

Construction of an Evaluation Model for Free/Open Source Project Hosting Sites

A thesis submitted in fulfillment of the requirements for
the degree of Doctor of Philosophy

Haggen Hau Heng So
B.Eng., Grad.Cert.Trans.

School of Business Information Technology
Business Portfolio
RMIT University
September 2005

Declaration

I certify that except where due acknowledgement has been made, the work is that of the author alone; the work has not been submitted previously, in whole or in part, to qualify for any other academic award; the content of the thesis is the result of work which has been carried out since the official commencement date of the approved research program; and, any editorial work, paid or unpaid, carried out by a third party is acknowledged.

.....

Haggen So

9 September 2005

The current version of this document is 0.9. Some of the permissions to include figures and tables from external source are not yet obtained. So do not be surprised if you see a big "X" in certain diagrams or tables. This document is regrettably created using MS Word 2000. The reasons to use these tools were functionalities and communication with the establishments. On the other hand, most of the diagrams in this thesis were drawn with OpenOffice Draw and I used PDFCreator 0.8.1 to convert the document to PDF. I believe if I start my PhD now, probably I can use mostly Free/Open Source Software.

Acknowledgements

Many people have offered assistance during this PhD programme and I cannot name all of them here. A number of people come to mind when I look back...

I would like to sincerely thank my senior supervisor, Dr. Nigel Thomas, for guiding me through the PhD programme as well as providing ample freedom so that I can develop the discipline of becoming an independent researcher. My gratitude also goes to Mr. Hossein Zadeh on his effort in providing technical support on the web server. The Faculty of Business provided funding for the scholarship and conference to make this PhD programme possible. The staff in the Research and Development Unit was very helpful. They even took up the task of appointing the examiners. Dr. France Cheong also proofread this dissertation and his suggestions on the grammatical mistakes and typos are taken. Last but definitely not the least, the companionship and expertise of the students in the business research lab were absolutely indispensable in this journey.

I would also like to thank the people who assisted during data collection. The participants of the Delphi survey included Kasper Edwards, Frank Tobin and 30 other anonymous participants. The online Delphi survey was tested by the system testing team, which was made up of Alpha Lau, Boon Chew Tay, Chandana Unnithan, John Yu, Kin-Kee Sit and Tat Wai Ho. Furthermore, Dr. Brett Scarlett and Professor Clive Morley gave invaluable comments on the design of the Delphi survey methods. During the detailed investigation of external FOSPHost sites, Roger Dingleline, Ryan Gordon, John Minnihan, Christian Reiniger and Chris Ryan gave useful and interesting comments on the topic and ibiblio.org provided the hosting service required. By releasing this thesis to the public under the Creative Commons Attribution-NonCommercial-NoDerivs license (see Appendix B), I wish to express my gratitude to above volunteers for their contributions to the common good.

My family was also very supportive during the programme. I would like to express my heartfelt gratitude my elder sister and brother in law with whom I lived with. They provided a warm home and my brother in law read through my writings and gave useful comments. The PhD programme is a journey of understanding phenomenon outside of myself as well as a journey inwards of self-discovery. In retrospect, I would never reach the end if my parents have not prepared me for this journey from the very beginning of my existence. The skills that I have, the personality that I possess, and many other things that I take for granted – their existence is not a coincidence, but the fruit of years of hard work and love. I can hardly find the right words to express my gratitude.

This research project is an exploratory study, and I have to stop when there are still many mysteries lying around. It has been said by Confucius (500 BC) that 'What you know, you know, what you don't know, you don't know. This is knowledge.' Nevertheless, when loneliness haunts, it is tempting to draw ungrounded conclusions for instant gratification. The friendship with the ultimate mystery, the God incarnated that suffered, helped and is still helping me to make friends with these mysteries. For the desire of conquering mysteries comes from fear, but the friendship provides peace.

Last but not least, I would like to thank the reader of this dissertation - yes, that is YOU. Most people would hardly care about this acknowledgements section, and your interest in this work is very much appreciated.

If you find anything worthy of praise in this work, probably it is due to someone mentioned above, and all the forerunners who kindly let me stand on their shoulders – their discoveries, theories and wisdom. Nevertheless, if you find any fault, then the blame probably lies with me.

H. So

Table of Content

Declaration	ii
Acknowledgements	iii
Table of Content.....	v
List of Figures	xi
List of Tables	xiv
Abbreviations	xvi
Abstract	2
Chapter I Introduction.....	3
1.1 Introduction	3
1.2 Rationale of the Research	6
1.3 Objectives of the Research	7
1.4 Research Methods.....	8
1.5 Contribution of the Research	8
1.6 Structure of This Dissertation	8
Chapter II Literature Review.....	11
2.1 Introduction	11
2.2 Formal Definition of Free/Open Source Software	11
2.3 The Most Well-Known Model - The Cathedral and The Bazaar.....	13
2.4 Relevant Areas of Interest in the Topic of Free/Open Source Phenomenon	16
2.5 Summary of Chapter Two.....	17
Chapter III Research Questions	19
3.1 Introduction	19
3.2 Free/Open Source Project Hosting Sites	19
3.3 Developing the Research Questions	21
3.4 Summary of Chapter Three	23

Chapter IV	Development of Analytical Frameworks	24
4.1	Introduction	24
4.2	Background for Analytical Frameworks.....	24
4.2.1	Free/Open Source Community, a definition	25
4.2.2	A Framework on Computer-Supported Co-operative Work (CSCW) and Analysis of an Free/Open Source Community	27
4.3	4C Model of a Free/Open Source Community	29
4.3.1	Communication	29
4.3.2	Contributions	30
4.3.3	Co-ordination.....	30
4.3.4	Culture	33
4.4	A Model of Individual Participation to a Free/Open Source Community	34
4.5	Notes on the Construction of the Model of Individual Participation to a Free/Open Source Community	37
4.6	Comparison Between the Bazaar Model and the Model of Individual Participation to a Free/Open Source Community	40
4.7	Comparison Between the Other Models and the Model of Individual Participation to a Free/Open Source Community	40
4.7.1	Software Development Based Models	41
4.7.2	Comprehensive Models	43
4.7.3	Comparison of the Models	46
4.8	The Model of Individual Participation to a Free/Open Source Community and FOSPHost Design and Deployment	49
4.9	Summary to Chapter Four	49
Chapter V	Methodology	51
5.1	Introduction	51

5.2	Overall Research Strategy	51
5.3	Selection of Research Methodologies and Methods.....	55
5.3.1	Considerations in the Construction of the FOSPHost Evaluation Model	59
5.3.1.1	Introduction	59
5.3.1.2	Software Evaluation During Development	60
5.3.1.3	Software Product Evaluation.....	67
5.3.1.4	Software Evaluation and FOSPHost	70
5.3.1.5	Presentations of Evaluation	72
5.3.1.6	Users of Evaluation	78
5.3.1.7	Summary.....	79
5.3.2	Delphi Survey.....	79
5.3.2.1	Administration of Instruments and Procedures	81
5.3.2.2	Participants of Survey.....	81
5.3.2.3	Questionnaire Development for the Survey	82
5.3.2.4	Implementation of Survey on the Web Server.....	85
5.3.2.5	Data Analysis.....	121
5.3.3	Detailed investigation on External Hosting Sites.....	123
5.4	Summary of Chapter Five.....	126
	Chapter VI Results and Analysis of the Delphi Survey	128
6.1	Introduction	128
6.2	Results of the Delphi Survey.....	128
6.2.1	Invitations and Responses	128
6.2.2	Agreed Answers.....	137
6.2.3	Controversial Answers.....	144
6.3	Analysis of the Results of the Delphi Survey.....	155
6.3.1	Delphi Survey Method	155

6.3.2	Responses and Validity.....	157
6.3.3	Possible Improvements in Delphi Survey Method.....	162
6.3.4	Data Analysis.....	164
6.3.5	Discussion of Results	167
6.4	Summary of Chapter Six	171
Chapter VII Detailed Investigation on External Hosting Sites		173
7.1	Introduction	173
7.2	Data Collection and Selection of Sites	173
7.2.1	Infrastructure and Non-infrastructure sites.....	174
7.2.2	Introduction to Infrastructure Sites.....	175
7.2.3	Introduction to Non-infrastructure Sites.....	178
7.3	Comparison of External Hosting Sites	179
7.3.1	General Information	180
7.3.2	Project Tools - Tools for Public/Developers.....	183
7.3.3	Project Tools - Tools for Project Administrators	194
7.3.4	Personal Tools for Developers.....	202
7.3.5	Community Tools	206
7.3.6	Others	215
7.4	Discussion of the Comparison.....	222
7.5	Summary of Chapter Seven.....	231
Chapter VIII Construction of the Evaluation Model		232
8.1	Introduction	232
8.2	Data Collected and Choice of Evaluation Presentation.....	232
8.3	Implementation of the Evaluation Presentations	234
8.3.1	Tools Chosen for Implementation	235
8.3.2	Evaluation Model Implemented with the Chosen Tools	237

8.4	Discussion of the Evaluation Model.....	251
8.5	Summary of Chapter Eight.....	252
Chapter IX Discussion of Results		254
9.1	Introduction	254
9.2	Reflections and Limitations of this research	254
9.3	Implications of the Findings – Free/Open Source as a Different Paradigm	256
9.4	Summary of Chapter Nine.....	264
Chapter X Conclusion.....		265
10.1	Introduction	265
10.2	Summary of Findings	265
10.2.1	The Model of Individual Participation to a Free/Open Source Community and Software Evaluation Classification.....	266
10.2.2	Delphi Survey.....	267
10.2.3	Detailed investigation.....	267
10.2.4	Evaluation Model	268
10.2.5	Contributions of the Findings.....	268
10.4	Further Research.....	269
10.4.1	Further Research on FOSPHost.....	269
10.4.2	Further Research on the Broader Context of the Free/Open Source Phenomenon	272
10.5	Possible Future of the Software Industry and the Potential Applications of the Findings	278
List of References.....		285
Appendix A	Related Publications.....	305
Appendix B	Licenses of Different Portions of the Dissertation and Other Copyright Issues.....	306

Appendix C	Content of Enclosed CD-ROM.....	310
Appendix D	Jane Jacob's Systems of Survival.....	311
Appendix E	Susceptibility of Average and Variance	313
Appendix F	Software Configuration System and Source Code Repository.....	315
F.1	Historical Influences.....	315
F.2	A Closer Look at CVS.....	319
F.3	Limitations of CVS	322
F.4	Other Systems Used in the Free/Open Source Communities	323
F.5	Conclusion	324
Appendix G	Results of Free/Open Source Hosting (FOSPHost) Sites Delphi Survey	325
Appendix H	WakkaWiki Pages.....	326
Appendix I	Source Code for Delphi Survey	327
Appendix J	Source Code for Evaluation Model.....	339

List of Figures

Figure 2-1 Three areas of interest in Free/Open Source	16
Figure 4-1 Integrative Three Phase Model of Virtual Communities and Society (Romm, Pliskin & Clarke 1997, p. 269)	26
Figure 4-2 A Framework on CSCW	28
Figure 4-3 4C Model of a Free/Open Source Community	29
Figure 4-4 A Model of the Social Structure of Free/Open Source Community (Lawrie, Arief & Gacek , 2002, p. 77) modified by the authors (* denotes the modification)	31
Figure 4-5 Communication Pattern of GCC (Yamauchi et al. 2000, p. 7)	33
Figure 4-6 A Model on individual participation in an Open Source/Free Software Community	35
Figure 4-7 Open source characteristics - common and variable (Gacek, Lawrie & Arief 2001, p. 79)	43
Figure 4-8 OSS Model (Sharma, Sugumaran & Rajagopalan 2002, p. 18)	44
Figure 5-1 Exploratory, Descriptive and Explanatory Research	52
Figure 5-2 The overall research strategy of this research	54
Figure 5-3 Quality of Use Measures Determined by the Context of Use (Bevan 1995)	66
Figure 5-4 Site Map of Major Elements for Delphi Survey	87
Figure 5-5 Register/Login Page	88
Figure 5-6 Participants Details	89
Figure 5-7 Information Centre	90
Figure 5-8 Round 1 Questionnaire Question Page	91
Figure 5-9 Round 1 Questionnaire Answer Page	92
Figure 5-10 Menu for Question 2	92
Figure 5-11 Adding a New Tool	93
Figure 5-12 Adding Name and Description	93

Figure 5-13 Suggesting Features	94
Figure 5-14 Selecting Preset Usability Factors	94
Figure 5-15 User-defined Usability Factors	95
Figure 5-16 Summary of Responses	97
Figure 5-17 Verification Page	98
Figure 5-18 Additional Clarification	99
Figure 5-19 Results of Round 1 Sorted by Questions in Short Form	100
Figure 5-20 Results of Round 1 Sorted by Questions in Long Form	100
Figure 5-21 Participants Grouped by Self Rating	101
Figure 5-22 Results of Round 1 by Participants in Short Form	102
Figure 5-23 Results of Round 1 by Participants in Long Form	103
Figure 5-24 Round 2 Questionnaire Answer Page	104
Figure 5-25 Randomisation of Statement Order	105
Figure 5-26 Show Only Top Ten	106
Figure 5-27 Show Only Numerical Data	107
Figure 5-28 Show All Relevant Data (Sort by Rating)	108
Figure 5-29 Show All Relevant Data (Sort by Controversy)	109
Figure 5-30 Detail Chart of Distribution of Responses	110
Figure 5-31 Responses of a Participant	111
Figure 5-32 Detail Responses of a Participant	112
Figure 5-33 Round 3 Questionnaire Question Page	113
Figure 5-34 Round 3 Questionnaire Answer Page	114
Figure 5-35 Checking Glossary for Difficult Terms	115
Figure 5-36 Checking Qualitative Results from Last Round	116
Figure 5-37 Checking Quantitative Results from Last Round	117
Figure 5-38 Results of Round 3	118

Figure 5-39 Post-Delphi Survey	119
Figure 5-40 Post-Delphi Survey Results	120
Figure 6-1 No. of Invitation Sent in Round 3	133
Figure 6-2 Histogram of Average Ratings	137
Figure 6-3 Histogram of Variance of Ratings	144
Figure 7-1 Overview of Bugs	189
Figure 7-2 Details of a Bug Report	190
Figure 8-1 Site Map for Evaluation Model	238
Figure 8-2 Front Page of Wiki	239
Figure 8-3 What is a FOSPHost Site?	240
Figure 8-4 Preferred Attributes of a FOSPHost Site	241
Figure 8-5 Preferred Attributes - Utility	241
Figure 8-6 Preferred Attributes - Context	242
Figure 8-7 Controversial Issues of a FOSPHost Site	243
Figure 8-8 We love freedom, but how far can it go?	243
Figure 8-9 Opinions for Generating Customised Comparison Table	244
Figure 8-10 Example of Customised Comparison Table Generated (1)	245
Figure 8-11 Example of Customised Comparison Table Generated (2)	246
Figure 8-12 Step 1 of Weighed Checklist	247
Figure 8-13 Step 2 of Weighed Checklist	249
Figure 8-14 Step 3 of Weighed Checklist (1)	250
Figure 8-15 Step 3 of Weighed Checklist (2)	251
Figure F-1 Illustration of the operation of <i>diff</i>	317
Figure F-2 Typical CVS Operation	319
Figure F-3 The Tree Diagram for Branching and Merging for a Certain File	321

List of Tables

Table 4-1 Three Types of Free/Open Source Projects (Nakakoji et al. 2002)	41
Table 4-2 Comparison of the Four Models based on the Model on Individual Participation in an Open Source/Free Software Community	46
Table 5-1 A Summary of Differences among the Three Approaches to Research (Neuman, Bondy & Knight 2003, p. 91)	56
Table 5-2 An Example of Courseware Evaluation	74
Table 5-3 Sample evaluation matrix using scores and weighing	75
Table 5-4 Implications of Different Forms of Presentation	77
Table 5-5 Preset Usability Factors	97
Table 6-1 Numbers of Participants Involved in Each Question	129
Table 6-2 Breakdown of Participants based on Expertise and Participation	131
Table 6-3 Amount of Participation	131
Table 6-4 Numbers of People Invited for Each Round	131
Table 6-5 Statistics for Invitation	132
Table 6-6 'No Comment' Responses from Question Page	134
Table 6-7 'No Comment' Responses from Answer Pages	134
Table 6-8 Round 2 References to Results	135
Table 6-9 Round 3 References to Results	135
Table 6-10 Difference in Rating in Round 2 and 3	136
Table 6-11 Division of Important Statements	138
Table 6-12 The Most Important Statements from the Delphi Survey	142
Table 6-13 Division of Controversial Statements	145
Table 6-14 The Most Controversial Statements from the Delphi Survey	149
Table 6-15 Comparison of Number of Participants for Different Delphi Survey	159
Table 6-16 Number of Statement Selected in Each Question	165

Table 7-1 Comparison of General Information of FOSPHost Sites	181
Table 7-2 Comparison of 'Project Tools - Tools for Public/Developers' of FOSPHost Sites (1)	185
Table 7-3 Comparison of 'Project Tools - Tools for Public/Developers' of FOSPHost Sites (2)	188
Table 7-4 Comparison of 'Project Tools - Tools for Public/Developers' of FOSPHost Sites (3)	193
Table 7-5 Comparison of 'Project Tools - Tools for Public/Developers' of FOSPHost Sites (1)	195
Table 7-6 Comparison of 'Project Tools - Tools for Public/Developers' of FOSPHost Sites (2)	198
Table 7-7 Comparison of 'Project Tools - Tools for Public/Developers' of FOSPHost Sites (3)	200
Table 7-8 Comparison of 'Personal Tools for Developers'	204
Table 7-9 Comparison of 'Community Tools' (1)	209
Table 7-10 Comparison of 'Community Tools' (2)	212
Table 7-11 Comparison of 'Community Tools' (3)	214
Table 7-12 Comparison of 'Others' (1)	216
Table 7-13 Comparison of 'Others' (2)	218
Table 7-14 Comparison of 'Others' (3)	220
Table 7-15 Comparison of 'Others' (4)	221
Table 7-16 Number of Features excluding 'General Information' and 'Others'	225
Table 7-17 Number of Features excluding 'General Information', 'Personal Tools for Developers' and 'Others'	226
Table B-1 List of Trademarks Acknowledged	308
Table D-1 Commercial and Guardian Moral Syndromes (Jacobs 1993, pp. 23-4)	311

Abbreviations

BGI	Barclays Global Investors
CGI	Common Gateway Interface
CMM	Capability Maturity Models
COTS	Commercial-Off-The-Shelf (Software)
CSCW	Computer-Supported Co-operative Work
CVS	Concurrent Versions System
FOSPHost	Free/Open Source Hosting (site)
FSF	Free Software Foundation
FTP	File Transfer Protocol
GNU	GNU Not Unix
GPL	General Public License
HCI	Human-Computer Interaction
HP	Hewlett-Packard
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
IBM	International Business Machines
IDE	Integrated Development Environment
IP	Internet Protocol
IRC	Internet Relay Chat
IT	Information Technology
ITS	Incompatible Time Sharing system
JSP	JavaServer Pages
LUG	Linux User Group
OSDN	Open Source Development Network
OSS	Open Source Software
PDF	Portable Document Format
PHP	PHP Hypertext Preprocessor
PSP	Personal Software Process
SCP	Secure Copy Protocol
SEUL	Simple End-User Linux project
SFTP	Secure File Transfer Protocol
SSH	Secure Shell

SSI	Server Side Include
SSL	Secure Sockets Layer
TSP	Team Software Process
WWW	World Wide Web

(Some definitions are recursive by 'definition')

**CONSTRUCTION OF AN
EVALUATION MODEL FOR
FREE/OPEN SOURCE
PROJECT HOSTING SITES**

Abstract

Free/Open Source software is a kind of software whose source code is available for comprehension, modification and re-distribution. This kind of software has increased in popularity in recent years and becoming an interesting topic for research. Most Free/Open Source software is produced through the facilitation of Free/Open Source Hosting (FOSPHost) sites and investigations into these sites may yield results that have theoretical and practical significance.

The purpose of study selected was exploratory and a positivist approach was adopted as main methodology. Literature was surveyed and suitable analytic frameworks were built. Based on these frameworks, an online Delphi survey was conducted to collect expert opinion on important issues of FOSPHost. A detailed investigation of ten FOSPHost sites was conducted. The results from the two data collection processes was condensed and presented in an evaluation format so that practitioners and researchers alike can gain more understanding in the design and the deployment of FOSPHost sites.

Chapter 1

Introduction

1.1 Introduction

The Free/Open Source phenomenon is a surprise with a mystery. The market share of a popular Free/Open Source web server, Apache, was 69% comparing to 23% for Microsoft servers in January 2004 (Netcraft 2004). In the operating system market at the end of 2001, Linux server, a Free/Open Source system, had 26% while Microsoft had 49% of the market share. Microsoft was still the leader of the market, but 45% of all new servers shipped were predicted to be Linux in the year of 2006 or 2007 (Wilcox & Shankland 2002). Another survey undertaken by a magazine for IT managers using Microsoft servers showed that two out of five enterprises also employed Linux. More than 800 enterprises were surveyed with an average number of servers running in these companies of 400 (McKendrick 2003). Though a number of companies such as IBM and HP now support Linux development as a strategy to combat Microsoft, the idea of Linux is owned or controlled by neither of these companies defies common business logic. Wilcox & Shankland (2002) claimed that Microsoft now take this opposition very seriously.

To explain simply, Free/Open Source software is a piece of software whose its source code is made freely available. Source code is the original form of a computer program as written by the programmer (Freedman 1998). A piece of software that is Free/Open Source ensures that any

person can readily understand how a piece of software works, modify it and redistribute it (Free Software Foundation 2000). In the early history of software, most source code was shared between companies and customers (Levy 1984). It was only later that the strategy of making money by hoarding the source code of software became the standard practice of the software industry.

Though Free/Open Source is very much about software and software development, its effect can reach even further. A number of Free/Open Source communities participate and shape political movement online (Free Software Foundation 2002; Raymond 2000c). Some people also have been trying to apply the idea of Free/Open Source in other areas such as education (Bull & Garofalo 2003) and even forestry management (Schweik & Semenov 2003). Therefore, in order not to lead readers to focus only on software or software development, the author will use a broader term 'the Free/Open Source phenomenon' to refer to what has happened so far in a broader context.

The reader may wonder why the term 'Free/Open Source' is used to qualify software that the source code is made freely available in this study, rather than the more commonly used term, 'Open Source'. 'Free/Open Source' is a combination of the terms 'Free Software' and 'Open Source'. The term 'Free Software' is promoted by the Free Software Foundation, which advocates Free Software as a social movement that non-Free Software is morally wrong (Free Software Foundation 2002). On the other hand, the term 'Open Source' is promoted by the Open Source Initiative, which advocates the practical benefit of Open Source software development to the commercial world (Open Source Initiative 2003b). These two views are both relevant and thus the term 'Free/Open Source' is used. The author here maintains a political view that is neutral to both movements.

The Free/Open Source phenomenon has the potential to attract the attention of the academic circle, as there are a number of issues that require explanations. First, it is hard to reconcile that the cost of the development of some highly complex Free/Open Source projects can be so low. For example, Red Hat Linux 7.1 was estimated to cost more than one billion US dollars to develop using conventional software development approach (Wheeler 2002). Significant monetary investment towards Linux is only a recent phenomenon and thus the estimation above was huge discrepancy with the reality.

Second, it is also difficult to reconcile the assertion proposed by Raymond (2000b) that the development process of this software was chaotic, which was a distinct diversion from the traditional controlled and structured paradigm of software development. In Raymond's article of the Cathedral and the Bazaar Raymond (2000b), he stated that for system with substantial complexity, the traditional method of software development process would involve the hard work of a small team of talented individuals (the Cathedral) at the start. He then explained that the development of Linux showed us how a collective effort of co-developers over the Internet (the Bazaar) could possibly produce quality software with better reliability and more useful features in a shorter time (Raymond 2000b). Moreover, Raymond critiqued the validity of a famous principle in software engineering, Brooks's law, in the light of the development of Linux. Brooks's law (Brooks 1995) stated that as the number of developers increase in a software project working on inter-related tasks, the communication cost will eventually become larger than the benefit of the work produced by the extra labour added. Raymond argued that as the number of developers contributing to Linux was large, Brooks's law could only be partly true.

Given the lack of reconciliation of these issues, there should be more academic and industrial investigations. Nevertheless, academic research on the topic has just began and in one of the

first academic books published on Free/Open Source, Feller & Fitzgerald (2002) suggested that more research was required on the nature of Free/Open Source.

Within the topic of Free/Open Source, the area of Free/Open Source Project Hosting (FOSPHost) sites was chosen as the focus of this study. A FOSPHost site is the infrastructure that supports and co-ordinates the development of Free/Open Source software projects on the Internet. In short, Free/Open Source developers collaborate through the FOSPHost sites to produce Free/Open Source software.

1.2 Rationale of the Research

The area of FOSPHost was chosen as it is an important subject both in application and in theory. On one of the most popular FOSPHost sites, SourceForge, 74,131 projects were hosted with 766,950 registered users on the day of 9 January 2004 (SourceForge 2004). These statistics may suggest that many developers employ FOSPHost sites for facilitating projects in the Free/Open Source communities.

Other than the Free/Open Source communities, the technology of FOSPHost also catches the attention of the business world. Sun Microsystem employed Collab.Net to host six Open Source projects externally such as OpenOffice and NetBeans (Collab.Net 2003a). These projects were hosted using the flagship product of Collab.Net, SourceCast, which was a collaborative software development environment inspired by FOSPHost with improvements such as access permissions to fit corporate needs. Collab.Net and VA Software (which sells an improved version of SourceForge) both had business alliances with major players in IT industry such as IBM and Oracle (Collab.Net 2003d; VA Software 2003).

One may wonder why businesses were interested in FOSPHost technology. A number of possible reasons could be found in the Yankee Group report on the employment of SourceCast

by a global financial firm, Barclays Global Investors (BGI) (Derome & Huang 2003). These reasons included the decrease in time-to-market of software products and increase in customer satisfaction due to improved communication between business units, technical units and customers. Internal software development infrastructure was streamlined around SourceCast and savings on administrations, personnel and hardware were obtained.

If Free/Open Source becomes more widely accepted, both in software and as a concept, the significance of the topic of FOSPHost will also increase. If an evaluation model of FOSPHost sites could be constructed, it could have the potential to become a useful tool for the examination of the design and employment of these sites.

Other than the application of FOSPHost sites, the study of these sites may also advance theoretical understanding of Free/Open software development process. This is simply because FOSPHost sites are where Free/Open Source software development is facilitated. The understanding of these sites can be a promising way to gain insights into the process.

1.3 Objectives of the Research

Though the concept of sharing source code was nearly as old as the invention of computers, at the commencement of this study, research on Free/Open Source was scarce. There are many unanswered questions in the Free/Open Source phenomenon. This study, therefore, aims at discovering the areas relevant to the topic of FOSPHost and establishing the boundaries for data collection. Analytical frameworks will be built from literature as a starting point for investigation. Important issues in the design and employment of FOSPHost sites will then be obtained. The findings will be presented in an evaluation format available on the Internet.

1.4 Research Methods

In order to achieve the objective stated, the purpose of research was chosen to be exploratory. Another choice was that positivism was adopted as the main methodology of this study. For exploratory studies, flexibility was required in order to discover new knowledge. Though positivism will be guiding methodology of this study, interpretivism may be employed at times when appropriate.

The study will begin from literature review on topic of Free/Open Source and FOSPHost. An online Delphi survey will then be conducted to collect expert opinion on the topic. A more detailed investigation on the backgrounds, policies and features of FOSPHost sites will then be conducted. The findings will then be presented as an evaluation model.

As the study is exploratory, one of the possible limitations is that there will be more emphasis on collecting a broad range of data with less emphasis on the depth of each issue.

1.5 Contribution of the Research

One of the potential contributions of this research is that the results might be useful to practitioners. Furthermore, academic investigations in the area of FOSPHost are rare and the findings of the study could uncover important issues based on data to promote understanding of the topic. Moreover, the theoretical frameworks built may also become useful analytical tools for researchers.

1.6 Structure of This Dissertation

This dissertation consists of ten chapters. The first chapter is the current chapter, which contains an overview of the study. Chapter two to four lay the foundation and define the boundary for the research. Chapter five contains the methodology for data collection and the

discussion for evaluation approaches for FOSPHost. Chapter six to nine contain the result and analysis of the data collected and the final chapter, chapter ten, is the conclusion.

The second chapter is the basic literature review. Literature on the Free/Open Source phenomenon will be surveyed to lay the foundation for the rest of the study. Specific literature of the topic of FOSPHost is presented in chapter three and the research question and sub-questions are formulated. Two analytic frameworks are developed in the fourth chapter to establish boundaries for the data collection on the topic of FOSPHost.

The fifth chapter covers methodology. Choices of methodologies and methods are explained. Detail designs of methods for data collection are elaborated. Software evaluation methods are also reviewed and a new software evaluation classification is built to suit the nature of Free/Open Source software and FOSPHost.

Chapters six to nine present the results and analysis of research. Chapter six contains the results and the analysis of the Delphi survey. Chapter seven contains the data collected from a detailed investigation of ten FOSPHost sites. The construction of the final product of this study, an evaluation model for FOSPHost sites, can be found in chapter eight. The overall quality of the result obtained is discussed in chapter nine and the limitations of the study are identified. The implications of the findings relating to other literature and the real world are elaborated.

Chapter ten is the final chapter when the study is concluded. Further research directions are suggested and the possible areas of the application of the findings in the future are proposed.

This dissertation is also available in digital PDF format in the CD-ROM enclosed (/eval_fosphost.pdf). The reader is encouraged to take advantage of the digital format by

employing extra functionalities such as searching and printing to complement the reading of this hard copy. Please also refer to appendix B for the copyright issues.

Chapter 2

Literature Review

2.1 Introduction

The literature review chapter will begin with a more formal definition of Free/Open Source Software. Eric Raymond's 'the Cathedral and the Bazaar' metaphor (Raymond 2000b) which was introduced in the last chapter will be further explained and analysed. Since the Free/Open Source phenomenon is relatively new, relevant literature may not contain obvious keywords such as Free Software or Open Source on the title. Relevant areas will first be identified to establish the boundaries for the research.

2.2 Formal Definition of Free/Open Source Software

To define Free/Open Source, the usual method was to start from software (Feller & Fitzgerald 2002; Open Source Initiative 2003a). A simple definition has already been given in the introduction that a piece of software that is Free/Open Source is one that any person can readily understand how it works, modify it and redistribute it (Free Software Foundation 2000). A more comprehensive and formal definition can be found at the Open Source Initiative web site (Open Source Initiative 2003a):

1. Free Redistribution

The license shall not restrict any party from selling or giving away the software as a component of an aggregate software distribution containing programs from several different sources. The license shall not require a royalty or other fee for such sale.

2. Source Code

The program must include source code, and must allow distribution in source code as well as compiled form. Where some form of a product is not distributed with source code, there must be a well-publicized means of obtaining the source code for no more than a reasonable reproduction cost—preferably, downloading via the Internet without charge. The source code must be the preferred form in which a programmer would modify the program. Deliberately obfuscated source code is not allowed. Intermediate forms such as the output of a preprocessor or translator are not allowed.

3. Derived Works

The license must allow modifications and derived works, and must allow them to be distributed under the same terms as the license of the original software.

4. Integrity of The Author's Source Code

The license may restrict source-code from being distributed in modified form only if the license allows the distribution of 'patch files' with the source code for the purpose of modifying the program at build time. The license must explicitly permit distribution of software built from modified source code. The license may require derived works to carry a different name or version number from the original software.

5. No Discrimination Against Persons or Groups

The license must not discriminate against any person or group of persons.

6. No Discrimination Against Fields of Endeavor

The license must not restrict anyone from making use of the program in a specific field of endeavor. For example, it may not restrict the program from being used in a business, or from being used for genetic research.

7. Distribution of License

The rights attached to the program must apply to all to whom the program is redistributed without the need for execution of an additional license by those parties.

8. License Must Not Be Specific to a Product

The rights attached to the program must not depend on the program's being part of a particular software distribution. If the program is extracted from that distribution and used or distributed within the terms of the program's license, all parties to whom the program is redistributed should have the same rights as those that are granted in conjunction with the original software distribution.

9. The License Must Not Restrict Other Software

The license must not place restrictions on other software that is distributed along with the licensed software. For example, the license must not insist that all other programs distributed on the same medium must be open-source software.

10. The License must be technology-neutral

No provision of the license may be predicated on any individual technology or style of interface.

(The Rationale section in the original text is deleted.)

This definition is widely accepted; Feller and Fitzgerald (2002) argued that this definition satisfied the necessary and sufficient conditions to characterize a piece of Free/Open Source software.

Though the formal definition of Open Source quoted was suggested to cover the necessary and sufficient conditions for Free/Open Source software, Gacek, Lawrie & Arief (2001) showed that a definition just on software could not completely illustrate the many underlying meanings of the Free/Open Source phenomenon. Thus, a number of explanations were devised (Feller & Fitzgerald 2002; Lawrie, Arief & Gacek 2002; Nakakoji et al. 2002; Raymond 2000b; Sharma, Sugumaran & Rajagopalan 2002; So, Thomas & Zadeh 2002) and they will be discussed in the following sections.

2.3 The Most Well-Known Model - The Cathedral and The Bazaar

The first model to be examined is Eric Raymond's 'the Cathedral and the Bazaar' metaphor (Raymond 2000b), which is the most well-known model to explain the Free/Open software development process. This was introduced in the previous chapter and it was one of the earliest explanations of the how Free/Open Source software could evolve into such a complex system like Linux. Indeed, the practice of making the source code freely available existed nearly as long as the invention of the computer itself and turning source code into proprietary software and distributing only the compiled binary is a relatively recent concept (Levy 1984). There were also written accounts on the some of the most open systems in history, such as ITS, the Incompatible Time-sharing System (Levy 1984; Turkle 1984), but the software development process associated was seldom investigated in depth. Therefore, Raymond's metaphor then became the most frequently used explanation for Free/Open Source software development process. This metaphor even had an influential impact on the decision of Netscape to open up

the source code of its browser product and started one of the most famous commercial Open Source projects, Mozilla (Hamerly, Paquin & Walton 1999; Moody 2001).

In Raymond's article of the Cathedral and the Bazaar (Raymond 2000b), he used the metaphor of Bazaar to explain the mechanism of Free/Open Source software development as a distinct paradigm from conventional approaches. Moreover, he argued if a project like Linux, which involve such large number of developers, could efficiently produce quality software with substantial complexity, then Brooks's law could only be partly true. Other forces were at work to increase efficiency. One of these other forces was egoless programming proposed by Weinberg (1971). This theory described that if programmers share their source code among their peers, errors in code can be discovered more readily. Other benefits included the improvement in the readability of code, the increase of familiarity with the code by other team members, and eventually, an improvement in efficiency. Raymond also suggested that as the Internet became available to the public, the boundary of egoless programming could be expanded even further to any interested parties globally. Linus Torvalds, the founder of Linux, was among the first to utilise the potential of this situation. More understanding of the Free/Open Source phenomenon is again required to further examine this argument. (A common misconception is that Raymond proved Brooks's law wrong and Brooks's law is not applicable anymore. In Raymond's own words, he claimed, "I don't consider Brooks' Law 'obsolete' any more than Newtonian physics is obsolete; it's just incomplete. Just as you get non-Newtonian effects at high energies and velocities, you get non-Brooksian effects when transaction costs go low enough. Under sufficiently extreme conditions, these secondary effects dominate the system -- you get nuclear explosions, or Linux." (Jones, P. 2000) As will be developed further in this dissertation, Brooks's law still has a role to play.)

Another important enabling factor suggested by Raymond (2000b) was the satisfaction in gaining reputation in a community of developers as the motivation. Egoless programming was suggested as one significant enabling factor in the Linux development, but this principle did not explain the willingness to collaborate. Raymond's answer to this question was that ego boosting among peers was the driving force.

Critics of the Bazaar metaphor suggested that the model provided 'too few data points' to construct a picture of the approach (Eunice 1998b). Extended interpretations to fill the gaps in the Cathedral metaphor can sometimes be found in literature. Examples of those are 'The Cathedral represents a monolithic, highly planned, top-down style of software development' (Eunice 1998a), 'All alternative models (considered to be one and called the "Cathedral model")' (Bezroukov 1999a) and 'The paper essentially ignored contemporary techniques in software engineering, using the Cathedral as a pseudonym for the waterfall lifecycle of the 1970s (Royce 1970)' (Johnson 1999). On the other hand, for the Bazaar metaphor, most interpretations did not go beyond the boundaries of Raymond's article. A yearning for a more detailed explanation is implied in the following quotes from literature, 'somehow results in high quality software' (Pavlicek 2000, p. 11) and 'for some mysterious reason' (Bezroukov 1999a). These authors were probably seeking a more substantial explanation of the exact mechanism of the Free/Open Source software development process.

In order to have a comprehensive understanding of the Free/Open Source phenomenon, more literature needs to be reviewed. Nevertheless, as the phenomenon is quite new, relevant literature may not have an obvious 'Free Software' or 'Open Source' tag in the title or abstract. Relevant areas of interest will be proposed instead in order to proceed.

2.4 Relevant Areas of Interest in the Topic of Free/Open Source Phenomenon

It can be proposed that there are three relevant areas of interest in Free/Open Source, namely the contextual, technological and socio-economical aspects. The three aspects proposed are not mutually exclusive and they all overlap with each other (Figure 2-1).

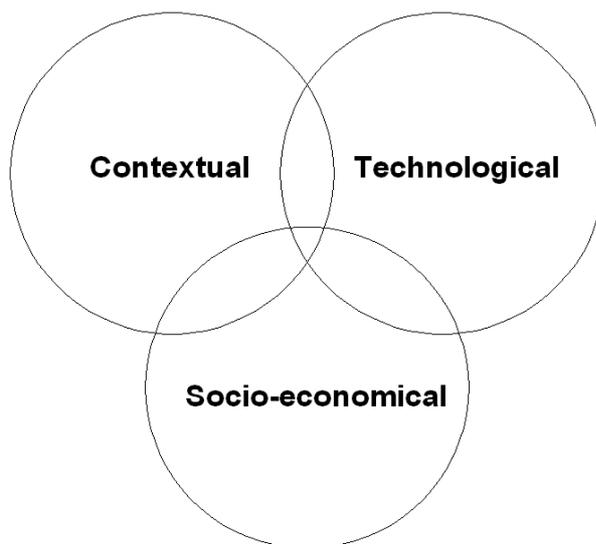


Figure 2-1 Three areas of interest in Free/Open Source

First of all, the Free/Open Source phenomenon emerged from its own historical context. Though the term 'Open Source' was coined on the 3rd February, 1998 (Open Source Initiative 2000), the historical context of the movement includes the history of Unix operating system (Hauben & Hauben 1997; Salus 1995), the Internet (Hauben & Hauben 1997; Licklider & Taylor 1968), and the hacker culture (Levy 1984; Raymond 2000c; Turkle 1984). The Free Software Foundation and the GNU project also played a very significant role (Feller & Fitzgerald 2002; Levy 1984; Moody 2001). The contemporary context of Free/Open Source includes business interest in Open Source (Apple Computer Inc. 2002; Hamerly, Paquin & Walton 1999; IBM 2003; SGI 2003; Sun Microsystems Inc.) such as how Linux was employed as a weapon against Microsoft and other competitors (Bezroukov 2002; Wladawsky-Berger 2001).

Free/Open Source communities also consist of a socio-economical aspect and relevant topics includes virtual communities and virtual organizations (Crowston & Scozzi 2002; Dafermos 2001; Gallivan 2001; Kollock 1996; Markus, Manville & Agres 2000; Rheingold 1993; Romm, Pliskin & Clarke 1997; Sharma, Sugumaran & Rajagopalan 2002; So, Thomas & Zadeh 2002; Wellman & Gulia 1999), the current state of hacker culture (Moody 2001; Pavlicek 2000; Raymond 2000b, 2000a), information economy (Clarke 1999; Ghosh 1998a; Kollock 1999; Lancashire 2001; Lerner & Triole 2002) and the political influences of Free/Open Source (Forge 2000; Free Software Foundation 2002; Newman 1999; The Associated Press 2000; Yee 1999).

Free/Open Source communities are mostly made up of members with technical background (Lakhani et al. 2003) and thus technology is another indispensable aspect. Topics such as architecture (such as the microkernel vs monolithic debate (DiBona, Ockman & Stone 1999)) and features (such as technical supremacy of Linux over Microsoft (The Unix vs NT Organisation 2001)) of software were always important focuses in the communities.

From the elaboration of the three areas of interest above, some relevant literature is identified. Nonetheless, the areas covered need to be further reduced to focus on FOSPHost and a structure is required to categorise and present this volume of relevant literature.

2.5 Summary of Chapter Two

In this chapter, a formal definition of Free/Open Source Software was introduced. The most well-known explanation to Free/Open Source phenomenon – the Cathedral and the Bazaar – was examined and its short-comings was discussed. In search for a more comprehensive explanation, three relevant areas of interest were identified, namely the contextual, technological and socio-economical aspects.

In the next chapter, the focus of this research, FOSPHost, will be explained further and the research question and sub-questions will be developed.

Chapter 3

Research Questions

3.1 Introduction

In the chapter, more details on Free/Open Source Project Hosting (FOSPHost) sites will be explained. The overall research question and sub-questions will also be formulated.

3.2 Free/Open Source Project Hosting Sites

A FOSPHost site is an important tool within the communities and it is defined as the infrastructure that supports and co-ordinates the development of Free/Open Source software projects on the Internet.

Surveying one of the most popular FOSPHost sites online, SourceForge (SourceForge 2003), it provides a dazzling array of services to manage a project, namely issue trackers, forums, mailing list, announcement area, document manager, task manager, file release system and concurrent versions system (CVS). It also provides a compile farm for porting software to other platforms. Since SourceForge hosts a large number of projects, it also provides facilities for inter-project communication. First of all, projects hosted are grouped into foundries to encourage communication between similar projects. Software metrics are also calculated for comparison and competition. There is also an area to call for contribution from other developers.

Not every FOSPHost site is required to be as extensive as SourceForge in order to be useful. Some are as simple as having a mailing list for communication and an FTP server to download the components of the project. Indeed, this was what Linus Torvalds employed for a substantial amount of time to co-ordinate the development of the Linux kernel (Asklund & Bendix 2002) before the deployment of a more sophisticated version control system called BitKeeper (Barr, J. 2002). Even after the introduction of BitKeeper to take the load of co-ordination, mailing lists and FTP servers still remain as important components of the system. The information in the Linux kernel mailing list is important enough that there is even a digest service on the content of the list (Brown et al. 2003).

One can even trace back the history of FOSPHost to the historical ITS system. As mentioned in sub-section 4.5, this system allowed any user to change any code on the system. Moreover, users could actually switch to other users' terminal and did programming collaboratively. ITS system thus was an infrastructure to support and co-ordinate software development and the code developed is still freely available on the Internet (Alan 2001).

On the other hand, web sites that aggregate information about Free/Open Source projects for queries such as Freshmeat (OSDN 2003b) are not regarded as FOSPHost. Popular geek community web sites such as Slashdot (OSDN 2003a) or Advogato (Advogato 2003) are also not classified as FOSPHost. These web sites are indeed very much related to Free/Open Source projects but they did not provide co-ordination tools for software development.

FOSPHost sites can also be classified as external hosting and self-hosting. The distinction between the two is the amount of control the users of the FOSPHost site have. For a self-hosting site, the users can adjust the internal configurations of the services provided. On

the other hand, for an external hosting site, a fixed set of services is provided with a common configuration. SourceForge is one example of external hosting site. Examples of self-hosting sites are sites that host the project of Linux, Mozilla and Apache. Regardless of the restrictions, externally hosted sites such as SourceForge can be popular as the amount of effort to start host is lower than self-hosting sites.

An impression that the above discussion may create is that all the required development tools are grouped into one FOSPHost site. Having many commonly used tools centralised in a web site is probably a common scenario, but services such as Internet Relay Chat (IRC) may not be provided by a FOSPHost site. One may need to look up the service by other providers. Another possibility can be some developers may also prefer to host some services themselves to increase the amount of control that they can assert.

In this dissertation, sites are always referred as FOSPHost sites. When the word 'FOSPHost' is not used together with the word 'site', it then means the general topic of FOSPHost.

Some readers may expect to find literature of software configuration management in this section. Nevertheless, the approach of this research is exploratory (which will be explained in the methodology section), and the importance of tools is also assumed to be unknown at the start. Literature review on tools will be done after discovering which tools are important in later sections.

3.3 Developing the Research Questions

As discussed above, both the Free/Open Source communities and the business world are probably interested in obtaining benefits from FOSPHost sites. As explained in the rationale of the research (sub-section 1.2), it is likely that the deployment of FOSPHost sites will increase.

As the need for Free/Open Source software development and FOSPHost increase, it is important to look into the design of FOSPHost and discover areas for improvement.

The approach taken in this research is to build an evaluation model for FOSPHost. To evaluate is to 'assess or form an idea of the amount, quality or value of' the matter (Hornby & Crowther 1995, p. 394). By building this model, important issues in FOSPHost will hopefully be discovered and the final model will hopefully be a useful tool to examine the design and deployment of FOSPHost. Also by the examination on the topic of FOSPHost, we may gain more understanding on the Free/Open Source phenomenon as a whole. The overall research question for the study is thus formulated as:

'How to construct an evaluation model for a FOSPHost site?'

In order to answer this question, a divide-and-conquer approach is required. Sub-questions are thus formulated to specify how the investigation is partitioned into smaller parts. From the discussion the previous chapter, specific literature related to FOSPHost need to be identified and a structure is required to categorise and present this literature. This task is summarise in the first research sub-question:

1. What relevant analytical frameworks can be built to facilitate the investigation of the design and deployment of FOSPHost?

After analytical frameworks are obtained, it is possible to collect data from a more focus area relating to FOSPHost. This task is formulated in the second research sub-question:

2. What are the important factors in FOSPHost design and deployment from data collection?

After the important factors are obtained, they need to be presented in some format as an evaluation tool for FOSPHost sites. This task is formulated in the second research sub-question:

3. How to build an evaluation model from these important factors in FOSPHost?

Referring to the objectives in the first chapter, the three sub-questions cover the area of research specified.

3.4 Summary of Chapter Three

In this chapter, the topic of FOSPHost is further explained and more precisely defined. The overall research question and sub-questions are then formulated. The overall research question is:

'How to construct an evaluation model for a FOSPHost site?'

And the research sub-questions are:

1. What relevant analytical frameworks can be built to facilitate the investigation of the design and deployment of FOSPHost?
2. What are the important factors in FOSPHost design and deployment from data collection?
3. How to build an evaluation model from these important factors in FOSPHost?

In the next chapter, the first sub-question will be tackled.

Chapter 4

Development of Analytical Frameworks

4.1 Introduction

In this chapter, the derivation of a new model for the analysis of the Free/Open Source phenomenon will be presented. Other models of explaining the Free/Open Source phenomenon will also be discussed and compared to the new model in order to gain more insight into this phenomenon. The relevance of employing the newly derived models to investigate the topic of FOSPHost will be discussed as well.

4.2 Background for Analytical Frameworks

Recalling the three relevant areas of interest, namely the contextual, technological and socio-economical aspects, the new frameworks proposed in this study – the 4C model and a model of individual participation to a Free/Open Source community - is an attempt to cover all three areas. It is based upon theories on virtual communities and Computer-Supported Co-operative Work (CSCW). The definition of CSCW is '[CSCW is] concerned with the ways in which people work together and with the ways in which computer systems can be designed to support the collaborative aspects of work.' (Rosenberg 1994, p. 1). Therefore, CSCW is related to the technical and socio-psychological aspects of a system. Since software is developed in a collaborative fashion by communication through computer systems in a Free/Open Source community, theories in CSCW are relevant to the examination of Free/Open Source (Yamauchi

et al. 2000). With its bases in both virtual communities and CSCW, the model has the potential to explain both the technical and socio-economical aspects of Free/Open Source. The contextual aspect will also be considered but theories relating to this aspect are rare, so it will be included in the content of the model, not its presuppositions.

4.2.1 Free/Open Source Community, a definition

Before the discussion of the details of the new frameworks, the definition for the term 'Free/Open Source community' need to be established. Nowadays, Free/Open Source projects are usually co-ordinated on the Internet with a group of developers. Therefore, theories in virtual communities could be relevant. The most common definition of virtual communities was given by Rheingold (1993, p. 5): 'social aggregations that emerge from the Net when enough people carry on those public discussions long enough, with sufficient human feeling, to form webs of personal relationships in cyberspace.' A more elaborate model for virtual communities was suggested by Romm, Pliskin and Clarke (Figure 4-1). Their criteria for virtual communities were 'shared goal and ideals; some degree of stability; growth; and loyalty and commitment by their members' (Romm, Pliskin & Clarke 1997, p. 262). Moreover, they identified three important aspects of virtual communities, namely 'variables which affect individuals' decision to join virtual communities', "variables which explain virtual communities' effects on their immediate environment" and 'variables which describe how virtual communities are transforming society' (Romm, Pliskin & Clarke 1997, p. 261).

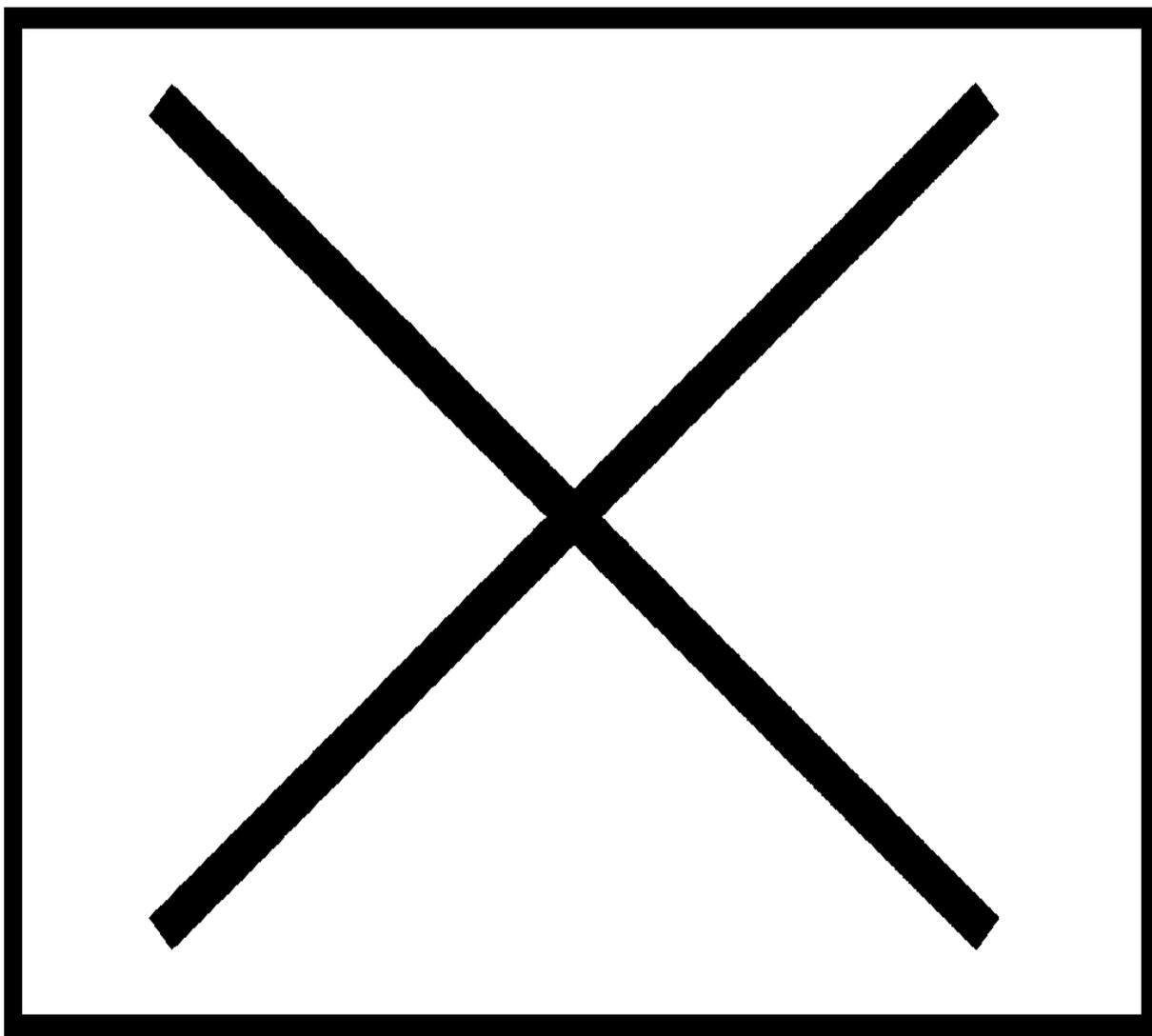


Figure 4-1 Integrative Three Phase Model of Virtual Communities and Society (Romm, Pliskin & Clarke 1997, p. 269)

After surveying different definitions of the term 'virtual communities', we can consider the situation of the Free/Open Source phenomenon and see how some these definitions can be applicable. Within the Free/Open Source movements, there are different sub-cultures. Raymond (2000a) stated that there are different ideologies within communities which support the idea of Free/Open Source. Two most prominent factions are Open Source Initiative vs Free Software Foundation. The difference between the two communities was nicely summarised by (Kelty 2001, p. 312) as 'Whereas FSF would sell freedom if they could, opensource.org sells a better mousetrap, or perhaps 'bug-trap' is the better metaphor.' While the Free Software Foundation was hardline in taking Closed-Source software as morally wrong, Open Source

Initiative is marketing the Open Source software development process as the definite method for software projects. It is not uncommon to find discussions on the differences and resolutions of the two communities in popular Free/Open Source online forums such as Advogato (Advogato 2000a, 2001b). Therefore, according to one of the four criteria stated on a virtual community, 'shared goal and ideals' (Romm, Pliskin & Clarke 1997, p. 262), it is more reasonable to say there are a number of communities within the Free/Open Source movements with different ideals rather than looking at these communities as a monolithic group. A simple definition of a Free/Open Source community can then be a group of developers collaborating mostly through the Internet on similar or related projects attached to a similar culture.

4.2.2 A Framework on Computer-Supported Co-operative Work (CSCW) and Analysis of an Free/Open Source Community

After defining what a Free/Open Source community is, in order to categorise and analyse what happens inside a Free/Open Source community, a framework on CSCW is considered. The framework is shown in Figure 4-2 (Dix 1994, p. 17). In the diagram, the circles with a 'P' denotes a person involved and the circle with an 'A' denotes an artefact(s) involved in CSCW. The persons involved can directly control the artefact and feedback is received from such manoeuvre. It is also possible to obtain information about how another person is controlling the artefact through the artefact itself. This event is called feedthrough and it is denote by a line connecting the two persons via the artefact. In a CSCW system, the persons involved usually are provided a communication media to exchange ideas. The line 'direct communication' denotes this kind of communication. The dotted line deixis represented the content in the direct communication that referred to the artefact. Moreover, the persons involved may also communicate on concepts of a higher level such as the goal of the co-operation. The line 'understanding' denotes this kind of communication.

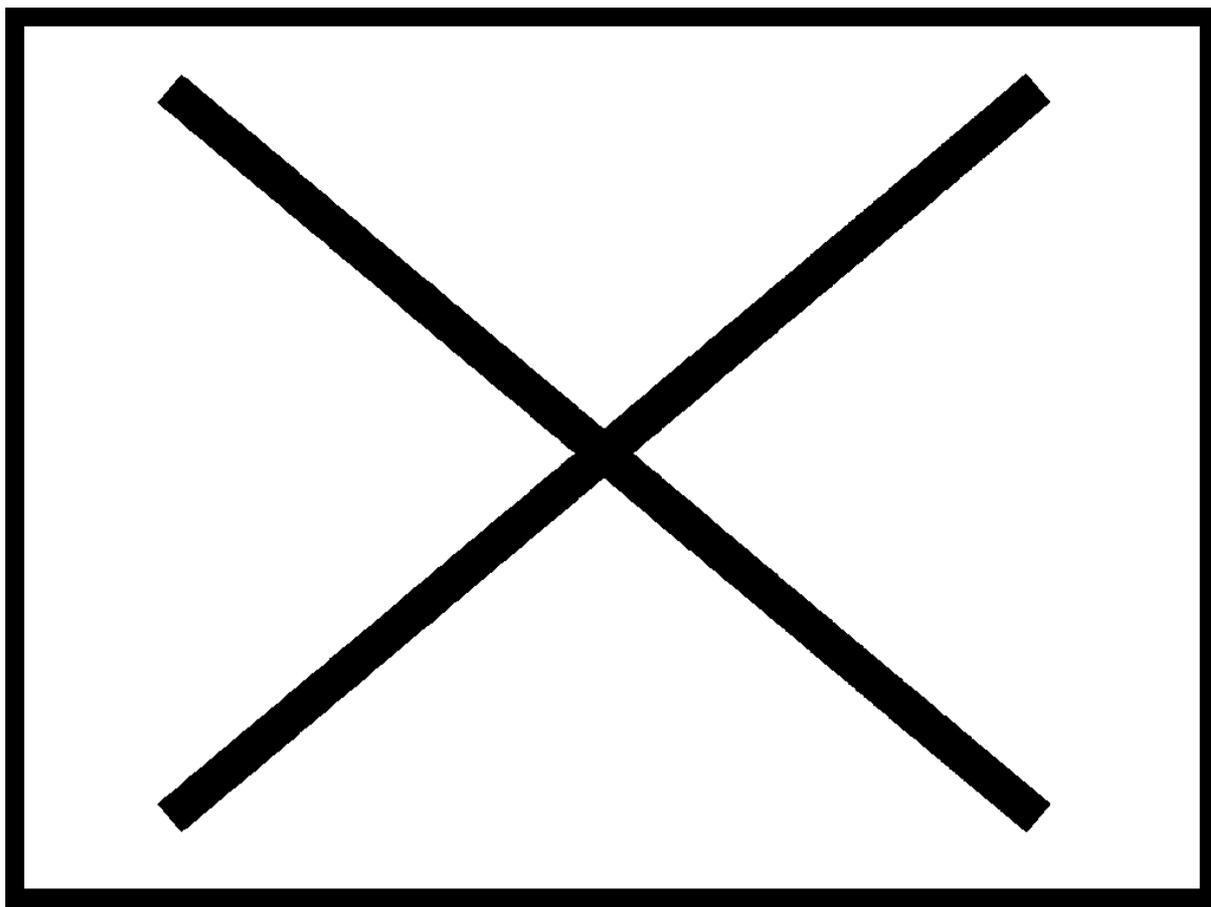


Figure 4-2 A Framework on CSCW

From this framework, important aspects of a Free/Open Source community can be identified. First of all, a Free/Open Source community is based on a communication media. As most of the important artefacts in a Free/Open Source community are information in digital format, these artefacts can also be contained in the communication media. The next important aspect is the artefacts, which are the contributions from the community members. An example of an artefact can be source code. By reading and understanding the source code, one programmer can learn what other programmers are trying to achieve. This is denoted by the feedthrough process. On top of the artefacts, the communication on how to manage the artefacts is also very important. The understanding of co-operation in CSCW is analogous to the culture of a Free/Open Source community, which embodied understanding of high-level concepts such as the goal and the identity of the community.

4.3 4C Model of a Free/Open Source Community

Based on the four important aspects identified in a Free/Open Source community, a model of a Free/Open Source community is built and shown in Figure 4-3. The model is presented in a four-layer (4C) model.

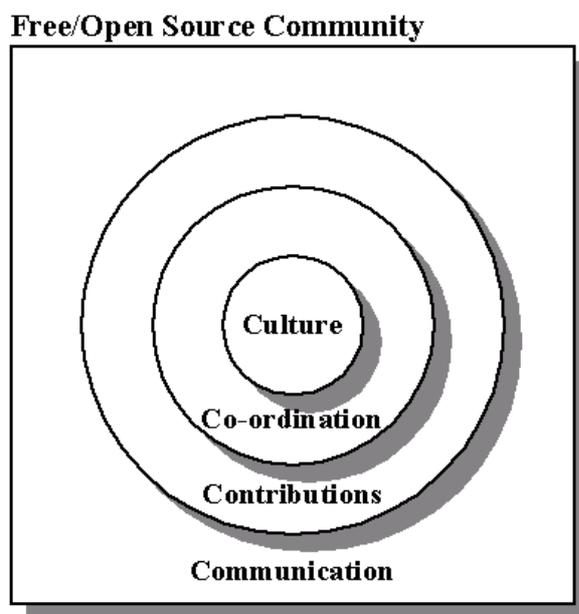


Figure 4-3 4C Model of a Free/Open Source Community

The four layers represented in the model in Figure 4-3 are communication, contributions, co-ordination and culture respectively. The communication medium is the basic infrastructure for any interaction. Contributions referred to the different pieces of assistance given by individual developers via the communication media. Co-ordination is the process of organising fragments of contributions into usable products and the culture of the community in turn governs the rules in co-ordination. These four layers will be explained below in sub-sections 4.3.1 to 4.3.4.

4.3.1 Communication

An important enabling factor for Free/Open Source communities to exist is a medium for communication. In most cases, the Internet is the most frequently used communication medium for Free/Open Source communities. Many (Bezroukov 2000; Moon & Sproull 2000;

Raymond 2000b) recognized the Internet as an important precondition for the Linux project to start. Kollock (1999) suggested that the Internet lowers the cost of collaboration. On the other hand, Ghosh (1998a) used a cooking-pot as a metaphor to describe collaboration on the Internet. In the case of a physical cooking-pot, when everyone put in some ingredients to boil a tasty broth, one can only take a small portion of the broth, more or less the same amount as what one has put in. In the case of the Internet, the digital cooking-pot, which is an efficient cloning machine, everyone who contributes can also get complete copies what others have contributed.

4.3.2 Contributions

A Free/Open Source project is built upon contributions from individual developers. These contributions included source code, suggested features (wish list), comments on project, bug reports and also documentations. Source code is the basis of any program and thus any software project. When a project starts, the existence of an executable program with source code attracts more developers to participate (Fogel 1999; Raymond 2000b). After using the program, developers or users may have suggestions on new features to add to the program. Comments may also be made on the direction of the project as well as the details of the source code. Zawinski (1999) pointed out that the contribution of quality comments could even worth more than source code. Bug reports (sometimes with patches (source code)) are also welcomed to improve the stability of the program. Finally, a program cannot be used and a project cannot be maintained without documentations, and thus contributions to documentation are also important. With a proper communication media, all these contributions can be collected.

4.3.3 Co-ordination

Co-ordination is required to package all these different contributions collected via the communication media into a piece of stable software. A mechanism to accept or reject a piece of contribution has to be established. This mechanism can be understood by studying the social structure of Free/Open Source community for individual rights and responsibilities. This structure can be summarised in a diagram suggested by Lawrie, Arief and Gacek (2002, p. 77)

and modified by the authors in Figure 4-4. The core developers are the most senior group and they had the final say. In the benevolent dictator system (Fogel 1999; Raymond 2000a), a maintainer is that the person who makes final judgements on decisions of the project. If an autocratic system (Fogel 1999; Raymond 2000a) is adopted, a membership system has to be setup to identify between developers and non-developers and it may also involve a voting system for decision-making.

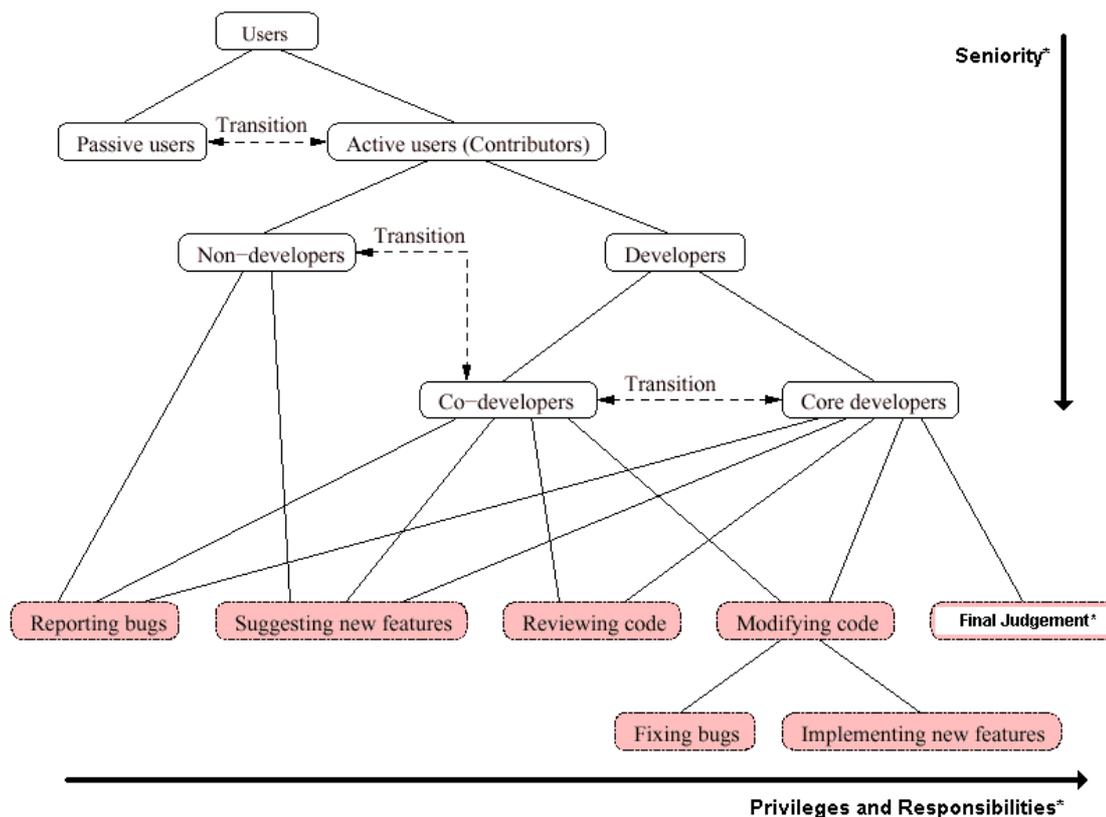


Figure 4-4 A Model of the Social Structure of Free/Open Source Community (Lawrie, Arief & Gacek , 2002, p. 77) modified by the authors (* denotes the modification)

It seems that the core developers are the most powerful class in the structure but it can be argued that all the classes of people in this social structure are inter-dependent and a stable balance of power can be achieved. Users, who seem to be dependent on the developer community for bug fix and implementation of new features, are actually very important to the developers. The popularity of the software is itself a measure of the success of a project (Advogato 2002a) because adoption of a piece of software itself is a compliment. Bugs will be more readily

discovered and the potential of recruiting developers with a larger user base. Therefore, Raymond's advice (2000b) on respecting users is sensible in this social structure. Co-developers and core developers could be argued to be inter-dependent as well. On the one hand, core developers would like contributions from other developers to share the load of development. On the other hand, co-developers can have another parties to carry the burden of co-ordination. If some of the core developers do not listen to the community, other members of the community can take the source code away and run the project separately and this is called forking (Fogel 1999; Raymond 2000a). Due to its disruptive nature, forking does not occur very often but the knowledge of its possibility is yet another force to promote the balance of power.

Further details of the exact sequence of how development is conducted under the social structure outlined above are chosen not to be discussed here. The reason for this decision can be showed firstly from considering the research by Yamauchi et al. (2000), where 552 messages on GCC development mailing list are classified into four groups, namely question, response, proposal and hand in. The probabilities of sequences of these classified messages are presented in Figure 4-5. In the figure, the probabilities of one message type followed by another are stated on the arrows connecting the two messages. Statistical significances of these probabilities are stated below the probabilities (NS denoted Not Significant). The fraction of occurrence of a certain kind of message over total 552 messages is showed inside the circle of the type of message. This diagram illustrates that the actual development process of a Free/Open Source project can be quite chaotic and there may be no exact sequence of processes for discussion. This may suggest that order in a Free/Open Source project can only be found on a more abstract level.

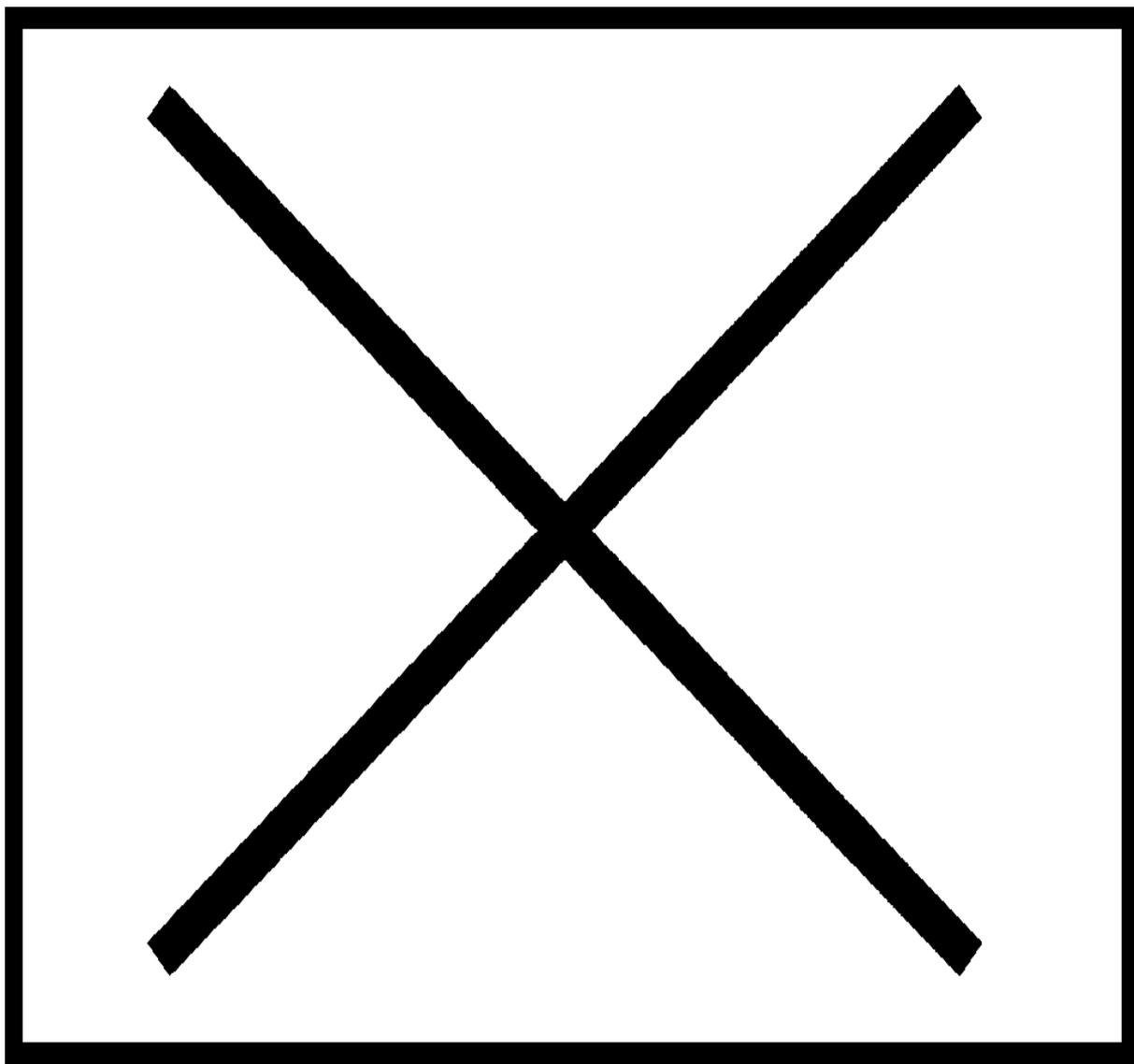


Figure 4-5 Communication Pattern of GCC (Yamauchi et al. 2000, p. 7)

4.3.4 Culture

The culture of a Free/Open Source community shapes the rules in the co-ordination of Free/Open Source projects. Culture is defined as 'the collective programming of the mind which distinguishes the members of one group or category of people from another' (Hofstede 1997, p. 5). The community of Free Software and Open Source movements can be argued to have enough affinity to be called a culture. First of all, most of the members in the community are technical people (Bentson 2000) that value hack (Levy 1984; Turkle 1984) (The word 'hack' in this paper does not refer to breaking into computers. It refers to the ultimate standard of

technical virtuosity and aesthetic in a Free/Open Source community) and technical correctness (Pavlicek 2000). A confessed mistake is more highly valued than a beautifully crafted lie (Pavlicek 2000) as the technical correctness attitude requires admissions of fact. Also with value of hack, Free/Open Source communities also bred humility (Raymond 2000a) as there will always be another person with a brighter idea. Secondly, Linus Torvalds, the original author of Linux, released the source code of the system on the USENET because the culture encouraged sharing (Ghosh 1998b). Thirdly, Raymond (2000a) also observed cultural rules in Free/Open Source communities in the transfers of maintainership and giving credits. Fourthly, being formed mostly by volunteers, the culture endorses loose charter over complicated legalisations when the community tries to put management rules in writing, as volunteers tend to cooperate and reach consensus rather than exploiting the loopholes in the system (Fogel 1999). The above is a general view of the culture and each Free/Open Source community also has its own variations.

4.4 A Model of Individual Participation to a Free/Open Source Community

After introducing a model to a Free/Open Source community, one can consider to represent the relationship of individual participants to the community by a model. Individual participants, who are probably one of the most influential groups on the assessment of FOSPHost, is chosen. Other stakeholders such as user communities, commercial organizations, and the non-commercial organizations that managed Free/Open Source projects (Feller & Fitzgerald 2002) are excluded to limit the scope of investigation.

The model built to explain this relationship is shown in Figure 4-6. The model includes the mentioned 4C model, the motivations and barriers when a developer decides to join a Free/Open Source community together with the positive and negative results after interaction with a Free/Open Source community. The motivations and barriers are analogous to the

"variables which affect individuals' decision to join virtual communities" and the results analogous to the effects from the three phase model on virtual communities (Romm, Pliskin & Clarke 1997). Since the group of individual participants is chosen, all these four factors are related just to them and a feedback loop is included as well.

