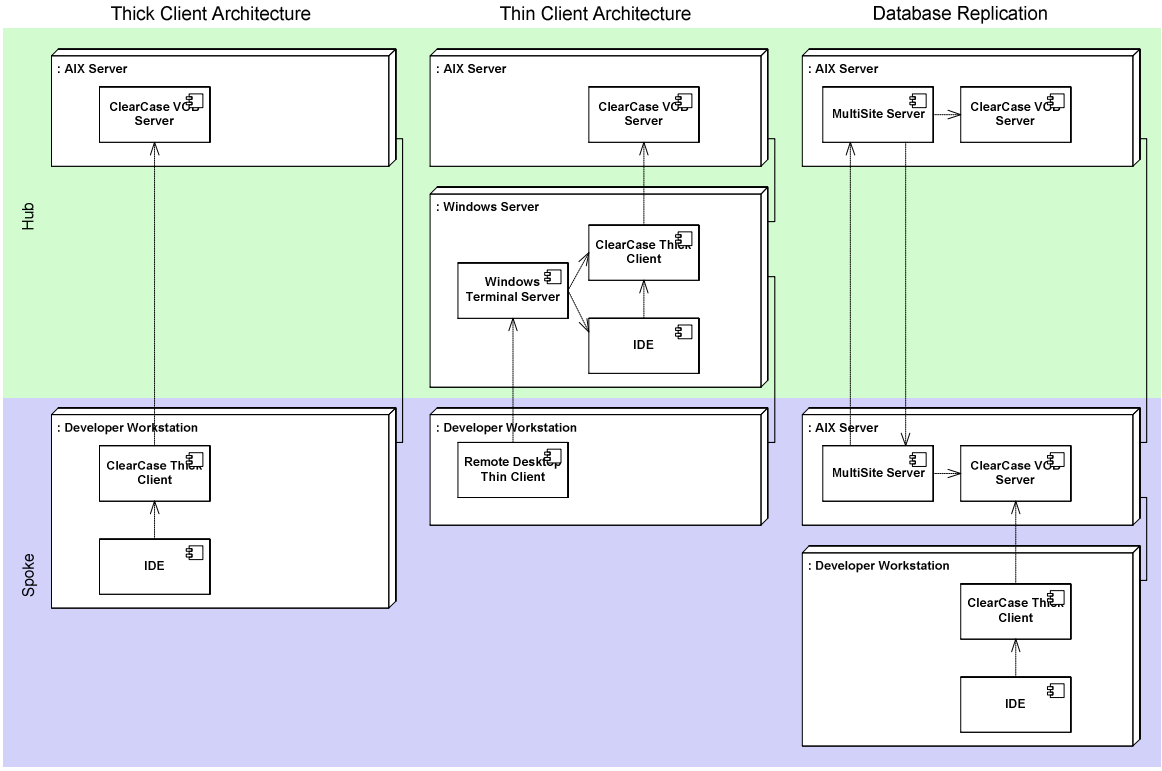
Analysis of the architectural patterns for ClearCase, ClearQuest and RequisitePro would lead to unnecessary duplication. The ClearCase version control system is the most important of the three and it has the widest range of possible configurations. The importance of shared version control systems in offshored projects has been noted [Treinen 06] [Carmel 99] [Kircher 01]. For this reason, only the architectural patterns for ClearCase and the IDE were investigated.

4.2. Acceptable Architectural Patterns

There are a number of architectural patterns that could be used to provide users with the ClearCase and IDE tools. The architectural patterns that were acceptable in the case study were as follows:

Architectural Pattern	Description
Thick Client	The end user has the thick client applications installed on her workstation. The workstation makes a direct connection to the ClearCase VOB server. The ClearCase thick client is unusual in that it introduces a network file system known as MVFS onto the client machine. Operations such as loading and saving files involve network traffic even when no check in or check out operation is performed. This is unusual for a version control system.
Thin Client	The end user accesses the thick client user interface via a virtual desktop provided using Windows Terminal Services.
Database Replication	The data is replicated to a location close to the end user. At this location, a ClearCase VOB server is deployed that provides access to the replicated data. The user can connect to this VOB server using the thick client application.

Description of the three acceptable architectural patterns



UML deployment diagram showing the three acceptable architectural patterns

The different architectural patterns have non-functional characteristics as shown in the table below:

Architecture	Infrastructure Cost	Availability	Scalability	Security
Thick client.	Low server management costs. High client management costs.	Possible for end-user to work without connectivity for a limited period of time. Star topology. Maintenance windows must be short. Component reliability must be high.	ClearCase server deployed on a platform that can scale up and out. Horizontal portioning possible.	Difficult to provide two-factor authentication when used from outside the corporation's network. Intellectual property is transferred to the end-user's workstation. Most likely architecture to result in compromise of central repository.
Thin client.	Heterogeneous server architecture with high management costs. Low client management costs. The IDE places very high load on Windows Terminal Server. As such, the client-server ratio will be poor.	Impossible for end-user to work without connectivity. Star topology. Maintenance windows must be short. Component reliability must be high.	ClearCase server deployed on a platform that can scale up and out. Windows Terminal Server can be scaled up (limited) and out. Horizontal portioning possible.	Thin client provides two-factor authentication (this is part of the corporation's existing infrastructure) Keeps all intellectual property within the corporation's data centre.

Database replication.	Low server management costs. There are more servers, but the need for availability is lower. High client management costs.	Possible for end-user to work without connectivity to local ClearCase server for a limited period of time. Possible for end-user to work without connectivity to central repository for a long period of time. Tree topology (mesh also possible). Maintenance windows may be longer. Component reliability may be lower.	ClearCase server deployed on a platform that can scale up and out. Flexible due to option to provide capacity at the hub or at the spoke. Vertical and horizontal portioning possible.	Intellectual property is transferred to the end-user's workstation. Connectivity to the central repository is limited and can be in line with corporate policy.

Non-functional characteristics of the acceptable architectural patterns

4.3. Rejected Architectural Patterns

In addition to the patterns listed above, various others were possible [IBM 07]. These were rejected or not included in the case study for reasons listed below:

Architectural Pattern	Description	Reason for Rejection
CCWeb	A web-based presentation layer	Limited functionality. Limited integration between ClearCase and IDE on the desktop.
CCRC	Clear Case Remote Client. This is an alternative thick client for Rational ClearCase that is optimised for use over WANs.	Limited functionality Limited integration between ClearCase and IDE on the desktop.
ClearCase static views	An alternative configuration whereby updates must be "pulled" by the end user. This is at odds with the philosophy of the ClearCase product, but it reduces network traffic.	Limited functionality
Application Streaming	Similar to thick client, but application components are deployed to the client on demand (e.g. Microsoft SoftGrid [Microsoft 07]).	Technology was new at time of project initiation, so was not considered as an option.
OS Virtualisation	Variant on the concept of remote desktop. Complete operating system is offered to remote users based on virtualisation (e.g. using products from VMware). Consumes more resources on the server, but offers a higher degree of isolation between the users and increases application compatibility.	Considered to be too expensive in terms of client-server ratio. Did not fit in with corporate standards and policies.

Table of architectural patterns that were rejected or not considered during the case study.

5. Extension Problem: Use of Communication and Collaboration Tooling

5.1. Background

Global software engineering teams cannot take part in daily face-to-face communication. This is detrimental because this is believed to be the most effective form of human communication [Cleland 06] [Carmel 05] [Carmel 99]. Global software engineering teams must turn to other forms of communication and may require more extensive collaboration tooling. It has been suggested that global software engineering teams could benefit from communication and collaboration tools integrated into their development environments [Hupfer 04] [Kircher 01].

A number of best practices are associated with RUP such as continuous verification of quality and cross-team collaboration [Gibbs 07] [Kroll 05]. These practices imply that there will be significant communication and collaboration between the teams.

Global teams use conventional telephony (often with audioconferencing) and e-mail extensively as a kind of "lowest common denominator". The case study did, implicitly, include conventional telephony and e-mail. Various studies of global teams suggest that the use of more advanced communication and collaboration tooling can increase their effectiveness [Lurey 01] [Cleland 06] [Carmel 05].

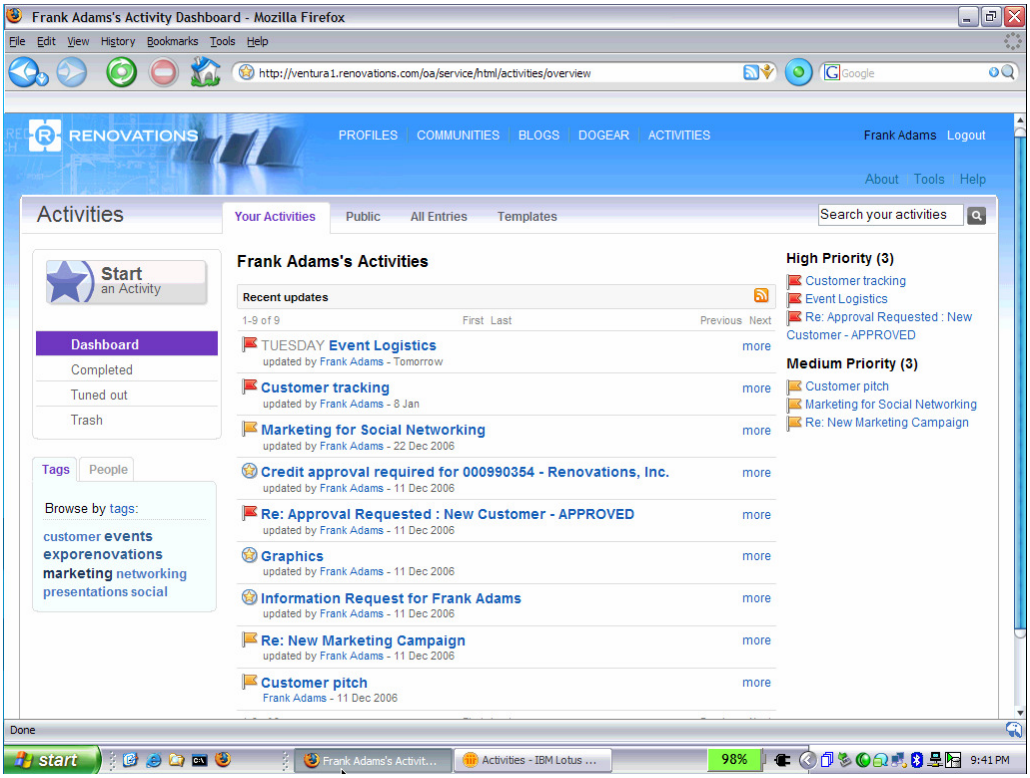
The extension problem is to propose an architecture for communication and collaboration tools that could be added to the system identified in the case study. The selection of appropriate tools should be based on network performance and likely effectiveness of the tools in enhancing team performance. In addition, limits on team size are to be computed.

The extension problem is limited to generic classes of tools, whilst the optimisation problem deals with the exact, actual toolset used in the case study.

5.2. Recent Developments in Communication and Collaboration Tooling

The last five years has seen a surge of interest in real-time communication technologies based on IP networks such as voice over IP (VoIP), videoconferencing over IP and instant messaging [Cleland 06]. There have even been some tentative movements towards the use of collaborative virtual environments such as Second Life [Bulkeley 07].

The last five years have also seen a surge of interest in collaboration technologies that have come to be classified as "social networking". They are typified by public services such as LinkedIn, del.ico.us and Wikipedia. A number of vendors offer packages providing similar facilities for use on corporate networks. Such packages include functionality such as wikis, social bookmarking [Millen 05], podcasting [Patterson 06], "communities", blogs and "team spaces".



Web-based social networking tools

5.3. The Effect of Time Zones

In a global team, time windows for real-time communication are constrained due to the effects of time zones. This leads to higher peaks in communication tool usage and corresponding higher peaks in network resource consumption.

GMT	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
San Francisco																								
New York																								
London																								
Mexico City																								
Amsterdam																								
Mumbai																								
Beijing																								
Johannesburg																								
Sofia																								
Moscow																								
Sydney																								

Approximate relationship between office hours in major centres of population. Low labour cost regions are shown in green, high labour cost regions are shown in blue.

Despite the fact that daily handovers between time zones are unlikely (see section 6), it is still desirable to be able to handover work in a short period of time. For the purposes of this thesis, a value of one hour for a handover has been taken as a requirement.

5.4. Psychological Value of Different Media

When considering which communication and collaboration tools to deploy, psychological factors should be taken into account. Members of a global team cannot regularly communicate face-to-face. They also have reduced opportunities for informal interaction. The aim of the tools is to provide a surrogate.

In a 1994 Harvard Business School study various communication and collaboration technologies were ranked by the degree of psychological involvement [Carmel 99] (see also [Daft 86]). This list is reproduced below, modified to include more recent technologies. The placement of the new technologies is based on the ratio of the participants and the similarity of the media. The original study included conventional media such as postal mail and face-to-face meetings. These are included in the list for reference purposes.

Medium	Example	Participant Ratio	Real-time	Predominant Architecture
Social Bookmarking	Lotus Connections	m:m	No	Star
RSS / ATOM	Blogger RSS Feeds	1:m	No	Star
Blogging	Blogger	1:m	No	Star
Postal mail				
Courier mail				
e-mail	Lotus Notes	1:m	Near-real-time	Star
Wiki	Wikipedia	m:m	No	Star
Fax				
Voicemail[4]				
IM 1:1	Lotus Sametime	1:1	Yes	Star / Peer-to-peer
Broadcast audio	MPEG4IP	1:m	Yes	Star
Broadcast video	MPEG4IP	1:m	Yes	Star
Audio on demand	Podcasting	1:m	No	Star
Audio annotation to presentation / e-mail	Microsoft PowerPoint	1:m	No	Star
Video on demand	Video Podcasting	1:m	No	Star
Broadcast audio with audio back channel	Lotus Sametime	m:m	Yes	Star
Conventional telephony				
Computer based telephony (VoIP)	Lotus Sametime	1:1	Yes	Peer-to-peer
Conventional telephony conferencing				
Computer based audioconferencing	Lotus Sametime	m:m	Yes	Peer-to-peer
Remote desktop viewer	Lotus Sametime	1:1	Yes	Peer-to-peer
Liveboard [Elrod 92] with telephony	Lotus Sametime	m:m	Yes	Peer-to-peer
Broadcast video with audio back channel	Lotus Sametime	1:m	Yes	Peer-to-peer
Videoconferencing	Lotus Sametime	1:1	Yes	Peer-to-peer
Collaborative virtual environment [5]	Second Life	m:m	Yes	Star
Face-to-face meeting				

Communication and collaboration tools, in approximate increasing order of psychological involvement

[4] Many VoIP systems allow voicemail, but this is not considered further here

[5] When the original list was complied in 1994 it included virtual reality. The concept of VR as an immersive technology using goggles and data gloves was then in the ascendancy. This thesis bows to the current fashion for videogame-like systems such as Second Life. Note that these systems can be combined with an audio channel and that Second Life itself features a beta version including audio at the time of writing.

6. A Note on "Follow-the-Sun" Software Engineering

The original proposal for this thesis included an investigation of "follow-the-sun" engineering whereby teams in different time zones hand over to each other on a daily basis. This could lead to interesting challenges for the infrastructure, mainly related to replication of version control and issue tracking systems between the teams when performing a daily handover.

It emerged that few software engineering teams have managed to put the concept of "follow-the-sun" into practice. In the book "Global Software Teams", Carmel presents three case studies of follow-the-sun [Carmel 99]. The larger of the case studies, IBM's Global Software Factory, failed to perform daily handovers despite attempts to operate in this way. The most successful of the follow-the-sun case studies was development of a screen saver; hardly a project with the complexity found in a major enterprise application. In Carmel's more recent book "Offshoring Information Technology" he states that "daily follow-the-sun coordination is simply too difficult for software teams" [Carmel 05].

Schiff provides a number of examples of "follow-the-sun" engineering [Schiff 06]. These examples are all of systems with a significant hardware component. As such, the conclusions are questionable in the context of enterprise software projects.

Treinen and Miller-Frost present two interesting case studies [Treinen 06]. The first case study describes development using teams in Australia and the USA. This case study was successful, but does not seem to have featured daily handover of work products. The second case involving offshoring to India was less successful. One of the major reasons given for failure was the lack of central software engineering infrastructure of the type described in this thesis.

Given the lack of support from the literature for follow-the-sun with daily handovers of work products, there seems little basis to perform a full analysis of this scenario. It appears that real world software engineering projects may have certain "follow-the-sun" characteristics, but that they do not generally hand over on a daily basis.

Part II. A Method for Modelling Networked Applications using Emulation, Analysis and Simulation

7. Introduction

This thesis aims to address the optimisation problem and, to a lesser extent, the extension problem identified in Part I by constructing a simulation model. Part II of the thesis is concerned with defining the simulation method and parameters. The following areas are addressed:

- Parameters arising from application demands
- Parameters arising from real-world network characteristics
- Tool selection
- Experimental method

In practice, development of an experimental method and tool selection was an iterative, combined process. Only by attempting to perform experiments did it become apparent that certain tools would be required, or that others would be insufficient. They are presented separately for the sake of clarity.

8. Application Demands on Networks

A networked application will make demands on the network upon which it is deployed. The demands will be largely dependent upon the application type and its architecture. The demands arise from the need of the application to deliver certain performance. The acceptability of a given level of performance is in turn related to the user's needs and expectations.

8.1. Demands of Applications with Graphical User Interfaces

8.1.1. Classification of Response Time

There are two possible definitions of application response time [Fowler 03]:

- The time taken for immediate user interface feedback. For example the time taken for a character to be displayed following a key being pressed. This is referred to in this thesis as "acknowledgement response time".

- The processing time for some business logic to take place. For example the time taken for the results of a search query to be performed. This is referred to in this thesis as "processing response time".

8.1.2. Psychological Factors

The acceptability of response time is determined by a number of factors:

- The user's mental model of the system [Bouch 00]. For example, the user may expect a complex search query to take a long time to complete.

- The type of user interface [Tolia 06]. Users have a different expectation when using a remote desktop technology such as Citrix or Windows Terminal Server compared to when they are using a web client.

- The frequency of use of the function [Bouch 00]. For a more intensively used function, the user expects a faster response.

- The degree of experience of the user [Bouch 00]. More experienced users expect faster response times.

- The variation in response time [Tolia 06] [Bouch 00]. If a system is sometimes slow, users will report that it is always slow.

Fast response times clearly improve ergonomics and productivity. They also affect user confidence [Bouch 00] and degree of anxiety [Tolia 06].

Slow response times can cause users to believe that an error has occurred. This can be counteracted by providing some visual feedback that processing is taking place [Myers 85]. It has been reported that users will accept a slow response time if they appreciate that the system is heavily loaded [Bouch 00].

8.1.3. Acceptable Response Times

Based on the literature, the following values for acceptable response times are proposed. A distinction is made between the two classes of response time mentioned above.

Acknowledgement Response Time	
Response Time, t	**User Perception**
t <=15ms	Instantaneous [Cheshire 96]
15ms < t <= 150ms	Crisp [Tolia 06]
150ms < t <= 1s	Acceptable [Tolia 06]
1s < t <= 5s	Unacceptable [Tolia 06]
5s < t <= 10s	Unusable [Tolia 06]
t > 10s	User begins to believe that an error has occurred. [Bouch 00]

User perception of acknowledgement response times

Processing Response Time	
Response Time, t	**User Perception**
t <= 5s	High quality of service
5s < t <= 11s	Average quality of service
t > 11s	Low quality of service

User perception of processing response times (adapted from [Bouch 00])

8.2. Demands of Communication and Collaboration Technologies

8.2.1. Classification

There are essentially two types of systems for communication and collaboration:

- Real-time systems involving live communication between the participants. This class of systems includes voice, video and (in some modes of use) collaborative virtual environments.