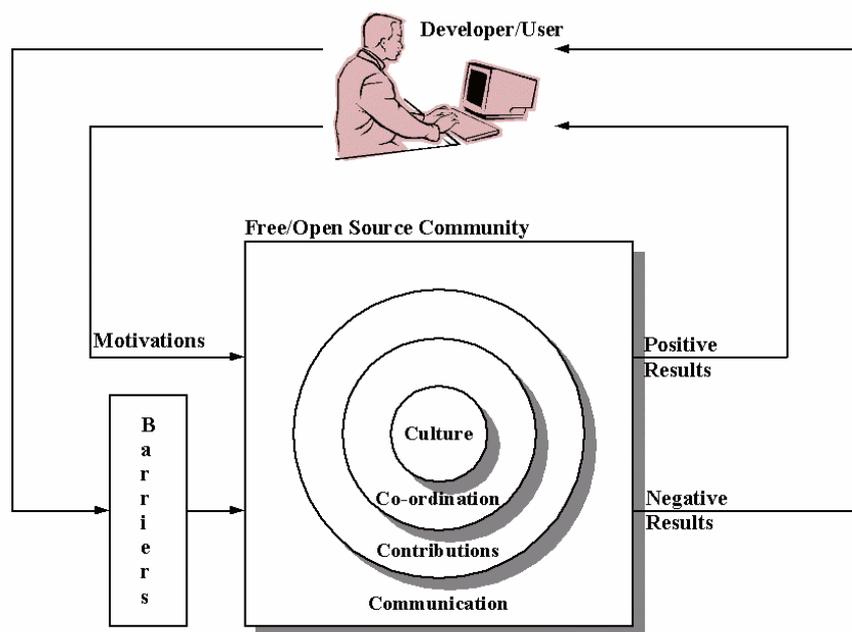


Figure 4-6 A Model on individual participation in an Open Source/Free Software Community

There are a number of motivations for a user or a developer to join a Free/Open Source Software community. An oftenly regarded motivation was stated in Raymond's 'the Cathedral and the Bazaar' - 'Every good work of software starts by scratching a developer's personal itch.' (Raymond 2000b) This essentially means that a developer needs a computer program to do a task for him or her. However, this need does not necessarily lead to joining a Free/Open Source community. For some developers, they may just obtain an executable binary of a piece of software that meets their needs. The most common example is a developer needs a new PC to work so this person installs a copy of Microsoft Windows. Alternatively, a developer may write a piece of software to meet his or her need but the source code of the software may never be shared. Therefore, when a developer joins a Free/Open Source community, he or she may be

motivated by other factors also, such as reciprocal behaviour (Kollock 1999; Wellman & Gulia 1999), reputation (Fogel 1999; Ghosh 1998a; Kollock 1999; Krishnamurthy 2002; Raymond 2000a) and attraction to community (Foster 1998; Kollock 1999). Availability of funding also enables members of Free/Open Source community to work on project devotedly such as support in BSD by DARPA (McKusick 1999) and Linux by University of Helsinki (Bezroukov 2000; Moody 2001). Lastly, altruism or idealism (Kollock 1999) may also motivate developers to contribute.

Although there are a number of motivations for developers to join a Free/Open Source community, barriers also exist to deter them, as in any virtual communities (Romm, Pliskin & Clarke 1997). Technically, Free/Open Source communities only accept developers who attain a high degree of competence (Raymond 2000b). The complexity of source code also created a barrier for contribution (Zawinski 1999). On the other hand, software with poor design and inadequate documentation may deter contribution (mettw 2000). Another barrier is that a developer may not be willing to share his or her own code. Cultural barriers may also exist. Firstly, language can be a barrier because people from certain backgrounds in some part of the world may find it hard to join a Free/Open Source community using English as the common language of communication (Fogel 1999). Cultural mysteries also exist and they have to be solved before a member could be accepted by certain Free/Open Source communities (Raymond 2000a). The last but obvious reason is that a developer cannot afford the time for one's involvement in a Free/Open Source community (Bezroukov 1999a).

There are several positive outcomes as a result of joining a Free/Open Source community. A developer may have one's own itch scratched (Raymond 2000b) and found that he or she enjoyed programming in collaboration (Fogel 1999; Raymond 2000a). He or she may learn

more skills (Fogel, 1999) and build up one's own reputation in the community as well (Fogel 1999; Ghosh 1998a; Kollock 1999; Krishnamurthy 2002; Raymond 2000a).

Negative results from participation in a Free/Open Source community may include a lack of interest on one's project (Fogel 1999; Raymond 2000b), rejection from others (Maclachlan 1999; Pennington), hurts in management issues (Hacker 1999; Raymond 2000a) and burn-out (Bezroukov 1999a, 1999b).

An example of the model can be that a computer literate required a certain application to fulfil her needs. She found a piece of Free/Open Source software (positive result) and added some modifications to fulfil her needs more comprehensively. She then tried to contribute the code back to the community but she found the code had to conform to the coding standard (barrier) and the core members of the project were not too friendly (negative result). Later on, a new version of the software was released with new features but not compatible with her modifications. It was a nuisance that she would need to adjust the modifications for each release. Then, she finally got her code to conform to the standard (motivation). Also, she was no longer new to the community and knew the core members better. Her modification was eventually accepted and it stayed in the code for the versions to come (positive result). The burden of maintenance was therefore shared (positive result).

4.5 Notes on the Construction of the Model of Individual Participation to a Free/Open Source Community

The model of individual participation to a Free/Open Source community presented above was first conceived in late 2000 when there were only a few explanations of the Free/Open Source phenomenon. It was devised as a basis to investigate FOSPHost and the questions asked in the Delphi survey conducted later were directly related to this model. Thus the model is kept as it

was without adding the latest academic findings. However, a comparison with other models from recent publications will be presented in sub-section 4.7.

When this model was designed, it took a less prescriptive and more flexible approach in modelling. Each aspect only has a general description. Moreover, the community's effect to the intermediate environment and global society were also not included in the model. There are evidences that a Free/Open Source community can cause changes in some of these areas. For example, one of the changes to the immediate environment is the change in the use of language. In the case of the Free/Open Source communities, the Jargon File (Raymond 2001), which is a dictionary with a collection of 2321 entries on hacker vocabulary, is a good piece evidence on this aspect. However, some of the impacts of the Free/Open Source communities, such as its impact to the software industry and its contribution to the debate of information freedom, are yet to be examined.

The 4C model had four layers with culture as the highest layer. This may present an impression that culture is the most influential factor. Looking back in the history of hacker culture, by considering the ITS System, a system regarded as the ultimate expression of hacker culture (Levy 1984), one might find some insight into this matter. According to Levy (1984), ITS was a multi-user system but did not has any passwords. Anyone can read and write anything on the system. Users could actually switch to other users' terminal and did programming collaboratively. Seemingly, there was one important factor that was minimised in this system – barrier. Indeed, there was no barrier to stop anyone to program on any code in the system. All source code written could be read and modified as well. This system was built by hackers in Massachusetts Institute of Technology as a rivalry system to CTSS (Compatible Time Sharing System), which was regarded to discourage hacking. This is indeed an example that supports the viewpoint of the influence of culture layer over communication layer. Nevertheless,

McLuhan's famous statement, 'The Media is the Message' (McLuhan 1964, p. 7), is the opposite of the argument above, claiming that the communication layer is more influential (the relevance of the statement to FOSPHost was suggested by Dr. Jason Robbins on 21 Feb 2002 during a visit to Collab.Net). A middle ground argument was proposed by Tuomi (2001) in an examination of the evolution of Linux development that 'In the evolution of complex system of resources and communities, social organization and tools co-evolve.' Therefore, further research will be beneficial in this area.

One of the important advantages as well as a disadvantage with the model is its flexibility. Arguably, the model is flexible enough even to include other non-Free/Open Source community. For example, the model can be used to examine communities that choose to use a Closed-Source license in a commercial environment. Moreover, by substituting contributions, co-ordination and culture by information, channels of communication and pedagogy, the model can be changed to analyse a virtual learning community. By looking at the model as the descendent of the three phase model on virtual communities (Romm, Pliskin & Clarke 1997) and the framework on CSCW (Dix 1994), it is not surprising that this model on Free/Open Source community could be expanded to explain many different systems as its parent models are general models on information systems. One obvious limitation is that there need to be collective agreement on the philosophy of how information should be managed within the system, which is called culture in the model, in order for the model to produce a useful analysis. The advantage of this flexibility is that a Free/Open Source community can be compared with other information systems by a similar framework under this model. The disadvantage is that the model may disappoint those who want to pin down what Free/Open Source really is. Indeed, this model was criticised on this aspect when it was first presented in the Open Source Software Development Workshop at Newcastle upon Tyne, U.K. (So, Thomas & Zadeh 2002). However, it seems that Free/Open Source actually includes a collection of different community structures

and practices (Gacek, Lawrie & Arief 2001; Nakakoji et al. 2002) rather than a few defined methods.

4.6 Comparison Between the Bazaar Model and the Model of Individual Participation to a Free/Open Source Community

The model of individual participation to a Free/Open Source community presented above covered technical and socio-economical aspects of Free/Open Source as well as context of the community. On the other hand, Kely (2000; 2001) pointed out that 'the Cathedral and Bazaar' described the process of how to run a Free/Open Source project as a replica of Linux. This focus unfortunately reduces the phenomenon of Free/Open Source into a series of technical processes. This is, however, not to say that Raymond did not know about culture. On the contrary, he was the compiler of 'The New Hacker's Dictionary' (Raymond 2001). Moreover, in the 'Homesteading the Noosphere' (Raymond 2000a), the next essay after 'The Cathedral and Bazaar', he mentioned various aspects of the different sub-cultures within Open Source. Unfortunately, probably in the process of marketing Free Software and by de-politicisation and renaming it to 'Open Source' (Kely 2000), the complexity of the phenomenon was reduced to technical processes. To conclude, the metaphor of the Cathedral and Bazaar is useful as an introductory, first estimate to the phenomenon of Free/Open Source but more is needed to explain the phenomenon. The model presented above is one of the many attempts to contribute towards a more comprehensive and complex explanation, which covers contextual, technical and socio-economical aspects.

4.7 Comparison Between the Other Models and the Model of Individual Participation to a Free/Open Source Community

Other than the models presented above, researchers around the world also devised different explanations to describe and investigate the Free/Open Source phenomenon. The models to be compared are 'Evolution patterns of Open-Source software systems and communities'

(Nakakoji et al. 2002), 'Open Source characteristics - common and variable' (Gacek, Lawrie & Arief 2001), OSS (Open Source Software) Model (Sharma, Sugumaran & Rajagopalan 2002) and 'A framework analysis of the Open Source software development paradigm' (Feller & Fitzgerald 2002). The focus of the first two models was mainly on the software development process and the latter two were attempts to develop a more comprehensive explanation.

4.7.1 Software Development Based Models

The first model to be introduced is 'Evolution patterns of Open-Source software systems and communities' by Nakakoji et al. (2002). This model was developed after generalising from case studies on four different Free/Open Source projects. The authors proposed that there was a hierarchical community structure starting from passive users, readers, bug reporters, bug fixers, peripheral developers, active developers, core members and project leader. Moreover, there are three types of Free/Open projects, namely exploration-oriented, utility-oriented and service-oriented and each type had different attributes (Table 4-1).

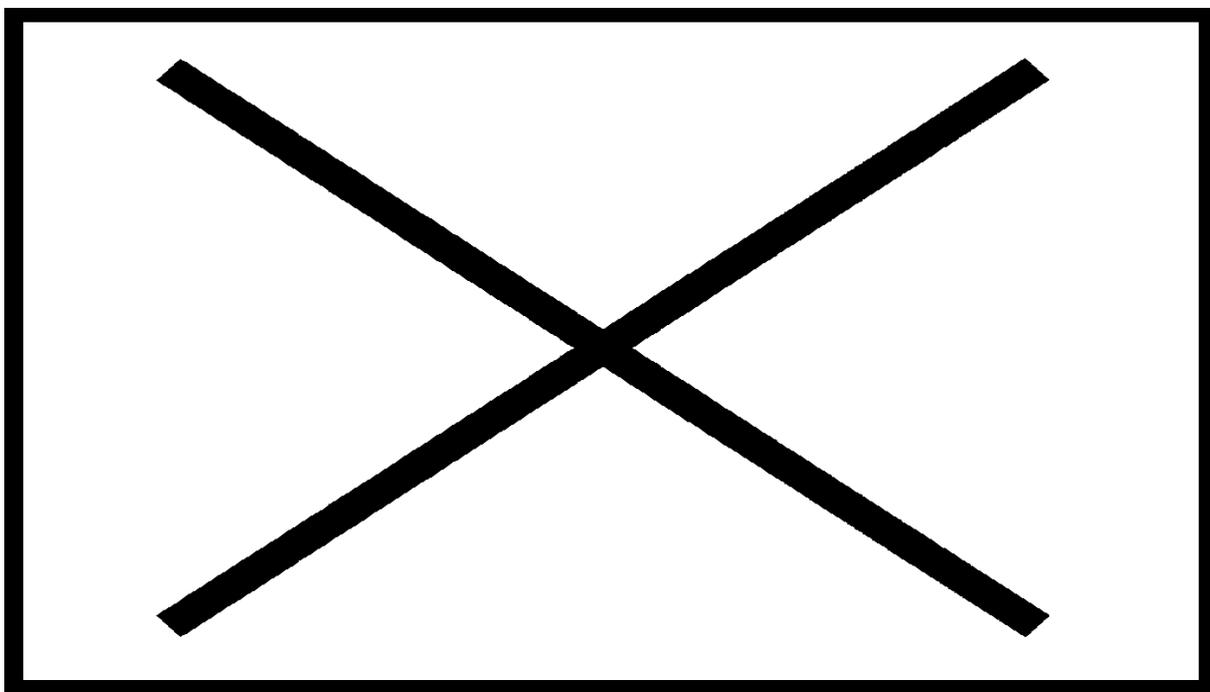


Table 4-1 Three Types of Free/Open Source Projects (Nakakoji et al. 2002)

The three types of Free/Open Source projects could also evolve into another type in different development stages. When there are new ideas to be implemented, the project may evolve into exploration-oriented type. When there were new needs to be satisfied, the project may evolve into utility-oriented type. When the project become mature, it may evolve into service-oriented type.

From the model, there is not just one approach in Free/Open Source software development but three approaches. These approaches did not just affect the process of the development but also the community structure. For example, Cathedral-like central control structure was found in exploration-oriented type projects and Bazaar-like decentralized control in utility-oriented projects. Comparing with the model of individual participation to a Free/Open Source community, this model belongs to the co-ordination layer of the 4C model with brief mentions of issues in culture and barriers. Indeed, this model has a more specific description over the description in co-ordination layer above.

The next model to be introduced is 'Open source characteristics - common and variable' by Gacek, Lawrie & Arief (2001) (Figure 4-7). They proposed that there were common attributes among Free/Open Source projects such as Open Source Definition, community, motivation, developers are users, process of accepting submissions, development improvement cycles and modularity of code. There were also variable attributes that changed from project to project, namely choice of work area, balance of centralisation and decentralisation, meritocratic culture, business model, decision making process, submission information dissemination process, project starting points, visibility of software architecture, documentation and testing, licensing, operational support and size.

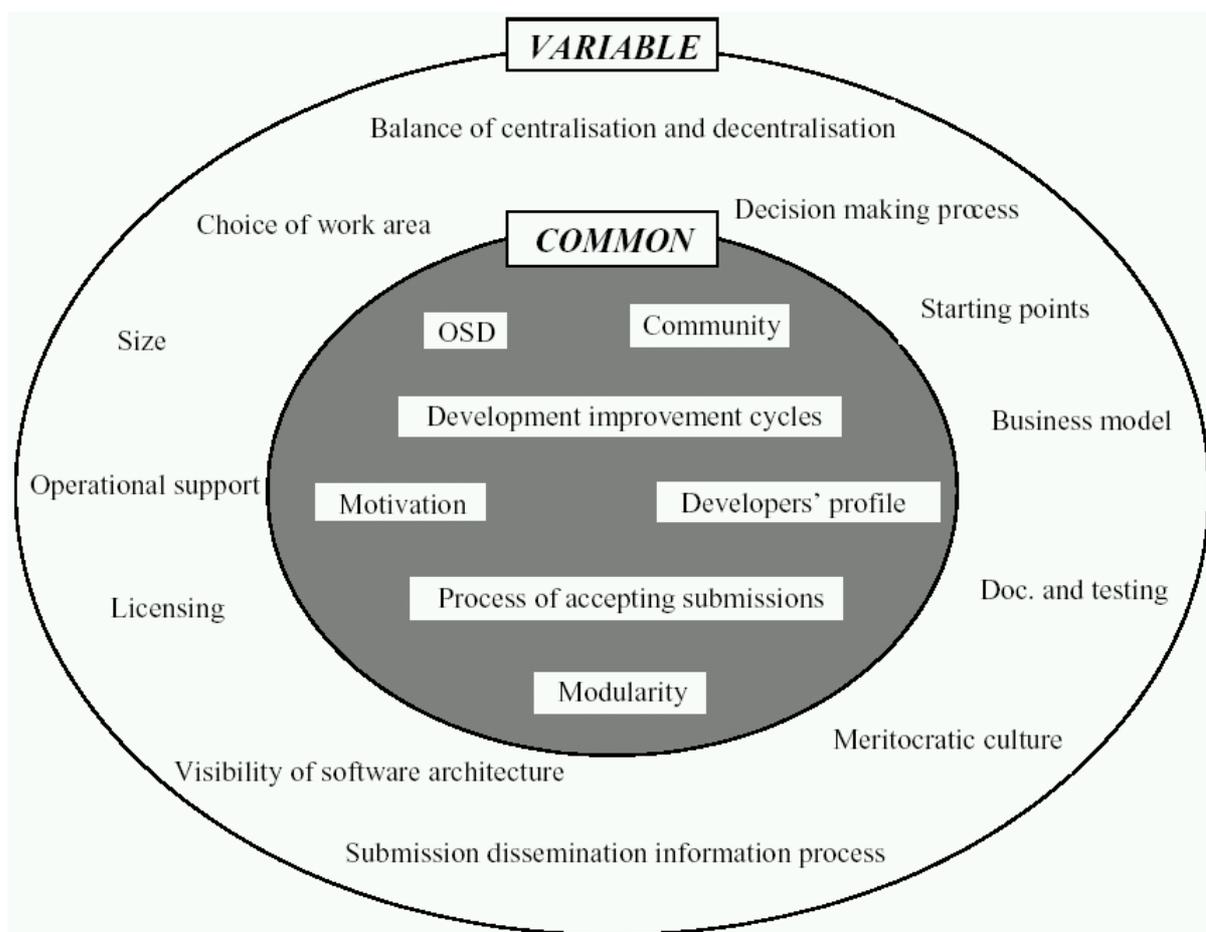


Figure 4-7 Open source characteristics - common and variable (Gacek, Lawrie & Arief 2001, p. 79)

Comparing with the model of individual participation to a Free/Open Source community, this model mainly belongs to the co-ordination layer of the 4C model with brief mentions of issues in communication, culture and motivation. Similar to last model, it has a more specific description over the description in co-ordination layer.

4.7.2 Comprehensive Models

The first model in this sub-section to be introduced is the OSS Model by Sharma, Sugumaran & Rajagopalan (2002). They needed to devise a model of the Free/Open Source phenomenon in order to postulate how traditional software development environment could fuse with Free/Open Source environment to create a hybrid-OSS community and obtain benefits from both methodologies. The benefits that the authors hoped to obtain were reduction in development time and time-to-market, improvement in quality, reduction of cost, gaining developer loyalty and increase developer talent pool without additional head count and

overhead. Derived from organisational theory literature, the authors proposed that there were three aspects in Free/Open Source phenomenon, namely structure, culture and process. Moreover, these three aspects also interacted with each other (Figure 4-8).

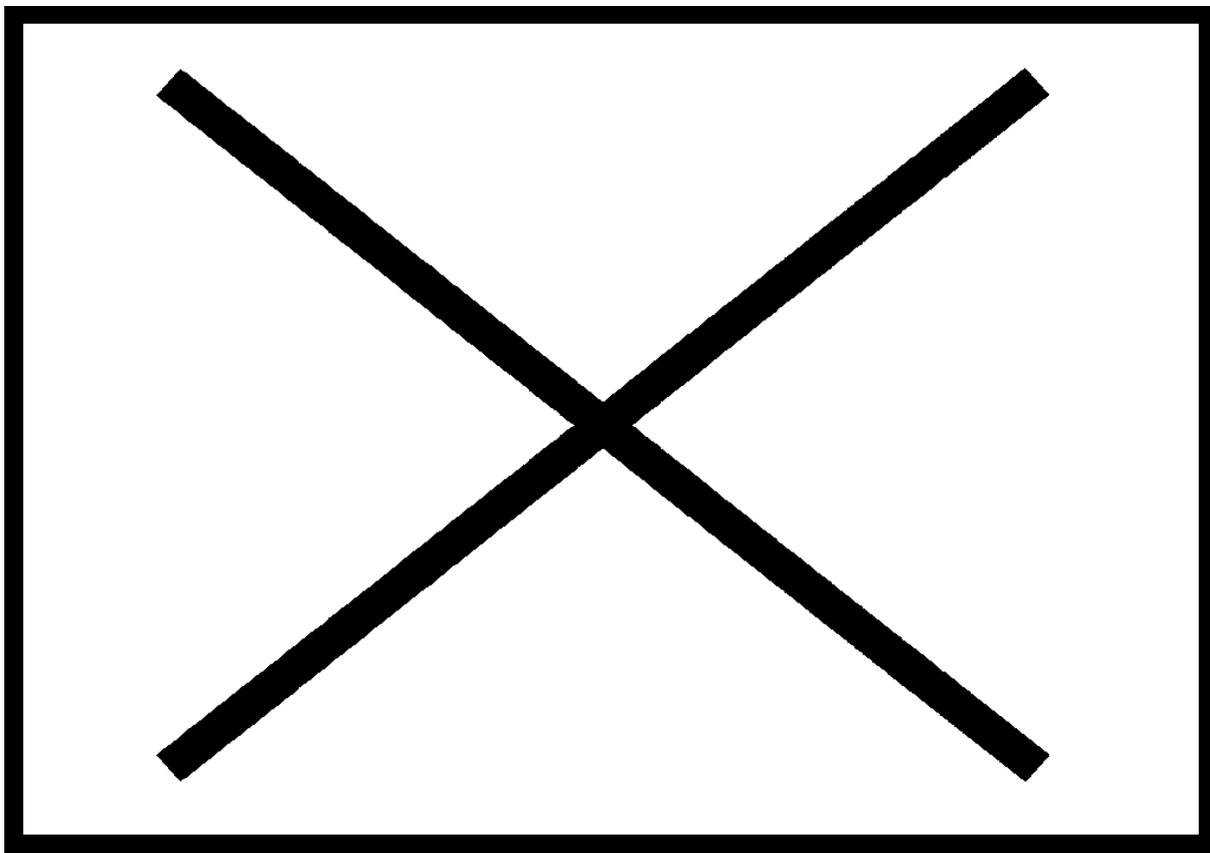


Figure 4-8 OSS Model (Sharma, Sugumaran & Rajagopalan 2002, p. 18)

Comparing with the model of individual participation to a Free/Open Source community, the OSS model covers co-ordination, culture and positive results. There were also brief mentions of version control system, which belongs to the communication layer. Motivations of developers and barriers in creating a hybrid community were also discussed. Nevertheless, the authors seemed to take less care in handling the issue of flexibility. For example, for motivations, the authors emphasized on altruism and ideology. According to the BCG survey (Lakhani et al. 2003), these motivations were only one of the four types of important motivations and a significant number of contributors in the survey were paid to program in Free/Open Source software. Also, without mentioning negative results, the Free/Open Source phenomenon portrayed was rosier than reality. For example, trust and loyalty were mentioned

without balancing the picture with hurts in management issues (Hacker 1999; Raymond 2000a) and flame wars (Jacobito 2001; Pennington).

The last model to be presented is an elaborate model, 'A framework analysis of the Open Source software development paradigm' by Feller & Fitzgerald (2002). This framework was derived from Zachman's IS (Information Systems) architecture framework (Sowa & Zachman 1992; Zachman 1987) and Checkland's CATWOE (Clients, Actors, Transformation, Weltanschauung (World-view), Owner, Environment) framework (Checkland 1981). Five aspects were proposed, namely qualification, transformation, stakeholders and environment and world-view. In the qualification aspect, the authors argued that Open Source Definition (Open Source Initiative 2003a) was a necessary and sufficient definition. In the transformation aspect, which means the process of Free/Open Source software development, the authors suggested that seven characteristics that existed in most projects such as peer review and prompt feedback. Then the authors commented on the taboos and norms of Free/Open Source communities. Lastly, the lifecycle for Free/Open Source software development was included, which was mostly adopted from a case study in FreeBSD (Jorgensen 2001). The stages in the lifecycle included code, review, pre-commit test, development release, parallel debugging and production release. In the stakeholders' aspect, four bodies were considered, including developer communities, user communities, commercial organizations, and the non-commercial organizations that managed Free/Open Source projects. Lastly, in the environment and world-view aspect, three categories of motivation were considered, namely technological, economic and socio-political.

Comparing with the model of individual participation to a Free/Open Source community, this model covers communication, co-ordination, culture, motivation and positive results. There was also a brief mention on barriers such as promotion from 'Developers' statue to 'Additional Contributors' required a test in CVS skills but negative factors were not the focus. Moreover, in

the explanation of the lifecycle for Free/Open Source software development, the FreeBSD development lifecycle was taken as a prime example. FreeBSD was known to be one of the more organised projects (Fuller 2004) and it will be beneficial to have other more chaotic approaches mentioned. Taking Nakakoji et al. (2002) as an example, there could be as least three types of projects and thus different approaches in development.

4.7.3 Comparison of the Models

After presenting the four models, the result of the comparison is tabulated in Table 4-2.

	Commu- nication	Contri- bution	Co-ordi- nation	Culture	Moti- vation	Barriers	Positive Results	Negative Results
Nakakoji et al. 2002			✓	✓		✓		
Gacek, Lawrie & Arief 2001	✓		✓	✓	✓			
Sharma, Sugumaran & Rajagopalan 2002	✓		✓	✓	✓	✓	✓	
Feller & Fitzgerald 2002	✓		✓	✓	✓	✓	✓	

Table 4-2 Comparison of the Four Models based on the Model on Individual Participation in an Open Source/Free Software Community

Most of the explanations from the four models are more elaborate than the model of individual participation to a Free/Open Source community. For example, the motivation categories proposed by Feller & Fitzgerald (2002) were far more sophisticated. There are also areas that are not included in the model of individual participation to a Free/Open Source community such as qualification by the Open Source Definition and stakeholders such as commercial organizations. Nevertheless, the model of individual participation to a Free/Open Source community is yet flexible enough to incorporate most of the materials in the four models (Table

4-2). Moreover, less discussed areas such as contributions, barriers and positive and negative results are also included. Also, though most of the content in the four models were based on actual facts, some of the facts might only reflect particulars of certain Free/Open Source communities. In contrast, by being less prescriptive, the model of individual participation to a Free/Open Source community may have the advantage of allowing its users to discover alternatives.

Recalling the aim of creating the model of individual participation to a Free/Open Source community is to identify important aspects in a FOSPHost site for further investigation. This aim can be regarded as completed since the model of individual participation to a Free/Open Source community includes most of the important aspects that the four models discussed. Moreover, it also includes other significant issues that the four models have less emphasis on. Furthermore, the omission of stakeholders other than developers is favourable as to narrow down the scope of investigation to the most important group of stakeholders (this omission and other limitations in the model were also documented in sub-section 4.5 above).

After the comparing the advantages and disadvantages of the four models and the model of individual participation to a Free/Open Source community and reviewing how suitable it is for the investigation, other observations can be discussed. From the analysis above, for comprehensive models, social theories are employed as a basis to derive explanations. Even for software development based models, discussions on social issues on Free/Open Source are included. This probably suggests the importance of the social aspect the discussion of the Free/Open Source phenomenon.

According to Table 4-2, contributions is one of the least discussed topic within the 4Cs. The obvious reason is that many regards contributions to be coding for Free/Open Source software.

Indeed, the 'show me the code' culture was strong (Raymond 2000b; Yamauchi et al. 2000). Nevertheless, Lakhani & Hippel (2003) found user-support as a significant type of contributions and thus conducted a study in Apache mailing-list on the responds of request for user assistance. Gabriel (2002) suggested that other contributions such as marketing and standards development were also notable. He further commented that hierarchical analysis of Free/Open Source communities based on authority on code (such as Figure 4-4) could be misleading. The code development community is just one of the many communities within the Free/Open Source phenomenon and the boundary of a community should be defined by these different kinds of contributions or interests in order to represent their significance.

Feller & Fitzgerald (2002) suggested that 'Is OSS truly successful?' is a question yet to be answered. It is then not surprising that the effects of Free/Open Source are less discussed in the models mentioned. In the model of individual participation to a Free/Open Source community, only the effects affecting individuals are mentioned. Also, negative factors such as barriers and negative effects of Free/Open Source are less discussed. Therefore, research in these areas will yield new knowledge.

From the models presented above, Nakakoji et al. (2002) pointed out that Free/Open Source projects with different co-ordination models possessed a number of different attributes. Gacek, Lawrie & Arief (2001) also showed that there were variables between different projects. Moreover, Feller & Fitzgerald (2002) also claimed that there were different practices in different organizations and developers. Indeed, flexibility was also an important consideration in the design of the model of individual participation to a Free/Open Source community. May be this collection of differences and variables are where the chaos of Free/Open Source lies. Therefore, further research on these variables will be profitable.

To conclude, after comparing the model of individual participation to a Free/Open Source community with four other models, the quality of the model is acceptable as the basis for this research.

4.8 The Model of Individual Participation to a Free/Open Source Community and FOSPHost Design and Deployment

After the development of the analytical frameworks, namely the 4C model and the model of individual participation in a Free/Open Source community, how do these models relate to the investigation of FOSPHost?

Recalling that 4C model of a Free/Open Source community consisted of communication, contributions, co-ordination and culture (Figure 4-3), a FOSPHost site is the communication tool that holds the contributions of the community. A FOSPHost site indeed creates a basis for the existent of a community. Moreover, the model of individual participation in a Free/Open Source community suggests that the important issues in improving a FOSPHost site are how well does a FOSPHost site support collection of contributions, co-ordinations of project(s) and cultivate a constructive culture for community. Other important issues include how the design of FOSPHost motivates users to participate and maximises positive results. On the other hand, barriers of participation should be lowered and negative results should also be minimised.

From the derivation above, the models thus suggested distinct focuses on how the study should proceed. The issues obtained above will be the starting point for the data collection stage.

4.9 Summary to Chapter Four

In this chapter, models for the analysis of the Free/Open Source phenomenon are examined and the model of individual participation to a Free/Open Source community is chosen as the

theoretical basis for the investigation of FOSPHost. Though this model is less specific than other models, it is comprehensive and flexible enough for the purpose of this research.

In the next chapter, methodologies and methods for collecting data from experts and the Free/Open Source communities will be presented and the basis for the construction an evaluation model for FOSPHost sites will be explained as well.

Chapter 5

Methodology

5.1 Introduction

In this chapter, the overall research strategy and research plan will be introduced. The rationale behind the selection of methodologies will also be presented. Literature on evaluation will be reviewed and a suitable classification for software evaluation will be devised to suit the nature Free/Open Source software. In terms of data collection and procedures, methods on conducting a Delphi survey on FOSPHost sites and a detailed investigation in external hosting sites will be discussed.

5.2 Overall Research Strategy

The overall research strategy consists of a Delphi survey, a detailed investigation in external hosting sites and finally the construction of an evaluation model for FOSPHost. An exploratory approach was taken in this research. Moreover, the conclusion of this research will be constructed from the empirical data collected, and thus an inductive approach was also adopted.

One way to classify social research is by the purpose of study. There are mainly three types of purposes, namely exploration, description and explanation (Babbie 2002; Neuman, Bondy & Knight 2003). Exploratory studies are conducted to learn more about topics that are little known to construct mental pictures based on basic facts and stakeholders. Descriptive studies

are conducted to observe and describe details of social phenomena. Explanation studies are conducted to verify certain theory on the relationships of different variables in a system.

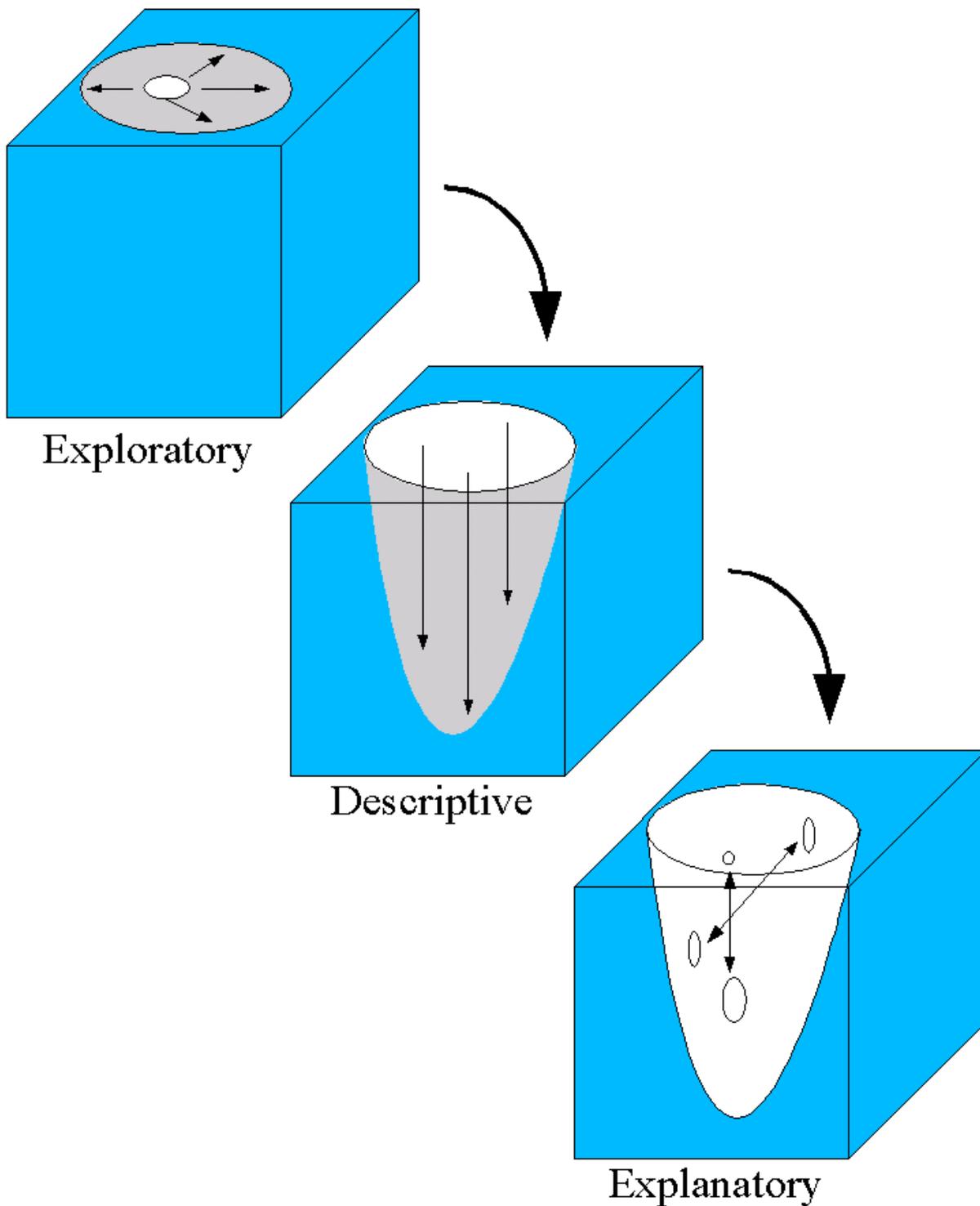


Figure 5-1 Exploratory, Descriptive and Explanatory Research

Neuman, Bondy & Knight (2003) suggested that when a new topic is studied, the sequence for three types of study to be executed would be exploration, description and finally explanation. In Figure 5-1 this concept is illustrated. The knowledge that is unknown is denoted as a cube in blue (grey in print), the area where it is known is denoted by white. For exploratory research, it is like increasing the white area of known knowledge on the surface of the cube. In descriptive research, it is to increase the depth of known knowledge, based on the results from previous exploratory research. Explanation studies are done last as substantial understanding of the topic was required before formulating theories about the topic. This illustration in one sense is not totally accurate as the results from each type of study are probably not mutually exclusive. For example, during an exploratory research, the result probably will have some depth. Casual relationships of elements within the topic may already be partly confirmed. Moreover, there is always more to discover even on a well-known topic, so using the idea of using white to denote known knowledge in a certain area can be misleading. Nevertheless, it may help to understand the underlying principle of purposes of research.

As mentioned above, when this research began in 2000, the amount of literature on the Free/Open Source phenomenon was not sufficient to form a comprehensive explanation. Therefore, an exploratory approach was adopted. In exploratory research, Neuman, Bondy & Knight (2003, p. 30) suggested that the researcher 'must be creative, open minded, and flexible; adopt an investigative stance; and explore all sources of information.' The disadvantages of exploration studies are the conclusion yielded may not be definitive and the representativeness of result may be weaker (Babbie 2002).

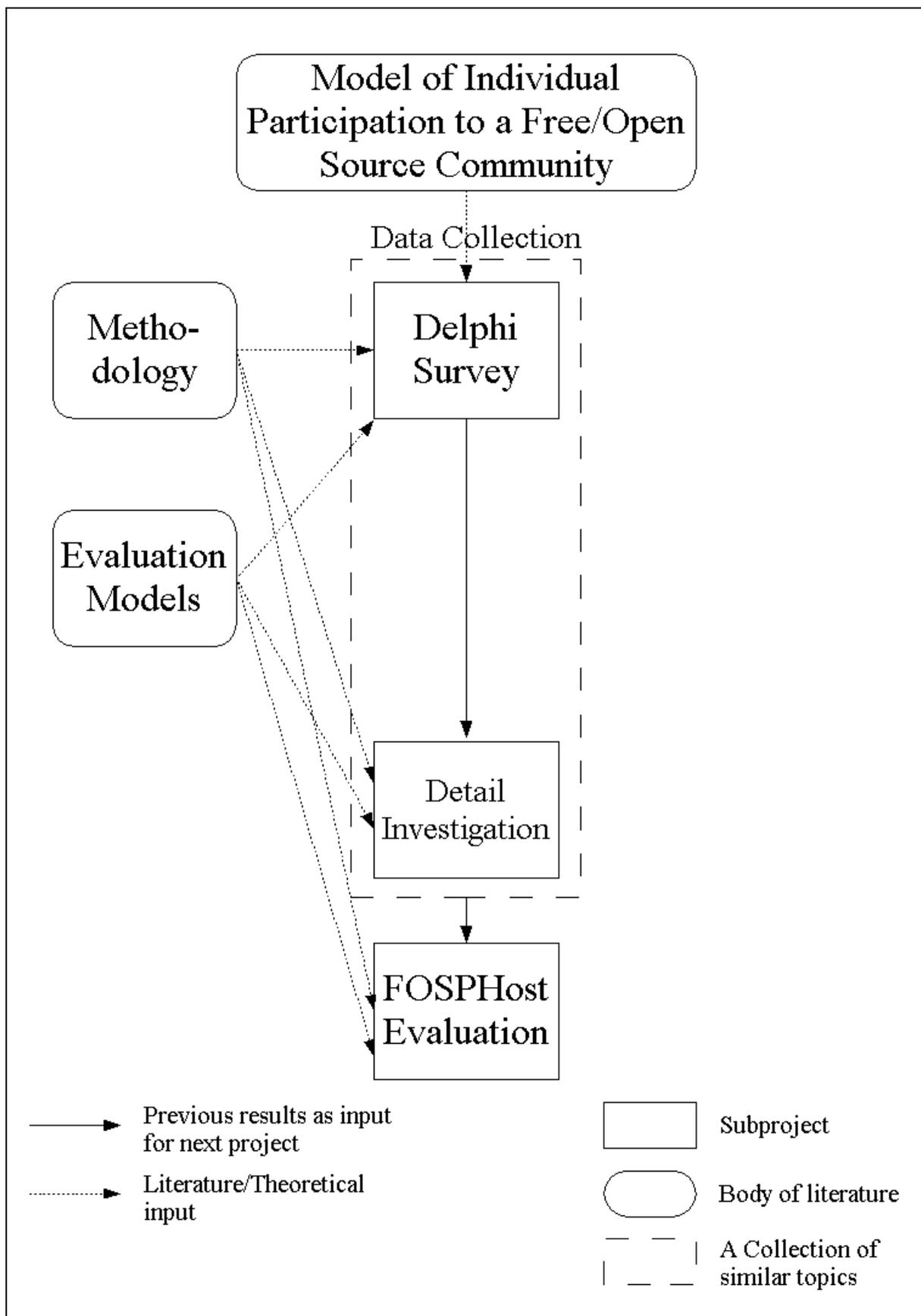


Figure 5-2 The overall research strategy of this research

Another choice of the research strategy in this study was between deductive and inductive approach. In a deductive approach, a hypothesis is formulated from pre-existing theoretical framework and empirical data is collected to prove or disprove this hypothesis. In an inductive approach, empirical data is collected to build a theoretical framework based on a few initial concepts (Neuman, Bondy & Knight 2003). Inductive approach was the obvious choice because the amount of pre-existing theoretical framework was not sufficient.

The overall research strategy is illustrated in Figure 5-2, including the relationships between literature and data collections. The initial Delphi survey employed the model of individual participation to a Free/Open Source community as the theoretical basis for the initial questions in the first round of the survey. After the survey, a detailed investigation was done to further collect data on different FOSPHost sites. The literature of methodology and evaluation was referred to in each of the three steps of sub-projects to ensure consistence.

5.3 Selection of Research Methodologies and Methods

In this section, the rationale for choosing an appropriate methodology will be presented and then the choice of research method for each phase of the research will be explained. The word 'method' of research is defined as 'the actual techniques or procedures used to gather and analyse data related to some research question or hypothesis.' (Blaikie 1993, p. 7) In contrast, methodology is a more philosophical 'analysis of how research should or does proceed' (Blaikie 1993, p. 7).

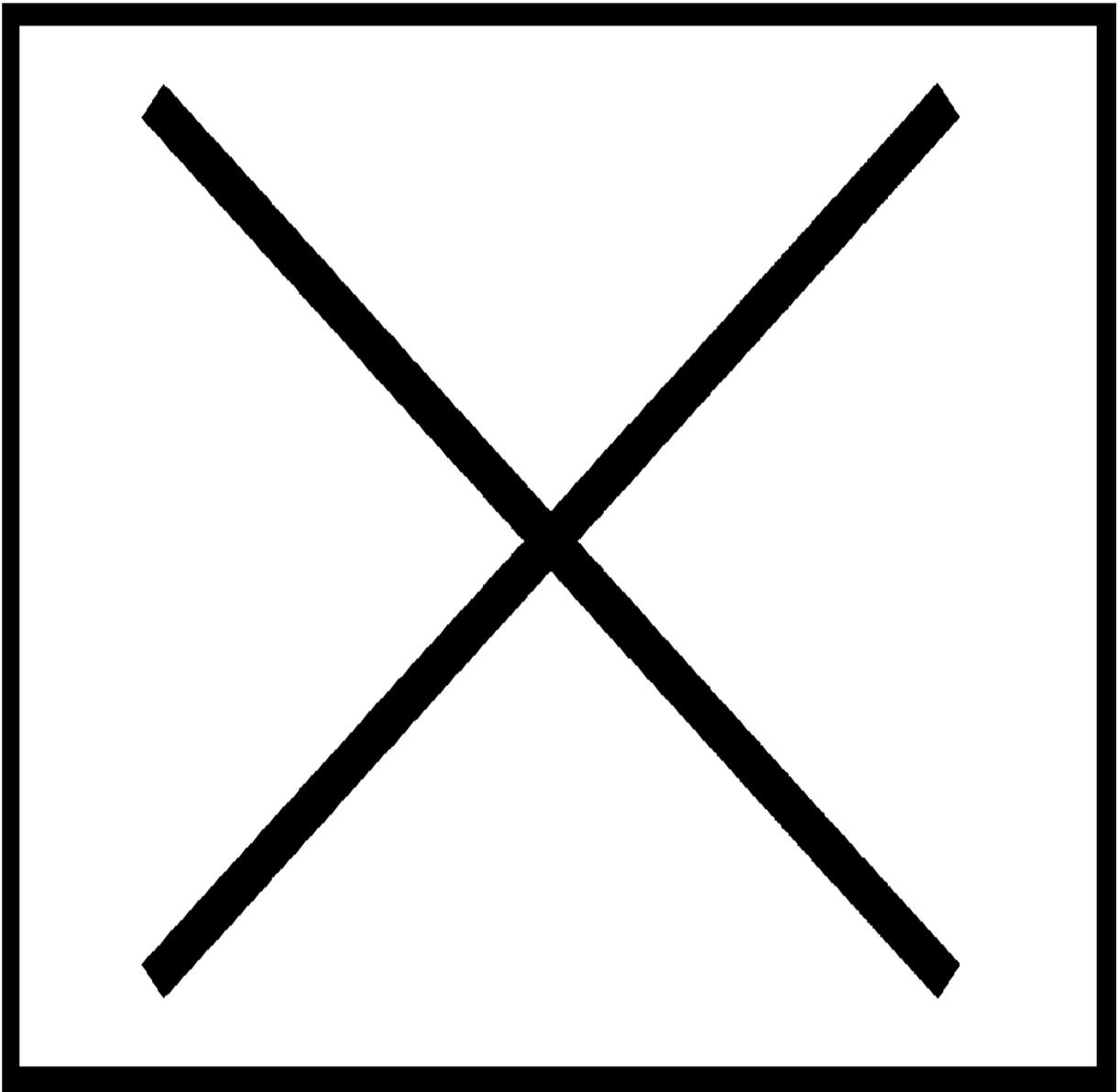


Table 5-1 A Summary of Differences among the Three Approaches to Research (Neuman, Bondy & Knight 2003, p. 91)

Neuman, Bondy & Knight (2003) proposed that there were three major methodologies and they are compared in Table 5-1. (Feminist and postmodern research methodologies were also mentioned in Neuman, Bondy & Knight (2003) but they were regarded as embryonic. Therefore they are omitted here.) All these three methodologies can be relevant to research in

the Free/Open Source phenomenon. As the Free/Open Source phenomenon is related to software engineering, and software engineering has its roots in mathematics and science and thus positivism is actually a pre-dominant methodology in Free/Open Source research. Examples of researches employing positivism are surveys on source code (Dempsey et al. 2002; Koch & Schneider 2002; Stamelos et al. 2002) and analysis of statistics from FOSPHost sites (Hunt & Johnson 2002; Kienzle 2001; Krishnamurthy 2002). On the other hand, some researchers hoped that interpretive approach could provide a more meaningful description to the chaotic Free/Open Source phenomenon. A discussion in the workshop on 'Advancing the Research Agenda on Free / Open Source Software' (Ghosh 2002) suggested that one of the methodological directions could be anthropological or even ethnographic in order to gain more insight in the organization of Free/Open Source software development. Ethnographic studies exist but the numbers are few (Arief et al. 2002; Scacchi 2002). Lastly, though there are very few researches employing the critical social science approach, it will be interesting to see what insight can a theory of classifying society by the degree of software freedom each class possesses and how the oppressed can be empowered by Free Software.

As argued above, all three major methodologies could probably yield interesting results. In this research, however, positivism is chosen, as it is the methodology that the majority of the audience is familiar with. The ontology (or world view) of positivism is that general laws, which are the fundamental operating principles of the world, exist and they are objectively observable (Neuman, Bondy & Knight 2003). Though each observation of the world is atomic, they are discrete and independent of each other. Conclusions can be drawn from them to discover the basic principle of the world (Blaikie 1993). Truth can thus be found on observations, not unexamined belief or metaphysics (Britannica.com 2000). This methodology is also consistent with the inductive strategy that is employed in this research (Blaikie 1993).

One interesting aspect of positivism is that if each observation of the outside world by a subjective human being is regarded as subjective, then how can an objective conclusion be drawn? Objective results can be obtained by drawing conclusions on common patterns from subjective observations collected in a scientific manner (Babbie 2002), and this is the main methodological philosophy of this research. Obviously, the validity in seeking the truth by employing this view of objectivity can be critiqued (Babbie 2002; Blaikie 1993; Neuman, Bondy & Knight 2003), but an in-depth debate is beyond the scope of this research.

Though it is not the author's intention to go into methodological debates, but one of the objections to positivism is important enough to be discussed here - relevancy. Neuman, Bondy & Knight (2003, p. 71) claimed that there could a danger that 'positivism reduces people to numbers and that its concerns with abstract laws or formulas are not relevant to the actual lives of real people.' As the author would like the final evaluation model to be relevant and useful to the general public, a more lenient approach from the orthodox positivism worldview will be taken when required. Lee (1991) suggested that though positivist and interpretive approaches were usually views as irreconcilable and incompatible approaches, it was possible to integrate them and reaped the benefits from both methodologies. As mentioned above that exploratory research required flexibility to construct a richer picture of the situation, interpretive approach will be used when needed to construct meaning to increase relevancy.

Before presenting the methods used in this research, the definition of evaluation and different evaluation approaches will be introduced first.

5.3.1 Considerations in the Construction of the FOSPHost Evaluation Model

5.3.1.1 Introduction

Different issues are required to be taken into account when building an evaluation model and they will be presented in this sub-section. A unified classification on software evaluation will be developed and its possible contributions to the evaluation of Free/Open Source software will be reasoned. Principles of choosing a suitable format for the evaluation model will also be discussed. Finally, the expectations of the users of the evaluation model are proposed and strategies to meet their expectations are discussed.

In terms of software evaluation methods, they can be classified by the stage in which the evaluation is performed. One category of evaluations is applied during the development of the software and the other category of evaluations is applied after the completion of the software products. The aim for the evaluations during development are usually for improvement of the software to meet the requirement laid with the developers (or commonly known as formative evaluation (Wadsworth 1997)). The aim for evaluations of finished software products (e.g. Commercial-Off-The-Shelf (COTS) software) is usually to inform a decision of purchase by the users (or commonly known as summative evaluation (Wadsworth 1997)). Details of evaluation methods in these two categories will be discussed below and the reader can probably see the difference in emphasis in these two methods as they are usually employed by two different groups of people.

An alternative software evaluation classification can be proposed. This classification consists of four categories based on different areas of concern, namely, intrinsic, utility, usability and context. Intrinsic software evaluation methods are examinations on software engineering process and code quality. Utility evaluation methods are assessments on functionalities of the

software (Grudin 1992). Usability is the quality of the interface design that affects learnability, efficiency, memorability, error rate, error severity and satisfaction of the software from the users' viewpoint (Nielsen 1993). This is a relatively narrow definition of usability (Bevan 1995) and the wider issues in the environment of the usage of the software is classified under the category of context. As proposed by Bevan (1995), it is important not just to investigate the quality of the interface but whether the software fit the quality of use in its own context. This context may include specifics of users, tasks and socio-organisational environments. The presentation of four categories above are arranged according to the distance of each category from software development team, with intrinsic the closest and context the furthest.

5.3.1.2 Software Evaluation During Development

Within the category of evaluation during development, it is common to find three out of the four areas of software evaluation, namely, intrinsic, utility and usability. Utility and usability evaluation will be explained first and then intrinsic due the familiarity of the former two. Context evaluation will be explained last.

Utility evaluation during development is usually found in the activity of verification and validation. Verification and validation are actually two different but related processes to evaluate the quality, performance and reliability of software systems (Lewis 1992). The objectives of verification and validation were given in IEEE standard 1012-1998 (Software Engineering Standards Committee of the IEEE Computer Society 1998a, p. 2) as:

The verification process provides supporting evidence that the software and its associated products

- 1) Comply with requirements (e.g., for correctness, completeness, consistency, accuracy) for all life cycle activities during each life cycle process (acquisition, supply, development, operation, and maintenance);
- 2) Satisfy standards, practices, and conventions during life cycle processes; and

3) Establish a basis for assessing the completion of each life cycle activity and for initiating other life cycle activities.

The validation process provides supporting evidence that the software satisfies system requirements allocated to software, and solves the right problem (e.g., correctly models physical laws, or implements system business rules).

Moreover, verification and validation can be carried out internally or externally by an independent body outside of the development team. For early computer systems, to perform to the required standard was one of the main concerns and the techniques of verification and validation were developed early on. In early 1970s, the U.S. Army already employed external, independent verification and validation on critical military systems (Lewis 1992).

A typical verification and validation may include processes such as requirement verification, design verification, code verification and validation, which correspond to different stages in software development (Lewis 1992). Obviously, many techniques are involved in all these procedures and it is beyond the scope of this dissertation to explain them in details, but some of the most commonly used techniques are whitebox and blackbox testing during the code verification. Whitebox testing means that the tester can see and use the control structure coded in the software for testing. Examples of techniques include testing of data flow and control structures such as loops and conditions (Pressman 1997). On the other hand, test cases of blackbox testing are derived from the functional requirement, rather than the knowledge of the inside working of the software. Another dimension of testing is the size of the software tested. For large systems, unit tests are done to each module (Pressman 1997). Integration test will then be performed during each level of integration of these modules (Pressman 1997) (Both unit test and integration test is a part of verification). After the software has been successfully assembled, validation test will be conducted. It is basically a number of blackbox tests to

validate the behaviour of the system against the requirement. The focus of verification and validation is mainly on utility of the system. In recent literature on verification and validation, usability is also included explicitly, but with less focus (for example, in Schulmeyer (2000), a book on verification and validation, only one chapter out of sixteen chapters was devoted to usability). After the process of verification and validation, there is usually another test called the system test, which includes testing the software when connected to other systems in order to examine the quality of the overall system. Due to the limit on the length and the distance of this topic from the central theme of this dissertation, this topic will not be discussed here.

Another category of evaluation during development is usability evaluation. Lindgaard (1994) indicated that in the early days of computing, operators of computers were usually a part of the technical team and computers were much less interactive. Getting computers to complete the tasks were the main concern. Nevertheless, as the use of computer became more widespread and less technical personnel were recruited as computer operators, human-computer interaction (HCI), which is a related subject to usability, became a concern. Grudin (1992) also pointed out that there were two different emphasis on software quality developed from two different development situations. In the situation of in-house software development in corporations, users of computer systems were usually operators and engineers using batch-mode processing. Utility, or what tasks could the system perform, was the measure of the quality. On the other hand, in the situation of COTS software development, users of the systems were assumed to have no formal training and the emphasis of the products was on the usability of the system.

As mentioned above, usability concerns arise from the need of less technical users and this discipline was developed later than techniques on verification and validation of the utility of software. As suggested by Lindgaard (1994), concerns for usability can also be found in nearly all phases of a development cycle for software, namely feasibility, research, development and

operation phase. (Research phase is usually known as requirement gathering, in which the author introduced a number of research methodologies to discover the need of the users and tasks in context.) As for the concern of usability evaluation, the benchmark for usability is set from the result from the research phase. Guidelines and heuristic analysis are introduced at the development phase. Empirical testings are also conducted during different iterations of the development to evaluate the product against the benchmark (Nielsen 1993).

Two common usability testing methods, heuristics evaluation and 'laboratory' testing will be presented as an introduction to the actual methods of evaluation. Heuristics evaluation is probably one of the quickest ways to assess usability (Lindgaard 1994; Nielsen 1993). It may involve a small number of evaluators giving subjective and informal comments on the interface. Areas to assess suggested by Nielsen (1993) were simple and natural dialogue, speak the users' language, minimize user memory load, consistency, feedback, clearly marked exits, shortcuts, good error messages, prevent errors and help and documentation. The disadvantage of this method is the result may be coarse and the reliability and validity of the result can be significantly influenced by the limitations of the evaluators (Lindgaard 1994; Nielsen 1993).

Another common usability testing method involves putting users in a 'laboratory' environment and collect quantitative measurements on how tasks were done (Dumas & Redish 1999; Lindgaard 1994; Nielsen 1993; Rubin 1994). The 'laboratory' or test room usually contains a bench and the test computer for the participating user to perform the prescribed tasks. Other personnel such as a test monitor and observers may be also involved. They may be in the same room as the participant or in another room observing using different techniques. Quantitative measurements such as time and number of errors made were recorded (Rubin 1994). The advantage of the test is empirical data can be obtained for comparison (Nielsen 1993). The disadvantage of this exercise is that the test environment is not the real environment that the

software will situate in (Rubin 1994) and the test may not discover 'unexpected' usability defects as the scope is fixed (Nielsen 1993).

After the discussion of utility and usability, intrinsic, which is the nearest evaluation to the development team, will be presented. As mentioned above, intrinsic evaluation is about software engineering process and code quality. An example of the evaluation of software engineering process is the Capability Maturity Models (CMM) (Software Engineering Institute 2003a). This standard was sponsored by the US Department of Defense the Office of the Under Secretary of Defense for Acquisition, Technology, and Logistics. Three dimensions were recognized, namely, process, technology and people, which would affect quality. Process, or system engineering, was chosen to be the key dimension (Bate et al. 1995). Examples of areas covered by the model are systems engineering (SE-CMM), software acquisition (SA-CMM), integrated product development (IPD-CMM), people management such as staffing, training, etc. (P-CMM), team software process (TSP) and personal software process (PSP) (Software Engineering Institute 2003a). For each of the areas, five levels of capabilities were defined. Level 1 was named 'Initial', where there was no management at all. Level 2 was named 'Repeatable', where there was basic management and successes could be repeated. Level 3 was named 'Defined', where management and software processes were documented and standardised. Level 4 was 'Managed', where performance was measured and controlled. Level 5 was named 'Optimizing' where processes within the organization were constantly improving based on the measured performance and innovation (Software Engineering Institute 2003b). Appropriate practices were suggested for each level for each of these areas and corresponding benchmarks were established.

Other than the process of the production of software, the quality of the source code itself can be measured. It is possible for a piece of software to satisfy the requirement but difficult to

maintain. Fenton & Pfleeger (1997) suggested that reusability, maintainability, portability and testability are important factors in during software revision. One way to obtain indicators on these factors is by measurement. For example, for testability, the tester needs to understand the logic within the source code and documentation and comments are necessary. An indicator to this situation is comment density, which can be obtained by dividing the number of lines of comments by the total number of lines of code (Fenton & Pfleeger 1997). Another example for indicators for maintainability is the complexity for the control-flow structure of modules. One common measurement is cyclomatic complexity measure, which is calculated by the number of independent arcs and nodes (McCabe 1976). Other than quantitative indicators, method such as code inspection (Pressman 1997) can also help to estimate the quality of source code.

After the discussion of intrinsic evaluation, the last category - context – will be presented. According to Bevan (1995), it is not enough to just consider the quality of the interface of a program because it will be used by different users on different tasks in different situations. The specifics of users, tasks and social organisational environments may also be determining factors on the success of the program (Figure 5-3). In other words, utility is about what a program can do, usability is about quality of the interface, and context is about the quality of use when the software is situated in its own environment. The importance of context is also echoed by Lindgaard (1994) and Nielsen (1993) that user profiles and task profiles need to be established for usability evaluation. Bevan (1995) suggested that the definition of usability should be expanded to include these factors, but here a separate category is defined to highlight its significance.

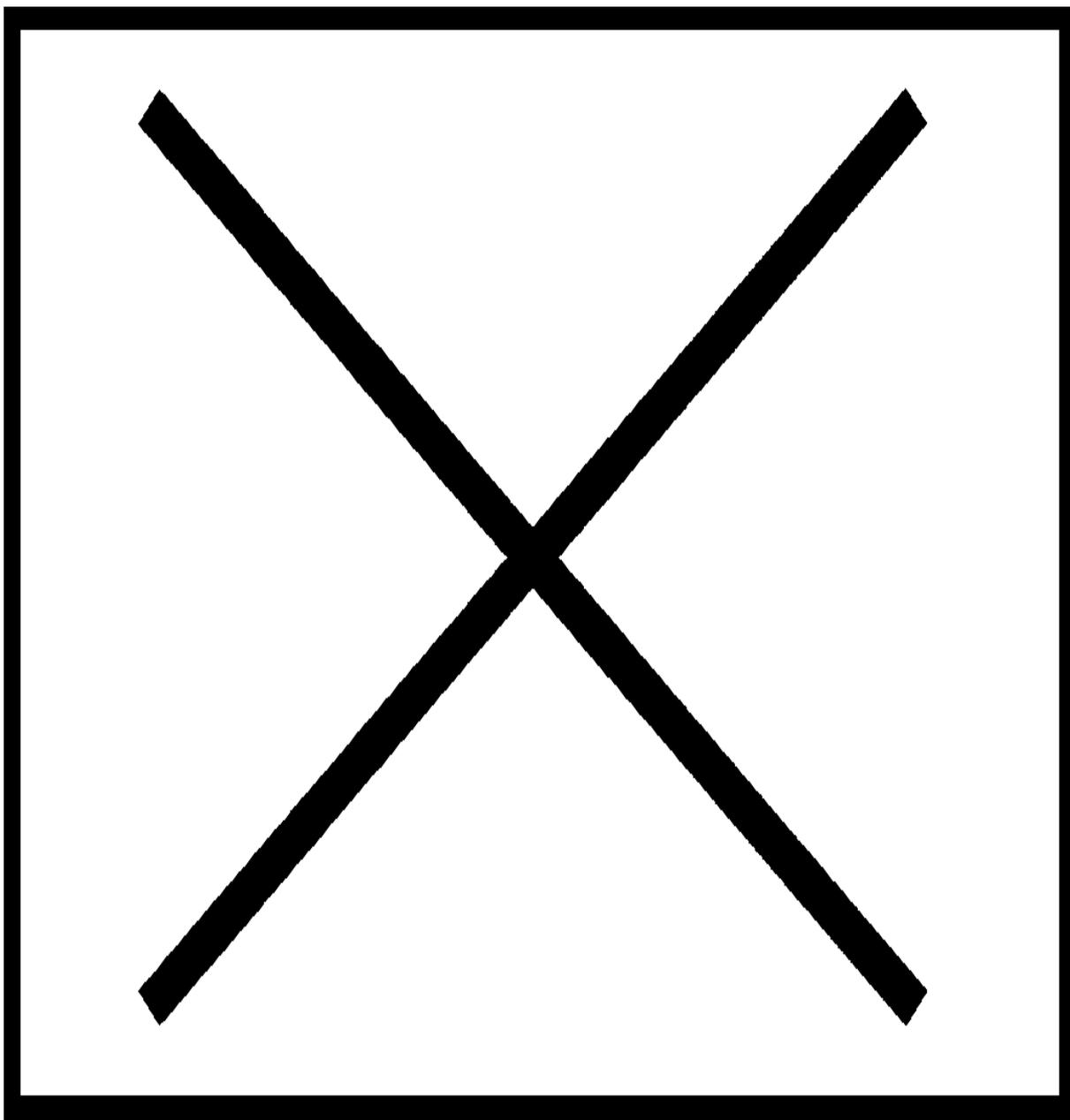


Figure 5-3 Quality of Use Measures Determined by the Context of Use (Bevan 1995)

The context category is proposed to be the most distant category from developers. Cooper (1999) explained that programmers usually have a different mentality and culture than designers (of software) and users. One of the differences is the focus on technical issues and it is difficult for programmers to see issues from the viewpoints of other parties. On the other hand, the complexity of the situation of the environment of deployment also institutes barriers for the developers. Therefore, the context category is proposed as the furthest from the

developers. The fact that there is not much literature on context evaluation during development also supports the proposal above and it can be observed that context evaluation is the least disciplined category of evaluation. Nevertheless, as we will see in the discussion of product evaluation, more contextual concerns can be discovered from literature from the users' perspective.

5.3.1.3 Software Product Evaluation

After the introduction of evaluation during development, evaluation of software product will also be discussed. Unlike evaluation during development, product evaluation usually bases on the viewpoint of the users. As explained above, intrinsic concerns are closer to developers and contextual concerns are closer to users. There is a possibility that in product evaluation that more contextual concerns can be discover. There may be also a lesser emphasis on intrinsic concerns.

Literature on product evaluation is collected and reviewed. It includes general software acquisition (Le Cornu 1996; Software Engineering Standards Committee of the IEEE Computer Society 1998b), library related procurement (Bosch, Promis & Sugnet 1994; Fraser & Goodacre 1993; Lee, S. D. 2002), accounting software selection (Australian Society of Certified Practising Accountants' Information Technology Centre of Excellence 1995) and education software evaluation (Squires & McDougall 1994). This collection is considerably more scattered than the literature on evaluation during evaluation and many of the literature only focus on a narrow domain. Most references mentioned above promote more or less one coherent method in evaluation, except for Squires and McDougall (1994), in which ten evaluation methods on education software were included and discussed, together with another method that is different from these ten methods. The authors devised this novel method partly from the reflection of these other methods.

From the survey of the mentioned literature, the focuses of most of product evaluations are on utility, usability and context. There are only a few include concerns in the intrinsic area. Considering the evaluation checklist provided in Lee (2002) for assessing electronic dataset for library (such as e-journals), a number of concerns on the functionalities and content of the dataset are listed. Quality of interface and ease of use are also included. Another example can be taken from IEEE Recommended Practice for Software Acquisition (Software Engineering Standards Committee of the IEEE Computer Society 1998b, pp. 31-2) "Software Evaluation" checklist. It includes concerns such as functionality, performance, ease of use and adequacy of documentation. Contextual concerns such as cost to acquire and use are also included. Intrinsic concerns are also found in this evaluation – availability of source code and its quality.

From the above analysis, though all the four categories are identified, utility seems to be the most prominent concerns. On the other hand, intrinsic concerns are relatively rare with the exception of the evaluation by IEEE. Most of the evaluation literature only mentions issues such as reputation of suppliers and support provided by suppliers. Though these issues are related to the suppliers, they are not related to software development process or quality of source code and thus cannot be categorised into intrinsic concern. Furthermore, they do not fit the other three concerns either. The approach chosen to resolve this situation in this study is to extend the definition of intrinsic concerns to include issues that relate to the developers or suppliers of the software.

Reasons can be proposed on why intrinsic concerns can be difficult for product evaluators. From the IEEE Recommended Practice for Software Acquisition (Software Engineering Standards Committee of the IEEE Computer Society 1998b, pp. 31-2) "Software Evaluation" checklist, intrinsic concerns such as availability of source code and its quality were included. Nevertheless, this type of evaluation can be technical and it may be only feasible for a company

with an IT department or IT experts. Another suggestion that falls into the category of intrinsic from this IEEE evaluation is to obtain opinions from other users of the suppliers. This can also be difficult due to lack of information, as suppliers are not obliged to provide a list of unsatisfied customers.

A number of contextual concerns are also raised. One of the most common contextual concerns is the amount of money available, or in short, budget (Lee, S. D. 2002; Nielsen 1993). Another contextual concern can be found during an evaluation of accounting software that a review of the organization, process, information flow, staff required and their attitude is recommended. The review is useful to the discovery of opportunities of re-engineering so that the re-engineering process can be introduced together with the employing of an accounting software (Australian Society of Certified Practising Accountants' Information Technology Centre of Excellence 1995). On the other hand, Squires and McDougall (1994; 1996) proposed the perspectives interactions paradigm in which evaluations needed to be performed in the situation of learning. Three major actors were considered, namely designer (of the software), teacher and student. Three perspectives were thus proposed: teacher and student, designer and student and designer and teacher. Taking the teacher and student perspectives as an example, the teacher of a class may adopt different roles such as resource provider, manager, coach, researcher or facilitator to implement certain pedagogical approach. Students may also be arranged to work in groups or have one-to-one access to computers. Moreover, the computer may also play a major or a minor role in the process of teaching and learning. The software was then evaluated on its quality in supporting teaching and learning with this known set of parameters inside a classroom. From the example in the literature above, it can be illustrated that contextual concerns can be very domain specific and complex.

Though a diversity of contextual factors is found in the literature, one observation is that most of the factors mentioned are practical factors such as cost, installation and maintenance. Socio-organisational implications of the introduction of the software to the whole system are less frequently asked. Examples of such questions are how well is the software integrated with the current system (Australian Society of Certified Practising Accountants' Information Technology Centre of Excellence 1995) or how does the perception of the designer of an educational software on the 'philosophy' of learning (such as learning means spoon-feeding information to student or a process to explore, experience and assimilate) affects teaching in classroom when that particular software is used (Squires & McDougall 1994). These concerns require substantial understanding of the current system by the evaluator and may even require simulation and imagination in the evaluator's mind. It may prove to be difficult for some evaluators, who are normal users. Nevertheless, if these factors are difficult for the users to foresee, they will be even harder for developers to even imagine. This may be one of the major difficulties for software to perform as expected in production environment.

5.3.1.4 Software Evaluation and FOSPHost

From the discussion above, one can see that the four evaluation categories proposed could effectively classify most of the evaluation concerns from both evaluation during development and product evaluation. One may ask why there is a need for a combined framework. The reason lies in the nature of Free/Open Source software. In Free/Open Source software, source code is available for evaluation. Examples can be seen from Schach (2002) and Stamelos et al. (2002) in which software metrics were employed to evaluate the maintainability and code quality of Linux kernel. As argued above, evaluation of source code can be too technical for most product evaluators who are just users. Nevertheless, other than the source code, information on the development process and the community built around the software are accessible as well to the evaluators. It has been suggested by Kenwood (2001) that when evaluating Free/Open Source software, factors such as amount of talents captured in the

development community, leadership reputation and structure, development speed, maturity of project and popularity can be considered. Some of these factors can be evaluated with less technical technology and the feasibility for intrinsic evaluation from product evaluators can probably increase in the situation of Free/Open Source.

On the other hand, the developers of Free/Open Source software may be users themselves (recalling one of the motivation for developers is to "scratch ones' own itch" (Raymond 2000b)). There may be a possibility that the distance from the developers to contextual factors can be decreased. To conclude, in the situation of Free/Open Source, it is possible that methods in evaluation during development and product evaluation are both applicable. A combined framework will hopefully form a basis for a more comprehensive analysis of the Free/Open Source phenomenon.

After the discussion of the advantage of the proposed classification, a few other observations can be explored also. The classification proposed is not perfect and overlapping does exist. An example is bugs in software. Bugs can be classified as defects in utility. The frequency of bugs can also be classified as a usability problem that affects the satisfaction of the users. This example shows that a particular factor (bug) can be classified into two different categories (utility and usability). Therefore, when there is an evaluation concerning bugs, it is related to both categories. Another example is that the availability of hardware and network can be a technical problem of what hardware the software can support but also a political problem of distribution of resources in the organization. Nevertheless, in most classification systems, several items that are difficult to categorise usually exist and ambiguities may sometimes be found. The classification proposed here probably achieve to form a conceptual framework for considering software evaluation as a whole, without the separation of evaluation during development and product evaluation.

Another possible shortcoming of employing this evaluation software approach on FOSPHost is that FOSPHost sites can be regarded as services to community(s). FOSPHost is indeed very much related to software but FOSPHost is not all about software as there is a service aspect. It is possible that software evaluation may not cover some areas of FOSPHost. Then evaluation methods in general program (such as some social welfare programs to aid the poor to get educated and leave poverty) can be relevant (Breakwell & Millward 1995; Owen & Rogers 1999; Rossi, Freeman & Rosenbaum 1982). Nonetheless, effective and sophisticated assessments of web sites using just the concept of usability (Nielsen 2000; Travis 2003) have been done and the evaluation approach suggested above covers areas more than these usability assessments. Moreover, literature on applying social evaluation methods on web sites are also few, and software evaluation methods are probably the closest evaluation methods for FOSPHost. Also, by the adjustment of intrinsic concerns, service related aspects such as quality of support could also be accounted for. Finally, with the small amount of the understanding we have on FOSPHost, it is difficult to judge what extra changes will be required. Aligning with the spirit of an exploratory study, the evaluation approach suggested will be employed first and its appropriateness will be discussed afterwards.

5.3.1.5 Presentations of Evaluation

After the introduction of the evaluation classification, the different formats of the presentation of evaluation need to be discussed. From data collection, important issues about FOSPHost are expected to be discovered. The final evaluation model then will be built on these issues and certain presentation formats will be adopted. Therefore, it will be beneficial to discuss common evaluation presentation formats and their characteristics. From the literature of evaluation during development and product evaluation, the checklist format with specific items seems to be the most common presentation. Another possible format is a variation of the checklist format called the framework format (Squires & McDougall 1994). Different categories of

software with different corresponding sub-checklists were proposed in the framework formats. For example, education software can be classified by application type, education role or education rationale. An example of framework based on application type may propose a different sub-checklist for tutorial and drill and practice program, arcade-type games, simulation games, laboratory simulations and content-free tools (Blease 1986). Lastly, there is also another type of evaluation that based on only a few broad topics, such as the perspectives interactions paradigm mentioned above (Squires & McDougall 1994, 1996).

As checklist is the most common presentation, it will be beneficial to have further examination. Generally, a checklist comprises of statements on particular characteristics of the software being evaluated. The evaluator usually will be asked to respond to the statement in one of the several forms, namely binary (Yes/No), subjective scale, weighed scale, measured results and qualitative answer.

A binary response requires the evaluator to choose one response out of two and the common form is a yes or no answer. An example of binary response is showed below (Software Engineering Standards Committee of the IEEE Computer Society 1998b, p. 32):

- 1) Will the software be easy to use? Yes No
- 2) Is it designed for straightforward operation with a well-documented
operating procedure? Yes No

A subjective scale provides the evaluator a scale with different values in order to describe the degree of the specific characteristic prescribed. An example of subjective scale is taken from an education software evaluation model:

	Rating SA –Strongly agree A –Agree D –Disagree SD –Strongly disagree NA –Not Applicable	
Usability	SA A D SD NA	1. The layout of the program is consistent.
	SA A D SD NA	2. It is easy to navigate around the program.

Table 5-2 An Example of Courseware Evaluation

A subjective scale can be qualitative (as in the example) or quantitative (for example, a scale from 1 to 5 where 5 denotes 'strongly agree' and 1 denotes 'totally disagree'). The results obtained from a quantitative scale may be processed using statistical means afterwards.

Weighed scale is a variation of quantitative subjective scale. The evaluator still needs to respond to a quantitative scale for an item. Additionally, in order to distinguish between the different significance of each item, a weight will be given to each item. The final score for a subject under evaluation can be obtained by summing the product of the subjective scale and the weight. In the example quoted below, the weight is assigned subjectively. Weights can also be adjusted so that the maximum total is exactly 100.