- Non real-time systems involving decoupled communication between the participants. This class of systems includes e-mail and collaborative document authoring.

8.2.2. Psychological Factors

For real-time systems facilitating human communication the generally accepted maximum latency is 100ms [Cheshire 96] [Baldi 99]. Latencies above this value tend to cause breakdown of normal conversation. Jitter can have as strong as an effect as latency [Chen 02]. Conversation is still possible with higher latencies or jitter, but it ceases to be fluid.

In order to reduce jitter, such traffic is often prioritised on the network (e.g. using DiffServ [Blake 98] or MPLS [Rosen 01]). This is difficult to achieve over the public Internet, but quite feasible over private networks such as those operated by large corporations.

8.2.3. Acceptable Latency

Based on the literature, the following table is proposed for real-time systems. No separate tables are proposed for non-real-time systems. Instead, the tables from section 8.1.3 should be used.

Real-time Communication and Collaboration Latency	
Latency, t	**User Perception**
t <=15ms	Instantaneous [Cheshire 96]
15ms < t <= 100ms	High quality [Cheng 06] [Baldi 99]
100ms < t <= 270ms	Low quality [Cheng 06]
t > 270ms	Very low quality, unacceptable [Cheng 06]

User perception of network latency when performing real-time communication and collaboration.

8.3. Bandwidth Demands of Real-time Communication Tools

8.3.1. Audioconferencing and Videoconferencing

Audioconferencing and videoconferencing over IP networks are well established technologies. They are offered using similar underlying platforms. Such platforms involve two basic components:

- Presence / session initiation

- Streaming audio / video

The presence / session initiation component is often part of a more general-purpose communication system supporting various media. Presence and session initiation are usually based on TCP. In many systems, SIP [Rosenberg 02] is used. This component typically has a star architecture. The bandwidth consumed by the presence / session initiation component is almost negligible compared to that for the streaming audio / video.

The streaming audio / video component is based on underlying codecs. The stream is encoded and sent in the most direct way possible to the other participant(s). Streaming audio / video is typically based on UDP. This component typically has a peer-to-peer architecture. Various sources state that the actual network bandwidth is higher than the raw output of the codec. The typical gain is of the order of 30%, but varies according to the exact

protocol used.

Audio		
System	Codec Bandwidth	Source
Skype	20 – 64 kbps	[Chen 06]
Various ITU standard codecs	10 – 40 kbps	[Skinnemoen 05]
Various ITU standard codecs	5.3 – 64 kbps	[Sharafeddine 03]

Example bandwidth requirements for audioconferencing systems

Video		
System	Codec Bandwidth	Source
Coliseum	616 kbps	[Baker 05]
Low bandwidth video	100 kbps	[Chen 02]
Typical H.323 system	384 kbps	[Calyam 04]

Example bandwidth requirements for videoconferencing systems

8.3.2. Collaborative Virtual Environments

In contrast to both audioconferencing and videoconferencing, collaborative virtual environments typically have a star architecture. This places high load on the central system.

The literature study revealed no data on typical bandwidth requirements for collaborative virtual environments. As part of this study a simple experiment was performed to obtain some data in this area (see section 19).

9. Network Characteristics

Having established the demands that applications place on networks, it is now possible to contrast these with the actual characteristics of networks. There are three main network characteristics affecting the infrastructure required by global software engineering teams: latency, bandwidth and availability. In addition, during project planning lead times must be considered.

9.1. Latency

The most important effect concerning the use of remote infrastructure for software engineering is network latency. It is the only characteristic that is impossible to significantly change, irrespective of the budget or technology available. It is fundamentally limited by the speed of light [Cheshire 96].

Geographical distance	Typical IP network latency (RTT)	Source
LAN Within a building	< 1ms	IBM project data
MAN Within a campus or a metropolitan area	5ms	Interpolation
WAN Intra-continental	50ms	[Pappalarado 02] [Cheshire 96]
WAN Inter-continental	200ms	[Pappalarado 02]
WAN Maximum (e.g. Europe to Australia)	400ms	Extrapolation, [Cheshire 96]

Table of typical real-world network latencies.

It should be noted that actual latency over corporate networks may be significantly higher than the values given above. This is due to the effects of components such as firewalls, virus scanners and overlay networks [Tolia 06].

9.2. Bandwidth

In addition to latency, a network connection is also characterised by its bandwidth. This is generally a function of the physical medium and the budget available.

LANs can be seen in two main settings. Within offices they are used to provide connectivity to end users. Within data centres they are used to provide connectivity to servers. It is typical to deploy an architecture wherein switches are connected to each other at 1Gbps (Gigabit Ethernet), with connectivity to individual desks and servers at 100Mbps. Some older installations are still using 10Mbps Ethernet. Some locations are using wireless networks, which are currently limited to 54Mbps (typical actual throughput closer to 20Mbps).

For MANs a variety of technologies are available. It is typical to think of MANs as having higher bandwidth than most WANs. Their relatively small size means that they are not limited by various factors affecting WANs. For example repeaters may not be necessary on optical networks. MANs typically offer bandwidth in the 100Mbps range.

For WAN links between major centres of population, bandwidth is largely determined by financial constraints, at least up to speeds in the 100Mbps range. Even in a developing country such as India, it is relatively straightforward to arrange 155Mbps STM-1 connections in cities such as Mumbai or Bangalore [Trivedi 06].

For WAN links to remote areas, the picture is very different. It may be impossible to arrange high bandwidth connectivity. Even low bandwidth connectivity may be expensive and unreliable. WAN bandwidth is limited not by the distance involved, but by the proximity of the desired endpoint to infrastructure operated by a major telecom provider.

Network Class	Example Standard	Typical bandwidth
Wireless LAN	IEEE 802.11g ("Wi-fi")	20Mbps
Wired LAN inter-switch	IEEE 802.3ab / 1000BASE-T	1Gbps
Wired LAN to the desktop or server	IEEE 802.3u / 100BASE-T	100Mbps
MAN	IEEE 802.6	150Mbps
WAN connectivity to service provider using ADSL.	ITU G.992.5	20Mbps
WAN connectivity to service provider using a digital hierarchy.	E1	2Mbps

Table of typical real-world network bandwidth

9.3. Availability

Availability is dictated by three factors.

- The reliability of individual components, such as routers and leased lines.
- The degree of redundancy.
- The impact of disasters (for example [BBC 06]).

The effects of these factors are well documented. They will not be dealt with further here as they are not the focus of this thesis. Note that typical connection reliability may be lower in developing nations [Trivedi 06].

9.4. Lead Time

Connectivity between cross-continental teams may be possible, but turning that possibility into reality can be a complex IT project in its own right.

Lead times from communications providers can seriously affect the critical path of a project. Setting up VPNs and MPLS routers can involve complex troubleshooting. In some cases, a project can be planned based on the assumption that connectivity will be present. If it is not forthcoming, this will have a negative impact on the success of the project [Treinen 06].

9.5. Anecdotal Evidence

As part of the research into this thesis, some anecdotal evidence of network performance of infrastructures used by real-world global software engineering teams was collected. This was as follows:

Countries Linked	Latency (RTT)	Bandwidth
The Netherlands / China	615ms	2Mbps
The Netherlands / India	200ms	Unknown
USA / India	Unknown	20Mbps

Examples of real-world networks used by global software engineering teams

9.6. Network Classes to be Modelled

This thesis will move on to examine the performance of various architectures on typical networks. The following classes of network will be considered, based on the data collected in this section:

Network Class	Latency (RTT)	Bandwidth
LAN	1ms	1Gbps (Gigabit Ethernet)
MAN	5ms	155Mbps (OC-3)
High performance WAN	100ms	20Mbps (Typical network provider leased line capacity)
Low performance WAN	400ms	2Mbps (Typical network provider leased line capacity)

Table of network classes to be modelled

10. Modelling Networked Applications

Most contemporary distributed applications communicate over IP networks. The architectural patterns used in such applications have significant influence on their performance. This is particularly notable when applications are deployed over wide area networks due to the effects of round trip time and (when TCP is employed) window sizes [Fall 05].

Performance limitations should be determined prior to deployment. One way to do this is through modelling. Three basic forms of network performance model are possible – static mathematical models, network emulation and network simulation. These forms can also be combined. This thesis places the emphasis on network simulation.

10.1. Static Mathematical Models

Static mathematical models attempt to predict network or application performance through a mathematical representation. Often the model is created by first performing network simulation or emulation, then attempting to generalise the results to a series of mathematical formulae, or a spreadsheet. This is classical engineering.

Static mathematical models are certainly useful. However, it is difficult to create models that mimic the real world behaviour of network protocol stacks, server queues or network congestion. It is relatively difficult to carry out ad hoc experiments, combining various infrastructure and application elements using them. Mathematical models are more difficult for stakeholders to understand than simulations and emulations.

10.2. Network Emulation

Network emulation is a technique whereby some functions of a network or application are reproduced in real-time using software. The software presents the same interfaces to other system components that the emulated components themselves would present. A common mode of use is to emulate a router that intentionally reduces bandwidth or introduces latency, out of order packets or errors.

Network emulation is useful, but it is difficult to use it to predict scalability. The reason for this is its real-time nature. The feasibility of experiments is tightly constrained by the computational resources available in the emulation environment.

Network emulation requires significant software and hardware resources in order to generate results. The components that are not emulated must be available in their usual form.

10.3. Network Simulation

Network simulation takes the approach of building a complete model of the network in software. The use of simulation when working with complex networks is now widespread [Heidemann 01].

Simulations do not have to be executed in real-time. Simple networks can be simulated faster than real-time, complex networks can be simulated slower than real-time. The simulation is not strongly constrained by the computational resources available in the simulation environment. It is possible to use abstraction (e.g. using “flows”) or parallel execution to enhance the scalability of the simulation [Heidemann 01] [Nicol 05].

The downside of simulation is that it must be possible to simulate all components in order to produce results. A complex simulation model is difficult to verify, and its validity may be called into question [Nicol 05].

10.4. Verification and Validation

It should be possible to verify and validate the results of a network model. Verification is a demonstration that the model is self-consistent. Validation is a demonstration that the model can predict behaviour of the real-world system [Heidemann 01].

There is a cost / benefit trade off to be made when performing verification and validation in a commercial project. For the purposes of this thesis, this was not an issue. However, when considering an approach similar to that described in this thesis in a commercial project, this question should be raised.

The model must be assessed in terms of its sensitivity to variation in parameters, and how it could be influenced by variation in the real world implementation (e.g. using a different version of router software).

Heidemann proposes various techniques for validation of a network model.

- Use alternative approaches and compare the results.
- Check that the results are reproducible.

In the case of simulation, which both this thesis and Heidemann's paper are focused on, the following techniques are also recommended:

- Use graphics, and be able to observe and inspect the simulation as it progresses.
- Introduce variations over time and make the simulation asynchronous if possible.
- Where the test machine imposes limitations, make sure that the simulation is still accurate.

11. Tool Selection

In order to create the simulation model two main tools were necessary:

- A network protocol analyser
- A network simulator

The choice of network protocol analyser was straightforward. The selection of the simulator was more thorough due to the focus of the thesis on this area.

Having selected the main tools, some integration work between them was required. This resulted in development of a small tool known as J-NAP.

11.1. Network Protocol Analyser Selection

The network protocol analyser "Wireshark" was selected (this product was previously known as "Ethereal"). This is a popular free and open source software product, but others such as tcpdump or Microsoft Network Monitor could have been used.

The acceptability of the tool was verified through a check on the number of references to the tool on IEEE Xplore, the search engine through the IEEE's journals. A large number of references were found.

11.2. Network Simulator Selection

One focus area of this thesis is network simulation. As such, it was necessary to evaluate possible candidates in order to select an appropriate and interesting tool. A table of features and other characteristics of network simulators was compiled based on a literature study [Nicol 01] [Breslau 00] [Heidemann 01] [Jansen 06] [Jansen 07]. From the table, the strongest candidate was selected. This was then subjected to a series of control experiments in order to ensure its suitability.

11.2.1. Table of Network Simulation Packages

Simulation Package	Platform	Last Release Date	General Purpose	Licensing	GUI	Programming Language for Modelling	Ability to model applications	IEEE XPlore Hits[6]
ns-2	Windows / UNIX	Sept 2006	Yes	Free	No standard GUI.	C++ or OTcl	Yes	>100
Opnet IT Guru[7]	Windows	Nov 2005	Yes	Free / Commercial[8]	Yes	?	Yes	>100
PlanetLab	Grid	2006	Yes	Closed User Group	?	?	Yes	>100
VINT	?	1997	?	?	?	?	?	>100
NEST	?	1980s	?	?	?	?	?	>100
Dummynet[9]	FreeBSD	2000	Yes	Free	No	Any	Yes	90
Lunar	Linux	2004[10]	No	?	?	Any	Yes	80
Rinse	?	2005	No[11]	?	?	?	?	72
OMNet++	Windows / UNIX	2006	Yes	Free[12]	Yes	C++	Yes	68
QualNet[13]	Windows / UNIX	2006	Yes	Commercial	Yes	C++	Yes	67
SSFNet[14]	JVM	2004	Yes	Unclear[15]	Yes	Java or C++	Yes	61
Maya[16]	?	2004	Yes	?	?	?	?	60
J-Sim	JVM	Feb 2004	Yes	Free	No	Java	Yes	43
GTNetS	UNIX	2006	Yes	Free	No	C++	Yes	29
ModelNet[17]	UNIX	2005	Yes	Free (GPL)	No	Any	Yes	19
NCTUns	Linux	May 2006	Yes	Free[18]	Yes	Any	Yes	16
Netbed[19]	?	2002	?	?	?	?	?	13
x-sim[20]	?	1999	?	?	?	?	?	8
Insane	?	1998	No[21]	?	?	?	?	6
Orbit	?	2005	No[22]	?	?	?	?	N/A
Empower[23]	Linux / grid	2003	Yes	?	Yes	?	?	N/A
REAL	?	1997	?	?	Yes	?	?	N/A
Entrapid[24]	FreeBSD	1999	?	?	?	?	?	0

Table of network simulation packages

[6] Package names that were not narrow search terms such as "Orbit" were discarded. These are marked with "N/A".
[7] A heavily limited "Academic Edition" is available for teaching purposes.
[8] Free for some research purposes (request was rejected for this thesis). Commercial licenses are believed to be approx. €40k per seat.
[9] Dummynet is, in practice, almost always used as an emulator.
[10] There may be later versions available
[11] Rinse is only intended for security research
[12] Free for academic and non-profit use. For commercial use a license is required.
[13] QualNet is a commercial product based on the GloMoSim academic research project that ceased in 2000.
[14] Some sources claim that the package is now unsupported. No longer listed on the company's website.
[15] Free for academic research and study. Commercial licenses are reasonably priced (<$500 per seat), but terms are unclear outside of USA.
[16] The Maya package does not appear to be generally available at the time of writing.
[17] Only available in beta at the time of writing.
[18] Free for non-profit, evaluation, study and research purposes. For commercial use a license is required.
[19] Netbed does not appear to be generally available.
[20] x-sim appears to have evolved into the more general x-kernel project at the University of Arizona.
[21] Special purpose IP-over-ATM simulator.
[22] Orbit is a special purpose emulator for wireless grids
[23] Empower does not appear to be generally available.
[24] Entrapid formed the basis for a commercial product from Ensim which now appears to be discontinued.

11.3. Network Simulator Selection

Based on the table, the aim was to select a network simulator with the following characteristics:

- Capable of being deployed on a conventional laptop computer (the author's primary workstation)
- Capable of modelling application characteristics (such as response time)
- A GUI (increases usability and is line with one of the recommendations for verification)
- A proven track record.
- An acceptable cost, ideally free software.
- Java API (as the author is familiar with this programming language)
- Software package that is current and maintained.

The most common choice for academic research with network simulators is ns-2. This package had a number of drawbacks for this thesis. There is no (native) Java interface. There is no standard, complete GUI. For these reasons it was not selected.

J-Sim and SSFNet were rejected on the grounds that have not been kept up-to-date.

The next best candidate was NCTUns. This package has remained somewhat obscure since its introduction in 2003, but it is a complete implementation with an interesting architecture. It employs a modified Linux kernel so that unmodified applications can be deployed on a simulated network.

Dummynet and Lunar take a similar approach to NCTUns, but neither of these tools features a complete GUI. The NCTUns package was therefore selected for the thesis.

11.4. The J-NAP Tool

The experimental method involves replaying network traces recorded using the network protocol analyser in the simulated environment. The simulator and network protocol analyser are not directly compatible so some integration effort was required. A small tool known as J-NAP (Java Network Analyser Playback) was developed as part of this thesis in order to provide the integration.

It would have been possible to perform some experiments using existing tools such as tcpreplay. However, such tools are not intended to extend a single client-server pair network trace to simulation of multiple clients accessing a single server. It is exactly this form of experiment that was required.

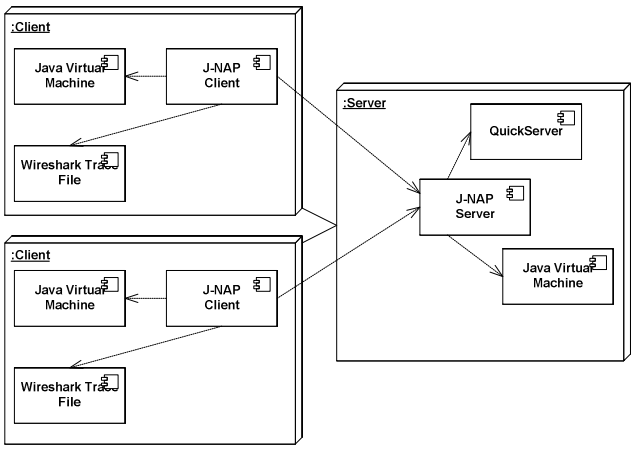
11.4.1. Goals of J-NAP

The J-NAP tool determines the total elapsed time for a sequence of client-server interactions. It can be used to directly assess the responsiveness of a client-server system under various client-server ratios and network conditions.

11.4.2. Architecture

J-NAP is a Java application consisting of a client-server pair. The server is stateless, the client is stateful. The server component is based on the open source QuickServer project [QuickServer]. This facilitated rapid development of a multi-threaded, stable server.

The client is lightweight and can run with just 2MB of working memory. This is important in terms of the scalability of the simulation itself.