Criteria	Weight	Web Browser 1	Web Browser 2
Functionalities	3	9 x 3=27	6 x 3=18 (Remark: Most of the new features are copied from Browser 1)
Usability	3	8 x 3=24	6 x 3=18
Extensions	1	9 x 1=9	4 x 1=4
Popularity	1	1 x 1=1	9 x 1=9
Support	2	8 x 2=16	6 x 2=12
Total		77	61

Table 5-3 Sample evaluation matrix using scores and weighing

Checklist with measured results is different from quantitative subjective scale on the fact that the results are measured. They can usually be found in usability laboratory test and software metrics measures. Though the results of subjective scale and measurement can both be quantitative, measurements usually give the impression of less subjectiveness. One list compiled by Stamelos et al. (2002) comprises of the following metrics in assessing components (functions in C language). Preferred ranges for each metric are given in square brackets:

1. number of statements [1-50]
2. cyclomatic complexity [1-15]
3. maximum levels [1-5]
4. number of paths [1-80]
5. unconditional jumps [0]
6. comment frequency [0.2-1]
7. vocabulary frequency [1-4]

8. program length [3-350]
9. average size [3-7]
10. number of inputs/outputs [2]

(Detail explanation for each metric is omitted.)

The last type of response form is qualitative, which means that answers are given in the form of paragraph(s) of text. An example is taken from the 'Systems Engineering Capability Maturity Model' under the section of 'Verify and Validate System' (Bate et al. 1995, pp. 4-66):

Base practices list

The following list contains the base practices that are essential elements of good systems engineering:

BP.07.01 Establish plans for verification and validation that identify the overall requirements, objectives, resources, facilities, special equipment, and schedule applicable to the system development.

BP.07.02 Define the methods, process, reviews, inspections, and tests by which incremental products that were verified against established criteria or requirements that were established in a previous phase.

BP.07.03 Define the methods, process, and evaluation criteria by which the system or product is verified against the system or product requirements.

BP.07.04 Define the methods, process, and evaluation criteria by which the system or product will be validated against the customer's needs and expectations.

BP.07.05 Perform the verification and validation activities that are specified by the verification and validation plans and procedures, and capture the results.

BP.07.06 Compare the collected test, inspection, or review results with established evaluation criteria to assess the degree of success.

The response to this list will probably be in a report format documenting the software development process of the subject evaluated in the area of 'Verify and Validate System'.

The implication of employing different forms of presentation can be showed by Table 5-4.

Checklist / Framework				Broad Topics
Binary	Subjective Scale	Weighed Scale	Measured Results	Qualitative Answer
Specific				Broad
Least Mental Effort				Most Mental Effort
Restrictive				Discovery

Table 5-4 Implications of Different Forms of Presentation

From the table, several limitations when employing different checklists can be explained. One limitation with quantitative response can be that the scope of evaluation is restricted to the specific statements in the list. For non-weighted checklist, each statement might seem to have the same importance (Squires & McDougall 1994). If calculation is involved, then there will be an assumption that the distance between each item in the subjective scale is equal, which may not be accurate (For example, in a 1-5 scale where 1 is the worst and 5 is the best, the distance between 3 to 4 may be shorter than 4 to 5 for a perfectionist).

On the other hand, one of the biggest limitations for broad topics is that more effort is required to learn to use the evaluation tool and also during evaluation. Taking the example of the perspectives interactions paradigm (Squires & McDougall 1994, 1996), three perspectives, teacher and student, designer and student and designer and teacher, were used for evaluation.

For each perspective, further details are provided, but much less specific comparing with 'Does feature X exist?' Evaluators who want quick result will probably not employ such evaluation presentation. Checklists with quantitative scoring also seem to be more objective.

It can then be concluded that each format has its own shortcomings. Definite guideline in what presentation format to choose may not exist but these considerations will be taken into account the construction of evaluation model in this research.

5.3.1.6 Users of Evaluation

In this last section on evaluation literature review, one of the determining factors on the presentation format of an evaluation is the users and audiences of the evaluation. The prospective users of the evaluation model of FOSPHost sites can be literally anyone who takes an interest on the topic. This is because the model will be available to the public through the World Wide Web. This decision is made to conform to the culture of Free/Open Source.

Under this circumstance, two different types of people may become the users of the evaluation – Free/Open Source developers and new comers. Free/Open Source developers are people who are already developing Free/Open Source software and probably using a FOSPHost site. They may want to know more about the topic and make improvement on their current practices. As suggested by Pavlicek (2000) on the general culture of Free/Open Source, this group of people usually focus on technical details and accuracy. Another type of people are the new comers. They may want to investigate the possibility of using a FOSPHost site and probably need more introductory and understandable information.

To satisfy the expectation from these two groups, one strategy may be to emphasize on utility. Discussing features of a FOSPHost site can fulfil the expectation of Free/Open Source developers on technical details of the site. With introductory information, new comers can also

form preliminary ideas on what a FOSPHost site. On the other hand, there should also be links between features to factors in other categories such as context so that a comprehensive picture on the topic of FOSPHost can be portrayed. These factors will be considered in the construction of the evaluation model.

5.3.1.7 Summary

In the above discussion, four categories of software evaluation are presented together with rationale on their relevancy to the evaluation of FOSPHost. Different evaluation presentation formats and users considerations are also discussed.

5.3.2 Delphi Survey

After the survey of different software evaluation methods, a researcher can just choose one of methods mentioned and apply it to FOSPHost. Nonetheless, the question will then be which methods will be promising to yield results that will reveal the nature of FOSPHost as well as the Free/Open Source phenomenon. In order to answer this question, we need to know which aspects of a FOSPHost site are more important. Thus, Delphi survey technique (Linstone & Murray 1975) was chosen to collect expert opinions to what are the important aspects to a FOSPHost site. The Delphi method is a structured group interaction process that is directed in 'rounds' of opinion collection and feedback (Turoff & Hiltz 1996). The name Delphi comes from the Greek mythology that future events could be foretold in the Oracle at Delphi, a Greek ancient city. The method was developed by Olaf Helmer and Norman Dalkey at the RAND corporation in 1953 for forecasting purposes (Helmer 1975; Lang). Nowadays, it has been used a variety of topics to collect expert judgements (Linstone & Murray 1975). Possible rationales for chosen this method is provided by Linstone & Murray (1975, p. 4):

- The problem does not lend itself to precise analytical techniques but can benefit from subjective judgments on a collective basis

- The individuals needed to contribute to the examination of a broad or complex problem have no history of adequate communication and may represent diverse backgrounds with respect to experience or expertise
- More individuals are needed than can effectively interact in a face-to-face exchange
- Time and cost make frequent group meetings infeasible
- The efficiency of face-to-face meetings can be increased by a supplemental group communication process
- Disagreements among individuals are so severe or politically unpalatable that the communication process must be refereed and/or anonymity assured
- The heterogeneity of the participants must be preserved to assure validity of the results, i.e., avoidance of domination by quantity or by strength of personality ("bandwagon effect").

At the time when the Delphi survey was designed and conducted, the literature on FOSPHost was few and the Free/Open Source phenomenon as argued in chapter 4 was diverse and complex. Therefore, employing Delphi survey is suitable (Linstone & Murray 1975; Twining 1999; Ziglio 1996). Moreover, a number of stakeholders from diverse backgrounds could also contribute to the survey and exchange ideas (Linstone & Murray 1975) and hopefully to obtain a more comprehensive set of data on the topic. This survey can also be done on the World Wide Web to collect data globally with a low cost. Detail execution of the surveys will be explained in the next sub-section.

The argument above presented the appropriateness of the application of the Delphi survey technique, but whether it conforms to the positivist philosophy is yet to be examined. Mitroff & Turoff (1975) showed that Delphi survey technique supported the Lockean inquiring system and this system is compatible with positivism. An inquiring system is a model concerning how

to transform a 'raw data set' found in the existing world into some kind of conclusion (Mitroff & Turoff 1975). Moreover, the Lockean inquiring system asserted that truth was based on empirical content, not pre-assumption. Delphi survey technique supported this inquiring system because the conclusion drawn was based in judgements that were experientially collected. This inquiring system is arguably similar to positivism in its understanding of drawing conclusion from a number of subjective observations to obtain an objective result. Therefore, Delphi survey technique is compatible with the positivist worldview and is suitable for this research. This is also compatible with the inductive strategy of this research as it based on few initial, broad questions and the result is based on empirical data collected.

5.3.2.1 Administration of Instruments and Procedures

The Delphi survey was designed to have three rounds. The first round would be qualitative and the questionnaire for the first round was derived from the model of individual participation in a Free/Open Source community. In rounds 2 and 3, participants would be asked to rate the importance of the different statements from the results from round 1. A survey on the design and execution of the Delphi survey would be conducted afterwards to collect opinion on the quality of the survey itself. Participants would be given a chance to give comment on any aspect of the whole process. Before the execution of the actual survey, a pilot survey will be executed.

5.3.2.2 Participants of Survey

Three groups of people were invited to participate in the Delphi survey, namely, expert Free/Open Source developers, FOSPHost administrators and academics in the area of Free/Open Source. In a positivist viewpoint, academics were believed to understand the objective knowledge of Free/Open Source so that the opinion obtained will be composed of participants of Free/Open Source communities as well as observers.

Further definitions of the three groups were developed for recruitment. For expert Free/Open Source developers, their names were found from two popular sites, Advogato (Advogato 2003) and SourceForge (SourceForge 2003), where Free/Open Source programmers met and ranked each other. For Advogato, those who achieved a Master status were chosen; for SourceForge, those who had been rated the top ten highest ranked users were chosen. A FOSPHost administrator was defined as a maintainer of any FOSPHost site, independent of whether the person also takes care of the server at the operating system level. Lastly, for academics, any person who has published any article on the topic of Free/Open Source can be invited.

The World Wide Web became the primary tool to build the invitation list. Names of potential participants could be found on Advogato or Free/Open Source project web sites pages. Then, their names would be searched using search engines to discover other relevant information. Since names are not a definite unique identification of people, the information found by the search engines had to contain convincing material about the projects that the potential participants were believed to be involved.

In terms of number of participants required, as Delphi Survey aims at expert opinion rather than a sample from a general population, a panel of 10-15 experts in each group can already generate quality results for a homogenous group (Delbecq, Van de Ven & Gustafson 1975; Ziglio 1996). As there is no explicit guideline the minimum number for a heterogenous group, it is assumed that each group should have at least 10 experts.

5.3.2.3 Questionnaire Development for the Survey

One important aspect in most survey is what questions to include. According to the model of individual participation to a Free/Open Source community, the important aspects for a FOSPHost site contain 4C (Communication, Contributions, Co-ordination and Culture) and community participants related factors (Motivations and Barriers to join community, Positive

and Negative effect). By referring to this model, 12 questions were devised as the initial questionnaire for the Delphi survey. (The original acronym for FOSPHost was IFHOSP, which stood for Infrastructure For Hosting Open Source Project. This acronym was not well received from the responses of the participants and thus the author changed the acronym into FOSPHost.)

1. What are the objectives of an IFHOSP site?
2. What tools can be employed on an IFHOSP site and what are the important features and usability factors for each of them?
 - 2.1.1 What is the name of the tool that can be employed?
 - 2.1.2 What are the important features of this tools?
 - 2.1.3 What are the important usability factors of this tools?
3. What work practices and culture should be promoted?
4. What are factors that motivate users to use an IFHOSP site?
5. What are barriers that prevent users from using an IFHOSP site?
6. What are the positive results for users in using an IFHOSP site?
7. What are the negative results for users in using an IFHOSP site?
8. What are factors that motivate administrators to setup or maintain an IFHOSP site?
9. What are barriers that prevent administrators from setting up or maintaining an IFHOSP site?
10. What are the positive results for administrators in setting up or maintaining an IFHOSP site?
11. What are the negative results for administrators in setting up or maintaining an IFHOSP site?
12. What are other important issues in IFHOSP?

Most of the 12 questions were deduced from the model, with the exception of question 1 and 12. Question 1 was aimed at collecting overall opinion on what a FOSPHost should do and question 12 was included to collect any additional opinion that did not fit the pre-set questions.

Question 2, 'What tools can be employed on an IFHOSP site and what are the important features and usability factors for each of them?', aimed at collecting opinion on the communication layer. Three sub-questions on utility and usability would be asked after participants suggested a new tool. Information on the contribution layer could be reflected from this question too.

For co-ordination and culture layer, as it is difficult to explain the exact difference between the two within one sentence, one question aimed at collecting opinion in both of these layers were devised: 'What work practices and culture should be promoted?'

As there were three groups of participants and two of them, namely, users and administrators, had direct experience with FOSPHost, two sets of motivations, barriers, positive and negative results questions were setup to collect opinion from each of these groups.

Analysing the questions using the software evaluation classification, explicit questions on utility, usability and context were presented. As the model of individual participation to a Free/Open Source community was developed from the viewpoint of a participant, questions for administrators were added to obtain intrinsic concerns.

For rounds 2 and 3, questions for the survey would be the summarised statements from the results of round 1. The data collected would then be the importance of these statements. Two methods were available, rating and ranking. Rating means to put a score on a statement out of a scale, while ranking is to re-arrange the statements in order of their importance. Statement

ranking is actually the method recommended by a number of researchers (Delbecq, Van de Ven & Gustafson 1975; Schmidt 1997; Turoff & Hiltz 1996). This method, however, does have its constraints. Firstly, the number of statements to be discerned cannot be too many, typically less than 20. Moreover, it was less flexible than rating. For example, in a Delphi survey on the United States drug abuse policies (Jillson 1975), opinion was collected by rating on both feasibility and desirability, which meant a participant had to select two numbers from a 1-5 scale for each statement. It would be more cumbersome to collect similar data using ranking, which could mean two separate pages with the same statements but ranked for a different factor. A number of Delphi surveys also employed rating (Enzer 1975; Goldstein 1975; Ludlow 1975). In this survey, rating was also chosen, as the number of statements will not need to be less than 20. A 1-7 scale was also chosen to give a larger differentiation of opinion.

5.3.2.4 Implementation of Survey on the Web Server

Turoff & Hiltz (1996) suggested that Delphi surveys could be conducted using electronic means and collected opinions that were not available to surveys using pen and paper. One of the unfortunate facts was that global communication via the World Wide Web was not yet very common by the time the paper Turoff & Hiltz (1996) was written. On the other hand, at the time the Delphi survey on FOSPHost was designed, the World Wide Web was an essential tool for participants in Free/Open Source communities and academia. Moreover, using the World Wide Web as a medium also save cost and time in mailing. The responses from participants would also go directly into a database for analysis. The author also had previous experience in implementing dynamic web sites. Therefore the method of implementing the survey using the World Wide Web was chosen.

Turoff & Hiltz (1996) suggested that Delphi survey was an asynchronous interaction. Participants could have the freedom to choose when and which questions to respond to.

Participants should also be provided the chance to return to the survey as many times as one would like before the closing of the survey.

Anonymity was another characteristics of Delphi survey. In most cases, participants will not be given any information about the identities of others. In such a setting, participants would be encouraged to express their opinion without the hindrance of their status (Linstone & Murray 1975; Ziglio 1996). Turoff & Hiltz (1996) took this point further to suggest that each participant could be given a pseudonym. The responses of a participant can then be known by a name without disclosing his or her identities. This could give participants chances to form richer pictures of characteristics other participants that agree or disagree with them.

The technical details of the survey will be presented below. As Delphi survey requires the feedback of previous results to the participants, the presentation of the results using web pages will also be explained. The analysis of the results, however, will be presented later in a separate chapter.

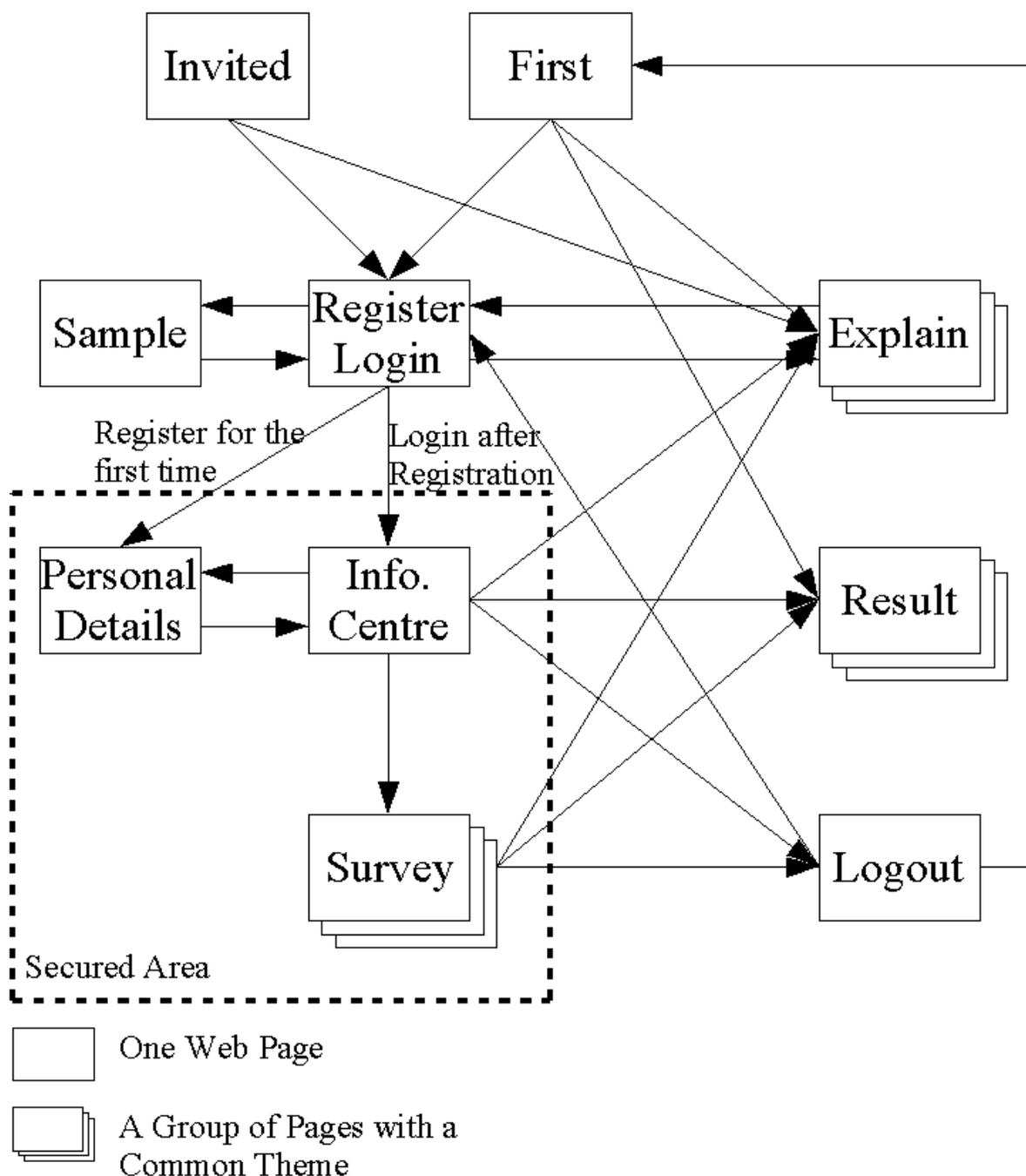


Figure 5-4 Site Map of Major Elements for Delphi Survey

The system developed was implemented on a Red Hat 7.0 Linux with Apache 1.3.27 web server, PHP 4.1.2 and PostgreSQL 7.0.2. The site map of the system was shown in Figure 5-4.

Users of the web site were expected to enter from the 'First' page or the 'Invited' page where the invitation email referred. Further explanation of the details of the survey was elaborated in the 'Explain' pages. Topics explained included 'What is Infrastructure For Hosting Open Source

Project (IFHOSP)?', 'What is a Delphi Survey?', 'Who is invited to participate in the Survey?', 'What is Required from the Participants?', 'Detail Procedure of Delphi Survey', 'What Results do we expect?' and 'Guideline for Participants'. There were also 'Point Form Summary' and 'Frequently Asked Questions' for users to find quick answers.

Registration\Login to IFHOSP Delphi Survey

Participants of the Delphi Survey are invited on an individual basis. If you are invited, please type in your UserID and password below to proceed.

Maybe you have the same or higher [level of expertise](#) than the participants but we have just missed you out in the invitation. Due to the workload of the researcher, unfortunately he cannot add anymore participants in at the moment. If you have any comment on the topic, please [click](#) here.

More information can be found below:
[Explanation of survey](#)
[Sample pages of survey \(with questions\)](#)

Registration\Login

UserID

Password

*If you encounter any login error with the Postgresql database server, please check the [Server Status](#).

Last Update: 22 Apr. 2002

Figure 5-5 Register/Login Page

Users could then go to the 'Register/Login' page (Figure 5-5) by following the links. There was a sample page for those who were still not sure about the survey. In the invitation email, each participant was given a username in the form of rXXXX (where X is a digit from 0-9) and a password made up of alphanumeric characters. Only those who had a username and a corresponding password can go into the secured area and each participant can only access his or her own survey.

Participant Details

This is the place to modify your details.

Change Password

You can change your password here or just leave the two textboxes blank if you do not want to change it.

New Password

Retype Password

Identity

Nickname Yolanda

Nickname is the name that you will be known in this survey. Initially, you can choose from a male and a female name. Then it will be fixed for the survey. For those who are involved in the previous rounds, I had to pick a name for you after receiving some advice that I have to make the nickname *STRICTLY* anonymous. If you really hate this nickname, email [me](#) and I will try my best to fix it.

Contact Email

Please give us your contact email if you want us to contact you via a different email address from the one that we are contacting you with. The above text box is always blank when this page is loaded. Please do not re-enter your email if you have already done so.

Expertise

1. Do you think you are an expert user in IFHOSP sites? Yes No

2. Are you an administrator of an IFHOSP site? Yes No

3. Have you ever published any article on the topic of Open Source/Free Software in an academic publication? Yes No

Figure 5-6 Participants Details

IFHOSP Delphi Survey Information Centre

Thank you for taking the time to participate in this survey.

Links to

- [Intro to Survey](#)
- [Round 1](#)
- [Round 2](#)
- [Round 3](#)
- [Post-Delphi Survey and Credit Procedure](#)
- [Personal Details](#)
- [Email Archive](#)
- [Logoff](#)

Latest News

2 Oct 2002

Sorry for the delay in releasing the [result of Round 3](#) of the survey, I went to a conference in Taiwan in August and other businesses in life and research hindered the progress. Please also express your opinion in the [Post-Delphi survey](#). If you have participated in the survey, you can also have your name listed in the [credit list](#).

21 Jun 2002

Round 3 is officially closed today. Thank you very much for giving your answers, and I will try to get the result out soon.

15 May 2002

[Round 3](#) has officially started today. Please click "round 3" on the left hand side under "Links".

The results for Round 1 & 2 can be found from [here](#).

Figure 5-7 Information Centre

After passing the verification process at the 'Register/Login' page, for those who register for the first time, they would be asked to fill in information about themselves, namely 'change password', 'choose nickname from a list' (with both male and female names), 'change contact email' and identifying expertise from the three categories (Figure 5-6). For those who filled in the personal details before, they would be directed to the information centre of the survey (Figure 5-7). This centre contained the latest news of the survey as well as links to current survey, 'Explain' pages, 'change personal details', the archive of email sent to the participants and a procedure to have their name listed in the credit list.

Round 1 Questionnaire



This is the questionnaire for round 1. There are 12 questions in total.

If you want to answer a question, please click on the question to enter the answer page. Otherwise, please click the no comment box on the right.

If you need any help, please click [here](#).

Questions	No Comment
1. What are the objectives of an IFHOSP site?	<input type="checkbox"/>
2. What tools can be employed on an IFHOSP site and what are the important features and usability factors for each of them?	<input type="checkbox"/>
3. What work practices and culture should be promoted?	<input type="checkbox"/>
4. What are factors that motivate users to use an IFHOSP site?	<input type="checkbox"/>
5. What are barriers that prevent users from using an IFHOSP site?	<input type="checkbox"/>
6. What are the positive results for users in using an IFHOSP site?	<input type="checkbox"/>
7. What are the negative results for users in using an IFHOSP site?	<input checked="" type="checkbox"/>
8. What are factors that motivate administrators to setup or maintain an IFHOSP site?	<input type="checkbox"/>
9. What are barriers that prevent administrators from setting up or maintaining an IFHOSP site?	<input type="checkbox"/>
10. What are the positive results for administrators in setting up or maintaining an IFHOSP site?	<input type="checkbox"/>
11. What are the negative results for administrators in setting up or maintaining an IFHOSP site?	<input type="checkbox"/>
12. What are other important issues in IFHOSP?	<input type="checkbox"/>

If you have finished the questionnaire, please proceed to the [summary](#) page where you can read all your answers to this questionnaire.

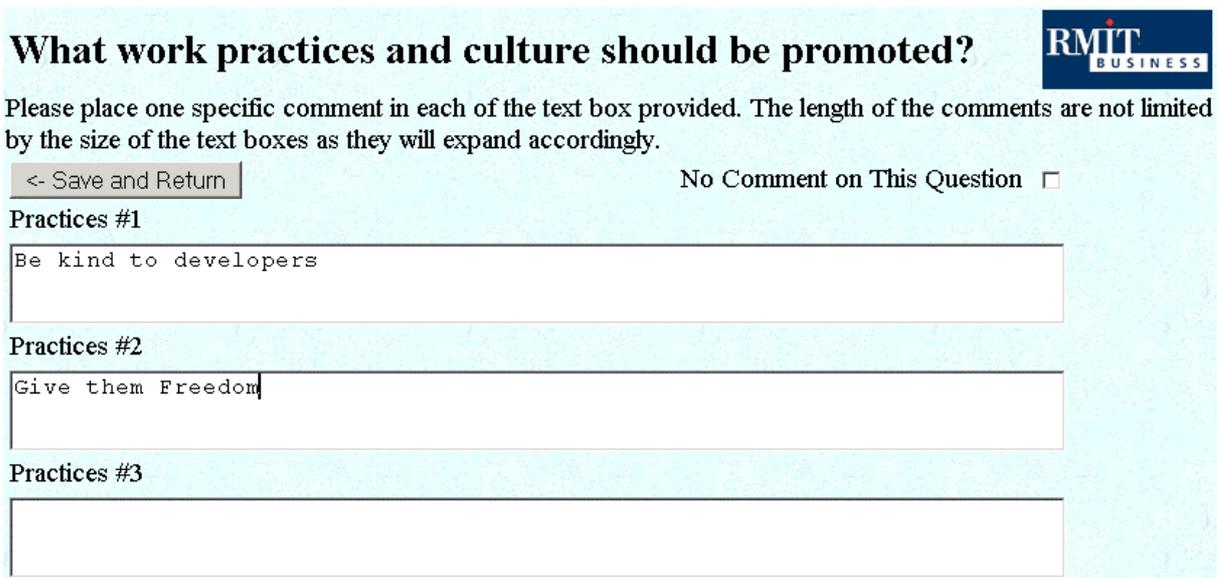
If you just want to logoff, please click [here](#).

If you need any help, please click [here](#).

Figure 5-8 Round 1 Questionnaire Question Page

For each round of survey, there was an introduction page, which then led to the question where the participant could choose to give answers to any of the twelve questions asked (Figure 5-8).

As required by the Delphi survey method that the participants could choose which questions to response to, 'no comment' check boxes were provided to obtain a more definite reply on their desire on not to respond to certain questions.



What work practices and culture should be promoted?

Please place one specific comment in each of the text box provided. The length of the comments are not limited by the size of the text boxes as they will expand accordingly.

<- Save and Return No Comment on This Question

Practices #1

Be kind to developers

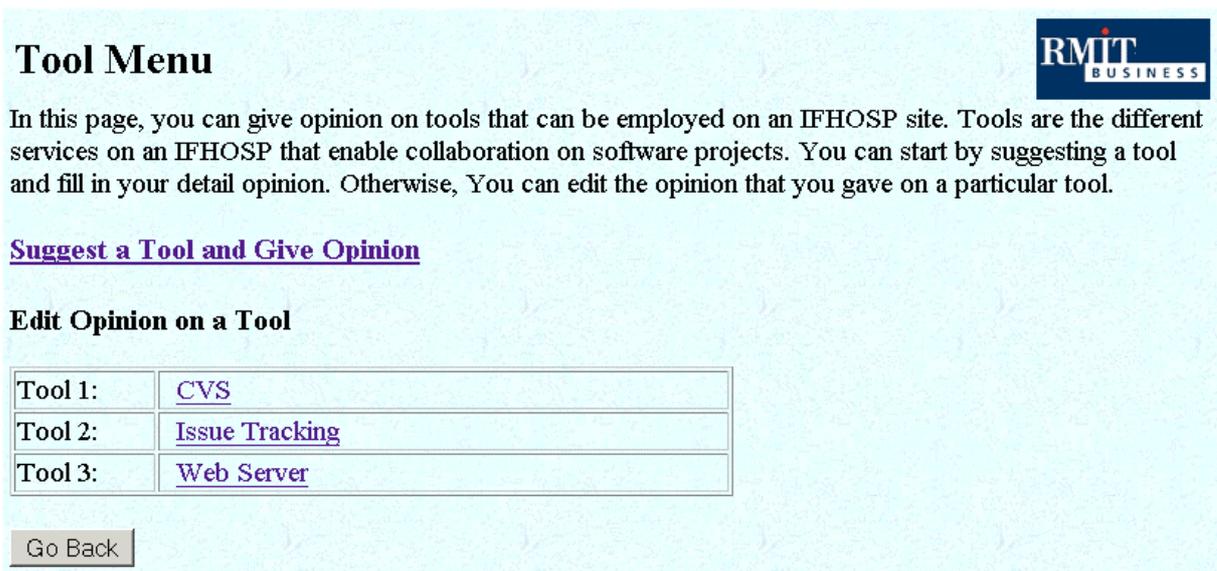
Practices #2

Give them Freedom

Practices #3

Figure 5-9 Round 1 Questionnaire Answer Page

Then the participant could choose one of the questions. In round 1, it would lead to the answer page where one could provide up to twenty answers to any questions (Figure 5-9).



Tool Menu

In this page, you can give opinion on tools that can be employed on an IFHOSP site. Tools are the different services on an IFHOSP that enable collaboration on software projects. You can start by suggesting a tool and fill in your detail opinion. Otherwise, You can edit the opinion that you gave on a particular tool.

[Suggest a Tool and Give Opinion](#)

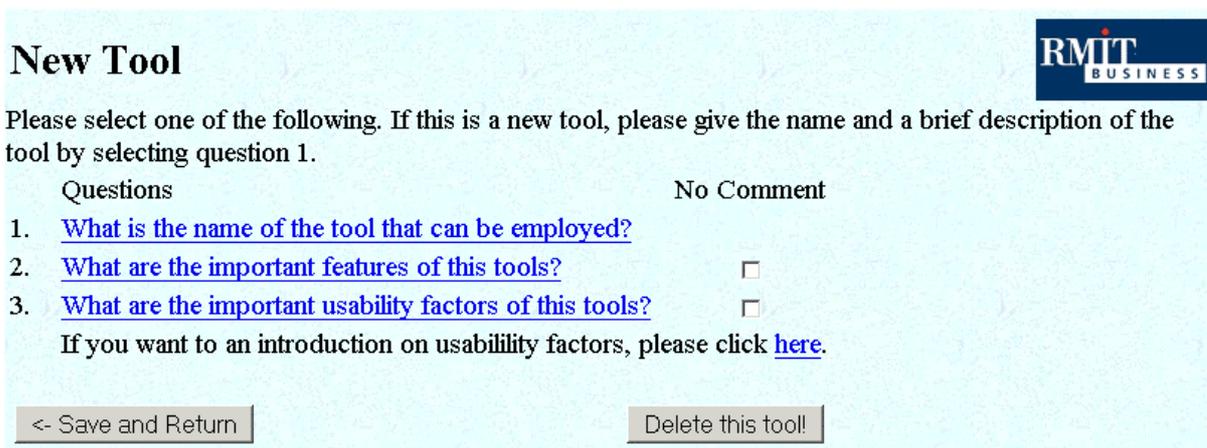
Edit Opinion on a Tool

Tool 1:	CVS
Tool 2:	Issue Tracking
Tool 3:	Web Server

Go Back

Figure 5-10 Menu for Question 2

For question 2, 'What tools can be employed on an IFHOSP site and what are the important features and usability factors for each of them?', a special structure was needed to capture the answers (Figure 5-10). This structure allowed the participants to suggest, edit and delete tools that they liked to comment on (the system can accommodate up to 999 tools).



New Tool 

Please select one of the following. If this is a new tool, please give the name and a brief description of the tool by selecting question 1.

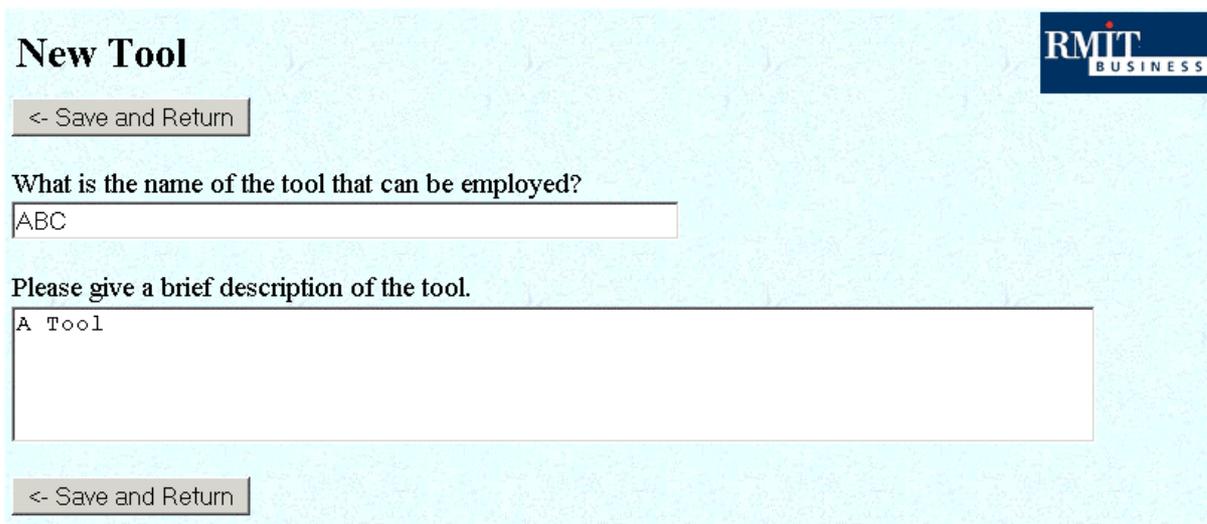
Questions	No Comment
-----------	------------

1. [What is the name of the tool that can be employed?](#)
2. [What are the important features of this tools?](#)
3. [What are the important usability factors of this tools?](#)

If you want to an introduction on usability factors, please click [here](#).

Figure 5-11 Adding a New Tool

When the participant selected the "Suggest a Tool and Give Opinion" link, a page was displayed to give the participant further instruction to comment on the tool (Figure 5-11).



New Tool 

What is the name of the tool that can be employed?

Please give a brief description of the tool.

Figure 5-12 Adding Name and Description

The first step in commenting a tool was to give the name and a brief description of the tool (Figure 5-12).

Tool 4: ABC

What are the important features of this tools?

Please place one specific comment in each of the text box provided. The length of the comments are not limited by the size of the text boxes as they will expand accordingly.

No Comment on This Question

<- Save and Return

Feature #1

Feature A

Feature #2

Feature B



Figure 5-13 Suggesting Features

Then the participant could suggest the important features of the tool (Figure 5-13).

Tool 4: ABC

What are the important usability factors of this tools?

You can give your answer by using the preset usability factors in the selection boxes.
OR
You can make up your own usability factors further down the page in the text boxes.

If you want to an introduction on usability factors, please click [here](#).

No Comment to This Question

<- Save and Return

Usability Factor #1

Accuracy	The information processed and presented can be
----------	------------------------------------------------

Usability Factor #2

Learnability	It is easy and intuitive to learn how to operate
--------------	--------------------------------------------------

Usability Factor #3

Efficiency	The users can achieve a level of high
International Interface	The site is designed for international audience
Links	The site contains useful links to relevant sites
Metaphor	The site employ metaphors to convey ideas
Printable	The web pages on site are designed to handle printing
Searchable	The content within the site can be accessed by a search mechanism on site
URL	The URL of the site is easy to remember and type
Use of Multimedia	Appropriate use of non-text elements to convey ideas
Accessibility	The site is accessible to users with disabilities
Connectivity	The connection to the site is fast
Security	The site is secure
Clearly marked exits	Exits are available when users want to abort a certain operation
Efficiency	The users can achieve a level of high



Figure 5-14 Selecting Preset Usability Factors

Usability Factor #10

<- Save and Return

User-defined Usability Factors #1

Usability Factors #1	Descriptions
Fun	The tool has to be fun to use

User-defined Usability Factors #2

Usability Factors #2	Descriptions

Figure 5-15 User-defined Usability Factors

When the participant commented on the important usability factors, one could select the preset usability factors or define one's own factors (Figure 5-13 and Figure 5-14). The preset usability factors were obtained from the literature review of a number of articles and books. A major proportion of the review was done before this PhD programme (Catella & Exploris Museum 1999; CIDOC Multimedia Working Group 1997; Ciolek 1997; Everhart 1996; Grassian 1998; Harris 1997; Henderson 1999; Hinchliffe 1997; Huang, Lee & Wang 1999; Jacobson & Cohen 1996; Jones, C. M. 1998; Kirk 1999; McIntyre Library 1998; O'Brien 1997; Ormondroyd, Engle & Cosgrave 1999; Purdue University Libraries 1999; Sarapuu & Adojaan 1998; Schrock 1999; Smith 1997, 1998; Strong, Lee & Wany 1997; Susan 1997; Wilkinson, Bennett & Oliver 1997) and other was done within this programme (Alexander & Tate 1999; Nielsen 1993, 2000). These usability factors are listed below.

Usability Factor	Brief Description
Accuracy	The information processed and presented can be relied upon to be correct
Updated Frequently	The site is frequently updated with latest information
Coherence	The information is presented logically and without contradiction
Completeness	All relevant materials are presented
Objectivity	The site is managed in an impartial manner without stereotyping or bias, such as gender and cultural prejudice

Usability Factor	Brief Description
Origin of Material	The origins of the material used on site are clearly stated
Quality of Expression	Writing material on site follows common language usage which is easy to understand
Uniqueness	The site consists of material that cannot be found elsewhere
Ease of Navigation	The information is in a structure that is easy to navigate
Easy to find from Outside	The site can be easily found by search engines, resource lists or from other advertisements outside the site
International Interface	The site is designed for international audience
Links	The site contains useful links to relevant sites
Metaphor	The site employ metaphors to convey ideas
Printable	The web pages on site are designed to handle printing
Searchable	The content within the site can be accessed by a search mechanism on site
URL	The URL of the site is easy to remember and type
Use of Multimedia	Appropriate use of non-text elements to convey ideas
Accessibility	The site is accessible to users with disabilities
Connectivity	The connection to the site is fast
Security	The site is secure
Clearly marked exits	Exits are available when users want to abort a certain operation
Efficiency	The users can achieve a level of high productivity when using the site
Feedback	The site gives informative feedback to users within reasonable response time
Few Errors	There are very few number of operational errors on site
Good error messages	Well written error messages that exactly indicates the problem and suggests solution(s).
Help and documentation	Useful documents and help messages are presented to assist users in using the site
Learnability	It is easy and intuitive to learn how to operate the site
Memorability	It is easy to remember how to operate the site
Prevent errors	The site is designed to prevent possible error like spelling mistake from users
Satisfaction	Users are subjectively satisfied when using the site

Usability Factor	Brief Description
Shortcuts	Shortcuts are available to expert users to speed up increase efficiency
Simple and natural dialogue	Interaction between computer and users is as simple as possible and information in the interaction is presented in natural and logical fashion
Speak the users' language	During the interaction between computer and users, terms and concepts that are familiar to users are used rather than system-oriented terms

Table 5-5 Preset Usability Factors

The participants were supplied with both the name of the factors and their respective descriptions to assist them in making an informed decision. They can also define any other usability factors if needed (Figure 5-15).

Summary



[Back to Questionnaire](#)

All your answers are presented in this page.

1. What are the objectives of an IFHOSP site?

- 1.1 To host projects
- 1.2 To support project co-ordination
- 1.3 To build a community

[Edit Answers to this Question](#)

2. What tools can be employed on an IFHOSP site and what are the important features and usability factors for each of them?

Tool 1: CVS

2.1 Brief Description

A Version Control Tool

Figure 5-16 Summary of Responses

After the participants finished the questionnaire, one could also generate a summary page to review the answer given on the summary page. There were direct links from the summary page to individual questions so that participants could update their responses conveniently (Figure 5-16).

Verification

Could you please check if your answers are correctly summarised?

Answer from Brendan

Brendan regards himself/herself as:

- An expert user in IFHOSP sites
- An administrator of an IFHOSP site
- has published one or more article(s) on the topic of Open Source in an academic publication

1. What are the objectives of an IFHOSP site?

1.1 build a better mousetrap

Concept -> To facilitate software development in a better way

1.2 distribute a better mousetrap

Concept -> To make software with better quality available to the world

1.3 maintain a better mousetrap

Concept -> To facilitate software development in a better way

Figure 5-17 Verification Page

After collecting responses in round 1, the qualitative data would be summarised and turned into questions of round 2. Moreover, to increase the validity of the survey, a verification procedure was added before round 2 to receive feedback from participants on the quality of the summary (Schmidt 1997).

In Figure 5-17, a verification page is shown. Each verification page was first generated from processed data based on the responses and then customised to each individual participant. Each participant received a unique email pointing to his or her page. One could then comment on the summarised concepts on the page and then sent it back through the web server.

Clarification

Could you also clarify or explain the following answer so that the researcher can summarise them properly?

3. What work practices and culture should be promoted?

3.6 fulfilling contracts

```
doing what you say that you will do and expecting others to do what they
say they will do
```

3.8 critique for the same of the task

```
mistyping on my part. should be critique for the sake of the task. that
is being constructively critical of products, procedures and people
including task leaders
```

Figure 5-18 Additional Clarification

Moreover, when the meanings of some of the responses were not immediately clear to the researcher, the researcher also took the chance of verification to have these responses clarified (Figure 5-18).

After the above process, the data collected was summarised and the results of round 1 was presented using HTML on the web (available not just to the participant, but the general public, in the spirit of Free/Open Source). It was presented in two sorting order, namely questions and participants.

Result for Question 1



Responses Hidden

[Show Responses](#)

[Go back to the List of Questions](#)

[Go to Question 2](#)

1. What are the objectives of an IFHOSP site?

- 1.1. To enable distributed software development for developers from different geographic locations
- 1.2. To support concurrent and collaborative software development
- 1.3. To facilitate communication between developers
- 1.4. To facilitate communication between developers and users (of Free Software/Open Source software)
- 1.5. To facilitate cooperation between related parties (programmers, designers, documentation writers, advocates/salesman, etc.)
- 1.6. To facilitate testing to the source code in different environments
- 1.7. To promote existing project(s) hosted on site to users of software
- 1.8. To promote existing project(s) to developers and attract contribution
- 1.9. To introduce the concept of Free Software/Open Source the general public and welcome new comers
- 1.10. To facilitate documentation

Figure 5-19 Results of Round 1 Sorted by Questions in Short Form

Result for Question 1



Responses Shown

[Hide Responses](#)

[Go back to the List of Questions](#)

[Go to Question 2](#)

1. What are the objectives of an IFHOSP site?

- 1.1. To enable distributed software development for developers from different geographic locations
 - [Mark](#) [to facilitate shared, concurrent, version-controlled development of source code and documentation by multiple developers](#)
 - [Jacob](#) [Provide common infrastructure for distributed software development. Including but not limited to version, control, bug management, mailing lists](#)
 - [Schulhoff](#) [Support development of Open Source Software.](#)
 - [Alvin](#) [To provide technical tools \(such as CVS\) to aid in development](#)
 - [Austin](#) [Serve as central site for recruiting and organizing contributors.](#)
 - [Austin](#) [Provide machine and/or human resources to build and distribute product releases.](#)
 - [Jason](#) [To provide tools to support the community of users that develop and use the software on that site.](#)

7 Responses
- 1.2. To support concurrent and collaborative software development
 - [Terence](#) [To provide a means of storing and updating source code](#)
 - [Mark](#) [to facilitate shared, concurrent, version-controlled development of source code and documentation by multiple developers](#)

Figure 5-20 Results of Round 1 Sorted by Questions in Long Form

In the presentation sorted by questions, the reader of the results could choose to examine the results in short or long form. In short form, only the summarised statement of the data were shown (Figure 5-19). Alternatively, the related opinions from the participants were also shown in long form (Figure 5-20).

Participants of Round 1 of Infrastructure For Hosting Open Source Project (IFHOSP) Delphi Survey



In this page, the results are sorted by individual participants. They are grouped by their own self rating.

Grouped by Self Rating

Participants were asked to rate themselves on three questions?

- Do you think you are an expert user in IFHOSP sites (Expert User)?
- Are you an administrator of an IFHOSP site (Administrator)?
- Have you ever published any article on the topic of Open Source in an academic publication (Academic)?

Three sets of yes/no answers were obtained and participants were classified into eight categories according these answers:

Expert User	Administrator	Academic	Expert User +Administrator	Expert User +Academic	Administrator +Academic	Expert in all 3 groups	Not an Expert in any group
Austin Luke Mark Michael Alvin William Garrett	Leslie	Joseph Schulhoff Patrick Joanne Phil Gabriel	Troy Jason	Terence Noah		Brendan Eugene	Jacob Dave



[Back to Round 1 Results Main Page](#)

Figure 5-21 Participants Grouped by Self Rating

In the presentation sorted by participants, they were grouped by their self-ratings (Figure 5-21). Nicknames were used to ensure anonymity while responses from any particular person can be grouped and identified (Turoff & Hiltz 1996).

Answer from Mark



Summary Hidden

[Show Summary](#)

[Go to List of Participants](#)

Mark regards himself/herself as:

- An expert user in IFHOSP sites

1. What are the objectives of an IFHOSP site?

- 1.1 to facilitate communication between geographically disparate developers
- 1.2 to facilitate shared, concurrent, version-controlled development of source code and documentation by multiple developers
- 1.3 to reliably archive communication, documentation, and source code for retrieval by the public
- 1.4 to provide automated building and testing facilities
- 1.5 to act as a central location for developers to discover, browse and select from existing code bases, rather than rewriting.

2. What tools can be employed on an IFHOSP site and what are the important features and usability factors for each of them?

Tool 1: cvs

2.1 Brief Description

concurrent version system; client/server source control system

2.1.2 What are the important features of this tools?

- 2.1.2.1 established standard, widely available and understood
- 2.1.2.2 client/server, works over a network
- 2.1.2.3 permits concurrent development without locking

2.1.3 What are the important usability factors of this tools?

- 2.1.3.1 Learnability : It is easy and intuitive to learn how to operate the site
- 2.1.3.2 Memorability : It is easy to remember how to operate the site
- 2.1.3.3 Simple and natural dialogue : Interaction between computer and users is as simple as possible and information in the interaction is presented in natural and logical fashion
- 2.1.3.4 Speak the users' language : During the interaction between computer and users, terms and concepts that are familiar to users are used rather than system-oriented terms

Figure 5-22 Results of Round 1 by Participants in Short Form

Answer from Mark



Summary Shown

[Hide Summary](#)



[Go to List of Participants](#)

Mark regards himself/herself as:

- An expert user in IFHOSP sites

1. What are the objectives of an IFHOSP site?

- 1.1 to facilitate communication between geographically disparate developers
[To facilitate communication between developers](#)
- 1.2 to facilitate shared, concurrent, version-controlled development of source code and documentation by multiple developers
[To enable distributed software development for developers from different geographic locations](#)
[To support concurrent and collaborative software development](#)
[To facilitate documentation](#)
- 1.3 to reliably archive communication, documentation, and source code for retrieval by the public
[To provide an archive of Open Source/Free Software development related materials to the general public](#)
- 1.4 to provide automated building and testing facilities
[To include automated building and testing facilities in releases](#)

Figure 5-23 Results of Round 1 by Participants in Long Form

In the presentation sorted by participants, the reader could choose to examine the results in two forms. While in short form, only the opinion of the participant was shown (Figure 5-22), in long form, the related summarised statements were also shown (Figure 5-23).

The long form presentation also took advantage of hyperlinks. For example, in the long form, answer sorted by participants page (Figure 5-23), if the reader clicked the summarised statement 'To support concurrent and collaborative software development' under response 1.2, this would lead to the corresponding statement in answer sorted by questions page (Figure 5-20). The reader could then find out who were the participants that made similar comments. This hyperlink action could also work in the reverse direction. If the reader selected the opinion of participant 'Mark' (Figure 5-20), for example 'to facilitate shared, concurrent, version-controlled development of source code and documentation by multiple developers', this

would lead to the corresponding opinion in answer sorted by participants page (Figure 5-23). The nicknames were also hyperlinked to the corresponding participants' page. This design was implemented to give the reader a more thorough understanding of the data. As the participants were all potential readers of these results, these results were also designed to maximise their understanding of the subject under investigation as well as understanding other participants' point of views.

Q3 What work practices and culture should be promoted?



In this page, you can choose the degree of relevancy for the summarised answer(s) given. If you have opinion on certain answers, please give your opinion in the text boxes provided. The arrangement of the summarised answer(s) is randomised. You can find a particular answer by searching for its number.

< Save and Return

	Extremely Relevant			Totally Irrelevant				No Comment
	1	2	3	4	5	6	7	
3.26 Critique for the sake of the task	<input type="radio"/>	<input checked="" type="radio"/>						

Comment

	Extremely Relevant			Totally Irrelevant				No Comment
	1	2	3	4	5	6	7	
3.36 Standards coding style	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>				

Comment

Figure 5-24 Round 2 Questionnaire Answer Page

In round 2 of the survey, the main task for the participants was to rate the summarised statement quantitatively so that the importance of the statements could be obtained. Participants could also give qualitative comments for any statements (Figure 5-24).

Q3 What work practices and culture should be promoted?



In this page, you can choose the degree of relevancy for the summarised answer(s) given. If you have opinion on certain answers, please give your opinion in the text boxes provided. The arrangement of the summarised answer(s) is randomised. You can find a particular answer by searching for its number.

<- Save and Return

	Extremely Relevant			Totally Irrelevant				No Comment
	1	2	3	4	5	6	7	
3.22 Cooperation and collaboration, encourage involvement of developers to share the load of development	<input type="radio"/>							
Comment	<input type="text"/>							
3.30 Easy to use, high usability	<input type="radio"/>							
Comment	<input type="text"/>							

Figure 5-25 Randomisation of Statement Order

Figure 5-25 was the page generated for a different participant. The order of the statements for each participant was different as any arrangements of statements may cause bias to the results. A randomisation mechanism was thus implemented in the hope to minimise this bias.

In the presentation of the results of round 2, more sorting orders were provided than round 1. First, in the sorting by questions category, there were four different arrangements, namely 'Only Top Ten' (Figure 5-26), 'Only Numerical Data' (Figure 5-27), 'All Relevant Data (Sort by Rating)' (Figure 5-28) and 'All Relevant Data (Sort by Controversy)' (Figure 5-29).

Result for Question 1



Show Only Top Ten

[Show Only Numerical Data](#)

[Show All Relevant Data \(Sort by Rating\)](#)

[Show All Relevant Data \(Sort by Controversy\)](#)

[Go back to the List of Questions](#)

[Go to Question 2](#)

1. What are the objectives of an IFHOSP site?

		Average		No. of
		Totally Irrelevant	Extremely Relevant	
1	To facilitate communication between developers		1.4	12
2	To support concurrent and collaborative software development		1.5	11
3	To allow potential developers to contribute to projects		1.6	11
4	To enable distributed software development for developers from different geographic locations		1.3	12
5	To facilitate communication between developers and users (of Free Software/Open Source software)		1.9	12

Figure 5-26 Show Only Top Ten

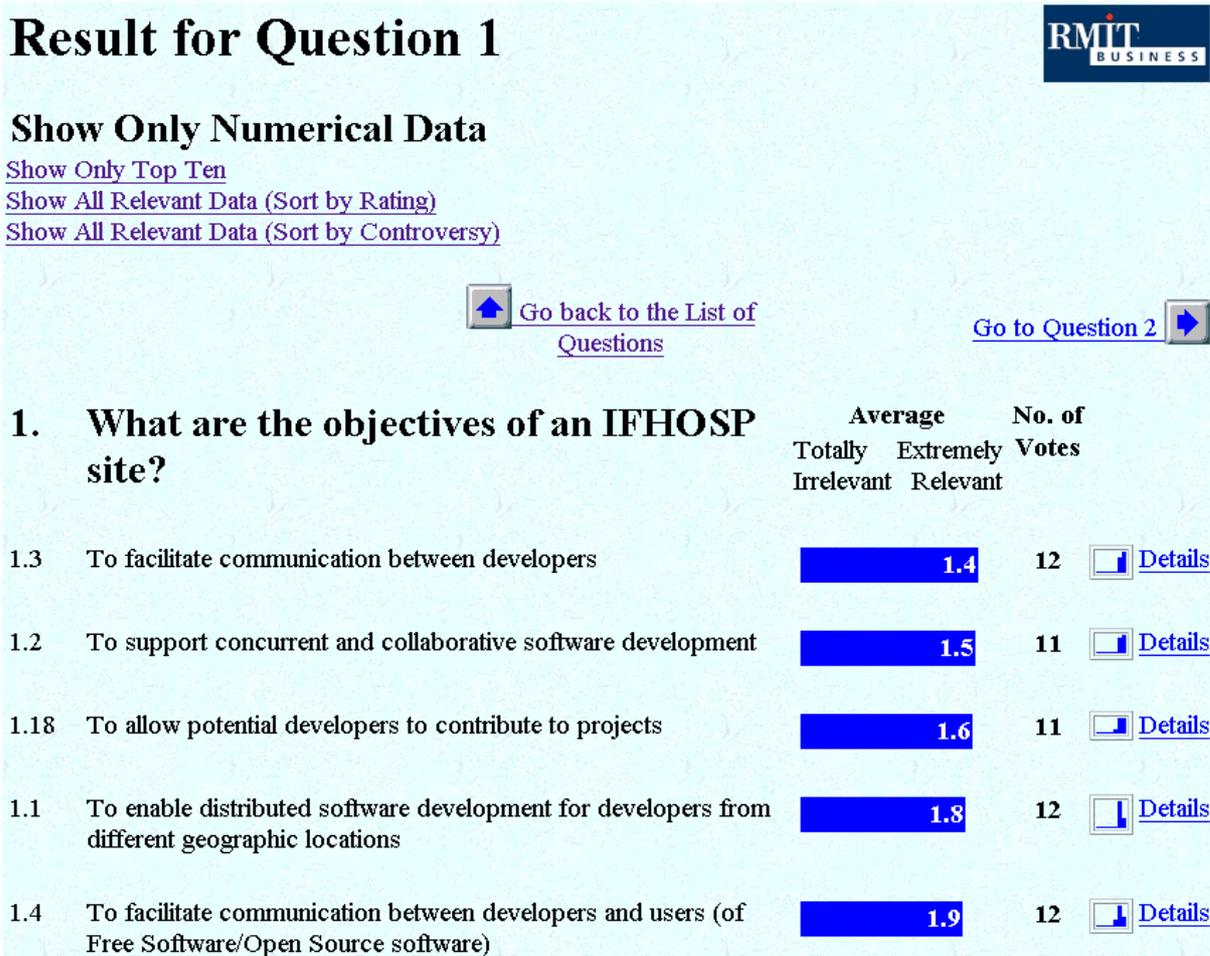


Figure 5-27 Show Only Numerical Data

The differences of 'Only Top Ten' and 'Only Numerical Data' were that all statements were presented in 'Only Numerical Data' and the reader could select 'Details' on the right of the miniature distribution graph to find out the particulars.



Result for Question 1

Show All Relevant Data (Sort by Rating)

[Show Only Top Ten](#)

[Show Only Numerical Data](#)

[Show All Relevant Data \(Sort by Controversy\)](#)

[Go back to the List of Questions](#)

[Go to Question 2](#)

1. What are the objectives of an IFHOSP site?

Average		No. of
Totally Irrelevant	Extremely Relevant	

1.3 To facilitate communication between developers

1.4	12	Details
-----	----	-------------------------

This sub-question is summarised from the following answer(s) from round 1

Terence [To facilitate knowledge sharing and discussions among developers \(e.g. via discussion groups and a repository for design documents, etc.\)](#)

Mark [to facilitate communication between geographically disparate developers](#)

Alvin [To make communication between developers fast and easy. \(Mailing lists, possibly instant messaging, web forums, etc\)](#)

Austin [Provide a forum for project-related discussions.](#)

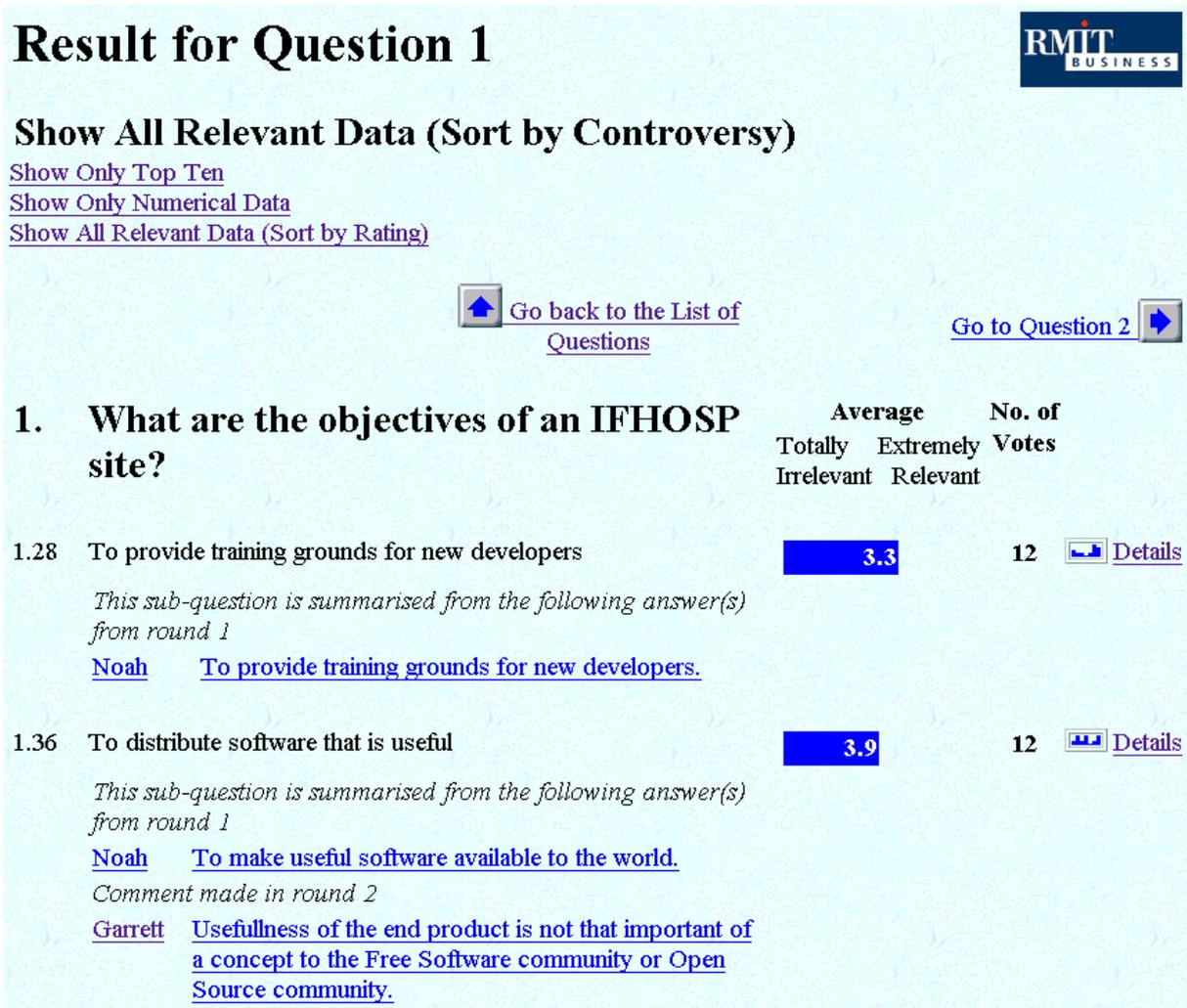
Noah [To provide a forum for communication between developers of Open Source software.](#)

5 Responses

Comment made in round 2

Joseph [i think is nearly a duplicate to some above](#)

Figure 5-28 Show All Relevant Data (Sort by Rating)



Result for Question 1

Show All Relevant Data (Sort by Controversy)

[Show Only Top Ten](#)
[Show Only Numerical Data](#)
[Show All Relevant Data \(Sort by Rating\)](#)

[Go back to the List of Questions](#) [Go to Question 2](#)

1.	What are the objectives of an IFHOSP site?	Average		No. of
		Totally Irrelevant	Extremely Relevant	
1.28	To provide training grounds for new developers <i>This sub-question is summarised from the following answer(s) from round 1</i> Noah To provide training grounds for new developers.	3.3		12 Details
1.36	To distribute software that is useful <i>This sub-question is summarised from the following answer(s) from round 1</i> Noah To make useful software available to the world. <i>Comment made in round 2</i> Garrett Usefulness of the end product is not that important of a concept to the Free Software community or Open Source community.	3.9		12 Details

Figure 5-29 Show All Relevant Data (Sort by Controversy)

Qualitative data such as comments in round 1 and 2 were shown in two 'All Relevant Data' presentation style. Similar to round 1, comments and nicknames were hyperlinked. The sorting order of '(Sort by Rating)' was by average rating of each statements and '(Sort by Controversy)' was by the variance of the rating each statements.

Chart for Question 1.36

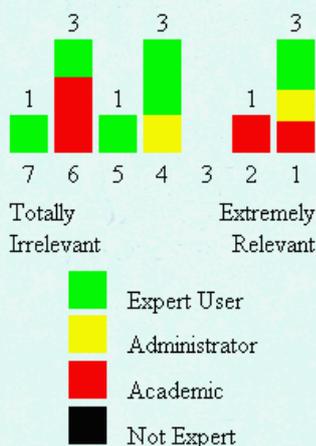


Identities Shown

[Do not show identities](#)

[Back to Question 1.36](#)

1.36. To distribute software that is useful



Average = 3.9
No. of Votes = 12

Participants	Expert User	Administrator	Academic	Not Expert	Totally Irrelevant	Extremely Relevant	Comment
Gabriel			1		6		
Brendan	1	1	1			1	
Matthew	1	1			4		
Garrett	1				7		<i>Usefulness of the end product is not that important of a concept to the Free Software community or Open Source community.</i>
Chris	1				4		
Phil			1			2	
Mark	1				6		
Alvin	1				5		
Noah	1		1			1	
Joseph			1		6		
John	1	1				1	
Neil	1	1			4		

Figure 5-30 Detail Chart of Distribution of Responses

The reader could select 'Details' on the right of the miniature distribution graph to find out the particulars (Figure 5-30). Each group of experts was given a colour in the distribution graph

and thus the distribution of opinions of different groups could be observed. Responses and comments from round 2 from individual participants were also listed and nicknames were hyperlinks for the reader to discover the participants' other responses.

Answer from Chris for Round 2 of Infrastructure For Hosting Open Source Project (IFHOSP) Delphi Survey



[Go to List of Participants](#)

Chris regards himself/herself as:

- ◆ An expert user in IFHOSP sites

1*. [What are the objectives of an IFHOSP site?](#)

2*. [What tools can be employed on an IFHOSP site and what are the important features and usability factors for each of them?](#)

3*. [What work practices and culture should be promoted?](#)

4*. [What are factors that motivate users to use an IFHOSP site?](#)

5*. [What are barriers that prevent users from using an IFHOSP site?](#)

6*. [What are the positive results for users in using an IFHOSP site?](#)

7*. [What are the negative results for users in using an IFHOSP site?](#)

8. [What are factors that motivate administrators to setup or maintain an IFHOSP site?](#)

9. [What are barriers that prevent administrators from setting up or maintaining an IFHOSP site?](#)

10. [What are the positive results for administrators in setting up or maintaining an IFHOSP site?](#)

11. [What are the negative results for administrators in setting up or maintaining an IFHOSP site?](#)

12*. [What are other important issues in IFHOSP?](#)

* denotes that the participant responded to that question.

[Go to List of Participants](#)

Figure 5-31 Responses of a Participant

In the sorting by participant presentation style, the answers of each participant were divided into the corresponding twelve questions. Asterisks were placed at the questions that the participant chose to respond (Figure 5-31).

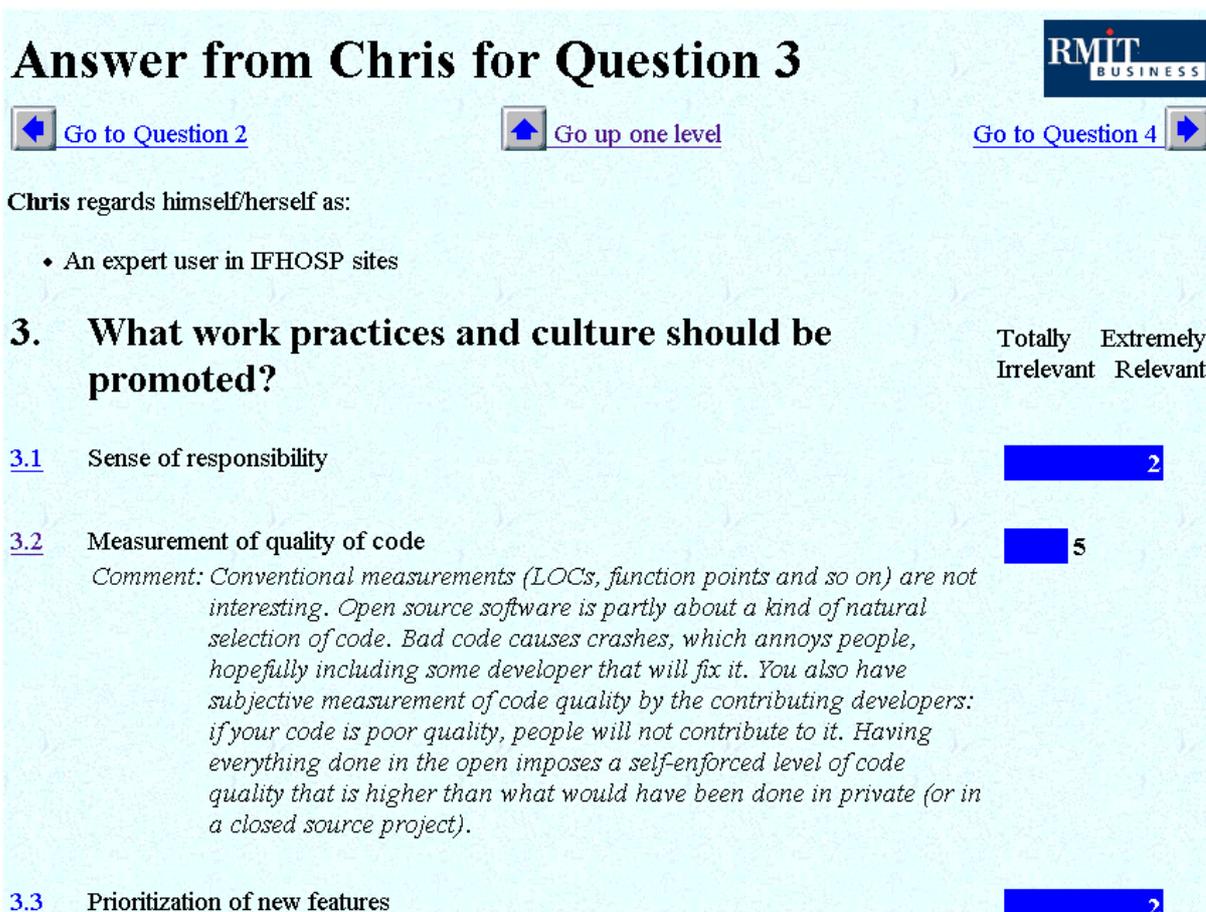


Figure 5-32 Detail Responses of a Participant

Within each question, quantitative data as well as qualitative were presented (Figure 5-32). The statement numbers were hyperlinked to encourage the reader of the results to explore responses from other participants.

Round 3 of the Survey was very similar to round 2 except that the results of round 2 was feedback to the participant. In order to enhance this feedback mechanism, there were several changes to the web interface.

Round 3 Questionnaire



This is the questionnaire for round 3. Again, 12 questions are presented.

If you want to give answers to a specific question, please click on the question. Otherwise, please click the no comment box on the right.

Some the question numbers may be highlighted. This indicated that you rated some of the statements in those questions in round 2.

If you need any help, please click [here](#).

Questions	No Comment
1. What are the objectives of an IFHOSP site?	<input checked="" type="checkbox"/>
2. What tools can be employed on an IFHOSP site and what are the important features and usability factors for each of them?	<input type="checkbox"/>
3. What work practices and culture should be promoted?	<input type="checkbox"/>
4. What are factors that motivate users to use an IFHOSP site?	<input type="checkbox"/>
5. What are barriers that prevent users from using an IFHOSP site?	<input checked="" type="checkbox"/>
6. What are the positive results for users in using an IFHOSP site?	<input checked="" type="checkbox"/>
7. What are the negative results for users in using an IFHOSP site?	<input type="checkbox"/>
8. What are factors that motivate administrators to setup or maintain an IFHOSP site?	<input type="checkbox"/>
9. What are barriers that prevent administrators from setting up or maintaining an IFHOSP site?	<input type="checkbox"/>
10. What are the positive results for administrators in setting up or maintaining an IFHOSP site?	<input type="checkbox"/>
11. What are the negative results for administrators in setting up or maintaining an IFHOSP site?	<input type="checkbox"/>
12. What are other important issues in IFHOSP?	<input type="checkbox"/>

If you just want to logoff, please click [here](#).

If you need any help, please click [here](#).

Figure 5-33 Round 3 Questionnaire Question Page

At the question page, the question numbers of the questions that the participant responded in round 2 were given a different colour and an increase in font size (for those who have colour blindness) (Figure 5-33).

Q4 What are factors that motivate users to use an IFHOSP site?

In this page, you can choose the degree of relevancy for the summarised answer(s) given. If you have opinion on certain answers, please give your opinion in the text boxes provided. The arrangement of the summarised answer(s) is randomised.

If you gave answers in the previous round, the answers chosen before will be highlighted among the scale of the ratings.

You can also click on 'Glossary and Responses from previous rounds' to check previous comments and 'Expand Graph' to check the statistic of previous ratings. An extra window will be opened to present this information. Please do not close this window and leave it in the background for quicker response.

[Go to Question 3](#)
[Go up one level](#)
[Go to Question 5](#)

4.8 The tools provided are effective and productive Glossary and Responses from previous rounds	Totally Irrelevant	Extremely Relevant	No Comment
Your Response from last round	7 6 5 4 3 2 1	7 6 5 4 3 2 1	○
Your Response for this round	○ ○ ○ ○ ○ ● ○	○ ○ ○ ○ ○ ○ ○	○
Average Response from all participants from last round	1.5	8 votes	Expand Graph

Comment for this round

Figure 5-34 Round 3 Questionnaire Answer Page

In the answer page, the ratings of the participants' previous answer were also given a different colour and an increase in font size (Figure 5-34). Summary of responses from round 2 were shown. The participants could also explore further into the glossary and the results from previous rounds (Figure 5-35, Figure 5-36 & Figure 5-37).