UML Deployment Diagram showing J-NAP with two clients and one server

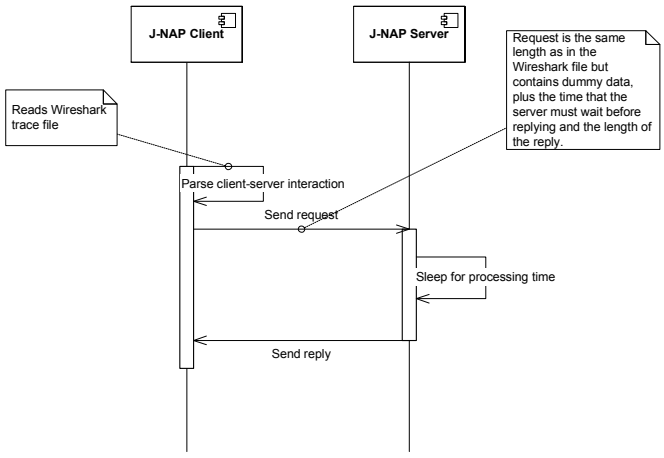
11.4.3. Operation

During operation, the client component reads a file exported from the Wireshark tool. It simulates the client traffic, sending packets to the server. The data in the packets is not the same as in the original trace, but it does contain the same volume of data. The data is replaced with instructions to the server regarding how it should reply. This includes the size of the reply and the processing delay before sending the reply. Multiple replies are possible. It is possible for the client to send data without requesting a reply.

The tool has a special "anti-skew" mode that provides a first approximation when simulating protocols such as RDP. In this mode data will begin to be dropped if the playback is slower than the original trace due to network latency or congestion. This is similar to how an adaptive protocol such as RDP works.

The J-NAP tool can replay a Wireshark trace a number of times. This is referred to as the number of runs.

The output of J-NAP is the time taken to perform the playback in milliseconds. For each run, a new value is sent to standard output.



Sequence diagram of a single J-NAP client-server interaction.

11.4.4. Size

J-NAP was coded in approximately 450 lines of code. It is a small application, only built for the purposes of this thesis. It is deployed as a 7KB JAR file, plus approximately 1MB of third-party libraries.

11.4.5. Validation

Validation of the tool was performed by capturing the traffic between the J-NAP client and server using Wireshark. This traffic should be identical in terms of data volume and timing to the originally captured traffic.

When the comparison was performed, there were slight differences in the TCP initiation as might be expected

when comparing different operating systems. The net result of this was four additional Ethernet frames during start up. However, all other traffic was shown to have limited variation in terms of data volume and timing. It was concluded that the J-NAP traffic was a reasonable simulation of the captured traffic.

Parameter	Original Trace	J-NAP Trace	Deviation (%)
Elapsed time (sec)	30.925	30.380	1.8
Packets	193	197	2.1
Mean Packets / sec	6.241	6.484	3.3
Mean Packet Size (bytes)	130	128	1.5
Total Traffic (bytes)	25134	25282	0.59
Mean Bytes / sec	812.744	832.183	2.4

J-NAP validation statistics from the network protocol analyser

```
No.     Time        Source                Destination           Protocol Info
     11 0.002704    192.168.1.133         192.168.1.208         TCP      1260 > microsoft-ds
[PSH, ACK] Seq=380 Ack=436 Win=64295 Len=76

Frame 11 (130 bytes on wire, 130 bytes captured)
Ethernet II, Src: VMware_77:95:a2 (00:0c:29:77:95:a2), Dst: VMware_c5:6d:26
(00:0c:29:c5:6d:26)
Internet Protocol, Src: 192.168.1.133 (192.168.1.133), Dst: 192.168.1.208 (192.168.1.208)
Transmission Control Protocol, Src Port: 1260 (1260), Dst Port: microsoft-ds (445), Seq: 380,
Ack: 436, Len: 76
Data (76 bytes)

0000  00 00 00 48 ff 53 4d 42 32 00 00 00 00 18 07 c8   ...H.SMB2.......
0010  00 00 00 00 00 00 00 00 00 00 00 00 01 00 7c 03   ..............|.
0020  64 00 c0 9e 0f 04 00 00 00 02 00 40 00 00 00 00   d..........@....
0030  00 00 00 00 00 00 00 04 00 44 00 00 00 00 00 01   .........D......
0040  00 07 00 07 00 00 5f 00 c0 22 ed 03               ......_.."..
```

Ethernet frame captured in the emulation laboratory

```
No.     Time        Source                Destination           Protocol Info
     15 0.049951    192.168.179.1         192.168.179.130       TCP      3030 > 5667 [PSH,
ACK] Seq=381 Ack=437 Win=65099 Len=76

Frame 15 (130 bytes on wire, 130 bytes captured)
Ethernet II, Src: VMware_c0:00:01 (00:50:56:c0:00:01), Dst: VMware_2a:84:d7
(00:0c:29:2a:84:d7)
Internet Protocol, Src: 192.168.179.1 (192.168.179.1), Dst: 192.168.179.130 (192.168.179.130)
Transmission Control Protocol, Src Port: 3030 (3030), Dst Port: 5667 (5667), Seq: 381, Ack:
437, Len: 76
Data (76 bytes)

0000  58 58 58 58 58 58 58 58 58 58 58 58 58 58 58 58   XXXXXXXXXXXXXXXX
0010  58 58 58 58 58 58 58 58 58 58 58 58 58 58 58 58   XXXXXXXXXXXXXXXX
0020  58 58 58 58 58 58 58 58 58 58 58 58 58 58 58 58   XXXXXXXXXXXXXXXX
0030  58 58 58 58 0d 0a 72 65 70 6c 79 20 32 2e 38 30   XXXX..reply 2.80
0040  39 39 39 38 45 2d 34 20 38 38 0d 0a               9998E-4 88..
```

Corresponding Ethernet frame generated by J-NAP. Note the dummy data (the "X"s) plus instructions for the server in terms of when to reply and how much data to send in the reply.

12. Method for Network Simulation, based on Data from Emulation and Protocol Analysis.

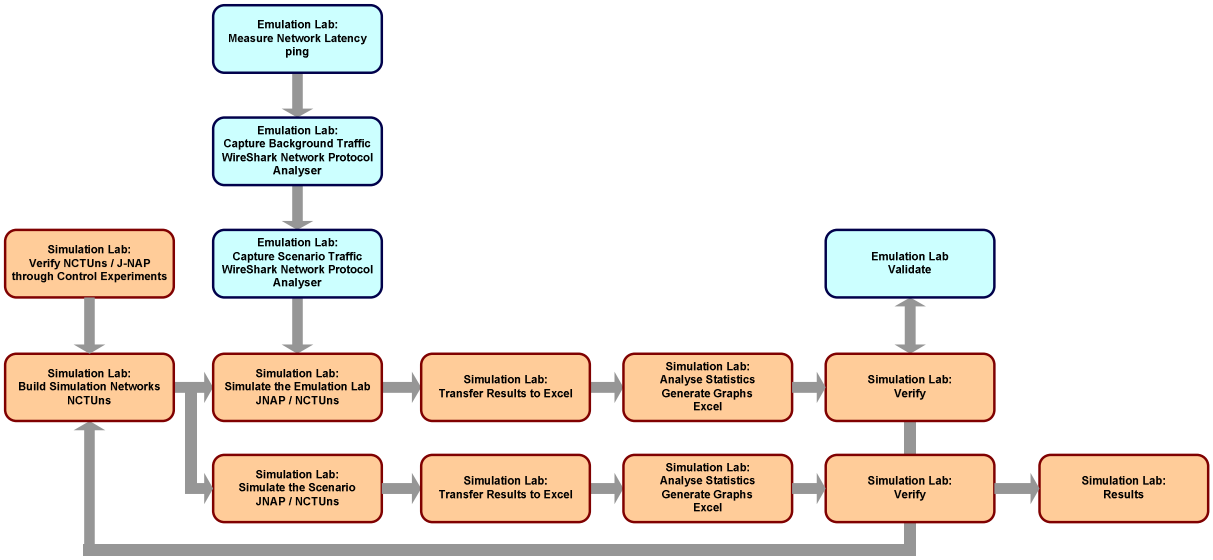
12.1. Overview

The method used was based on samples of network traffic captured in an emulation laboratory. These samples were used as input for simulation. Simulation was used in order to predict performance under a range of conditions, many of which are impractical to emulate.

Through comparison to simple mathematical models and by examining logs, it was possible to verify the results of the simulation. This allowed confirmation that the simulation worked as intended.

Validation was performed by comparing results from emulation to results from simulation with similar parameters. It was expected that these results would be the same. Any divergence should be attributed to cumulative errors in the method.

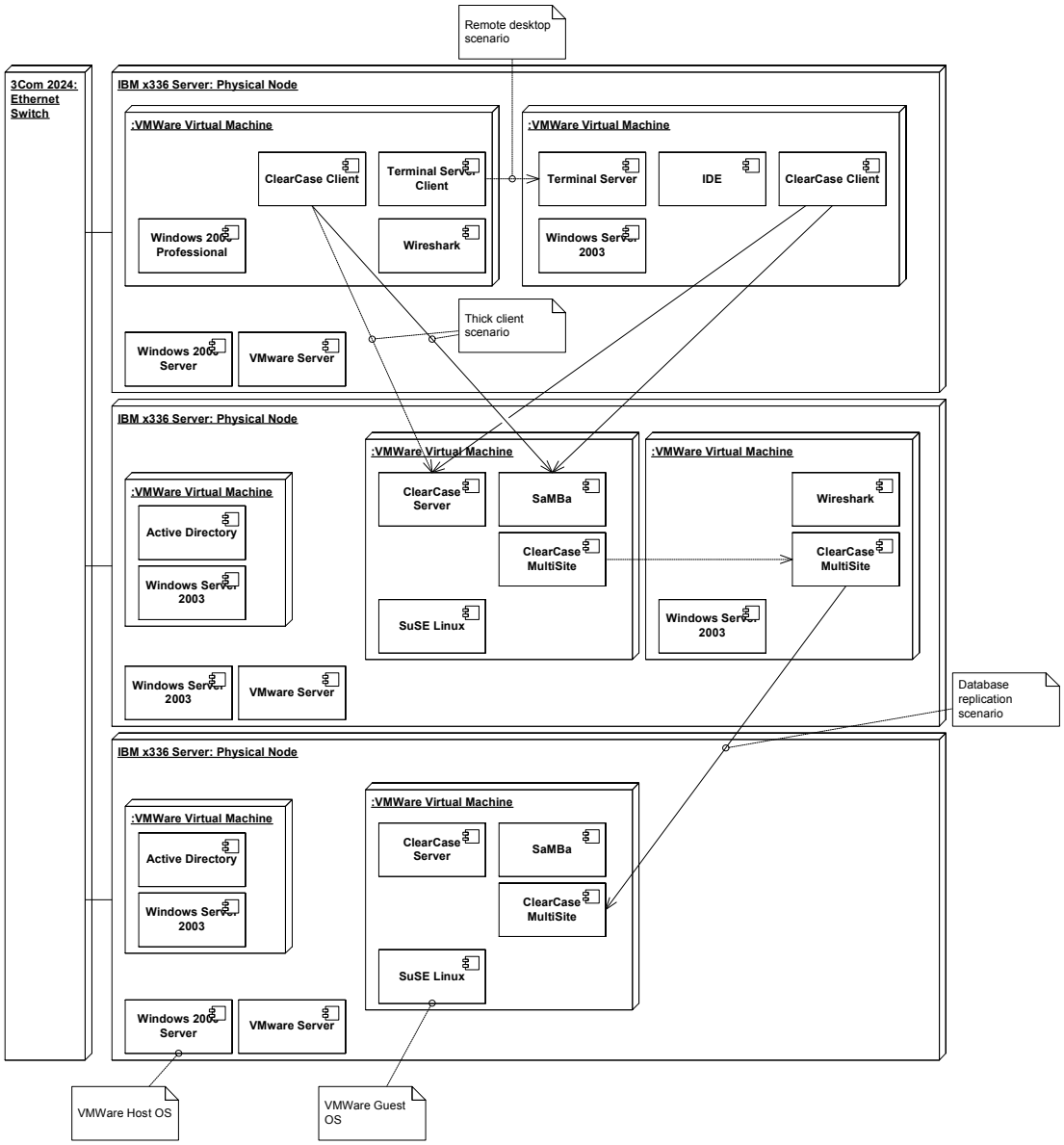
The complete process is shown in the diagram below. The individual steps are outlined in the following sections.



Emulation, simulation, verification and validation method

12.2. Emulation and Traffic Capture

12.2.1. Configuration



UML Deployment Diagram showing operation of the emulation laboratory

As part of the SDLC project, the Rational toolset was deployed on the emulation environment as shown in the diagram above.

In order to perform the experiments, a dedicated ClearCase database known as a VOB (Versioned Object Base) was configured and loaded with test data. The Wireshark network protocol analyser was installed on the client systems.

12.2.2. Establish Network Latency

Capturing traffic for each scenario proceeded in a number of steps. First the client-server latency was established using the well-known ping utility. The reason for this is to allow any delay caused by the network itself to be taken into account when working with the captured traffic.

12.2.3. Set up Capture Filter

The next step was to configure the network protocol analyser to capture only the traffic of interest. This was performed by setting parameters for the network addresses and protocols to be captured. These settings were

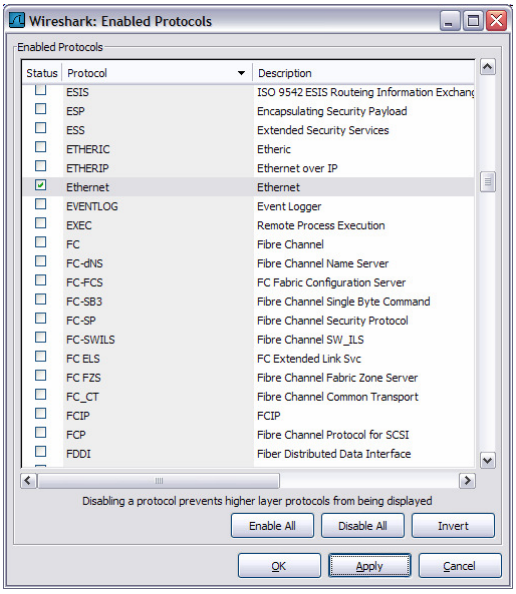
based on documentation of the products in use.

12.2.4. Capture Background Traffic

Any background client-server traffic must be established. This was performed by simply doing nothing other than running the network protocol analyser for a period of time. The reason for this part of the process is to ensure that the sample traffic is actually caused by using the tools.

12.2.5. Capture Scenario Traffic

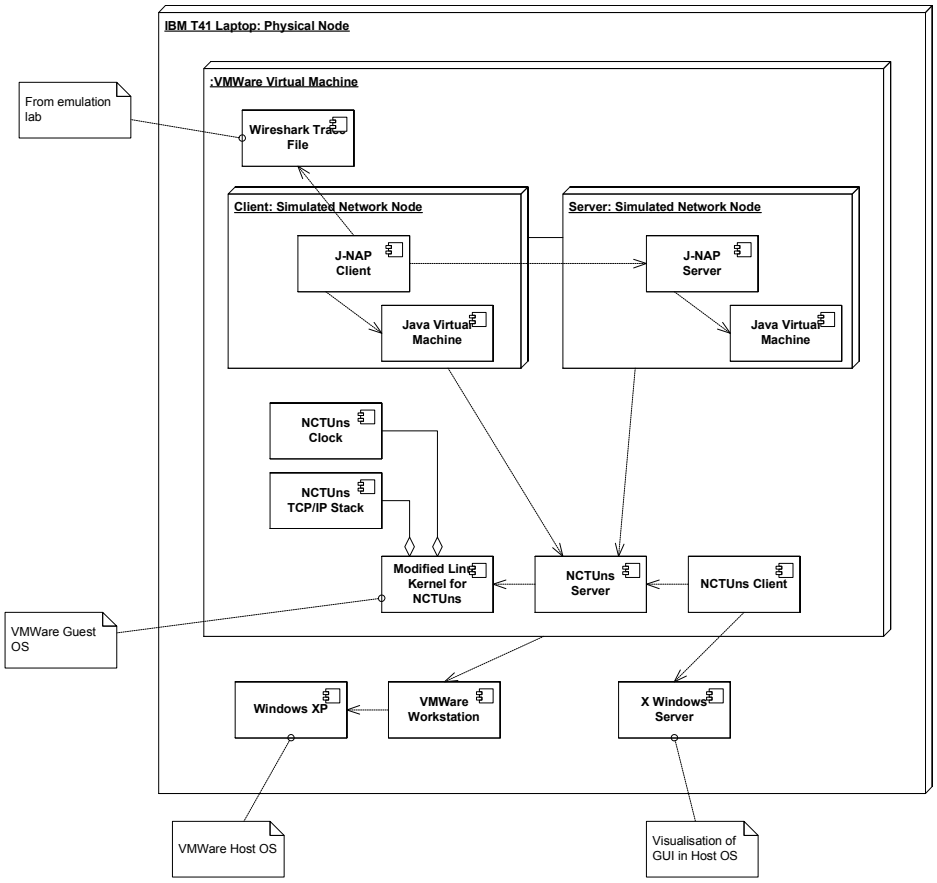
Having established the latency and the background traffic, it was then possible to capture the actual traffic for a given scenario. This is a case of starting the network protocol analyser with the appropriate filter, using the tools according to the scenario, and stopping the network protocol analyser. It was possible to export the captured traffic in a file format that could be interpreted by the J-NAP tool.



Setting up a filter for export using the Wireshark tool.

12.3. Simulation

12.3.1. Configuration



UML Deployment Diagram showing operation of the simulation laboratory

The simulation laboratory consisted of the NCTUns simulator combined with the J-NAP tool. This was deployed as a virtualised system on an otherwise normal Windows XP workstation.

For each scenario, the file output by the network protocol analyser was transferred to the simulation system. This file is the input for the J-NAP playback tool.

12.3.2. Control Experiments

In order to ensure that the simulation laboratory configuration was capable of generating reliable results, a series of control experiments were carried out. These involved replaying control Wireshark capture files over simple networks. The control experiments were also used to explore the limits of the simulation environment,

12.3.3. Configure Simulation Networks

The NCTUns package includes a GUI that can be used to construct complex models of IP networks. This was used to build a model of the desired network for the SDLC project including multiple WANs and LANs. This model included a configuration that was similar to the emulation laboratory. This was used for the purposes of validation.

12.3.4. Perform Simulation

An actual Wireshark trace was taken and used as the basis for simulation runs. This was performed by setting up appropriate configuration of the models using the NCTUns GUI.

The output of the simulation consists of files generated by J-NAP and NCTUns itself. The data from J-NAP gave the time taken for a particular client-server interaction to take place. The data from NCTUns gave the bandwidth consumed at given points in the network. This could be used to make predictions about system scalability.

12.3.5. Analyse Statistics

The data from J-NAP and NCTUns were transferred to the Microsoft Excel spreadsheet application. This application was used to analyse and visualise the data.

12.3.6. Verification and Validation

Verification and validation took place based on the techniques recommended in [Heidemann 01]. These were as follows:

Recommendation	Technique Applied
Assess sensitivity to parameter variation	Experiments were performed to assess variation of RTT on a small, control model (see section 18.2).
Use alternative approaches and compare the results.	This recommendation was not followed.
Check that the results are logical and reproducible.	The logs and statistics were analysed in order to check for out of range values or excessively high standard deviation. Experiments were occasionally repeated in order to check for reproducibility.
Use graphics, and be able to observe and inspect the simulation as it progresses.	The NCTUns package allowed the use of graphics, such that observation and inspection of the simulation as it progressed was possible.
Introduce variations over time and make the simulation asynchronous if possible.	It was not possible to do this (easily) with the NCTUns package.
Where the test machine imposes limitations, make sure that the simulation is still accurate.	The load on the hardware was monitored while the experiments were running using the “top” utility.

Recommended verification techniques

Recommendation	Technique Applied
For well-specified protocols, direct comparison with a real world system can be used for the purposes of validation.	Validation was performed by comparing the simulation model that was close to the emulation laboratory with the actual emulation laboratory.