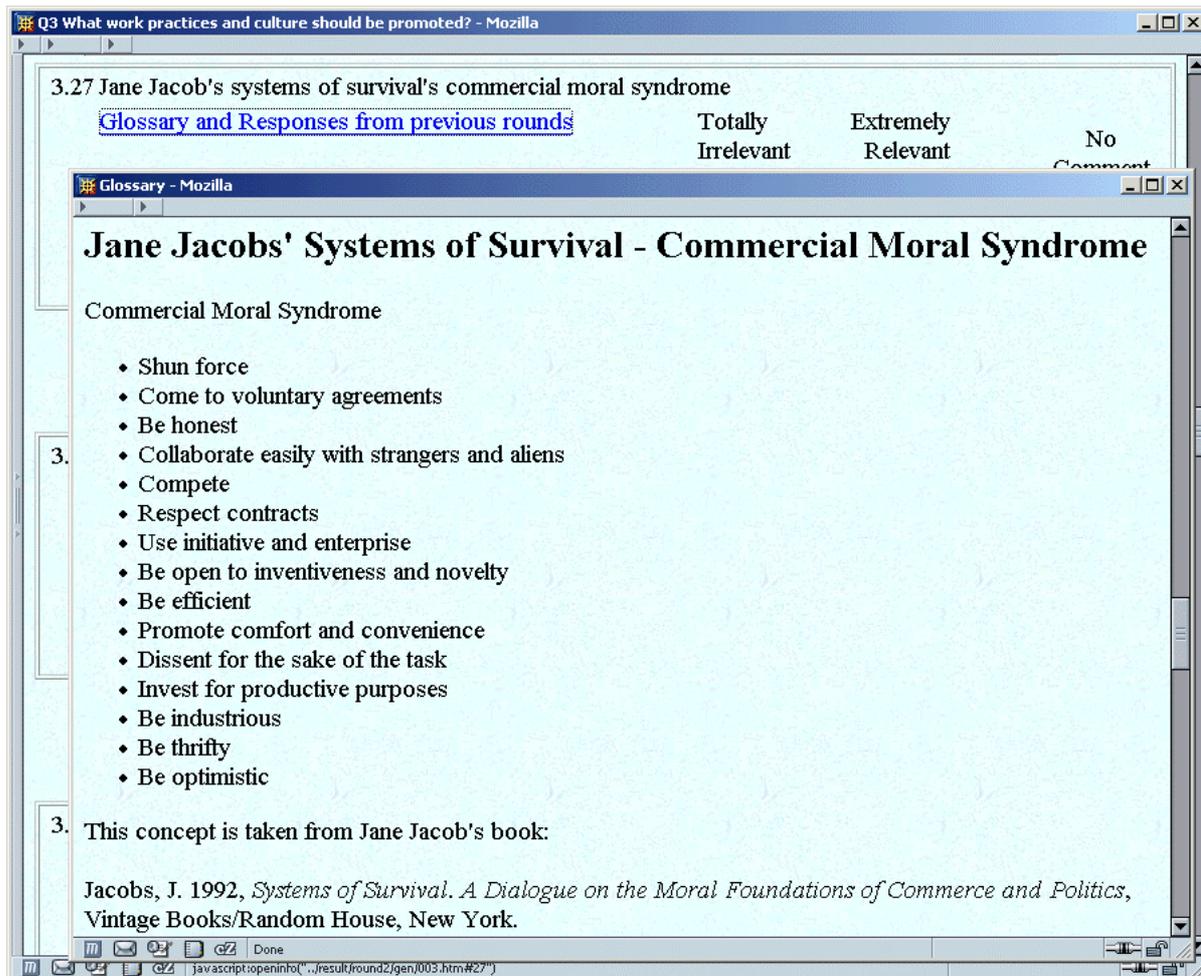


Figure 5-35 Checking Glossary for Difficult Terms

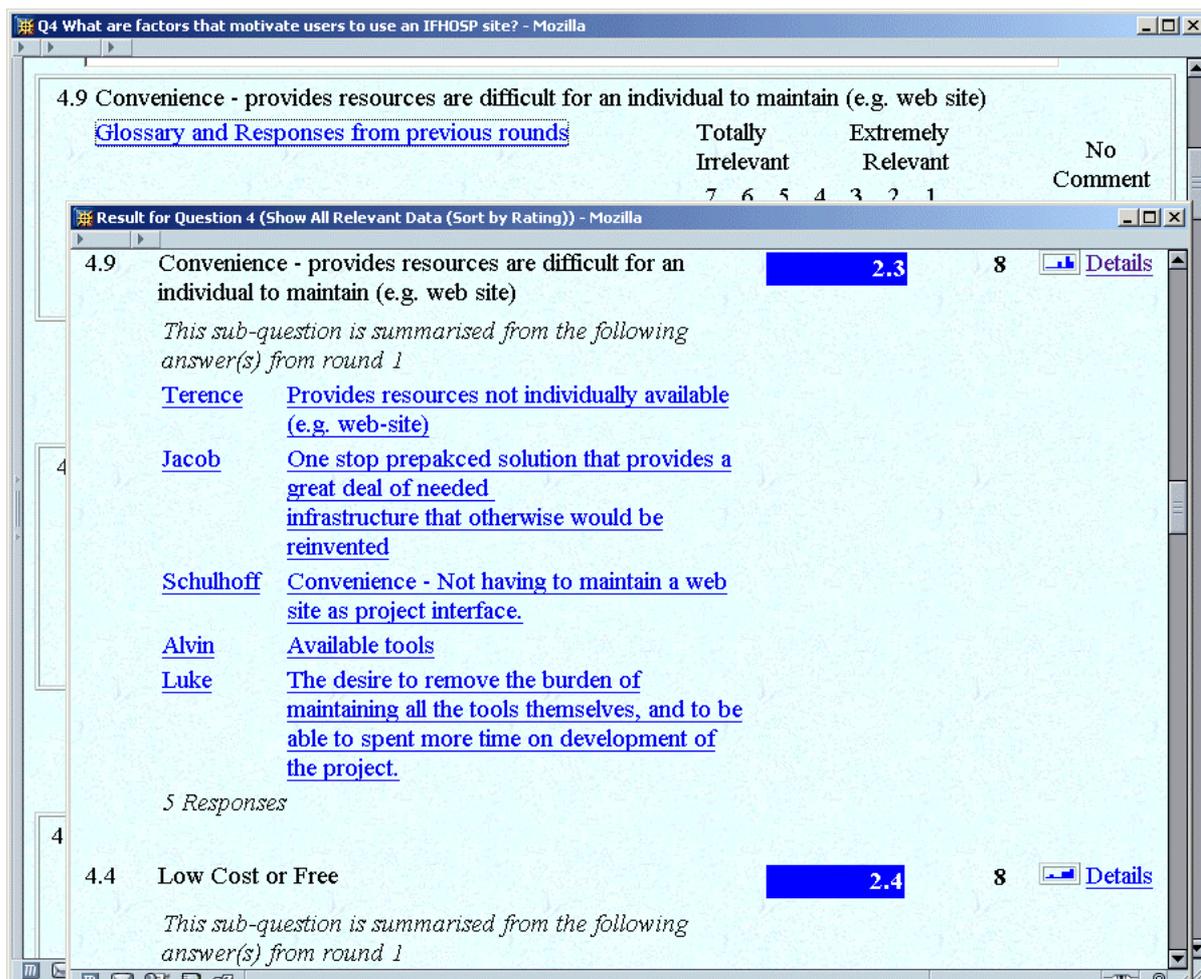


Figure 5-36 Checking Qualitative Results from Last Round

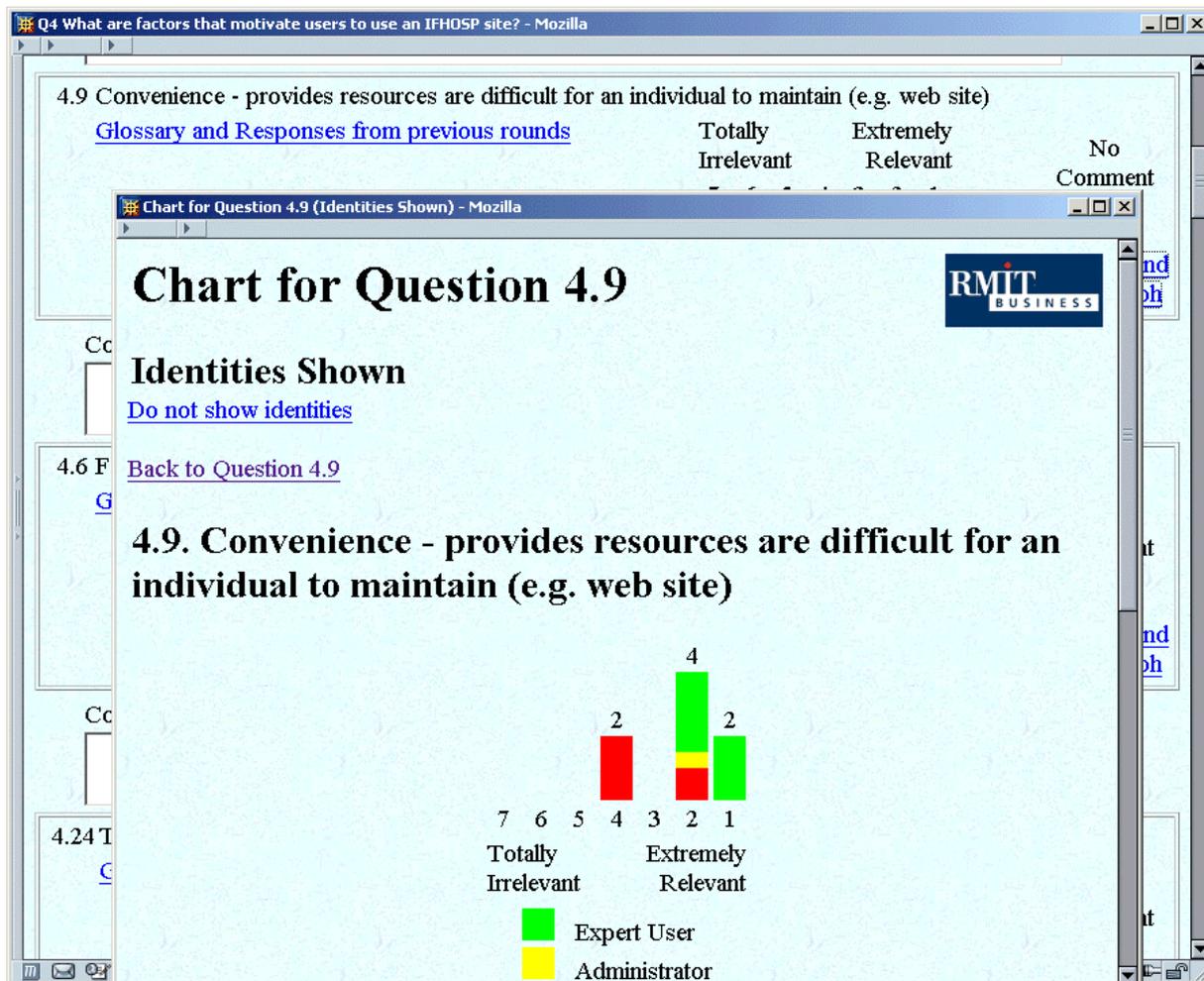


Figure 5-37 Checking Quantitative Results from Last Round

For the presentation of the results of round 3, it was again similar to round 2, except the excerpt of the results in round 2 were also shown (Figure 5-38).



Result for Question 1

Show All Relevant Data (Sort by Rating)

[Show Only Top Ten](#)

[Show Only Numerical Data](#)

[Show All Relevant Data \(Sort by Controversy\)](#)

[Go back to the List of Questions](#)

[Go to Question 2](#)

1. What are the objectives of an IFHOSP site?

Average	No. of
Totally Irrelevant	Extremely Relevant Votes

1.3 To facilitate communication between developers

1.4	15	Details
-----	----	-------------------------

Answer from Previous Round

1.4	12	
-----	----	--

This sub-question is summarised from the following answer(s) from round 1

- [Terence](#) [To facilitate knowledge sharing and discussions among developers \(e.g. via discussion groups and a repository for design documents, etc.\)](#)
- [Mark](#) [to facilitate communication between geographically disparate developers](#)
- [Alvin](#) [To make communication between developers fast and easy. \(Mailing lists, possibly instant messaging, web forums, etc\)](#)
- [Austin](#) [Provide a forum for project-related discussions.](#)
- [Noah](#) [To provide a forum for communication between developers of Open Source software.](#)

5 Responses

Comment made in round 2

[Joseph](#) [i think is nearly a duplicate to some above](#)

Figure 5-38 Results of Round 3



Post-Delphi Survey

This is a simple survey to collect opinions on possible improvements of the previous survey. Please place responses to questions in each of the text box provided. The length of the comments are not limited by the size of the text boxes as they will expand accordingly. You can also select the "no comment" checkbox to indicate that you do not want to comment.

1.How has the Delphi process facilitated (or not facilitated) communication between participants?

No Comment

I am doing this survey by myself. Where does communication between participants come from?

2.Do you find participation in this survey a worthwhile experience?

No Comment

3.What can be improved in the process of the survey?

No Comment

4.What can be improved in the web interface of the survey?

No Comment

More Responsive

5.What can be improved in the questions posed in the survey?

No Comment

Less Questions

6.Other Comments

No Comment

Save Abort

Figure 5-39 Post-Delphi Survey

The Post-Delphi survey was a simple survey to gain understanding of the quality of the web Delphi survey method (Figure 5-39). Six simple questions were asked:

1. How has the Delphi process facilitated (or not facilitated) communication between participants?
2. Do you find participation in this survey a worthwhile experience?
3. What can be improved in the process of the survey?
4. What can be improved in the web interface of the survey?
5. What can be improved in the questions posed in the survey?
6. Other Comments

Result for Post-Delphi Survey



This informal survey was conducted from 2 to 25 Oct 2002. It was aimed for those who responded to the survey but all who were invited were informed in the hope of getting information on why they did not respond. Only two responded and both of them had participated in the previous survey. One responded via web interface while another responded via email.

The responses are listed below:

1. How has the Delphi process facilitated (or not facilitated) communication between participants?

Garrett I don't know any other participants, so how could it facilitate discussion with any particular one?

Schulhoff I only found time to participate in two rounds and I dont see how the process facilitated communication. I had zero interaction with other participants.

2. Do you find participation in this survey a worthwhile experience?

Garrett Somewhat. I think I got burnt out answering it sometimes. Even though there was the ability to answer the survey in pieces, I would still answer it in all one shot, for whatever reasons, be they technological or psychological. And it's a long survey to do that for.

Schulhoff Sorry to say: No. The problem is that the survey is rather time consuming and I dont feel that I get anything in return.

Figure 5-40 Post-Delphi Survey Results

The presentation of the results of the Post-Delphi survey was also very straightforward. It was just a page with all the comments with the nicknames of the participants on the left hand side (Figure 5-40).

The Delphi survey was completed after the Post-Delphi survey and the analysis of the results will be presented later in chapter 6. The actual results can be found on the CD-ROM enclosed (/Delphi/result/index.html) and the reader can browse through it to gain a better understanding of the description above.

5.3.2.5 Data Analysis

Several data analysis techniques can be employed in a Delphi survey. The first technique was to summarise the results from round 1 and express the concepts in the results into statements. The next technique was to process the quantitative data from the rating of the statements in round 2 and 3.

The summarising process that was adopted for round 1 was qualitative. Firstly, responses were broken into unit concepts. Then, every concept was related to the corresponding response(s) and vice versa using hyperlink(s) on the round 1 results pages. This ensured that every concept originated from at least one of the responses and every response was summarised. The researcher tried to make every unit concept self-contained and mutually exclusive to other concepts, but this could not be achievable with every concepts. After the summary process, it was verified and clarified by the participants.

In rounds 2 and 3, both quantitative and qualitative data was collected. On the quantitative side, ratings out of a 1-7 scale were obtained. The average and variance rating for each statement were computed. Variance was chosen over standard deviation because the built-in the standard deviation routine was not yet implemented in PostgreSQL 7.0.2 and thus calculating variance from $\overline{x^2} - \bar{x}^2$ explicitly was the easiest method for comparison purposes. As stated in sub-section 5.3.1.5, the calculation for a scale implied that the distance between each subjective scale is equal. Though it may not be proven, the benefits of the statistics obtained will be substantially higher than not to have the calculation process at all.

Different methods were proposed to select statements as the conclusion of the survey, namely ranking, first interquartile and standard deviation. The ranking method first arranged the statements in the order of their average ratings and took an arbitrary number of statements from the top, for example top 10 statements (Jillson 1975). The interquartile method also first arranged the statements in the order of their average ratings. Then they were broken into four quarters according to their average ratings. The statements in the quarter with the top ratings were taken (Jones, C. G. 1975; Ludlow 1975). The standard deviation method assumed that the ratings formed a normal distribution and the top statements beyond one standard deviation from the average of all average ratings were important (Scarlett 2001). It could be argued that interquartile method and standard deviation method were similar, as for normal distribution, 68% of the ratings would fall between the distances of plus or minus one deviation. Therefore the important statements were the top 16% of all the statements. The difference for the interquartile method was that the top 25% was taken.

To choose from the above methods, recall that the Delphi survey was conducted as an exploratory study, and one of the aims was to construct a mental picture of the situation (Neuman, Bondy & Knight 2003). A more lenient 33% top statements criterion, or first third of the statements, was chosen to include more statements in order to achieve the objective.

On the other hand, the variances of the ratings were computed to discover the polarisation of the opinions. This method was employed in several surveys before (for example (Ludlow 1975)). Again the top 33% controversial statements would be selected.

For the qualitative part of the data from rounds 2 and 3, the comments would be grouped and interpreted according to the content. This interpretation exercise again was aimed at

constructing a mental picture of the situation (Neuman, Bondy & Knight 2003). On the other hand, as the guiding methodology of the study was positivism, the researcher would balance the need for objectiveness and the potential for the discovery of new knowledge from further interpretation of the data.

Other additional data such as 'No Comment' and the results from the Post-Delphi survey will also be collected and discussed to assess the quality of the survey conducted.

5.3.3 Detailed investigation on External Hosting Sites

After the data collection in Delphi survey, one of the analyses was done by categorising the concluding statements into the four classes of software evaluation, namely intrinsic, utility, usability and context. The amount of data collected on the class of utility was not substantial enough. Recalling that the users of the evaluation model may be Free/Open developers and new comers, an emphasis on the category of utility was suggested to meet their expectations. A further investigation into the features of the different tools available on a FOSPHost was thus conducted.

As resources were limited, the area for detailed investigation needed to be designated to the most appropriate topics. As explained in sub-section 3.2, FOSPHost sites can be classified as external hosting and self-hosting. External hosting sites were chosen because a number of them actively promote the FOSPHost aspect of the site. For example, the free version of SourceForge can be seen as a demonstration and an advertisement to the commercial world for the improved Enterprise version of SourceForge as well as the concept of FOSPHost. For self-hosting site such as the FOSPHost site for Linux (Kernel.Org), the main focus is on Linux and the FOSPHost site is probably just to get the project co-ordinated. Therefore, external hosting sites probably are better known and users of the evaluation model may identify the concept of FOSPHost with them more readily. One may argue that the obvious may not be the

necessary condition for direction in research, but the counter argument is to explore what is familiar first and the chance of finding promising research area will increase. An understanding of the basic facts could build a solid foundation for the next stage of research.

The method employed for this part of the research was similar to the preliminary stage of case study method (Yin 1994). Usually case study is applicable in 'how' and 'why' questions on contemporary events. The researcher also does not need to have control over the behaviour of the events. Though 'how' and 'why' questions are the focus of case study method, 'what' questions are usually asked at the preliminary stage so that relevant data was collected to construct answers for the 'how' and 'why' questions. Under the limitation of this research, the 'what' questions, which the basic facts on the topics, were the focus. Some attempts to answer the 'how' and 'why' questions were made, but it was regarded as secondary.

In the design of a case study, Yin (1994, p. 20) commented that 'what questions to study, what data are relevant, what data to collect, and how to analyse the results' were the relevant areas. The detail description on the process of the method below will be adopted these four areas as a framework of discussion. The method for the investigation on external FOSPHost sites will be presented first.

Recalling that the focus of the investigation on external FOSPHost sites should be on the utility of the sites. The aspects for investigation posed are:

- What sites are relevant to the investigation?
- What features are offered on the sites investigated?
- What categories could be given to the sites and the features?
- What features do each site offered?
- What are the background of the sites and policies do they adopted?

- How do the facts collected relate to the Delphi survey results?

The criteria for the selection of external FOSPHost sites for investigation were based on the definition of FOSPHost. Recalling that a FOSPHost site is the infrastructure that supports and co-ordinates the development of Free/Open Source software projects on the Internet, the first criterion is Free/Open Source projects are hosted on site. The next criterion is that the site welcomes the hosting of Free/Open Source projects from other parties. This then fulfils the condition of an external FOSPHost site. The next criterion is that it supports and co-ordinates the development of projects hosted. From the Delphi survey, the most important tool for a FOSPHost site was a source code repository. The criterion is then devised that the site should as least include a source code repository with basic version control capability. The criterion of version control capability is added so FTP sites are not included. The scope of the study can thus be narrowed down to a manageable size. On the other hand, this criterion is also broad enough to fulfil the purpose of an exploratory study.

Data related to the aspects of investigation were collected by visiting the FOSPHost sites via the Internet, reading documents and source code of the sites. If possible, administrators of the sites would be asked to clarify issues that could not be understood by the methods stated above. The data collected was tabulated in a comparison table and comments was obtained from authors and administrators of the software investigated. This method did not involve interview and it was less involved than employing a full-scale case study method. It was chosen, as the objective of the research was exploratory. By employing this method, a broad range of data could be collected with less effort, but of course the depth was less.

Since the importance on answering 'how' and 'why' questions were lower, the effort spent on the analysis of the data could be less. Simpler methods were employed and analysis method such

as pattern-matching was not required. As mentioned above, the data collected would be tabulated in a comparison table. Relationships between the data and the Delphi survey would also be explored.

The choice of method could be argued to be compatible with the positivism methodology and inductive strategy of this research. The method described here is based on observations and discussions of objective facts and thus matches the spirit of positivism. The results obtained are also based on empirical data collected with few presumptions. This hence qualifies the method as inductive.

The validity of this method can be argued as similar to case study as it is based on observations, documentations and source code which is regarded as reliable in case studies. Though interviews were not conducted, sites administrators were asked to check for discrepancies on the investigation. Permissions were asked and obtained to make their feedback public as well. As the evaluation model was available to the public as well, there may also be a peer review mechanism to strengthen the validity of the research.

5.4 Summary of Chapter Five

In this chapter, the plan for the research was described in details and the rationale behind the different choices was also explained. The purpose of the research was chosen to be exploratory and the main methodology was positivism. A classification of software qualities for evaluation of Free/Open Source software was built to suit the nature of FOSPHost sites. Different evaluation presentations were also reviewed as potential candidates for the final implementation of the evaluation model. Two data collection methods were also chosen, namely Delphi survey and detailed investigation. Detail execution and data analysis procedures for Delphi survey were explained and the method for the execution of detailed investigation was also described.

In the next chapter, the results of the Delphi survey will be presented and analysed.

Chapter 6

Results and Analysis of the Delphi Survey

6.1 Introduction

In this chapter, the results of the Delphi survey will be presented. The method of data collection, validity and content of the data collected will then be discussed.

6.2 Results of the Delphi Survey

In this section, the procedure on invitations, the survey results and various types of auxiliary data collected will be provided. Quantitative and qualitative presentation of agreed and controversial answers will then be given.

6.2.1 Invitations and Responses

In this sub-section, data that relate to the quality of responses will be presented. These include the invitation process, the number of responses, 'No Comment' responses, feedback mechanism and the results from Post-Delphi survey.

Recalling from previous discussion, to reach acceptable validity, each group should have at least 10 experts giving opinion. About 40 experts were short-listed for each group. Before the start of the survey, a pilot run was conducted in early June, 2001. 12 experts were invited to participate, 4 from each group. Unfortunately, all of them were too busy to make substantial comment.

After three rounds, the numbers of participants involved in each question are listed (Table 6-1):

Questions	Expert Users	Administrators	Academics
1	18	9	17
2	11	5	9
2.1.2	4	2	3
2.1.3	4	2	3
2.2.2	5	3	4
2.2.3	5	3	4
2.4.2	5	3	4
2.4.3	5	3	4
2.8.2	5	3	3
2.10.2	5	3	4
2.10.3	5	3	4
3	11	5	12
4	10	4	10
5	10	4	11
6	10	2	8
7	10	2	7
8	6	3	4
9	5	1	3
10	6	2	3
11	5	3	4
12	10	3	9

Table 6-1 Numbers of Participants Involved in Each Question

From the table, ten or more expert users responded to question 1 to 7 and 12 and thus reached the requirement for acceptable validity for the group of expert users. Further implications of these figures to the validity of the survey will be discussed in the next sub-section. Further

breakdown of participants based on expertise and participation in each round are listed (Table 6-2):

Participants	Expertise			Round 1	Round 2	Round 3
Alvin	Expert User			✓	✓	✓
Anthony	Not Expert					✓
Austin	Expert User			✓		
Brendan	Expert User	Administrator	Academic	✓	✓	✓
Brent	Expert User		Academic			✓
Chris	Expert User				✓	
Dave	Not Expert			✓		
Eugene	Expert User	Administrator	Academic	✓		
Gabriel			Academic	✓	✓	✓
Garrett	Expert User			✓	✓	✓
Gary			Academic			✓
Jacob	Not Expert			✓		
Jason	Expert User	Administrator		✓		
Jessica			Academic			✓
Joanne			Academic	✓		
John	Expert User	Administrator			✓	✓
Joseph			Academic	✓	✓	✓
Leslie		Administrator		✓		
Luke	Expert User			✓		
Mark	Expert User			✓	✓	✓
Matthew	Expert User	Administrator			✓	
Michael	Expert User			✓		
Neil	Expert User	Administrator			✓	
Noah	Expert User		Academic	✓	✓	
Patrick			Academic	✓		
Peter	Expert User	Administrator				✓
Phil			Academic	✓	✓	✓
Schulhoff			Academic	✓		✓

Participants	Expertise			Round 1	Round 2	Round 3
Steven			Academic			✓
Terence	Expert User		Academic	✓		✓
Troy	Expert User	Administrator		✓		
William	Expert User			✓		
Total Number of Participants:						
32	19	9	14	22	12	16

Table 6-2 Breakdown of Participants based on Expertise and Participation

From the table, most participants had expertise in more than one area. A few were quite humble not to claim any expertise but most regarded themselves to have some knowledge in at least one area. Many of the participants did not contribute in all three rounds, and this can be shown below (Table 6-3):

Total Round(s) Participated	Number of Participants	Overall Percentage
1	21	66%
2	4	13%
3	7	22%

Table 6-3 Amount of Participation

66% of the participants only contributed in one round and 22% participated in all three. The numbers of people invited for each round are listed (Table 6-4):

Round	Academics	Administrators	Expert User
1	54	33	54
2	53	32	53
3	49	43	53

Table 6-4 Numbers of People Invited for Each Round

The researcher aimed at inviting more than 40 people for each group. Unfortunately, due to an operation error, only 33 administrators were invited. The number of the people invited decreased for later rounds because some of the recipients of the invitation email replied and hoped not to participate. Therefore, they were not invited in later rounds. There was one exception that the number of administrators invited increased in round 3. It was decided in round 3 that, although it was not planned in the survey to recruit more participants after the survey began, having more administrators would definitely improve the quality of the results. Therefore, 12 more administrators were invited. Unfortunately, none of them participated and thus the results were not affected by this invitation.

Detail invitation figures are presented below (Table 6-5):

Round	Email		Phone				Login (Persons)	Responded (Persons)
	Invitation	Reminder	Contact Established	Left Message	Cannot Contact	Incorrect		
1	141	136	10	22	15	5	47	22
2	138	131	N/A	N/A	N/A	N/A	18	12
3	147	119	21	33	30	12	26	16

Table 6-5 Statistics for Invitation

For round 1 and round 3, both email and phone invitations were executed. For round 2, only email invitation was sent. The original plan was that phone invitation would be only done in round 1 and the participants should then be aware of the survey. With the decrease in respondents in round 2, phone invitation was again implemented in round 3. The numbers in the table referred to the number of times an action was done except for login and responded. For example, in the 'Cannot Contact' figure, some of the calls were repeated calls to the same

individual. Another figure that cannot be incorporated into the table was the number of participants who verified the summarised statements in round 1. The number was 22, which indicated that all the participants in round 1 verified the statements.

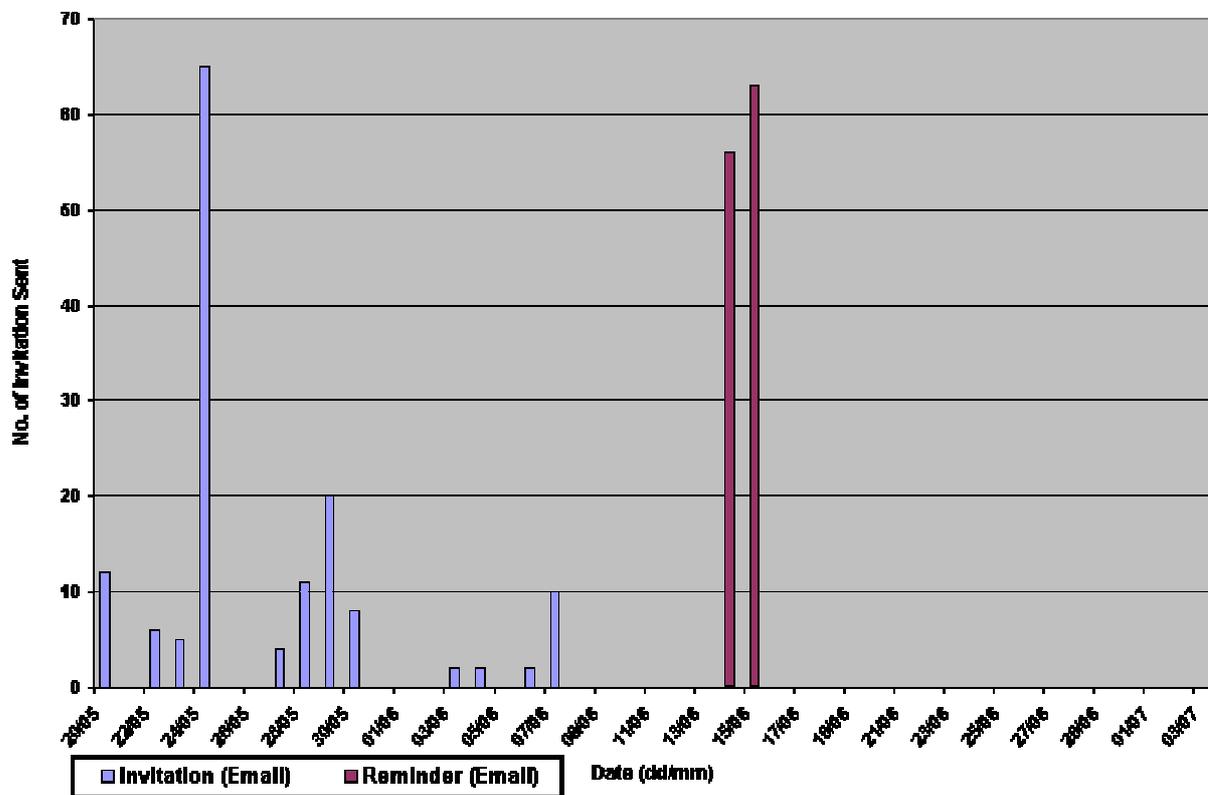


Figure 6-1 No. of Invitation Sent in Round 3

Email invitations and reminders were sent in batches on different dates to disperse the possible workload on the web server (Watt 1999) and the researcher. An example of this strategy can be seen on the distribution of email sent in round 3 (Figure 6-1). For potential participants that could be contacted by phone, they would be called first and then the email would be sent.

As discussed in the previous chapter that 'No Comment' check boxes were provided to obtain a more definite reply on the participants' desire not to response to certain questions.

Round	1	2	3
With Answers	107(36%)	91(63%)	104(51%)
No Comment	101(34%)	13(9%)	9(4%)
No Response	92(30%)	40(28%)	91(45%)

Table 6-6 'No Comment' Responses from Question Page

Table 6-6 refers to 'No Comment' responses collected on the question page (Figure 5-8) where the 12 questions were asked. If the participant went into the answer page of a particular question and gave answers, then it is classified as 'With Answers'. If the participant did not give answer and did not select the 'No Comment' check box, then it is classified as 'No Response'. The figure in the table is the count of the three types of responses to each question on question page.

Round	1	2	3
With Answers	N/A	1904(96%)	2345(89%)
No Comment	N/A	78(4%)	211(8%)
No Response	N/A	3(0%)	75(3%)

Table 6-7 'No Comment' Responses from Answer Pages

Table 6-7 refers to responses collected on the answer pages (Figure 5-24). For each statement on an answer page, the participant could choose to select a rating, select 'No Comment' check box or did nothing. These three actions correspond to the three categories in the table. As in round one, answers given were qualitative and there was no rating involved, therefore, there was no data collected.

Another measurement of quality is the amount of feedback from previous results that a participant received. It could be measured in two ways, references to result pages and changes in responses.

Location of Referral	Number of References	%
Information Centre	6	67.67%
Help	1	11.11%
Directed from Email	2	22.22%
Total	9	

Table 6-8 Round 2 References to Results

Location of Referral	Number of References	%
Information Centre	11	61.11%
Survey Introduction	1	5.56%
Additional Information	5	27.78%
Unknown	1	5.56%
Total	18	

Table 6-9 Round 3 References to Results

In the design of Delphi survey, the results of the previous round(s) were presented to the participants. By taking advantage of the log of the web server, the effectiveness of this mechanism could be measured. The figures in Table 6-8 and Table 6-9 were obtained by examining the Apache web server log at the point after the participant had logged-in. Instances of viewing the result pages were identified and the page accessed just before the viewing of the result pages were obtained. They were the locations of referral to the result pages. Locations of referral identified included information centre, survey introduction, help, additional information and directed from invitation email. After the participants entered from the referral

page to the result page, for round 2, the average pages of results viewed directly after the references were 6.1 and 1.9 pages for round 3.

Another method to examine the effect of feedback is to measure the change in responses between round 2 and 3. 93.30% of the answers were unchanged.

	Difference in Rating	Number of Responses	%	Number of Responses	%
More Important	3	1	0.07%	35	2.58%
	2	4	0.29%		
	1	30	2.21%		
Unchanged	0	1267	93.30%	1267	93.30%
Less Important	1	45	3.31%	56	4.12%
	2	10	0.74%		
	3	1	0.07%		

Table 6-10 Difference in Rating in Round 2 and 3

The opinion of the participants on the design and procedure of the survey could also be discovered from the results of the Post Delphi survey. Only two participants replied. Both of them were unaware of the communication process between participants via the format of the survey. Both felt that the survey was too long but one felt the survey was worthwhile while the other did not like it. The improvements suggested were to shorten the survey and use an interview strategy instead. Technological comments included using no JavaScript in web pages, improvement on check box comment and the bandwidth of the web server to the Internet was low. The design of the interface could be improved as well. One participant also comments that he or she found the results interesting.

In this sub-section, different statistics and data related to the quality of the survey were presented. This data will be further examined in the discussion sub-section below.

6.2.2 Agreed Answers

Within the data collected of the Delphi survey, statements that the participants agreed on were short-listed. These statements will be presented in this sub-section.

According to Table 6-2, there were four participants who only participated in the second round and not the third round. Therefore, it was assumed that the answers the four participants gave in round 2 were their final answers.

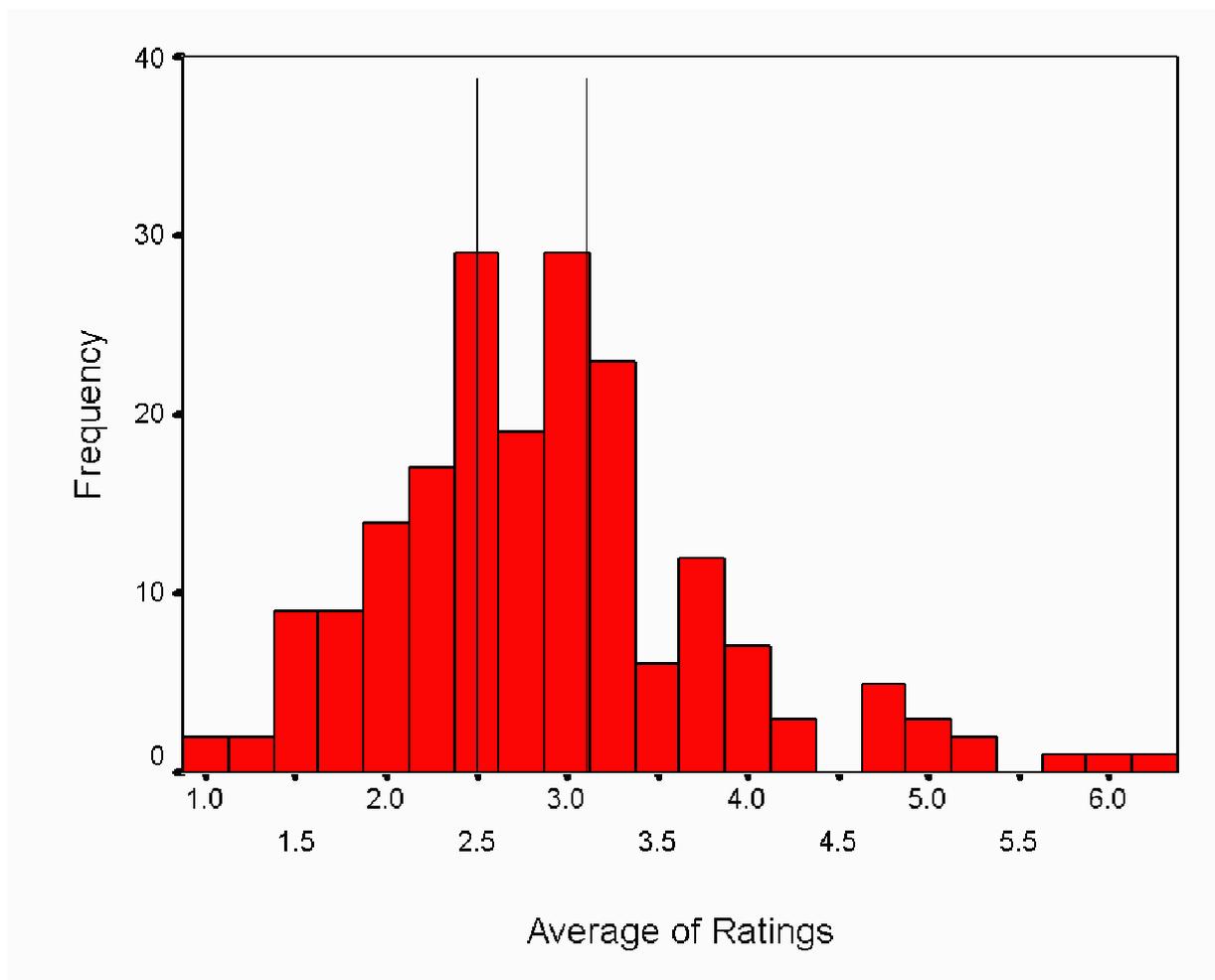


Figure 6-2 Histogram of Average Ratings

The average ratings for statements from questions 1-7 and 12 were charted in a histogram (Figure 6-2) and there were 194 statements. To obtain the statements that the participants agreed as the important issues, the histogram was divided in three equal parts (Table 6-11).

	No. of Statements	Range
Most Important (First Third)	61	Ratings<2.49
Less Important (Second Third)	68	2.49<Ratings<3.112
Least Important (Third Third)	65	3.112<Ratings

Table 6-11 Division of Important Statements

The first third was the most important statements, the second third was the less important and the third third was the least. The number of the statements for the first third was slightly lower as there were 11 statements with the rating 2.5 and it was decided that they would be grouped to the second third.

Under this selection criterion, the 61 most important statements were listed in the following table (Table 6-12). Rating 1 for a statement denotes that it is extremely important while rating 7 means that it is totally irrelevant.

Qn no.	Description	Average
1.	What are the objectives of an IFHOSP site?	
1.2	To support concurrent and collaborative software development	1.4
1.3	To facilitate communication between developers	1.4
1.18	To allow potential developers to contribute to projects	1.5
1.1	To enable distributed software development for developers from different geographic locations	1.6
1.4	To facilitate communication between developers and users (of Free	2.0

	Software/Open Source software)	
1.5	To facilitate cooperation between related parties (programmers, designers, documentation writers, advocates/salesman, etc.)	2.0
1.20	To facilitate software development in a better way	2.2
1.37	The site should be fast and has high availability.	2.4
1.38	To facilitate high levels of communication multiple means	2.4
1.29	To build a sense of community between developers for a project	2.5
2.	What tools can be employed on an IFHOSP site and what are the important features and usability factors for each of them?	
2.1	Source Code Repository	1.1
2.2	Mailing List	1.4
2.5	WWW Server	1.5
2.4	Tracking System	1.8
2.11	Security Measures (e.g. ssh)	2.2
3.	What work practices and culture should be promoted?	
3.27	Jane Jacob's systems of survival's commercial moral syndrome	1.7
3.17	Flexibility towards volunteers	1.8
3.37	Fun and good spirit and hope	1.8
3.35	Clarity, simpleness of code	1.9
3.28	Using centralised repository for source code	1.9
3.29	To include automated building and testing facilities in releases	2.1
3.9	Creating a public library atomsphere, giving users as much freedom as possible and staying out of the users' way	2.1
3.16	Listening to others	2.2

3.20	Openness in attitude, no hidden agenda	2.2
3.21	Openness in procedures and policies	2.2
3.30	Easy to use, high usability	2.4
3.13	Tolerance, respect and patience	2.4
3.1	Sense of responsibility	2.4
3.38	Frequent submissions of contributions	2.4
3.5	Reuse of existing source code	2.4
4.	What are factors that motivate users to use an IFHOSP site?	
4.2	Available 24 hours a day	1.1
4.8	The tools provided are effective and productive	1.4
4.3	Reliable	1.6
4.20	To attract more contributors	2.0
4.9	Convenience - provides resources are difficult for an individual to maintain (e.g. web site)	2.0
4.6	Fast access, responsive (high bandwidth and power server)	2.1
4.4	Low Cost or Free	2.2
4.7	The tools provided are standard and commonly used	2.2
4.16	The site is frequently updated	2.3
5.	What are barriers that prevent users from using an IFHOSP site?	
5.1	Unreliable	1.7
5.5	The opposite of the answers in question 4	2.0
5.7	Counterproductive user interface	2.2
6.	What are the positive results for users in using an IFHOSP site?	

6.2	Increase communication within the developers	1.3
6.13	More reliable than individually hosted servers (e.g. with only dial-up connection)	1.5
6.1	Facilitation of developers to update the source code in the repository directly and reduction of the need to interact with the project leader to change the code	1.6
6.6	Getting people to contribute to the development from all over world	1.8
6.22	Tools with better quality than individually hosted sites	1.8
6.25	Decrease the startup cost of hosting an Open Source/Free Software	2.1
6.17	Facilitating collaboration and reduction of duplicated effort	2.1
6.3	Increase communication between the developers and other parties	2.1
6.20	Obtaining up to date information	2.2
6.26	Decrease the possibility of producing multiple non-synchronized versions of software	2.2
6.5	Decrease time in administration of an IFHOSP site individually	2.2
7.	What are the negative results for users in using an IFHOSP site?	
7.10	Loss control of the choice of tools hosted on site	2.4
12.	What are other important issues in IFHOSP?	
12.4	IFHOSP site should be careful on the usage agreements with users and provide them with enough freedom	1.5
12.11	The acronym IFHOSP is pointlessly obscure	1.9
12.8	Anyone wanting to setup an IFHOSP needs to be aware of the responsibility involved	2.0
12.9	An IFHOSP should be run in an open fashion and users should be well	2.0

	informed	
12.1	Expanding an IFHOSP into multiple mirror sites to increase reliability and obtain more credibility from users	2.2
12.5	Fair to all efforts	2.3
12.10	An IFHOSP should be have up to date information of the site and employ novel techniques	2.4

Table 6-12 The Most Important Statements from the Delphi Survey

A qualitative presentation of question Q1, 3-7 is shown below. Question 2 is omitted, as the statements were short and may not benefit from a qualitative presentation. The issues in question 12 are too diverse to be grouped.

One major theme in the objectives in a FOSPHost site is communication, such as to allow potential developers to contribute, to facilitate effective communication between developers, users and other stakeholders in multiple means and to build a sense of community between developers for a project. Another focuses in the list of objectives include supporting concurrent and collaborative software development in a distributed fashion and implementing a software development process that is above common standard.

Work practices and culture that should be promoted in a FOSPHost site includes attitude such as the commercial moral syndrome of Jane Jacob's systems of survival (see appendix D), flexible towards volunteers, fun, hope and good spirit, listening to others, openness in attitude, procedures and policies with no hidden agenda, tolerance, respect, patience and sense of responsibility. In terms of coding practices and co-ordination, clarity, simplicity of code, using centralised repository for source code, frequent submissions of contributions and reuse of existing source code. The management of a FOSPHost site also should include automated

building and testing facilities in releases and create a public library atmosphere to give users as much freedom as possible and staying out of the users' way. Creating an inviting environment by making the site easy to use is also important.

To motivate developers to use a FOSPHost site, it could acquire qualities such as available 24 hours a day, reliable, fast access, responsive (high bandwidth and power server), convenience (provides resources are difficult for an individual to maintain), low cost or free and frequently updated. The tools provided on the site should be effective, productive, standard and commonly used. The ability of the site to attract more contributors is also another important factor.

Barriers for using a FOSPHost site are factors that are the opposite to the motivation factors including unreliability and counterproductive user interface.

Positive results in users employing a FOSPHost site are improvements in communication and co-ordination. These include the increase in communication within the development team and also between the developers and other parties. Other positive factors are facilitation of developers to update the source code in the repository directly and reduction of the need to interact with the project leader to change the code, obtaining up to date information and decrease the possibility of producing multiple non-synchronized versions of software. Other benefits consist of facilitating collaboration and reduction of duplicated effort and receiving contributions to the development from people all over world.

Other positive results are hosting related. By using a FOSPHost site not hosting by the user, one can obtain services that may be more reliable than individually hosted servers. Tools on

external FOSPHost may also have better quality. Startup cost of hosting and administration time is less comparing to an individually hosted FOSPHost site.

There are a number of negative results but the only statement rated as important is that the control of the choice of tools hosted will be lost if an external FOSPHost is employed.

The implications of the results will be discussed in later sub-sections.

6.2.3 Controversial Answers

Other than the statements that the participants agreed on, there were statements that they strongly disagreed on. These are called controversial statements and they will be presented in this sub-section.

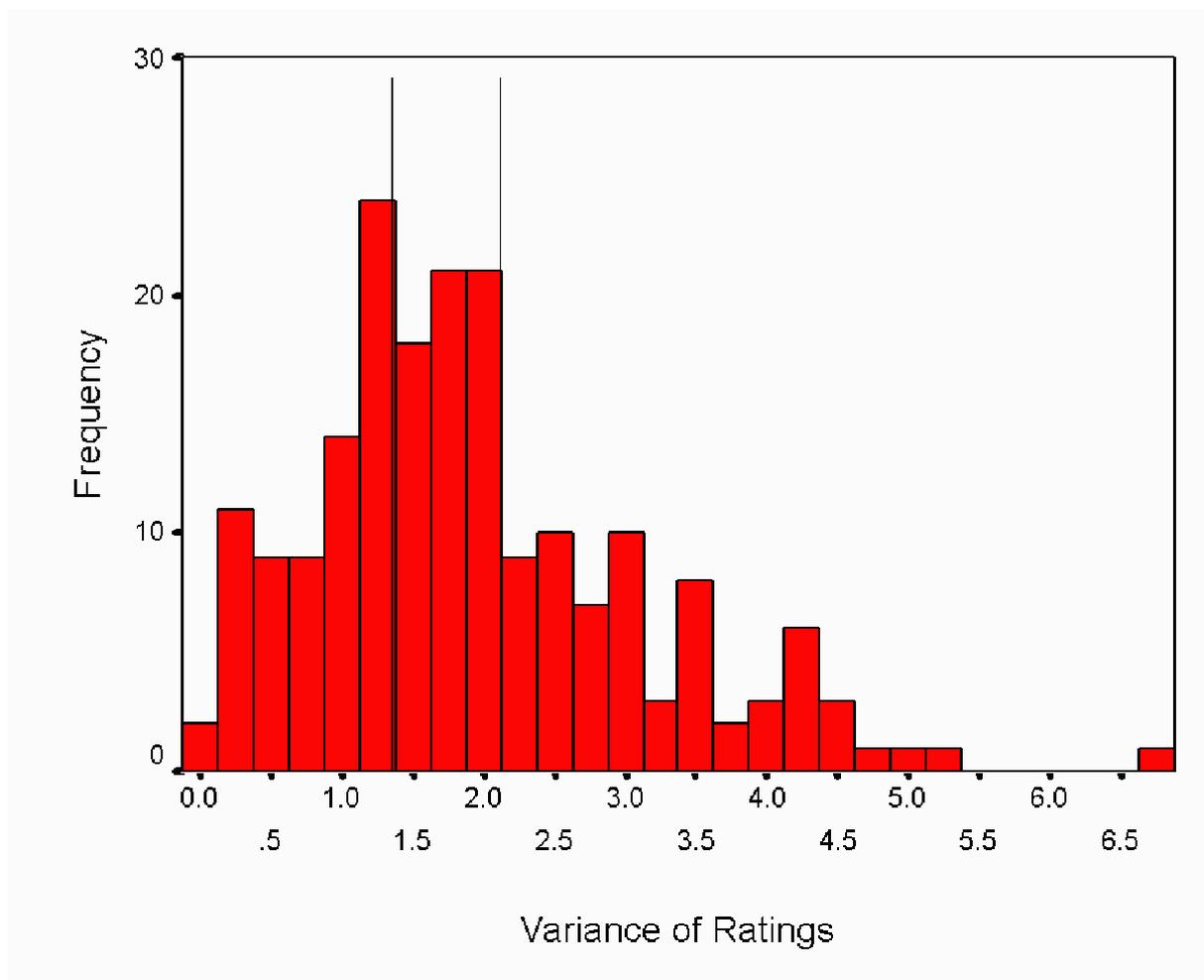


Figure 6-3 Histogram of Variance of Ratings

The variance ratings for statements from questions 1-7 and 12 were charted in a histogram (Figure 6-3) and there were 194 statements. To discover the most controversial statements, the histogram was divided in three equal parts (Table 6-13). Then the first third contained the most controversial statements.

	No. of Statements	Range
Most Controversial (First Third)	65	2.12<Ratings
Less Controversial (Second Third)	64	1.345<Ratings<2.12
Least Controversial (Third Third)	65	Ratings<1.345

Table 6-13 Division of Controversial Statements

Under this selection criterion, the 65 most controversial statements were (Table 6-14):

Qn no.	Description	Avg.	Var.
1.	What are the objectives of an IFHOSP site?		
1.28	To provide training grounds for new developers	3.7	4.5
1.36	To distribute software that is useful	4.1	3.6
1.17	To provide data for research	4.8	3.4
1.27	To facilitate the development of software that is affordable by everyone	3.8	3.4
1.32	To provide tools needed to achieve the objectives of IFHOSP	2.7	3.2
1.16	To serve the Free Software/Open Source community	3.4	3.0
1.24	To attract other Free Software/Open Source projects to come in and host on the site	4.0	2.9
1.31	To provide an archive of Open Source/Free Software development related materials to the general public	3.1	2.9
1.21	To make software with better quality available to the world	3.7	2.6

1.7	To promote existing project(s) hosted on site to users of software	3.2	2.5
1.13	To provide a centralised location for Free Software/Open Source project(s)	3.8	2.1
2.	What tools can be employed on an IFHOSP site and what are the important features and usability factors for each of them?		
2.10	Wiki Wiki Web	3.9	4.3
2.11	Security Measures (e.g. ssh)	2.2	2.9
2.9	Discussion Forum	2.9	2.7
2.12	News Stand	3.7	2.5
3.	What work practices and culture should be promoted?		
3.34	The practices of Extreme Programming	3.6	5.1
3.25	Avoid force	3.7	4.4
3.18	The value of heterogeneity, differences as assets	2.7	3.4
3.36	Standards coding style	3.1	3.1
3.2	Measurement of quality of code	3.2	3.1
3.4	Reinforcing explicit development roles	4.3	3.0
3.19	Nothing should be 'promoted'.	4.8	2.9
3.5	Reuse of existing source code	2.4	2.6
3.14	Awareness of different culture and language background	2.9	2.4
3.13	Tolerance, respect and patience	2.4	2.3
3.11	Distributed style of development and decentralised	2.6	2.3
3.31	Flexibility in tools for rapid project administration	3.3	2.2
3.32	A system to attribute credit	2.8	2.2
3.37	Fun and good spirit and hope	1.8	2.1

4.	What are factors that motivate users to use an IFHOSP site?		
4.12	Many tools are provided	3.1	4.1
4.11	High security	2.7	4.0
4.24	To compete with other projects	5.0	3.6
4.25	Users of software hosted on IFHOSP will use the site	3.0	3.0
4.4	Low Cost or Free	2.2	2.6
4.5	Sufficiently large capacity (e.g. storage, cpu, memory)	2.8	2.5
5.	What are barriers that prevent users from using an IFHOSP site?		
5.14	Intellectual property issues - the host of the IFHOSP may impose some rights to the projects hosted	2.9	4.5
5.11	Not having enough control over the IFHOSP comparing with a local machine	3.3	4.2
5.15	Incompatibility with the format that users had in software development	3.3	4.0
5.20	Difficult to casually browse or some information cannot be accessed without a username	2.6	3.6
5.13	No control the content and the development direction of the IFHOSP	4.0	3.4
5.6	Low storage capacity	3.0	3.3
5.12	Not trusting the host of the IFHOSP	3.2	3.3
5.19	The IFHOSP does not reach a critical mass of users and projects to achieve its advertising function	3.1	3.1
5.5	The opposite of the answers in question 4	2.0	3.0
5.8	The odd urge to pay for software rather than help build it yourself	5.8	2.5
5.3	Low in technical skill, inexperience in using an IFHOSP	3.3	2.5

5.17	Legal issues on software distribution	3.9	2.4
6.	What are the positive results for users in using an IFHOSP site?		
6.24	Interesting parties on a certain project can be found before the project begins	3.8	4.2
6.18	Possibility of reviving dead projects	3.2	3.4
6.16	Greater possibility for getting community credit	3.0	2.7
6.10	Flexibility, being able to select from many ways of doing things	3.3	2.7
6.22	Tools with better quality than individually hosted sites	1.8	2.4
7.	What are the negative results for users in using an IFHOSP site?		
7.8	Low cost in setup will encourage starting unserious projects	4.0	6.7
7.18	Limitation placed on Intellectual Property rights. Depending on the license under which you get to use the IFHOSP, you may suddenly find that you've allowed licensing you hadn't planned.	3.8	5.3
7.7	Low cost in setup will encourage projects to be publish before they are ready	3.8	4.7
7.9	No negative results	4.7	4.3
7.6	Low cost in setup will encourage development of projects that are similar (reinvention of the wheel)	4.2	4.2
7.5	Low cost in setup will encourage forking of projects	4.7	4.0
7.4	Possible extra "collaboration" with unwanted parties ("back-seat programmers", trolls)	3.1	2.8
7.12	Limited tool customisation	2.8	2.8
12.	What are other important issues in IFHOSP?		

12.11	The acronym IFHOSP is pointlessly obscure	1.9	3.7
12.13	A remedy of the administrator(s) of IFHOSP interfering with the development of project(s) is to provide mirroring or withdrawal paths for users if they want to host their projects elsewhere	2.5	3.4
12.7	IFHOSP sites need to focus on maximising productivity	2.9	2.6
12.12	Big IFHOSP are bad (e.g. Freshmeat) Small IFHOSP are good.	5.2	2.3
12.2	Maintaining a commitment to hosting only those projects which are under a sufficiently liberal license.	3.0	2.2

Table 6-14 The Most Controversial Statements from the Delphi Survey

Other than using quantitative data for analysis, qualitative data collected also contained useful clues for the discovery of controversial ideas. Qualitative data were collected in round 1, where the opinions from the participants were summarised into the statements above. Furthermore, in round 2 and 3, they were invited to give comments to these statements. A number of the comments fell on controversial issues as controversies usually attracted discussions. A qualitative examination of the results by grouping similar arguments is thus carried out to present a richer picture of the controversial issues involved. Controversial concepts are obtained, namely 'We love freedom, but how far can it go?', 'What characteristics are admirable in source code?', 'What is a worthy motivation?' and 'Important but not urgent tasks'.

The first controversial concepts is named 'We love freedom, but how far can it go?' One of the answers to question 3, 'What work practices and culture should be promoted?' in round 1 by Alvin was 'They shouldn't promote any particular practice. The heterogeneity of approaches is one of the strengths of the way things are done without these infrastructure sites' (Q3.18, Q3.19). Schulhoff also suggested that 'Acknowledgement of developers different cultural and technical backgrounds as a positive element - very different from corporate monocultures' (Q3.19).

Moreover, the summarised statement for Q3.9 is "Creating a public library atmosphere, giving users as much freedom as possible and staying out of the users' way". These opinions represented the importance for FOSPHost sites to allow freedom and promote heterogeneity.

Another manifestation of the desire for freedom can be seen in discussion of Intellectual Property in Q5.14, Q7.18 and Q12.4. Neil suggested that the administrators of the FOSPHost sites should not own any of Intellectual Property of the project hosted. Mark also suggested that FOSPHost sites should 'provide complete mirroring/withdraw paths for users of the site' (Q12.13) to ensure maximum freedom.

When it comes to management style, 'avoid force' (Q3.25) and 'tolerance, respect and patience' (Q3.13) was proposed. Attitudes such as 'awareness of different technology background' (Q3.15) and 'awareness of different culture and language background' (Q3.14) were also suggested to promote heterogeneity.

Negative effects of heterogeneity and lack of control can be seen in 'the fear of getting flame from other people' (Q5.21) and the 'possible extra "collaboration" with unwanted parties ("back-seat programmers", trolls)' (Q7.4).

Remedies to negative effects could be methods such as 'ability to operate private or semi-private developer groups necessary to avoid "collaboration" with unwanted parties' (Q12.14), 'reinforcing explicit development roles' (Q3.4) which emphasize on a heavier hand on management. Brendan, who proposed the idea of tolerance, patience and avoid force, also suggested the need of firmness (Q3.23). Chris also suggested that if peaceful solutions could not be found, at the end, forceful action such as forking would be needed (Q3.25). Moreover, if there were no mechanism to discern quality, then the natural selection scheme of Free/Open

Source would not function (Q3.13). There was also a comment in response to the statement "Nothing should be 'promoted'" (Q3.19) in round 2 by Garrett:

Hosting sites are great places to promote projects which are successful, so that others will learn about them. Lack of knowledge about what quality projects exist is one of the great faults of the Free Software/Open Source community today.

The next controversial concept was named 'What characteristics are admirable in source code?' Indeed, different participants had different ideas on what characteristics of a piece of software were valuable.

The first important value proposed was 'usefulness'. Statements such as 'to distribute software that is useful' (Q1.36) and 'do not focus on the volume of software created, but usefulness' (Q3.6) were suggested.

The next value was quality. Supporting statements included 'to make software with better quality available to the world' (Q1.21) and 'measurement of quality of code' (Q3.2).

Another value of software was that it could be reused. Statements such as 'to facilitate the reuse of source code and reduce duplication of effort' (Q1.14), 'emphasis on history, reuse old resources' (Q3.7), 'reuse of existing source code' (Q3.5) and 'possibility of reviving dead projects' (Q6.18) exemplified this view.

Producing useful and quality software was believed to require some discipline in programming. Therefore, practices such as 'standards in software design' (Q3.33), 'standards coding style' (Q3.36) and 'computer science/software engineering knowledge' (Q3.8) were suggested. As in

Free/Open Source software development, collaboration was emphasized, 'clarity, simpleness of code' (Q3.35) was also favoured.

FOSPHost sites also lowered the barrier for sharing software projects, but it could also have possible negative effects. Effects included 'low cost in setup will encourage forking of projects' (Q7.5), 'starting unserious projects' (Q7.8), 'projects to be publish before they are ready' (Q7.7) and 'development of projects that are similar (reinvention of the wheel)' (Q7.6). Therefore, forking, unserious, unready and similar projects were regarded as problems in software development from these opinions.

Counter arguments existed for the suggestions above. In terms of usefulness, different values were suggested. For example, Terence commented that 'alpha code is typically NOT useful' (Q1.36) while Garrett mentioned that "A lot of software is created merely to 'learn.'" (Q3.6)

In terms of quality, the problem of defining quality could be troublesome. Garrett made a sarcastic comment that 'measuring the quality of code? You've got to be kidding me. It's hard enough deciding what "quality code" even is' (Q3.2). Chris also claimed that 'conventional measurements (LOCs, function points and so on) are not interesting' and quality could be obtained by peer-review instead (Q3.2).

The positive side of discarding source code rather than reuse was also discussed. Chris suggested that 'there is a part about reusing old resources, but if something becomes irrelevant, it is replaced with extreme prejudice. There is no place for legacy code in open source, which is both a quality and a problem. The code is clearer and more stable (in medium and long term) because of this, but it might be less stable at short term (while change is ongoing) and backward compatibility suffers ("upgrade or die" seems to be the motto of library developers)' (Q3.7). Terence also pointed out that reusing code might also have it own cost that other related

components might be upgraded and thus incompatible (To the extent that this is efficient (given necessary incompatibilities in data-models, etc., across applications) (Q1.14)).

When it came to standard and software engineering, Neil claimed that 'K&R is Evil(tm). The "GNU Coding Standards" are wrong and lead directly to lesser quality code' (Q3.36) (K&R referred to Brian Kernighan and Dennis Ritchie's influential book – 'The C Programming Language' (Kernighan & Ritchie 1988)). On software engineering, there was no substantial comment except Garrett replied that 'Helps a great deal, but is not necessary'. Nevertheless, the average rating for the statement 'computer science/software engineering knowledge' (Q3.8) was 3.0 and the variance was 2.0. Comparing the most controversial statement 'Low cost in setup will encourage starting unserious projects' (Q7.8), which had a variance of 6.7, a variance of 2.0 was quite low, but it could be argued that a number of participants disagree on the importance of this discipline.

Alternative values for software were also suggested. Neil claimed that 'all software is valuable, regardless of size or purpose. None should be discouraged' (Q3.6), which seemed to imply that all software had intrinsic value. Mark, when promoting reuse of software, wrote, "don't focus the site purely on the 'creation of new software' (though this is perhaps the most fun part of programming)" (Q3.5), this also implies that he was aware of 'fun' as another value of software.

What is a worthy motivation? Other than fulfilling the objective of producing useful and quality software, motivations such as fun, credit and speed were also suggested in the survey. Each of these motivations also had their own twist.

From literature (Lakhani et al. 2003; Torvalds & Diamond 2001), the statement 'fun and good spirit and hope' (Q3.37) was expected to be important. Indeed, the average rating of this

statement was 2.1, which also reflected this expectation. Two academics, however, rated this statement negatively and thus resulted as a controversial topic. Both of them rated statements on reuse of source code, usefulness and standard coding style highly.

Another motivation was credit. While most participants agreed that credit was reasonably important ('fair to all efforts' (Q12.5) had an average rating of 2.3 and a variance 1.3), how credits were attributed could be controversial. Considering the comment on the 'a system to attribute credit' (Q3.32), Chris suggested that 'one of the motivation of open source is getting credit and recognition (instead of money), so there should definitely be a way to get "paid". But at the same time, systematizing this kind of thing could seem wrong and counter to some programming practices (egoless programming (Weinberg 1971), extreme programming (Beck 1999)). When it came to the topic of 'giving hosts of IFHOSP sites every credit that they deserve' (Q12.6), Chris also commented that 'they certainly deserve credit, but beware: they have their own advantages in doing so, either reputation-wise or in a financial way'.

Another controversial motivation was speed. Garrett commented on 'frequent submissions of contributions' (Q3.38) and 'having things move fast is one of the things that people like about FS/OS. Without it, users lose interest in projects, and following that, developers also lose interest.' Another similar motivation was 'to compete with other projects' (Q4.24). Nevertheless, Neil suggested that 'speed of projects is irrelevant - particularly volunteer led ones' (Q3.38) and 'to compete with other projects' (Q4.24) has an average rating of 5.0 and a variance of 3.6, which was considerably controversial.

Diverged rating are also found in the topic of welcoming and training less mature developers (Q1.28, Q3.12) and the role of FOSPHost in providing data for research (Q1.17). This may imply a diverse view of topics that were important but not urgent.

Obviously, a number of the statements are not included in the qualitative analysis above. Nonetheless, if a statement did not form any common topic with other statements, then a forced grouping would twist its meaning and more harm would be done. The discussion of the controversial answers can be found in later sub-sections.

6.3 Analysis of the Results of the Delphi Survey

After the presentation of the data collected, a review on the choice made in the design of the Delphi survey based on the responses will be performed. The quality of responses of the survey will also be assessed. Possible improvement to the Delphi survey will then be suggested. The pros and cons of the data analysis methods used and the content of the data obtained will be discussed.

6.3.1 Delphi Survey Method

In this sub-section, the appropriateness of employing the Delphi survey as a method of investigation and the online implementation will be examined. The design of the questions in round 1 and the rating system will be discussed.

The first question to examine is the appropriateness of the Delphi survey method in answering the research questions. By examining the results, the breadth of opinions collected from this method was seldom seen in other methods. A number of surveys were done on the topic of Free/Open Source (Hertel, Niedner & Herrmann 2002; Lakhani et al. 2003; Reis 2002b). In these surveys, a number of questions were premeditated from the researchers' own knowledge domain and the participants could not suggest ideas during the survey. The Delphi survey allowed a less presumptuous approach and thus resulted in a more diverse collection of opinion. If interviews were conducted instead, the depth of data collected would increase but again the breadth might decrease. The argument then follows that whether the diversity of opinion is an appropriate response to the research questions. A possible answer would be yes, if the results of

this study has to be ultimately useful to the Free/Open Source communities. As argued in the analytical framework chapter (chapter 4), diversity exists in the Free/Open Source communities. By considering different voices in the communities, the evaluation model built can be more relevant to actual needs without projecting a 'theory from the ivory tower' image.

The next question to consider is the appropriateness of employing the World Wide Web as a media of data collection. As most of the potential participants were computer literates, thus their ability of accessing a web survey should be a relatively minor concern. The researcher also did not receive any request for help in this area. Comparing to a paper-based system, using a web-based system reduced mailing cost and delay. The researcher also had the freedom to present the results in more comprehensive way by taking advantage of hyperlinks. Other conveniences included data collected in the survey is directly fed into a database for analysis and the web server logs were available to trace the behaviour of the participants. The results on 'References to Results' for rounds 2 and 3 (Table 6-8 and Table 6-9) are examples of tracing behaviour in the Apache web server log. Nevertheless, the system took the researcher a number of months to develop and it could not be done without former commercial programming experience. The error in running the project was relatively few, except in the invitation of participants. This error, however, was not related to the main part of the survey system, which was found to be satisfactorily stable. Some potential participants did not allow JavaScript for security reason and one of the participants complained about bandwidth available in the Post Delphi survey. It was unfortunate that the provided web server was the only available one in the department for surveys, and this was a limitation beyond the researcher's control.

The quality of the questions posed in the first round can also be examined. In the previous sub-section, the reasons for asking the 12 questions were explained. From the data collected, one can review the effectiveness of these questions. A number of questions were not answered

by more than ten expert users. One group of them were the sub-questions for each tools in question 2. The possible reasons could be the design of sub-question within a survey was too complex. Another group of questions were related to the motivations, barriers, positive and negative results of administrators. As the number of administrators who participated were few, such outcome was inevitable. For other questions, the answer received generally corresponded to the intended purpose. Some similar answers appeared simultaneous in different questions, for example Q1.37 'The site should be fast and has high availability' was similar to Q4.2 'Available 24 hours a day' and Q4.6 'Fast access, responsive'. Nevertheless, such cases were relatively rare and the effect on the overall quality of the survey would probably be minor.

A rating system using a 1-7 scale was employed to collect importance of statements in rounds 2 and 3. '1' was assigned as the most important scale and '7' the least. This assignment was counter-intuitive when it came to presentation of results graphically with blue colour bars because the length of the bars for '1' had to be longest while '7' the shortest. The formula $r_n = 7 - r_o$ was introduced to reverse this relationship where r_o was the original rating and r_n was the magnitude to be displayed on the blue bar.

To conclude, Delphi survey was an appropriate method to employ. It was successfully conducted online and had a number of advantages over paper-based method. The questions asked in the round 1 questionnaire generally yielded answers that were relevant. A technical problem was found in employing the 1-7 scale in ratings and was consequentially resolved. After the discussion of the method, the validity and quality of the results will be examined.

6.3.2 Responses and Validity

In this sub-section, different aspects of the quality of the data collected will be discussed. A revisit of the validity criterion of the Delphi survey will first be presented, and then the statistics on the number of participants will be discussed. A reflection on different methods in the

measurement of number of participants will also be presented and the reason for choosing the question-based method as a measure for the validity will be proposed. The number of participants in this survey will also be compared with other published Delphi surveys. The possible barriers for participating in the survey will also be suggested. Other factors that affect the quality of response such as statistics on 'No Comment' and 'No Response', 'Reference to Results' and change of opinion between round 2 and 3 were also be presented.

From the methodology chapter (chapter 5), the minimum number of experts in each of the three groups is 10. This is different from a general survey in which participants were selected from a sample of the whole population. Recalling that Delphi survey was chosen because the topic of investigation was relatively new and thus expert opinions were sought. Therefore, it has a different logic in validity when compared with the response rate approach in a common survey. The logic behind the Delphi survey is when the number of expert participated reach a certain level, there will be little variation in the quality of results obtained (Ziglio 1996).

Under this criterion, according to Table 6-1, there were more or equal to 10 expert users participated in questions 1 to 7 and 12. For the group of academics, question 1, 3, 4 and 5 met the criterion. Unfortunately, for administrators, none reached the target. Therefore, it could be concluded that the results of the survey reached the minimum requirement for validity only for some of the questions with some groups of participants. Results from question 1-7 and 12 are taken as valid in this study.

Another factor that could affect the validity of the survey was how many round did the participants contribute. From Table 6-3, 66% of the participants only contributed in one round. This also had a negative impact on the validity of the survey, as feedback was an important mechanism in the survey to facilitate communication between experts.

The responses of this survey can be compared with other published surveys (Table 6-15).

	Number of Participants for at least One Round	Number of Sub-Groups
(Ludlow 1975)	60	3
(Jillson 1975)	25	5
(Goldstein 1975)	34	12
(Goldschmidt 1996)	121	5
(Twining 1999)	7	Homogeneous
This Survey	32	3

Table 6-15 Comparison of Number of Participants for Different Delphi Survey

Though the responses of this survey are not the best in the comparison, the number of participants are close to Goldstein (1975) and higher than Jillson (1975) and Twining (1999). This comparison is in no way comprehensive but it shows that there were other published surveys that had similar number of participants.

Reasons for not having a higher response rate were firstly the topic was quite narrow and some participants did not feel interested enough while others considered themselves not knowledgeable enough to contribute. Secondly, a number of the participants invited were also too busy to respond. Thirdly, some participants were not familiar with Delphi survey and qualitative data collection. Some of them preferred quantitative instead and participated only in round 2 or round 3. Fourthly, from the Post-Delphi survey, the survey might be too long and the web server too slow. The sub-question structure for question 2 was not effective as well. In retrospect, it was also the naivety of the researcher that the Free/Open Source communities would consist of motivated members that would like to participate in this survey. Indeed most

responses came from the group of expert users, but better research data could be obtained if the researcher invited more.

Other than using the total number of participants of the whole survey as a basis to measure validity, two other methods are available, namely number of participants for each question and each statement. The number for the whole survey (19 expert users, 9 administrators and 14 academics from Table 6-2) is closer to the validity criterion but it does not account for the variations in participation for each round and each question. If the number of participants for each statement was chosen as the measure of validity, then all the statements with no responses would affect the figure. Nevertheless, some of these statements might probably be the result of conscious decisions from participants who just did not choose 'No Comment'. Therefore, the number of participants contributed for each question is used for estimating the validity. This argument could be strengthened by the assumption that if one responded to even some of the statements within a question, one would probably have considered other statements but did not respond, as all statements were on the same page.

The validity of agreed and controversial statements can also be discussed. The agreed statements were selected based on the calculation of average ratings while controversial statements were chosen by variances. The susceptibility of these two calculations to changes can be compared. The hypothetical scenario of adding one more participants can be considered as a reference. It can be proposed that for the agreed statements, as most of the participants, agreed them were important, the possibility of a new participant that disagreed would be low. And even at the event of disagreeing, the effect to the average rating would be minor. On the other hand, for controversial statements, the calculation of variance would be more susceptible if more participants had contributed. So the proposition is that the calculation of variance is more susceptible. Nevertheless, after some mathematic derivations (see appendix F), this

proposition is not supported and there is no conclusion on which calculation is more susceptible to changes. Therefore, there is no conclusion on which type of statements are relatively lower in validity than the other type.

Another measure of the quality of the data is the expertise of the participants. From the statistics of 'No Comment' and 'No Response' (Table 6-6, Table 6-7), there were a significant percentage of 'No Comment' and 'No Response' for the 12 questions at the question page. For round 1 the total percentage for 'No Comment' plus 'No Response' was 64% and for round 3 was 49%. This may suggest that a significant number of experts may not be able to comment in a comprehensive manner. This may reflect the situation that FOSPHost was still a fairly new topic at the time of the survey.

Another measure of the quality of the data was the communication between the participants or feedback. The figure for 'Reference to Results' could be a possible indicator. The total number of referral was 9 for round 2 and 18 for round 3. These figures were the total number of times that all participants referred to the results. If each participant referred twice, then for round 2 the figure would be 24. In round 3, the feature of 'additional information' where participants could look up results and glossary in another browser window (Figure 5-35, Figure 5-36 and Figure 5-37) was added and the number of referral increased. This may suggest that would be most participants would just go through the survey with minimal referral to the results other than the information on the question page. Nevertheless, these figures could be an underestimation of feedback as participants could read the results before login. These actions could not be traced by the researcher, as the researcher had no way to relate the IP addresses in the web server log with the participants' userid.

Another indicator of feedback was the change of opinion between round 2 and 3. From Table 6-10, the change was minimal. This may either indicate that the feedback mechanism was not effective or the participants were firm in their opinion.

To conclude, the validity of the survey reached the minimum standard but it can be improved. It will be argued below that the results did make a contribution to knowledge and therefore the validity and quality of the results will be important pieces of information for those who build their research on these results. With the analysis above, those who would like to build on these results can have substantial information to decide which part of the results should be accepted, reprovved or disproved. The two indicators on feedback showed that the feedback mechanism was not effective. These indicators may not be conclusive, but they served as an example for the potential of online Delphi survey of having different ways of measuring data quality. In the next sub-section, lesson learnt in this survey and ways to improve surveys in Free/Open Source communities will be presented.

6.3.3 Possible Improvements in Delphi Survey Method

In this sub-section, lesson learnt from this Delphi survey and other surveys in Free/Open Source communities will be presented.

An obvious improvement is more participants should be invited. At the time of the survey, as the topic of Free/Open Source was quite new, the number of academics that published at least one paper was low. The number of academics participated in question 2 was lower than question 1 and 3 may also suggest that their understanding of the practical side of the topic could also be limited. Administrators of FOSPHost were also harder to find than expert users, as users were always more. Within the constraint of the survey, where time and effort was limited, it was difficult to find more in these two categories.

Another method may be to shorten the survey to get more responses. Nonetheless, as argued above, the diversity of the survey would be lost. It was possible to make the survey non-FOSPHost specific by asking the questions extracted directly from the model of individual participation to a Free/Open Source community, such as 'what is the negative results of joining a Free/Open Source community?' and then deduce the implications to a FOSPHost site. This approach, however, may get more participants as the scope of 'expertise' increase but the FOSPHost specific responses would be lost.

Learning from experiences in other surveys, there may be other ways to increase the number of responses. Some communities may be more open to surveys than others. For example, the Linux kernel mailing list could be one of the worst places to conduct survey. Lakhani, Wolf & Bates (2002) estimated that there were about 4000 participants in the list and the web survey they conducted received 134 responses. Another survey by Hertel, Niedner & Herrmann (2002) obtained 141 replies. Lastly, Kuwabara (2000) asked for interviews and obtained only 32 replies. These response rates were all worse than the Delphi survey. On the other hand, Lakhani, Wolf & Bates (2002) received 526 responses from 1648 developers (32%) on SourceForge and Reis (2002a) received 521 valid responses from a sample of 1102 invitations (47%) to developers of a variety of projects.

It may be proposed that the design of the survey may affect the response rate. A preliminary examination of "The Free Software Engineering Survey" by Reis (2002a), which achieved the highest response rate, showed that the presentation of survey was direct on the task required and the intention was clear. The survey consisted of about 70 items and it only took an estimated 15 minutes to complete. As a participant of Free/Open Source communities, his writing style (colloquial and sarcastically self-boasting) also related to potential participants (Reis 2002b):

8. Who on earth are you?

My name is Christian Reis. I'm a Brazilian developer involved with ORBit-Python, Bugzilla, PyGTK, Stoq and occasionally some other free software projects. I started my MSc in 2000 and this survey is an important part of the research (which is why you should be nice and help out). I have something of a webpage, too.

These might be the possible factors to increase response rates in surveying the Free/Open Source communities.

6.3.4 Data Analysis

In this sub-section, methods of analysis will be discussed.

The data were analysed both quantitatively and qualitatively. In interpreting the Delphi survey, qualitative analysis is seldom used and there may even be a tendency to use strict numerical procedures (for example (Schmidt 1997)). The decision to use qualitative method was that this method could bring out the meaning of the data to the fullness of its potential. Recalling the purpose of the research is exploratory and practices such as to be 'flexible' (Neuman, Bondy & Knight 2003, p. 30) and 'explore all sources of information' (Neuman, Bondy & Knight 2003, p. 30) were recommended. Another reason is because the participants and the researcher alike had invested time into the survey, detail analysis should be done to make good use of the data collected. The discussion on quantitative analysis will be presented below and then qualitative.

During the selection process, the ratings from four participants who only participated in the second round were included. This was not a common Delphi survey analysis practice, but according to Table 6-10, most participants who contributed in two rounds gave similar answers in round 3. Therefore, the error in assuming that the answers the four participants gave in round

2 were their final answers should be relatively low. The positive side of this strategy was the number of data points in the survey would increase.

For the agreed statements, the selection is based on the first third rule from all the ratings from every statement in the survey. Another method of selecting the first third from each question could have been executed. The assumption behind this method is that every question is equally important and thus the computation of the first third in every question should be handled separately. Undoubtedly, the design of the questions was based on literature. Nevertheless, the basis for the assumption that every question is equally important is not found. By calculation the cut-off score for first third from all the ratings, the importance of each question could be assessed by the number of statements above the cut-off score.

Question	Agreed	Controversial
1	10	11
2	5	4
3	15	14
4	9	6
5	3	12
6	11	5
7	1	8
12	7	5

Table 6-16 Number of Statement Selected in Each Question

From Table 6-16, the numbers of agreed and controversial statements in most questions were similar except for question 5 and 7, where controversies seemed to reign. This may suggest that the lack of common consent of barriers and negative effects of FOSPHost in the opinions of the experts. A possible explanation was a fair number of participants were expert users of FOSPHost, who were probably enjoying a number of positive effects over the negative ones.

The people who suffered significantly from the barriers and negative effects were not included in the survey. As surveys recruited participants on a voluntary basis, this kind of bias is unavoidable, as for most surveys. The implications of this finding and whether the findings relate to the mentioned bias can be further substantiated by additional research.

After the discussion of the quantitative analysis, the qualitative analysis will be also examined. From a positivist viewpoint, one of the weaknesses of the qualitative analysis above is that statements with different ratings were included in the same paragraph with no specification of the difference of their importance. This will promote a false sense of equality between the statements. Also, some of the opinions were personal and quoting them may not help in obtaining a generalised conclusion. Nevertheless, as the purpose of the study was exploratory, gathering information on the topic has priority and a more interpretive approach was used. Based on the data collected, explanations of the phenomenon can then be devised and a more positivist approach can be used to substantiate the claims of the research.

For the agreed statements, the task was mainly on grouping and most of the content of the statements were included except question 2 and 12. Participant comments were seldom quoted. In contrast, the qualitative analysis for controversial statements quoted more heavily on comments and a number of statements were left out of the analysis, as they did not form concepts with other statements. A number of non-controversial statements were included instead to strengthen the different sides of the controversy. A more interpretive approach was also adopted. The reason for these differences could be that the results for agreed statements were closer to the common understandings on the topic and controversial statements by nature were diverse. Therefore, more quotations were needed to portray a clearer picture of the concepts in disagreement.

After the examination of the methods of analysis, the meanings from the data analysed will be further explored.

6.3.5 Discussion of Results

In this sub-section, the content of the results will be discussed.

From the agreed statements, one major theme can be identified – communication. Facilitating communication of different parties from different locations (Q1.1, Q1.3, Q1.4, Q1.5, Q1.18, Q6.2, Q6.6) via multiple means (Q1.38) is both an essential objectives and positive outcomes of a FOSPHost site. Some important work practices also fall into the category of encouraging communication, such as clarity and simpleness of code (Q3.35), listening to others (Q3.16), openness in attitude (Q3.20), tolerance, respect and patience (Q3.13). This communication model is different from the conventional hierarchical management in a business organization where communication from a low-level staff to another low-level staff in different department must be done through managers levels above them. Dafermos (2001) argued the communication process in this conventional hierarchical management was less effective than in a Free/Open Source software development process. One may thus deduce that effectiveness of communication might be one of the success factors of Free/Open Source software development process over a hierarchical system.

Within the grand objective of facilitating communication, such communication is conducted by people with different purposes and styles, indicated from the controversial statements. Some developers aim at producing useful software (Q1.36, Q3.6) while others are just programming to learn (Q3.6) or prototyping (Q1.36). Some may be very lenient towards co-developers and willing to include even the inexperienced (Q1.28, Q3.12) but others may want to have more control. Some even suggest that external FOSPHost sites made it too easy to start a Free/Open Source software project and the results were many unserious (Q7.8), low-quality projects (Q7.7)

were produced. With the diversity of the results collected, this confirms the conclusion from the literature review that there were a number of variables in a Free/Open Source community. By identifying different opposing arguments from the survey, more variables in Free/Open Source communities are discovered and a more substantial picture of the situation can be obtained.

We can also analyse the results using the software evaluation classification proposed in sub-section 5.3.1. The major categories found in the agreed answers from question 2, 3, 4 and 12 will be discussed first and then question 1 and 6. Answers from question 5 and 7 will not be discussed, as there are too few of them to conclude which category is more dominant.

As suggested in the design of the questionnaire for the first round of the Delphi survey, question 2 was aimed at the utility and usability aspect of FOSPHost. The results obtained were mainly on utility and some on usability, but unfortunately the validity of the results for the specifics of each tool was high enough. For other questions, answers related to any of the four categories were possible. As question 3 was expected at collecting opinion of culture and work practices, and thus most of the answers obtained can be categorized as context, for example 'fun and good spirit and hope' (Q3.37), 'tolerance, respect and patience' (Q3.13) and 'openness in attitude, no hidden agenda' (Q3.2). Most of these comments referred to the working relationships between developers and some referred to the administration of the site as well, for example, 'openness in procedures and policies' (Q3.21) and 'giving users as much freedom as possible' (Q3.9). Factors that motivate users to use a FOSPHost site were collected in the responses to question 4. They were mainly utility and usability concerns, for example, 'the tools provided are effective and productive' (Q4.8), 'the tools provided are standard and commonly used' (Q4.7), 'available 24 hours a day' (Q4.2), 'reliable' (Q4.3) and 'fast access, responsive' (Q4.6).

Questions 8 to 11 were aimed at collecting intrinsic concerns. Unfortunately, the number of responses to statements within these questions did not satisfy the requirement for validity. Nevertheless, a number of intrinsic concerns were collected in question 12. The last question, number 12, is basically an 'any other business' question. Most of the responses were related to the management issues and attitudes of the administration of a FOSPHost site, for example 'IFHOSP site should be careful on the usage agreements with users and provide them with enough freedom' (Q12.4), 'An IFHOSP should be run in an open fashion and users should be well informed' (Q12.9), 'Anyone wanting to setup an IFHOSP needs to be aware of the responsibility involved' (Q12.8), 'Expanding an IFHOSP into multiple mirror sites to increase reliability and obtain more credibility from users' (Q12.1) and 'An IFHOSP should be have up to date information of the site and employ novel techniques' (Q12.1). Most of the responses in question 12 can be classified as intrinsic, except responses such as 'The acronym IFHOSP is pointlessly obscure' (Q12.8).

Most answers to question 3 and 12 corresponded to the intrinsic and context categories respectively, but by comparing the content of the statements, similarities can be found. An agreed answer to question 3 'openness in attitude, no hidden agenda' (Q3.2) refers to work practices in the community was similar to another statement in question 12 'An IFHOSP should be run in an open fashion and users should be well informed' (Q12.9). Another pair with similar theme were 'sense of responsibility' (Q3.1) and 'Anyone wanting to setup an IFHOSP needs to be aware of the responsibility involved' (Q12.8). It is then possible to postulate that the users may regard the administration of a FOSPHost a part of the community and measure them with similar values as in the community, or as least expect the administration to understand the respect these values. Evidences from real cases of disputes on a FOSPHost site (Dachary 2001; Kuykendall 2001; OSDir.com 2002) also support this postulation of expectations of the administration.

Question 1 was designed to investigate the objectives of a FOSPHost site and question 6 the positive results from using the site. A number of the agreed answers from both questions were similar as some positive results suggested in question 6 were the fulfilment of the corresponding objectives suggested in question 1. So answers from both questions will be analysed together. Many of the answers obtained were broad purposes that can only be achieved by a combined effort from all four categories. For example, the objective of 'to support concurrent and collaborative software development' (Q1.2) may imply providing source code repository such as Concurrent Versions System (CVS) (Q2.1), providing tools with interfaces that is effective, productive and high usability (Q3.3, Q4.8). Other than utility and usability, the culture of the community on the site should be welcoming and practices such as fun, good spirit, flexible, tolerance and respect (Q3.37, Q3.17, Q3.13) are probably essential. Additionally, not just the users of the site should foster these values; the administrators of the FOSPHost site also may need to respect these values. It can then be argued that a number of answers to question 1 and 6 are related to most of the four categories.

From the above analysis, there were answers that related to all four categories. For trends in answers to different questions, answers to question 2 were utility; question 3 were mostly context; question 4 were mostly utility and usability; question 12 were mostly intrinsic; and question 1 and 6 covered most of the four categories. It can be observed that there were a significant number of agreed answers to question 3 and many related to context. It can then be postulated that the awareness of users of FOSPHost on contextual issues are high.

To summarise, by using both quantitative and qualitative analyses, the meaning in data can be more readily extracted. Recalling the purpose of this research as exploratory, the results did construct a more comprehensible picture of important issues in FOSPHost. As Delphi survey,

unlike conventional surveys, starting by asking broad, open questions, some of the results obtained did not conform to 'conventional ideas' and a number of diverse views were expressed.

6.4 Summary of Chapter Six

The data collected in the survey was presented and analysed in this chapter. In the result presentation section, agreed and controversial answers were showed as well as a variety of data on the responses of the participants. In the analysis section, first, the Delphi survey was examined as a method in data collection. Second, the validity and the quality of the data were discussed. Third, improvements to the survey were suggested. Fourth, the method in handling and interpreting the data was discussed. Lastly, important findings in the results were discussed.

The Delphi survey was examined as a method to collect data and found to be reasonably appropriate. Conducting the survey using the World Wide Web did not pose a high barrier to most participants. This online system provided convenience to the participants as well as additional functionality to the researcher in analysing participants' response. Most of the questions in the questionnaire for the first round were also found to produce the desired response.

The validity of the survey was examined against the criterion prescribed by literature and also other published literature Delphi surveys. The validity was found to just satisfy the criterion prescribed. Nevertheless, a number of other Delphi surveys had similar number of responses. The quality of the data was also examined by the number of statements with 'No Comment' or 'No Response' and found that a significant number of participants might not have the knowledge to ask all the questions asked. The feedback mechanism was also reviewed and found that the mechanism was not used much by the participants.

Improvements to the survey were also suggested. This included to invite more participants, a short survey with clear intention and to speak the language of the Free/Open Source communities.

Both quantitative and qualitative analyses were used in handling and interpreting of the data. This approach extracted a richer and more comprehensive picture of the topic, which satisfied the purpose of an exploratory research.

From the results of the survey, facilitation of communication was found to be the most important agreed theme. On the basis of communication, participants also contribute in their own diverse purposes and styles.

To conclude, the Delphi survey was probably an appropriate method to collect data on the topic of FOSPHost and fulfilled the purpose of an exploratory research. The survey was successful conducted obtaining useful results.

According to the analysis in this chapter, the results of the survey will probably have some contribution to the understanding of the Free/Open Source phenomenon. More data will be presented in the next chapter to portray a clearer picture of the topic of FOSPHost.

Chapter 7

Detailed investigation on External Hosting Sites

7.1 Introduction

In this chapter, the execution and the results of the detailed investigation will be presented. The backgrounds of the sites studied will be described and the data will be presented with comparison on each features and policies. The classification of infrastructure and non-infrastructure sites will be introduced and a preliminary exploration of differences between these two categories will be performed.

7.2 Data Collection and Selection of Sites

The data collection process started on 24 February 2003 and the first version (v0.02) of the comparison table was published online on 19 March 2003. Emails were sent to administrators of the sites investigated. An updated version (v0.03) based on the feedback was published on 4 April 2003. An interactive feature evaluation interface (see chapter 8) was then implemented and three more FOSPHost sites (GBorg, GForge and SEUL) were added. The last batch of emails was sent on 22/9 to collect feedback and statistics. Statistics was updated and minor changes were added as a result. This version (v0.04) was published on 8 December 2003. The discussion below was a report of the comparison of this version.

Recalling the three criteria for selection of sites:

1. Free/Open Source projects are hosted on site.
2. The site welcomes the hosting of Free/Open Source projects from other parties.
3. The site should employ as least a source code repository with basic version control capability.

Ten sites were selected according to these criteria. They include Asynchrony (asynchrony.com 2001), BerliOS (BerliOS), freepository (Minnihan 2003), GBorg (GBorg development team 2003), GForge (Tim Perdue et al.), icculus.org (Gordon 2003), Savannah (Free Software Foundation 2003b), SEUL (SEUL.org 2002), SourceForge (SourceForge 2003) and SunSITE.dk (Sunsite.dk staff group 2003a). Other FOSPHost sites that fit the criteria but were not investigated may include ibiblio (ibiblio.org 2003a), Novell Forge (Novell 2003) and Tigris.org (Collab.Net 2002a). The assistance from the administrators of these sites was not sufficient and the data collected was not enough to make substantial comparisons between other sites. Nevertheless, the sites included probably could represent most of the external hosting sites available. Also, after the completion of the data collection and analysis, Asynchrony was closed down on the 5 Jan, 2004 (asynchrony.com 2004). Nevertheless, the data collected from the site served as an interesting comparison and thus it is still included in this study.

In order to further analyse the sites investigated, a classification system is introduced - infrastructure and non-infrastructure sites, which will be elaborated in the next section.

7.2.1 Infrastructure and Non-infrastructure sites

FOSPHost sites can generally be categorised into infrastructure and non-infrastructure sites. On infrastructures sites, most information about the developers and the projects hosted is stored in databases and standard tools are provided by the site. SourceForge is an example of this category. On SourceForge, there is a standard project page for every project. The page is generated from the corresponding information retrieved from database. On the other hand, for

non-infrastructure sites, information of projects is not stored in a database format and users of the site usually construct their project page out of HTML pages. Indeed a number of infrastructure sites also provide the facility of HTML pages hosting and some users created their own project pages by themselves, but many just use the standard page for convenience.

Within the sites investigated, BerliOS, freepository, GBorg, GForge, Savannah, and SourceForge can be classified as infrastructure sites and icculus.org, SEUL and SunSITE.dk can be classified as non-infrastructure sites. For other FOSPHost sites that were not investigated, Novell Forge (Novell 2003) and Tigris.org can be classified as infrastructure sites and ibiblio a non-infrastructure site.

7.2.2 Introduction to Infrastructure Sites

The backgrounds of infrastructure sites investigated will be presented in this section. The background of the best known site of this category, SourceForge, will be presented first. Then the backgrounds of forked projects of SourceForge, which consist of BerliOS, Savannah and GForge, will then be explained. Finally, sites that do not have much relationship with SourceForge, namely Asynchrony, freepository and GBorg, will be examined.

Not much was written on the history of SourceForge and the information presented below is based on the interview of the original project leader, Tim Perdue (OSDir.com 2002). The SourceForge project was instigated by a few engineers in a Linux hardware company called VA Linux. The original vision of the project was to create a distributed software project management tool for IT managers so that project information can be downloaded and managed on the managers' own client software. This project did not gain much support from the company until a survey company reported that SourceForge was the only best known name of the company. In the light of the report, the management of VA Linux then asked the team of engineers to improve SourceForge for a launch at a major trade conference. The hosting

functions were decided to be improved first, leaving the original vision of a client for IT managers to be put aside. The launch was very successful and a client for IT managers was not mentioned again in the management of VA Linux.

The hardware business of the company continued to deteriorate and the management decided to find new ways to obtain revenue – and selling SourceForge was one of them. The source code of SourceForge was no longer available on the SourceForge site and an improved version called SourceForge Enterprise was marketed (Wire 2000). This move made some of members of the Free/Open Source community angry because SourceForge was GPL licensed and the source code should be available. The original project leader, Tim Perdue, also left the company due to the disappointment in the handling of this event. Even before the closing up of the source code, there were complaints about the alleged action of appropriating the copyright of the work of the users to the company, trying to entrap users by closing down export features and not listening to the needs of users (Advogato 2001a; Dachary 2001; Kuykendall 2001). The financial position of the company was also in doubt (Advogato 2001a; Kuykendall 2001). Despite of these complaints, SourceForge is still the best known FOSPHost site to date hosting a number of famous projects.

Several FOSPHost projects were based on earlier versions of SourceForge and were developed independently (fork). BerliOS developer FOSPHost site was one of the early forked projects probably based on SourceForge v1.5 (Moen 2002). The goal of the site was to act as a neutral mediator for developers, users and businesses in the area of Open Source. The site was also trilingual - English, Danish and Spanish. Thus one of the major changes of BerliOS developer FOSPHost site on the SourceForge v1.5 source code was the translation of the interface into these two other languages.

Savannah was another fork developed by the Free Software Foundation based on SourceForge v2.0 (Moen 2002). Free Software Foundation had been running the GNU project (a project to create a Free version of Unix) for years and hosting various software projects on the GNU web site (Free Software Foundation 2003d). These projects were arranged in a tree/directory structure. For example, Emacs was classified under the category of 'Text creation and manipulation' and the sub-category of 'Editors'. An information page was available for each project on basic information such as download and contact information (Free Software Foundation 2003c). Employing Savannah thus strengthened the hosting capability of the organization. Other motivations of forking may due to the dissatisfaction of SourceForge from the reasons listed above, especially the intention of appropriating the copyright of the work of the users to VA Linux and entrapment of users, which was expressed in a Free Software Foundation Europe article by Loic Dachary, a member of the Savannah project team (Dachary 2001).

Another fork was GForge, which was led by the original SourceForge project leader Tim Perdue based on the source code of last available SourceForge v2.61pre4 (Moen 2002). Tim Perdue was obliged not to work on projects related to VA Linux after leaving the company until recently. GForge was produced after this obligation was lifted with the collaboration of other developers and another existing fork debian-sf (Roland Mas, Christian Bayle & Kwon). Some saw it as the legitimate 'heir' of SourceForge project (Moen 2002). Changes from original SourceForge source code included removing code which catered for the need of extreme scalability (as SourceForge was a gigantic site), easier to install and the tabbed theme (OSDir.com 2002). The official site of GForge (<http://gforge.org/>) only provided hosting for the GForge project itself, not for any other projects. There was no intention to run another 'SourceForge' using the GForge software. Nevertheless, a number of FOSPHost sites did employ the GForge software (Copeland 2003). In the comparison below, unlike other sites,

GForge actually represents the software, rather than the site (because it only hosted one project). Nonetheless, this comparison will be meaningful for the collection of FOSPHost sites that employed GForge.

As we will see in the discussion below, SourceForge and its forked counterpart had a number of similarities in features. The term SourceForge codebase sites will be used to refer to this group of FOSPHost sites for convenience.

Other infrastructure sites that were not related to SourceForge are Asynchrony, freepository and GBorg. The Asynchrony web site was an attempt to leverage the networking capability of the Internet to bring talented programmers and business people together to make money by creating software (Elfanbaum 2001). A large proportion of the site was to facilitate distributed software development and both proprietary and Free/Open Software can be hosted. Therefore, it is qualified as a FOSPHost site. Freepository was a FOSPHost site that only provided the source code repository tool. It was found by John Minnihan back in 1999. GBorg was a FOSPHost site for hosting Free/Open Source projects related to the PostgreSQL database (The PostgreSQL Global Development Group).

There were also other infrastructure sites that were not investigated in this research including Novell Forge (which was based on XoopsForge (Arjen van Efferen & Black 2002)) and Tigris.org (which was running SourceCast by Collab.Net).

7.2.3 Introduction to Non-infrastructure Sites

After the presentation of the backgrounds of infrastructure sites, the backgrounds of the non-infrastructure sites will be explained in this section. icculus.org, SEUL and SunSITE.dk are the members of this category and they will be introduced in this order.

icculus.org was run by Ryan Gordon, a former Loki employee (Loki was an Free/Open Source game software company). The projects hosted on the sites were mainly games. SEUL was the acronym for 'Simple End-User Linux' and the mission for the site was to promote the adoption of Linux by end-users through supporting the development and distribution of high quality Free Software. The accessibility of Linux would hopefully be increased as a result (SEUL.org 2001). Lastly, SunSITE.dk was an affiliated project under the SunSITE (Sun Information and Technology Exchange), which was sponsored by Sun Microsystems to Universities globally (Sunsite.dk staff group 2003b). The goal of SunSITE.dk was 'to help power the development of Open Source Software in the world' (Sunsite.dk staff group 2003c).

There were also other non-infrastructure sites that were not investigated in this research, one of them was ibiblio, which was termed 'the public's library and digital archive' on the Internet (ibiblio.org 2003a).

7.3 Comparison of External Hosting Sites

Sites that were investigated will be compared in this section. Features and other important information were grouped into six categories and they will be presented in the following order:

- General Information
- Project Tools - Tools for Public/Developers
- Project Tools - Tools for Project Administrators
- Personal Tools for Developers
- Community Tools
- Others

The list of features investigated was first built by discovering features available in SourceForge. Then, more features were added to the list when they were found in subsequent sites investigated.

7.3.1 General Information

The first category to be presented is 'General Information' of the site. In this category, overall statistics of the site and whether feedback was obtained from the respective site administrators were included. The comparison for these factors are tabulated in Table 7-1 in ascending order of their respective numbers of projects hosted.

The statistics for a number of FOSPHost sites could be found on the front pages of the sites, but others were obtained by asking the administrators. For the number of developers, for the non-infrastructure sites, as databases were not used to record the details of developers, users with Unix shell account and CVS access were counted instead. Since the administrators gave responses on the statistics independently, the date and time for obtaining the statistics was not uniform. The researcher tried to obtain them within a few days tolerance and most figures were obtained between 22 Sep 2003 and 23 Sep 2003.

It can be seen that non-infrastructure site generally had less projects than infrastructure sites. Another interesting observation is that all the administrators from the non-infrastructure sites were willing to communicate with the researcher but not all of the infrastructure sites, especially the larger ones. For the comparison table presented below, the sorting order of the sites will remain the same as Table 7-1 as it emphasized the differences between non-infrastructure and infrastructure sites.

Site	Type	No. of Projects, excluding projects mirrored (Regardless of activities)	No. of Members (Regardless of activities)	Input from site administrator(s) to this table
SEUL	Non-Infrastructure	About 50 ACTIVE projects (23 Sep 03 17:30 +10)	About 300 with shell access, about 225 with CVS read/write access (23 Sep 03 17:30 +10)	Yes
icculus.org	Non-Infrastructure	61 (19 Mar 03 17:30 +10)	111 Shell Accounts + more developers (19 Mar 03 17:30 +10)	Yes
SunSITE.dk	Non-Infrastructure	111 (23 Sep 03 19:30 +10)	205 CVS access, exact number of members not available (23 Sep 03 19:30 +10)	Yes
GBorg	Infrastructure	123 (8 Aug 03 23:00 +10)	3,012 (8 Aug 03 23:00 +10)	Yes
BerliOS	Infrastructure	818 (23 Sep 03 13:00 +10)	4,583 (23 Sep 03 13:00 +10)	No
Asynchrony	Infrastructure	1,848 (23 Sep 03 16:30 +10)	33,309 (23 Sep 03 16:30 +10)	No
Savannah	Infrastructure	1,886 (23 Sep 03 13:00 +10)	20,575 (23 Sep 03 13:00 +10)	No
freepository	Infrastructure	over 2,500 (22 Sep 03 23:00 +10)	2,948 (22 Sep 03 23:00 +10)	Yes
SourceForge	Infrastructure	68,586 (23 Sep 03 13:00 +10)	703,365 (23 Sep 03 13:00 +10)	No
GForge	Infrastructure	Irrelevant in this Comparison	Irrelevant in this Comparison	Yes

Table 7-1 Comparison of General Information of FOSPHost Sites

GForge was a special case in this comparison that only one project was being hosted on the site <http://gforge.com/> - GForge itself. If one judges only on size, this site may not seem like a worthwhile FOSPHost to investigate. Nevertheless, the focus for the comparison GForge with other sites in this case is actually the features of the FOSPHost sites that employed the GForge software. Some of those sites consisted of hundreds of projects and thousands of developers (Copeland 2003). Therefore, the figures on the site <http://gforge.org/> are not irrelevant in this comparison. One may then ask how many projects are hosted in total by the sites that employed GForge software. This will be quite difficult to answer. Just like the estimation of the number of Linux machines on earth (Miller 2002), anyone can use the GForge software and is not obliged to report it. Therefore, an estimation of these figures was not done. Also, the contact that the researcher made was to the development team of GForge who were also administrating <http://gforge.com/> as well.

Another factor to consider for the comparison of GForge was that sites that employ the GForge software could choose what tools and services to provide – tools that were supported by the GForge software could be disabled and tools that were not supported could be added, as GForge was a Free/Open Source software. The features provided in this comparison were what GForge could support without disabling any feature and without addition. Therefore information of sites that employed 'altered' GForge was not accounted for in this comparison.

Before we begin the discussion of feature comparison, one more matter is worth mentioning. As explained before, for non-infrastructure sites, users of the sites can present their projects in free format HTML pages. As we will see in the comparison, some sites also provided tools such as web server-side scripts and database. In this way, users of the sites could have great flexibility in installing tools of their own choices, even though the sites may not provide the

tools as a standard feature. This type of situation will be denoted by 'DIY' in the comparison ('DIY' stands for 'Doing It Yourself').

The two categories to be examined are 'Project Tools - Tools for Public/Developers' and 'Project Tools - Tools for Project Administrators'. Both of these categories relate to 'Project Tools' but the former can be used by the public and developers while the latter can be used only by the administrators of projects. The 'Tools for Public/Developers' will be examined first.

7.3.2 Project Tools - Tools for Public/Developers

'Project Tools - Tools for Public/Developers' is the category that contains the most numerous items for comparison. Therefore the items are further divided into three groups. The first group concerns the public the most. Features in this group are useful for people who just want to use the product of the projects – general information of the project, the software and the documentation. The next two groups consist of tools that facilitate more communication between users and developers of the projects. One of the groups consists of tools that were voted as important in the Delphi survey. The other group includes other tools that were found on the sites investigated.

The comparison of the first group of features is tabulated in Table 7-2. Features compared include standardized format for general information, free format HTML project homepage, project role assignment, project news, download service, document management and statistics. These features are mostly used by people who just want to know about the project and use the product of the project with minimal further participation. The direction of communication of these tools is mostly unidirectional. That is, from the project team to outsiders. For standardized format for general information, all the non-infrastructure sites did not have a standardized format while infrastructure ones had. One exception was freepository as it only provided source code repository service and had no general information or project page. For

free format HTML project homepage, most sites had this service except GBorg and freepository. For GBorg, the standardized general information page was the only choice. It was then obvious that a number of infrastructure sites also provided free format html pages for project homepage and some projects actually preferred them to the standardized format. For the comparison of project news, the results were again similar to standardized format for general information. For download service, common existing protocols such as HTTP and FTP were employed on a number of sites. For some infrastructure site using the SourceForge codebase, a more sophisticated mechanism was provided – 'File Release'. This system enabled the project administrators to organise download files and distribute them via different mirror sites. Some sites also had another dedicated system for documentation of the project. They were given different name, such as DocManager or FAQ, but the basic function was to store and distribute documents. The 'DIY' item in the 'Document Management' feature meant that the project administrators needed to use the free format HTML pages for dissemination of the documentation. The last feature to discuss is statistics for projects. Usually, the basic statistics provided was number of web pages accessed (net traffic). Project activities statistics included actions such as CVS commits and tracker activities (CVS and tracker will be explained below). Comparative statistics here meant that statistics from one project was presented together with statistics of other projects or rankings were given to a project based on statistics of all projects. The availability of statistics in non-infrastructure sites varied as well as infrastructure sites.

Site	Type	Standardized Format for General Information	Free Format HTML Project Homepage	Project Role Assignment	Project News	Download Service	Document Management	Statistics
SEUL	NI	No	Yes	DIY	DIY	HTTP, FTP	DIY	Comparative statistics via Webalizer at http://stats.seul.org/ (Apache and CVS log available for admin)
Icculus.org	NI	No	Yes	No (See 'Policies')	DIY	HTTP	DIY	Unknown
SunSITE.dk	NI	No	Yes	DIY	DIY	FTP, rsync	DIY	net traffic only
GBorg	I	Yes	No	Yes	Yes	FTP	FAQ, genpage (Free format HTML only pages management)	No
BerliOS	I	Yes	Yes	Yes	Yes	File Release, FTP	DocManager	Project activity, net traffic, comparative stat.
Asynchrony	I	Yes	Yes	Yes	Yes	HTTP	DIY	Number of beta download, number of versions produced
Savannah	I	Yes	Yes	Yes	Yes	FTP	FAQ	Unknown
freepository	I	No General Information	No	Four member types based on permissions to CVS: Admin, Basic, Read-Only, Disabled	No	HTTPS	No	Unknown
SourceForge	I	Yes	Yes	Yes	Yes	File Release	DocManager	Project activity, net traffic, comparative stat.
GForge	I	Yes	Yes	Yes	Yes	File Release	DocManager	Project activity, net traffic, comparative stat.

(NI – Non-Infrastructure, I – Infrastructure)

Table 7-2 Comparison of 'Project Tools - Tools for Public/Developers' of FOSPHost Sites (1)

The comparison of the second group of features is tabulated in Table 7-3. Tools discussed in this group were voted as important in the Delphi survey. From the survey results, five items were voted important, namely source code repository, mailing list, World Wide Web (WWW) server, tracking system and security measures, in the order of importance. WWW server was such a basic infrastructure of FOSPHost that employed by most and would not be compared here. Security measures were used most by project administrators and thus will be discussed in the category of 'Project Tools - Tools for Project Administrators'. Therefore, three tools will be presented here, namely source code repository, mailing list management system and tracking system.

The first tool to be compared is source code repository, which was voted as the most important tool on a FOSPHost site. The basic function of a source code repository is to serve as a central location to manage different versions of a project. Details discussion of this tool can be found in appendix G and only issues that are directly related to FOSPHost sites will be examined here. For all the sites investigated, the basic system employed was CVS (Concurrent Versions System). For all sites, CVS was network-enabled (CVS server). Except for Asynchrony, all sites employed a system for browsing the repository via the web. There were two commonly employed systems, CVSweb (The FreeBSD Project 2003) and ViewCVS (ViewCVS Users Group 2002). CVSweb was one of the early systems to present CVS via the web interface. Nevertheless, CVSweb was found to be difficult to maintain (most of the code was concentrated in one file, cvsweb.cgi, which was more than 100K bytes) and a clone called ViewCVS was implemented. Indeed, more sites employed ViewCVS than CVSweb. As mentioned above, these web-enabling systems were only for browsing, but more functionalities were added in freepository for creating, updating and deleting of directories and files for the repository with versioning capabilities via the web interface. Another additional function was member management, in which different permissions to the repository could be assigned to

different members. freepository also promoted the use of Eclipse plug-in for CVS SSL (Secure Sockets Layer) (Wilms 2003). Eclipse is an IDE (Integrated Development Environment) for programming in different computer language (eclipse.org) and this plug-in could assist the user of Eclipse to communicate with the CVS repository more conveniently. It might be possible that for other sites, the CVS SSH plug-in for Eclipse could be used to achieve similar function, but it was not promoted at other sites. It could be seen that though freepository lacked a number of features comparing to other sites, it was probably the most specialised site on the management of source code repository.

Mailing list management system is another tool that was voted as important. Its basic function is to deliver email sent to the list to subscribers. An obvious function that follows is the management of subscribing members. In the comparison, Mailman (Free Software Foundation 2003a) was most commonly used. This system included most common functionalities with a web interface. Another system used was Ezmlm, which was built upon qmail (Nelson et al. 2003). Ezmlm-idx could also be employed to add extra functionalities on top of Ezmlm such as multi-message threaded message retrieval from the archive and a web interface (Lindberg & Ringel 1999). A probably obsolete system, Majordomo was used by SEUL (both its project homepage (Great Circle Associates 2001) and FAQ (Barr, D.) were not maintained). This system was not web-enabled and to view its mail archive via the web, extra software called MHonArc (Hood 2003) was employed.

Site	Type	Source Code Repository	Mailing List	Tracker
SEUL	NI	CVS Server with ViewCVS and CVSweb	Majordomo + MHonArc	Jitterbug
icculus.org	NI	CVS Server and ViewCVS	Ezmlm-idx	Bugzilla
SunSITE.dk	NI	CVS Server and ViewCVS	Ezmlm, Mail Filtering	DIY
GBorg	I	CVS Server and ViewCVS	Mailman	Bug / Feature / Task
BerliOS	I	CVS Server and ViewCVS	Mailman	Bug / Support / Patch / Task
Asynchrony	I	CVS Server	Yes	Project Change Request / Task
Savannah	I	CVS Server and ViewCVS	Mailman	Bug / Support / Patch / Task
freepository	I	CVS Server, CVSWeb, Eclipse plugin and Web-based CVS management tool (login, add/update/delete directories/files, member management)	No	No
SourceForge	I	CVS Server and ViewCVS	Mailman	Bug / Support / Patch / Feature / Task / Task Dependency
GForge	I	CVS Server and CVS Web Interface (ViewCVS, Ronald Petty's php CVS and Dracos Moinescu's php OO CVS)	Mailman	Tracker made by GForge, theme can be defined by user

(NI – Non-Infrastructure, I – Infrastructure)

Table 7-3 Comparison of 'Project Tools - Tools for Public/Developers' of FOSPHost Sites (2)

The next feature to be compared is tracking systems. Tracking systems can be used to register different issues in a project. A number of the infrastructure sites employed their built-in tracking systems. An example can be taken from a GForge bug tracker. In (Figure 7-2), a screen capture of a bug report is shown. Major parameters on the bug report included date, summary, category (what type of bug), priority, state (what stage of resolution is this bug report in), the person who submitted the report and the developer(s) who the bug was assigned to. The system also accepted comments on the bugs and relevant files. A tabulated overview of bug reports was also available (Figure 7-1). The items in the table could also be re-arranged according to parameters such as state and priority. Though trackers were often used to manage bug reports, it was also possible to be employed for other issues such as feature requests, support requests, task assignments and patches submission.

Tracker: Bugs

[Submit New](#) | [Browse](#) | [Admin](#)

Assignee: (?) State: (?) Category: (?) Group: (?)
 Any Open Any Any
 Order by: (?) ID Ascending Browse

ID	Summary	Open Date	Assigned To	Submitted By
2	Who can use this interface?	2003-11-05 19:53	Haggen So	Haggen So
3	Cannot connect to server	2003-11-05 19:56	Nobody	Haggen So

* Denotes requests > 30 Days Old

Priority Colors:

1 2 3 4 5 6 7 8 9

Figure 7-1 Overview of Bugs

Tracker: [Bugs](#)

[Submit New](#) | [Browse](#) | [Admin](#)

[#3] Cannot connect to server

Date:

2003-11-05 19:56

Submitted By:

Haggen So (haggen)

Category:

Network

Summary:

Cannot connect to server

After installation, cannot connect to the server even with the correct IP address. Firewall checked, corresponding port opened.

Priority:

5

Assigned To:

Nobody (None)

State:

Open

Add A Comment:

[Please login](#)

If you **cannot** login, then enter your email address here:

Followup

No Followups Have Been Posted

Attached Files:

Name	Description	Download
No Files Currently Attached		

Changes:

No Changes Have Been Made to This Item

Figure 7-2 Details of a Bug Report

Two out of the three non-infrastructure sites provided tracking systems. Jitterbug, a web-based system designed by the instigator of Samba, (Tridgell & Shearer), was provided at SEUL. Its basic functions were similar to the example above, but it also accepted bug reports from both email and web interface. On the other hand, an even more sophisticated tracking system, Bugzilla, was provided at icculus.org. Advanced features included supporting a more structured bug resolving procedure, dependencies between bugs, user permissions and others.

The comparison of the last group of features is tabulated in Table 7-4. Features compared include IRC (Internet Relay Chat), webmail, forum, Wiki, survey and other tools.

The first tool compared, IRC is a synchronous, real time, text-based communication tool via the Internet. In a Free Software survey (Reis 2002a), in which the participants were developers in the communities. 22.4% of all the participants voted IRC as important. Although IRC servers were available freely around the globe, it was interesting to see that two out of the three non-infrastructure sites had their own IRC servers. For the third one, icculus.org, they had a dedicated channel on irc.freenode.net. On the other hand, for infrastructure sites, only Asynchrony had an IRC system via a Java client interface.

A forum is a discussion board on the web for facilitation of opinions. It was provided by sites using the SourceForge codebase and Asynchrony. It is possible that the function of this tool overlaps with the mailing list and thus not adopted by other sites.

A Wiki is 'a freely expandable collection of interlinked Web "pages", a hypertext system for storing and modifying information - a database, where each page is easily editable by any user with a forms-capable Web browser client' (Leuf & Cunningham 2001, p. 14). It was voted as the most controversial tool to be included on a FOSPHost site. Some may not be in favour of

Wiki due to its chaotic nature (anyone can change anything). The fact that only BerliOS provided such a service may also be an indicator of its controversy. It was, however, Wiki might be integrated into GForge in the near future (Nikhil Goel et al. 2003).

A survey system is a polling mechanism on the web to collect quantitative opinions. Two types of survey system were found among the sites. One type was user-defined, and the other was pre-defined. For user-defined, the topics of the survey and items to vote for could be defined by the users. Nevertheless, in the case of pre-defined survey, the topics and items were decided beforehand. In Asynchrony, survey was employed as a mechanism to rank projects. Five pre-defined factors were polled, marketability, feasibility, profitability, 'coolness' and uniqueness. In BerliOS and GForge, user-defined survey systems were available.

'Other tools' is a category for tools that exist in one site only. For SunSITE.dk, a USENET server was setup for projects. For GBorg, there was a patch management system that was different from a normal tracker with additional functions such as versioning and indications of the applications of patches. There was also a dedicated area for errata in GBorg. For BerliOS, there was a dedicated area for screenshots of the project. For GForge, Gantt charts could be generated from the task tracker. On the other hand, a number of the special tools/features mentioned above can be implemented if free format HTML pages with web server-side scripts were available.

Site	Type	IRC	Forum	Wiki	Survey	Other Tools
SEUL	NI	IRC server at irc.seul.org	DIY	DIY	DIY	
icculus.org	NI	#icculus.org on irc.freenode.net	DIY	DIY	DIY	
SunSITE.dk	NI	Server provided SunSITE.dk:6667 and irc.ircnet.dk:6667	DIY	DIY	DIY	Usenet
GBorg	I	No	No	No	No	Patch Management / Errata
BerliOS	I	No	Yes	Yes	Yes	Screenshots
Asynchrony	I	via a Java client interface on site	Yes	Unknown	Ratings on pre-defined attributes of a project	
Savannah	I	No	Yes	No	Unknown	
Freepository	I	No	No	No	No	
SourceForge	I	No	Yes	DIY	DIY	
GForge	I	No	Yes	DIY	Yes	Task Manager & Gantt Chart

(NI – Non-Infrastructure, I – Infrastructure)

Table 7-4 Comparison of 'Project Tools - Tools for Public/Developers' of FOSPHost Sites (3)

Important issues in the category 'Project Tools - Tools for Public/Developers' have been discussed above. Before starting the comparison of the next category, it is important to reiterate that users of sites that did not provide the tools mention as standard features may still be able to employ those tools, but it has to be managed by the project administrators themselves (DIY).

7.3.3 Project Tools - Tools for Project Administrators

The next category to be examined is 'Project Tools - Tools for Project Administrators'. These tools were available for project administrators only. They included management for 'tools for public/developers', ask for help (recruitment), activity history, web server-side scripts, shell, database, compile farm, uploading, export, virtual hosting, security and backup. The comparison is tabulated into three tables (Table 7-5, Table 7-6 and Table 7-7).

The first item to be compared is "management for 'tools for public/developers'". This function is basically designed for infrastructure sites to config the database generated project pages and to select what tools to offer to developers and the public. The result of this comparison was therefore similar to 'standardized format for general information'. Except for freepository, all the infrastructure sites provided this service while non-infrastructure sites did not.

The next item is 'Ask for Help (Recruitment)'. It is a comparison of facilities on different sites for recruitment of new developers. For infrastructure sites, many of them had dedicated sections for recruitment (Asynchrony, BerliOS, GForge, Savannah and SourceForge). For GBorg, administrators of projects could put up 'join now' signs on the project front pages or advertised the recruitment on the site news. Two non-infrastructure sites, icculus.org and SEUL adopted this approach too.

Site	Type	Management for 'Tools for Public/Developers'	Ask for Help (Recruitment)	Activity History	Web Server-Side Scripts
SEUL	NI	DIY	Invite others to join by advertising at the project page or in public areas	DIY	SSI, PHP3 and 4, CGI, Embperl
icculus.org	NI	DIY	Invite others to join by advertising at the project page or in public areas	DIY	PHP
SunSITE.dk	NI	DIY	DIY	DIY	SSI, PHP, JSP, other CGI on approval
GBorg	I	Yes	Invite others to join by enabling the 'join now' function at the project page or advertises it in public areas	No	No
BerliOS	I	Yes	Yes	Unknown	PHP3
Asynchrony	I	Yes	Yes	Unknown	Unknown
Savannah	I	Yes	Yes	Yes	No
freepository	I	No	No	No	No
SourceForge	I	Yes	Yes	Yes	PHP3 and 4
GForge	I	Yes	Yes	Yes	Depends on the availability of individual sites

(NI – Non-Infrastructure, I – Infrastructure)

Table 7-5 Comparison of 'Project Tools - Tools for Public/Developers' of FOSPHost Sites (1)

The next feature to evaluate is activity history. Different from project statistics, activity history focuses on changes that only the administrators can make, for example, updating of the general information at the database generated project page or giving permissions to developers to assign items in bug trackers. This function was provided by GForge, Savannah and SourceForge.

The next feature to discuss is web server-side scripts. These scripts enable web pages to be generated according to the input from the users at the browser side. Most sites that offered free format HTML pages also offered this service as well except Asynchrony and Savannah. The most common script supported was PHP (PHP Hypertext Preprocessor) (The PHP Group 2003), which was one of the most popular Free/Open Source web server-side scripts.

The next feature to be discussed is database. Database is usually employed together with web server-side scripts. As expected, the sites that offered this service was the same as the ones that offered web server-side scripts. MySQL (MySQL AB 2003) was the most popular database offered.

The next feature compared is command shell. Command shell is a powerful Unix command line management tool to organise files and perform other lower level tasks on the server (comparing to web-based tools). Nevertheless, offering shell accounts to users could also pose a higher security risk to the system due to the powerfulness of the tool. Considerable attention was needed to run this service.

There was a number of complementary web-based tools programmed in the SourceForge codebase to manage shell accounts and thus many sites using SourceForge codebase offered the service except Savannah posed criteria for the provision. For non-SourceForge related

infrastructure sites, this service was probably not provided. For non-infrastructure sites, selected personnel could access the shell accounts. One of the administrators of the non-infrastructure site mentioned that shell account was regarded as a 'scarce' service and a possible underlying message of giving out an account was a token of acknowledgement to the recipient. One remark is that the 'Yes' in this feature (and shell accounts) for GForge means that the GForge software had provision for database shell accounts management. As for data in other features on GForge, the assumption is that no function is disabled.

The next feature is compile farm. This service enables users to compile and test a piece of software on different computer platforms. Considerable amount of resources was required to run this service and only SourceForge provided it.

The next feature to discuss is export. This function is to let the users retrieve data residing on the FOSPHost site. Recalling the complaints about SourceForge, the ability to export was seen to be a measure of the 'freedom' of a FOSPHost site. For sites with SourceForge codebase, all of them provided CVS tarball. A tarball is a file that contains a collection of compressed file processed by utilities tar and gzip. In this case, the tarball contained the CVS repository. Other export functions ranged from contents of trackers, forums, project summary (a part of the database generated project page), project news and document manager. For freepository, there was an option to backup the source code or the whole repository with other configuration files. For non-infrastructure site, SunSITE.dk had a policy on export that it could be provided on request. For other two non-infrastructure site, the policy was unknown.

Site	Type	Database	Shell	Compile Farm	Export
SEUL	NI	MySQL and PostgreSQL	Available to selected developers	No	Unknown
icculus.org	NI	MySQL on request	Available to team leaders	No	Unknown
SunSITE.dk	NI	MySQL	Available to selected developers	No	On request to administrator
GBorg	I	No	No	No	CVS export on request to the administrator
BerliOS	I	MySQL	Yes	No	CVS Nightly tarball, Forums, Bugs Tracker, Support Tracker and Patches Tracker
Asynchrony	I	Unknown	Unknown	No	Unknown
Savannah	I	No	Only some people had it	No	CVS Nightly tarball, Forums
freepository	I	No	No	No	Members can auto-tar up their projects and download them on the fly. These tarballs may be source code only, or the whole CVS repository (.v files)
SourceForge	I	MySQL	Yes	Yes	CVS Nightly tarball, Trackers, Project Summary, Project News, Document Manager
GForge	I	Yes	Yes	Depends on the availability of individual sites	CVS Nightly tarball, Project Summary, Project News, Forums, Bugs Tracker

(NI – Non-Infrastructure, I – Infrastructure)

Table 7-6 Comparison of 'Project Tools - Tools for Public/Developers' of FOSPHost Sites (2)

The next feature is called virtual hosting. This is a mechanism to allow the project front page to acquire a domain name, such as `http://www.projectname.com/`, instead of `http://FOSPHost.com/projects/projectname/`. There were web-based management tools programmed in the SourceForge codebase to manage virtual hosting and thus many sites using SourceForge codebase offered the service except Savannah. For Asynchrony, an alternative method, which required fewer configurations on the server, was employed. Every project was assigned the domain `http://projectname.asynchrony-projects.com/`. For non-infrastructure site, SunSITE.dk provided service similar to Asynchrony – `http://projectname.sunsite.dk/` with addition email service using the same domain name. On the other hand, SEUL and icculus.org provided full virtual hosting service.

The next two features relates to the security measures of the FOSPHost sites. Security measures were voted as an important feature for tools on FOSPHost sites and two common encryption protocols are usually employed. The first type is SSH (Secure Shell) (Konig 1997). SSH is a replacement of the rlogin protocol for remote access of command shell accounts. This was further developed into other protocols such as SCP (Secure Copy Protocol) and SFTP (Secure File Transfer Protocol) for file transfer and other purposes. Two versions of SSH were available, SSH1 and SSH2, in which SSH2 was the improved, better version (Acheson 2001). On the other hand, SSL was developed to provide encrypted communication via WWW (Netscape Communications Corporation 1998). Therefore, file transfer using FTP and HTTP were not encrypted but SCP, SFTP or HTTPS (HTTP with SSL encryption) were encrypted. For a CVS server, it can be coupled to either SSH or SSL for authentication.

Site	Type	Virtual Hosting	Uploading	Security	Backup
SEUL	NI	Yes	SCP, SFTP and rsync	SSH (1 & 2)	No formal policy, users could do backup themselves
icculus.org	NI	Yes	SCP	SSH, SSL	Backup daily with one week archive
SunSITE.dk	NI	Host name at projectname .sunsite .dk with email	FTP and rsync		Daily
GBorg	I	No	HTTP Form, FTP to an incoming directory and move file(s) via web management		No explicit policy
BerliOS	I	Yes	SCP	SSH (1 & 2), SSL on web	Unknown policy
Asynchrony	I	host name at projectname .asynchrony-projects .com	FTP	SSH, SSL	Unknown policy
Savannah	I	No	CVS via SSH	SSH1, SSL on web	Unknown policy
freepository	I	No	CVS via SSL	100% SSL	Unknown policy
SourceForge	I	Yes	SCP, SFTP, FTP to an incoming directory and move file(s) via web management (SSL)	SSH (1 & 2), SSL on web	Explicit backup policy (5 types of data, daily)
GForge	I	Yes	HTTPS Form	Depends on the policy of individual sites	Depends on the policy of individual sites

(NI – Non-Infrastructure, I – Infrastructure)

Table 7-7 Comparison of 'Project Tools - Tools for Public/Developers' of FOSPHost Sites (3)

In terms of encryption used onsite, only SunSITE.dk and GBorg did not use encrypted protocols. For other sites, SSH was usually employed for command shell related communication and SSL for web related communication. Most of them employed SSH as the authentication CVS except for freepository as only SSL was available on site. For freepository, all communications to the site were SSL authenticated. Uploading was initially not categorised as a feature for security comparison but this feature was later found to have be a number of security issues as login name and password were sent within the process. For site that employed encryption, most of them had encrypted upload as well, except for Asynchrony, where FTP was employed to upload HTML pages for project pages. SourceForge also employed an interesting combination of uploading method. For uploading to shell and project HTML pages, SCP or SFTP could be used. To upload to file release system, files were uploaded by FTP. Then the uploaded file would appear on a web interface accessible via SSL. This file then needed to be handled via the web interface to be assigned into one of the released file of the file release system. This method was also used in GBorg, without the SSL security. Other methods such as rsync (The Samba Team 2003), a program that can be used to synchronise mirror sites, was employed by SEUL and SunSITE.dk. Encryption could also be added to rsync to increase security. Other interesting method could be using CVS with SSH encryption for uploading in Savannah. One of the main functions for CVS is version control, and it is more complex and harder to learn than FTP or HTTP. It is then seldom used for upload. Nevertheless, for users of FOSPHost, they probably know how to use CVS, thus one less service could be employed on Savannah.

The last feature compared is the backup policy. One method to recover from failures in computer systems is to have a regular backup policy. Major parameters in a backup policy include frequency and data. Taking SourceForge as an example, the backup policy was stated in the on site documentation explicitly that backup was done daily on five types of data, namely

site data (user and project records, trackers content, etc.), host data (operating systems, tools, etc.), project file release data, project CVS data, project-specific and user-specific content (user files on compile farm, shell, project web pages, etc.). Unfortunately, for other infrastructure site, such explicit statement was not found. For non-infrastructure sites, icculus.org had a daily backup schedule with one week archive. SunSITE.dk also had a daily backup schedule. For SEUL, users were welcome to make their own backup.

7.3.4 Personal Tools for Developers

For infrastructure sites, many provided memberships for individual developers. Some provided dedicated pages for developers on issues concerning the projects that they were involved. The category 'Personal Tools for Developers' is a preview of these features. Items compared include web space to host personal information, skill and experience, tracker/forum/file monitoring, projects involved, assigned/submitted issues from trackers, survey, diary, bookmark and money earned. The comparison is tabulated in Table 7-8.

The first function to be examined is web space to host personal information. For most FOSPHost sites, there was either a database generated personal page or no personal page at all. One exception was icculus.org, where web space was provided for hosting personal information (possibly for selected individuals only). For other non-infrastructure sites, the focus seemed to be on projects and thus personal pages were not available. For infrastructure sites, though web space was assigned to projects not individuals, most had database generated personal pages with information such as username, skills and project involved, except for freepository, where there was none.

The next feature compared is skill and experience. For sites that had database generated pages for individual developers, all of them had provision for the developers to state their skills and

experiences. Such information can be useful in the recruitment of developers to projects. This function seemed to be one of the main reasons for the existence of the database generated pages.

The next function was tracker/forum/file monitoring. This function either presented the files or messages posted of the tools subscribed or sent notification emails when there were updates. With this function, developers did not have to go through the subscribed tools one by one. This function was provided by all the SourceForge codebase sites. Only tracker monitoring was available at GBorg.

The next function to be discussed is projects involved. This function was actually a list of projects in which the developer was officially a team member of. This was again a common feature that was present in all the infrastructure sites except for freepository.

In some FOSPHost site, a list of issues, which were assigned to or submitted by the developer in different trackers, was provided as a summary. This function was provided by the SourceForge codebase sites.

The next feature is survey. As mentioned above, survey is a tool to collect quantitative opinions from users. A list of surveys that concerned the developer was provided at BerliOS and GForge.

Site	Type	Web Space to Host Personal Information	Skill and Experience	Tracker/Forum/File Monitoring	Projects Involved	Assigned/Submitted Issues from Trackers	Survey	Webmail	Diary	Bookmark	Money Earned	Rating
SEUL	NI	No	No	No	No	No	No	DIY	No	No	No	No
icculus.org	NI	Yes	DIY	DIY	DIY	DIY	DIY	Yes	DIY	DIY	DIY	No
SunSITE.dk	NI	No	No	No	No	No	No	DIY	No	No	No	No
GBorg	I	No	Yes	Tracker only	Yes	No, but could sent notification via monitoring system	No	No	No	No	No	No
BerliOS	I	No	Yes	Yes	Yes	Yes	Yes	Redirect	Yes	Yes	No	Yes
Asynchrony	I	No	Yes	Unknown	Yes	Unknown	No	Redirect	No	No	Yes	Yes
Savannah	I	No	Yes	Yes	Yes	Yes	No	Redirect	No	Yes	No	Unknown
freepository	I	No	No	No	No	No	No	No	No	No	No	No
SourceForge	I	No	Yes	Yes	Yes	Yes	No	Redirect	Yes	Yes	No	No
GForge	I	No	Yes	Yes	Yes	Yes	Yes	Redirect	Yes	Yes	No	Yes

(NI – Non-Infrastructure, I – Infrastructure)

Table 7-8 Comparison of 'Personal Tools for Developers'

A webmail system is a web interface to handle emails. Two types of systems were found in the sites investigated. The first type was a redirection service. Email sent to a developer on site, such as `users@fosphost.org` will be redirected to his or her own email, such as `users@own-host.com`. Another system was to store email sent to developers in a mail server on site. Only `icculus.org` provided the second type of service while `Asynchrony` and `SourceForge` codebase sites employed the first type.

Individual developers could write personal notes or diary entries if the diary function was available. This was not a very important function and was only provided by `BerliOS`, `SourceForge` and `GForge`.

Another function was bookmark. This function enabled the developer to mark some of the pages on site. Then the developer could have quick access to those pages by selecting them from the bookmark list. All the `SourceForge` codebase sites provided this function.

The next function to be discussed is money earned. This was a function only available on `Asynchrony`. As mentioned in the background of the site, the main objective of `Asynchrony` was to make money by selling the product of the projects. The amount of money earned was an indication of the ultimate result of this process. It was not surprising that other sites did not have this function.

The last function to be compared is rating. As many sites did not measure success with money, rating systems were employed instead. Two rating systems were found, one by `Asynchrony`, and another one from the `SourceForge` codebase. For the `SourceForge` codebase rating system, it was inspired by the rating system at `Advogato` (`Advogato` 2003). Five aspects were rated, namely teamwork/attitude, code (code-fu), design/architecture, follow-through/reliability and

leadership/management. Other developers on the site gave scores on these five aspects. A site-wide ranking of the developer was also calculated. This service was hosted at BerliOS and GForge. SourceForge used to host such as service, but it was removed later on. On the other hand, the Asynchrony rating system was even more sophisticated. The final rating of a developer were based on three scores, peer (75%), experience (20%) and quality (5%). The peer score based a peer rating system similar to the SourceForge codebase system. The experience score was calculated from the shares earned from the projects involved. The quality score was obtained by the quality review from customers of the products of the projects. By combining the weighed sum of these three factors, the final rating was calculated. This seemed to be an attempt to construct an indicator for recruitment purposes.

7.3.5 Community Tools

In all FOSPHost sites investigated, there are facilities to foster communication between developers in different projects and the general public. In most sites, these features are provided or at least linked at the first page of the site. This page will be referred as the community page in our discussion. Features such as access, search project, project listing at front page, classification of projects, search people, project help wanted, latest news, get support and a number of other features will be examined. The comparison is tabulated in Table 7-9, Table 7-10 and Table 7-11.

The first item for comparison is the permission to access the community page. In most sites, anyone could access the pages as they were provided at the first pages of the sites. Asynchrony was the only exception, where only registered members were granted the privilege. The community page was the first page that a member was directed to after logging in.

The next group of functions to be compared is related to finding projects and people on site. The first feature to be examined is search project. In most infrastructure sites, this feature

existed except for freepository. For sites with SourceForge codebase, searches based on the keywords of projects were provided. The search function provided by Asynchrony had more elaborated options to search on type, status, keywords and various dates of projects such as project starting date. Preferences on the sorting of the results were also provided. For non-infrastructure sites, this function was not provided.

Another way to find projects on sites is by listing(s) provided at the front community page of the site. Different sites provided different listing of selected projects. Most infrastructure sites displayed the best project listings with different selection criteria such as activities or times of download. Latest or newest project listings were also presented in some sites. Interesting variations could also be found. On Savannah, latest projects were further divided in GNU, non-GNU and www.gnu.org projects. GNU stands for the GNU Not Unix project, which is an attempt to produce a Free Unix system. To become a part of the GNU project meant that the software had to align with the aim and requirements of the GNU project and interoperable with other GNU software. www.gnu.org projects were specific tasks related to the mentioned site. On the other hand, recalling that the objective of Asynchrony was to make money, two listings of 'Need Beta Testers' and 'Completed' were displayed. In the context of the site, 'Completed' meant that the products of the listed projects were on sale in the market and started making money. Therefore, a list of these projects was displayed to promote the success of the site. Before products can be sold, beta testings were required. Beta testers could be recruited on site and those who could suggest useful feedbacks could obtain monetary payments. A listing of projects that 'Need Beta Testers' was thus designed to facilitate this process. Lastly, a list of categories of projects was displayed at the front community page of GBorg for easy access. For non-infrastructure sites, SEUL provided a drop down menu with all projects hosted. For icculus.org, a list of projects was provided at the first page of the site. The leaders of the project could request to be shown on the list.

On most FOSPHost sites, projects were classified so that users of the sites had yet another method to locate projects. Most sites classified projects by different topics such as usage areas and programming languages. In Savannah, classification was done based on GNU and non-GNU. In Asynchrony, keywords suggested by project leaders such as vb (as in visual basic) and rpg (as in role-playing game) were used for categorisation. For two non-infrastructure sites, icculus.org and SunSITE.dk, there was no classification system and all projects were presented in one list.

	Type	Access	Search Project	Project Listing at Front Page	Classification
SEUL	NI	Public	No	A drop down menu for all projects	Topics
icculus.org	NI	Public	No	Projects listed by the requests of the leaders of the project	Just a single list of projects
SunSITE.dk	NI	Public	No	No Listing at front page, but a link to listing	Just a single list of projects
GBorg	I	Public	No	Top / Latest / Category List	Topics
BerliOS	I	Public	Keywords	Top download / Most active / Newest	Topics
Asynchrony	I	Only for registered members	Type / Status / Keywords / Various Dates with sorting preferences	New / Top / Need Beta Testers / Completed	Keywords
Savannah	I	Public	Keywords	Newest GNU / non-GNU / www.gnu.org projects	GNU and non-GNU
freepository	I	Public	No	No	No
SourceForge	I	Public	Keywords	Most Active / Top Download	Topics
GForge	I	Public	Keywords	Most Active / Top Download	Topics

(NI – Non-Infrastructure, I – Infrastructure)

Table 7-9 Comparison of 'Community Tools' (1)

After the discussion of features for finding projects, features provided to locate developers registered on site were investigated. Obviously, in order for this function to exist, a registration system is required. Non-infrastructure sites thus did not provide such service. For infrastructure sites, freepository also did not have this function. For SourceForge codebase sites, most of them provided keyword search on the login name and the real name of registered developers on site. This function was altered on GForge to allow a distinct name or skill search. In Asynchrony, people could be searched by name, skill or rating.

The next function is an interaction between projects and developers – project help wanted. This is a function to facilitate recruitments that were initiated by project leaders. Developers can use this function to find projects that they want to contribute to. This function was provided in SourceForge codebase sites and Asynchrony. In SourceForge codebase sites, recruitment requests were categorised into different skills. Latest recruitment requests were also shown (all SourceForge codebase site except BerliOS). Recruitment skill categories were displayed at the front page of Savannah. For other SourceForge codebase sites, there were links from the front page to the 'Help Want' page. In Asynchrony, the same search engine for general project search can be used for finding recruitment requests from projects as well.

For most sites, there were dedicated locations for announcements and latest news. The most common announcements were new releases of software from projects hosted on site. Some would also post news about server statuses and matter related to the organization of hosting. The most interesting case was SEUL that the news section was dedicated to promote Linux by showing only news in 'Real World' on Linux (this was changed after the investigation was completed).

The next item for examination is how to get support for the sites. As for SourceForge codebase sites, the software used as the FOSPHost interface was usually hosted on site as well. Therefore, those who encountered problems with the site were recommended to report those issues to the corresponding trackers of the FOSPHost project hosted on site. In BerliOS, an email address to the administrator was provided as well. Similar approach was adopted by freepository, where a user forum and Bugzilla bug tracking system was setup. In GBorg, the GBorg interface was also hosted as a project with trackers and users were welcomed to email to the administrator for support and other comments. In Asynchrony, a chat interface using Java was available on site. For non-infrastructure sites, email was the recommended method to get support. For SunSITE.dk, there was an IRC channel to contact staff too.

	Type	Search People	Project Help Wanted	Latest News	Get Support
SEUL	NI	No	No	News in "Real World" on Linux	Email admin
icculus.org	NI	No	No	On new releases	Email admin
SunSITE.dk	NI	No	No	On server status and important projects news	Email admin, IRC with staff
GBorg	I	No	No	On new releases	Trackers, email admin
BerliOS	I	Keywords	Yes	On new releases	Submit request to a tracker, email to staff
Asynchrony	I	Name / Skill / Rating	Search interface available	On site changes and company businesses	Chat with support via a Java client interface on site
Savannah	I	Keywords	Yes (Front page)	On new releases	Submit request to a tracker
freepository	I	No	No	No	User forum, Bugzilla
SourceForge	I	Keywords	Yes	On new releases	Submit request to a tracker
GForge	I	Name / Skill (by keywords)	Yes (Project Openings)	Depends on the editor	Depends on the policy of individual sites

(NI – Non-Infrastructure, I – Infrastructure)

Table 7-10 Comparison of 'Community Tools' (2)

Other than the community tools compared above, some sites also provided other special features. One of these features is advertising. As the most popular external FOSPHost site, having advertisements at SourceForge seemed to make sense. They were hosted as banners on the top of most pages of the site. On the other hand, free advertisements for customers, sponsors and Free/Open Source projects were available at the front page of GBorg. In SEUL, Linux advocacy documents were hosted on site (Savannah was of course strong in advocacy on Free Software but there was no obvious on site feature such as catchy hyperlinks or banners to promote it).

Another special feature is discussion area for topics not directly related to projects. For example there was a mailing list hosted on SEUL called `seul-edu`, which was a list on introducing Linux to schools. Other similar list existed on SEUL as well. Discussion areas were also provided at Asynchrony on general technical issues regarding different operating systems and languages.

Other miscellaneous features included an opinion poll at the front page of SunSITE.dk on technical topics such as blog and IPv6. On Asynchrony, in addition to having skill profile of register members, there was collaboration with an external skill certification company to substantiate the 'claims', of the members. For `icculus.org`, three lists were found on the community page of the site. The first was a list of personal sites, then a list of web sites that was virtually hosted by `icculus.org`. The third list was the most interesting list of all, a list of credits. Contributions mentioned on that list included cash as well as hardware, expertise and even `icculus.org` icons. According to the administrator of the site, most of the donation actually went to developers on the site. For example, when a developer required a particular hardware in order to progress in development, it was donated by others. These good deeds were the origin of this list.

	Type	Other Features						
SEUL	NI	Linux Advocacy	Mailing Lists for Topical Discussion					
icculus.org	NI					List of Personal web sites	List of virtually hosted web sites	List of credits for donation & call for donation
SunSITE.dk	NI			Opinion Polls				
GBorg	I	Advertisements from customers, sponsors and also from Free/Open Source projects (for free)						
BerliOS	I							
Asynchrony	I		General Discussion Area (System / Technical / Projects)		SkillDrill (External company for skill certification)			
Savannah	I							
freepository	I							
SourceForge	I	Advertisements bar from customers						
GForge	I							

(NI – Non-Infrastructure, I – Infrastructure)

Table 7-11 Comparison of 'Community Tools' (3)

7.3.6 Others

The category 'others' is comprised of items that cannot be classified into categories above. Nevertheless, these items are also important in their own terms. Items to be compared in this category are software for web interface of FOSPHost, license of web interface, development methodology, license requirement for project hosted, copyright, advertisement, legal and language related issues, flexibility, donation, miscellaneous items and additional services from sites in the same organization. The comparison is tabulated in Table 7-12, Table 7-13, Table 7-14 and Table 7-15.

The first item is software for web interface of FOSPHost. The web interface is what glues the tools and information of projects together. For two out of the three non-infrastructure sites, static HTML pages were used to present the content of the site and link to different projects hosted and tools. For SunSITE.dk, a content management system, Drupal (drupal.org), was employed. For infrastructure sites, all of them developed their own software for the interface, except that the SourceForge codebase sites based their systems on SourceForge code from different versions.

For the licenses of the web interface, most of them were GPL licensed, which was one of the most popular license for Free/Open Source software in general. Exceptions were GBorg was licensed under the Great Bridge Open Source License and Asynchrony was proprietary.

Site	Type	Software for Web Interface of FOSPHost	License of Web Interface	Development Methodology	License Requirement for Project Hosted
SEUL	NI	Just a few HTML pages	Probably no need for a license	Probably no need for a "Methodology"	Free/Open Source Projects, Exceptions permitted
icculus.org	NI	Just a few HTML pages	Probably no need for a license	Probably no need for a "Methodology"	Free/Open Source projects
SunSITE.dk	NI	Drupal	GPL	Free/Open Source	Free/Open Source Projects
GBorg	I	GBorg	Great Bridge Open Source License	Free/Open Source, Self-Hosted on site	OSI Licenses
BerliOS	I	BerliOS (probably based on SourceForge v1.5)	GPL	Free/Open Source, Self-Hosted on site	Free/Open Source Projects
Asynchrony	I	Developed by Asynchrony	Proprietary	Closed Source	Both Proprietary and Open Source
Savannah	I	Savannah (based on SourceForge v2.0)	GPL	Free/Open Source, Self-Hosted on site	GPL compatible licenses
freepository	I	Freepository	GPL	Free/Open Source, Self-Hosted on site	No restriction
SourceForge	I	SourceForge	GPL (but download files not found and no CVS)	Closed Source	Free/Open Source Projects
GForge	I	GForge (based on SourceForge v2.61pre4)	GPL	Free/Open Source, Hosted at http://gforge.org	Policies can vary for different sites

(NI – Non-Infrastructure, I – Infrastructure)

Table 7-12 Comparison of 'Others' (1)

Another closely related issue is development methodology. As most of the software for web interface was Free/Open Source software, the development methodologies were expected to be Free/Open Source as well. This software could also be hosted as an individual project on their own sites (self-hosting). This was in fact the prevailing methodology, except for SourceForge, where there was no source code for download and no CVS available (It was checked by the researcher on 24/2/03 and a few other times). This confirmed with the background study of SourceForge that the source was closed.

After the discussion of the license of the software for web interface, the license requirements for the projects hosted on varied sites need to be examined too. As the criteria for the selection of sites for investigation, all the sites need to accept Free/Open Source software for hosting. Most of them in fact only allowed Free/Open Source licenses except for Asynchrony and freepository, where hosting proprietary software were a valid option. In SEUL, license was negotiable under some circumstances, but unlike Asynchrony and freepository, it was not an official option, which would be approved automatically. On the other hand, only GPL compatible licenses were allowed on Savannah, which could be regarded as the most 'restrictive' of all.

The next item to be discussed is the copyright of the software developed on the FOSPHost sites investigated. For most sites, the developers owned the copyright of the source code (and the corresponding cells on the comparison table were left blank). Asynchrony was the exception. Another interesting aspect was data preservation. This policy stated that even if a project was officially moved to another FOSPHost site, the original FOSPHost reserved the right to host the data and the project leader could not remove it. This policy was adopted on GBorg and SourceForge.

Site	Type	Copyright	Advertisement	Legal and Language Related
SEUL	NI			
icculus.org	NI			
SunSITE.dk	NI		No Commercial ads and/or banners	Satisfy Danish law, content in English or Danish
GBorg	I	The developers owns the code but the site can keep hosting them even if closed later	No Commercial ads and/or banners	
BerliOS	I			Some documentations and interfaces had European language translations other than English
Asynchrony	I	Asynchrony owns the code		
Savannah	I	Free Software / Rights to code (of users)	No revenue-generating Advertisements	No dependencies to non Free Software, no GIF files
freepository	I	Users own the code, not the site		
SourceForge	I	Data Preservation - contributors own the code but it will cannot be deleted from the site		
GForge	I			

(NI – Non-Infrastructure, I – Infrastructure)

Table 7-13 Comparison of 'Others' (2)

The next item to be compared is advertisement on the web pages hosted. Both SunSITE.dk and GBorg had adopted policy to avoid commercial advertisement or banners. Savannah had an even more exact definition that advertisements could not be revenue-generating.

The next item is legal and language related issues. While probably none of the sites investigated would tolerate illegal materials, some had interesting requirements on the project hosted. For SunSITE.dk, the content of the project hosted needed to be either English or Danish and legal under Danish law. For Savannah, the software hosted needed to have no dependencies on non Free Software. Image files with GIF format should not be used. For BerliOS, multi-lingual seemed to be encouraged. Even some of the getting start guide of the FOSPHost web interface was translated in the 'fourth' language on site, Hungarian.

The next issue for discussion is the flexibility of the management of the sites to accommodate special needs of the users. For infrastructure sites, many have fixed templates and established workflows in management. These structures may help users of the sites to decrease in confusion. Nevertheless, structure also can imply a decrease in flexibility. For non-infrastructure sites, SunSITE.dk seemed to welcome specific requests on services from developers and encouraged prospective developers to host on other infrastructure sites if they did not need any special services. Similarly, on icculus.org, administrator of site mentioned his intention to manage in a flexible and informal manner to serve the best interests of each individual developer, as opposing to the structured services provided in other infrastructure sites. The administrator was also selective on developers and projects to maintain a smaller, more elitist community around the site. It could be argued that flexibility was indeed one of the advantages of non-infrastructure sites over infrastructure. Nonetheless, for most FOSPHost, by its nature of being open, still offered substantial flexibility over traditional development tools.

Site	Type	Flexibility	Donation	Miscellaneous
SEUL	NI			
icculus.org	NI	Flexible and informal to meet the need of individual developers, no restrictive infrastructure such as 'Role Management', elitist & selective on projects to form a smaller community where individual needs can be cared for	Most donations go to developers' needs (e.g. new video card, motherboard)	
SunSITE.dk	NI	Willing to adapt to individual needs, policy negotiable, tools negotiable	Welcome more volunteers to help run the site	SunSITE.dk URL must be visible when re-directed from other sites
GBorg	I			
BerliOS	I			
Asynchrony	I			
Savannah	I			
freepository	I		Donation encouraged	
SourceForge	I			
GForge	I			

(NI – Non-Infrastructure, I – Infrastructure)

Table 7-14 Comparison of 'Others' (3)

Donations were explicitly encouraged on some site. freepository was one of them. SourceForge also started a donation campaign recently but it was not included as it was discovered after the data collection was completed. On the other hand, people, not money was asked for on SunSITE.dk. The most interesting case was found at icculus.org. As mentioned above, most donations were not made to the need of the site, but to needs of other developers on site to help them progress in their projects.

Site	Type	Additional Services from sites in the same organization
SEUL	NI	
icculus.org	NI	
SunSITE.dk	NI	
GBorg	I	Other information on PostgreSQL
BerliOS	I	Other services that assist Open Source Business development such as SourceAgency and DevCounter
Asynchrony	I	
Savannah	I	Free Software related topics and philosophies
freepository	I	
SourceForge	I	Part of OSDN (Open Source Development Network) with Slashdot, NewsForge, etc
GForge	I	

(NI – Non-Infrastructure, I – Infrastructure)

Table 7-15 Comparison of 'Others' (4)

The last issue to be examined is additional services from sites in the same organization. A number of infrastructure FOSPHost sites were parts of some organizations. In GBorg, users were directed to other information on the PostgreSQL database on the front page of the site. Similarly, on Savannah, links to topics related to Free Software and GNU projects were also presented at the front. For BerliOS, the FOSPHost service was actually one of the many

services that the overall BerliOS web site provided. The goal of the site was to act as a neutral mediator for developers, users and businesses in the area of Open Source. Examples of other services were 'SourceBiz', which provided updated Open Source business news and 'SourceWell', where announcement of new version Open Source Software and retrieval information could be found. For SourceForge, it was a member of the OSDN (Open Source Development Network), which was made up of other prominent Free/Open Source sites such as Slashdot (OSDN 2003a) and Freshmeat (OSDN 2003b).