Recommended validation techniques

Problems revealed during verification and validation resulted in iterations of the simulation and modelling process. If no problems were found, then the results from the scenario were recorded.

12.4. Limitations of the Method

12.4.1. Caching, Adaptive Protocols and other Optimisations

The simulation does not take into account dynamic optimisations in the applications under investigation. If the applications make extensive use of caching or adaptive protocols then they may scale significantly better than the simulation would suggest. This limitation could be overcome in the future by attempting to simulate this behaviour in the J-NAP software.

12.4.2. Virtualisation

Both the simulation and emulation labs are heavily reliant on operating system virtualisation using the products from VMware. This may mean that the laboratory configurations are imprecise. There may be unexpected timing and clock effects [VMware 05]. The control experiments are intended to demonstrate that these effects have not distorted the results.

12.4.3. Operating System Divergence

Variations in TCP/IP stacks on different operating systems may make accurate emulation or simulation problematic. Variations of this type could influence the throughput results by as much as one order of magnitude [Jansen 06].

The NCTUns simulator inherently makes use of the Linux TCP/IP stack. It will accurately simulate Linux applications, but may not accurately simulate other operating systems. The control experiments are intended to demonstrate that these effects have not distorted the results.

12.4.4. Relative vs. Absolute Results

Simulation can be used to compare a number of alternatives, rather than to predict absolute performance. Validation of the model is much easier in such a case [Heidemann 01]. This thesis presents both comparative and absolute results. It is likely that the comparative results are more robust.

12.4.5. Scalability

The simulation environment itself is subject to scalability concerns. Simulation is not real-time, and so is much less susceptible to capacity constraints than emulation. This is one of the reasons that simulation is more suitable for modelling large networks. Nevertheless, the results must be delivered in a reasonable time, so there are concerns in this area.

Simulating a large network requires more resources than simulating a small network. This is particularly true in the case of the NCTUns simulator, or any simulator with a similar architecture. Real applications are executed on the simulated nodes. The system may not have sufficient memory or may be limited by other factors such as the maximum number of file handles or processes that the operating system can support.

12.4.6. Limits of Validation

The experimental method used in this thesis would have been stronger if use had been made of a network emulation package in the emulation laboratory. This would have facilitated stronger validation of the simulation results. The original intention was to use the netem emulation package that has recently been included in the Linux kernel for this purpose.

Unfortunately, attempts to set up configurations using this package were not successful. The netem package was not included in the older distribution of the Linux kernel that was used to set up the emulation environment. Various attempts to build and install a new Linux kernel including netem were not successful.

Part III. Modelling the Optimisation and Extension Problems

13. Introduction

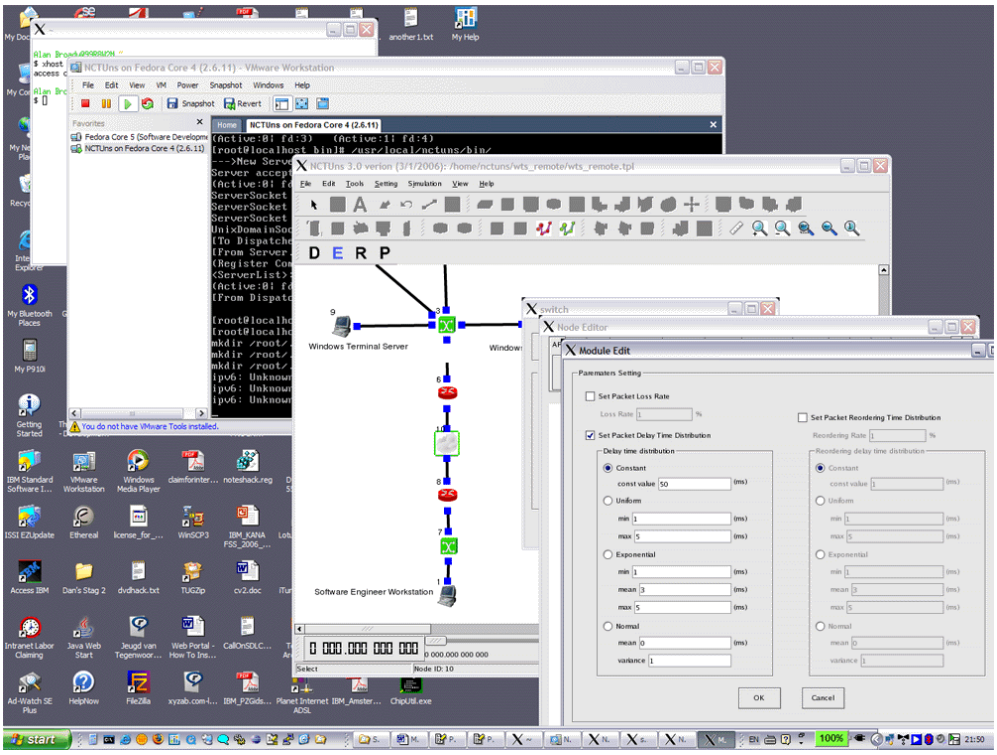
Part III of the thesis applies the parameters and general method described in Part II to the specific problems from the case study identified in Part I. The following areas are dealt with:

- Exact laboratory configurations and control experiments
- Model of the case study infrastructure
- Model of work patterns of software engineers
- Test data
- Simulation of the optimisation problem
- Simulation of the extension problem

14. Laboratory Hardware and Software Configurations

14.1. Simulation Laboratory

Simulation was carried out using a straightforward hardware configuration consisting of a standard laptop computer (IBM T41). An external USB 2.0 hard disk drive was used to provide capacity for the VMware images.



The network simulation toolset in use

The laboratory consisted of the following hardware (see diagram in section 12.3.1):

Hardware Component	Type	Details
Laptop	IBM T41	1 x CPU, Intel Pentium M, 1.7Ghz 1GB RAM Internal 37.2GB hard disk External 250GB hard disk connected via USB 2.0

Hardware used in the simulation laboratory

14.2. Emulation Laboratory

The emulation laboratory was set up for the purposes of the SDLC project. The main purpose of the environment was to act as a "proof of concept". This environment was used, in slightly modified form, for the purposes of the experiments that form part of this thesis.

The emulation laboratory did not include components that allowed emulation of different network characteristics (see section 12.4.6).

The laboratory consisted of the following hardware (see diagram in section 12.2.1):

Hardware Component	Type	Details
Server 1	IBM x336	4 x CPU, Intel Xeon, 3.6GHz 3.25GB RAM
Server 2	IBM x336	4 x CPU, Intel Xeon, 3.6GHz 3.25GB RAM
Server 3	IBM x336	4 x CPU, Intel Xeon, 3.6GHz 3.25GB RAM
Ethernet Switch	3Com Baseline Switch 2024	24 ports 100BaseT

Hardware used in the emulation laboratory

15. Control Experiments

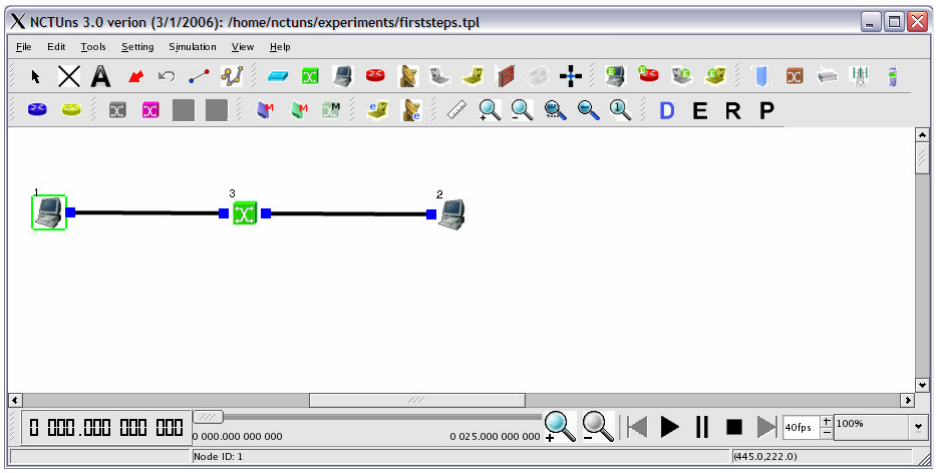
A bottom up approach based on “hierarchical composition” [Heidemann 01] was used. This approach started with simple control experiments, then gradually increased the complexity of the models.

15.1. Control Experiments Phase 1: ttcp on NCTUns

In order to establish reliable and accurate operation of NCTUns, a series of control experiments were carried out. The initial experiments were based on the use of the standard ttcp UNIX tool that is bundled with the NCTUns distribution.

15.1.1. Control Experiment 1.1: Two Nodes

The first experiment consisted of the simplest network that can be simulated with NCTUns. This consists of two nodes connected by an Ethernet switch.



Simple network model built using NCTUns

Scripts were created that used that standard ttcp tool in order to transmit and receive TCP/IP traffic. These scripts are shown below. Note the unusual IP addresses within the simulated network.

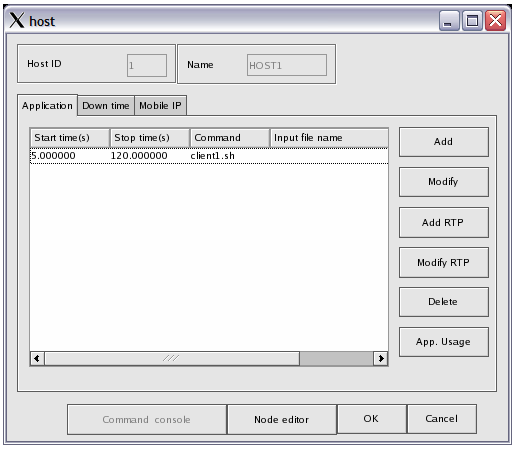
```
#!/bin/sh
/usr/local/nctuns/tools/ttcp -t 1.0.1.2 < ~nctuns/request.txt
sleep 0.001
/usr/local/nctuns/tools/ttcp -r
```

Client script

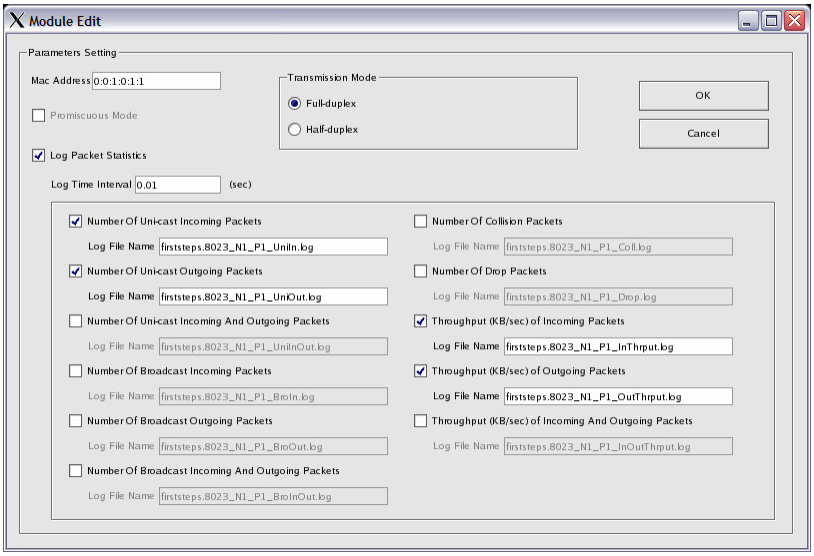
```
#!/bin/sh
/usr/local/nctuns/tools/ttcp -r
sleep 0.5
/usr/local/nctuns/tools/ttcp -t 1.0.1.1 < ~nctuns/response.txt
```

Server script

The NCTUns simulator was configured to run these scripts on the two nodes. It was also configured to log the volume of traffic transmitted and received on each node.

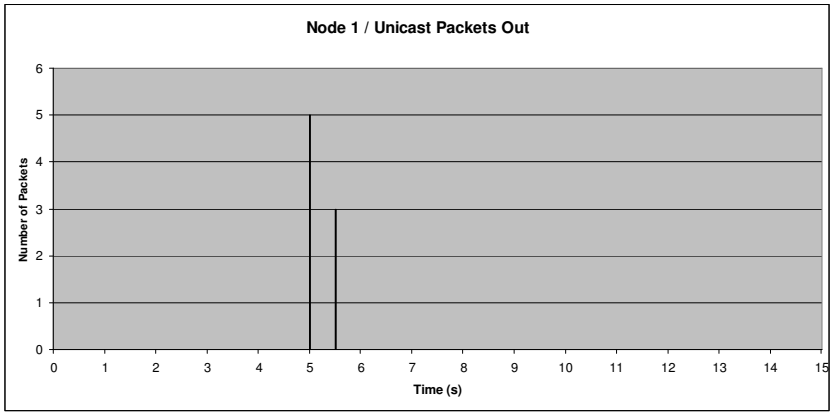


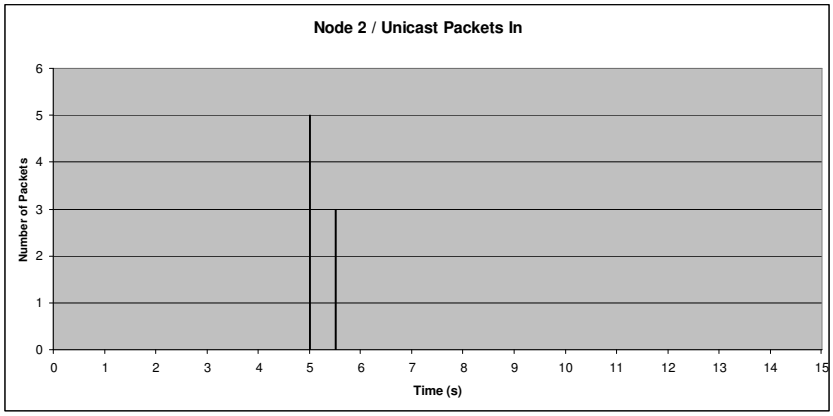
Configuring the simulator to initiate the client script



Configuring the simulator to log the traffic pattern

Having completed configuration, the simulation was executed. The logs from the simulator were transferred via FTP from the VMware guest to the VMware host operating system. The data was imported into the Excel spreadsheet package for analysis. For each of the two nodes, a plot was made of the traffic in and out.



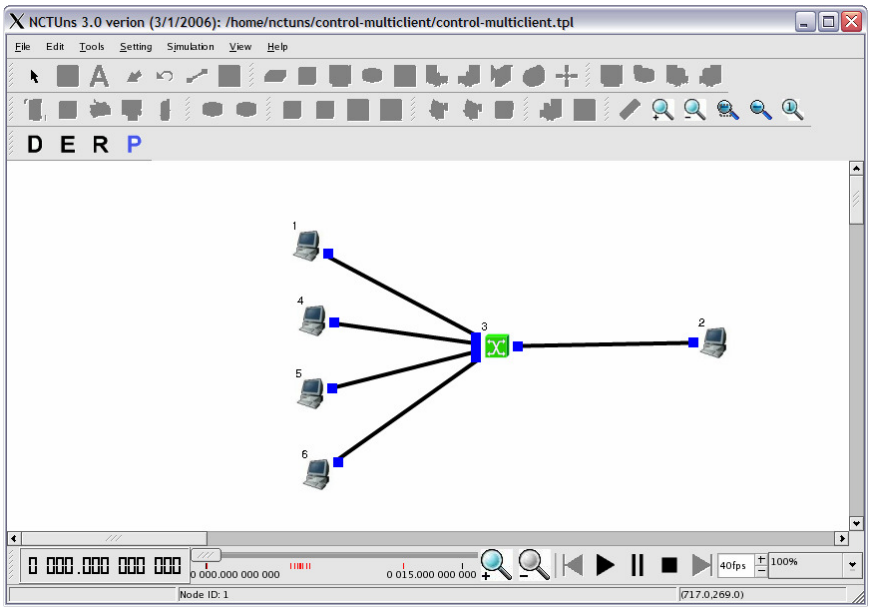


Results of control experiment 1.1

From the graphs, it is clear that there are two distinct client-server interactions, one initiated at 5 seconds in the simulation, the other at 5.5 seconds in the simulation. This corresponds to the expected behaviour from the scripts.

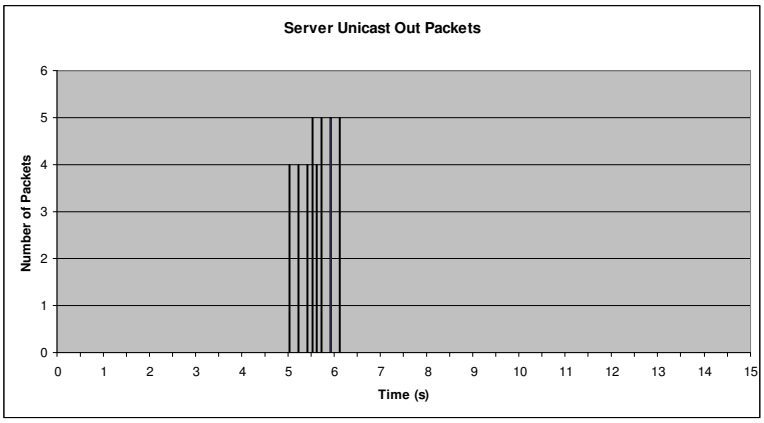
15.1.2. Control Experiment 1.2: Four Clients with One Server

The complexity of the model was increased. A model with four clients making concurrent requests to a server was created. This was simulated by creating a server script that spawned a number of sub-processes in order to serve the various clients.



Single server (node 2) with multiple clients in NCTUns

The experiment was repeated with the different clients set to initiate their connection at 200ms intervals. This produced the expected trace with peaks at that interval, plus peaks for the replies, 500ms following each client connection. This experiment also demonstrated that NCTUns was capable of working correctly with sub-processes.



Results of control experiment 1.2

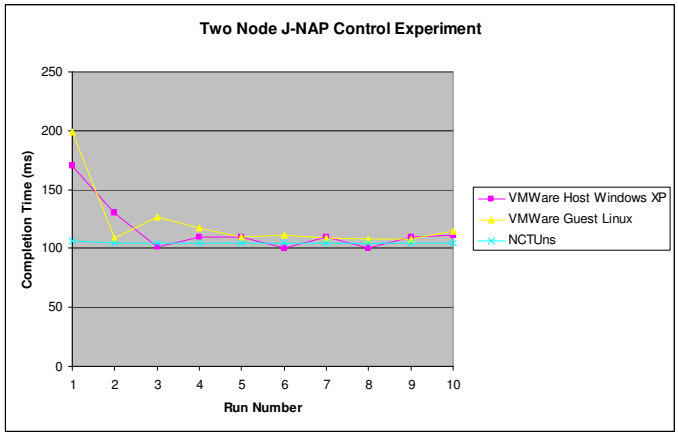
15.2. Control Experiments Phase 2: J-NAP in the NCTUns Environment

Having established that it was possible to produce some accurate results using NCTUns, the focus shifted to the combination of NCTUns with J-NAP. This series of experiments was based on using hand coded test files in Wireshark format as input for J-NAP.

15.2.1. Control Experiment 2.1: Comparison of J-NAP in Three Environments

This experiment was a comparison of J-NAP operation under three different environments:

- On the VMware host operating system, Windows XP Professional.
- On the VMware guest operating system, Fedora Core Linux modified for NCTUns.
- On simulated nodes within the NCTUns environment.

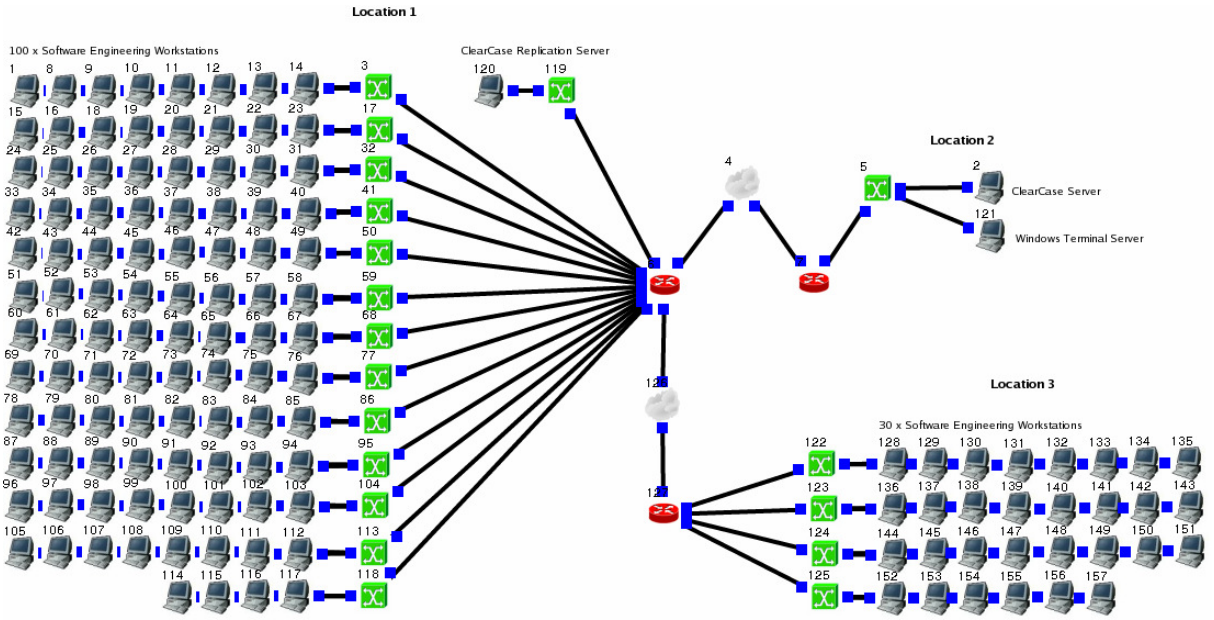


Results of control experiment 2.1

This experiment demonstrated that J-NAP performed as expected when using a known, simple Wireshark format input file. It also demonstrated that results were similar, but not identical under the three different environments. It is interesting to note that start-up effects can vary in a simulated environment.

16. Model of Case Study Infrastructure

In order to perform the simulations, a model of the SDLC project infrastructure was created using the NCTUns tool. The model comprised three locations – the hub and two software engineering spokes.



The infrastructure model created using NCTUns

16.1. Location 1

Location 1 is a typical software engineering spoke. It represents a location where 100 software engineers are employed.

Location 1: Software Engineering Spoke	
Component	**Number**
Workstation	100
Switch	13
Replication Server	1
Router	1

16.2. Location 2

Location 2 is the hub. It contains the central servers with connectivity to all of the software engineering spokes.

Location 2: Data Centre	
Component	**Number**
ClearCase Server	1
Windows Terminal Server	1
Switch	1
Router	1

16.3. Location 3

Location 3 is a second software engineering spoke. It is included for the purposes of modelling peer-to-peer communication and collaboration between the software engineers. It contains 30 workstations that represent the counterparties of the 30 workstations in location 1 that are taking part in communication and collaboration activities.

Location 3: Software Engineering Spoke	
Component	**Number**
Workstation	30
Switch	4
Router	1

17. Work Patterns of Software Engineers

In order to create a model for a software engineering environment, it is necessary to establish what software engineers do all day. This is more difficult that it might appear. No Tayloristic analysis of software engineers' work patterns emerged during the literature study. Performing experiments to determine work patterns would be possible, but beyond the scope of this thesis. A model is proposed based on some limited evidence in combination with the experience of the author.

Software engineers do not only perform component design, implementation and test. They will also be involved in other activities – taking part in meetings, gossiping at the water cooler etc. This is significant because in a global team these activities may be performed using communication and collaboration tools.

Office workers spend large parts of their working day involved in communication and collaboration activities [Frohlich 95]. It was decided to use the value of 30% for such activities in the model. This is based on the author's own experience, combined with the assumption that software engineers spend less time on these activities than the typical office worker.

A model was created with 100 workstations in location 1. This is the appropriate order of magnitude and allows a one to one correlation between the percentage of time spent on any activity and the number of workstations that should be simulated.

Activity	% Time Spent on Activity	Node Identifiers in Simulation Model
Use of IBM Rational toolset		
File check out	1	001
File check in	1	008
Perform a file comparison	1	009
Start the IDE	1	010
Stop the IDE	1	011
Save a file (note that saving is generally required prior to compilation, so occurs frequently)	7	012, 013, 014, 015, 016, 018, 019
Open a file	10	020, 021, 022, 023, 024, 025, 026, 027, 028, 029
Text file editing Programming, configuration file editing etc.	24	030, 031, 033, 034, 035, 036, 037, 038, 039, 040, 042, 043, 044, 045, 046, 047, 048, 049, 051, 052, 053, 054, 055, 056[25]
GUI usage Modelling, testing, build, configuration etc.	24	057, 058, 060, 061, 062, 063, 064, 065, 066, 067, 069, 070, 071, 072, 073, 074, 075, 076,

[25] Node 56 was set up to simulate very low intensity use of text editing. This acted as a means to measure acknowledgement response time.

		078, 079, 080, 081, 082, 083
Use of communication and collaboration tools		
IM	3	084, 085, 087
Remote desktop viewer	1	088
Audioconferencing	10	089, 090, 091, 092, 093, 094, 096, 097, 098, 099
Videoconferencing	3	100, 101, 102
Using social networking tools	3	103, 105, 106
E-mail	10	107, 108, 109, 110, 111, 112, 114, 115, 116, 117

Assignment of nodes in the model to software engineering activities

17.1. Test Data

In order to provide an example of a real-world project, the source code from a well-known open source project was used in the experiments. The project in question was the Tomcat Java Web Container from the Apache Software Foundation.

The use of this test data could be questioned in the context of a commercial software project. The main objection that could be raised is that the test data consisted only of source code and configuration files. It did not contain project documentation as is typical in commercial software engineering. This is an important distinction because these files are binary and large (typically Microsoft Office formats). As such, they present problems for version control systems because it is not easy to compute “deltas” for them in the same way as can be done for source code.

18. Optimisation Problem: ClearCase / IDE Architectural Patterns

Three architectural patterns for the Rational tools were acceptable because they met the functional requirements (see section 4.2). The architectural patterns that are ultimately selected must also meet the application demands for responsiveness (see section 8.1). This part of the thesis attempts to determine through experiment the effect of using the different patterns in combination with the four classes of network identified in section 9.5. The experiments were carried out according to the method described in section 12.

18.1. Emulation Laboratory Tests

Data capture was performed in the emulation laboratory. This provided data that could be transferred to the simulation laboratory. Prior to performing data capture, it was necessary to establish the characteristics of the emulation laboratory.

18.1.1. Network Latency

A network latency test was performed using the standard ping tool. It revealed sub-millisecond delay, which is negligible for the purposes of these experiments.

```
C:\>ping 192.168.1.231

Pinging 192.168.1.231 with 32 bytes of data:

Reply from 192.168.1.231: bytes=32 time<10ms TTL=128
Reply from 192.168.1.231: bytes=32 time<10ms TTL=128
Reply from 192.168.1.231: bytes=32 time<10ms TTL=128
Reply from 192.168.1.231: bytes=32 time<10ms TTL=128

Ping statistics for 192.168.1.231:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum =  0ms, Average =  0ms

C:\>ping 192.168.1.208

Pinging 192.168.1.208 with 32 bytes of data:

Reply from 192.168.1.208: bytes=32 time<10ms TTL=64
Reply from 192.168.1.208: bytes=32 time<10ms TTL=64
Reply from 192.168.1.208: bytes=32 time<10ms TTL=64
Reply from 192.168.1.208: bytes=32 time<10ms TTL=64

Ping statistics for 192.168.1.208:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum =  0ms, Average =  0ms
```

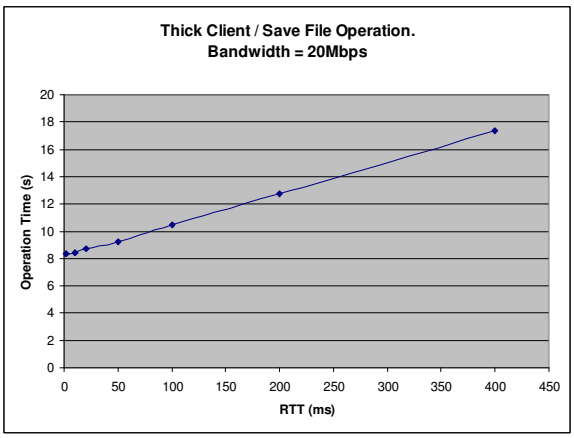
Results of the network latency test

18.1.2. Background Traffic

Background traffic was measured by performing a traffic capture without using the tools. In all cases no TCP traffic between the client and server was recorded.

18.2. Simulation Laboratory Tests

A series of experiments were performed in order to verify the simulation environment. The save file operation was simulated with various round trip times between the client and server. In each case the total time to perform the operation was recorded.



Variation in operation time with round trip time

The results showed linear proportionality with a constant offset between round trip time and operation time. The results also showed that for a round trip time close to zero, the operation time was close to the value recorded in the emulation environment. This was the expected result, providing verification of the configuration.

18.3. Thick Client Architecture

A series of experiments were conducted to evaluate the thick client architecture.

18.3.1. Emulation Environment Data Capture

Traffic was captured for the various scenarios, producing results as follows:

Scenario	Description	Elapsed Time (s)[26]	Total Traffic Volume (KB)
Check out one file	Transfer a single source code file from the version control repository to the client system.	11.4	65.3
Check in one file	Transfer a single source code file from the version control repository to the client system.	11.8	26.6
Perform a file comparison	Compare differences between the working copy of the file with the version control repository.	12.0	20.8
Start the client application	Start up the ClearCase tool.	12.4	24.0
Stop the client application	Exit the ClearCase tool.	0.00	0.053
Open a file	Open a file that is under source control.	30.9	25.1
Save a file	Save a file that is under source control.	8.33	5.68

Results of traffic capture for the thick client architecture

18.3.2. Simulation: Complete Infrastructure Model

In order to evaluate the thick client architecture, a simulation was performed based on the complete infrastructure model described in section 16. A mix of operations was performed using this model, weighted according to the anticipated usage pattern. The operations were performed repeatedly by the nodes over a one minute period. The experiment was repeated on the four classes of network identified in section 9.5.

In order to give a simple measure of performance, a weighted mean response time was calculated in each case. The weighting was based on the anticipated usage pattern. This value was also calculated for the data from the emulation environment in order to allow validation of the model.

The bandwidth consumption pattern was recorded using the facilities built into the NCTUns package.

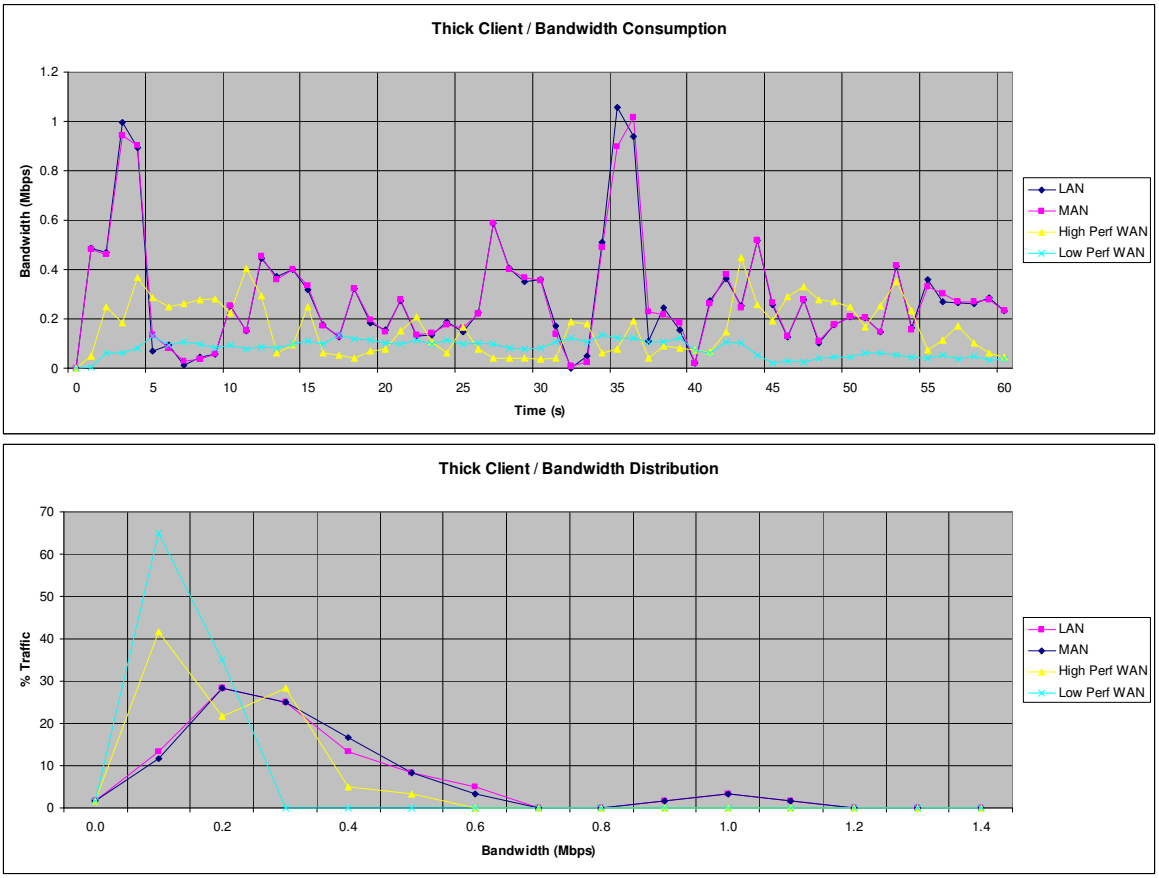
[26] It is possible that these times would be lower with repeated usage due to the effects of caching at various layers.

Variation in response time for different classes of network using the thick client architecture

The processing response time results show that the LAN and MAN classes of network gave almost identical performance. The results show that the LAN added approximately 2 seconds to the response time compared to the emulation environment. The expected result was that the LAN simulation should give the same results as the emulation environment. This divergence, less than 10%, should be attributed to experimental error.

The high performance WAN added approximately 6 seconds to the response time. The low performance WAN added approximately 24 seconds to the response time.

The bandwidth consumption pattern over the WAN link between the hub and the spoke was recorded. This indicates the extent to which bandwidth plays a role in the performance of this architecture.



Bandwidth consumption for the different classes of network using the thick client architecture

Network class	Bandwidth Available (Mbps)	Peak Bandwidth (Mbps)	% Bandwidth Consumed
LAN	1000	1.06	0.106
MAN	155	1.02	0.658
High performance WAN	20	0.448	2.24
Low performance WAN	2	0.135	6.75

Bandwidth consumption for the different classes of network using the thick client architecture

It should be noted that the acknowledgement response time was not measured in this case. That is because it is not affected by the network performance for this architectural pattern.

18.4. Thin Client Architecture

A series of experiments were conducted to evaluate the thin client architecture.

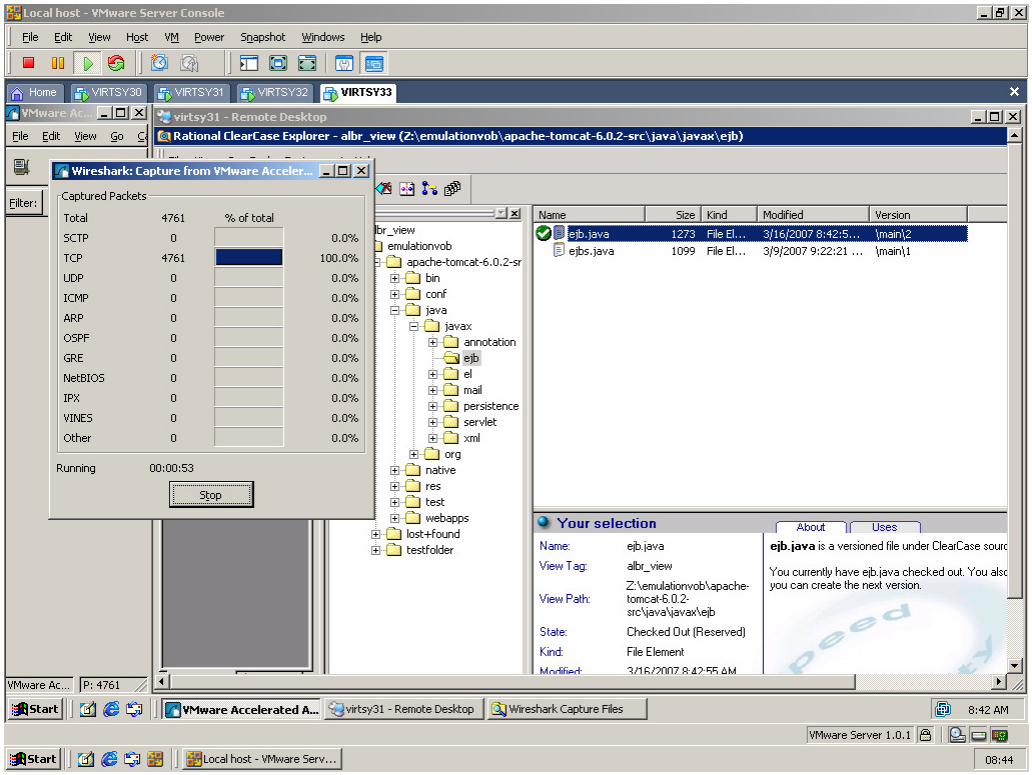
18.4.1. Emulation and Data Capture

Traffic was captured for the various scenarios, producing results as follows:

Scenario	Description	Elapsed Time (s)	Total Traffic Volume (KB)
Check out one file	Transfer a single source code file from the version control repository to the client system.	21.8	26.7
Check in one file	Transfer a single source code file from the version control repository to the client system.	41.3	36.1
Perform a file comparison	Compare differences between the working copy of the file with the version control repository.	58.4	54.0
Start the client application		37.2	20.1
Stop the client application		57.9	5.8
Open a file	Open a file that is under source control.	10.6	3.9
Save a file	Save a file that is under source control.	6.887	12.5
Perform graphical modelling	Use an IDE to perform UML modelling.	60.8	130
Perform component implementation.	Use an IDE to perform text editing.	58.0	150
Acknowledgement response time test	Perform controlled text editing, making one small change every five seconds. This gives a network trace with a clear “cause and effect” relationship that can be used to determine the acknowledgement response time, and so the responsiveness of the GUI.	78.4	10.6

Results of traffic capture for the thin client architecture

It should be noted that the elapsed times are much longer than for the thick client architecture. This is because they include time that was required for the user to interact with the application. In the case of the thick client architecture, these interactions did not cause network traffic and so are not included in the total time.