7.4 Discussion of the Comparison

From the comparison above, ten external FOSPHost sites were studied. Brief backgrounds of the sites were introduced. Tools that facilitated communication between developers, users and other interested parties (Q1.3, Q1.4, Q1.5 & Q1.38) in a distributed fashion (Q1.1) were examined. Different policies on different the sites were also compared. A total of 69 items were compared and they were divided into six aspects, namely general information, project tools - tools for public/developers, project tools - tools for project administrators, personal tools for developers, community tools and others. A more detail picture of FOSPHost sites in operation was thus depicted.

The ten FOSPHost sites examined represented a diverse collection of sites. Many of them had their own theme. For SEUL, the focus was on end-user application on Linux. For icculus.org, the focus was on games. Projects hosted on GBorg were related to the PostgreSQL database. Open Source businesses were promoted on BerliOS with emphasis on localisation. On Asynchrony, money making was the aim. Sophisticated computations on revenue sharing, rating of projects and members were adopted. For Savannah, the philosophy of Free Software and the GNU project was promoted. Though freepository only provided source code repository service, this service was probably the most feature-rich among the sites. A number of sites also did not have a particular theme, namely, SunSITE.dk, SourceForge and GForge.

As stated in the methodology chapter (chapter 4), the data collected was publicly available. Due to the nature of openness in FOSPHost, a lot of the data were readily accessible on the Internet. By having access to the source code of most of the sites, the job of determining which features existed was made easier. Administrators of a number of sites were also open in their response when asked. For administrators that did not respond, most of the sites contained documents to explain the policies of the sites. This also matched that agreed Delphi survey results on 'openness in attitude, no hidden agenda' (Q3.2), 'openness in procedures and policies' (Q3.21) and 'an IFHOSP should be run in an open fashion and users should be well informed' (Q12.9).

For all the sites investigated, though there were different criteria on allowing projects to be hosting, service fee was not one of the criteria. This could reflect the administrators' understanding on the desire of the users to obtain a FOSPHost service that was 'low cost or free' (Q4.4).

In the Delphi survey results, five tools were named important. They were source code repository (Q2.1), mailing list (Q2.2), WWW server (Q2.5), tracking system (Q2.4) and security measures (Q2.11). All the sites employed source code repositories and WWW servers. Except for Asynchrony, all source code repositories allowed anonymous web-based access. This matches with the agreed results from the Delphi survey on "creating a public library atmosphere, giving users as much freedom as possible and staying out of the users' way" (Q3.9). With the exception of freepository, all sites provided mailing lists and tracking systems. In terms of security measures, while icculus.org and SunSITE.dk did not provide encrypted protocol, others implemented them at least in some of the services provided.

Another result related to the tools offered was 'standard and commonly used' (Q4.7). In the case of source code repository, all sites employed CVS, which was the most popular Free/Open Source repository. There were some variations in mailing list, tracking system and security measures, but most of the tools provided were still commonly used.

The classification of infrastructure and non-infrastructure sites were used in the comparison and found that non-infrastructure sites were generally smaller in size in terms of project hosted. Relevant comments could be found in the controversial results from the Delphi survey on the size of a FOSPHost site. One of the barriers suggested in preventing users from using a FOSPHost was 'the IFHOSP does not reach a critical mass of users and projects to achieve its advertising function' (Q5.19). So increase in size is a favourable characteristic. Nevertheless, an opposing view was also found, 'big IFHOSP are bad (e.g. Freshmeat) small IFHOSP are good' (Q12.12). Unfortunately, further explanation of the strength of smaller sites was not elaborated. Furthermore, small in size and non-infrastructure sites may not bear any relationship with each other. It may be just a coincident. Further examination is thus needed.

A quantitative comparison of the number of features available between the ten sites can also be done. The results were listed in Table 4-1. Features that were not present or unknown were regarded as missing and DIY is not regarded as missing. Also, many of the items compared under the 'General Information' and 'Others' categories were backgrounds and policies of the sites, not features. Therefore they were excluded. Under these rules, 53 items were counted and three infrastructure sites GForge, SourceForge and BerliOS provided the most features. One non-infrastructure site came fourth in the comparison icculus.org, and it provided just one less feature than SourceForge and BerliOS.

Site	No of Features (Include DIY)	No of Features Missing
GForge	43	10
SourceForge	41	12
BerliOS	41	12
icculus.org	39	14
Asynchrony	33	20
Savannah	31	22
SunSITE.dk	30	23
SEUL	30	23
GBorg	23	30
freepository	8	45

Table 7-16 Number of Features excluding 'General Information' and 'Others'

One reason for icculus.org in providing more features is the availability of personal page for developers. If we further exclude the category of 'Personal Tools for Developers', then the number of features provided by SunSITE.dk and SEUL were the same as icculus.org and they all came fourth in the comparison (Table 7-17).

Site	No of Features (Include DIY)	No of Features Missing
GForge	34	8
SourceForge	34	8
BerliOS	32	10
SunSITE.dk	29	13
SEUL	29	13
icculus.org	29	13
Asynchrony	28	14
Savannah	25	17
GBorg	19	23
freepository	8	34

Table 7-17 Number of Features excluding 'General Information', 'Personal Tools for Developers' and 'Others'

One possible shortcoming of this comparison was counting every feature as equal. Moreover, omitting 'Personal Tools for Developers' might be reasonable only if the features that related to projects were the most important features. Nevertheless, the figure above may give an overview of the situation and may suggest that features provided may not be the major differences between infrastructure and non-infrastructure sites.

In the comparison above, the focus was on quantitative data. If we look into the qualitative content of the comparison, from the item 'Input from site administrator(s) to this table', all the administrators from non-infrastructure sites offered help but not all infrastructure sites. The willingness in communication could also be seen in the availability of synchronous, real-time communication tools such as IRC. The administrators could also be contacted directly by email. In contrast, in some larger infrastructure sites, trackers were the standard way of

communication. In terms of policies, SunSITE.dk and icculus.org were flexible in catering for special needs. In SEUL, even the license requirement for project hosted could be negotiable.

If flexibility and willingness to communicate were probably the attributes of non-infrastructure sites, then what kind of effect would they have on the developers? The handling of shell accounts in non-infrastructure sites may provide insight to this question. First considering in SourceForge, shell accounts and database service came with every project approved. To get an account was a matter of following the procedure prescribed in the documentation. On the other hand, from the examination above, shell accounts were not granted automatically in non-infrastructure sites. Only selected personnel could gain the right to this service. A possible underlying message was that having a shell account was a token of trust and acknowledgement. It could then be seen that the operation of non-infrastructure sites could give developers a more personalised service. Flexibility and willingness to communicate were probably the pre-conditions for the developers to feel the care and respect from the administration of the sites.

Another interesting example could be found on icculus.org, a non-infrastructure site. Donations could be made not only to the site, but donations such as hardware could also be given personally to developers. Such action was obviously encouraged by the site and a credit list of such deeds was presented on the front page. Quoting from the administrator, this policy was one of the measures to achieve the philosophy of 'happy developers are productive developers'. The administrator also admitted that the number of donations to developers were substantially more than donations to the site. In huge infrastructure sites such as SourceForge, the top 10 projects might have thousands of hit on the project front pages daily. Hosting on non-infrastructure sites might not have this effect of gaining popularity. Nevertheless, the

quality of attention in receiving hits on project page and receiving a gift from others could be substantially different.

Flexibility did not just enable the administration of a site to serve personal needs of developers; it may also allow developers to be themselves. One of the controversial results from the Delphi survey was 'Reinforcing explicit development roles' (Q3.4). Two comments were made against this statement. Chris argued that 'Hackers are often jacks-of-all-trades. Pigeon-holing them is bad.' Garrett also claimed that "It's good for 'roles' to not be taken too stringently in FS/OS. Many of us are 'all-purpose' developers. The diversity we are exposed to can't let us get stuck in one 'role'." As the 'standard format' did not exist in non-infrastructure, developers needed to make more decisions on the design of the project page and the employment of tools. The arguments proposed by Chris and Garrett might imply that some developers would rather be granted this freedom as the norm. Another relevant controversial result was found in Q3.18. The statement 'The value of heterogeneity, differences as assets' was a summary of two comments, one was suggested by Alvin that 'they (administrators of FOSPHost sites) shouldn't promote any particular practice. The heterogeneity of approaches is one of the strengths of the way things are done without these infrastructure sites.' (Alvin's comment was actually the origin of the idea of the classification of infrastructure and non-infrastructure sites) It is possible to postulate that the design of non-infrastructure sites may fit the work practices of certain groups of developers.

After the discussion of various aspects of non-infrastructure sites, we can revisit the relationship between size and non-infrastructure sites. One of the administrators of a non-infrastructure site pointed out that the size of the site was kept intentionally small otherwise the amount of personalisation in the communication cannot be maintained. On the other hand, by the nature of the infrastructure, more projects can be hosting as the structure can

effectively streamline the management on infrastructure sites. It will also be easier to locate a project hosted or a developer registered more easily as every entity has the same format. Flexibility is thus harder to achieve and communication may become more formal.

It can then be postulated that by creating a smaller, more flexible non-infrastructure sites, a closer community with more personalised attention can be built. The personalised attention given to developers can then be a positive motivation for advancing their projects. For infrastructure sites, the management was partly done through the structure and the policies of sites. For non-infrastructure sites, it could be argued that more emphasis was put on motivating the developers in a more intangible, personal level. Being flexible may also fit the project management style of particular developers.

After the discussion of the differences between the two categories, there is an issue that may require clarification. The discussion above is not an endorsement from the author on non-infrastructure sites. Non-infrastructure sites are probably less known and thus more attention was paid in the discussion above. The differences between infrastructure and non-infrastructure sites were examined and probably they are more applicable in some occasions than others. Due to the primarily focus of the data collected was on the 'what' questions on features, the possibility to establish a well-reasoned preference for one or the other was low. Further research into the 'how' and 'why' questions may be required.

From the analysis of infrastructure and non-infrastructure sites above, there seemed to be differences found between two groups of sites. This categorisation thus seemed to perform its job of separating sites with differences. Nonetheless, more research into the 'how' and 'why' questions will be required to determine the nature of these two groups of sites. Moreover,

SourceForge codebase sites also had a big influence among infrastructure sites and they may have the potential to become a category on their own rights.

Recalling that the overall purpose of the research was exploratory, the preference of this investigation should be given to breadth over depth. Nevertheless, one might find some of the features discussed descended into minute details. This approach was taken as administrators probably had a technical background and preferred clear, technical presentation. Fortunately, this assumption seemed to be correct and the comparison table opened doors for the researcher to start conversations with a number of administrators. One of the administrators appreciated the researcher's effort for making it clear how the sites measured up to SourceForge, while another found a feature from another site interesting and might add that to his/her own site. This approach of including details of features was also chosen as some of the users of the final evaluation model may also have similar technical background as the administrators. From the responses elaborated above, this approach had contributed to this study and possibly will contribute to the evaluation model as well.

The study presented was on one hand, a reasonably comprehensive research on external hosting sites; but, on the other hand, opened other doors for further research. From the agreed result of the Delphi survey, tools provided on a FOSPHost site should be effective and productive (Q4.8). The availability of the sites should be high as well as the bandwidth (Q1.37 & Q4.6). The usability of the tools was important as well (Q3.3). Moreover, 'How' and 'why' questions such as 'how does the sites serve the purpose of different type of users?' and 'why does different administrator run the sites different?' can be studied to deepen the understanding on infrastructure and non-infrastructure sites. Research method such as interviews on administrators and developers can also be conducted to collect more data to answer these questions.

7.5 Summary of Chapter Seven

Ten sites were investigated in this detail study. The backgrounds, features and policies of these sites were compared under the categories of general information, project tools - tools for public/developers, project tools - tools for project administrators, personal tools for developers, community tools and others. These sites were further classified into infrastructure and non-infrastructure based on their nature. Through this study, the meaning of a FOSPHost site fostering communication of different parties in multiple means was further expanded. Relationships between Delphi survey results and the findings in the investigation were found and the differences between infrastructure and non-infrastructure sites were explored.

After accumulating data from the Delphi survey and the detailed investigation, it is possible to construct an evaluation model on FOSPHost sites. The detail implementation of this evaluation will be elaborated in the next chapter.

Chapter 8

Construction of Evaluation Model

8.1 Introduction

In this chapter, the implementation of the evaluation model, which was the final product of the research, will be showed together with the rationale of the design of the implementation. The discussion of this rationale will begin from the examination of the nature of the data collected in the previous chapters and the choice of suitable types of evaluation presentations. After deciding the types of presentations, the choice of tools for implementations will be explained. Afterwards, the actual implementation will be presented. A discussion of the implementation will also be included.

8.2 Data Collected and Choice of Evaluation Presentation

Two types of data were collected in this research. One was from the Delphi survey and the other from the detailed investigation. The data from the Delphi survey was both quantitative and qualitative. It consisted of qualitative statements with quantitative numbers to indicate their respective validity, importance and controversy. A substantial proportion of the content of the statements covered a number of broad topics. On the other hand, the data from detailed investigation was mainly qualitative. Most of the content covered by each items in the comparison table of the detailed investigation was fairly specific.

According to the nature of data collected, different evaluation presentations were chosen. Recalling that there are two major types of presentations, namely checklist/framework and broad topics. Within the presentation of checklist/framework, the presentation may prompt the user to give different types of answers, namely binary, subjective scale, weighed scale, measured results and qualitative answer. With evaluation built on checklist/framework with binary answers, the focus of the evaluation tended to be very specific while broad topics presentation gave the users of the evaluation a lot of freedom to explore. Nevertheless, broad topics presentation may require more mental effort from the users (Table 5-4).

Matching the nature of the data collected to the nature of the evaluation presentation, a checklist with qualitative answers was chosen to present the Delphi survey results. This was chosen as many of the statements in the Delphi survey contained broad meanings. Broad topics presentation might not be applicable, as it required an established theoretical framework as well as mental effort from the users of the evaluation. One of the shortcomings of this presentation was the quantitative side of the data could not be easily represented. A possible compensation was to sort the statements in the checklist presentation according to the measure of importance voted in the Delphi survey. Nevertheless, some statements will be grouped together for simplicity's sake and ordering according to importance came second in terms of priority.

Comparing to Delphi survey results, the data collected in detailed investigation was more specific. Therefore, a checklist presentation with subjective scale or weighed scale could be chosen. Nevertheless, one of the drawbacks of subjective scale is that all features will be presented as equally important since the final score calculation will be a simple summation of the scores given to every feature. Employing weighed scale may rectify this shortcoming, but not totally. Two possible situations may arise. One situation will be that the user may find assigning numbers for different importance difficult. Another situation will be that in a

weighed system, the summation of the score of many minor features will be equal one major feature. Nevertheless, this situation might not be true in real life. In some cases, the major features will be so important that the presence of many other minor features will be irrelevant. A modified version of weighed scale was thus devised.

This modified presentation was similar to weighed scale, but the importance of features was weighed qualitatively. For example, each feature could be classified as nice to have, important and indispensable. Three scores were then calculated according to the features presented, such as the number of indispensable features present. The FOSPHost sites with the most indispensable features present will then be ranked first. They would then be ranked according to the important features and the nice to have features. In this way, the importance of major features and minor features would not be mixed.

From the data analysis in section 7.4, there were links between the data of Delphi survey results and detailed investigation. By taking the advantage of the hyperlinks of the World Wide Web, these links were represented as links on web pages. Another fact from the analysis was that many areas in this study would be benefited from further research. Further input from the Free/Open Source communities or any other parties to this evaluation would then be favourable and the evaluation was designed to accept comments.

8.3 Implementation of the Evaluation Presentations

From the discussion above, there would be two formats of presentation employed. For Delphi survey results, the format would be checklist with qualitative answers. For detailed investigation of external FOSPHost sites, a checklist with modified weighed answers would be used. The related issues between the two formats would be joined by hyperlink and the system should also accept comments from users of the evaluation. The implementation of the specification stated will be explained below. The choice of tools to accomplish the task will

first be introduced and then the actual implementation of the evaluation model will be presented.

8.3.1 Tools Chosen for Implementation

In this section, tools employed for the implementation of the evaluation will be introduced and explained. One of the most basic resources was to find a web hosting service. As the Delphi survey was executed online before, the researcher could use the same web hosting service again. Nonetheless, another hosting service was chosen for two reasons. First, from the Delphi survey results (Post-Delphi survey), the bandwidth of web server that the researcher could access in the University was too narrow. Second, it would be nobody's duty to maintain the site after the graduation of the researcher in his PhD programme, as this research project would end. ibiblio.org (ibiblio.org 2003a) was chosen as the host. ibiblio.org was 'a conservancy of freely available information, including software, music, literature, art, history, science, politics, and cultural studies.' (ibiblio.org 2003b) There were on average three million information requests per day and the site was well managed. Also, the site did not just provided its contributors with basic web hosting service, services such as web server-side scripts and database were also available. Tools such as PHP and MySQL were found to be indispensable in order to accomplish the task.

From the description above, ibiblio.org seemed to fit the needs of this research. Nevertheless, the question asked from the reverse direction 'did this research satisfy the requirement of becoming a collection of the site?' also needed to be considered. One of the aims of the site was to 'create, expand, improve, publish, and distribute research on the open source communities' (ibiblio.org 2003b). This research thus matched this goal. Another requirement was non-commercial. This project also did not generate any revenue and thus satisfied this condition. The researcher hence applied to the site and was successfully granted the status of a contributor.

For presenting the Delphi results as a checklist evaluation, Wiki was employed. Recalling that a Wiki is 'a freely expandable collection of interlinked Web "pages", a hypertext system for storing and modifying information - a database, where each page is easily editable by any user with a forms-capable Web browser client' (Leuf & Cunningham 2001, p. 14). It was chosen as Wiki is strong in its hyperlink functionalities and allows inputs from users.

There were many implementations of Wiki available (may be close or more than a hundred (Wiki Engines 2003)). WakkaWiki (Mans 2003) was chosen in this case as it provided several important features, namely comment area and access control. From the Delphi survey results, Wiki was voted as the most controversial tool. The reason may be due to its chaotic nature of Wiki (anyone can change anything). By having a separate comment area and access control, more control could be gained while users could still give their opinion. This was important, as the PhD programme required the author to produce original work. If others could 'contribute' directly to the Wiki online, it would be difficult to prove that the work was original. Therefore, commenting was chosen to be the feedback mechanism, rather than the standard Wiki practice of allowing anyone to change anything.

In the implementation of the modified weighed scale checklist, the method of tailor-made programming using PHP and MySQL was chosen as this checklist presentation was seldom found in other places. The tailor-made method could give the researcher total control on the design of the presentation of the checklist and thus this method was chosen.

Tools selected for the implementation was introduced in this section together with the rationale of the choices. The actual implementation will be presented in the next section.

8.3.2 Evaluation Model Implemented with the Chosen Tools

After the introduction of tools employed, the implementation of the presentations of the evaluation will be showed in this section. The implementation should be up and running at <http://www.ibiblio.org/fosphost/wakka/EvaluateFOSPHost> and the reader is encouraged to browse it online. The site map of the implementation is illustrated in Figure 8-1. The section that was implemented by Wiki contained mostly the results from the Delphi survey and it was divided into four sections, namely 'What is a Free/Open Source Hosting (FOSPHost) Site?', 'Common Objectives and Possible Benefits of Using a FOSPHost Site', 'Preferred Attributes of a FOSPHost Site' and 'Controversial Issues of FOSPHost Sites'. Issues on infrastructure and non-infrastructure sites also were included in Wiki under 'Controversial Issues of FOSPHost Sites'. On the other hand, the weighed checklist was implemented by PHP and MySQL. A comparison table was also implemented for those who did not want to go through the process of assigning weights on different features. Relevant items in both sections were linked by hyperlinks. Hyperlinks were also built to link statements in Wiki to the results from Delphi survey so that readers could discover the origins of these statements in their contexts.

From the site map (Figure 8-1), the front page of Wiki contained introductory information and links to six other sections (Figure 8-2). The idea of FOSPHost was explained in the page 'What is a Free/Open Source Hosting (FOSPHost) Site?' The classification of external hosting and self-hosting sites was also introduced and advantages for each type of sites were also stated. References to results of Delphi survey were made through the (Q?.?) links (Figure 8-3). The next section, 'Common Objectives and Possible Benefits of Using a FOSPHost Site', was designed to convey a realistic expectation to a FOSPHost site. The content was mainly made up of statements from question 1 (objectives) and question 6 (benefits) from the Delphi survey.

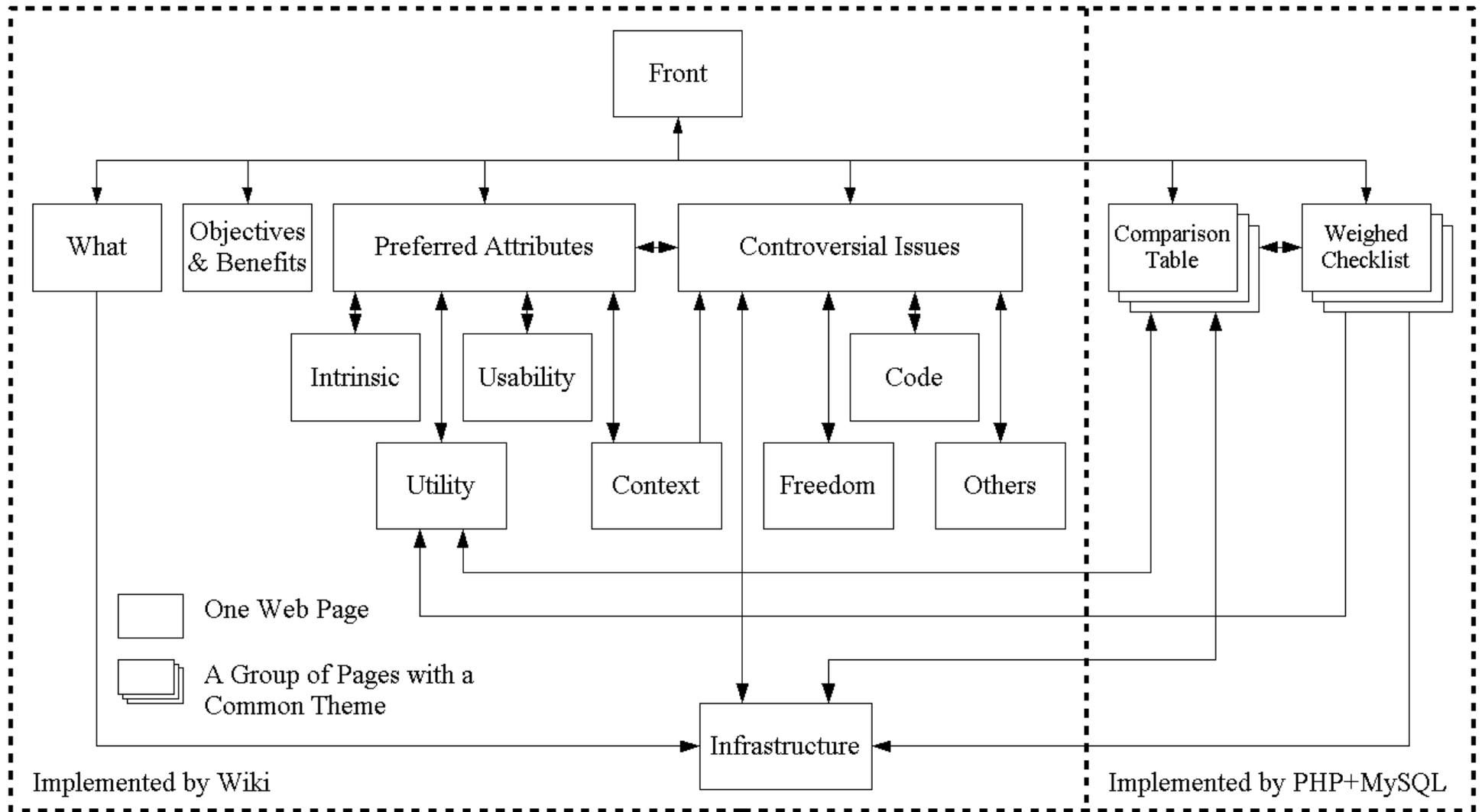


Figure 8-1 Site Map for Evaluation Model

Evaluation of FOSPHost Sites : EvaluateFOSPHost

EvaluateFOSPHost :: [PageIndex](#) :: [RecentChanges](#) :: [RecentlyCommented](#) :: [UserSettings](#) :: You are [AdMin](#)

Welcome to Evaluation of Free/Open Source Hosting (FOSPHost) Sites

On this site, you can find

- Checklists for different attributes of FOSPHost Sites
- A comparison table of features and policies of [external hosting sites](#)
- An feature evaluation of [external hosting sites](#) sites based on user-defined criteria

Comments are welcomed!! You can type them in on any page you like. Just select the link "[Display comments/form]" at the bottom of a page.

You may also find some links like this (Q?.?). These are references to the original answers a [survey](#) on important attributes of FOSPHost sites, which are the basis of this evaluation.

Please select one of the following to begin:

[What is a Free/Open Source Hosting \(FOSPHost\) Site?](#)
[Common Objectives and Possible Benefits of Using a FOSPHost Site](#)
[Preferred Attributes of a FOSPHost Site](#)
[Controversial Issues of FOSPHost Sites](#)
[Comparison of FOSPHost Sites Available for Hosting Projects Externally from Project Owners](#)
[Features Evaluation of FOSPHost Sites Available for Hosting Projects Externally from Project Owners based on User-defined Criteria](#)

About this Evaluation

This evaluation is the result of my PhD research on "Construction of an Evaluation Model for Free/Open Source Project Hosting (FOSPHost) sites". The source of data came from a [Delphi survey](#) and a detail investigation on different [external hosting sites](#).

One of the strength of Wiki is that users can change the content directly. Unfortunately, this function will be turn off before the examination process of my dissertation is completed. The reason is that a dissertation is supposed to be the original work of the author and direct updating of content makes it difficult to distinguish between original or contributed work (It is, of course, possible to distinguish as Wiki has versioning capability, but difficult). This function will be activated after the examination.

If you have any opinion not on the content, but the design and management of this evaluation, you can either post it at the [forum](#), make a comment on this page "[Display comments/form]" at the bottom of this page, or email to fosphost@ibiblio.org.

License

Since the data collected by the Delphi survey is released under [OpenContent License](#) (a local copy can be found [here](#)), the wiki content is released under the same license too.

The content in the comparison of FOSPHost sites Available for hosting projects externally from project owners is released in the public domain.

The software that drives the Delphi survey, comparison table and features evaluation is GPLed. It will be available soon.

How to use this Wiki or Wakka Thing

The original [WIKI](#) site that explains the concept behind Wiki
 There are many [Implementations](#) of Wiki and you are now using [WakkaWiki](#)

You can find the specific [WakkaFormatting](#) page for details formatting rule for this implementation of Wiki
 This is a [SandBox](#) that is setup for testing!!

Useful pages: [OrphanedPages](#), [WantedPages](#), [TextSearch](#).

There is no comment on this page. [\[Display comments/form\]](#)

[Edit this page](#) :: 2003-12-28 22:41:53 [XML](#) :: You are the owner of this page. :: [Edit ACLs](#) :: [Referrers](#) :: Search:

Figure 8-2 Front Page of Wiki

Evaluation of FOSPHost Sites : FOSPHost

[EvaluateFOSPHost](#) :: [PageIndex](#) :: [RecentChanges](#) :: [RecentlyCommented](#) :: [UserSettings](#) :: You are [AdMin](#)

What is a FOSPHost site?

A Free/Open Source Project Hosting (FOSPHost) site the infrastructure that supports and co-ordinates the development of Free/Open Source software projects on the Internet. One of the most famous example is SourceForge. Nevertheless, it can be as simple as a mailing list and a FTP server to download the components of the project.

External Hosting and Self-hosting

FOSPHost sites can also be classified as external hosting and self-hosting. The distinction between the two is the amount of control the users of the FOSPHost site have.

- Self-hosting sites - the users can adjust the internal configuration of the services provided
- External hosting sites - a fixed set of services is provided with a common configuration.

Advantages for External Hosting Sites

- More reliable than individually hosted servers ([Q6.13](#))
- Tools with better quality than individually hosted sites ([Q6.22](#))
- Decrease the startup cost of hosting an Free/Open Source Software (if the service is low cost or free) ([Q6.13](#))
- Decrease the workload comparing to administrating a FOSPHost site individually ([Q6.5](#))

Advantages for Self-hosting Sites

- Have control on the choice of tools hosted on-site and their configurations ([Q7.10](#))

Further classification for External Hosting Sites - [Infrastructure and Non-infrastructure](#)

Back to [Evaluation of FOSPHost Sites](#)

There is no comment on this page. [[Display comments/form](#)]

[Edit this page](#) :: [2003-12-27 01:11:32](#) [XML](#) :: You are the owner of this page. :: [Edit ACLs](#) :: [Referrers](#) :: Search:

[Valid XHTML 1.0 Transitional](#) :: [Valid CSS](#) :: Powered by [Wakka 0.1.2](#)

Figure 8-3 What is a FOSPHost Site?

The next section, 'Preferred Attributes of a FOSPHost Site', was made up of the agreed answers from the Delphi survey. As analyzed in sub-section 6.3.4, the agreed answers could be categorised using the software evaluation classification, namely intrinsic, utility, usability and context. This system was adopted and four corresponding sub-sections were thus created (Figure 8-4). In the intrinsic sub-section, preferred attributes related to the administration of a FOSPHost site were listed. These attributes were mainly contributed from the results of question 12 of the Delphi survey. In the utility sub-section, tools voted as important from question 2 of the Delphi survey were listed (Figure 8-5). They were linked directly to the comparison of FOSPHost sites implemented by PHP and MySQL so that the user could lookup how these features were implemented at each of the ten sites tabulated. The next sub-section was usability, where important usability factors for a FOSPHost site were listed. Most of these factors were obtained from answers to question 4 of the Delphi survey. For the last sub-section,

issues on context, culture and work practices were stated (Figure 8-6). Most of the content was obtained from the answers of question 3 of Delphi survey, which was a question asked exactly on these issues.

Evaluation of FOSPHost Sites : PreferredAttributes

[EvaluateFOSPHost](#) :: [PageIndex](#) :: [RecentChanges](#) :: [RecentlyCommented](#) :: [UserSettings](#) :: You are [AdMin](#)

Preferred Attributes of a FOSPHost Site

- [Intrinsic](#)
- [Utility](#)
- [Usability](#)
- [Context](#)

Also see [Controversial Issues of a FOSPHost Site](#)

Back to [Evaluation of FOSPHost Sites](#)

There is no comment on this page. [[Display comments/form](#)]

[Edit this page](#) :: 2003-12-05 02:09:41 [XML](#) :: You are the owner of this page. :: [Edit ACLs](#) :: [Referrers](#) :: Search:

[Valid XHTML 1.0 Transitional](#) :: [Valid CSS](#) :: Powered by [Wakka 0.1.2](#)

Figure 8-4 Preferred Attributes of a FOSPHost Site

Evaluation of FOSPHost Sites : Utility

[EvaluateFOSPHost](#) :: [PageIndex](#) :: [RecentChanges](#) :: [RecentlyCommented](#) :: [UserSettings](#) :: You are [AdMin](#)

Preferred Attributes - Utility

Utility refers to the functionalities of the software (Grudin 1992). Five most important tools of a FOSPHost are listed below

- Source Code Repository ([Q2.1](#))
- Mailing List ([Q2.2](#))
- WWW Server ([Q2.5](#))
- Tracking System ([Q2.4](#))
- Security Measures (e.g. SSH) ([Q2.11](#))

[Comparison table of the above features on external hosting sites](#)

Back to [Preferred Attributes of a FOSPHost Site](#)

Back to [Evaluation of FOSPHost Sites](#)

Grudin, J 1992, 'Utility and usability: research issues and development contexts', *Interacting with Computers*, vol. 4, no. 2, pp. 209-17.

There is no comment on this page. [[Display comments/form](#)]

[Edit this page](#) :: 2003-12-27 01:39:41 [XML](#) :: You are the owner of this page. :: [Edit ACLs](#) :: [Referrers](#) :: Search:

[Valid XHTML 1.0 Transitional](#) :: [Valid CSS](#) :: Powered by [Wakka 0.1.2](#)

Figure 8-5 Preferred Attributes - Utility

Evaluation of FOSPHost Sites : Context

[EvaluateFOSPHost](#) :: [PageIndex](#) :: [RecentChanges](#) :: [RecentlyCommented](#) :: [UserSettings](#) :: You are [AdMin](#)

Preferred Attributes - Context

The context of FOSPHost includes specifics of users, tasks and socio-organisational environments (Bevan 1995). The concerns listed below relate mostly to recommended work practices and culture of a community developed around a FOSPHost site. Be aware these are not just expectation towards a community, you should expect yourself to be judged by similar standard if you joined the community.

- Flexibility towards volunteers ([Q3.17](#))
- Fun, good spirit and hope ([Q3.37](#))
- Listening to others ([Q3.16](#))
- Openness in attitude, procedures and policies, no hidden agenda ([Q3.2](#), [Q3.21](#))
- Fair to all efforts ([Q12.5](#))
- Tolerance, respect and patience ([Q3.13](#))
- Sense of responsibility ([Q3.1](#))

Also see [Controversial Issues in a FOSPHost Site](#)

Back to [Preferred Attributes of a FOSPHost Site](#)

Back to [Evaluation of FOSPHost Sites](#)

Bevan, N 1995, 'Usability is Quality of Use', paper presented to Proceedings of the 6th International Conference on Human Computer Interaction, Yokohama, viewed 11 Feb 2003, <<http://www.usability.serco.com/papers/hcistd95.pdf>>.

There is no comment on this page. [[Display comments/form](#)]

[Edit this page](#) :: 2004-01-05 20:29:25 [XML](#) :: You are the owner of this page. :: [Edit ACLs](#) :: [Referrers](#) :: Search:

[Valid XHTML 1.0 Transitional](#) :: [Valid CSS](#) :: Powered by [Wakka 0.1.2](#)

Figure 8-6 Preferred Attributes - Context

The last section on Wiki was 'Controversial Issues of FOSPHost Sites'. Four sub-sections were created, namely 'Infrastructure and Non-Infrastructure Sites', 'We love freedom, but how far can it go?', 'What characteristics are admirable in source code?' and 'Other Controversial Issues' (Figure 8-7). Except for the sub-section 'Infrastructure and Non-Infrastructure Sites', which was based on the classification and analysis from the detailed investigation, the other three sub-sections were based on the results from the Delphi survey. From the analysis of controversial answers, the answers were categorised into four groups, namely 'We love freedom, but how far can it go?', 'What characteristics are admirable in source code?', 'What is a worthy motivation?' and 'Important but not urgent tasks'. Nevertheless, as the different issues within the sub-sections were presented mostly in point-form format, it was difficult to explain issues in 'What is a worthy motivation?', which required detail explanation. In order to solve this problem, three groups were formed rather than three, namely 'We love freedom, but how far can it go?' (Figure 8-8), 'What characteristics are admirable in source code?' and 'Other Controversial Issues'. Similar to preferred attributes, relevant statements were linked to Delphi

survey and the comparison of FOSPHost sites implemented by PHP and MySQL as references to users.

Evaluation of FOSPHost Sites : ControversialIssues

[EvaluateFOSPHost](#) :: [PageIndex](#) :: [RecentChanges](#) :: [RecentlyCommented](#) :: [UserSettings](#) :: You are [AdMin](#)

Controversial Issues of a FOSPHost Site

[Infrastructure and Non-Infrastructure Sites](#)
[We love freedom, but how far can it go?](#)
[What characteristics are admirable in source code?](#)
[Other Controversial Issues](#)

Also see [Preferred Attributes of a FOSPHost Site](#)

Back to [Evaluation of FOSPHost Sites](#)

There is no comment on this page. [[Display comments/form](#)]

[Edit this page](#) :: 2003-12-05 02:10:04 [XML](#) :: You are the owner of this page. :: [Edit ACLs](#) :: [Referrers](#) :: Search:

[Valid XHTML 1.0 Transitional](#) :: [Valid CSS](#) :: Powered by [Wakka 0.1.2](#)

Figure 8-7 Controversial Issues of a FOSPHost Site

Evaluation of FOSPHost Sites : FreedomvsControl

[EvaluateFOSPHost](#) :: [PageIndex](#) :: [RecentChanges](#) :: [RecentlyCommented](#) :: [UserSettings](#) :: You are [AdMin](#)

We love freedom, but how far can it go?

Freedom

- Give users as much freedom as possible to do anything ([Q3.9](#))
- Avoid force; promote tolerance, respect and patience ([Q3.25](#), [Q3.13](#))
- Embrace heterogeneity, do not promote any standard practice ([Q3.18](#), [Q3.19](#))

Control

- Problems with freedom – getting flame, extra 'collaboration' with unwanted parties ([Q5.21](#), [Q7.4](#))
- Use private or semi-private developer groups to reduce unwanted collaboration' ([Q12.14](#))
- Reinforce explicit development roles ([Q3.4](#))
- Low cost in setting up will encourage forking, unserious, unready or similar projects ([Q7.5](#), [Q7.8](#), [Q7.7](#), [Q7.6](#))
- Firmness in decision making ([Q3.23](#))
- Promote successful projects as examples for learning ([Q3.19](#))

Back to [Controversial Issues of a FOSPHost Site](#)

Back to [Evaluation of FOSPHost Sites](#)

There is no comment on this page. [[Display comments/form](#)]

[Edit this page](#) :: 2003-12-28 04:23:55 [XML](#) :: You are the owner of this page. :: [Edit ACLs](#) :: [Referrers](#) :: Search:

[Valid XHTML 1.0 Transitional](#) :: [Valid CSS](#) :: Powered by [Wakka 0.1.2](#)

Figure 8-8 We love freedom, but how far can it go?

After the detail description of the Wiki implementation, the implementation of the weighed checklist by PHP and MySQL will be explained. There were two major components in this implementation, the comparison table and the weighed evaluation. The comparison table will be introduced first.

The comparison table implemented was simply a concatenation of the comparison tables in chapter 7 with additional functionalities. The items showed on the table were arranged into different groups, namely 'General Information', 'Project Tools - Tools for Public/Developers', 'Project Tools - Tools for Project Administrators', 'Personal Tools for Developers', 'Community Tools', 'Others', which were the same as the grouping in chapter 7. A complete table was available on site in static html format. This table, however, was quite huge and users might just want a portion of the whole table. Options were available for them to generate a table with specified feature groups and sites (Figure 8-9). After the table was generated, in case the user wanted further adjustment, one could modify one's choice on the same page and re-generate the table (Figure 8-10 & Figure 8-11).

Comparison of Free/Open Source Project Hosting (FOSPHost) Sites Available for Hosting Projects Externally from Project Owners

Welcome to this comparison, you can either go to

[One gigantic included everything static html comparison table](#)

Or

Generate your own table

Items to Compare	Feature Groups
<input type="checkbox"/> Asynchrony	<input type="checkbox"/> General Information
<input type="checkbox"/> BerliOS	<input type="checkbox"/> Project Tools - Tools for Public/Developers
<input type="checkbox"/> freepository	<input type="checkbox"/> Project Tools - Tools for Project Administrators
<input type="checkbox"/> GBorg	<input type="checkbox"/> Personal Tools for Developers
<input type="checkbox"/> GForge	<input type="checkbox"/> Community Tools
<input type="checkbox"/> icculus.org	<input type="checkbox"/> Others
<input type="checkbox"/> Savannah	
<input type="checkbox"/> SEUL	
<input type="checkbox"/> SourceForge	
<input type="checkbox"/> SunSITE.dk	
<input type="button" value="Generate Table"/>	<input type="button" value="Clear"/>

Or

 [Go Back to Evaluation of FOSPHost Sites](#)

Figure 8-9 Opinions for Generating Customised Comparison Table

Comparison of Free/Open Source Project Hosting (FOSPHost) Sites Available for Hosting Projects Externally from Project Owners

Disclaimer: All the data from this table was obtained from Freely/Openly/Publicly available sources (e.g. browsing the sites, reading the source code). None of the data was received by the author on a confidential basis.

Others		
	icculus.org	SourceForge
Software for Web Interface of FOSPHost	Just a few HTML pages	SourceForge
License of Web Interface	Probably no need for a license	GPL (but download files not found and no CVS)
Development Methodology	Probably no need for a "Methodology"	Closed Source
License Requirement for Project Hosted	Free/Open Source projects	Free/Open Source Projects
Copyright		Data Preservation - contributors own the code but it will cannot be deleted from the site
Advertisement		
Legal and Language Related		
Flexibility (Possible relationship with different type of sites)	Flexible and informal to meet the need of individual developers, no restrictive infrastructure such as 'Role Management', elitist & selective on projects to form a smaller community where individual needs can be cared for	
Donation	Most donations go to developers' needs (e.g. new video card, motherboard)	
Miscellaneous		
Additional Services from sites in the same organisation		Part of OSDN (Open Source Development Network) with Slashdot, NewsForge, etc

Modify This Table

Items to Compare

- Asynchrony
- BerliOS
- freepository
- GBorg
- GForge
- icculus.org
- Savannah
- SEUL
- SourceForge
- SunSITE.dk

Feature Groups

- General Information
- Project Tools - Tools for Public/Developers
- Project Tools - Tools for Project Administrators
- Personal Tools for Developers
- Community Tools
- Others

Generate Table

Clear

Figure 8-10 Example of Customised Comparison Table Generated (1)

Note:

1. DIY stands for 'Doing It Yourself'. A number of hosts allow users to run CGI scripts and access databases. Therefore, some services (e.g Wiki) that are not immediately available can be setup by the users themselves.
2. www.gforge.org actually only hosts one project - GForge itself (so please do not go there and ask for hosting). Nevertheless, a good number of FOSPHost sites are employing GForge at the moment (see [List of GForge sites](#)). Therefore, by comparing 'GForge' with other sites, the comparison can roughly cover the sites that employ GForge (Each individual site has the choice of enabling/providing each of the service/functionality listed above, but it should be safe to assume most of the service/functionality enabled/provided will be similar.) The number of members and projects on www.gforge.org are thus irrelevant as they carry a different meaning comparing to other sites.

Criteria for candidate selection:

1. Available to host new projects (so [WebFrame](#) is not included).
2. I can only read English and Chinese so site such as [TuxFamily.org](#) cannot be evaluated. Sorry.

Technically, GForge violate rule no. 1, as it is not available for new projects. Nonetheless, as explained above, 'GForge' actually represents the sites that employ GForge (see [List of GForge sites](#)) and a number of these sites do accept new projects. Therefore, it is included in this comparison.

A number of possible candidates (e.g. [ibiblio](#) and [Tigris.org](#)) did not respond to the invitations to be compared on this table and thus they are not included. The next possible candidate may be [Novell Forge](#), but it will take some time.

Credits:

- Roger Dingleline
- Ryan Gordon
- John Minnihan
- Christian Reiniger
- Chris Ryan

You can also express your opinion at the discussion [forum](#).

This comparison is done by [Haggen So](#) and the data of the comparison is released into the public domain. The PHP scripts that drive the site can also be shared under GPL.

 [Evaluate sites based on criteria defined by you](#)



[Go Back to Evaluation of FOSPHost Sites](#)



[Go Back to the First Page of the Site](#)

Figure 8-11 Example of Customised Comparison Table Generated (2)

After the explanation of the implementation of the comparison table, detail description of the weighed checklist can be found below. There were three steps involved in using the checklist.

In the first step, the user was asked to specify which feature groups should be included for evaluation. This groups were the same as the grouping in the comparison table. Users were also asked to define several features classes, which were the weights to be used. The default included three classes, indispensable, important and nice to have. Different colours could also be assigned to each of the classes (Figure 8-12).

Evaluation of Free/Open Source Project Hosting (FOSPHost) Sites Available for Hosting Projects Externally from Project Owners based on User-defined Criteria

Welcome to this evaluation. The process of this evaluation should be reasonably obvious. Nevertheless, rather than explaining everything in details, you can try this evaluation out by choosing different configurations and options and gradually approach your desired result.

Please choose from the following feature groups

- Project Tools - Tools for Public/Developers** includes:
Standardized Format for General Information, Free Format HTML Project Pages, Project Role Assignment, Project News, Download Service, Document Management, Statistics, Source Code Repository, Mailing List, Tracker, IRC, Forum, Wiki, Survey
- Project Tools - Tools for Project Administrators** includes:
Management for 'Tools for Public/Developers', Ask for Help (Recruitment), Activity History, Web Server-Side Scripts, Database, Shell, Compile Farm, Export, Virtual Hosting, Uploading, Security
- Personal Tools for Developers** includes:
Web Space to Host Personal Information
- Project Tools - Tools for Project Administrators** includes:
Backup
- Personal Tools for Developers** includes:
Skill and Experience, Tracker/Forum/File Monitoring, Projects Involved, Assigned/Submitted Issues from Trackers, Survey, Webmail, Diary, Bookmark, Money Earned, Rating
- Community Tools** includes:
Access, Search Project, Project Listing at Front Page, Classification, Search People, Project Help Wanted, Latest News, Get Support

Customisation of feature classes

Indispensable	Important	Nice to have				
<input type="radio"/> None	<input type="radio"/> None	<input type="radio"/> None	<input checked="" type="radio"/> None	<input checked="" type="radio"/> None	<input checked="" type="radio"/> None	<input checked="" type="radio"/> None
<input type="radio"/> Red	<input type="radio"/> Red	<input type="radio"/> Red	<input type="radio"/> Red	<input type="radio"/> Red	<input type="radio"/> Red	<input type="radio"/> Red
<input type="radio"/> Lime	<input checked="" type="radio"/> Lime	<input type="radio"/> Lime	<input type="radio"/> Lime	<input type="radio"/> Lime	<input type="radio"/> Lime	<input type="radio"/> Lime
<input type="radio"/> Blue	<input type="radio"/> Blue	<input type="radio"/> Blue	<input type="radio"/> Blue	<input type="radio"/> Blue	<input type="radio"/> Blue	<input type="radio"/> Blue
<input type="radio"/> Aqua	<input type="radio"/> Aqua	<input type="radio"/> Aqua	<input type="radio"/> Aqua	<input type="radio"/> Aqua	<input type="radio"/> Aqua	<input type="radio"/> Aqua
<input checked="" type="radio"/> Fuchsia	<input type="radio"/> Fuchsia	<input type="radio"/> Fuchsia	<input type="radio"/> Fuchsia	<input type="radio"/> Fuchsia	<input type="radio"/> Fuchsia	<input type="radio"/> Fuchsia
<input type="radio"/> Yellow	<input type="radio"/> Yellow	<input checked="" type="radio"/> Yellow	<input type="radio"/> Yellow	<input type="radio"/> Yellow	<input type="radio"/> Yellow	<input type="radio"/> Yellow
<input type="radio"/> Orange	<input type="radio"/> Orange	<input type="radio"/> Orange	<input type="radio"/> Orange	<input type="radio"/> Orange	<input type="radio"/> Orange	<input type="radio"/> Orange

Generate Evaluation Form Clear

 [Go Back to Evaluation of FOSPHost Sites](#)

Figure 8-12 Step 1 of Weighed Checklist

After choosing these options, an evaluation form was generated and it was the second step of employing the checklist. In the evaluation form generated, the users could indicate the importance of each feature in the feature groups chosen by assigning it to an appropriate feature class. The user could also choose the sites to be included in the evaluation (Figure 8-13).

After the user had assigned the importance of the features to the feature classes, the scores of each of the sites selected were calculated and presented in the result page, which was the third and final step. The calculation of the score was that 1 was awarded for a presence of the feature and 0 for none. The score for each feature class was calculated individually. Finally, the sites were ranked according to the score of the most important feature class and then the classes followed. An exception was given to features that required DIY. Recalling DIY meant that users of a FOSPHost site could install a certain tool on their own accord. DIY could be a nuisance or an advantage, depending on the need and the attitude of the user. Therefore, it was designed so that the user of the checklist could choose to award a DIY item a score from 0 to 2 to account for this variation (Figure 8-14 & Figure 8-15).

From the screen captures (Figure 8-12, Figure 8-13, Figure 8-14 & Figure 8-15), one may notice that a different name, 'Evaluation of Free/Open Source Project Hosting (FOSPHost) Sites Available for Hosting Projects Externally from Project Owners based on User-defined Criteria', was given to the title of the web pages of the weighed checklist rather than variations of the phrase 'weighed checklist'. This was done so because it was too complicated to explain what was a modified weighed checklist and why was implemented. Alternatively, the user was encouraged to iterate through the three steps a few times to generate the desired result rather than presenting an explicit explanation in details.

Evaluation of Free/Open Source Project Hosting (FOSPHost) Sites Available for Hosting Projects Externally from Project Owners based on User-defined Criteria

Great!! Now you have generated an evaluation form for yourself. If you want to adjust the form, please go [back](#). Otherwise, filling the criteria and options and the system will sort the rest out.

Personal Tools for Developers				
	Indispensable	Important	Nice to have	
Web Space to Host Personal Information	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	Clear
Skill and Experience	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Clear
Tracker/Forum/File Monitoring	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Clear
Projects Involved	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Clear
Assigned/Submitted Issues from Trackers	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Clear
Survey	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	Clear
Webmail	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	Clear
Diary	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Clear
Bookmark	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Clear
Money Earned	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	Clear
Rating	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Clear

Additional Options

Items to Compare

- Asynchrony
- BerliOS
- freepository
- GBorg
- GForge
- icculus.org
- Savannah
- SEUL
- SourceForge
- SunSITE.dk

DIY

Show Match By Colour By a Tick

Note:

- DIY stands for 'Doing It Yourself'. A number of hosts allow users to run CGI scripts and access databases. Therefore, some services (e.g Wiki) that are not immediately available can be setup by the users themselves. In the evaluation, a score of 1 will be awarded to a feature present and 0 to absent. DIY is regarded as a grey area and it is up to you to give a value.

 [Go Back to Last Page to re-generate this form](#)

 [Go Back to Evaluation of FOSPHost Sites](#)

Figure 8-13 Step 2 of Weighed Checklist

Evaluation of Free/Open Source Project Hosting (FOSPHost) Sites Available for Hosting Projects Externally from Project Owners based on User-defined Criteria

Great!! Now you have generated an evaluation form for yourself. If you want to adjust the form, please go [back](#). Otherwise, filling the criteria and options and the system will sort the rest out.

Personal Tools for Developers							
		icculus.org	GForge 2	BerliOS	SunSITE.dk	SEUL	Asynchrony
Web Space to Host Personal Information	Indispensable	Yes ✓	No	No	No	No	No
Skill and Experience		DIY 1	Yes	Yes	No	No	Yes
Tracker/Forum/File Monitoring		DIY 1	Yes	Yes	No	No	Unknown
Projects Involved		DIY 1	Yes	Yes	No	No	Yes
Assigned/Submitted Issues from Trackers		DIY 1	Yes	Yes	No	No	Unknown
Survey	Indispensable	DIY 1 ✓	Yes ✓	Yes ✓	No	No	No
Webmail	Important	Yes ✓	Redirect ✓	Redirect ✓	DIY 1 ✓	DIY 1 ✓	Redirect ✓
Diary		DIY 1	Yes	Yes	No	No	No
Bookmark		DIY 1	Yes	Yes	No	No	No
Money Earned	Nice to have	DIY 1 ✓	No	No	No	No	Yes ✓
Rating		No	Yes	Yes	No	No	Yes
Final Score							
		icculus.org	GForge 2	BerliOS	SunSITE.dk	SEUL	Asynchrony
		Indispensable 2.25	Indispensable 1	Indispensable 1	Indispensable 0	Indispensable 0	Indispensable 0
		Important 1	Important 1	Important 1	Important 1.25	Important 1.25	Important 1
		Nice to have 1.25	Nice to have 0	Nice to have 0	Nice to have 0	Nice to have 0	Nice to have 1

Make sure that you also refer to the general information and policies of each items. It can be found [here](#).

Note:

- DIY stands for 'Doing It Yourself'. A number of hosts allow users to run CGI scripts and access databases. Therefore, some services (e.g Wiki) that are not immediately available can be setup by the users themselves.
- [www.gforge.org](#) actually only hosts one project - GForge itself (so please do not go there and ask for hosting). Nevertheless, a good number of FOSPHost sites are employing GForge at the moment (see [List of GForge sites](#)). Therefore, by comparing 'GForge' with other sites, the comparison can roughly cover the sites that employ GForge (Each individual site has the choice of enabling/providing each of the service/functionality listed above, but it should be safe to assume most of the service/functionality enabled/provided will be similar.) The number of members and projects on [www.gforge.org](#) are thus irrelevant as they carry a different meaning comparing to other sites.

Criteria for candidate selection:

- Available to host new projects (so [WebFrame](#) is not included).
- I can only read English and Chinese so site such as [TuxFamily.org](#) cannot be evaluated. Sorry.

Technically, GForge violate rule no. 1, as it is not available for new projects. Nonetheless, as explained above, 'GForge' actually represents the sites that employ GForge (see [List of GForge sites](#)) and a number of these sites do accept new projects. Therefore, it is included in this comparison.

A number of possible candidates (e.g. [ibiblio](#) and [Tigris.org](#)) did not respond to the invitations to be compared on this table and thus they are not included. The next possible candidate may be [Novell Forge](#), but it will take some time.

Figure 8-14 Step 3 of Weighed Checklist (1)

Credits:

- Roger Dingleline
- Ryan Gordon
- John Minnihan
- Christian Reiniger
- Chris Ryan

You can also express your opinion at the discussion [forum](#).

This comparison is done by [Haggen So](#) and the data of the comparison is released into the public domain. The PHP scripts that drive the site can also be shared under GPL.

 [Go Back to Change the Criteria](#)

 [Go Back to Evaluation Form Generation](#)

 [Obtain a comparison table of the sites](#)

 [Go Back to Evaluation of FOSPHost Sites](#)

 [Go Back to the First Page of the Site](#)

Figure 8-15 Step 3 of Weighed Checklist (2)

In this section, the implementation of the evaluation model was explained and screen captures were showed. Two different types of tools were employed in the implementation, namely Wiki and PHP with MySQL and the content of the data collected was presented.

8.4 Discussion of the Evaluation Model

After the description of the implementation of the evaluation model, the quality of the model itself can be evaluated also. A suitable evaluation presentation format was chosen based on the nature of the results of the Delphi survey and the detailed investigation of external hosting FOSPHost sites. Recalling that the main focus of the detailed investigation was selected to be on the features of the FOSPHost sites to accommodate the expectations of both Free/Open Source developers and new comers. The weighed checklist then could be a helpful tool for sites comparison based on features offered. Hyperlinks were also used extensively to related relevant pieces of information all the way back to their origins upon the empirical results collected. The system also accepted comments from the users of the evaluation. Therefore, it can be regarded that the original specification of the evaluation model was successfully implemented.

Nevertheless, further examination can be performed for further improvement. Several limitations could actually be found. As stated above, it would be difficult to present the quantitative results obtained from the Delphi survey together with the qualitative presentation. Moreover, while benefited from the breadth of the results collected from the Delphi survey, the arguments in the preferred attributes in the sub-sections on intrinsic, usability and context would certainly be strengthened if more in-depth data was available.

Within the constraints of the present research programme, these limitations would be difficult to eliminate. Nonetheless, in terms of presenting quantitative ratings to qualitative statements, modification of the Wiki engine may be required. The specific implementation of Wiki engine chosen for the implementation of the evaluation, WakkaWiki, was a lightweight implementation in terms of the amount of source code used. The decision to choose a lightweight implementation was done intentionally so that alteration to the code could be done more easily. Furthermore, the edit page function of the Wiki will be activated after the completion of the examination process of this dissertation. Hopefully, more opinion could be collected to suggest directions for further in-depth research in the area of FOSPHost. In short, by taking the advantage of Free/Open Source, in source code and in concept, the site was designed for growth and thus the limitations stated above can be more readily overcome in the future.

8.5 Summary of Chapter Eight

In this chapter, the construction process of the final product of the research, an evaluation model for FOSPHost sites, was presented. The chapter started by reviewing the nature of the results obtained from the Delphi survey and the detailed investigation of external hosting FOSPHost sites. Qualitative presentation and modified weighed checklist presentation were chosen correspondingly and a detail specification was devised. Tools were chosen accordingly and the

specification was successfully implemented. Though limitations to the implementation were found, promising methods of improvement were suggested.

In the next chapter, the overall quality of the result and the evaluation model will be discussed. A further interpretation of the implication of the results together with other supportive literature will be presented.

Chapter 9

Discussion of Results

9.1 Introduction

In this chapter, the overall quality of the findings and the evaluation model presented on the web will be examined. Implications arise from the results will be explored. These implications will be connected to the wider discussion of the Free/Open Source phenomenon.

9.2 Discussions and Limitations of this research

In this project, results from the Delphi survey and the detailed investigation were presented in an evaluation web site constructed to the evaluation presentation chosen. The rough estimate of the responds from the Free/Open Source communities can be obtained by a google search on 'www.ibiblio.org/fosphost/'. About 50 entries were obtained on 30 August 2005 (A few of the 50 entries obtained was the related to the attacks from indecent spam on Wiki, so the actually number should be lower). Many of the web pages in the search result were Free/Open Source related sites or mailing lists concerning the topic of FOSPHost. Most of them prefer the table format rather than the evaluation format (For example, a good number of links referred to 'http://www.ibiblio.org/fosphost/exhost.htm', which is the all-in-one comparison table). The content of the Wiki was also less discussed. Moreover, even in the table format, the utility aspect was the more popular. One of the sites that linked to the evaluation, 'Loads of Linux Links' (Willard & Irwin 2005), picked the link to the comparison of 'Project Tools - Tools for

Public/Developers' and 'Project Tools - Tools for Project Administrators' only. Moreover, a leader from the communities, Karl Fogel, who is a leader developer of CVS and Subversion, and also wrote an important book on version control systems and the co-ordination of Free/Open Source project (Fogel 1999), also left a thank you note on the evaluation site forum. A former version of this dissertation was emailed to him on his request. It is possible that some of the result in this dissertation will be included in the coming edition of his book.

The estimate obtained above thus supports the assumption that the Free/Open Source communities would prefer a detailed investigation. It was also no surprise that the communities prefer the utility aspect of the comparison. The evaluation mechanism was also not preferred. Three other aspects, namely intrinsic, usability and context, are built from the result Delphi and the content were relatively thin. By employing a Delphi survey using the flexible model of individual participation to a Free/Open Source community as a basis of the initial questionnaire, a broader range of data could be collected but the depth for each of the answer collected was less. A similar situation occurred in the detailed investigation of external hosting sites. The basic sources of data were the presentation of the sites on the Internet, on site documentation and the source code of the site. It was difficult to reach a more solid conclusion on the nature of infrastructure and non-infrastructure sites as the rationale of the design of these sites seldom was explicitly stated in the sources of data investigated. Comments from some site administrators from the confirmation process of the comparison tables were found to be invaluable in understanding their rationale and thus conducting interviews may be a promising direction to obtain such data. Nevertheless, due to the limitation of resources, the research had to stop at this point. Therefore, the evaluation model, which was built on the data collected, covered a broad range of topics. It would be desirable to cover each topic in depth, but it was again the limitation of this study. This shortcoming definitely decreases usefulness of this evaluation to the user.

Another limitation is that the presentation also could not sufficiently represent the quantitative data collected during the Delphi survey. It could be improved by modifying the source code of WakkaWiki but again this is another area for further research. Moreover, only the developers were the focus of the model of individual participation to a Free/Open Source community and views from other stakeholders may not be collected.

As a summary, the research probably fulfilled its original purpose as an exploratory research. Under this purpose, in-depth investigation was not the priority and therefore one of the obvious limitations was the depth of the study. Nonetheless, the results of this research probably have open ways to further studies.

9.3 Implications of the Findings – Free/Open Source as a Different Paradigm

In this section, the results obtained will be further interpreted and related to literature and the Free/Open Source phenomenon itself. The proposition of Free/Open Source software development process as a radically different method comparing with conventional software engineering process (the Bazaar model) will be examined. The results of this examination will then shed light on how the Free/Open Source phenomenon could be investigated and understood.

At the commencement of this dissertation, the idea of using the metaphor of the Cathedral and the Bazaar (Raymond 2000b) to demonstrate the differences between the traditional software engineering approach and the Free/Open Source software development process was introduced. What does the results of the Delphi survey imply on the nature of the Free/Open Source software development process? Is it a radically different process, or it is just a change in some of the parameters in managing a software project?

The examination of the question starts from the 'Cathedral' metaphor. Johnson (1999) suggested that the metaphor referred to the waterfall model of software development process (Royce 1970). This process starts from system requirements, software requirements, analysis, program design, coding, to testing and operations. Several assumptions seemed to be made. The requirement of the system was assumed to be relatively stable with clear purposes. Or in short, order was expected in whole process and if something went wrong, for it to be pulled back to order. As software development processes are seldom linear, a number of modifications were added to the waterfall model to incorporate iterative elements. Nevertheless, the basic assumption of orderliness still remained.

Another aspect is that process is the focus of the development and programmers are assumed to be more or less obedient. They will still follow the process prescribed. An example of such view can be found in the Capability Maturity Models (CMM) (Software Engineering Institute 2003a). These models were established to improve the quality of software produced. Three dimensions were recognized, namely, process, technology and people, which would affect quality. Process was chosen to be the key dimension (Bate et al. 1995). Even in the model that is probably closest to individual programmers, the Personal Software Process (PSP) (Humphrey 2000), the idea was to improve a programmer's performance by adopting certain disciplines. Variables such as time, size of code written and defects were measured and fed back to the programmers as well. The goal is to improve the overall productivity. Factors on work environment and psychology of programmers were also addressed in the People Capability Maturity Model (P-CMM) (Curtis, Hefley & Miller 1995) such as communication, staffing, career development, managing performance and team building. Other materials have also been written for managers to exploit these factors and maximise the motivations of programmers (DeTienne, Smart & Jones 1995; Humphrey 1997; Whitehead 2001). For

example, from the book 'Managing Technical People' written by the founder and a fellow of the Software Engineering Institute, Watts Humphrey (1997), much insightful advice on managing programmers was given. Techniques such as recognition, delegation, and feedback of performances were discussed. Nevertheless, the bottom line seems to be to meet the target with the limited resources that a programmer has. The process dimension is probably placed in a high priority than the human dimension.

Referring to the discussion of the Free/Open Source software development process, a number of academics also proposed different models of the process (Aoki et al. 2001; Jorgensen 2001; Nakakoji et al. 2002; Wu & Lin 2001). Some of these models even have similarities with the waterfall model. On the other hand, Jones (2000) found difficulties in comparing conventional software engineering with in Free/Open Source software development. He was originally trying to investigate whether Brooks's law was proven wrong by the Free/Open Source software development process. An obvious place to investigate was the schedules of software projects. Then he found that the scheduling practices in the two processes were different and thus difficult to use this method of comparison to draw conclusion on Brooks's law. An interviewee in that article compared the flexibility in scheduling in Free/Open Source software development with Michelangelo creating an artistic masterpiece. 49% of the participants in BCG Hacker survey also agreed that programming was 'like composing poetry or music' (Lakhani et al. 2003). This confirms with one of the top agreed practices to be promoted – flexibility towards volunteers (Q3.17).

Another difference could be that Free/Open Source was said to be 'very different from corporate monocultures' (Q3.18). As a Free/Open Source project is run openly, it is then more possible to encounter unexpected voices and even accidents. Considering Linux, the Free/Open Source project 'par excellence', was named 'an accidental revolution' in a biography of Linus Torvalds

(Torvalds & Diamond 2001). In fact, accidents are anticipated and welcomed. Raymond (2000b) explained that Free/Open Source software may be used for purposes beyond its original design. By communication with end-users, such unexpected needs can be heard and accommodated. Fogel (1999) explained in a similar idea under the name of evolution-centered design. The essence of this design is flexibility and comprehensibility so that the code-base can be used and re-used to solve programs that one may not even expect. An example was that there should not be an arbitrary length limit on an input stream. Without such limit, a data structure that holds text strings can be used to hold binary graphic data instead. This approach in project management is seldom heard in conventional software engineering practices. The process dimension seems to take a lower priority.

As discussed above, in conventional software engineering, the process dimension is more important than the people dimension. Techniques to motivate programmers were employed, but the aim seems to be to maintain the process. According to research on motivation in work (Herzberg, Mausner & Snyderman 1959) and also specific surveys in the field of information technology (Couger 1988), the nature of work (originally called 'work itself' in the surveys) was the among the top motivators. If a programmer who is not even happy with the nature of the job of computer programming, he or she is unlikely to have the motivation to pursuit excellence or enjoy the challenge of an aggressive schedule suggested by Humphrey (1997). In contrast, most participants of Free/Open Source projects were found to be highly motivated due to the love of programming. Lakhani et al. (2003) surveyed 684 developers on SourceForge and found 60% of the participants agreed that 'With one more hour in the day, I would spend it programming'. Another survey by found that 80% of respondents participated in Free/Open Source due to self-determination (Hars & Ou 2001). Flexibility in management (Q3.17) may also increase the possibility in creating a win-win situation to achieve both the project goal and personal goals of the developers. Keeping developers happy is one of the recommended

practices (Q3.37). This practice of flexibility and emphasis on motivation of developers was again reflected from the non-infrastructure FOSPHost from the detailed investigation. Again, this is different from the traditional software development situation mentioned above where the bottom line seems to be to a programmer has meet the project target with limited resources.

This difference in goal may cause different consequences to arise. Powell (2002) observed a circumstance in the KDE project that the developers picked tasks that interested them and 'boring' tasks were accumulating as the project grew. On the other hand, motivation can also be utilised to improve project efficiency. For example, Collab.Net is a company selling web-based collaborative software development environments and consulting services inspired by Free/Open Source software development processes. In this web-based environment, different permissions can be assigned (Collab.Net 2003b). One application of this permission feature is about protecting intellectual property of the company, but another use of this feature is not to make all source code immediately available internally to employed developers. They had to 'earn' their right to access some of the source code, which creates an extra motivation system (Carpenter 2001). This is different from conventional practice as process, rather than motivation, is emphasized (Robbins 2002).

When the consideration of the purpose of a software project becomes more diverse, how the benefits a project is evaluated may also be changed. In a discussion on project failure on Advogato, higg said:

'I consider my mpEDIT project to be dead, but it got to be one chapter in somebody's book, was used in a college course, and was picked up briefly by the NCSA. Along the way it helped people learn programming. Even if it never got to be a widely used tool, I count it as a measured success'.
(Advogato 2000c)

The example may suggest a more comprehensive consideration of the benefits, rather than just judging from the functionality and usability of software. Indeed, from the results of the survey, a number of different values of software are obtained such as learning (Q3.6) and fun (Q3.37).

Other than qualitative arguments, there is quantitative evidence from empirical data that some Free/Open Source Software can evolve differently from conventional closed source development in the laws of software evolution (Scacchi 2003). During the investigation of these discrepancies, Scacchi suggested that Free/Open Source software 'constitute a distinct technological regime' (Scacchi 2003, p. 25). To understand the evolution process of Free/Open Source software, alternative ontologies needed to be established. From the discussion above, some important rules are different in the world of Free/Open Source software such as order, project goals and motivation. Borrowing from science fiction or cosmological physics, there may be many possible universes out there. The one that is familiar, software engineering, may be only one of these many possibilities. Free/Open Source software development allows developers to run project on their own terms, and thus it is possible to run software projects in many different ways. Within the results of the survey, embracing practice of software engineering (Q3.8), aiming to produce useful software (Q1.36, Q3.6), seeking to reuse software (Q3.5, Q3.7), reinforcing explicit roles in projects (Q3.9) and not including inexperienced developers (Q1.28) are closer to conventional values in software engineering, but they represent only a part of the opinion voiced. A number of different universes could be out there.

If the Free/Open Source phenomenon is really a collection of other universes, is there any chance to find rules within them? Referring back to Jones' (2000) article on Brooks's law, he seemed to suggest that conventional methods of measuring project progress such as schedule comparison may not apply, but it did not prove or disprove the law. In contrast, the inclusion of

an application of Brooks's law in project management within Red Hat seemed to suggest that Brooks's law still holds. Another piece of evident is found when Mockus, Fielding & Herbsleb (2002) compared two well-known Free/Open Source projects, Mozilla and Apache. They found that Mozilla had a higher level of module interdependence comparing to Apache. A larger core group for each module was thus needed and the speed of development was slower. This may suggest a confirmation of Brooks's law rather than disproving it. If we refer back to the metaphor, when one enters another universe, things may look chaotic, because some of the laws of physics are different. Nonetheless, laws may still exist, and some may even be the same as before, but others may be unknown to newcomers, as they bring the assumptions from the previous universe. Similar in the study of Free/Open Source, when some of the ground rules change, many questions that are not significant before needed to be asked and investigated. Nevertheless, it is optimistic that rules can be found, as some rules may be still the same.

A few words need to be said on the arguments laid above. The arguments may seem to portray a positive impression that Free/Open Source software development process is better than traditional software development process. A number of arguments presented above can be regarded as responses to the question, 'What works in Free/Open Source that fails in Closed-Source?' With this assumption, the impression of the argument will probably be in favour to Free/Open Source. It is common to find studies in success factors in success projects such as Linux (Moon & Sproull 2000) Mozilla and Apache (Mockus, A., Fielding & Herbsleb 2002) but research on failure factors in failed projects are hard to find. One example of failure is the number of abandoned projects on SourceForge. As discussed above, diversities exist in Free/Open Source and it is not the author's intention to support sweeping statements such as 'the Free/Open Source software development process is better'. More research is needed to define and categorise different Free/Open Source projects, processes and developers. Then failure factors can be probably more readily uncovered.

Another impression that the argument above may portray is that Free/Open Source is considerably different. This again is due to the question asked at the beginning that it is an investigation of differences. Even with the source code freed/opened, it is possible to run the project using the waterfall model. It is argued by Collab.Net (2003c) that the processes support on a FOSPHost site can also support CMM goals (Though the original analysis is based on the Closed-Source product of Collab.Net, SourceCast, which was inspired by FOSPHost sites, many of the arguments in the paper referred can still apply to general FOSPHost sites). Saying 'Some of the Free/Open Source processes can be similar to traditional software development process' is not a contradiction to the multiple universes metaphor. Particular types of Free/Open Source universes may be close to the traditional universe that we are familiar with and other Free/Open Source universes may be far away.

To conclude, one of the most agreed results from Delphi survey was the importance of communication. It is not surprising as software is flexible in multiple dimensions and thus software development is one of the most complicated human processes. In the Free/Open Source communities, communication is even more essential as participants can come from a diverse background with different purposes and styles. Mutual understanding can hardly be achieved without some of the attitudes suggested such as tolerance (Q3.13), patience (Q3.13) and listening to others (Q3.16). Maybe it is also time for academics to communicate more with the communities and find out what the other universes look like. Existing knowledge from the academic world is probably useful to the Free/Open Source communities (for example Wilson (1999) stressed the important of design and good practices in programming, not heroic stories of bug eradication at 2:00 a.m.), but the context of how it can be applied needs further exploration to ensure relevancy.

9.4 Summary of Chapter Nine

In this chapter, the overall quality of the evaluation was assessed. The utility dimension was probably the most popular dimension of the four. Other limitations of the research were identified as well. The result of the research was further interpreted to contribute to the discussion of the nature of the Free/Open Source phenomenon.

In the next chapter, the summary of the dissertation will be presented. Areas for further research with suggestions to potential methodologies and methods will be explored. Finally, the possible future of software industry and how the results of this research may be even more important in such context will be discussed.

Chapter 10

Conclusion

10.1 Introduction

In this chapter, a summary of the findings of this study will be presented. The contributions of the findings to the body of knowledge will be discussed. Further research on both the specific area of FOSPHost and the general area of the Free/Open Source phenomenon will be suggested. Research methodologies and methods will be suggested for further investigation of the general area of the Free/Open Source phenomenon. Based on 'futurological' academic writings and economic analysis, the possible future of the software industry will be projected and the potential contributions of the findings will be discussed.

10.2 Summary of Findings

The summary of the findings from frameworks developed to the evaluation model built will be presented below. The contributions of knowledge of the findings will also be discussed.

Recalling the objectives of the research are 'discovering the areas relevant to the topic of FOSPHost and establishing the boundaries for data collection. Analytical frameworks will be built from literature as a starting point for investigation. Important issues in the design and employment of FOSPHost sites will then be obtained. The findings will be presented in an

evaluation format available on the Internet.' Following these objectives, the research question and sub-questions formulated are:

'How to construct an evaluation model for a FOSPHost site?'

And the research sub-questions are:

1. What relevant analytical frameworks can be built to facilitate the investigation of the design and deployment of FOSPHost?
2. What are the important factors in FOSPHost design and deployment from data collection?
3. How to build an evaluation model from these important factors in FOSPHost?

How the result of this research responded to above sub-questions will be presented in the following sub-sections.

10.2.1 The Model of Individual Participation to a Free/Open Source Community and Software Evaluation Classification

At the beginning of this research project, the most influential explanation to the Free/Open Source phenomenon was the Bazaar model. As argued in sub-section 2.3, the model by itself did not provide enough details to explain the inner workings of the phenomenon. The model of individual participation to a Free/Open Source community was thus devised to establish important aspects within a Free/Open Source community, namely communication, contributions, co-ordination and culture. Essential elements such as motivations, barriers, positive and negative results from the developers' viewpoint were also included. This model were compared with four other models and found to be a suitable framework for research in FOSPHost. This was the response to research sub-question number 1.

The study is also about evaluation and thus the methods employed in software evaluation were reviewed. Two separate software evaluation approaches were identified, namely software

evaluation during development and software product evaluation, which were usually employed by the development team and the users of the software respectively. As in the situation of Free/Open Source, though the users of the software may not also be the developers, the developers of the software are probably the users. Moreover, the developers were usually quite accessible to the users as well. Therefore, the two approaches were merged to become a new evaluation classification framework, which includes the evaluation of intrinsic, utility, usability and context qualities of the software. This framework laid the foundation for the response to research question number 3.

The research was then built on these two frameworks for further investigation and data collection.