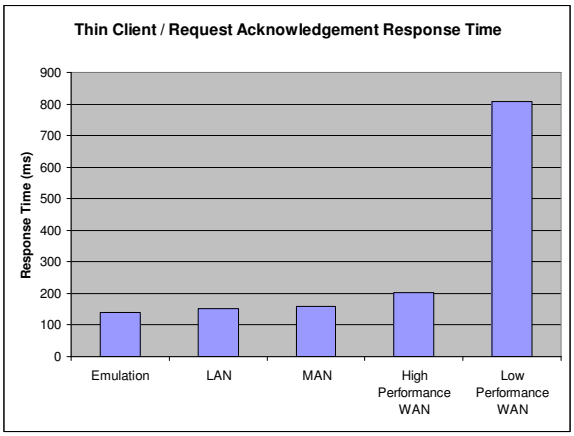
Capturing data during a trial using Windows Terminal Server

18.4.2. Simulation: Complete Infrastructure Model

With this architectural pattern there is little value in comparing the processing response time. The main client-server interaction that determines the processing response time takes place over a LAN between the terminal server and the ClearCase VOB server.

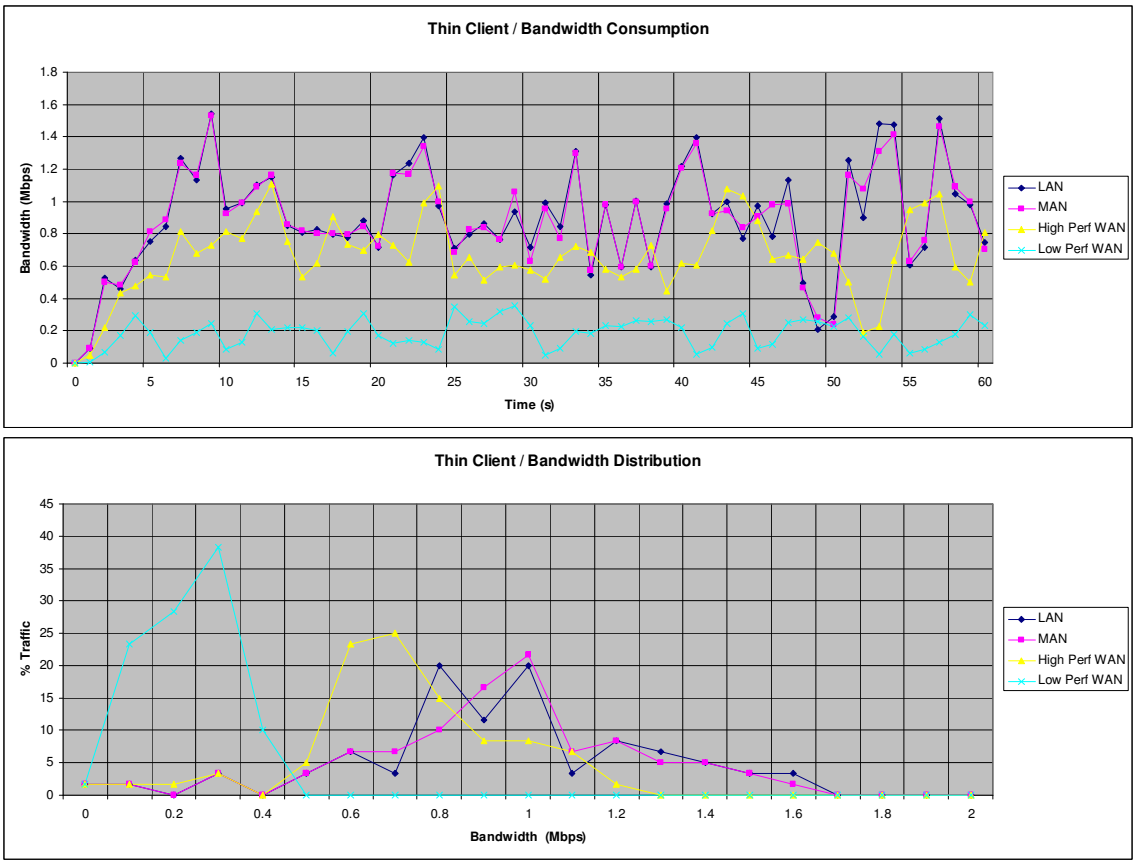
More interesting is the acknowledgement response time. This is a measure of the "crispness" of the response perceived by the user [Tolia 06]. This was measured by using a prepared network traffic pattern. With this traffic pattern, it is possible to associate a set of network packets with a certain user interface interaction.

Based on an analysis of the data produced by the simulator, the acknowledgement response time for each class of network was established.



Variation in response time for different classes of network using the thin client architecture

The bandwidth consumption pattern over the WAN link between the hub and the spoke was recorded. The bandwidth consumption pattern indicates the extent to which bandwidth plays a role in the performance of this architecture.



Bandwidth consumption for the different classes of network using the thin client architecture

Network class	Bandwidth Available (Mbps)	Peak Bandwidth (Mbps)	% Bandwidth Consumed
LAN	1000	1.54	0.154
MAN	155	1.53	0.99
High performance WAN	20	1.10	5.50
Low performance WAN	2	0.354	17.7

Bandwidth consumption for the different classes of network using the thin client architecture

18.5. Database Replication Architecture

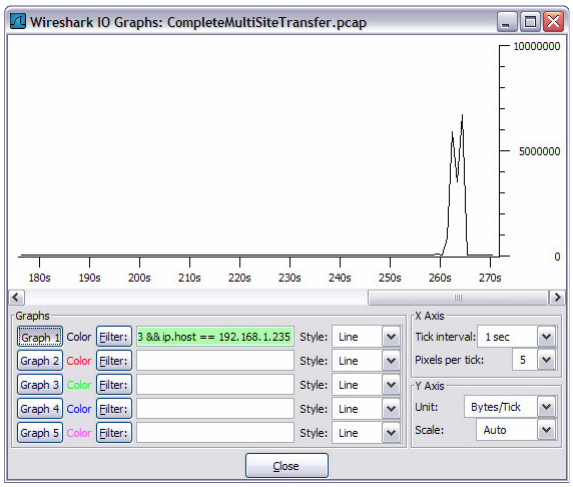
A series of experiments were conducted to evaluate the database replication architecture.

18.5.1. Emulation and Data Capture

Traffic was captured for the following scenarios:

Scenario	Description	Elapsed Time (s)	Traffic Volume (KB)
Replicate an entire source code archive	Initial replication between two sites.	35.3	16944
Replicate a source code archive with a single change.	Minimum replication between two sites.	29.8	6.967

Results of traffic capture for the database replication architecture



Bandwidth consumption in the emulation environment for replication of an entire source code archive.

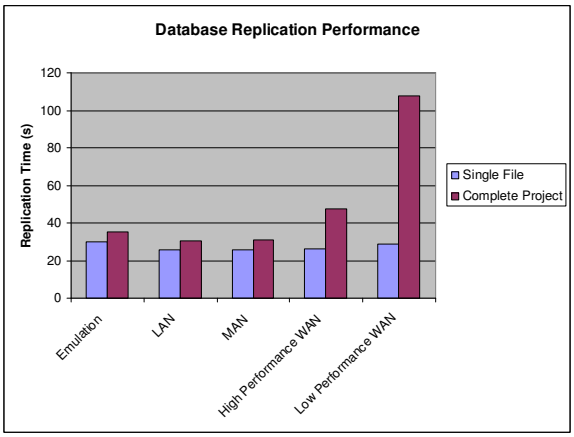
The traffic capture revealed three clear phases, which are interpreted as follows:

Phase	Period	Traffic Observed	Interpretation
Initiation	< 1s	Minimal traffic setting up the connection.	Application starts up, checks it can connect to the other server. Perhaps receives the date of last update.
Analysis	Approx 30s	None	Application is analysing the version control archive to determine changes since the last replication.
Transfer	Approx 5s	High bandwidth TCP flowing one way.	The files that have changed are transferred.

Phases in the database replication process

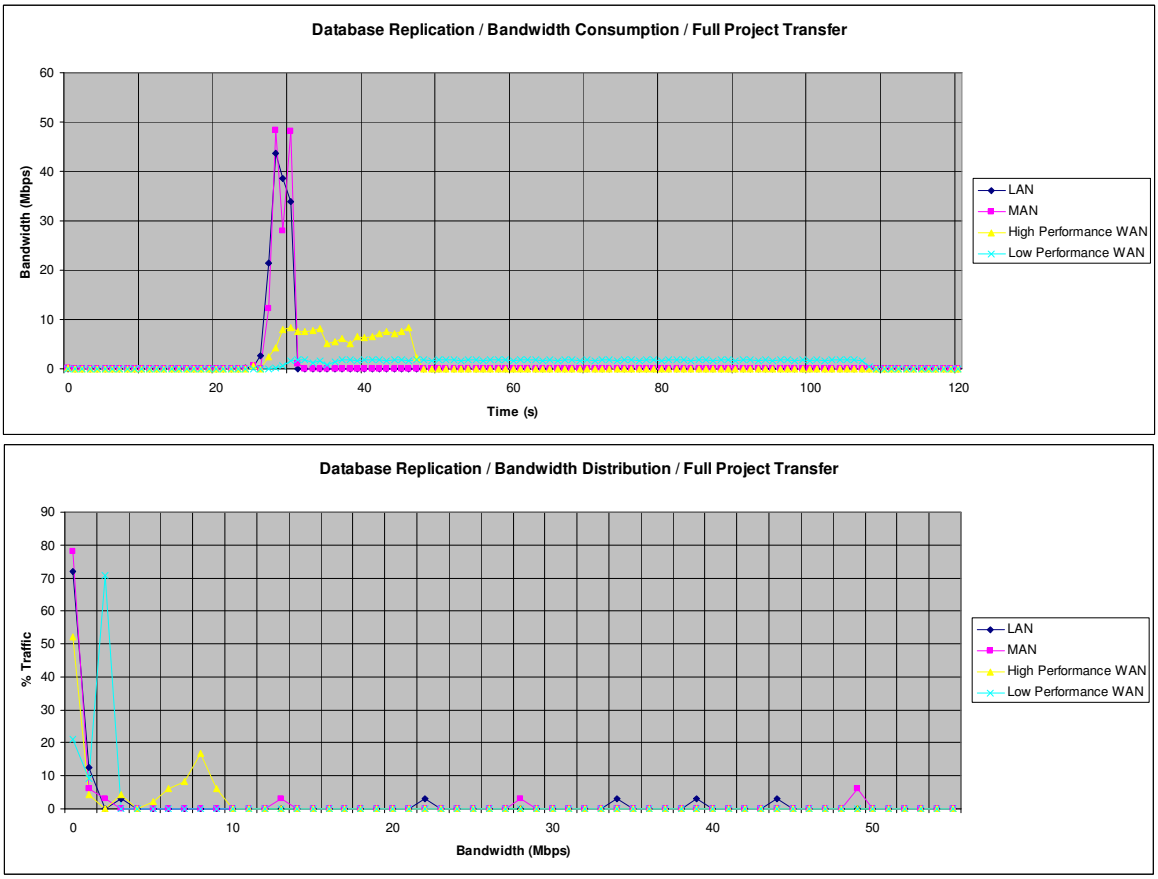
18.5.2. Simulation: Server Only Model

For each class of network the minimum and the complete database replication were performed. In each case the total time to perform the replication was recorded. The simulation was performed on a simplified version of the model without the workstations. This was done to improve the performance of the simulation itself.



Database replication time for each class of network

The bandwidth consumption pattern over the WAN link between the hub and the spoke was recorded. The bandwidth consumption pattern indicates the extent to which bandwidth plays a role in the performance of this architecture.



Bandwidth consumption for the different classes of network using the database replication architecture

Network class	Bandwidth Available (Mbps)	Peak Bandwidth (Mbps)	% Bandwidth Consumed
LAN	1000	43.6	4.36
MAN	155	48.3	31.2
High performance WAN	20	8.46	42.3
Low performance WAN	2	1.80	90.0

Bandwidth consumption for the different classes of network using the database replication architecture

18.6. Discussion of Results

18.6.1. Verification and Validation

Verification and validation took place according to the recommendations in [Heidemann 01] as follows:

Recommendation	Discussion of Results
Assess sensitivity to parameter variation.	Varying the RTT gave the expected result (see section 18.2).
Check that the results are logical and reproducible.	In all cases, it was checked that the time to perform an operation was longer than the RTT over the network connection. Experiments were occasionally repeated in order to check for reproducibility.
Use graphics, and be able to observe and inspect the simulation as it progresses.	It was possible to observe the simulation as it proceeded and to some extent to replay it using the NCTUns package. This was useful when creating the model in order to verify it.
Where the test machine imposes limitations, make sure that the simulation is still accurate.	The "top" utility was used to monitor the load and track the various processes as the simulation proceeded. Load remained under 4.0. When load was high, the experiment was stopped. The experiment was then repeated at a slower speed. This led to more accurate results, with the trade off that the simulation took longer to complete. Various experiments were performed in order to determine the optimum speed at which to run the simulator.

Verification results

Recommendation	Discussion of Results
For well-specified protocols, direct comparison with a real world system can be used for the purposes of validation.	The main form of validation was to compare the results for the LAN simulation with the results from the emulation environment. Similar results should be expected in both environments. In all cases the difference between the two results was within 10%. This corresponds to the degree of experimental error.

Validation results

18.6.2. Network Characteristics Limiting Application Performance

Two network characteristics were examined – bandwidth and latency. These variables affected the architectures in different ways as shown in the table below.

Architecture	Effect of Latency	Effect of Bandwidth
Thick Client	Higher latency leads to higher processing response time.	Bandwidth does not play a significant role in network performance for the network classes studied.
Thin Client	Higher latency leads to higher acknowledgement response time.	
Database Replication	Minimal.	Higher bandwidth leads to faster database replication. For the test data set processing time was as a significant component of the total time to complete the replication.

Limiting network characteristics for the three architectural patterns

18.6.3. Comparison of Thick Client and Thin Client

The bandwidth consumption was higher for the thin client architecture than the thick client architecture. This can be explained by the fact that every operation requires network traffic under the thin client architecture. For the thick client architecture, most operations were self-contained within the local workstation.

18.6.4. Comparison of Results to Application Demands

When the results are compared to the application demands identified in section 7, the following picture emerges:

Architecture	Application Demand	Result		Comments
Thick client	Processing response time < 11s	**Network Class**	**Result**	In all cases, including emulation, the application failed to meet the performance criteria. Response time increased linearly with increasing network latency.
		LAN	20.6s	
		MAN	20.6s	
		High Perf WAN	26.6s	
		Low Perf WAN	44.8s	
Thin client	Acknowledgement response time < 150ms	**Network Class**	**Result**	All network classes gave unacceptable performance, but in the case of the LAN and MAN the threshold was crossed by less than the experimental error. They should be considered to be acceptable. The low performance WAN is clearly unacceptable and the high performance WAN is marginal.
		LAN	152ms	
		MAN	159ms	
		High Perf WAN	203ms	
		Low Perf WAN	807ms	
Database replication	Complete replication inside a one hour window.	**Network Class**	**Result**	Replication was very fast for the test data, even on the low performance WAN. As such, this architecture should be considered acceptable for all classes of network.
		LAN	30.4s	
		MAN	30.8s	
		High Perf WAN	47.5s	
		Low Perf WAN	108s	

Comparison of the results with application demands

19. Extension Problem: Use of Communication and Collaboration Tooling

A global team may require extensive communication and collaboration tooling. This part of the thesis attempts to collect data that can be used to determine which classes of tools are appropriate. The experiments were carried out using tools and techniques described in section 12.

The aim of this series of experiments was to establish only the likely bandwidth consumption of such tools. This can be used to examine the effect on the size of team that can be supported for a given class of network.

19.1. Emulation Environment Tests

In this case the experiments were carried out using publicly available tools offered by service providers on the Internet. As such, the network latency and bandwidth were not controlled.

19.2. Collaboration Tools

19.2.1. Emulation and Data Capture

In order to establish an example of bandwidth requirements for web-based social networking tools, a simple experiment was carried out using a variety of public social networking systems.

Scenario	Description	Elapsed Time (s)	Traffic Volume (KB)
Approximately one minute usage of web-based social networking tools.	Use a web browser to interact with systems providing wikis, social bookmarking and RSS feeds.	57.2s	321.75

Results of traffic capture for social networking tools

The bandwidth measured was 45.0 kbps.

19.2.2. Simulation

Due to the fact that the aim of the experiment was only to establish typical bandwidth consumption of the tools, no simulation was performed.

19.3. Communication Tools

As noted in section 8.3, there is extensive data from other studies available on traffic patterns for various real-time communication tools. No data was found for collaborative virtual environments. Therefore an experiment was performed to establish data for this class of tool.

19.3.1. Emulation and Data Capture

In order to establish an example of bandwidth consumption for a collaborative virtual environment, a simple experiment was carried out using the public Second Life system.

Using the Second Life collaborative virtual environment

Scenario	Description	Elapsed Time (s)	Traffic Volume (KB)
Connection to Second Life	Use an avatar in Second Life. Explore the environment and interact with other avatars.	75.7	3947

Results of traffic capture for a collaborative virtual environment

This experiment suggests that bandwidth consumption for collaborative virtual environments is comparable with that required for videoconferencing. The actual bandwidth measured was 417kbps.

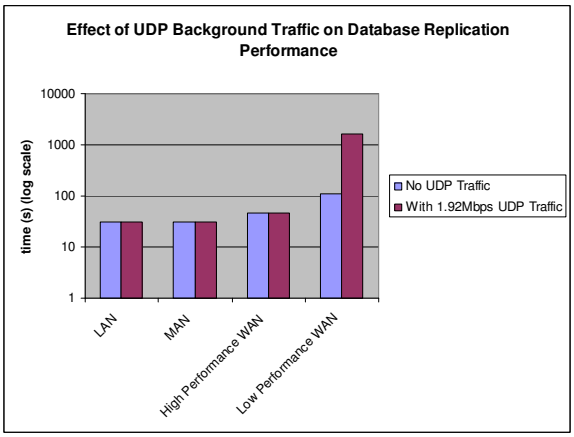
The Wireshark protocol analyser revealed that 95.5% of packets were UDP. The remaining packets were TCP. Little is known further about the Second Life protocol as it is proprietary and encrypted.

19.3.2. Simulation: Complete Infrastructure Model

A simulation was performed in order to examine the effect of combining real-time communications tools with the software engineering infrastructure from the case study. Real-time communication tools were modelled as generators of UDP traffic. The scenario modelled was one whereby 30 users each performed peer-to-peer communication using a UDP stream of 64kbps (equivalent to an audio stream at POTS quality).

J-NAP was not used to model the communication tool traffic. The real-time tools were modelled as using the standard features of the NCTUns package.

The simulation involved performing database replication while simultaneously using the real-time communication tools. This scenario was selected due to the sensitivity of this architecture to bandwidth noted in section 18.6.2.



Effect of introducing extensive UDP traffic into a network used for database replication

Performance degradation in excess of one order of magnitude was observed for the low performance WAN. For the other classes of network the performance was not significantly changed.

19.4. Discussion of Results

The results indicate that collaborative virtual environments are network intensive applications, comparable to videoconferencing in terms of bandwidth and type of traffic. Unlike videoconferencing the main traffic flow is between a central server and the clients, not peer-to-peer.

Windows for collaboration can be short in a global team due to time zone differences. It might be necessary to replicate the version control database (so that the team that is just starting work can get the latest changes) and perform real-time communication in parallel. Both of these processes consume significant bandwidth.

The results suggest that database replication would complete inside a one hour time window even while the team was engaged in significant real-time communication.

The experiments demonstrated that the simulation technique can be applied to a mix of TCP and UDP traffic.

Part IV. Conclusions

20. Introduction

Part IV of the thesis is concerned with drawing conclusions about the problems defined in Part I. These conclusions are based on data from the simulations carried out in Part III in combination with a literature study and experience gained during the case study.

Recommendations are made for each of the four areas identified in section 1:

- Business processes within global software engineering teams and their relationship to the tools employed.
- Selection of the optimum configuration for tools for global software engineering teams.
- The use of communication and collaboration tools by global software engineering teams.
- The efficacy of network simulation as a means to create a performance model for infrastructure supporting global software engineering teams.

21. Business Processes and Tooling for Global Software Engineering Teams

21.1. Business Processes

Organisations have been running software engineering projects for decades. There are many well established software engineering methodologies, of which IBM RUP is but one example. Almost all of these methodologies carry the assumption that the team is co-located, or were originally developed with that assumption.

Some software engineering methodologies have been adapted to a global setting (e.g. [CollabNet 06] [Kircher 01]). When organising a global software engineering team, it would be wise to consider these adapted methodologies, and to consider the differences between a global and a co-located team.

The most obvious area of difference is in how the team communicates and collaborates. All global teams face challenges in the area of communication and collaboration [Cleland 06]. This has led to extensive research and development into tools for global teams. This thesis concludes that use of such tools in global software engineering teams is likely to increase the cohesiveness, or "gel" [DeMarco 99] of the team. Ultimately this is likely to lead to higher productivity.

A more subtle issue is the reduction in trust caused by offshoring and outsourcing. This can be seen in the requirements for the case study. It was clear that the corporation had concerns about loss of control resulting in requirements for centralisation and strict security measures.

21.2. Tooling

The changes to business processes and the characteristics of WANs lead on to changes required in tooling. Key issues to be considered when selecting software engineering tools are described below. Vendors of software engineering tools should also consider these issues when developing next generation products. The Jazz project [Hupfer 04] is an example of an effort to create such a product.

This thesis is based on a case study using the current IBM Rational suite. Other products were not evaluated. However, the points raised are broadly applicable when adapting software engineering tools to global teams.

21.2.1. Effect of High Network Latency

The software engineering tools in the case study did not all have a technical architecture that is suitable for use with high network latency. Those that were provided (such as CCRC) tended to lack features.

21.2.2. Integration with Communication and Collaboration Tools

The tools used in the case study provide facilities that are necessary for maintaining the formal agreements between the team. The tools do not provide an environment for the informal and social interactions that a global team requires.

Tools that promote social networking exist and are well established. What appears to be lacking is integration between such tools and more traditional software engineering tools. An example of such a feature would be a version control tool that allowed the user to see the online presence or calendar of the author of the last change. More examples can be found in [Hupfer 04].

21.2.3. Reduction in Trust

The reduction in trust leads to a range of issues that were not well addressed by the tools in the case study. Identity management of users is complex due to the involvement of more than one organisation. Integration with systems supporting federated identity management may provide a solution. There is a perception of increased risk of malicious code insertion. This could be countered through integration between version control systems and virus scanning tools.

A more subtle point is that intellectual property issues arise. This may make certain architectures preferable. In the future techniques such as DRM or trusted computing could be used in an attempt to keep control of intellectual property.

22. The Optimisation Problem

22.1. Recommended Technical Architecture

Various factors will determine the suitability of a particular architecture for a given situation. The main factor examined in this thesis was the impact on user experience caused by network latency and bandwidth. The results reveal that network latency is the key driver when selecting the appropriate architectural pattern. Latency can rise rapidly with the use of overlay networks, traffic prioritisation and advanced firewalls [Tolia 06], so high latency could exist in networks over any geographical distance.

The results were as follows (see section 18.6.4):

- The thick client architecture gave poor processing response time on all four classes of network. Even with very low network latency the application failed to meet application demands. As latency increases, performance worsens rapidly.

- The thin client architecture gave marginally acceptable acknowledgement response time for high performance WANs and faster networks. For the low performance WAN, performance was unacceptable. It should be noted that the processing response time will still be poor with this architecture, but will be as fast as possible.

- The database replication architecture gave excellent performance for all four classes of network. Again processing response time will still be poor with this architecture, but will be as fast as possible.

There were four key requirements for the system (see sections 3.6 and 4.2):

- The requirement that there must be a single repository for all artefacts of the engineering process was met by all architectures (replication was considered acceptable).

- A preference for thin client architecture was given based on corporate standards and policies. This could be met for some classes of network but with a poor client-server ratio, resulting in high infrastructure cost.

- A preference for keeping source code and other artefacts within the corporate network was given. This could be met by any architecture for deployment on LAN or corporate MAN. For connectivity to outsourced parties this was only met by the thin client architecture.

- A requirement that there must be encryption and two factor authentication for direct end-user remote connectivity was given. This could be met by any architecture for deployment on LAN or corporate MAN. For connectivity to outsourced parties this was met by the thin client and database replication architectural patterns.

The architectures were also to be assessed in terms of infrastructure cost, availability, scalability and security (see sections 4.1 and 4.2):

- It is likely that the thick client architecture will have the lowest architecture cost.

- Availability is poorest with the thin client architecture. Thick client architecture has intermediate availability. Database replication has the highest availability.

- All three architectures could be described as scalable.

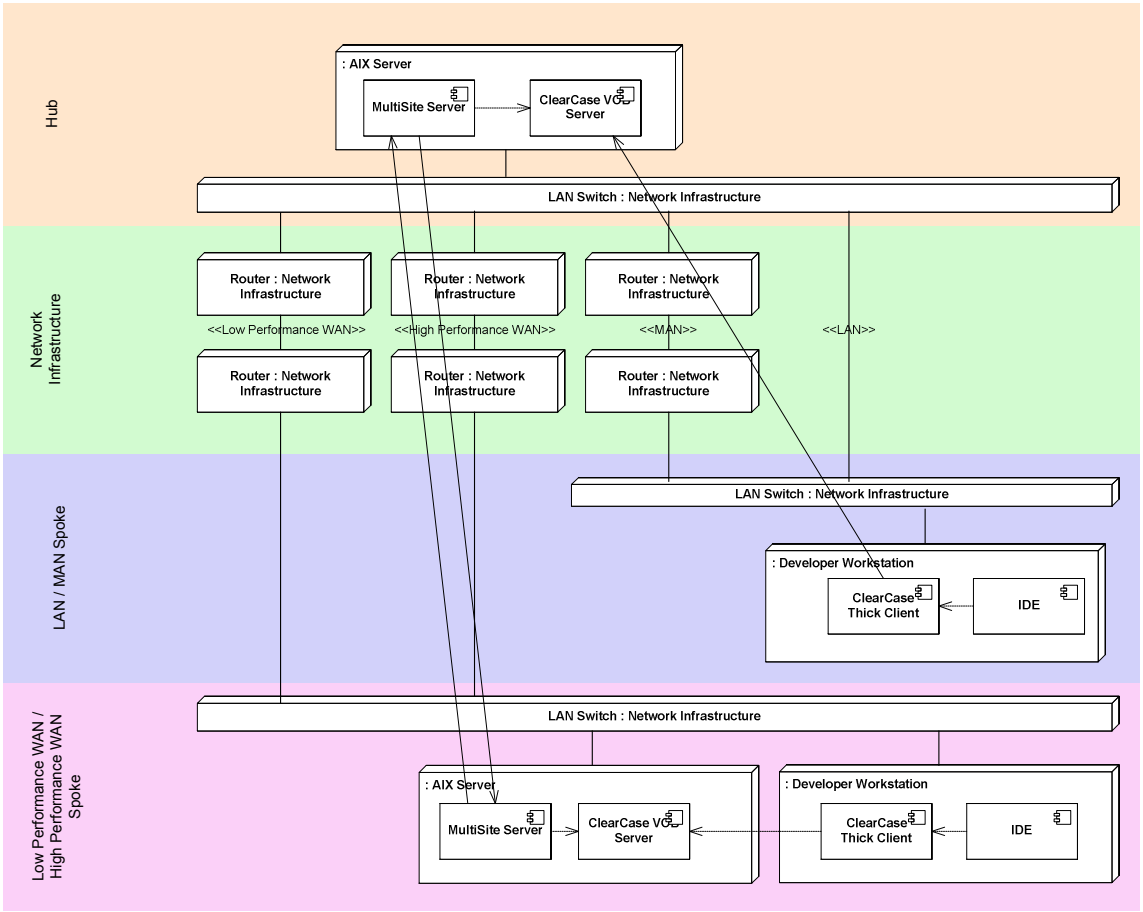
- Security was dealt with above as part of the key requirements.

Based on the above, the following table of suitable architectural patterns can be constructed:

Network class	Suitable architectural patterns	Comments
LAN	All	Database replication and thin client would add (largely) unnecessary cost and complexity. The user can simply work using the thick client and connect directly to the hub.
MAN	All	On the corporate MAN situation is the same as for LAN.
High performance WAN	Thin client Database replication	Thick client provides too much additional response time to an already poorly performing application. Database replication recommended because thin client, while acceptable, will give poor availability and poorer response times.
Low performance WAN	Database replication	Response times are unacceptable for both the thick client and the thin client architectures.

Suitability of architectural patterns for the various network classes in the optimisation problem

Given that there is only a small class of networks for which the thin client architecture could be justified (WANs where latency is lower than 100ms), it seems wise to keep the complete system architecture simple and propose only the thick client over the LAN and MAN network classes. Database replication should be used for all other network classes.



UML deployment diagram showing recommended technical architecture for the optimisation problem

22.2. Limits on Team Size

22.2.1. Thick and Thin Client Architecture

Limits on team size have been computed by linear extrapolation of bandwidth consumption. These limits should be taken only as an order of magnitude indication.

Architectural Pattern	Network Classes	Approximate Limit on Team Size based on Bandwidth Consumption (users)
Thick client	LAN / MAN	10000
	High performance WAN	1000
	Low performance WAN	Unsuitable due to response time.
Thin client	LAN / MAN	5000
	High performance WAN	500
	Low performance WAN	Unsuitable due to response time.

Limits on team size for the thin and thick client architectural patterns

It is possible to perform some validation of these conclusions by considering what is regarded as a "normal" bandwidth per user for such services. Sources suggest values similar to those given here (e.g. [Posey 05]).

22.2.2. Database Replication Architecture

In the case of database replication the limits are based on the size of the project as opposed to the number of users. Database replication must complete in one hour due to time zone overlaps. Comparing this to the elapsed time of replication for the various classes of network provides a value for the maximum size of projects that can be supported.

Network class	Replication Elapsed Time (s)	Maximum size project (factor compared to the test data)
LAN	32	113
MAN	32	113
High performance WAN	48	75
Low performance WAN	110	33

Replication time and implications for sizing using the database replication architecture

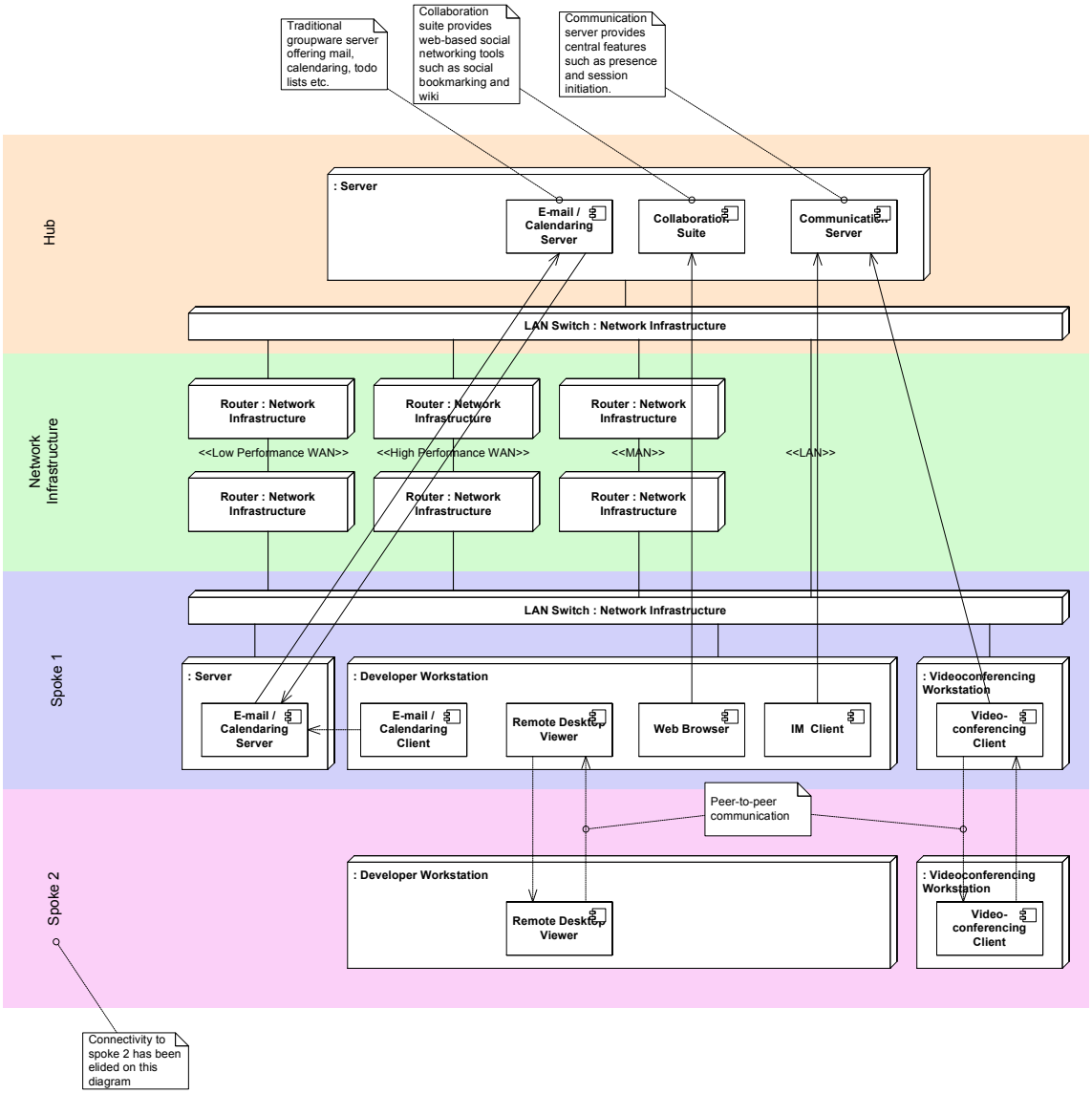
The results suggest that even with a project that is more than one order of magnitude larger than the test data, replication will be possible inside the one hour window, even on the low performance WAN. These figures are based on the worst case scenario whereby the entire version control archive must be transferred between the sites. In short, the data suggests that replication is very unlikely to prove problematic.

23. The Extension Problem

23.1. Recommended Technical Architecture

Various studies conclude that communication and collaboration tooling may counterbalance some of the negative aspects of global teams – lack of presence, reduction in trust, lack of face-to-face communication, reduced informal communication and time zone differences [Carmel 99] [Carmel 05] [Cleland 06] [Treinen 06] [Hupfer 04]. A general theme seems to be the use of multiple modes of communication and collaboration [Braithwaite 05].

Based on this conclusion, the possibility of extending the project from the case study with this type of tooling was considered. The aim was to determine appropriate types of tooling to be added, and the likely impact on network resource consumption. Conclusions are drawn largely from a literature study. Experiments were performed to provide additional data where necessary.



UML deployment diagram showing recommended technical architecture for the extension problem

Component	Location	Comments
Collaboration suite	Hub	Web-based social networking tools.
Communication server	Hub	Provides presence and session initiation.
Remote desktop viewer	Developer workstation	Allows users to view each other's desktops.
IM client	Developer workstation	
E-mail infrastructure	Hub, spokes, developer workstation	Unchanged from the case study infrastructure.
Videoconferencing workstations	Spokes	Specialised hardware, e.g. EyeCatcher [Exovision 07]

Components of the recommended architecture for the extension problem

23.1.1. E-mail and Conventional Telephony

The case study included e-mail and conventional telephony, albeit tacitly. These are two very useful media and it is not proposed to replace them.

It would be technically possible to replace conventional telephony with VoIP, but this is a purely economic trade off. Using conventional telephony reduces load on the corporate IP network and provides independent infrastructure that will increase availability. For this reason it is not proposed to use VoIP in the recommended architecture.

23.1.2. Communication Tooling

The aim of introducing communication tooling is to provide a surrogate for the following:

- Face-to-face communication (an important factor in building trust)
- Informal communication
- Presence
- Ability to view other user's screens

In section 5.4, a table of various communication media was presented. Two types of tooling are strong candidates as surrogates for face-to-face communication based on the degree of psychological involvement: videoconferencing and collaborative virtual environments.

Informal communication can be fostered through use of IM and various collaboration technologies.

Presence can be provided through the use of IM.

Ability to view other user's screens can be provided through the use of remote desktop viewing.

23.1.3. Videoconferencing

Videoconferencing is a mature technology, but one that still seems unable to make real inroads [Bekkering 06] [Lurey 01]. This lack of breakthrough success for videoconferencing is probably caused by two factors: bandwidth requirements and lack of direct eye contact.

High quality video requires bandwidth of greater than 500 kbps. Lower quality video is of little communicative benefit [Whittaker 95].

Lack of direct eye contact is one reason that videoconferencing is not more popular [Bekkering 06]. Some systems such as the Exovision EyeCatcher overcome this through advanced optics [Exovision 07]. This makes the equipment cumbersome and expensive, but perhaps still cost effective and environmentally responsible

compared to air travel.

The recommended architecture includes a small number of high quality videoconferencing setups, provided in meeting rooms. Three setups in a team of 100 should be sufficient. Meeting rooms in large corporations often feature multimedia equipment, so this is not particularly unusual.

The videoconferencing facilities should be used when an important meeting needs to take place in order to ensure more effective communication. If possible, the equipment used should allow direct eye contact to be made. Minimum bandwidth should be 500kbps.

23.1.4. Collaborative Virtual Environments

The only collaborative virtual environment that was examined was the public service known as Second Life. This service is being advocated as a serious collaboration tool in some quarters [Bulkeley 07].

The Second Life service shows strong potential but for the case study it would be immediately rejected on security grounds because it is hosted externally. It is the view of the author that a similar system made less videogame-like, operational without the need for a service provider and with support for voice communication would be a useful communication and collaboration tool.

An experiment was performed to gain some insight into the network resources consumed by such a system. This experiment revealed a star architecture. Bandwidth consumption was comparable to videoconferencing. The 3D rendering involved in such a system makes significant demands on the end-user's workstation. These points count against use of such a system.

The conclusion is to reject collaborative virtual environments for the time being, but to review new products that may emerge in this area.

23.1.5. Instant Messaging

Instant messaging is proposed to provide a surrogate for informal communication and presence. Providing that IM is used solely for "chat", its impact on network resources will be very low indeed[27].