10.2.2 Delphi Survey

One of the major data collections was conducted through an online Delphi survey. A total of 32 experts participated in 3 rounds of Delphi survey and the validity of the survey was acceptable. 61 agreed statements and 65 controversial statements were obtained. The software evaluation classification was found to be useful in analysing the agreed statements. Recommendations in all four software evaluation categories were found. For the controversial statements, four themes were developed, namely 'We love freedom, but how far can it go?', 'What characteristics are admirable in source code?', 'What is a worthy motivation?' and 'Important but not urgent tasks'. These themes suggested that there were diverse views in some practices on FOSPHost sites. The data collected was available on the Internet at <http://www.ibiblio.org/fosphost/IFHOSP/ibibliologo.htm?URL=index.html>. The result of the Delphi survey was a part of the response to research question number 2.

10.2.3 Detailed investigation

Though the agreed statements collected covered all four software evaluation qualities, the amount of data collected on utility was less than expected. A detailed investigation on external

hosting sites was then conducted. Ten sites were examined and compared based on their backgrounds, features and policies. The classification of infrastructure and non-infrastructure FOSPHost sites was suggested and diversity was likely found in from the analysis of data obtained using this classification. The result of the detailed investigation enriched the response to research question number 2.

10.2.4 Evaluation Model

Based on the data collection from the Delphi survey and the detailed investigation, an evaluation model was constructed. From the nature of the two different data sources, a checklist with qualitative answers was chosen for the presentation of Delphi survey results and a modified weighed scale checklist for detailed investigation results. The evaluation model was implemented as a web site using WakkaWiki (Mans 2003), PHP and MySQL. The evaluation is available at <http://www.ibiblio.org/fosphost/wakka/EvaluateFOSPHost>. This model was built as the response to research question number 3 and subsequently to the overall research question.

10.2.5 Contributions of the Findings

This research is one of the earliest comprehensive investigations on the topic of FOSPHost. It is not uncommon to find research on a specific tool available on FOSPHost sites such as source code repositories (Asklund & Bendix 2002; MacDonald, Hilfinger & Semenzato 1998; Shapiro & Vanderburgh 2002a; Shapiro, Vanderburgh & Lloyd 2003; van der Hoek 2000) or on SourceForge statistics (Crowston & Scozzi 2002; Hunt & Johnson 2002; Kienzle 2001; Krishnamurthy 2002; Lakhani et al. 2003). Nevertheless, a comprehensive study on the topic of FOSPHost is seldom found. The scope of this research ranged from the actual tools provided to the culture and work practices on FOSPHost sites. Views from the administration of the FOSPHost sites as well as issues from the context of usage of the users were included. An exploratory study that covered a similar topic with a scope of similar size was rare. This study likely contributed to the body of knowledge in the area of FOSPHost.

Though there were limitations to this study (which was discussed in the previous chapter), the first contribution of this research is to give the readers and the stakeholders of FOSPHost sites preliminary ideas but with known source and validity on what FOSPHost sites are and how they work. This then may lead to the next contribution to the understanding of Free/Open Source communities and the whole Free/Open Source phenomenon. The final products of this research are also freely and openly available on the Internet for dissemination of these findings.

Other than its results, this research also established sound frameworks such as the model of individual participation to a Free/Open Source community and the software evaluation classification. Smaller but also important classifications such as self-hosting and external hosting FOSPHost sites and infrastructure and non-infrastructure FOSPHost sites were also devised for the analysis of the topic of FOSPHost. All these can be promising tools for the research in the area of Free/Open Source.

10.4 Further Research

In this section, areas for further research will be suggested. The areas for further research in the specific topic of FOSPHost and also the further development of the model of individual participation to a Free/Open Source community will be presented first. Based on the discussion above on the diversity of Free/Open Source communities, potential methodologies and methods will be recommended for further research in the broader context of the Free/Open Source phenomenon.

10.4.1 Further Research on FOSPHost

Within the scope of FOSPHost sites, basic utilities of different external hosting sites were investigated and compared. An overall picture of recommended practices was also obtained from the Delphi survey. Nevertheless, there are more areas that can be further investigated. The detailed investigation on FOSPHost sites was focussed on the functionalities of external

hosting sites. More detailed investigation into individual important tools such as source code repositories and tracker systems can be done. The usability of each tools and the overall FOSPHost site can be evaluated. The results obtained from the Delphi survey did contain opinions on intrinsic and context qualities of FOSPHost sites. Nonetheless, as the emphasis of Delphi survey was more on breadth and less on depth, more in-depth studies can be done on these areas. A number of possible methods of evaluation in these areas are also suggested in the literature on software evaluation in sub-section 5.3.1. The characteristics of infrastructure and non-infrastructure sites can be further investigated using methods such as interviews too. Moreover, details of other possible methodologies and methods will also be discussed below.

Other than external hosting sites, self-hosting sites such as the Mozilla project (Reis & Fortes 2002; The Mozilla Organization 2003b) can be examined. The effect of employing tools hosted on different sites rather than one centralised site can also be explored. Moreover, employment of FOSPHost sites in corporate situations is also an interesting and practical area for research. Some literature on this topic is already available (Derome & Huang 2003; Fink 2003; O'Mahony 2003).

Another possible area for further research is to improve the evaluation model. As discussed above, it could be difficult to present the quantitative data collection in the Delphi survey using the current WakkaWiki interface. This interface can be adjusted by altering the source code of the WakkaWiki.

On the other hand, a more fundamental review can be performed on the evaluation model. Quoting from Breakwell & Millward on the definition of evaluation methods (1995, p. 2):

Evaluation methods are distinguishable from other research methods in terms of their purpose, which is to establish whether specified activities, systems and physical arrangements are effective.

They are used to assess how far certain provisions, practices or procedures (what might be called 'the three Ps') are actually achieving the objectives set for them. Evaluations may, on occasion, go further and attempt to establish why objectives are not achieved by the three Ps.

Therefore, the breath and depth of evaluation can be substantially broadened.

This variety in evaluation methods can also be showed from a preliminary literature review. Breakwell & Millward (1995) also suggested five subjects of evaluation, namely activity, personnel, provision of resources, organisational structure and objectives. From an organizational point of view, evaluation can be classified by the agent who conducts the assessment, namely internal-external, invited-imposed and participatory and non-participatory (Breakwell & Millward 1995). In addition, Owen & Rogers (1999) suggested that different evaluation approaches can be employed at different stages of the execution of a project, namely proactive (before execution), clarificative (during execution), interactive (during execution), monitoring (during execution) and impact (after execution). Wadsworth (1997) also compiled a list of more than 80 different philosophies, models and techniques in evaluation.

The evaluation model developed in this research was based on software evaluation. Though FOSPHost sites are very much software driven, they are also providing a service operated by the administrators of the sites. Extension to the definition of intrinsic was already required to accommodate this service aspect of FOSPHost sites. Some of the approaches and techniques suggested above may be relevant to the evaluation of FOSPHost sites and improvements can be made.

After the discussion of the further research on the topic of FOSPHost, it is also possible to further develop the underlying model used to study FOSPHost - the model of individual

participation to a Free/Open Source community. More knowledge on the details of each aspect of the model is obtained from the data collection processes in this research, and other literature and research since the conception of the model in late 2000. This knowledge can be incorporated into the content of the model with the specification that whether the content is generally applicable to most Free/Open Source communities or just to a particular group of communities in order to preserve the flexibility of the model. Other than aggregating knowledge that is currently available, more research could be conducted to conduct more data especially in the area of contributions and negative results of participation of a FOSPHost site. The structure of the model can be further expanded to include other stakeholders such as user communities, commercial organizations, and the non-commercial organizations that managed Free/Open Source projects (Feller & Fitzgerald 2002). Effects of the community on the intermediate environment and global society (Romm, Pliskin & Clarke 1997) can also be incorporated.

10.4.2 Further Research on the Broader Context of the Free/Open Source Phenomenon

After the discussion of possible areas for further research on FOSPHost, possible directions for research in the broader context of the Free/Open Source phenomenon will be proposed. Relevant methodologies and methods will also be suggested.

From the discussion above, there was considerable diversity in the different practices adopted in the Free/Open Source communities. The metaphor of alternative universes was used to illustrate this situation. Laws in the Free/Open Source communities may exist, but where can one find them? One of the differences mentioned above between conventional software engineering process and Free/Open Source software development process is personal motivations. Therefore, one of the possible points of entry for research is personal factors. Feller & Fitzgerald (2002) created an extensive list of motivations of individuals, organizations

and communities in three levels, namely technological, economic and socio-political context. Lakini et al. (2003) surveyed and categorised four types of motivations - 'learning & simulations', 'hobbyists', 'professionals' (do it for work reasons) and 'community believers'. Hertel, Niedner & Herrmann (2002) surveyed the Linux kernel mailing list and compared the motivations of the participants with general motivation model such as 'Extended Klandermans Model' and found it to be similar. The results of from the Delphi survey also found different purposed and styles among participants. Nevertheless, there are few researches on the process from motivation to participation, the course of participation and the consequential positive and negative results.

For example, if someone was motivated to contribute to the Linux kernel, where would one start? Did one start from subscribing to the Linux kernel mailing list and have one's mailbox flooded with email from the list? (It was quite common to have more than 7000 messages per month (Linux Kernel Mailing List 2003)) Or did he/she just track down issues at kernel traffic (Brown et al. 2003)? Did one overcome the barrier and learn about the norm of the group just by reading about the mailing list, or did one attend a local LUG (Linux User Group) and learn from experienced participants? He or she probably downloaded, changed, compiled and tested the latest version of the kernel. What level of skill was required? How did one come to the understanding of the process of submitting a patch starting from the command diff and patch to the lieutenant organization structure of Linux kernel development? The questions asked above are related to one of the most complicated form of participation in the community – code submission. Other forms of participation, for example, testing and bug submission, may be less involved. Nevertheless, these questions just touch on the surface of what is required to portray a picture of a personal process of participation. More research is probably needed.

One may argue that this kind of investigation neither formed any generalisation, nor proved any causal relationship within the system. Recalling the suggestion by Neuman, Bondy & Knight (2003) data from exploratory and descriptive research were needed before carrying out exploratory research. Without a solid understanding of the background of the topic, the possibility of discovering relationships may decrease and proposing causal relationships between unrelated or partially related variables may increase. For example, Crowston & Scozzi (2002) tried to verify the competency rally theory using the data from SourceForge. With only numeric data and discrete categories (such as alpha, beta, ... , mature) on a set of non-modifiable variable, according to the authors, the validity of the verification was low. (This is not a criticism on the authors' lack of understanding of different factors and underlying process, as the authors were aware of them in the discussion of the limitations of the research. This is only a demonstration of the effect of a limited data set.) To conclude, a better understanding of factors and underlying processes will lay the groundwork for theory building in the research of the Free/Open Source phenomenon. One of possible directions is to investigate personal factors and processes.

Another advantage in studying personal factors that is the understanding of personal factors might assist in the measurement of project success. As argued above, personal success can be an influential factor in project success. Nevertheless, in most Free/Open Source software development research, success is still measured in the traditional ways. For example, in a number of SourceForge statistics analyses (Crowston & Scozzi 2002; Hunt & Johnson 2002; Kienzle 2001; Krishnamurthy 2002), information such as maturity, activity rate and number of downloads were used to define success. In contrast, a research done by Kienzle on variety aspects of Free/Open Source software project success also included a list of personal success/outcomes (Advogato 2002a). This can be another direction for further investigation.

(To do justice to the research mentioned, factors such as maturity were chosen probably due to the availability of the statistics. The researchers may be aware of other factors of success.)

After the discussion of personal factors, the developers participate not in vacuum but against a specific backdrop - the culture and work practices of a particular community. According to Elliott and Scacchi (2003, p. 66), 'the fruition and persistence of rich cultural beliefs and values in the work itself' can be one of the important factors of a successful Free/Open Source online community. The culture and work practices are an important context for the investigation of participation.

After considerations of possible areas for investigation, methodologies and methods of investigation in the Free/Open Source phenomenon can be examined based on previous discussions. More focus will be placed on potential methodologies and methods that can possibly discover rules in alternative universes.

In order to study personal factors and culture, several methodologies can be considered. Phenomenology (Garfinkel 1967; Gubrium & Holestein 2000) is a methodology that deals the study of subjective beliefs and values. Ethnomethodology (Gubrium & Holestein 2000; Schutz 1972), while related to phenomenology, put more emphasis on the interaction between individuals and the discovering social orders of the community. Both methodologies require the researcher to suspend his or her own value system and investigate the phenomenon as it is. If the Free/Open Source phenomenon really contains alternative universes, this practice of suspension of value system may be helpful in situations where social rules are not familiar to the researcher. Only a few studies employed these methodologies (Lawrie, Arief & Gacek 2002; Ratto 2003) and their potential is yet to be explored. Another methodology employed for investigation in related areas was ethnography (Tedlock 2000). An example of the applications

of this methodology was by Scacchi (2002) on the development of requirement of Free/Open Source software systems. And the research by Elliott & Scacchi (2003) mentioned above employed ethnography coupled with grounded theory (Glaser & Strauss 1967; Strauss & Corbin 1990) to study the culture of a Free/Open Source community and interesting results were obtained.

In terms of research methods, protocol analysis (Ericsson & Simon 1993) is probably a method with potential but seldom employed. Referring to studies on the design processes of mechanical engineers, Waldron & Waldron (1996) suggested a number of methods including interviews, protocol analysis (Ericsson & Simon 1993) and case study. One of the lesser-known methods out of the three is protocol analysis and it will be explained briefly below.

'A protocol is defined as a description of the activities (ordered in time) in which a subject engages while performing a task.' (Waldron & Waldron 1996, p. 24). The method of protocol analysis is to collect and analyse the protocols obtained. Several methods are available to collect data on protocols. One of them is the verbal or think-aloud protocols. The designer is required to speak out relevant information about the design during the design process without the researcher's intervention. Another method is called discussion protocols. This method is usually applied in group design by recording the discussion. Another method is called the depositional method where the designer is not required to speak aloud but to explain the design process at convenient intervals. The researcher can also ask questions during the process when the designer forget to explain. The benefit of protocol analysis is the richness of the data obtained (Waldron & Waldron 1996).

Referring to literature, methods such as interviews (Asklund & Bendix 2002; Jorgensen 2001; Kuwabara 2000; Yamauchi et al. 2000) and case studies (Aoki et al. 2001; Kenwood 2001; Mockus, A., Fielding & Herbsleb 2002; Moon & Sproull 2000) were employed in different studies in Free/Open Source. Nonetheless, protocol analysis is seldom used. Obviously, studies of mailing lists and IRC records may coincide with some of the practices of protocol analysis such as discussion protocols. Nevertheless, a complete adoption of this methodology may be worth trying. The advantage of using protocol analysis is that the richness of the data obtained is high and thus the Free/Open Source phenomenon can be studied based on empirical data with fine details that may be crucial to the discovery of unknown laws. One drawback of this methodology can be a high level of cooperation is required with the designer(s) involved. With the future mass adoption of video conferencing equipment, video analysis of design process may be also possible. (The application of protocol analysis to the study of the Free/Open Source phenomenon was actually first suggested by Professor Paula Swatman in Royal Melbourne Institute of Technology in August 2000 together with the Delphi survey method. Delphi survey was adopted consequently in this research while protocol analysis was dropped due to drawbacks suggested by others.)

Scacchi (2003) also suggested that borrowing from biological evolutionary research, the method of taxonomic analyses, phylogentic analyses and software systematics may help in the research of Free/Open Source software. Other analysis techniques may also be relevant. Nevertheless, analysis techniques usually require empirical data, in this case, from the Free/Open Source communities. The methods discussed above such as interviews, case study and protocol analysis are possible choices for obtaining data for such analysis together with code analyses. By employing these methods, it is possible to establish a firm basis for rigid research on the phenomenon of Free/Open Source.

One reason for conducting research in Free/Open Source may be the differences with conventional practices and beliefs in respect of software development, but when we look into the future of Free/Open Source, several benefits may be obtained. The availability of the data provided by the Free/Open Source communities is one of them. Scacchi (2003) suggested that data for Free/Open Source software projects are more readily available than Closed-Source commercial projects as it depend on the willingness of those company to disclose relevant data. Thus, research using Free/Open Source data may prevail in the future. Not only is the availability an advantage, the diversity of these projects is also favourable for research (So, Thomas & Zadeh 2002). A greater diversity in projects may imply more variables on the different aspects of these projects can be analysed and more theories can be built. It is also possible that some fundamental laws in all universes (principles that are applicable for all projects) can be discovered. Laws in another universe can also inform alternate practices in this universe (for example, 'Why Not Improve Coordination in Distributed Software Development by Stealing Good Ideas from Open Source?' by Mockus, Audris & Herbsleb (2002)) and the possibility of mixed universes can also be explored (for example, 'A Framework for Creating Hybrid-OSS Communities' by Sharma, Sugumaran & Rajagopalan (2002)).

In this section, possible directions of research on FOSPHost, personal factors and culture of communities were recommended together with relevant methodologies and methods. The future potential of the research in Free/Open Source was predicted to be positive.

10.5 Possible Future of the Software Industry and the Potential Applications of the Findings

Relevancy of the findings to the current situation of the Free/Open Source phenomenon and areas for further research were presented in the previous sections. In order to fully evaluate the significance of this research and potential of further research, future trends need to be explored. In order to look into the future, a brief history of software will be reviewed first and four future

trends will be introduced. The implications of these trends to the Free/Open Source phenomenon and the potential applications of the findings of this study will then be discussed.

A substantial amount of research was done that led to the invention of computer and the Internet, and a considerable amount of early research work in computers and the Internet was funded by military organizations (Gromov 2002; IEEE Computer Society 2002; The Computer Museum History Center). Therefore, it can be argued that at the beginning computers and the Internet were devised and controlled by organizations with a more centralised structure. There was little distinction of roles in computer related staff – they just had to make it work and almost all of them were technically knowledgeable. Moreover, software was usually written for a particular organization for a specific purpose.

As computer systems became more complex and more people were involved, operators of computer programs were no longer programmers themselves. From the discussion of the origin of usability and HCI in sub-section 5.3.1.2, these topics became important as operators became less and less technical capable. The importance of usability increased even more as personal computer became more popular and some users might know nothing about computers before (Lindgaard 1994). COTS software was an obvious example. This situation of separation of producers and consumers of software was also reflected in the evaluation classification for software proposed in sub-section 5.3.1. There were two categories, intrinsic and context, which were closer related to producers of software and users respectively. Moreover, software companies would usually want to create software that can be used by a group of users with similar needs, so that more copies can be sold. In other words, software was created for a problem domain, not just for solving a particular problem. This is probably a fair description of the current software industry.

After a brief review of the history of software, what will the future of the Free/Open Source phenomenon be? It is possible to extrapolate first by studying general future trends suggested by futuristic writers and then deduces the possible effects that the Free/Open Source phenomenon may have. Four trends will be discussed below, namely prosumption, Internetworking between organizations, globalisation and market segmentation.

Futuristic writers such as Toffler (1981) and Tapscott (1996) suggested that prosumption, the combination of producers and consumers, may be a possible trend. Toffler (1981) explained that before the industrial revolution, most people were farmers and they consumed what they grew. Selling goods from producers to consumers was a relatively infrequent activity. Nevertheless, when the industrial revolution arrived, production was modernised and producers of goods and consumers of goods were usually two separate groups of people. Toffler observed trends in contemporary economy that more products employed models such as self-help, self-service or DIY to give consumers more power in self-determination. Tapscott (1996) also quoted the example from the Chrysler automobile company that customers could made special orders to tailor-made cars (with limitations, of course).

The Free/Open Source phenomenon matched the prosumption concept stated above. Raymond's (2000b) suggested that one of the important motivations of Free/Open Source developers is "scratching one's itch". The availability of source code also allows users to customise the software or even contribute back to the community. In this case, the line between producers and consumers disappears. Therefore, on the one hand, the Free/Open Source phenomenon is not a particularly distinct wonder but it is just another example of the trend of prosumption. On the other hand, unlike automobiles, many types of software can now be created from ground up by personal computers and the extensive distribution of such computers means that most people can become potential producers. (Of course, some specialised software

still requires expensive hardware and expertise.) Therefore, the flexibility that a software consumer can acquire is greater than an automobile consumer and we are just beginning to see the manifestation of such power.

The next trend to be discussed is globalisation (Castells 2000; Tapscott 1996). This means that the interaction between the people in different parts of the world will increase. Miller (2003) suggested that personal computers will be further adopted globally and more people will become computer literate. The supply of programmers from different countries through the Internet will hence increase in quantity as well as in quality in the future. Therefore, more potential contributors will be available in the future for the Free/Open Source communities. From the Delphi survey results, FOSPHost sites enable distributed software development for developers from different geographic locations (Q1.1 & Q6.6). Therefore, as the trend of globalisation continues, the reliance on collaboration tools such as FOSPHost sites will increase.

Another trend that was closely related to globalisation was the Internetworking within and between organizations (Castells 2000; Tapscott 1996). The examples quoted by the two authors were usually business related, but Free/Open Source provided an unexpected collaboration between the commercial and non-commercial worlds. IBM, HP and Sun Microsystems all employed Linux in some of their products. This type of collaboration was new and sometimes conflicts can arise (O'Mahony 2003; Stark 2003). Even businesses that did not deal with Free/Open Source foundations or communities employed Free/Open Source methodology and FOSPHost sites as a paradigm for Internetworking within and between organizations (Derome & Huang 2003). Indeed, from the Delphi survey results, FOSPHost sites could facilitate and enhance communication for commercial and non-commercial producers, consumers and other

stakeholders (Q1.3, Q1.4, Q1.5 & Q6.2), if run properly. Therefore, the study of FOSPHost as a communicative and collaborative tool may be even more significant.

The last trend for discussion is market segmentation. McCraw & Tedlow (1997) suggested a model of the three phases of marketing, namely fragmentation, unification and segmentation. The first stage is fragmentation, which occurs in the early stage of developing a new type of products, such as automobiles or computers. There were many independent producers manufacturing different products with a similar concept. None of them yet becomes well-known or the leader of the market. The volume of production for each of the producers is relatively low and the margins and prices are high. The next stage is unification, where only one player will become the brand of the product. The volume of production of this company is high. Though the margin and price for the product is lower than in the fragmentation stage, the profit can be enormous as the company captured the majority of the market share. In the third stage, segmentation, products were designed to target different needs and aspirations of the perspective consumers. Comparing this model with the development of the software market, Microsoft is probably a sign of the unification stage. Nevertheless, the rise of Free/Open Source operating systems may suggest a transition to the segmentation stage. In fact, though Linux is the most famous Free/Open Source operating system, there are others such as FreeBSD, NetBSD and OpenBSD. There is not just one desktop environment but also GNOME, GNUStep, KDE, Xfce and others. Segmentation can be based on functionalities. For example, OpenBSD is focused on security. Nevertheless, it can be observed that sometimes segmentation is due to ideology, such as KDE and GNOME. From the discussion of diversity in the sub-section 9.3, developers may come from different backgrounds, with different personal factors and needs. Such differences may be manifested in different work practices and culture in the different development community. A variety of software may thus be developed.

Intrinsic qualities of software will then be the important factors for differentiating between segments.

If there is a transition of the market from unification to segmentation, consumers may find the situation chaotic as their choices diversifies. The evaluation of software will thus become an even more important method to distinguish the similarities and differences of software. The software evaluation classification proposed in this research could possibly provide a more comprehensive model that is suitable to the evaluation of Free/Open Source software. When an organization acquires a piece of software, it can be the beginning of a relationship with the Free/Open Source community that produces it. As mentioned in the discussion Internetworking between organization above, there could be conflicts between commercial and Free/Open Source communities (O'Mahony 2003; Stark 2003). One possible cause of conflicts is the mismatch between the culture and work practices of the Free/Open Source communities (intrinsic) and the commercial organization (context). As these factors are already accounted for in the software evaluation classification proposed, it has a good potential to produce useful analysis on Free/Open Source software.

There are some comments to the futuristic extrapolation above. One comment is that the discussion above is in no way comprehensive. Relevant trends such as molecularization and disintermediation (Tapscott 1996) are not mentioned. Only the most relevant trends are elaborated due to the limitations of this dissertation.

Another comment is that all the trends suggested are in favour of the growth and expansion of the Free/Open Source phenomenon. It will then be beneficial to present some trends that do not encourage this growth. A current threat at the moment when this document is composed is the legal battle between SCO and IBM on the intellectual property of Linux (The SCO Group 2003).

Many were not confident of SCO winning, but even if SCO loses the case, the topic of intellectual property will still probably be a threat until the legal system catches up with this new frontier of the Internet (Barlow 1993). Nonetheless, a more possible threat from the inside was suggested by Raymond (2003) - the elitist attitude of Free/Open Source communities. This attitude may have the potential to keep the culture as a minority.

To conclude, a number of general future trends are in favour of the growth and expansion of the Free/Open source phenomenon and FOSPHost will probably remain an important topic. The software evaluation classification developed in this research also may provide a useful basis for software analysis in the segmented software market.

List of References

- Acheson, S 2001, *The Secure Shell(TM) Frequently Asked Questions*, viewed 10 Nov 2003, <<http://www.employees.org/~satch/ssh/faq/ssh-faq.html>>.
- Advogato 2000a, *Fallible Hacker Figureheads*, viewed 1 Feb 2002, <<http://www.advogato.org/article/123.html>>.
- Advogato 2000b, *Version Control Systems: The Next Generation*, viewed 27 Oct 2000, <<http://www.advogato.org/article/145.html>>.
- Advogato 2000c, *Ask the Advogatos: why do Free Software projects fail?*, viewed 19 Oct 2000, <<http://www.advogato.org/article/128.html>>.
- Advogato 2001a, *Leaving SourceForge*, viewed 23 Apr 2002, <<http://www.advogato.org/article/357.html>>.
- Advogato 2001b, *On Holy Wars and a Plea for Peace #2*, viewed 1 Feb 2002, <<http://www.advogato.org/article/396.html>>.
- Advogato 2001c, *CVS mixed-tagging for massive Open Source Project Management*, viewed 14 Mar 2003, <<http://www.advogato.org/article/247.html>>.
- Advogato 2002a, *Project Success - Measuring it/Facilitating it*, viewed 3 Feb 2003, <<http://www.advogato.org/article/441.html>>.
- Advogato 2002b, *CVS Considered Harmful*, viewed 14 Mar 2003, <<http://www.advogato.org/article/519.html>>.
- Advogato 2003, *Advogato*, viewed 24 Feb 2003, <<http://www.advogato.org/>>.
- Alan 2001, *ITS: The Incompatible Time Sharing System*, viewed 12 Feb 2003, <<http://www.its.os.org/>>.
- Alexander, JE & Tate, MA 1999, *Web wisdom : how to evaluate and create information quality on the Web*, Lawrence Erlbaum Associates, Mahwah, N.J.
- Aoki, A, Hayashi, K, Kishida, K, Nakakoji, K, Nishinaka, Y, Reeves, B, Takashima, A & Yamamoto, Y 2001, 'A case study of the evolution of Jun: An object-oriented open-source 3D multimedia library', paper presented to Proceedings of the 23rd International Conference on Software Engineering, Toronto, Ontario, Canada, May 2001.
- Apple Computer Inc. 2002, *Darwin - Open Source*, viewed 31 Jan 2003, <<http://developer.apple.com/darwin/>>.
- Arief, B, Bosio, D, Gacek, C & Rouncefield, M 2002, *Dependability Issues in Open Source Software*, Department of Computing Science, University of Newcastle, viewed 9 Dec 2002, <<http://www.cs.ncl.ac.uk/research/trs/papers/760.pdf>>.
- Arjen van Efferen & Black, BW 2002, *XoopsForge Official Website*, viewed 31 Oct 2003, <<http://xoopsforge.sourceforge.net/modules/news/>>.
- Asklund, U & Bendix, L 2002, 'A Study of Configuration Management in Open Source Software Projects', *IEE Proceedings - Software*, vol. 149, no. 1, pp. 40-6.

- asynchrony.com 2001, *Asynchrony: Where Great Ideas Meet Their Potential*, <<http://www.asynchrony.com/>>.
- asynchrony.com 2004, *Asynchrony has been closed down*, viewed 20 Feb 2004, <<http://www.asynchrony.com/>>.
- Australian Society of Certified Practising Accountants' Information Technology Centre of Excellence 1995, *Commercial accounting software selection principles*, Australian Society of Certified Practising Accountants, Melbourne, Vic.
- Babbie, ER 2002, *The basics of social research*, 2nd edn, Wadsworth Thomson Learning, Belmont, CA.
- Barlow, JP 1993, *The Economy of Ideas*, viewed 18 Mar 2002, <<http://www.eff.org/~barlow/EconomyOfIdeas.html>>.
- Barr, D *David Barr*, viewed 5 Nov 2003, <<http://www.visi.com/~barr/>>.
- Barr, J 2002, *Linus tries to make himself scale*, viewed 5 Dec 2002, <http://www.linuxworld.com/site-stories/2002/0211.scale_p.html>.
- Bate, R, Kuhn, D, Wells, C, Armitage, J, Clark, G, Cusick, K, Garcia, S, Hanna, M, Jones, R, Malpass, P, Minnich, I, Pierson, H, Powell, T & Reichner, A 1995, *A Systems Engineering Capability Maturity Model, Version 1.1*, viewed 15 Sep 2003, <<http://www.sei.cmu.edu/pub/documents/95.reports/pdf/mm003.95.pdf>>.
- Beck, K 1999, *Extreme Programming Explained: Embrace Change*, Addison-Wesley, Reading, Massachusetts.
- Ben-Menachem, M 1994, *Software configuration management guidebook*, McGraw-Hill International Software Quality Assurance Series., McGraw Hill Book Co., London ; New York.
- Bentson, R 2000, *The Proper Image for Linux*, viewed 29 Dec 2000, <<http://www2.linuxjournal.com/lj-issues/issue57/2931.html>>.
- BerliOS *BerliOS Developer: Welcome*, viewed 9 Apr 2003, <<http://developer.berlios.de/>>.
- Bevan, N 1995, 'Usability is Quality of Use', paper presented to Proceedings of the 6th International Conference on Human Computer Interaction, Yokohama, viewed 11 Feb 2003, <<http://www.usability.serco.com/papers/hcistd95.pdf>>.
- Bezroukov, N 1999a, 'Open Source Software Development as a Special Type of Academic Research (Critique of Vulgar Raymondism)', *First Monday*, vol. 4, no. 10, viewed 2 Jun 2000, <http://firstmonday.org/issues/issue4_10/bezroukov/index.html>.
- Bezroukov, N 1999b, 'A Second Look at the Cathedral and Bazaar', *First Monday*, vol. 4, no. 12, viewed 2 Jun 2000, <http://firstmonday.org/issues/issue4_12/bezroukov/index.html>.
- Bezroukov, N 2000, *4.1. Linus and Linux; Linus Torvalds' Short Unauthorized Biography*, viewed 3 Nov 2000, <http://www.softpanorama.org/People/Torvalds/Linus_Torvalds_biography.shtml>.
- Bezroukov, N 2002, *California Sunrise (1997-2000): Linux Hype Festival and IPO Gold Rush*, viewed 6 Dec 2002, <http://www.softpanorama.org/People/Torvalds/linus_california_sunrise.shtml>.

- BitMover 2002, *Feature summary*, viewed 14 Mar 2003, <http://www.bitkeeper.com/Products.BK_Pro.Feature.html>.
- Blaikie, N 1993, *Approaches to Social Enquiry*, Polity Press, Cambridge, England.
- Blease, D 1986, 'Educational Software Selection Criteria', in D Squires & AM 1994 (eds), *Choosing and using educational software : a teachers' guide*, Falmer Press, London ; Washington, D.C, pp. 135-9.
- Bolinger, D & Bronson, T 1995, *Applying RCS and SCCS*, O'Reilly, Sebastopol, California.
- Bosch, S, Promis, P & Sugnet, C 1994, *Guide to selecting and acquiring CD-ROMS, software and other electronic publications*, Acquisitions guidelines ; no. 9, American Library Association, Chicago.
- Breakwell, GM & Millward, L 1995, *Basic evaluation methods : analysing performance, practice and procedure*, Personal and professional development, British Psychological Society, Leicester.
- Britannica.com 2000, *Positivism*, viewed 6 Feb 2000, <<http://www.britannica.com/bcom/eb/article/5/0,5716,62575+1+61024,00.html?query=positivism>>.
- Brooks, FP 1995, *The mythical man-month : essays on software engineering*, Anniversary edn, Addison-Wesley Pub. Co., Reading, Mass.
- Brown, Z, Sullivan, P, Vincent, B, Pouech, E, Emsley, P, Seigo, AJ, Samuelson, P, Flagg, C, Harford, A, Appel, J, Kriukovas, A, Waugh, J, Quirk, J, Mundkur, P, Miller, R, Robbins, S, Law, S, Phillips, S, Buchbinder, A, Kaper, R, Rusin, Z, Muller, J, Rufian-Zilbermann, EC, Butler, TR, Cohn, S, Kohler, B, Eckersley, P, Martinez, D, Michlmayr, M, Miramon, Cd, Guthrie, J, Howells, C, O'Sullivan, R, Team, KP, Jensen, L, Mous, F, Team, KT & Zealey, M 2003, *Kernel Traffic*, viewed 15 Jul 2003, <<http://kt.zork.net/>>.
- Buckley, FJ 1996, *Implementing configuration management : hardware, software, and firmware*, 2nd edn, IEEE Computer Society Press ; IEEE Press, Los Alamitos, Calif.; New York.
- Bull, G & Garofalo, J 2003, 'Rationale for Building an Educational Source Forge', *Learning & Leading with Technology*, vol. 30, no. 8, May 2003, pp. 18-21.
- Carpenter, K 2001, *Opening New Doors - Sun Microsystems and CollabNet Make NetBeans Platform an Open Source Success*, viewed 11 Sep 2001, <<http://www.softwarebusinessonline.com/articles/aug01-2.htm>>.
- Castells, M 2000, *The rise of the network society*, 2nd edn, Blackwell Publishers, Oxford.
- Catella, C & Exploris Museum 1999, *World Wide Web Page Evaluation Form*, viewed 9 Dec 1999, <<http://www.ncsu.edu/midlink/WWW.eval.html>>.
- Checkland, P 1981, *Systems Thinking, Systems Practice*, Wiley, Chichester, Sussex; New York.
- CIDOC Multimedia Working Group 1997, *Multimedia Evaluation Criteria*, viewed 15 Sep 1999, <<http://www.archimuse.com/cidoc/cidoc.mmwg.eval.crit.html>>.
- Ciolek, TM 1997, *Information Quality - Catalogue of Potent Truisms*, viewed 27 Jul 1999, <<http://www.ciolek.com/WWWVLPages/QltyPages/QltyTruisms.html>>.

- Clarke, R 1999, 'The Willingness of Net-Consumers to Pay: A Lack-of-Progress Report', paper presented to Proc. 12th International Bled EC Conf., Slovenia, June, viewed 22 Apr 2000, <<http://www.anu.edu.au/people/Roger.Clarke/EC/WillPay.html>>.
- Collab.Net 2002a, *Tigris.org: Open source software engineering*, viewed 9 Apr 2003, <<http://www.tigris.org/>>.
- Collab.Net 2002b, *subversion.tigris.org*, viewed 9 Apr 2003, <<http://subversion.tigris.org/>>.
- Collab.Net 2003a, *Customers*, viewed 9 Apr 2003, <<http://www.collab.net/customers/index.html>>.
- Collab.Net 2003b, *CollabNet SourceCast Guided Tour*, Collab.Net, viewed 11 Jul 2003, <http://www.collab.net/downloads/media/pdfs/sourcecast_guided_tour.pdf>.
- Collab.Net 2003c, *CMM Impact Analysis - The CollabNet SourceCast Environment and Collaborative Development*, viewed 4 Aug 2003, <http://www.collab.net/downloads/media/pdfs/cmm_impact_analysis.pdf>.
- Collab.Net 2003d, *Alliances*, viewed 28 Oct 2003, <<http://www.collab.net/about/alliances/ta.html>>.
- Confucius 500 BC, *The Analects*, trans. Muller C, viewed 20 Feb 2004, <<http://www.human.toyogakuen-u.ac.jp/~acmuller/contao/analects.htm>>.
- Cooper, A 1999, *The Inmates are Running the Asylum: Why High-Tech Products Drive us Crazy and How to Restore the Sanity*, SAMS, Indianapolis, Indiana.
- Copeland, T 2003, *GForge Sites*, viewed 31 Oct 2003, <<http://gforge.org/docman/view.php/1/52/gforge-sites.html>>.
- Couger, JD 1988, 'Motivators vs. Demotivators in the IS Environment', *Journal of Systems Management*, vol. 39, no. 6, pp. 36-41.
- Crowston, K & Scozzi, B 2002, 'Open Source Software Projects as Virtual Organizations: Competency Rallying for Software Development', *IEE Proceedings - Software*, vol. 149, no. 1, pp. 3-17.
- Curtis, B, Hefley, WE & Miller, S 1995, *Overview of the People Capability Maturity Model*, viewed 15 Sep 2003, <<http://www.sei.cmu.edu/pub/documents/95.reports/pdf/mm001.95.pdf>>.
- Dachary, L 2001, *SourceForge drifting*, viewed 21 Nov 2001, <<http://www.fsfeurope.org/news/article2001-10-20-01.en.html>>.
- Dafermos, GN 2001, 'Management and Virtual Decentralised Networks: The Linux Project', *First Monday*, vol. 6, no. 11, viewed 28 Jan 2003, <http://firstmonday.org/issues/issue6_11/dafermos/index.html>.
- Delbecq, AL, Van de Ven, AH & Gustafson, DH 1975, *Group techniques for program planning : a guide to nominal group and Delphi processes*, Scott Foresman, Glenview, Illinois.
- Dempsey, BJ, Weiss, D, Jones, P & Greenberg, J 2002, 'Who is an Open Source Software Developer? Profiling a community of Linux Developers', *Communications of ACM*, vol. 45, no. 2, pp. 67-72.

- Derome, J & Huang, K 2003, *Creating and Delivering Value with Collaborative Software Development Tools*, viewed 28 Oct 2003, <http://www.collab.net/downloads/media/pdfs/creating_and_delivering_value.pdf>.
- DeTienne, KB, Smart, KL & Jones, BD 1995, 'Motivating your IS staff', *Journal of Systems Management*, vol. 46, no. 2, pp. 40-4.
- DiBona, C, Ockman, S & Stone, M 1999, 'Appendix A: The Tanenbaum-Torvalds Debate', in C DiBona, S Ockman & M Stone (eds), *Open Sources: Voices from the Open Source Revolution*, O'Reilly & Associates, Sebastopol, California, viewed 7 Nov 2000, <<http://www.oreilly.com/catalog/opensources/book/appa.html>>.
- Dix, A 1994, 'Computer Supported Cooperative Work: A Framework', in D Rosenberg & C Hutchison (eds), *Design Issues in CSCW*, Springer-Verlag, London, pp. 9-26.
- drupal.org *drupal.org community plumbing*, viewed 14 Nov 2003, <<http://drupal.org/>>.
- Dumas, JS & Redish, J 1999, *A practical guide to usability testing*, Rev. edn, Intellect Books, Exeter, England; Portland, Oregon.
- eclipse.org *eclipse.org*, viewed 20 Nov 2003, <<http://www.eclipse.org/>>.
- Elfanbaum, D 2001, *A Slightly Rambling White Paper and Personal Rant on the Dawn of TransCapitalism and the WeCcosystem Vision*, viewed 30 Oct 2003, <<https://www.asynchrony.com/vision.jsp>>.
- Elliott, MS & Scacchi, W 2003, *Free Software: A Case Study of Software Development in a Virtual Organizational Culture*, viewed 9 Jul 2003, <<http://opensource.mit.edu/papers/elliottscacchi.pdf>>.
- Enzer, S 1975, 'Plastics and Competing Materials by 1985: A Delphi Forecasting Study', in HA Linstone & M Turoff (eds), *The Delphi Method: Techniques and Applications*, Addison-Wesley, London, pp. 195-209.
- Ericsson, KA & Simon, HA 1993, *Protocol Analysis*, MIT Press, Cambridge, Massachusetts.
- Eunice, J 1998a, *Beyond the Cathedral, Beyond the Bazaar*, viewed 19 Oct 2001, <<http://www.illuminata.com/public/content/cathedral/intro.htm>>.
- Eunice, J 1998b, *Beyond the Cathedral, Beyond the Bazaar - The Key is Community*, viewed 19 Oct 2001, <<http://www.illuminata.com/public/content/cathedral/cathedral5.htm>>.
- Everhart, N (ed.) 1996, *Web Page Evaluation Worksheet*, viewed 15 Sep 1999, <<http://www.duke.edu/~de1/evaluate.html>>.
- Feller, J & Fitzgerald, B 2002, *Understanding Open Source Software Development*, Addison Wesley, London.
- Fenton, NE & Pfleeger, SL 1997, *Software metrics : a rigorous and practical approach*, 2 edn, PWS, London.
- Fink, M 2003, *The Business and Economics of Linux and Open Source*, Prentice Hall PTR, Upper Saddle River, New Jersey.
- Fogel 1999, *Open Source Development With CVS*, Coriolis, Scottsdale, Arizona.
- Forge, S 2000, 'Open Source: The Economics of Giving Away Stuff, and Software as a Political Statement', *Info*, vol. 2, no. 1, February, pp. 5-7.

- Foster, E 1998, *1997 Product of the Year: Best Technical Support Award: Linux User Community*, InfoWorld, viewed 27 Nov 2000, <<http://www.infoworld.com/cgi-bin/displayTC.pl?/97poy.suppl.htm>>.
- Fraser, D & Goodacre, CF 1993, *Selecting library management software : evaluative criteria for circulation and cataloguing functions*, Centre for Information Studies Charles Sturt University-Riverina, Wagga Wagga, New South Wales.
- Free Software Foundation 2000, *What is Free Software?*, viewed 21 Jun 2000, <<http://www.fsf.org/philosophy/free-sw.html>>.
- Free Software Foundation 2002, *Why "Free Software" is better than "Open Source"*, viewed 30 Jan 2003, <<http://www.gnu.org/philosophy/free-software-for-freedom.html>>.
- Free Software Foundation 2003a, *Mailman, the GNU Mailing List Manager*, viewed 28 Mar 2003, <<http://www.gnu.org/software/mailman/index.html>>.
- Free Software Foundation 2003b, *Savannah: Welcome*, viewed 7 Apr 2003, <<http://savannah.gnu.org/>>.
- Free Software Foundation 2003c, *Emacs - Extensible, real-time editor*, viewed 31 Oct 2003, <<http://www.gnu.org/directory/text/editors/emacs.html>>.
- Free Software Foundation 2003d, *GNU's Not Unix! - the GNU Project and the Free Software Foundation (FSF)*, viewed 31 Oct 2003, <<http://www.gnu.org/>>.
- Freedman, A (ed.) 1998, *The Computer Glossary: The Complete Illustrated Dictionary*, 8th edn, American Management Association, New York.
- Fuller, MD 2004, *BSD For Linux Users :: Philosophies*, viewed 23 Apr 2004, <<http://www.over-yonder.net/~fullermd/rants/bsd4linux/bsd4linux8.php>>.
- Gabriel, RP & Goldman, R 2002, 'Open Source: Beyond the Fairy Tales', *Center for Business Innovation Journal*, no. 8, pp. 59-65, viewed 27 Aug 2003, <http://www.cbi.cgey.com/journal/Issue8/Open_Source.pdf>.
- Gacek, C, Lawrie, AT & Arief, LB 2001, *The many meanings of Open Source*, CS-TR-737, Department of Computing Science, University of Newcastle upon Tyne, viewed 31 Oct 2001, <<http://www.cs.ncl.ac.uk/people/l.b.arief/home.formal/Papers/TR737.pdf>>.
- Gallivan, MJ 2001, 'Striking a Balance between Trust and Control in a Virtual Organization: A Content Analysis of Open Source Software Case Studies', *Information Systems Journal*, vol. 11, no. 4, pp. 277-304.
- Garfinkel, H 1967, *Studies in ethnomethodology*, Prentice-Hall, Englewood Cliffs, New Jersey.
- GBorg development team 2003, *GBorg -- The Development website*, viewed 30 Oct 2003, <<http://gborg.postgresql.org/>>.
- Ghosh, RA 1998a, 'Cooking pot markets: an economic model for the trade in free goods and services on the Internet', *First Monday*, vol. 3, no. 3, viewed 2 Jun 2000, <http://www.firstmonday.dk/issues/issue3_3/ghosh/index.html>.
- Ghosh, RA 1998b, 'FM Interview with Linux Torvalds: What motivates free software developers', *First Monday*, vol. 3, no. 3, viewed 1 Aug 2000, <http://www.firstmonday.org/issues/issue3_3/torvalds/index.html>.

- Ghosh, RA 2002, 'FLOSS Workshop report : Advancing the Research Agenda on Free / Open Source Software', paper presented to Workshop on Advancing the Research Agenda on Free / Open Source Software, Brussels, 14 October 2002, viewed 24 Jan 2003, <<http://www.infonomics.nl/FLOSS/report/workshopreport.htm>>.
- Glaser, BG & Strauss, AL 1967, *The discovery of grounded theory : strategies for qualitative research*, Aldine, New York.
- Goldschmidt, P 1996, 'A Comprehensive Study of the Ethical, Legal and Social Implications of Advances in Biomedical and Behavioural Research and Technology', in M Adler & E Ziglio (eds), *Gazing into the Oracle: The Delphi Method and Its Application to Social Policy and Public Health*, Jessica Kingsley Publishers, London, pp. 89-132.
- Goldstein, NH 1975, 'A Delphi on the Future of the Steel and Ferroalloy Industries', in HA Linstone & M Turoff (eds), *The Delphi Method: Techniques and Applications*, Addison-Wesley, London, pp. 210-26.
- Gordon, RC 2003, *icculus.org -- The Helping Phriendly Box*, viewed 9 Apr 2003, <<http://icculus.org/>>.
- Grassian, E 1998, *Thinking Critically about World Wide Web Resources*, viewed 27 Jul 1999, <<http://www.library.ucla.edu/libraries/college/instruct/web/critical.htm>>.
- Great Circle Associates 2001, *Majordomo*, viewed 5 Nov 2003, <<http://web.archive.org/web/20020823114958/http://www.greatcircle.com/majordomo/>>.
- Gromov, bGR 2002, *History of Internet and WWW: The Roads and Crossroads of Internet History*, viewed 12 Sep 2003, <<http://www.netvalley.com/netvalley/intval.html>>.
- Grudin, J 1992, 'Utility and usability: research issues and development contexts', *Interacting with Computers*, vol. 4, no. 2, pp. 209-17.
- Gubrium, JF & Holestein, JA 2000, 'Analyzing Interpretive Practice', in NK Denzin, YS Lincoln & H Fehring (eds), *The handbook of qualitative research*, 2nd edn, Sage Publications, Thousand Oaks, California, pp. 487-508.
- Hacker, JQ 1999, *Feature: Conflicting Open Source Developers*, viewed 26 Nov 2000, <<http://slashdot.org/features/99/07/12/1639202.shtml>>.
- Hamerly, J, Paquin, T & Walton, S 1999, 'Freeing the Source: The Story of Mozilla', in C DiBona, S Ockman & M Stone (eds), *Open Sources: Voices from the Open Source Revolution*, O'Reilly & Associates, Sebastopol, California, viewed 7 Nov 2000, <<http://www.oreilly.com/catalog/opensources/book/netrev.html>>.
- Harris, R 1997, 'Evaluating Internet Research Sources', viewed 15 Sep 1999, <http://www.sccu.edu/faculty/R_Harris/evalu8it.htm>.
- Hars, A & Ou, S 2001, 'Working for free? - Motivations of participating in Open Source Projects', paper presented to The 34th Hawaii International Conference on System Sciences.
- Hauben, M & Hauben, R 1997, *The Netizens and the Wonderful World of the Net: An Anthology*, viewed 22 May 2000, <<http://studentweb.tulane.edu/~rwoods/netbook/contents.html>>.
- Helmer, O 1975, 'TOC & Preface', in HA Linstone & M Turoff (eds), *The Delphi Method: Techniques and Applications*, Addison-Wesley, London, pp. iv-xx.

- Henderson, JR 1999, *The IYouSee Guide to Critical Thinking About What You See on the Web*, viewed 15 Sep 1999, <<http://www.ithaca.edu/library/Training/hott.html>>.
- Hertel, G, Niedner, S & Herrmann, S 2002, 'Motivation of Software Developers in Open Source Projects: An Internet-based Survey of Contributors to the Linux Kernel', *Research Policy, special issue on Open Source Software Development*, vol. 32, no. 7, pp. 1159-77.
- Herzberg, F, Mausner, B & Snyderman, BB 1959, *The motivation to work*, 2nd edn, Wiley ; Chapman & Hall, New York.
- Hinchliffe, LJ 1997, *Evaluation of Information*, viewed 19 Nov 1999, <<http://alexia.lis.uiuc.edu/~janicke/Eval.html>>.
- Hofstede, GH 1997, *Cultures and Organizations*, McGraw-Hill, Berkshire, England.
- Hood, E 2003, *MHonArc Home Page*, viewed 5 Nov 2003, <<http://www.mhonarc.org/>>.
- Hornby, AS & Crowther, J 1995, *Oxford advanced learner's dictionary of current English*, 5 edn, Oxford University Press, Oxford.
- Huang, K, Lee, YW & Wang, RY 1999, *Quality Information and Knowledge*, Prentice-Hall, New Jersey.
- Humphrey, WS 1997, *Managing technical people : innovation, teamwork, and the software process*, SEI series in software engineering., Addison-Wesley, Reading, Massachusetts.
- Humphrey, WS 2000, *The Personal Software Process (PSP)*, viewed 15 Sep 2003, <<http://www.sei.cmu.edu/pub/documents/00.reports/pdf/00tr022.pdf>>.
- Hunt, F & Johnson, P 2002, 'On the Pareto distribution of open source projects', paper presented to Proceedings of the Open Source Software Development Workshop, Newcastle upon Tyne, U.K., 25-26 Feb. 2002, viewed 25 Mar 2002, <<http://www.dirc.org.uk/events/ossdw/OSSDW-Proceedings-Final.pdf>>.
- ibiblio.org 2003a, *ibiblio - April is Great*, viewed 9 Apr 2003, <<http://www.ibiblio.org/>>.
- ibiblio.org 2003b, *About ibiblio*, viewed 27 Nov 2003, <<http://www.ibiblio.org/about.html>>.
- IBM 2003, *developerWorks: Linux*, viewed 31 Jan 2003, <<http://www.ibm.com/developerworks/linux/>>.
- IEEE Computer Society 2002, *Events in the History of Computing - World War II*, viewed 12 Sep 2003, <<http://www.computer.org/history/development/wwii.htm>>.
- jacobito 2001, *Gnome Hackers Sorting Out Differences RE:2.0*, viewed 6 Feb 2003, <<http://slashdot.org/article.pl?sid=01/06/19/1223245&mode=thread>>.
- Jacobs, J 1993, *Systems of survival : a dialogue on the moral foundations of commerce and politics*, Hodder & Stoughton, London.
- Jacobson, T & Cohen, L 1996, *Evaluating Internet Resources*, viewed 15 Sep 1999, <<http://www.albany.edu/library/internet/evaluate.html>>.
- Jillson, IA 1975, 'The National Drug-abuse Policy Delphi', in HA Linstone & M Turoff (eds), *The Delphi Method: Techniques and Applications*, Addison-Wesley, London, pp. 124-59.
- Johnson, K 1999, *Open-Source Software Development*, viewed 22 Dec 2000, <<http://www.cpsc.ucalgary.ca/~johnsonk/SENG/SENG691/open.htm>>.

- Jones, CG 1975, 'A Delphi Evaluation of Agreement Between Organizations', in HA Linstone & M Turoff (eds), *The Delphi Method: Techniques and Applications*, Addison-Wesley, London, pp. 160-7.
- Jones, CM 1998, *Evaluation of Effective Instructional Web Sites: A Pilot Study*, viewed 4 Nov 1999, <<http://ccwf.cc.utexas.edu/~jonesc/research/evaluation.htm>>.
- Jones, P 2000, *Brooks' Law and open source: The more the merrier? Does the open source development method defy the adage about cooks in the kitchen?*, viewed 1 Jun 2000, <<http://www-4.ibm.com/software/developer/library/merrier.html>>.
- Jorgensen, N 2001, 'Putting it all in the trunk: incremental software development in the FreeBSD open source project', *Information Systems Journal*, vol. 11, no. 4, pp. 321-36.
- Kelty, CM 2000, *Scale and Convention: Programmed Languages in a Regulated America*, Mass. Institute of Technology, Ph.D Dissertation.
- Kelty, CM 2001, 'Hau to do things with words', in *Knowledge and Society*, JAI Press, vol. 13, viewed 29 Jan 2002, <<http://www.kelty.org/or/papers/hauto.kelty.pdf>>.
- Kenwood, C 2001, *A Business Case Study of Open Source Software*, viewed 3 Jun 2002, <http://www.mitre.org/support/papers/tech_papers_01/kenwood_software/kenwood_software.pdf>.
- Kernel.Org *The Linux Kernel Archives*, viewed 30 Oct 2003, <<http://www.kernel.org/>>.
- Kernighan, BW & Ritchie, DM 1988, *The C programming language*, 2nd edn, Prentice Hall, Englewood Cliffs, N.J.
- Kienzle, R 2001, *Sourceforge Preliminary Project Analysis*, viewed 23 Jan 2002, <<http://www.osstrategy.com/sfreport/>>.
- Kirk, EE 1999, *Evaluating Information Found on the Internet*, viewed 15 Sep 1999, <<http://milton.mse.jhu.edu:8001/research/education/net.html>>.
- Koch, S & Schneider, G 2002, 'Effort, Cooperation and Coordination in an Open Source Software Project: GNOME', *Information Systems Journal*, vol. 12, no. 1, pp. 27-42.
- Kollock, P 1996, 'Design Principles for Online Communities', paper presented to Harvard Conference on the Internet and Society, viewed 16 Jun 2000, <<http://www.sscnet.ucla.edu/soc/faculty/kollock/papers/design.htm>>.
- Kollock, P 1999, 'The Economies of Online Cooperation: Gifts and Public Goods in Cyberspace', in MAK Smith, Peter (ed.), *Communities in Cyberspace*, Routledge, London, pp. 220-42.
- Konig, T 1997, *Ssh (Secure Shell) FAQ - Frequently asked questions*, viewed 7 Nov 2003, <<http://www.rz.uni-karlsruhe.de/~ig25/ssh-faq/index.html>>.
- Krishnamurthy, S 2002, 'Cave or Community?: An Empirical Examination of 100 Mature Open Source Projects', *First Monday*, vol. 7, no. 6, viewed 5 Feb 2003, <http://firstmonday.org/issues/issue7_6/krishnamurthy/>.
- Kuwabara, K 2000, 'Linux: A Bazaar at the Edge of Chaos', *First Monday*, vol. 5, no. 3, viewed 31 Jul 2000, <http://firstmonday.org/issues/issue5_3/kuwabara/index.html>.

- Kuykendall, D 2001, *phpGroupWare is leaving SourceForge*, viewed 8 Feb 2002, <<http://mail.gnu.org/pipermail/phpgroupware-developers/2001-November/000002.html>>.
- Lakhani, KR & Hippel, Ev 2003, 'How open source software works: "free" use-to-use assistance', *Research Policy*, vol. In Press, Corrected Proof, Available online 27 November 2002, viewed 15 Jan 2003, <<http://www.sciencedirect.com/science/article/B6V77-479TM54-1/1/1206f45569b80b7bc5c0b0981c437493>>.
- Lakhani, KR, Wolf, B & Bates, J 2002, *The Boston Consulting Group Hacker Survey*, viewed 6 Feb 2002, <<http://www.osdn.com/bcg/BCGHACKERSURVEY.pdf>>.
- Lakhani, KR, Wolf, B, Bates, J & DiBona, C 2003, *The Boston Consulting Group Hacker Survey - Release 0.73*, viewed 15 Jan 2003, <<http://www.osdn.com/bcg/BCGHACKERSURVEY-0.73.pdf>>.
- Lancashire, D 2001, 'The Fading Altruism of Open Source Development', *First Monday*, vol. 6, no. 12, viewed 20 Jan 2003, <http://www.firstmonday.org/issues/issue6_12/lancashire/index.html>.
- Lang, T 'An Overview of Four Futures Methodologies', *The Manoa Journal of Fried and Half-Fried Ideas*, vol. 7, viewed 14 Feb 2003, <<http://www.soc.hawaii.edu/future/j7/LANG.html>>.
- Lawrie, T, Arief, B & Gacek, C 2002, 'Interdisciplinary Insights on Open Source', paper presented to Proceedings of the Open Source Software Development Workshop, Newcastle upon Tyne, U.K., 25-26 Feb. 2002, viewed 25 Mar 2002, <<http://www.dirc.org.uk/events/ossdw/OSSDW-Proceedings-Final.pdf>>.
- Le Cornu, P 1996, *Hardware/software selection and purchasing*, TAFE national information technology series, ITE402, Eastern House, Victoria, Australia.
- Lee, AS 1991, 'Integrating Positivist and Interpretive Approaches to Organizational Research', *Organization Science*, vol. 4, no. 2, pp. 342-65.
- Lee, SD 2002, *Building an electronic resource collection : a practical guide*, Library Association, London.
- Lerner, J & Triole, J 2002, 'Some Simple Economics of Open Source', *The Journal of Industrial Economics*, vol. 50, no. 2, pp. 197-234.
- Leuf, B & Cunningham, W 2001, *The Wiki Way: Collaboration and Sharing on the Internet*, Addison-Wesley.
- Levy, S 1984, *Hackers: Heroes of The Computer Revolution*, Anchor Press/Doubleday, Garden City, New York.
- Lewis, RO 1992, *Independent verification and validation : a life cycle engineering process for quality software*, New dimensions in engineering, Wiley, New York.
- Licklider, JCR & Taylor, R 1968, 'The Computer as a Communication Device', *In Memoriam: J.C.R. Licklider 1915-1990*, reprinted by permission from Digital Research Center; originally published as 'The Computer as a Communication Device,' in *Science and Technology*, April, 1968, pg. 40, p. 40.

- Lindberg, F & Ringel, FB 1999, *EZMLM/IDX MAILING LIST MANAGER*, viewed 28 Mar 2003, <<http://ezmlm.org/>>.
- Lindgaard, G 1994, *Usability testing and system evaluation : a guide for designing useful computer systems*, Chapman & Hall, London.
- Linstone, HA & Murray, T 1975, 'Introduction', in HA Linstone & M Turoff (eds), *The Delphi Method: Techniques and Applications*, Addison-Wesley, London, pp. 3-12.
- Linux Kernel Mailing List 2003, *MARC: Mailing list ARChives at AIMS*, viewed 15 Jul 2003, <<http://marc.theaimsgroup.com/?l=linux-kernel>>.
- Lord, T *The arch Low-Budget Home Page*, viewed 9 Apr 2003, <<http://regexps.srparish.net/www/>>.
- Ludlow, J 1975, 'Delphi Inquiries and Knowledge Utilization', in HA Linstone & M Turoff (eds), *The Delphi Method: Techniques and Applications*, Addison-Wesley, London, pp. 102-23.
- MacDonald, J, Hilfinger, PN & Semenzato, L 1998, 'PRCS: The Project Revision Control System', paper presented to System Configuration Management ECOOP'98 SCM-8 Symposium, Belgium, July, viewed 17 Mar 2003, <<http://prdownloads.sourceforge.net/prcs/scm98.pdf>>.
- Maclachlan, M 1999, *Panelists Describe Open Source Dictatorships*, viewed 26 Nov 2000, <<http://www.techweb.com/news/story/TWB19990812S0003>>.
- Mans, HM 2003, *WakkaWiki : WakkaWiki*, viewed 10 Dec 2003, <<http://www.wakkawiki.com/WakkaWiki>>.
- Markus, ML, Manville, B & Agres, CE 2000, 'What Makes a Virtual Organization Work?' *Sloan Management Review*, vol. 42, no. 1 (Fall), pp. 13-26.
- McCabe, TJ 1976, 'A Complexity Measure', *IEEE Transactions on Software Engineering*, vol. 2, pp. 308-20.
- McCraw, T & Tedlow, R 1997, 'Henry Ford, Alfred Sloan, and the Three Phases of Marketing', in T McCraw (ed.), *Creating Modern Capitalism : How Entrepreneurs, Companies, and Countries Triumphed in Three Industrial Revolutions*, Harvard, Cambridge, pp. 266-308.
- McIntyre Library 1998, *Ten C's For Evaluating Internet Sources*, viewed 15 Sep 1999, <<http://www.uwec.edu/Admin/Library/Guides/tencs.html>>.
- McKendrick, J 2003, *Is Open Source the New Normal?*, viewed 21 Feb 2003, <<http://www.entmag.com/news/article.asp?EditorialsID=5704>>.
- McKusick, MK 1999, 'Twenty Years of Berkeley Unix From AT&T-Owned to Freely Redistributable', in C DiBona, S Ockman & M Stone (eds), *Open Sources: Voices from the Open Source Revolution*, O'Reilly & Associates, Sebastopol, California, viewed 7 Nov 2000, <<http://www.oreilly.com/catalog/opensources/book/kirkmck.html>>.
- McLuhan, M 1964, *Understanding media : the extensions of man*, Routledge & K. Paul, London.
- mettw 2000, *Contribution balance*, viewed 19 Oct 2000, <<http://www.advogato.org/article/128.html>>.
- Miller, P 2003, *Aegis 4.11*, viewed 9 Apr 2003, <<http://aegis.sourceforge.net/>>.

- Miller, RR 2002, *Counting desktop Linux users is impossible*, viewed 20 Sep 2002, <<http://newsforge.com/newsforge/02/09/17/0111258.shtml?tid=19>>.
- Miller, RR 2003, *The programmer as (starving) artist*, viewed 9 Dec 2003, <<http://www.newsforge.com/article.pl?sid=03/12/02/1256211>>.
- Minnihan, J 2003, *freepository - Enabling Global Software Development Collaboration*, viewed 30 Oct 2003, <<https://www.freepository.com/>>.
- Mitroff, II & Turoff, M 1975, 'Philosophy and Methodological Foundations of Delphi', in HA Linstone & M Turoff (eds), *The Delphi Method: Techniques and Applications*, Addison-Wesley, London, pp. 15-36.
- Mockus, A & Herbsleb, JD 2002, 'Why Not Improve Coordination in Distributed Software Development by Stealing Good Ideas from Open Source?' paper presented to 2nd Workshop on Open Source Software Engineering, Proceedings of 23rd Int'l Conf. on Software Engineering, Orlando, Florida, May 25, viewed 23 Aug 2002, <<http://opensource.ucc.ie/icse2002/MockusHerbsleb.pdf>>.
- Mockus, A, Fielding, R & Herbsleb, J 2002, *Two Case Studies of Open Source Software Development: Apache and Mozilla*, ALR-2002-003, Avaya Labs, viewed 9 Oct 2002, <<http://www.research.avayalabs.com/techreport/ALR-2002-003-paper.pdf>>.
- Moen, R 2002, *GForge: possible renaissance for open-source SourceForge*, viewed 22 Jul 2003, <<http://lwn.net/Articles/17369/>>.
- Moody, G 2001, *Rebel Code: The Inside Story of Linux and the Open Source Revolution*, Perseus, Cambridge, Massachusetts.
- Moon, JY & Sproull, L 2000, 'Essence of Distributed Work: The Case of the Linux Kernel', *First Monday*, vol. 5, no. 11, viewed 4 Dec 2002, <http://firstmonday.org/issues/issue5_11/moon/index.html>.
- MySQL AB 2003, *MySQL The World's Most Popular Open Source Database*, viewed 7 Nov 2003, <<http://www.mysql.com/>>.
- Nakakoji, K, Yamamoto, Y, Nishinaka, Y, Kishida, K & Yunwen, Y 2002, 'Evolution Patterns of Open-Source Software Systems and Communities', paper presented to Proceedings of 5th International Workshop on Principles of Software Evolution (IWPSE2002), Orlando, Florida, May 2002, viewed 24 Jan 2003, <<http://www.kid.rcast.u-tokyo.ac.jp/~kumiyo/mypapers/IWPSE2002.pdf>>.
- Nelson, R, Cole, S, Mueller, O & Theodoropoulos, P 2003, *The qmail home page*, viewed 5 Nov 2003, <<http://qmail.glasswings.com.au/top.html>>.
- Netcraft 2004, *Netcraft Web Server Survey*, viewed 9 Jan 2004, <http://news.netcraft.com/archives/2004/01/01/january_2004_web_server_survey.html>.
- Netscape Communications Corporation 1998, *Introduction to SSL*, viewed 7 Nov 2003, <<http://developer.netscape.com/docs/manuals/security/sslin/contents.htm>>.
- Neuman, WL, Bondy, J & Knight, N 2003, *Social research methods : qualitative and quantitative approaches*, 5th edn, Allyn and Bacon, Boston.

- Newman, N 1999, *The Origins and Future of Open Source Software: A NetAction White Paper*, viewed 28 Jul 2000, <<http://www.netaction.org/opensrc/future/oss-whole.html>>.
- Nielsen, J 1993, *Usability engineering*, Academic Press, Boston.
- Nielsen, J 2000, *Designing Web Usability: The Practice of Simplicity*, New Riders, Indianapolis, Indiana.
- Nikhil Goel, Christian Bayle, Adriano Nagelschmidt Rodrigues, Peter Masiar & Douglas, B 2003, *Discussion Forums: Devel Forum*, viewed 6 Nov 2003, <http://gforge.org/forum/forum.php?thread_id=708&forum_id=5>.
- Novell 2003, *Welcome to Novell Forge*, viewed 30 Oct 2003, <<http://forge.novell.com/modules/news/>>.
- O'Brien, JA 1997, *Introduction to information systems*, 8th edn, Irwin Book Team, Chicago.
- O'Mahony, S 2003, 'Non-Profit Foundations and their Role in Community-Firm Software Collaboration', paper presented to HBS - MIT Sloan Free/Open Source Software Conference: New Models of Software Development, Boston and Cambridge, MA., 19-20 Jun, 2003, viewed 6 Oct 2003, <<http://opensource.mit.edu/papers/conf-omahony.pdf>>.
- Open Source Initiative 2000, *History of the Open Source Initiative*, viewed 18 Jun 2000, <<http://www.opensource.org/history.html>>.
- Open Source Initiative 2003a, *The Open Source Definition*, viewed 30 Jan 2003, <<http://www.opensource.org/docs/definition.php>>.
- Open Source Initiative 2003b, *Open Source Initiative OSI - Welcome*, viewed 30 Jan 2003, <<http://www.opensource.org/>>.
- Ormondroyd, J, Engle, M & Cosgrave, T 1999, *How to Critically Analyze Information Sources*, viewed 15 Sep 1999, <<http://www.library.cornell.edu/okuref/research/skill26.htm>>.
- OSDir.com 2002, *Interview with OSDir: Interview with Tim Perdue. GForge and behind the scenes at SourceForge*, viewed 22 Jul 2003, <<http://osdir.com/modules.php?op=modload&name=News&file=article&sid=102>>.
- OSDN 2003a, *Slashdot: News for nerds, stuff that matters*, viewed 24 Feb 2003, <<http://slashdot.org/>>.
- OSDN 2003b, *freshmeat.net*, viewed 24 Feb 2003, <<http://freshmeat.net/>>.
- Owen, JM & Rogers, PJ 1999, *Program evaluation : forms and approaches*, 2nd edn, Allen & Unwin, St Leonards, N.S.W.
- Pavlicek, RC 2000, *Embracing Insanity: Open Source Software Development*, Sams, Indiana.
- Pennington, H *Working on Free Software*, viewed 25 Nov 2000, <<http://www106.pair.com/rhp/hacking.html>>.
- Pesch, R 2002, *CVS--Concurrent Versions System v1.11.2*, viewed 29 Jan 2003, <<http://ftp.cvshome.org/cvs-1.11.2/cederqvist-1.11.2.html.tar.gz>>.
- Powell, DE 2002, *KDEvelopers on KDE users*, viewed 5 Jul 2002, <<http://www.linuxandmain.com/modules.php?name=News&file=article&sid=128>>.
- Pressman, RS 1997, *Software engineering : a practitioner's approach*, 4th edn, McGraw-Hill, New York.

- Purdue University Libraries 1999, *Evaluation of Information Sources*, viewed 12 Dec 1999, <http://www.lib.purdue.edu/library_info/departments/ugrl/ref/bib/evalinfo.html>.
- Queensland University of Technology Division of Information and Academic Services 2003, *A Copyright Guide for Students*, viewed 22 Jan 2004, <http://www.tils.qut.edu.au/copyright/QUT_Student_Copyright_Guide.html>.
- Ratto, M 2003, 'The Power of Openness: the hybrid work of Linux Free/Open Source kernel developers', PhD thesis, University of California.
- Raymond, ES 1999, 'The Revenge of the Hackers', in C DiBona, S Ockman & M Stone (eds), *Open Sources: Voices from the Open Source Revolution*, O'Reilly & Associates, Sebastopol, California, viewed 7 Nov 2000, <<http://www.oreilly.com/catalog/opensources/book/raymond2.html>>.
- Raymond, ES 2000a, *Homesteading the Noosphere*, viewed 22 Dec 2000, <<http://www.tuxedo.org/~esr/writings/homesteading/homesteading/>>.
- Raymond, ES 2000b, *The Cathedral and the Bazaar*, viewed 30 May 2000, <<http://www.tuxedo.org/~esr/writings/cathedral-bazaar/cathedral-bazaar.html>>.
- Raymond, ES 2000c, *A Brief History of Hackerdom*, viewed 30 May 2000, <<http://www.tuxedo.org/~esr/writings/hacker-history/hacker-history.html>>.
- Raymond, ES 2001, *The Jargon File*, viewed 28 Aug 2001, <<http://www.tuxedo.org/~esr/jargon/jargon.html>>.
- Raymond, ES 2003, 'Problems in the culture of Unix', in *The Art of Unix Programming*, viewed 8 Apr 2003, <<http://www.catb.org/~esr/writings/taoup/html/ch19s04.html>>.
- Reis, C 2002a, *The Free Software Engineering Survey*, viewed 9 Apr 2003, <<http://www.async.com.br/~kiko/fsp/results.php>>.
- Reis, C 2002b, *The Free Software Engineering Survey - (not really) Frequently Asked Questions*, viewed 6 Jun 2003, <<http://www.async.com.br/~kiko/fsp/faq.php>>.
- Reis, C & Fortes, RPdM 2002, 'An Overview of the Software Engineering Process and Tools in the Mozilla Project', paper presented to Proceedings of the Open Source Software Development Workshop, Newcastle upon Tyne, U.K., 25-26 Feb. 2002, viewed 25 Mar 2002, <<http://www.dirc.org.uk/events/ossdw/OSSDW-Proceedings-Final.pdf>>.
- Rheingold, H 1993, *The Virtual Community: Homesteading on the Electronic Frontier*, Addison-Wesley, Reading, Massachusetts.
- Robbins, J 2002, *Personal Communication (during a visit to Collab.Net)*, 21 Feb 2002.
- Roland Mas, Christian Bayle & Kwon, S-S *debian-sf Homepage*, viewed 31 Oct 2003, <<http://www.nongnu.org/debian-sf/>>.
- Romm, C, Pliskin, N & Clarke, R 1997, 'Virtual Communities and Society: Toward an Integrative Three Phase Model', *International Journal of Information Management*, vol. 17, no. 4, pp. 261-70.
- Rosenberg, DH, C. 1994, 'Introduction', in DH Rosenberg, C. (ed.), *Design Issues in CSCW*, Springer-Verlag, London, pp. 1-8.