23.1.6. Remote Desktop Viewing

Remote desktop viewing is a particularly useful tool for software engineering [Braithwaite 05] that also scored highly in terms of psychological involvement (see section 5.4). During an audio conversation it can be used to demonstrate a particular user interface problem, or to view a model or source code. Network resource consumption will be very similar to the thin client architectural pattern investigated as part of the optimisation problem.

23.1.7. Collaboration Tooling

The aim of introducing collaboration tooling is to provide a surrogate for the following:

- Access to information from team members who are not available due to time zone differences
- Informal collaboration

In terms of architecture, collaboration tooling could be offered using the architectural patterns examined in the optimisation problem. However, collaboration tooling for social networking is typically offered using web technology. This architecture was not evaluated for the optimisation problem because it failed to meet the functional requirements.

It has already been roundly demonstrated that web technology is suitable for deployment over high latency WANs. The WWW itself is sufficient evidence for this. Therefore, it is reasonable to deduce that web-based collaboration tooling would be suitable for extending the case study. When performing evaluation of specific products, it would be possible to use the techniques demonstrated in this thesis to model network performance and resource consumption.

Based on recommendations from the literature study, a suite should be deployed that provides a user directory

[27] This is an assumption that was not demonstrated experimentally.

[Bulkeley 07], wiki [Kroll 05] [Bulkeley 07] [Kircher 01], shared calendar [Kroll 05] and social bookmarking [Millen 05]. In order to provide an estimate for sizing, a simple experiment was performed into typical bandwidth consumption for web-based collaboration tooling.

23.1.8. Business Processes

Users will need to be trained to make use of the communication and collaboration tooling. The engineering processes should be designed so that it is not possible to avoid using the tools. Despite the provision of the tools, key staff should still take part in occasional face-to-face meetings, particularly during project initiation.

23.2. Limits on Team Size

The model for activities of software engineers proposed usage patterns for communication and collaboration tools. Based on this, a model for total bandwidth consumption can be constructed.

Activity	Typical bandwidth per user (kbps)	Source
IM	Negligible[28]	Theoretical
Remote takeover / screen viewing	20	Section 18.4.2
Audioconferencing using conventional telephony	0	
Audioconferencing using VoIP	50	Section 8.3.1
Videoconferencing	500	Section 8.3.1
Using social networking tools	45	Section 19.3.1
e-mail	1.0[29]	Computed based on data from [Ducheneaut 01] and [Lyman 03]

Based on such a model, the example 100 member team would require 1.6Mbps of bandwidth for the use of communication and collaboration tools. This is dominated by the use of videoconferencing.

Given the peer-to-peer nature of most real-time communication tooling, the size limit should be based on the network class between the peers, as opposed to the network class between the hub and spoke. Limits are as follows:

Network class (peer-to-peer)	Approximate Limit on Team Size based on Bandwidth Consumption (users)	Comments
LAN / MAN	30000	If the peers are connected by LAN, then videoconferencing is unnecessary. This result is therefore largely irrelevant.
MAN	5000	
High performance WAN	1000	
Low performance WAN	100	

The actual maximum team size will be determined by a combination of the team size that can be supported for conventional software engineering tooling and the team size that can be supported for communication and collaboration tooling.

[28] Providing that IM is used purely to "chat". If it is not, then it is reasonable to assume that it is used as a replacement for e-mail, thus having no net effect on the total bandwidth consumed.
[29] In the case of e-mail, the percentage of time spent is not particularly relevant. The total volume of messages determines the bandwidth requirement.

An experiment demonstrating such a combination was given in section 19.3.2. In this experiment, it was demonstrated that the database replication architecture could be safely combined with communication tool traffic on a low performance WAN.

24. The Efficacy of Network Simulation during Network Infrastructure Design

24.1. The Challenge of Complexity

IP networks are examples of complex systems. Complex systems exhibit "emergent properties" that are difficult to ascertain a priori [Von Bertalanffy 68]. Predicting real world network performance and behaviour based on controlled experiments is at best problematic [Watson 97] and is increasingly difficult as complexity rises [Nicol 05]. Nevertheless, project managers and customers demand early sizing of network infrastructure; not surprising considering the costs and lead times involved in arranging MAN or WAN connectivity. This leaves the engineer in a quandary.

A method based on simulation such as that used in this thesis could offer relief. However, as can be seen from the limitations mentioned in section 12.4, the accuracy of such a method is questionable. Most real world systems will be affected by usage patterns and so by human behaviour. This has yet to be accurately modelled!

24.2. Comparison to Other Approaches

There are advantages to simulation when compared to other means of sizing. Compared to emulation it is rapid to set up and can be run on limited hardware, typically on the engineer's own workstation. Compared to static mathematical models (usually built with spreadsheets) it has the potential to be more accurate, model non-linear effects such as jitter and to allow a greater range of scenarios to be explored. When combined with a GUI, a simulator can also act as a training and communication tool within a project team.

24.3. The NCTUns Simulator

This thesis made use of the NCTUns simulator[30]. This is a relatively new package that is not in widespread use. It has its roots in the university research lab.

With hindsight, the selection of NCTUns instead of ns-2 could be questioned. The modal, somewhat amateurish GUI was not as useful as had been hoped. The package is complex to install and use, with a steep learning curve. Carefully performing control experiments is essential. It was necessary to carefully set simulation parameters and use workarounds in order to generate reliable results. The package is not yet ready for mainstream use.

The innovative architecture of the NCTUns package was of great benefit. It was straightforward to deploy a new application, J-NAP, on the simulated network. In this respect the package shows great potential. However, the fact that is was necessary to develop J-NAP reveals a drawback with the overall approach. Integration between simulation, emulation and traffic analysis packages is not yet mature.

The results suggest that a simulator such as NCTUns can be quite accurate in terms of comparison with the same traffic pattern on a real network. The error was around 10%, based on comparisons to the emulation environment.

[30] This thesis made use of NCTUns 3.0. As this thesis approached its end, NCTUns 4.0 was released.

Appendix A. References

[Baker 03] H. H. Baker et al., "Computation and Performance Issues In Coliseum, An Immersive Videoconferencing System", *MM'03, November 2-8, 2003, Berkeley, California, USA, ACM 1-58113-722-2/03/0011*

[Baker 05] H. H. Baker et al., "Understanding Performance in Coliseum", *ACM Transactions on Multimedia Computing, Communications and Applications, Vol. 1, No. 2, May 2005.*

[Baldi 99] M. Baldi, "End-to-End Delay Analysis of Videoconferencing over Packet-Switched Networks", *IEEE/ACM Transactions on Networking, Vol. 8, No. 4, August 2000*

[BBC 06] BBC News, "Asia Communications Hit by Quake", *http://news.bbc.co.uk/2/hi/asia-pacific/6211451.stm, December 2006*

[Bekkering 06] E. Bekkering, J.P. Shim, "i2i Trust in Videoconferencing", *Communications of the ACM, July 2006/Vol. 49, No.7*

[Benjamin 04] B. S. Benjamin, "System Engineering Management", *ISBN 0-471-29176-5, John Wiley and Sons, 2004*

[Blake 98] S. Blake et al., "An Architecture for Differentiated Services", *IETF RFC 2475, December 1998, http://www.ietf.org/rfc/rfc2475.txt*

[Bouch 00] A. Bouch, A. Kuchinski, N. Bhatti, "Quality is in the Eye of the Beholder: Meeting Users' Requirements for Internet Quality of Service", *ACM CHI Letters, volume 2, issue 1, 2000*

[Bouch 99] A. Bouch, M. Sasse, "It ain't what you charge, it's the way that you do it: A User Perspective of Network QoS and Pricing", *Proceedings of IM 1999.*

[Braithwaite 05] K. Braithwaite, T. Joyce, "XP Expanded: Distributed Extreme Programming" / "Extreme Programming and Agile Processes in Software Engineering", *ISBN 978-3-540-26277-0, Springer Berlin / Heidelberg 2005*

[Breslau 00] L. Breslau et al, "Advances in Network Simulation", *IEEE Computer May 2000, pp. 59-67*

[Bulkeley 07] W. M. Bulkeley "Playing Well With Others", *Wall Street Journal, June 18th, 2007*

[Calyam 04] P. Calyam et al., "H.323 Beacon: An H.323 Application Related End-to-End Performance Troubleshooting Tool", *SIGCOMM'04Workshops, Aug. 30+Sept. 3, 2004, Portland, Oregon, USA. ACM 1-58113-942-X/04/0008*

[Carmel 05] E. Carmel, P. Tjia, "Offshoring Information Technology", *ISBN 978-0-521-84355-3, Cambridge University Press 2005*

[Carmel 99] E. Carmel, "Global Software Teams", *ISBN 0-13-924218-X, Prentice Hall 1999*

[Chang 99] X. Chang, "Network Simulations with Opnet", *ACM Proceedings of the 1999 Winter Simulation Conference, pp. 307-314*

[Chen 02] M.Chen, "Achieving Effective Floor Control with a Low-Bandwidth Gesture-Sensitive Videoconferencing System", *Multimedia'02, December 1-6, 2002, Juan-les-Pins, France. ACM 1-58113-620-X/02/0012*

[Chen 06] K.Chen et al,, "Quantifying Skype User Satisfaction", *ACM Proceedings of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications*

[Cheshire 96] S. Cheshire, "Latency and the Quest for Interactivity", *Commissioned Paper, November 1996, http://www.stuartcheshire.org/papers/LatencyQuest.ps*

[Cleland 06] D. I. Cleland, R. Gareis, "Global Project Management Handbook: Planning, Organizing, and Controlling International Projects, Second Edition", *ISBN 9780071460453, McGraw-Hill 2006*

[CollabNet 06] CollabNet, "Distributed Rational Unified Process in an On-Demand Geographically Distributed Development Environment", *Commercial White Paper 2006, http://www.collab.net/news/library/index.html*

[Daft 86] R.L. Daft, R.H. Lengel, "Organizational Information Requirements, Media Richness and Structural Design", *Management Science 32, 5 (1986), p 554–571.*

[DeMarco 99] T. DeMarco, T. Lister, "Peopleware: Productive Projects and Teams", *ISBN 0932633439, Dorset House Publishers, 1999*

[Ducheneaut 01] N. Ducheneaut, V. Bellotti, "E-mail as Habitat: An Exploration of Embedded Personal Information Management", *ACM Interactions, Volume 8 , Issue 5, Sept./Oct. 2001, ISSN:1072-5520*

[Elrod 92] S. Elrod et al, "Liveboard: A Large Interactive Display supporting Group Meetings, Presentations, and Remote Collaboration", *ACM Proceedings of the SIGCHI conference on Human factors in computing systems, 1992*

[Exovision 07] Exo'Vision, "EyeCatcher 3.0 Technical Specification", *Commercial White Paper, 2007* http://www.exovision.nl/download/EyeCatcher-leaflet.pdf

[Fall 05] K. Fall, S. McCanne, "You Don't Know Jack About Network Performance", *ACM Queue, May 2005*

[Frohlich 95] D. Frohlich, "Requirements for Interpersonal Information Management*", P.J. Thomas (Ed.) Personal information systems: Business applications. Stanley Thornes in association with Unicom Seminars. 1995 Available from Hewlett-Packard:* http://www.hpl.hp.com/techreports/94/HPL-94-98.pdf

[Fowler 03] M. Fowler et al, "Patterns of Enterprise Application Architecture", *pp. 7-9, ISBN 0-321-12742-0, Pearson, 2003*

[Gibbs 07] R. D. Gibbs, "Project Management with the IBM Rational Unified Process: Lessons From The Trenches", *ISBN 0321336399, IBM Press, 2007*

[Heidemann 01] J. Heidemann, K. Mills, "Expanding Confidence in Network Simulations", *IEEE Network September / October 2001, pp. 58-63*

[Hupfer 04] S. Hupfer, L-T Cheng, S. Ross, J. Patterson, "Introducing Collaboration into an Application Development Environment", *CSCW '04, November 6–10, 2004, Chicago, Illinois, USA. ACM 1-58113-810-5/04/0011*

[IBM 07] IBM, "Feature differences between CCRC, CCWeb and Native ClearCase" http://www-1.ibm.com/support/docview.wss?uid=swg21233090

[Jansen 06] S. Jansen, A. McGregor, "Performance, Validation and Testing with the Network Simulation Cradle", *Proceedings of the 14th IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS) 2006.*

[Jansen 07] S. Jansen, "Blog on Network Simulation Cradles", *University of Waikato Network Research Group website:* http://www.wand.net.nz/~stj2/blog/

[Kircher 01] M. Kircher, P. Jain, A. Corsaro, D. Levine, "Distributed eXtreme Programming", *Second International Conference on eXtreme Programming and Agile Processes in Software Engineering (XP2001), Cagliari, Sardinia, Italy, 2001*

[Kroll 05] P. Kroll, W. Royce, "Key Principles for Business-driven Development", *The Rational Edge, IBM, October 2005*

[Lyman 03] P. Lyman, H, Varian, "How Much Information 2003?", *School of Information Management and Systems at the University of California at Berkeley,* http://www2.sims.berkeley.edu/research/projects/how-much-info-2003/

[Lurey 01] J. S. Lurey, M. S. Raisinghani, "An Emprical Study of Best Practices in Virtual Teams", *Information & Management 38 (2001) 523–544*

[Microsoft 07] Microsoft, "SoftGrid Overview", *Commercial White Paper 2007,* http://www.softricity.com/news/ecollateral/public/SoftGrid-Overview.pdf

[Millen 05] D. Millen, J. Feinberg, B.Kerr, "Social Bookmarking in the Enterprise", *ACM Queue, November 2005*

[Myer 06] B. Meyer, "The Unspoken Revolution in Software Engineering", *IEEE Computer, January 2006*

[Myers 85] B. A. Myers, "The Importance of Percent-Done Progress Indicators for Computer-Human Interfaces", *ACM Proceedings of CHI 1985*

[Nicol 05] D. M. Nicol, M. Liljenstam, J. Liu, "Advanced Concepts In Large-Scale Network Simulation", *Proceedings of the IEEE 2005 Winter Simulation Conference, 2005*

[Pappalarado 02] D. Pappalarado, "Cable & Wireless ups Latency SLAs in North America, Europe", *Network World, June 2002*

[Patterson 06] Laurie J. Patterson, "The Technology Underlying Podcasts", *IEEE Computer, October 2005*

[Posey 05] B. Posey, "Juggling Terminal Server Resources", *MSTerminalServices.org Website, November 2005,* http://www.msterminalservices.org/articles/Juggling-Terminal-Service-Resources.html

[Rosen 01] E. Rosen, R. Callon, "Multiprotocol Label Switching Architecture", *IETF RFC 3031, January 2001,* http://www.ietf.org/rfc/rfc3031.txt

[Rosenberg 02] J. Rosenberg et al., "SIP: Session Initiation Protocol", *IETF RFC 3261, June 2002,* http://www.ietf.org/rfc/rfc3261.txt

[Schiff 06] D. Schiff, "Global Teams Rock Around the Clock", *EE Times Online, August 2006*

[Sharafeddine 03] S. Sharafeddine, A. Riedl, J. Glasmann, J. Totzke, "On Traffic Characteristics and Bandwidth Requirements of Voice over IP Applications", *Proceedings of the Eighth IEEE International Symposium on Computers and Communication (ISCC'03)*

[Skinnemoen 05] H. Skinnemoen et al., "VoIP Over DVB-RCS With QoS and Bandwidth on Demand", IEEE Wireless Communications, October 2005

[Trienen 06] J. J. Treinen, S. L. Miller-Frost, “Following the Sun: Case Studies in Global Software Development”, *IBM Systems Journal, Volume 45, No 4, 2006*

[Trivedi 06] G. Trivedi, “Network and Internet Services and Service Providers in India, 2005”, Gartner *Research ID Number G00137257, January 2006*

[Tolia 06] N.Tolia. D.G. Andersen, M. Satyanarayanan, “Quantifying Interactive User Experience on Thin Clients”, *IEEE Computer, March 2006 pp. 46-52*

[Vance 07] A. Vance, "Effectively Complying with Sarbanes-Oxley in Dynamic Business Environments: A Knowledge Traceability Approach," *40th Annual Hawaii International Conference on System Sciences (HICSS'07), 2007*

[VMware 05] VMware, “Timekeeping in VMware Virtual Machines”, *Commercial White Paper 2005, http://www.VMware.com/pdf/VMware_timekeeping.pdf*

[von Bertalanffy 68] L. von Bertalanffy, “General System Theory”, *George Braziller, ISBN 978-0807604533, pp. 55*

[Wahli 04] U. Wahli et al. “Software Configuration Management: A Clear Case for IBM Rational ClearCase and ClearQuest UCM”, *IBM RedBooks, ISBN 978-0738491592, 2004*

[Wang 03] S.Y. Wang et al. "The Design and Implementation of the NCTUns 1.0 Network Simulator," *Computer Networks, Vol. 42, Issue 2, June 2003, pp.175-197*

[Wang 07] J.H. Wang, J.Y. Pan, Y.C. Cheng, “Session Recognition and Bandwidth Guarantee for Encrypted Internet Voice Traffic : Case Study of Skype”, *Proceedings of the 2007 IEEE Symposium on Computational Intelligence and Data Mining (CIDM 2007)*

[Watson 97] A. Watson, M. Sasse, “Multimedia Conferencing via Multicast”, *Proceedings of the International Workshop on Audio-Visual Services over Packet Networks 1997*

Appendix B. Software Package References

[NCTUns] Name: NCTUns
Supplier: Simreal Technologies
License: Free for non-commercial / academic use, commercial license on a per-case basis
Type: Network Simulation Package
Homepage: http://nsl10.csie.nctu.edu.tw/

[QuickServer] Name: QuickServer
Supplier: Free software project
License: LGPL
Type: Java library for rapid development of TCP/IP server software
Homepage: http://www.quickserver.org/

[Rational] Name: Rational Suite for Software Engineering
Supplier: IBM
License: Commercial
Type: Integrated software engineering suite
Homepage: http://www.ibm.com/rational

[Wireshark] Name: Wireshark (previously known as Ethereal)
Supplier: Free software project
License: GPL
Type: Network Analysis Tool
Homepage: http://www.wireshark.org

Appendix C. Glossary

COTS	Commercial off-the-shelf
FTP	File Transfer Protocol
Gbps	Gigabits per second
GPL	GNU Public License
GUI	Graphical User Interface
IDE	Integrated Development Environment
IP	Internet Protocol
kbps	Kilobits per second
LAN	Local Area Network
LGPL	Lesser / Library GNU Public License
MAN	Metropolitan Area Network
Mbps	Megabits per second
MPEG3	MPEG Layer 3, a format for digital audio
MPEG4	MPEG Layer 4, a format for digital video
MPLS	Multi-protocol Label Switching
ms	Milliseconds
MVFS	Multi Version File System
NFS	Network File System
POTS	Plain Old Telephone System
RDP	Remote Desktop Protocol
RPC	Remote Procedure Call
RTP	Real-time Transfer Protocol
RTT	Round-trip time
RUP	Rational Unified Process
SDLC	Software Development Life Cycle
TCP	Transfer Control Protocol
UCM	Unified Configuration Management
UDP	User Datagram Protocol
USB	Universal Serial Bus
VOB	Versioned Object Base
WAN	Wide Area Network
WWW	Worldwide Web