- Rossi, PH, Freeman, HE & Rosenbaum, S 1982, *Evaluation : a systematic approach*, 2nd edn, Sage Publications, Beverly Hills, California.
- Royal Melbourne Institute of Technology 2004, *Fair Dealing*, viewed 22 Jan 2004, <<http://www.rmit.edu.au/browse:ID=6xaviaznzwnf>>.
- Royce, WW 1970, 'Managing the Development of Large Software Systems: Concepts and Techniques', paper presented to Proc. WESCON, Los Angeles, Aug. 25-28.
- Rubin, J 1994, *Handbook of usability testing : how to plan, design, and conduct effective tests*, Wiley technical communication library, Wiley, New York.
- Salus, PH 1995, *A Quarter Century of Unix*, Addison-Wesley, Reading, Massachusetts.
- Sarapuu, T & Adojaan, K 1998, 'Evaluation Scale of Educational Web Sites', paper presented to Internet and Intranet Proceedings, Orlando, FL, Nov 7-12.
- Scacchi, W 2002, 'Understanding Requirements for Developing Open Source Software Systems', *IEE Proceedings - Software*, vol. 149, no. 1, pp. 24-39.
- Scacchi, W 2003, *Understanding Open Source Software Evolution: Applying, Breaking, and Rethinking the Laws of Software Evolution*, viewed 9 Jul 2003, <<http://opensource.mit.edu/papers/scacchi3.pdf>>.
- Scarlett, B 2001, 'Entrprise Effectiveness: A Cross-Cultural Study of Business Goals', PhD thesis, Royal Melbourne Institute of Technology.
- Schach, SR, Jin, B, Wright, DR, Heller, GZ & Offutt, AJ 2002, 'Maintainability of the Linux Kernel', *IEE Proceedings - Software*, vol. 149, no. 1, pp. 18-23.
- Schmidt, RC 1997, 'Managing Delphi surveys using nonparametric statistical techniques', *Decision Sciences*, vol. 28, no. 3, pp. 763-74.
- Schrock, K 1999, *Kathy Schrock's Guide for Educators - Critical Evaluation Surveys*, viewed 15 Sep 1999, <<http://school.discovery.com/schrockguide/eval.html>>.
- Schulmeyer, GG & MacKenzie, GR 2000, *Verification and validation of modern software-intensive systems*, Prentice Hall, Upper Saddle River, NJ.
- Schutz, A 1972, *The phenomenology of the social world*, Heinemann Educational, London.
- Schweik, CM & Semenov, A 2003, 'The Institutional Design of Open Source Programming: Implications for Addressing Complex Public Policy and Management Problems', *First Monday*, vol. 8, no. 1, viewed 10 Jan 2003, <http://www.firstmonday.org/issues/issue8_1/schweik/index.html>.
- SEUL.org 2001, *SEUL Manifesto*, viewed 31 Oct 2003, <<http://www.seul.org/what/manifesto.html>>.
- SEUL.org 2002, *Seul.org Home Page*, viewed 9 Apr 2003, <<http://www.seul.org/>>.
- SGI 2003, *SGI - SGI Global Developer Program SGI Open Source*, viewed 31 Jan 2003, <<http://www.sgi.com/developers/oss/>>.
- Shapiro, JS 2002, *OpenCM User's Guide*, viewed 26 Mar 2003, <<http://www.opencm.org/opencm.html>>.
- Shapiro, JS & Vanderburgh, J 2002a, 'CPCMS: A Configuration Management System Based on Cryptographic Names', paper presented to Proc. 2002 USENIX Annual Technical Conference,

- FreeNIX Track, Monterey, CA, viewed 24 Feb 2003, <<http://www.opencm.org/papers/cpcms2001.pdf>>.
- Shapiro, JS & Vanderburgh, J 2002b, 'Access and Integrity Control in a Public-Access, High-Assurance Configuration Management System', paper presented to Proc. 11th USENIX Security Symposium, San Francisco, CA, viewed 20 Mar 2003, <<http://www.opencm.org/papers/usenix-sec2002.pdf>>.
- Shapiro, JS, Vanderburgh, J & Lloyd, J 2003, 'OpenCM One Year Later', paper presented to Proc. 2003 USENIX Annual Technical Conference, FreeNIX Track, viewed 20 Mar 2003, <<http://www.opencm.org/papers/usenix-sec2002.pdf>>.
- Sharma, S, Sugumaran, V & Rajagopalan, B 2002, 'A Framework for Creating Hybrid-OSS Communities', *Information Systems Journal*, vol. 12, no. 1, pp. 7-26.
- Smith, AG 1997, *Criteria for Evaluation of Internet Resources*, viewed 27 Jul 1999, <<http://www.vuw.ac.nz/~agsmith/evaln/index.htm>>.
- Smith, AG 1998, 'Criteria for Evaluation of Internet Resources in a Digital Library Environment', paper presented to Proceedings Of the First Asia Digital Library Workshop, Hong Kong, 6-7 Aug.
- So, H, Thomas, N & Zadeh, H 2002, 'What is in a Bazaar? A Model of Individual Participation in an Open Source Community', paper presented to Proceedings of the Open Source Software Development Workshop, Newcastle upon Tyne, U.K., 25-26 Feb. 2002, viewed 25 Mar 2002, <<http://www.dirc.org.uk/events/ossdw/OSSDW-Proceedings-Final.pdf>>.
- Software Engineering Institute 2003a, *Capability Maturity Models*, viewed 2 Oct 2003, <<http://www.sei.cmu.edu/cmm/cmms/cmms.html>>.
- Software Engineering Institute 2003b, *Capability Maturity Model (SW-CMM) for Software*, viewed 9 Oct 2003, <<http://www.sei.cmu.edu/cmm/cmm.sum.html>>.
- Software Engineering Standards Committee of the IEEE Computer Society 1998a, *IEEE Standard for Software Verification and Validation, IEEE Std 1012-1998*, Institute of Electrical and Electronics Engineers, New York.
- Software Engineering Standards Committee of the IEEE Computer Society 1998b, *IEEE Recommended Practice for Software Acquisition, IEEE Std 1062-1998*, Institute of Electrical and Electronics Engineers, New York.
- SourceForge 2003, *SourceForge.net: Welcome*, <<http://sourceforge.net/>>.
- SourceForge 2004, *SourceForge.net: Welcome*, viewed 9 Jan 2004, <<http://sourceforge.net/>>.
- Sowa, J & Zachman, J 1992, 'Extending and formalizaing the framework for IS architecture', *IBM Systems Journal*, vol. 31, no. 3, pp. 590-616.
- Squires, D & McDougall, A 1994, *Choosing and using educational software : a teachers' guide*, Falmer Press, London ; Washington, D.C.
- Squires, D & McDougall, A 1996, 'Software evaluation: a situated approach', *Journal of Computer Assisted Learning*, vol. 12, no. 3, pp. 146-61.
- Stamelos, I, Angelis, L, Oikonomou, A & Bleris, GL 2002, 'Code Quality Analysis in Open-Source Software Development', *Information Systems Journal*, vol. 12, no. 1, pp. 43-60.

- Stanford University 2003, *Copyright & Fair Use*, viewed 22 Jan 2004, <<http://fairuse.stanford.edu/>>.
- Stark, M 2003, *The Organizational Model for Open Source*, viewed 9 Jul 2003, <<http://workingknowledge.hbs.edu/pubitem.jhtml?id=3582&t=technology>>.
- Strauss, AL & Corbin, JM 1990, *Basics of qualitative research : grounded theory procedures and techniques*, Sage, Newbury Park, California.
- Strong, DM, Lee, YW & Wany, RY 1997, 'Data Quality in Context', *Communications of the ACM*, vol. 40, no. 5, pp. 103-10.
- Sun Microsystems Inc. *Welcome to SunSource.Net - an information and links site edited by the Sun Open Source Program Office*, viewed 31 Jan 2003, <<http://www.sunsource.net/>>.
- Sunsite.dk staff group 2003a, *SunSITE.dk - Supporting Open Source*, viewed Apr 9 2003, <<http://sunsite.dk/>>.
- Sunsite.dk staff group 2003b, *About the SunSITE Program*, viewed 31 Oct 2003, <<http://sunsite.dk/node/id/66>>.
- Sunsite.dk staff group 2003c, *About sunsite.dk*, viewed 31 Oct 2003, <<http://sunsite.dk/node/id/67>>.
- Susan, B 1997, *Evaluation Criteria, The Good, The Bad & The Ugly: or, Why It's a Good Idea to Evaluate Web Sources*, viewed 15 Sep 1999, <<http://lib.nmsu.edu/staff/susabeck/evalcrit.html>>.
- Tapscott, D 1996, *The Digital Economy: Promise and Peril in the Age of Networked Intelligence*, McGraw-Hill, New York.
- Tedlock, B 2000, 'Ethnography and Ethnographic Representation', in NK Denzin, YS Lincoln & H Fehring (eds), *The handbook of qualitative research*, 2nd edn, Sage Publications, Thousand Oaks, California, pp. 455-86.
- The Associated Press 2000, *Security firm tests FBI limits with e-mail surveillance tool*, viewed 31 Jan 2003, <<http://www.cnn.com/2000/TECH/computing/09/19/email.surveillance.ap/>>.
- The Computer Museum History Center *CHM: Timeline of Computer History*, viewed 12 Sep 2003, <http://www.computerhistory.org/timeline/timeline.php?timeline_year=1970>.
- The Free On-line Dictionary of Computing 1998, *client-server*, viewed 13 Mar 2003, <<http://wombat.doc.ic.ac.uk/foldoc/foldoc.cgi?query=client-server&action=Search>>.
- The FreeBSD Project 2003, *FreeBSD CVSweb Project*, viewed 5 Nov 2003, <<http://www.freebsd.org/projects/cvsweb.html>>.
- The Mozilla Organization 2003a, *bonsai*, viewed 26 Mar 2003, <<http://www.mozilla.org/bonsai.html>>.
- The Mozilla Organization 2003b, *mozilla*, viewed 18 Dec 2003, <<http://www.mozilla.org/>>.
- The PHP Group 2003, *PHP Hypertext Preprocessor*, viewed 7 Nov 2003, <<http://www.php.net/>>.
- The PostgreSQL Global Development Group *PostgreSQL*, viewed 17 Nov 2003, <<http://www.postgresql.org/>>.

- The Samba Team 2003, *Welcome to the rsync web pages*, viewed 7 Nov 2003, <<http://samba.anu.edu.au/rsync/index.html>>.
- The SCO Group 2003, *Letter To Linux Customers*, viewed 16 May 2003, <http://www.sco.com/scosource/letter_to_linux_customers.html>.
- The Unix vs NT Organisation 2001, *The Unix vs NT Organisation*, viewed 14 Sep 2001, <<http://www.unix-vs-nt.org/>>.
- Tichy, WF 1985, 'RCS-A System for Version Control', *Software - Practice and Experience*, vol. 15, no. 7, pp. 637-54, viewed 22 Nov 2002, <<http://citeseer.nj.nec.com/rd/0%2C111753%2C1%2C0.25%2CDownload/http://citeseer.nj.nec.com/cache/papers/cs/5588/http:zSzzSzwww.cs.yorku.cazSz%7EbrechtzSz4321zSzhandoutszSzrcs.pdf/tichy91rcs.pdf>>.
- Tim Perdue, Christian Bayle, Roland Mas, Daniele Franceschi, Tony Guntharp, Michael Jennings, Reinhard Spisser & Copeland, T *GForge3: Welcome*, viewed 30 Oct 2003, <<http://gforge.org/>>.
- Toffler, A 1981, *The third wave*, Pan Books, London.
- Torvalds, L & Diamond, D 2001, *Just for Fun: The Story of an Accidental Revolutionary*, TEXERE, New York.
- Travis, D 2003, *E-commerce usability : tools and techniques to perfect the on-line experience*, Taylor & Francis, New York.
- Tridgell, A & Shearer, D *JitterBug project suspended*, viewed 5 Nov 2003, <<http://samba.anu.edu.au/jitterbug/>>.
- Tuomi, I 2001, 'Internet, Innovation, and Open Source: Actors in the Network', *First Monday*, vol. 6, no. 1, viewed 24 Jan 2003, <http://www.firstmonday.org/issues/issue6_1/tuomi/index.html>.
- Turkle, S 1984, *The Second Self: Computers and the Human Spirit*, Simon & Schuster, New York.
- Turoff, M & Hiltz, SR 1996, 'Computer Based Delphi Processes', in M Adler & E Ziglio (eds), *Gazing into the Oracle: The Delphi Method and Its Application to Social Policy and Public Health*, Jessica Kingsley Publishers, London, pp. 56-85.
- Twining, J 1999, 'Electronic Delphi Among Collaboratory Pioneers', in *A Naturalistic Journey into the Collaboratory: In Search of Understanding for Prospective Participants*, PhD Dissertation, Chapter 9, pp. 174-212, viewed 28 Aug 2000, <<http://intertwining.org/dissertation/dissertation.PDF>>.
- VA Software 2003, *Alliances*, viewed 28 Oct 2003, <<http://www.vasoftware.com/company/alliances.php>>.
- van der Hoek, A 2000, 'Configuration Management and Open Source Projects', paper presented to 22nd Intl' Conf. On Software Engineering, 3rd Workshop on Software Engineering over the Internet, 6 Jun. 2000, viewed 4 Oct 2000, <<http://sern.ucalgary.ca/~maurer/icse2000ws/submissions/Hoek.pdf>>.

- ViewCVS Users Group 2002, *ViewCVS: Viewing CVS Repositories*, viewed 26 Mar 2003, <<http://viewcvs.sourceforge.net/>>.
- Wadsworth, Y 1997, *Everyday evaluation on the run*, 2nd edn, Allen & Unwin, St. Leonards, New South Wales.
- Waldron, MB & Waldron, KJ 1996, 'Methods of Studying Mechanical Design', in MB Waldron & KJ Waldron (eds), *Mechanical Design*, Springer, New York, pp. 21-34.
- Watt, JH 1999, 'Internet Systems for Evaluation Research', in G Gay & T Bennington (eds), *Information Technologies in Evaluation: Social Moral, Epistemological, the Practical Implications, New Directions for Evaluation*, Jossey-Bass, San Francisco, CA, pp. 23-43.
- Weinberg, GM 1971, *The psychology of computer programming*, Computer science series, Van Nostrand Reinhold, New York.
- Wellman, B & Gulia, M 1999, 'Net Surfers Don't Ride Alone: Virtual Communities as Communities', in B Wellman (ed.), *Networks in the global village : life in contemporary communities*, Westview Press, Boulder, Colorado, pp. 331-66.
- Wheeler, DA 2002, *More Than a Gigabuck: Estimating GNU/Linux's Size*, viewed 5 Feb 2003, <<http://www.dwheeler.com/sloc/redhat71-v1/redhat71sloc.html>>.
- White, BA 2000, *Software configuration management strategies and Rational ClearCase : a practical introduction*, Addison-Wesley object technology series, Addison-Wesley, Boston, MA.
- Whitehead, R 2001, *Leading a software development team : a developer's guide to successfully leading people and projects*, Addison-Wesley, Harlow.
- Wiki Engines, 2003, viewed 1 Dec 2003, <<http://c2.com/cgi/wiki?WikiEngines>>.
- Wilcox, J & Shankland, S 2002, *Analysts: Microsoft feels tug of Linux*, viewed 20 Feb 2003, <<http://news.com.com/2102-1001-976755.html>>.
- Wilkinson, GL, Bennett, LT & Oliver, KM 1997, *Evaluating the Quality of Internet Information Sources*, viewed 19 Nov 1999, <<http://itech1.coe.uga.edu/Faculty/GWilkinson/webeval.html>>.
- Willard, A & Irwin, BE 2005, *Loads of Linux Links: Software Archives*, viewed 30 Aug 2005, <http://loll.sourceforge.net/linux/links/Software_Archives/>.
- Wilms, R 2003, *An SSL connection method for CVS (sserver)*, viewed 20 Nov 2003, <http://home.arcor.de/rolf_wilms/cvsssl/cvsssl_help.html>.
- Wilson, G 1999, 'Is the Open-Source Community Setting a Bad Example?' *IEEE Software*, vol. 16, no. 1, pp. 23-5.
- Wire, B 2000, *VA Linux Launches 'SourceForge OnSite' Enterprise Solution*, viewed 6 Feb 2001, <http://www.businesswire.com/cgi-bin/f_headline.cgi?bw.120500/203400154&ticker=LNUX>.
- Wladawsky-Berger, I 2001, *Wladawsky-Berger on Linux and open standards*, viewed 28 Aug 2001, <<http://www.ibm.com/news/us/2001/08/15.html>>.

- Wu, M-W & Lin, Y-D 2001, 'Open Source Software Development: An Overview', *IEEE Computer*, June 2001, pp. 33-8, viewed 14 Jul 2003, <<http://speed.cis.nctu.edu.tw/~ydlin/insideopen.pdf>>.
- Yamauchi, Y, Yokozawa, M, Shinohara, T & Ishida, T 2000, 'Collaboration with Lean Media: How Open-Source Software Succeeds', paper presented to ACM Conference on Computer Supported Cooperative Work (CSCW2000), Philadelphia, PA, viewed 18 Jun 2001, <http://www.bol.ucla.edu/~yutaka/papers/yamauchi_cscw2000.pdf>.
- Yee, D 1999, *Development, Ethical Trading, and Free Software*, viewed 11 Sep 2000, <<http://danny.oz.au/freedom/ip/aidfs.html>>.
- Yin, RK 1994, *Case study research : design and methods*, 2nd edn, Applied social research methods series ; v. 5, Sage, Thousand Oaks, California.
- Zachman, J 1987, 'A framework for IS architecture', *IBM Systems Journal*, vol. 26, no. 3, pp. 276-92.
- Zawinski, J 1999, *resignation and postmortem*, viewed 22 Jun 2000, <<http://www.jwz.org/gruntle/nomo.html>>.
- Ziglio, E 1996, 'The Delphi Method and its Contribution to Decision-Making', in M Adler & E Ziglio (eds), *Gazing into the Oracle: The Delphi Method and Its Application to Social Policy and Public Health*, Jessica Kingsley Publishers, London, pp. 3-33.

Appendix A Related Publications

Two related articles were published during the course of the PhD programme. They are listed below.

1. So, H, Thomas, N & Zadeh, H 2002, 'What is in a Bazaar? A Model of Individual Participation in an Open Source Community', paper presented to Proceedings of the Open Source Software Development Workshop, Newcastle upon Tyne, U.K., 25-26 Feb. 2002, viewed 25 Mar 2002, <<http://www.dirc.org.uk/events/ossdw/OSSDW-Proceedings-Final.pdf>>.
2. So, H., Thomas, N. & Zadeh, H. 2002, 'The Keys to Succeed in Building a Free/Open Source Community for Software Development : a study on China, Hong Kong and Taiwan', International Conference on Open Source, Taipei, 2-4 Aug. 2002.

Appendix B Licenses of Different Portions of the Dissertation and Other Copyright Issues

As a substantial part of the data was collected from Free/Open Source communities that have a strong culture of sharing, it is appropriate to make this research available to the public as well under a license that encourage sharing. The general principle for licensing of this dissertation is that body will be licensed under the Creative Commons Attribution-NonCommercial-NoDerivs v2.5 License (CC) and the source code written in this research is licensed under the General Public License v2.0 (GPL).

The author is not an expert in licenses and CC and GPL were chosen partly because they were commonly used. Creative Common is a well-known organization for providing flexible protection on copyright and GPL was the most commonly used license for source code. These licenses are enclosed below. This dissertation was formerly released under OpenContent License v1.0 (OPL), which is now obsoleted by CC.

There are some minor issues worth mentioning. The content of the comparison table of the external hosting FOSPHost sites was released into public domain as well as licensed under CC as the content of this dissertation so that more flexibility was given to the usage of the content.

Another issue is that permission may be required to include the quoted the diagrams and tables in this dissertation for digital dissemination under CC. Though these diagrams and tables quoted from other sources can be included in the physical copy of the dissertation based on the principle of fair use (Stanford University 2003) or fair dealing (Queensland University of Technology Division of Information and Academic Services 2003; Royal Melbourne Institute of Technology 2004), the rules for digital dissemination are different and the author will ask for permission after the examination of this dissertation. The author would like to ask for the

co-operation of the readers not to distribute the digital format of this dissertation on the CD-ROM. The author will make a digital copy of this dissertation with the authorised diagrams and tables only available to the public on the Internet later.

Another copyright issue is trademark acknowledgement. A list of trademarks appeared in the dissertation is provided below.

Apache	Trademark of the Apache Software Foundation
Bugzilla	Trademark of the Mozilla Organization
Capability Maturity Models (CMM)	Registered trademark of Software Engineering Institute
Chrysler	DaimlerChrysler
CollabNet	Registered trademarks of CollabNet, Inc.
FreeBSD	Registered trademark of Wind River Systems, Inc.
GNOME	Trademark of the GNOME Foundation
Java	Trademark of Sun Microsystems, Inc.
Javascript	Trademark of Netscape Communications Corporation
JavaServer Pages	Trademark of Sun Microsystems, Inc.
JavaServer Pages (JSP)	Trademark of Sun Microsystems, Inc.
Linux	Trademark of Linus Torvalds
Microsoft	Trademark of Microsoft Corporation
Microsoft Windows	Trademark of Microsoft Corporation
Mozilla	Trademark of the Mozilla Organization
MySQL	MySQL AB
NetBeans	Registered trademark of NetBeans Corporation
NetBSD	Trademark of the NetBSD Foundation
Netscape	Trademark of Netscape Communications Corporation
Novell	Trademark of Novell, Inc.
Personal Software Process (PSP)	Service Mark of Software Engineering Institute
Portable Document Format (PDF)	Trademark of Adobe Corporation
PostgreSQL	Trademark of the PostgreSQL Global Development Group
SCO	Trademark of the SCO Group

Secure Sockets Layer (SSL)	Trademark of Netscape Communications Corporation
SourceCast	Registered trademarks of CollabNet, Inc.
SourceForge	Trademark of VA Software
Sun	Trademark of Sun Microsystems, Inc.
Sun Microsystems	Trademark of Sun Microsystems, Inc.
Sun Site	Trademark of Sun Microsystems, Inc.
Team Software Process (TSP)	Service Mark of Software Engineering Institute
UNIX	Registered trademarks of X/Open Company Limited
VA Software	Trademark of VA Software

Table B-1 List of Trademarks Acknowledged

All other trademarks and copyrights referred to in this dissertation are the property of their respective owners.



creative commons
COMMONS DEED

Attribution-NonCommercial-NoDerivs 2.5

You are free:

- ♦ to copy, distribute, display, and perform the work

Under the following conditions:

-  **Attribution.** You must attribute the work in the manner specified by the author or licensor.
-  **Noncommercial.** You may not use this work for commercial purposes.
-  **No Derivative Works.** You may not alter, transform, or build upon this work.

- ♦ For any reuse or distribution, you must make clear to others the license terms of this work.
- ♦ Any of these conditions can be waived if you get permission from the copyright holder.

Your fair use and other rights are in no way affected by the above.

For more details on the legal code (full license) and disclaimer, please refer to <http://creativecommons.org/licenses/by-nc-nd/2.5/deed.en>

Appendix C Content of Enclosed CD-ROM

As CD-recording technology is commonly adopted, a part of the content of the appendices is also recorded in CD-ROM format to be more environmental-friendly and reduce the weight of this dissertation. Moreover, extra functionalities can be provided in some of the content such as the copy of dissertation in PDF format and the hyper-linked presentation of the Delphi survey results in digital format. The content in the CD-ROM is listed below.

Content	Directory/File
PDF Version of Dissertation	/eval_fosphost.pdf
Delphi Survey Result	/Delphi/result/index.html
Capture of the Evaluation Model on the WakkaWikki	/wakka_pages_capture/index.html
Source Code of Delphi and Database Dump	/Delphi
Source Code of Evaluation and Database Dump	/eval_code

For convenience sake, introductory information for Delphi survey results, capture of the evaluation model on the WakkaWikki, source code of Delphi and Database dump and source code of Evaluation and Database dump are included in appendices G, H, I and J respectively.

Appendix D Jane Jacob's Systems of Survival

Jane Jacob suggested that there were two systems of morality that a society can become sustainable (Jacobs 1993). They were called systems of survival. One of them was named commercial moral syndrome and the other guardian moral syndrome.

Any society has to adopt morality such as cooperation, courage, moderation, mercy, common sense, foresight, judgment, competence, perseverance, faith, energy, patience, wisdom in order to survive. Nonetheless, there are also two systems of opposite morality that a society could adopt. These are the commercial and guardian moral syndromes and the details of each syndrome are listed below (Table D-1):

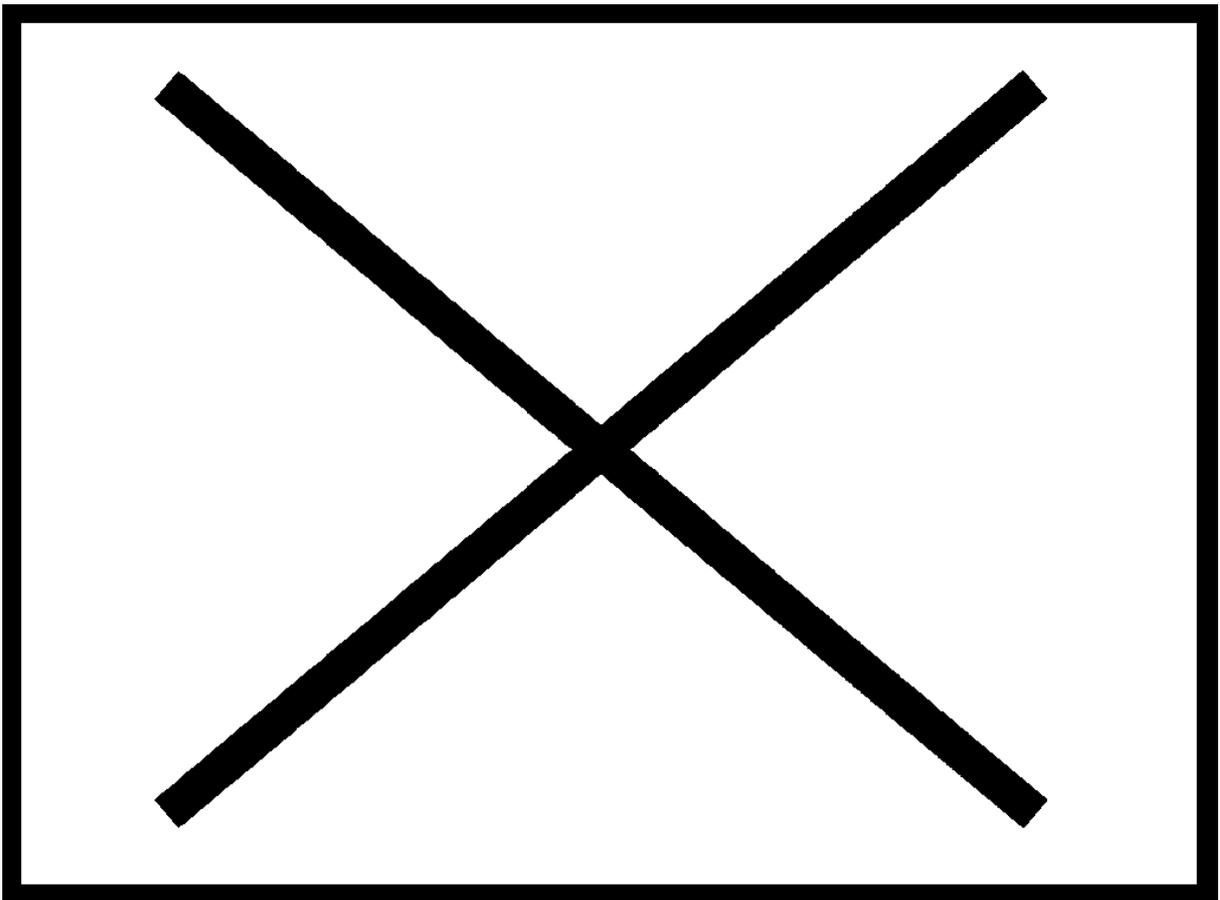


Table D-1 Commercial and Guardian Moral Syndromes (Jacobs 1993, pp. 23-4)

Two syndromes were proposed based on the observation that human could adopt animal like terrestrial behaviour in building society, which was represented by the guardian moral syndrome. In contrast, unlike animals, human could exchange possession and doing trading, which was represented by the commercial moral syndrome.

In the context of this research, opinion collected from participants in Delphi survey indicated that commercial moral syndrome was the favourable syndrome to be promoted in a FOSPHost site. This opinion may be a potential area of research.

Appendix E Susceptibility of Average and Variance

In the Delphi survey, both average and variance were employed to process the data. One may ask the question as the number of participants in a Delphi survey is not too big, if there is one more participants giving responses, how much change will there be in the result. Moreover, will the change in variance be larger than the change in average? The conclusion can be deduced algebraically from basic definitions of average and variance.

$$\begin{aligned}
 \text{new average} &= \frac{x_1 + x_1 + x_1 + \cdots + x_n + x_{n+1}}{n+1} \\
 &= \frac{n\bar{x} + x_{n+1}}{n+1} \\
 \text{new average} - \text{old average} &= \frac{n\bar{x} + x_{n+1}}{n+1} - \bar{x} \\
 &= \frac{x_{n+1} - \bar{x}}{n+1} \tag{Eq. E-1} \\
 \text{new variance} &= \frac{x_1^2 + x_2^2 + x_3^2 + \cdots + x_n^2 + x_{n+1}^2}{n+1} - (\text{new average})^2
 \end{aligned}$$

$$\begin{aligned}
 &= \frac{n(\overline{x^2}) + x_{n+1}^2}{n+1} - \left(\frac{n\bar{x} + x_{n+1}}{n+1} \right)^2 \\
 \text{new variance} - \text{old variance} &= \frac{n(\overline{x^2}) + x_{n+1}^2}{n+1} - \left(\frac{n\bar{x} + x_{n+1}}{n+1} \right)^2 - \left(\overline{x^2} - \bar{x}^2 \right) \\
 &= \frac{x_{n+1}^2 - \overline{x^2}}{n+1} \\
 &\quad - \frac{n^2\bar{x}^2 + 2n\bar{x}x_{n+1} + x_{n+1}^2 - (n+1)^2\bar{x}^2}{(n+1)^2} \\
 &= \frac{nx_{n+1}^2 - 2n\bar{x}x_{n+1} + (2n+1)\bar{x}^2 - (n+1)\overline{x^2}}{(n+1)^2}
 \end{aligned}$$

If we assume that n is big then $n \approx n+1$ and $2(n+1) \approx 2n+1$:

$$\begin{aligned} &\approx \frac{x_{n+1}^2 - 2\bar{x}x_{n+1} + 2\bar{x}^2 - \overline{(x^2)}}{(n+1)} \\ &= \frac{(x_{n+1} - \bar{x})^2 - (\overline{(x^2)} - \bar{x}^2)}{(n+1)} \end{aligned} \quad (\text{Eq. E-2})$$

Comparing the differences in average and variance, the term $x_{n+1} - \bar{x}$ exists in both differences. Nevertheless, in variance, it is squared. Therefore, this squared term in the difference in variance will always be greater than the term in the difference in average. On the other hand, we know that $\overline{(x^2)} - \bar{x}^2$ is the original variance and cannot be negative. Therefore, if the original variance is large and the term $x_{n+1} - \bar{x}$ is large, the effect may cancel out, but the term $x_{n+1} - \bar{x}$ is close to zero, the new variance will decrease in value. As we do not know the distribution of the response, further derivation is difficult. Therefore, there is no conclusion that difference in average is always smaller to the difference in variance or vice versa.

Appendix F Software Configuration System and Source Code Repository

From the Delphi survey and the Free Software Engineering Survey (Reis 2002a), source code repositories were the most essential tools of a FOSPHost site. One of the corresponding studies of source code repositories in software engineering is software configuration system (SCM). Both of the above topics are important, but as there should be only one central argument for a dissertation, information on these two topics are included as an appendix. Relevant literature will be reviewed on software configuration system and background, features and limitations of CVS will be presented below. Other related source code repositories would be introduced as well.

F.1 Historical Influences

At least two historical developments are relevant to the design of commonly used source code repositories, namely the development of the discipline of software configuration management and hacker culture.

The discipline of SCM was a branch of configuration management (CM), which could be traced back for about 50 years. This method was initially applied to management engineering drawings in hardware engineering processes (Buckley 1996). As more products included software as one of its components, the discipline of software configuration management was formed.

The formal definition of a configuration item is 'a collection of hardware, software, and/or firmware, which satisfies an end-use function and is designated for configuration management.' (Buckley 1996, p. 5) CM is the administration of these configurations. Four activities are usually identified in CM, namely identification, change control, status accounting and audits

(Ben-Menachem 1994; Buckley 1996). A simple definition of SCM is given by White (2000, p. 1) 'SCM is about managing change to software'. The four activities mentioned are applied in a similar fashion to software. It is obvious that the practice of SCM emphasizes control and processes.

Another radically different approach was hacker culture developed from universities such as Massachusetts Institute of Technology (The word 'hack' in this paper does not refer to breaking into computers. It refers to the ultimate standard of technical virtuosity and aesthetic in a Free/Open Source community). One emphasis in this culture is to minimising barriers in collaborative programming. Considering the ITS (Incompatible Time Sharing) System, a system regarded as the ultimate expression of hacker culture (Levy 1984), there was no password and anyone could change anything on the system. There was no procedure similar to the 'proposal, justification, evaluation, co-ordination, approval' in CM change control procedures but only the social pressure for quality programming (Levy 1984). As argued above, one of the important aspects of this design was to minimise barriers in collaborative programming by abolishing control and processes.

The hacker community in Massachusetts Institute of Technology did not last and after its dispersion. Richard Stallman started the GNU project to produce a Free Unix operating system. Since developers of the GNU no longer wrote code in the same room and social pressure could not be asserted physically and verbally. Tools such as *diff* and *patch* were employed to cope the need of distributed development (This was a depressing time for hackers, please refer to Raymond (1999) and following sections for discussion of the development of this culture). *diff* is a program that compares and presents the differences (and similarities) between two text files on a line-to-line basis (Figure F-1). *Patch* is a program that does the reverse of *diff* – take the original file and the output of *diff* to form the file with the changes described in the output of *diff*.

These two tools seem to be quite primitive but they have been in use as the primary tools for Linux kernel development for years (Bezroukov 2002).

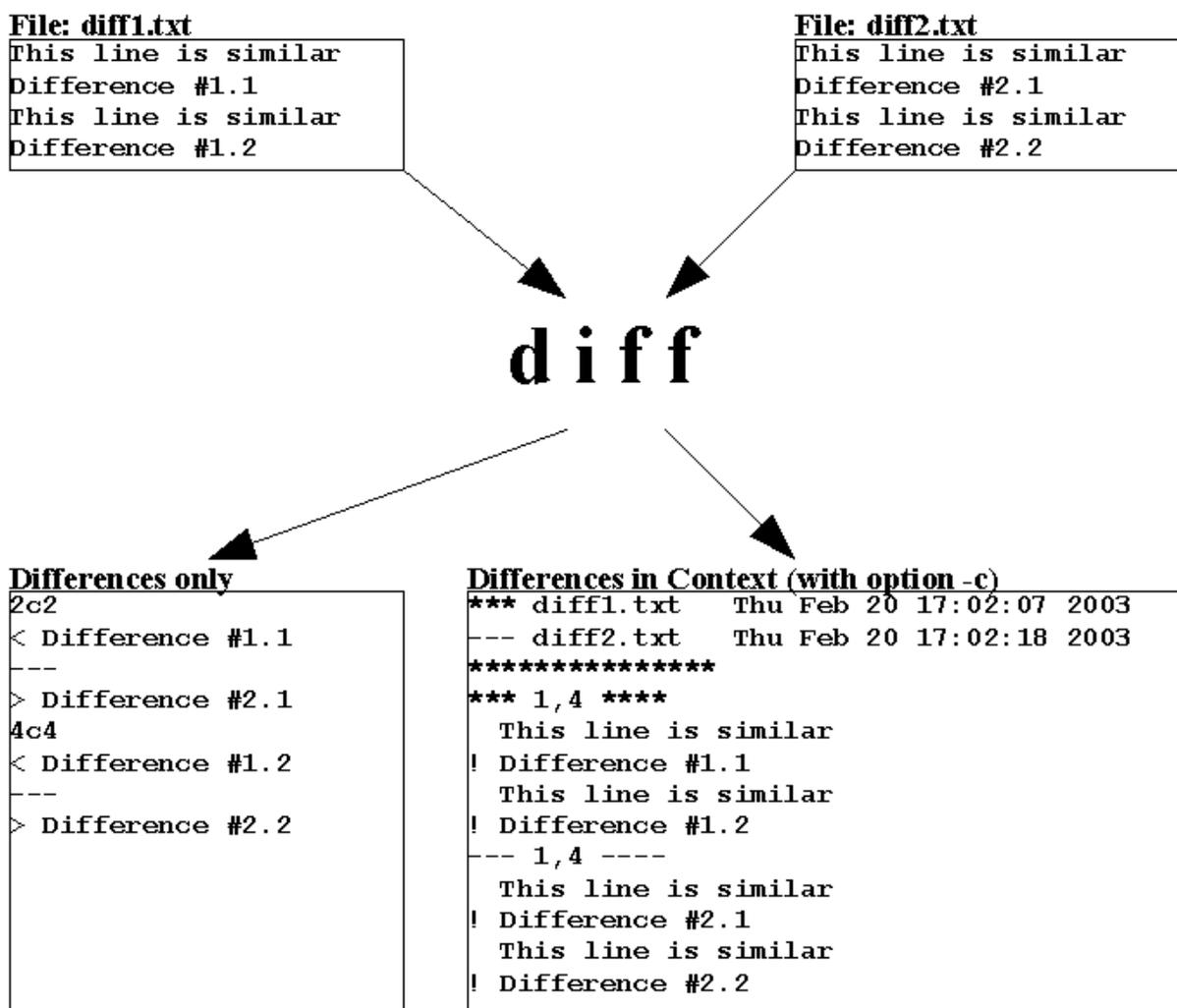


Figure F-1 Illustration of the operation of `diff`

In the early 1980s, Walter Tichy also released a program called *Revision Control System (RCS)* as Free software (Bolinger & Bronson 1995; Fogel 1999; Tichy 1985). This system has basic function of a versioning system - giving each newly updated file a version number and keeping track of the changes between versions in a central repository. Files can also be managed in a group, which was referred by Tichy as a configuration, by giving them a common attribute, such as a common version number or updated before a certain date. In order to preserve the integrity of the system, a pessimistic locking system (i.e. 'lock-modify-unlock') policy is

adopted. From Tichy's own explanation of the system (1985), he was aware of the drawback that other programmers might be deterred when a file was locked. The remedy for this was a branching version mechanism. This mechanism allow a locked file to be read and updated using another set of version number. For example, a branched file from version 1.2 with has version number 1.2.1.1, 1.2.1.2 and so on. This branch can then be merged back to the main trunk at the later time. The design of this system was influenced by SCM, as Tichy consciously put SCM functionalities in the system.

As *RCS* is free and portable (Fogel 1999), it became popular around the Free Software circle. Though the branching functionality was provided, locking was such a nuisance that a number of people wrote scripts around *RCS* and eventually rewrote the system in C. The new system was called the *Concurrent Versions System (CVS)*. The word 'Concurrent' means there is no locking and programmers can work on the same file at the same time. The mechanism of the system is illustrated in Figure F-2 (the diagram only shows one of the many possible scenarios).

Although *CVS* gets rid of the nuisance from locking, this implementation is less fool-proof than *RCS*. In the official manual of *CVS*, the Cederqvist manual (Pesch 2002), it is explicitly explained that if a person tries to get an update when another person is committing several files, some of the files will be the latest update while others will not. The design was chosen because the performance of *CVS* was not fast and locking while committing would deter collaboration (Fogel 1999).

One can argue that on one hand, *CVS* is a version control system that has an explicit procedure in controlling which it inherited from *RCS* and ultimately from the SCM discipline. However, the design of replacing locking by a less fool-proof system could be an influence from the hacker culture to increase participation. Therefore, this design can be regarded as the result of

the interactions between order and chaos. This mechanism is probably one of the fundamental bases in Free/Open Source software development co-ordination.

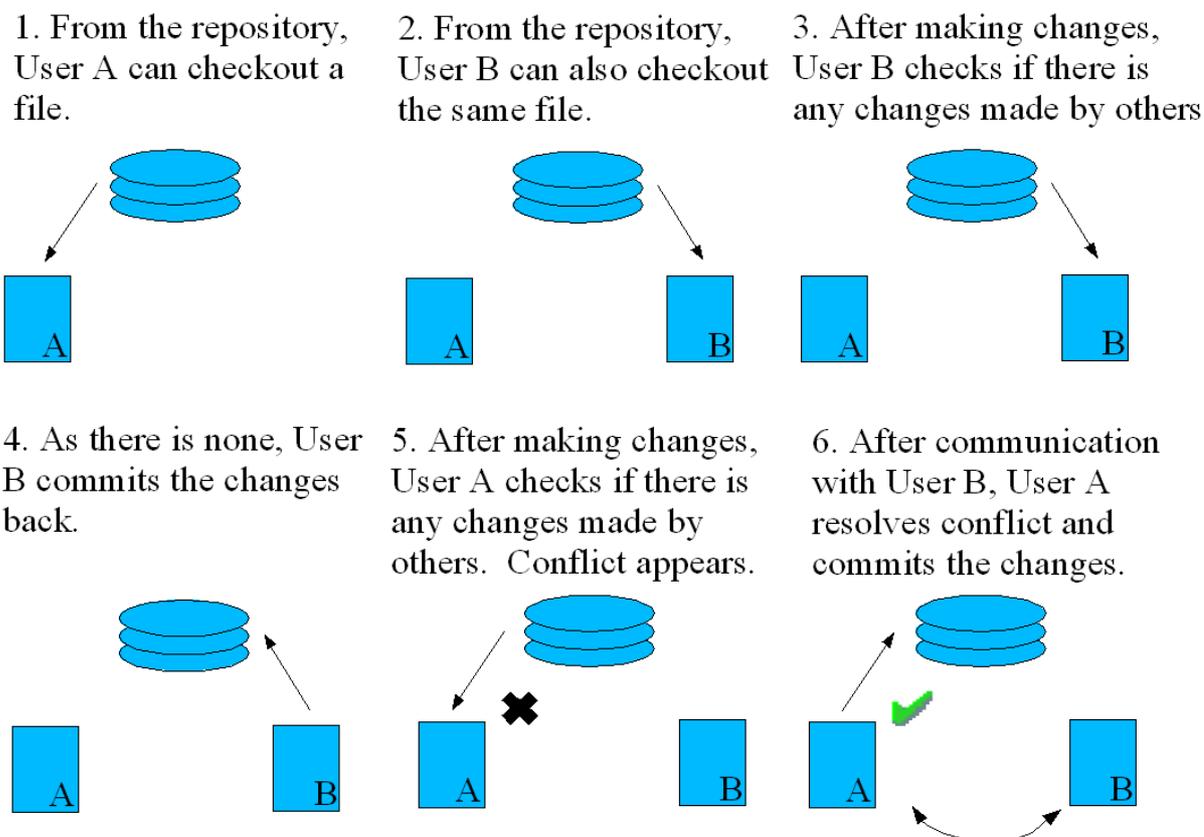


Figure F-2 Typical CVS Operation

F.2 A Closer Look at CVS

As CVS is the most common used source code repository in Free/Open Source communities, it will be profitable to understand some of its basic functionalities. The functionalities chosen to be discussed here are mainly based on the results from the Delphi Survey, where experts selected several important aspects of a source code repository, namely version control, client-server model, concurrency of development, tagging, security, branching, and accessibility through web. Therefore, the discussion in this sub-section should portray a fair picture of the system.

The basic operation model of *CVS* is the client-server central repository model. The client-server model is explained by the Free On-line Dictionary of Computing (1998) as

A common form of distributed system in which software is split between server tasks and client tasks. A client sends requests to a server, according to some protocol, asking for information or action, and the server responds.

This is analogous to a customer (client) who sends an order (request) on an order form to a supplier (server) who despatches the goods and an invoice (response). The order form and invoice are part of the "protocol" used to communicate in this case.

In the case of *CVS*, the server transmits the version of files that the clients request and accepts the files checked-in by the clients. It also keeps track of different versions of the files submitted. Moreover, the server also detects if there is any conflict during checkins (Figure F-2).

The Free On-line Dictionary of Computing (1998) also explained that the client-server model can be centralised or distributed. In the case of *CVS*, it is centralised. This means that there is only one server holding the complete information of the repository.

Another characteristic of *CVS* is the versioning of information. The smallest unit of information that *CVS* could handle is a single file. A typical series of version number given to a file is 1.1, 1.2, ... and the last number of the version is usually increased by one after each checkin (Pesch 2002). Therefore, if a project consists of different files, their individual version numbers will probably be different from one another as their frequencies of update will be different. Nevertheless, they can be assigned manually to a common number which is higher than all the existing numbers.

Although *CVS* handled files individually, it is common for users to manage several files in a project (though Figure F-2 is illustrated with single files to emphasize the fact that *CVS* is a per-file based system only). The mechanism to manage all the files within a project as a group (configuration) is essential. *CVS* facilitates this using the function called tag. A tag can be placed to a set of files and all the different version numbers of the files are recorded. The same set of files can then be retrieved later just using this tag, rather than individual version numbers of the files (Fogel 1999).

CVS also supports simultaneous development of the project in different branches. When a branch is created, files in the new branch will be given a new version number. For example, a file with version number 1.2 may have a new version number as 1.2.2.1 (Figure F-3). The next version of this file within the new branch will have the number 1.2.2.2 and so on. The progress of the branch and the main trunk will therefore have no effect on each other unless a merge is done. During merging, some files may be changed in both the branch and the main trunk, conflicts thus arise and manual resolutions are needed. As *CVS* is handling the version number of each file in the repository independently, the tree diagram for each file can be different. This is the reason why managing files in as a configuration is essential as it will be burdensome for a developer to manually keep track of these different version numbers.

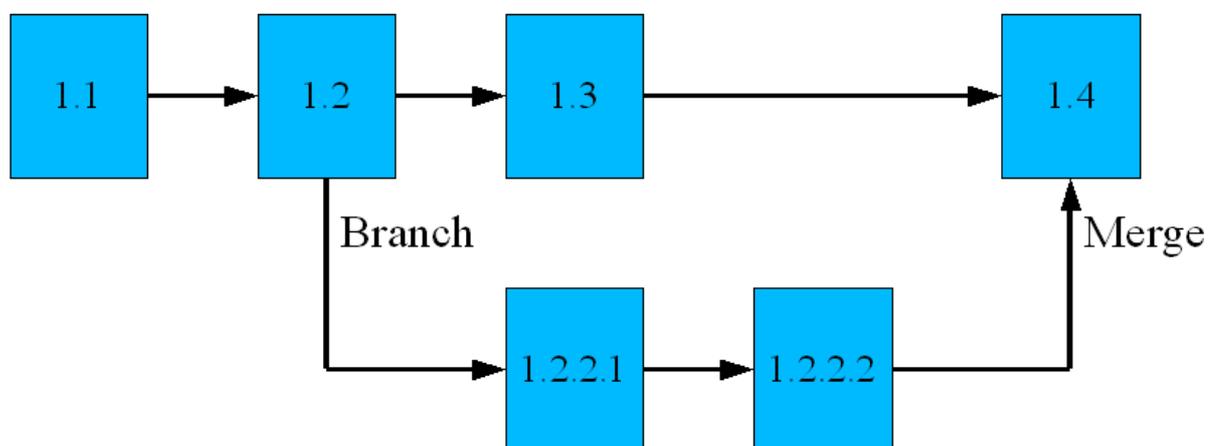


Figure F-3 The Tree Diagram for Branching and Merging for a Certain File

When accessing *CVS* from a client to the server through a network, it is possible to setup the server so that password authentication is required. Encryption system such as *rsh*, *kerberos* and *GSSAPI* can also be employed to enhance security (Pesch 2002).

From the results of the Delphi Survey, World Wide Web interface of *CVS* was found to be important. This feature allows easy access to the source code without switching to the *CVS* client when browsing the web page of a project. The most commonly used World Wide Web interface is *ViewCVS* (ViewCVS Users Group 2002). By employing *ViewCVS*, developers can access the repository through lists of directories and files with version number, author, change date and the most recent *ChangeLog* (text comment for every version checkin). It is also possible to explore the project at different tags and branches. For each file in the repository, one can view the content of the file in several ways. For source code, colorization of reserve words and other programming structures are available. Annotation of origin (from which author and which version) of each line of the file can also be generated. One can also compare the any two version of the file using *diff* on the web interface. Indeed, one may argue that the web interface may be easier to use than the command line client in exploring a project, thus encourage contribution. Another extension to *CVS* is *Bonsai* (The Mozilla Organization 2003a), a World Wide Web query interface to search content of a repository based on different criteria.

F.3 Limitations of CVS

CVS is more than 10 years old and Shapiro (2002) suggested it is a solution that effectively solve 80% of the needs of the Free/Open Source Communities and thus good enough to gain popularity. There are, however, a number of complaints on the limitations of this system, several of these complaints are discussed below. Firstly, van der Hoek suggested that (2000) developers usually checkin files that had substantial changes to the central repository. These changes may involve a few intermediate versions their own private workspaces, but as there is

only one repository, developers were left with no tools to manage these intermediate versions. Secondly, there are also needs for distributed and replication of repositories (Advogato 2000b; van der Hoek 2000). Thirdly, to rename files or directories in *CVS*, one has to first to delete the object and then add the object with its new name into the repository. The system thus will regard this object as new and versioning information will be lost (Advogato 2000b, 2002b; van der Hoek 2000). Fourthly, the branching and merging procedure in *CVS* is too tedious (Advogato 2000b, 2001c). Fifthly, as explained above, if one retrieves files from a repository while another person is checking-in, some of the files retrieved may still be the old version. Lastly, there is also suggestions to implement a stronger structure on configuration (Advogato 2000b) In *CVS*, unless a tag or branch command is issued, it is difficult for a developer to retrieve the whole project at a certain time. MacDonald, Hilfinger and Semenzato (1998) demonstrated by their Project Revision Control System (*PRCS*) that it is not necessary to give version number in a per-file basis. In *PRCS*, every checkin involves the whole project, not just individual files that are changed and the project as a whole bears a version number.

F.4 Other Systems Used in the Free/Open Source Communities

Other than *CVS*, there are other similar system used in the Free/Open Source communities, such as *Aegis* (Miller, P. 2003), *Arch* (Lord), *Bitkeeper* (BitMover 2002), *OpenCM* (Shapiro & Vanderburgh 2002b, 2002a; Shapiro, Vanderburgh & Lloyd 2003), *PRCS* (MacDonald, Hilfinger & Semenzato 1998) and *Subversion* (Collab.Net 2002b). Most of them include improvements from the shortcomings stated above. Even many of the systems above were structurally different from *CVS*, some of their designs were, however, influenced by it. For example, in the case of *OpenCM*, the command line operation of the client was designed to 'feel' like *CVS* (Shapiro & Vanderburgh 2002a).

F.5 Conclusion

In this appendix, the background of SCM and CVS was covered. The basic operations and limitations of CVS were also explained. Other source repositories were also briefly introduced. (References used in this appendix are included in the 'List of References' chapter in this dissertation.)

Appendix G Results of Free/Open Source Hosting (FOSPHost) Sites Delphi Survey

The data collected in the Delphi survey is presented in two formats. The first format is in HTML with hyperlinks at </Delphi/result/index.html> (An introduction of this format can be found in chapter 5). The other format is the raw data in PostgreSQL database dump format can be found at </Delphi/db.out>. Some of the secondary data such as log of IP addresses are deleted to keep the participants anonymous.

Appendix H WakkaWiki Pages

The qualitative part of the evaluation model was hosted using WakkaWiki on <http://www.ibiblio.org/fosphost/wakka/EvaluateFOSPHost>. It was captured into static html pages using a Free/Open Source program called WinHTTrack. Wiki sites are usually organised as 'http://hostname/wiki/keyword'. Nevertheless, WinHTTrack adds the extension '.htm' to the filename of the page capture so that the Windows operating systems can recognise the file as a html page. For example, the page 'keyword' becomes 'keyword.htm' after capture. Links in the captured pages to other captured pages are also adjusted automatically to this addition of extension.

The capture was done on 22 Jan 2004 17:28:29 +10:00. The complete capture can be found in the CD-ROM enclosed in the directory '/wakka_pages_capture'. To access the capture, please open the index.html and you will be re-directed to the correct page.

Appendix I Source Code for Delphi Survey

The readme file of the source code for the evaluation model is included in this appendix to give the reader more understanding of the code. The audience of the readme file is the developers/users of the code and basic programming knowledge is assumed. The style of the file is more informal and less academic in format. The complete source code can be found in the CD-ROM enclosed in the directory 'Delphi'.

readme starts here...

What to expect

This is a system to conduct Delphi survey online for a PhD project to collect opinion on Free/Open Source Project Hosting (FOSPHost) sites. This system was highly customised to the specific survey that was conducted. In order for it to be usable to other situations, you must modify the code. This implies that you need to know PHP PostgreSQL and prepare to read code. This is my first project in PHP (but I have experience in web applications before) and so there are definitely many areas for improvement. So much about the 'downside' of this system, I better 'sell' the benefits of using or at least referring to this system. Through this system, real data was successfully collected. In short, it worked. Therefore, there are a lot of details included in this system that can help you to imagine what it is like to run a Delphi survey online even if you hate reading and modifying code. This file contains a brief introduction of the system and more details about the code is included in the code_issues.txt file.

The source code is licensed under GPL and the content of the database is OPL. These license are included in the files gpl.txt and opl.txt. The current version is 0.10 written by Haggen So. This document is updated on 3 Feb 2004.

To install, you need:

1. A web server
2. PHP (≥ 4)
3. PostgreSQL 7.0 or above

How to install:

1. Create a database called 'delphi' and create the user 'haggen'. Pour the 'db.out' in.
2. Put the files in some directories visible from through the web server.
3. Done

As the files are distributed in the state of the end of the survey, the login page is cannot be accessed directly through 'index.html'. You can find it at 'old_register.html'. For different rounds, their corresponding directories 'round1', 'round2' and 'round3' all contain information that the survey was closed. The original directories with PHP scripts were moved to 'c_round1', 'c_round2' and 'c_round3' instead.

The process

A detail description of the process of the survey can be found in the PhD dissertation 'Construction of an Evaluation Model for Free/Open Source Project Hosting (FOSPHost) sites'

chapter 5. It can be found at <http://www.ibiblio.org/fosphost/> when it is ready. A brief outline will be provided below.

The survey consisted of 3 rounds: the first round was qualitative and second and third round both quantitative & qualitative. 12 broad questions were asked in round 1 with Q2 as multi-level question. The qualitative responses collected were then summarised into unit concepts and the participants were asked to verify these concepts. Round 2 of the survey commenced together with the presentation of the results for round 1. Participants were asked to rate different statements (from unit concepts) and gave comments as well. Round 3 was a repetition of round 2 with round 2 results supplied. A post Delphi survey were also done to collect opinion on the implementation of the survey.

Table Structures

An introduction of the database tables will be helpful for the understanding of the code. A brief outline of the main tables will be presented below.

The first issue to be explained was the anonymous participants design of the system. When the participants were invited, they were given a UserID and a password. On the web server, the UserID was the sole identification for the participants. The name and the email address of the participants were not stored on the server. Therefore even in the case of the server being compromised, the participants could still remain anonymous. The data collected thus could also be easily made available to all without ethics concerns by just deleting sensitive data such as IP addresses. The structure table that held the information of the participants is listed below.

```
create table userinfo (  
id          serial primary key,
```

```
userid          varchar (10) unique,  
nickname        varchar (200),  
password        varchar (100),  
expertuser     int2,  
administrator  int2,  
academic       int2,  
pleader        int2,  
nonpub         int2);
```

For most tables in the database, primary keys, id, are inserted for identification purpose. For userid, the ones that started with r (i.e. r4404) were assigned to actual participants of the survey. The ones that started with p were assigned to pilot participants. Other odd ones such as s000-s100, 123, 000, eugene and jasmine were created for testing purposes. As it was difficult to refer to a person using an ID number, a unique nickname was given. A list of nicknames was stored in the nickname table and a new participant could choose an unassigned name from the list.

Whenever a participant tried to login, successfully or not, his or her action would be stored in the log table:

```
create table log (  
id          serial primary key,  
userid      varchar (10),  
nickname    varchar (200),  
password    varchar (100),  
time        timestamp default current_timestamp,  
login       int2,
```

```
ip          cidr);
```

After an introduction to the participants' related tables, the tables for each round will be explained. Before going into the details of the table structures, the concept of qnid and qnno have to be explained. qnno was usually the question number displayed, such as 1.2 or 2.1.2.1. Nevertheless, when this number was stored, it was in the form of 001002 or 002001002001 and called qnid.

For round 1, the most important table was answer1, which stored the responses from the participants.

```
create table answer1 (  
id          serial,  
userid     varchar (10),  
ip         cidr,  
time       timestamp default current_timestamp,  
qnid       varchar (20),  
answer     text,  
supans     varchar (255),  
nj         int2,  
deleted    timestamp);
```

The field answer held the qualitative responses and supans held the quantitative responses. nj stood for 'no judgment', and this field would be set to 1 when the 'No Comment' checkbox was selected.

When a participant left a page that collected responses, his or her responses would be saved. The action of saving involved assigning the previously saved responses as deleted (putting a timestamp on the field deleted) and appending the new responses to the table. This action could give the researcher extra information on the changes of responses and also the participant's movements across pages.

After the responses were collected, they were summarised into unit concepts. The unit concepts were stored in table.

```
create table codebook1 (  
id            int4    unique,  
qnid         varchar (20),  
sid          int2,  
summary      varchar (255));
```

The field sid was a unique number given to each concept within a question. The number increased from 1 onwards. For convenience, when displaying the unit concepts, the descriptions of the questions were stored in the records with sid=0. By when the table was retrieved ordered by qnid and sid, the questions and concepts would both be retrieved in one table.

The relationship between the answers given and the unit concepts summarised were many-to-many. This was because many participants had expressed similar ideas, and within one answer more than one unit concept could be expressed. This relationship was stored in table coding1.

```
create table coding1 (  

```

```
id            int4    unique ,
ansid        int4 ,
codeid       int4 );
```

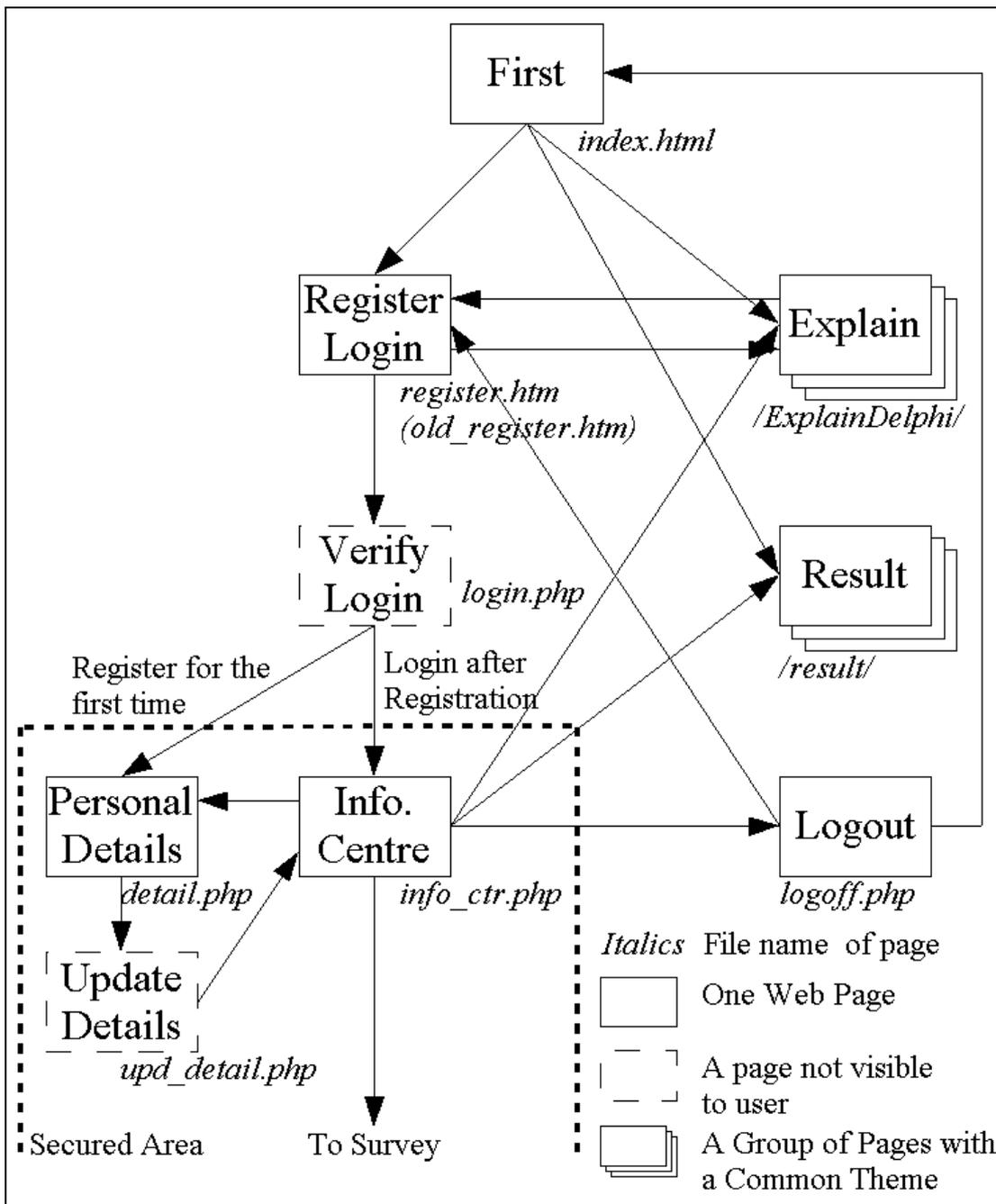
For round 2 and round 3, the ratings of statements (unit concepts) related to the questions were collected. The structures of the tables that held the responses were the similar to round 1 answer1. One of the differences was randomisation of statements. In order to collect a more unbiased data, the order of the statements presented for ratings were randomised when displayed. The statements were randomised into a different order for each participant and they were stored in the table qn2seq. By retrieving this table ordered by id, the randomised order could be obtained.

```
create table qn2seq (
id            int4    unique ,
userid       varchar (10) ,
qnid         varchar (20) ,
qnum        varchar (20) ,
descn       text ,
leaf        int2 );
```

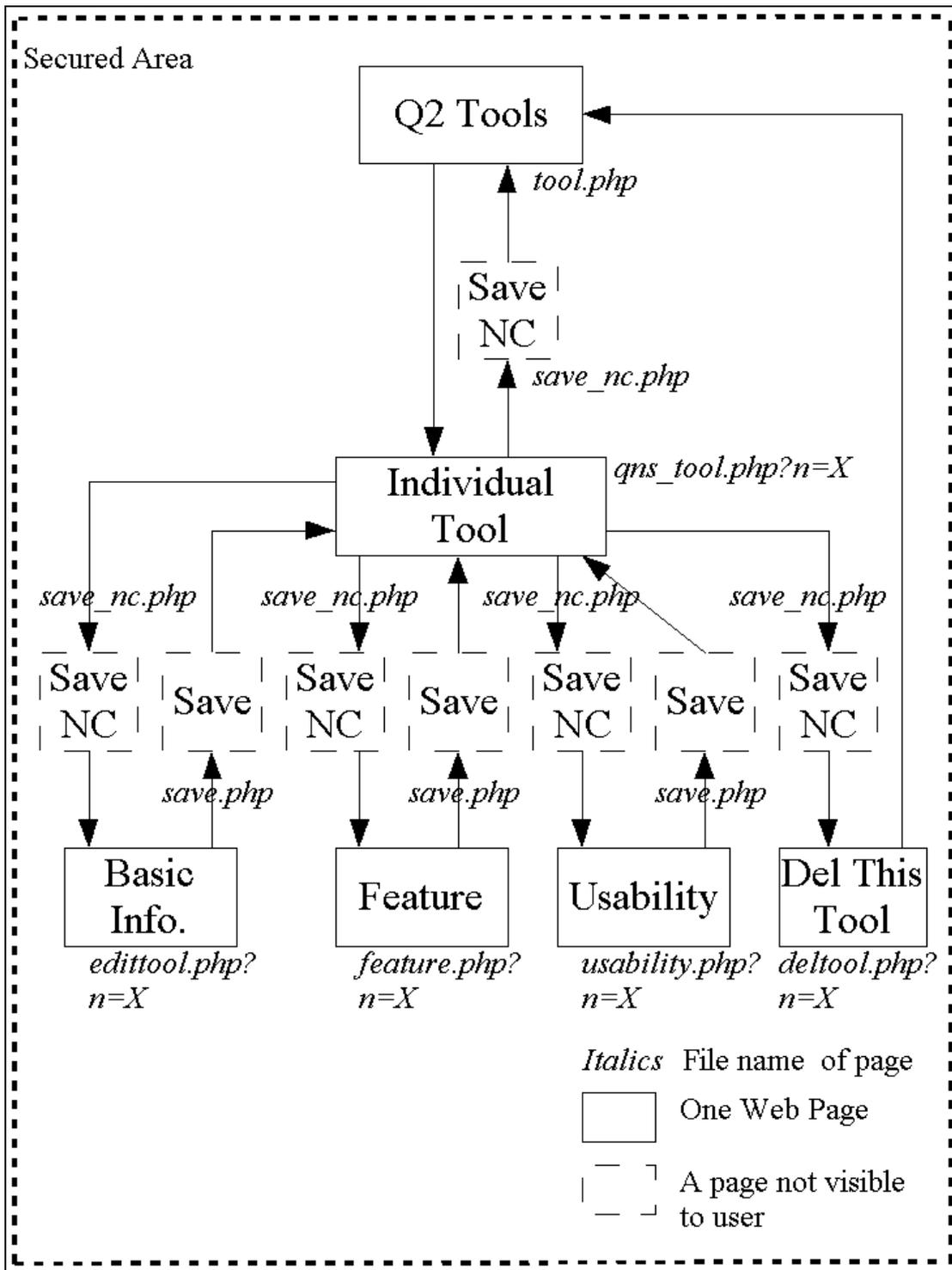
Site map and related filenames

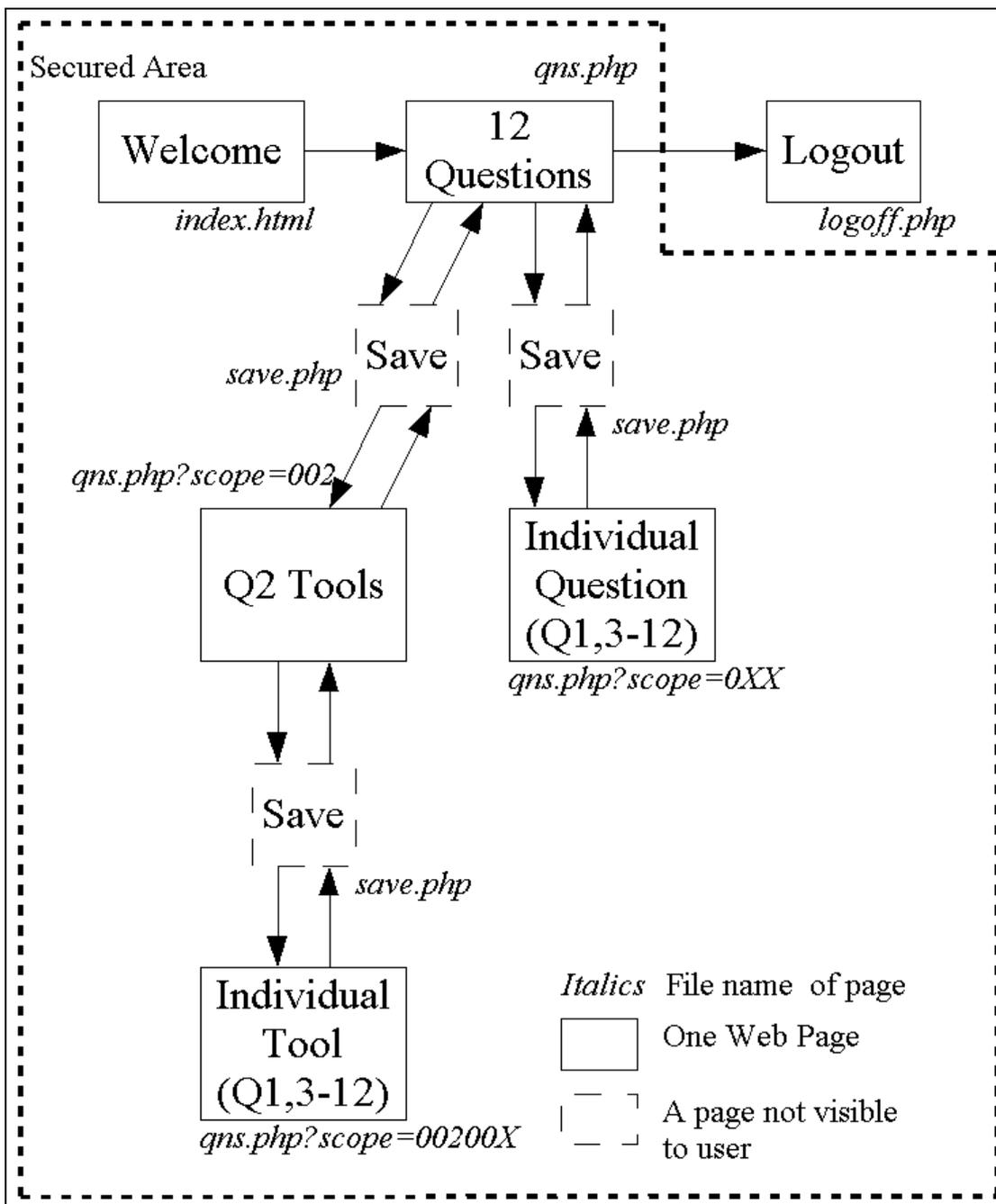
To make the process and the related programs more understandable, site maps with related filenames are provided below. Some minor files were not included.

Initial Login Area



Round 2 & 3





Results Generation

The majority of the generation of the presentation of the result was automated. The mechanism was to ask the PHP interpreter to output to a file rather than to the web server. The code for generation are located at /result/gen1, /result/gen2 and result/gen3 and the main program for generation was createhtm.php.

This mechanism was also used to generate pages for the verification process in round 1. The generation program was `/result/gen1/gencheck.php`. The generated page could then be viewed through `/check1/check.php`. One can also customize the generated pages, e.g. `'/check1/r9214.htm'`.

Appendix J Source Code for Evaluation Model

The readme file of the source code for the evaluation model is included in this appendix to give the reader more understanding of the code. The audience of the readme file is the developers/users of the code and basic programming knowledge is assumed. The style of the file is more informal and less academic in format. The complete source code can be found in the CD-ROM enclosed in the directory '/eval_code'.

readme starts here...

A demo of the software can be found at

http://www.ibiblio.org/fosphost/exhost_choose.php

http://www.ibiblio.org/fosphost/exhost_evalset.php

You can work out which file does what from the demo.

The source code is licensed under GPL and the content of the database is public domain (this license can be found in the file 'gpl.txt'). The current version is 0.10 written by Haggen So.

This document is updated on 3 Feb 2004.

To install, you need:

1. A web server
2. PHP (>=4)
3. MySQL 3 or above

How to install:

1. Create a database called 'fosp' and pour the 'fosp_dbdump' in.
2. Update the function connectdb in lib/lib.php with the correct connection parameters.
3. Put the files in some directories visible from through the web server.
4. Done

To customize:

Let's first study 'exhost_choose.php'

```
<?php
require('lib/lib.php');
pagehead();
connectdb($conn);
require('exhost_choose_head.php');
$cat='exhost';
require('lib/choose.php');
require('exhost_choose_tail.php');
?>
```

Short and simple, right. The first obvious thing is that there are many 'exhost*'. This is designed so that more than one set of evaluation can be hosting in the same directory and using the same database. That is, all files and tables related to external hosting are named 'exhost'. Also, before including 'lib/choose.php', which is the general routine corresponding to

'exhost_choose.php', the variable \$cat is set to 'exhost' so that the routine 'knows' to load data from the 'exhost_*' tables.

So if you want to create another evaluation, for example, on operating systems, you can copy all the 'exhost' files and tables with the prefix 'OS' and change all the \$cat assignments into 'OS'. Of course you need to fill the tables with new data ;)

If you want to change the content of a certain page, notice that usually the generated content is in the middle of a HTML page and the header and footer are more or less static. Therefore they are included as separate header and footer files. Any modification of the header and footer should go into those files, not the 'program files' in /lib. Don't update the 'program files' for content change unless you really need to.

To update a cell:

1. Generate a table that contains the cell
2. Add '&edit=1' to the URL and reload the page
3. An extra link will be added to every cell and click the link of the specific cell
4. Update the information
5. Copy the SQL statements generated into MySQL monitor

Usually, two SQL statements will be generated and one of them is to copy the old information into a table called 'exhost_ver_*'. These tables have similar structures to 'exhost_*' with an extra date field, 'deleted'. The function of these tables is to record all the changes.

An introduction to the code

The most important product of the program is the comparison table, and it can be generalised as below:

		Item 1	Item 2
Feature Group1	Feature 1	Cell	Cell
	Feature 2	Cell	Cell

All the content in the table is stored in database and then loaded into different objects in runtime.

The HTML page of the comparison table is then generated from the content in the objects. So let's start from the tables in the database.

From the table above, there are four types of content, namely feature groups, features, items and cells. So, there are four corresponding tables in the database. For each table, there is a primary key called 'id' (except for feature_grps). There are also other keys to relate the tables, namely 'rid' (row), 'cid' (column), 'grpid' (feature group). Two extra keys 'rsortid' and 'csortid' are included to control the sort order when displayed. The table definitions are listed below:

```
CREATE TABLE exhost_feature_grps (
    grpid bigint(20) NOT NULL auto_increment,
    eval tinyint(4) default NULL,
    name varchar(255) binary default NULL,
    PRIMARY KEY (grpid)
) TYPE=MyISAM;
```

```
CREATE TABLE exhost_features (
    rid bigint(20) NOT NULL auto_increment,
    rsortid int(11) NOT NULL default '0',
    grpid bigint(20) NOT NULL default '0',
    name varchar(255) binary default NULL,
```

```
PRIMARY KEY (rid),

UNIQUE KEY rsortid (rsortid),

KEY rsortid_2 (rsortid),

KEY grpid (grpid)

) TYPE=MyISAM;

CREATE TABLE exhost_items (

cid bigint(20) NOT NULL auto_increment,

csortid int(11) NOT NULL default '0',

name varchar(255) binary default NULL,

PRIMARY KEY (cid),

UNIQUE KEY csortid (csortid),

KEY csortid_2 (csortid)

) TYPE=MyISAM;

CREATE TABLE exhost_data (

id bigint(20) NOT NULL auto_increment,

rid bigint(20) NOT NULL default '0',

cid bigint(20) NOT NULL default '0',

content varchar(255) binary default NULL,

PRIMARY KEY (id),

KEY rid (rid),

KEY cid (cid)

) TYPE=MyISAM;
```

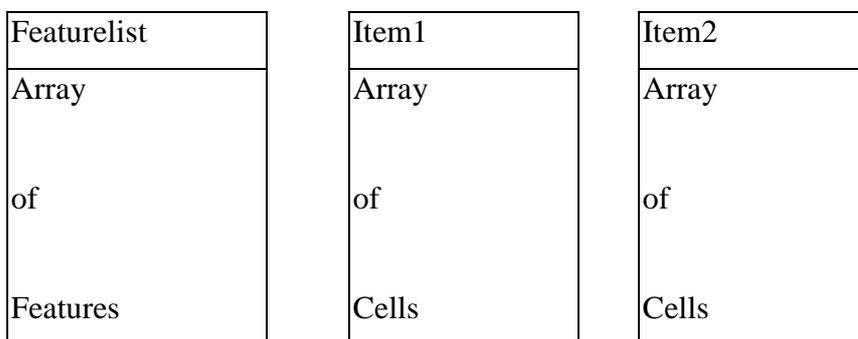
After the introduction of the structure of the tables, the data structure in the program will be explained. A simple illustration of the major objects can be shown like this:

```

itemlist
+-featurelist
| +-array of features
+-array of items
    +-array of cells
    
```

itemlist contains two major elements, featurelist and array of items. For each item, it contains the description to itself and also all the cells related to that item.

Another way to show them is by mapping them to the table:



After explaining the data structure, it is time to outline the major algorithm in loading the data from the tables to the objects:

Load featurelist according to condition

Load Itemlist according to condition

For each item in itemlist

Load content into array of cells according to featurelist

This algorithm can be seen from `lib/evalres.php` (Another implementation can be found in `lib/gentable.php`, which is actually an earlier version of this algorithm. `gentable.php` will definitely look nicer by calling the methods defined in `obj.php`, but it is not broken, so nothing is done).